

Washington University in St. Louis

## Washington University Open Scholarship

---

All Computer Science and Engineering  
Research

Computer Science and Engineering

---

Report Number: WUCS-2007-7

2007-05-01

### Comparing Features of Three-Dimensional Object Models Using Registration Based on Surface Curvature Signatures

Timothy David Gatzke and Cindy M. Grimm

This dissertation presents a technique for comparing local shape properties for similar three-dimensional objects represented by meshes. Our novel shape representation, the curvature map, describes shape as a function of surface curvature in the region around a point. A multi-pass approach is applied to the curvature map to detect features at different scales. The feature detection step does not require user input or parameter tuning. We use features ordered by strength, the similarity of pairs of features, and pruning based on geometric consistency to efficiently determine key corresponding locations on the objects. For genus zero objects, the corresponding locations... [Read complete abstract on page 2.](#)

Follow this and additional works at: [https://openscholarship.wustl.edu/cse\\_research](https://openscholarship.wustl.edu/cse_research)



Part of the [Computer Engineering Commons](#), and the [Computer Sciences Commons](#)

---

#### Recommended Citation

Gatzke, Timothy David and Grimm, Cindy M., "Comparing Features of Three-Dimensional Object Models Using Registration Based on Surface Curvature Signatures" Report Number: WUCS-2007-7 (2007). *All Computer Science and Engineering Research*.  
[https://openscholarship.wustl.edu/cse\\_research/921](https://openscholarship.wustl.edu/cse_research/921)

Department of Computer Science & Engineering - Washington University in St. Louis  
Campus Box 1045 - St. Louis, MO - 63130 - ph: (314) 935-6160.

## Comparing Features of Three-Dimensional Object Models Using Registration Based on Surface Curvature Signatures

Timothy David Gatzke and Cindy M. Grimm

### Complete Abstract:

This dissertation presents a technique for comparing local shape properties for similar three-dimensional objects represented by meshes. Our novel shape representation, the curvature map, describes shape as a function of surface curvature in the region around a point. A multi-pass approach is applied to the curvature map to detect features at different scales. The feature detection step does not require user input or parameter tuning. We use features ordered by strength, the similarity of pairs of features, and pruning based on geometric consistency to efficiently determine key corresponding locations on the objects. For genus zero objects, the corresponding locations are used to generate a consistent spherical parameterization that defines the point-to-point correspondence used for the final shape comparison.

2007-7

## Comparing Features of Three-Dimensional Object Models Using Registration Based On Surface Curvature Signatures, Doctoral Dissertation, May 2007

Authors: Timothy D. Gatzke

Corresponding Author: [timothy.d.gatzke@boeing.com](mailto:timothy.d.gatzke@boeing.com)

**Abstract:** This dissertation presents a technique for comparing local shape properties for similar three-dimensional objects represented by meshes. We develop a shape representation that is sensitive to subtle shape differences but relatively insensitive to noise, and use this representation to detect features of the objects. The features are used, along with a measure for shape similarity, to compute a correspondence between the objects, which then allows shape comparison based on the shape properties at corresponding points. An advantage of this approach is that the final comparisons depend on the similarity-based correspondence and not on a physical three dimensional alignment.

Our novel shape representation, the curvature map, describes shape as a function of surface curvature in the region around a point. A multi-pass approach is applied to the curvature map to detect features at different scales. The feature detection step does not require user input or parameter tuning. We use features ordered by strength, the similarity of pairs of features, and pruning based on geometric consistency to efficiently determine key corresponding locations on the objects. For genus zero objects, the corresponding locations are used to generate a consistent spherical parameterization that defines the point-to-point correspondence used for the

Type of Report: Other

WASHINGTON UNIVERSITY

Sever Institute  
School of Engineering and Applied Science  
Department of Computer Science and Engineering

Dissertation Examination Committee:

Cindy Grimm, Chair  
Burchan Bayazit  
Roger Chamberlain  
Tao Ju  
Erik Trinkaus  
Victor Wickerhauser

COMPARING FEATURES OF  
THREE-DIMENSIONAL OBJECT MODELS USING  
REGISTRATION BASED ON  
SURFACE CURVATURE SIGNATURES

by

Timothy David Gatzke

A dissertation presented to the  
Graduate School of Arts and Sciences  
of Washington University in  
partial fulfillment of the  
requirements for the degree  
of Doctor of Philosophy

May 2007

Saint Louis, Missouri

WASHINGTON UNIVERSITY

Sever Institute  
School of Engineering and Applied Science  
Department of Computer Science and Engineering

ABSTRACT

COMPARING FEATURES OF  
THREE-DIMENSIONAL OBJECT MODELS USING  
REGISTRATION BASED ON  
SURFACE CURVATURE SIGNATURES

by

Timothy David Gatzke

Advisor: Professor Cindy M. Grimm

May 2007

Saint Louis, Missouri

This dissertation presents a technique for comparing local shape properties for similar three-dimensional objects represented by meshes. We develop a shape representation that is sensitive to subtle shape differences but relatively insensitive to noise, and use this representation to detect features of the objects. The features are used, along with a measure for shape similarity, to compute a correspondence between the objects, which then allows shape comparison based on the shape properties at corresponding points. An advantage of this approach is that the final comparisons depend on the similarity-based correspondence and not on a physical three-dimensional alignment.

Our novel shape representation, the curvature map, describes shape as a function of surface curvature in the region around a point. A multi-pass approach is applied to the curvature map to detect features at different scales. The feature detection step does not require user input or parameter tuning. We use features ordered by strength, the similarity of pairs of features, and pruning based on geometric consistency to efficiently determine key corresponding locations on the objects. For genus zero objects, the corresponding locations are used to generate a consistent spherical parameterization that defines the point-to-point correspondence used for the final shape comparison.

To my family

# Contents

List of Tables . . . . .	vii
List of Figures . . . . .	viii
Acknowledgments . . . . .	xviii
Glossary . . . . .	xix
<b>1 Introduction . . . . .</b>	<b>1</b>
1.1 Detailed Shape Comparison Goal . . . . .	4
1.2 A New Curvature-Based Shape Matching Technique . . . . .	6
1.3 Organization of the Dissertation . . . . .	7
<b>2 Background: Reasoning About Object Shape . . . . .</b>	<b>8</b>
2.1 Shape . . . . .	9
2.1.1 Object Representation . . . . .	9
2.1.2 Shape Decomposition . . . . .	12
2.1.3 Shape Similarity . . . . .	12
2.1.4 Medical Imaging . . . . .	14
2.2 Shape Representations . . . . .	15
2.2.1 Global Descriptors . . . . .	18
2.2.2 Local Descriptors . . . . .	21
2.2.3 Graph Representations . . . . .	22
2.2.4 Segmentation . . . . .	24
2.2.5 Features . . . . .	27
2.3 Shape Comparison . . . . .	29
2.3.1 Point-based Comparison . . . . .	33
2.3.2 Models and Morphing . . . . .	35
2.3.3 Graph Matching . . . . .	38

2.3.4	Energy Functions . . . . .	41
2.3.5	Template Matching . . . . .	41
2.3.6	Parameterization . . . . .	42
2.4	Chapter Summary . . . . .	48
<b>3</b>	<b>Curvature Calculation on Meshes . . . . .</b>	<b>49</b>
3.1	Overview . . . . .	49
3.1.1	Definitions . . . . .	52
3.1.2	Related Work . . . . .	53
3.2	Curvature Estimation . . . . .	55
3.2.1	Fitting Methods . . . . .	55
3.2.2	Discrete Methods . . . . .	60
3.2.3	Estimating the Curvature Tensor . . . . .	63
3.3	Evaluations . . . . .	65
3.3.1	Curvature Estimation Test Cases . . . . .	66
3.3.2	Experimental Results . . . . .	70
3.3.3	Discussion of Results . . . . .	76
3.3.4	Extension to General Surfaces . . . . .	81
3.4	Chapter Summary . . . . .	82
<b>4</b>	<b>Curvature Map Similarity Measure . . . . .</b>	<b>83</b>
4.1	Defining Local Surface Shape . . . . .	85
4.1.1	Defining Rings of a Mesh . . . . .	85
4.1.2	Geodesic Fans . . . . .	86
4.1.3	Estimating Curvature . . . . .	89
4.1.4	1-D Curvature Maps . . . . .	89
4.1.5	2-D Curvature Maps . . . . .	93
4.2	Shape Similarity Evaluations . . . . .	93
4.2.1	Comparing 0-,1-, and 2-D Curvature Maps . . . . .	94
4.2.2	Applying Curvature Maps to Other Objects . . . . .	100
4.2.3	Efficiency Comparison . . . . .	100
4.2.4	Finding Unique Features with Curvature Maps . . . . .	101
4.2.5	Guidelines for Computing Similarity . . . . .	102
4.3	Chapter Summary . . . . .	104



<b>5</b>	<b>Feature Detection Using Multi-Scale Graph Cuts</b>	<b>105</b>
5.1	Feature Detection Process	106
5.2	Local Shape Property	107
5.3	Multi-Scale Feature Detection	108
5.3.1	Graph Cut Parameters	110
5.3.2	Multi-Scale Parameters	111
5.3.3	Group Merging Criteria	113
5.4	Results	114
5.5	Chapter Summary	115
<b>6</b>	<b>Shape Matching</b>	<b>119</b>
6.1	Shape Comparison Goals	120
6.1.1	Technical Challenges	121
6.2	Determining Object Correspondence	121
6.2.1	Feature Correspondence	122
6.2.2	Surface Parameterization	123
6.2.3	Beyond Rigid Body Transformations	126
6.3	Implementing a Feature-Based Matching Process	126
6.3.1	Curvature, Curvature Map, and Feature Generation	127
6.3.2	Computing the Best Feature Pairs	127
6.3.3	Refining the Feature-to-Feature Fit	131
6.3.4	Physical Correspondence and Area of Overlap	132
6.3.5	Generating Additional Feature Pin Point Pairs	133
6.3.6	Adding Non-Feature-Based Pin Point Pairs	134
6.3.7	Similarity-Based Adjustment of Point Pair Locations	134
6.3.8	Correspondence and Parameterization	136
6.4	Bone Shape Comparisons	139
6.4.1	Lunate Bone Data Set	139
6.4.2	Ulna Bone Data Set	141
6.4.3	Radius Bone Data Set	146
6.4.4	Intermediate Cuneiform Bone Data Set	150
6.5	Face Scan Comparisons	152
6.6	Discussion	152
6.6.1	Timing	153

6.6.2	Scaling . . . . .	155
6.7	Chapter Summary . . . . .	157
<b>7</b>	<b>Quantifying Shape Differences . . . . .</b>	<b>159</b>
7.1	Shape Comparison Goals . . . . .	159
7.2	Analysis and Visualization Approach . . . . .	160
7.3	Shape Difference Measures . . . . .	160
7.3.1	Distances Between Corresponding Points . . . . .	161
7.3.2	Change in Surface Normal . . . . .	163
7.3.3	Similarity-based Shape Comparison . . . . .	165
7.4	Chapter Summary . . . . .	168
<b>8</b>	<b>Conclusions . . . . .</b>	<b>169</b>
8.1	Research Summary . . . . .	169
8.2	Ideas for Future Work . . . . .	171
	<b>References . . . . .</b>	<b>173</b>
	<b>Vita . . . . .</b>	<b>189</b>

# List of Tables

2.1	Shape Representation - Descriptors . . . . .	16
2.2	Shape Representation - Graphs, Segmentation, and Features . . . . .	17
2.3	Shape Comparisons - Points and Models . . . . .	31
3.1	Curvature Calculation Taxonomy - Fitting Methods . . . . .	56
3.2	Curvature Calculation Taxonomy - Discrete Operators . . . . .	61
3.3	Curvature Calculation Taxonomy - Curvature Tensor Estimation . . . . .	64
4.1	Preprocessing Times (milliseconds per Vertex) for 1.7 GHz Pentium M Processor . . . . .	101
4.2	Comparison Times (microseconds per Vertex) for 1.7 GHz Pentium M Processor . . . . .	101
5.1	Graph Cut Weights . . . . .	111
6.1	Shape Matching Execution Times . . . . .	154

# List of Figures

1.1	Bone structure in a knee joint. From “Accurate Measurement of Three-dimensional Natural Knee Kinematics Using Single-Plane Fluoroscopy” by Rahman et al., 2003 Summer Bioengineering Conference. . . . .	2
1.2	Machined parts. From Stamping Press - Preferred Tool and Die, Inc., URL= <a href="http://www.preferredtool.com/oemmarkets.htm">http://www.preferredtool.com/oemmarkets.htm</a> . . . . .	2
1.3	Two similar bones that are candidates for detailed shape comparison.	5
2.1	A mesh representation of the surface of a bone. (a) 3-D surface mesh. (b) Surface mesh flattened to the plane. Note the distortion of the triangle areas in the flattened mesh compared to the 3-D surface mesh.	10
2.2	Shape distributions for four object classes. Adapted from “Shape Distributions”, Osada et al., ACM Transactions on Graphics, 21(4), 2002.	20
2.3	Best matches returned from a database for select query objects. Adapted from “Shape Distributions”, Osada et al., ACM Transactions on Graphics, 21(4), 2002. . . . .	21
2.4	Geodesic distance function for a frog in two positions. From “Topology Matching for Fully Automatic Similarity Estimation of 3D shapes”, Hilaga et al., SIGGRAPH 2001. . . . .	23
2.5	Correspondence between two shock graphs. From “Shock Graphs and Shape Matching”, Siddiqi et al., ICCV 1998. . . . .	38
2.6	Brain surface segmentation. From “Segmentation, Modeling, and Registration - Introduction to Computer-Integrated Surgery” by Russell Taylor. . . . .	39
2.7	Shock graphs for two fish. From “Shape matching using edit-distance: an implementation”, Klein et al., Symposium on Discrete Algorithms, 2001. . . . .	40
2.8	Texture mapped bone surface. (a) Surface. (b) Checkerboard texture map applied to the surface. . . . .	43

3.1	Definition of curvature in two-dimensions. . . . .	51
3.2	Definition of curvature in three-dimensions. . . . .	51
3.3	Sample test case meshes. Left: 1-ring neighborhood (valence=6), Middle: 2-ring neighborhood (valence=5), Right: 3-ring neighborhood (valence=4). . . . .	53
3.4	Test case geometric shapes. . . . .	67
3.5	Detailed behavior test cases for a mesh projected onto a sphere. Left: Noise component normal to the surface at the target vertex ( $dR_T$ ) and at an adjacent vertex ( $dR_A$ ). Center: Mesh regularity component based on moving the target vertex away from center( $d\phi_T$ ). Right: Mesh regularity components based on moving an adjacent vertex toward/away from the target ( $d\phi_A$ ) or moving the adjacent vertex along the ring toward a neighboring adjacent vertex ( $d\theta$ ). . . . .	69
3.6	Statistical analysis test case. Left: The vertex layout in the $X - Y$ plane has valence ranging from three to ten (for interior vertices), and contains a mixture of obtuse and non-obtuse triangles. Right: The mesh projected to an exponential surface. . . . .	70
3.7	Comparison of fitting methods applied to a paraboloid surface. A valence of six was used for the data shown. The cubic fit with computed normals does not converge to the exact curvature of 0.367. The one-ring and two-ring fits behaved similarly, with the one-ring fits being a little more accurate than the two-ring fits. . . . .	71
3.8	Impact of noise normal to a cylindrical surface on the discrete and fitting methods. A valence of six was used for the data shown. The discrete methods and one-ring fitting methods exhibit extreme sensitivity to noise. The cubic fit behaves as a two-ring method and, along with the two-ring quadric fitting methods, shows the least sensitivity to noise. . . . .	72
3.9	Impact of valence ( $n$ ) on the accuracy of the angle deficit method, applied to a cubic polynomial. Increasing Cell Size ( $\phi$ ) represents decreasing mesh resolution. Only valence six converges to the actual Gaussian curvature (0.338). This method is extremely inaccurate for a valence of three, probably due to the effect of obtuse triangles. . . . .	73

3.10	Impact of moving the target vertex along the surface toward a one-ring point ( $d\phi_T$ ). The valence is six for all methods. The discrete methods and the cubic fit with computed normals are very susceptible to this mesh quality issue, while the other methods show little sensitivity. . .	74
3.11	Integral Eigenvalue method applied to a sphere as a function of cell size. A valence of six is used for the data shown. The methods match well for a regular mesh centered at the origin, but degrade due to the minor distortion from projection of an offset mesh. . . . .	77
3.12	Effect of perturbation of an adjacent vertex along the surface toward or away from the target vertex. Both variation of the Integral Eigenvalue method show sensitivity to the mesh regularity. . . . .	78
3.13	Perturbation introducing non-uniform spacing around the ring. Again, both methods exhibit sensitivity to this mesh regularity parameter. .	78
3.14	Goldfeather's general surface test case. Left: Overall mesh. Right: Expanded view of local mesh. . . . .	80
3.15	Surface plot of exact Gaussian curvature for Grimm's general surface test case with complex curvature. . . . .	80
4.1	Conceptual view of a 1-D curvature map. The curves on the left represent mean and a Gaussian curvature functions for two sample vertices, $A$ and $B$ . The distance can be considered as expanding concentric rings (first five rings shown on the right), with the curvature value found by averaging the curvature values in the associated ring. . . . .	84
4.2	Test surface with vertices $A$ and $B$ highlighted. The first nine rings defined around vertex $A$ are color coded. The mesh is fairly uniform except for blending between sections. Note that the ring structure is still well-defined in spite of the skewness near its right edge. . . . .	87
4.3	Geodesic fans at two vertices. The first spoke of each fan is highlighted and used to track the relative orientation for 2-D fan comparisons. Fan parameters include the number and length of spokes, and the number of points per spoke. . . . .	88
4.4	Gaussian curvature (left) and mean curvature (right). Note that the Gaussian curvature ranges over $[-4, 6]$ , while the mean curvature ranges over $[-1.5, 1.5]$ . . . . .	89

4.5	Maps of Gaussian curvature and mean curvature as a function of distance from the point. Peaks and valleys tend to be more pronounced for the Gaussian curvature curve, which is the product of the principal curvatures ( $\kappa_g = \kappa_1\kappa_2$ ), compared to the mean curvature, which is an average ( $\kappa_m = \frac{\kappa_1+\kappa_2}{2}$ ).	90
4.6	Map of Gaussian (left) and mean (right) curvature as a function of distance from the selected vertex. The ring-based and fan-based curves for a particular point start out at the same value and initially have similar shape, but diverge due to a) non-uniformity of the rings, and b) fans sampling only a subset of the data, as the distance from the center increases. In this case, the fans cover a smaller area than the rings.	92
4.7	Similarity for points on two Ulna samples. Point A and B are similar, and Point C differs from Points A and B.	94
4.8	Two views of the test surface used for shape comparison. The left and right lobes in the front view are the same except for the addition of a dent (concave region) in the end of the left lobe.	95
4.9	Top: Similarity measure relative to vertex A plotted on the test surface. Bottom: vertex B. The color scale ranges from blue (high similarity to the selected point) to magenta (most dissimilar). Nine rings were used in the ring-based calculation. 20 spokes, 11 samples per spoke, were used in the 1- and 2-D fan-based calculation; the surface area is approximately the same as the ring version. Note that the 0-D measure (far left) is very noisy compared to the 1-D ring-based (center left) and fan-based (center right) measures. The 2-D measure (far right) shows few points with similarity to the selected vertex.	96
4.10	Similarity measure relative to a vertex on the tip of the ear of the Stanford Bunny. The color scale ranges from blue (high similarity to the selected point) to magenta (most dissimilar). The 0-D similarity has significant noise, while the 1-D methods isolates the tips of the ears much more cleanly. The 2-D method is even more discriminating, with similarity limited to the tip of the other ear.	97

4.11	Similarity measure for the Bull mesh. The color scale ranges from blue (high similarity to the selected point) to magenta (most dissimilar). The view on the left shows that the ring structure is very non-symmetric about the selected vertex, due to the irregularity of the bull mesh. Even so, the ring-based and fan-based 1-D methods provide comparable similarity measures. . . . .	97
4.12	Distance grids for select points. The similarity within groups, indicated by the darkest $3 \times 3$ boxes along the diagonal, and dissimilarity between groups, based on lighter off-diagonal squares, was most consistent for the 1-D ring-based measure. . . . .	99
4.13	The similarity curves for the most and least similar vertices. . . . .	102
4.14	The three-hundred most unique points based on similarity to to all other points. The 0-D method (left) picks up most of the peak curvatures, but finds a lot of isolated points in the neck and face region. The 1-D method (right) finds consistent groups of points reflecting key features in the mesh. . . . .	103
5.1	Bones making up the human wrist. Natural objects have subtle shape variations that are challenging to characterize. . . . .	106
5.2	Feature detection test surface. Left: Surface shape with peaks, pit, ridge, and valley. Center: Mean curvature scalar function. Right: Features highlighted by selecting a function threshold. With the proper threshold, this function can highlight useful features, however, the threshold must be found by experimentation. . . . .	107
5.3	The min-cut/max-flow graph cutting algorithm finds an optimal separation of the vertices of a mesh into a feature group and a background group. The cut is based on weights assigned to the mesh edges (solid lines) and to edges connecting the graph vertices to the feature and background nodes (dotted lines). . . . .	110



5.4	Graph cuts generated by the min-cut/max-flow algorithm on the local shape property for three graph cut categories: absolute value (left), negative (center), and positive (right). The absolute value graph cut picks up the peak and pit features, while the valley feature is only found in the negative graph cut and the ridge feature is only found in the positive graph cut. . . . .	112
5.5	Effect of the scale factor $\alpha$ on features identified using the min-cut/max-flow graph cutting algorithm. Representative cuts from the negative of the local shape property are shown. As $\alpha$ increases, more features are detected, and existing features become larger. At larger $\alpha$ the saddle region at the base of the peaks is detected. . . . .	112
5.6	Feature counts for the absolute value (left), negative (center), and positive (right) graph cut categories. Maintaining separate frequency counts for the three graph cut categories allows extraction of more well-defined features. . . . .	113
5.7	Composite feature sets for the absolute value, negative, and positive graph cuts, and the master feature set made by merging them. . . . .	114
5.8	Test case with Gaussian noise added. The function and final feature set are similar to the test case without noise. . . . .	115
5.9	Features detected on a Cyberware low resolution female face scan. The absolute value graph cuts pick up the nose chin and hair features, while the negative cuts detect the eyes. In spite of the smoothness of the mesh, master feature set captures the prominent features of the face. . . . .	116
5.10	Comparison of graph cut feature detection with sign of curvature segmentation for a high resolution Cyberware face scan. Before coloring by the sign of Gaussian and mean curvature, the curvature values were smoothed. The segmentation in the center uses a zero threshold to separate low curvature regions from higher curvature features. However, the master feature set provides more well-defined features. . . . .	116
5.11	Features detected for the Stanford bunny. Several features, such as the large sections of the ears and the features in the face region, are very intuitive. . . . .	117

5.12	Master feature sets for selected bone meshes. The Ulna is challenging due to the limited number of pronounced features and the significant difference between the scales of the features. Similar features were detected for cases A and B even though the resolution of the meshes is very different. Reasonable features were also identified for the Pisiform (second from right) and Capitate (far right).	117
6.1	Shape matching process steps.	120
6.2	Values of $s$ in $[0, 1]$ define a correspondence between a point of $f_1$ (blue) and a point of $f_2$ (red).	123
6.3	Surface meshes flattened into the plane for two bones.	124
6.4	Flattened mesh with Gaussian curvature for the radius bones.	125
6.5	Mapping from mesh to spherical parameterizations to common parameterization.	125
6.6	Steps in the process for determining candidate feature pairings.	128
6.7	Average curvature map for a feature (left) and feature reference points (right). Note: only the Gaussian curve is shown.	128
6.8	Triangles formed by three feature reference point pairs on the surface of two ulna bones.	129
6.9	Refinement of the feature reference points for the selected feature pairs on the surface of two ulna bones.	131
6.10	Adjustment of pin point locations based on similarity. The original pin point locations (left) are moved in an iterative process (center) to their final locations (right).	135
6.11	Iterative smoothing with a proper choice of weights can enable the spherical parameterization to better represent the relationships on the original mesh.	136
6.12	Creation of the common spherical parameterization. A common base mesh is created as an intermediate step in the alignment process.	138
6.13	Mesh and surface views for two lunate bones.	139
6.14	Features identified for two lunate bones.	139
6.15	Alignment of the lunate bones based on feature correspondence, and the pin points generated.	140

6.16	Spherical parameterizations used to determine correspondence (two left views). The two right views show the features from each lunate mapped onto the other lunate. . . . .	140
6.17	Geometry and mesh representation for two ulna samples. . . . .	141
6.18	Features detected for two ulna samples. . . . .	142
6.19	Alignment and trimming of the samples to a common portion of the ulna. Dark gray (center) indicates regions for which there is no corresponding point in the other object. . . . .	143
6.20	Alignment of the trimmed samples and the pin points generated. . . . .	143
6.21	Spherical parameterizations aligned using the pin points for the ulna samples. The views on the right show the features of one ulna mapped onto the other using the correspondence produced by the common spherical parameterization. . . . .	144
6.22	Additional pin points with adjustment based on similarity. Voronoi-like regions associated with each pin point are shown on the left. The adjusted spherical parameterizations are shown on the right. . . . .	144
6.23	The features of ulna 2 are shown on the left and mapped onto ulna 1 without (center) and with (right) additional pin points and pin point adjustment. . . . .	145
6.24	The features of ulna 1 are shown on the left and mapped onto ulna 2 without (center) and with (right) additional pin points and pin point adjustment. . . . .	145
6.25	Geometry and mesh representation for two radius samples. . . . .	146
6.26	Features detected for radius 1 and radius 2 bone samples. . . . .	146
6.27	Alignment and trimming of the samples to a common portion of the radius. Dark gray (center) indicates regions for which there is no corresponding point in the other object. . . . .	147
6.28	Spherical parameterizations aligned using the pin points for the radius samples. The views on the right show the features of one radius mapped onto the other using the correspondence produced by the common spherical parameterization. . . . .	147
6.29	Additional pin points with adjustment based on similarity. Voronoi-like regions associated with each pin point are shown on the left. The adjusted spherical parameterizations are shown on the right. . . . .	148

6.30	The features of radius 2 are shown on the left and mapped onto radius 1 without (center) and with (right) additional pin points and pin point adjustment. . . . .	148
6.31	The features of radius 1 are shown on the left and mapped onto radius 2 without (center) and with (right) additional pin points and pin point adjustment. . . . .	149
6.32	Geometry and mesh representation for two intermediate cuneiform samples. . . . .	149
6.33	Features detected for two intermediate cuneiform bone samples (left and center), and the resulting feature-based alignment (right). . . . .	150
6.34	Pin points generated for the intermediate cuneiform samples (left two views), and the spherical parameterizations aligned using those pin points. . . . .	151
6.35	Cross mapping features between intermediate cuneiform bone samples.	151
6.36	Geometry and mesh representation for two face scans. . . . .	152
6.37	Features detected for the two face scans (left and center) and the alignment based on these features (right). . . . .	153
7.1	Distances between corresponding points for the aligned lunate surfaces. The distance measures can identify where one surface is (a) inside, (b) outside, or (c) misaligned with respect to the other surface. . . . .	162
7.2	Distances between corresponding points for the aligned ulna surfaces. The maximum dimension is approximately 18. . . . .	162
7.3	Distance measure for corresponding points of the radius and intermediate cuneiform surfaces. The maximum dimensions are approximately 31 for the radius and 37 for the intermediate cuneiform bone. . . . .	163
7.4	Differences in surface normal for corresponding points for the aligned lunate surfaces. . . . .	164
7.5	Surface normal orientation measure for corresponding points of the radius and intermediate cuneiform surfaces. . . . .	164
7.6	Two lunate surfaces and the shape similarity measure for corresponding points plotted for two different scalar ranges. . . . .	165
7.7	Shape similarity measure for corresponding points of the ulna surfaces using different curvature map radii. . . . .	166

7.8	Shape similarity measure for corresponding points of the radius surfaces using different curvature map radii. . . . .	166
7.9	Shape similarity measure for corresponding points of the intermediate cuneiform bone surfaces using different curvature map radii. . . . .	167
7.10	Shape similarity measure for corresponding points of two face scans using different curvature map radii. . . . .	167

# Acknowledgments

I would very much like to thank my adviser, Dr. Cindy Grimm, for her support of my graduate study. I have learned a great deal from her in the process. I would also like to acknowledge the other members of my committee, Dr. Burchan Bayazit, Dr. Roger Chamberlain, Dr. Tao Ju, Dr. Erik Trinkaus, and Dr. Victor Wickerhauser, who have offered their time and expertise to assist in the evaluation of my research.

I would like to thank the members of the media and machines lab, particularly, Dr. Robert Pless, Dr. William Smart, Reynold Bailey, Robert Glaubius, Nathan Jacobs, Richard Souvenir, Ross Sowell, and Nisha Sudarsanam, who have always been willing to discuss new ideas and critique presentations and papers.

It has been a great pleasure to work with the department faculty and staff. Everyone has been very kind and supportive.

My thanks also go to The Boeing Company for their financial support and the flexibility that allowed me to pursue this research.

Finally, I would like to thank my parents and family, with special thanks to my sons Christopher and Jonathan for whom I try to set an example. Most importantly, I very much want to thank my wife Kathy, who has supported and encouraged me throughout. Her love has kept me going. And I also need to acknowledge my Lord, Jesus Christ, who has provided all of these blessings. To God be the glory.

Timothy David Gatzke

*Washington University in Saint Louis*  
*May 2007*

# Glossary

- Affine Transformation - A combination of rotation, translation, scale, and shear transformations.
- Block Matching - Finding the position in one image that corresponds to another image for a block of pixels (rather than single pixels).
- Correspondence - A 1-to-1 mapping between the points of one object and the points of another object.
- Curvature - A property of a curve or surface that indicates deviation from straight (curve) or flat (surface). Curvature can be different for each direction at a point on a surface, but there are two uniquely defined curvatures, called the principal curvatures, one being the maximum and the other being the minimum of all curvatures at a point. Two other useful curvatures, the mean and Gaussian curvatures, represent the average and product of the principal curvatures respectively.
- Feature - A point or region on a surface that has some shape property that makes it distinguishable from other locations on the surface.
- Geodesic - The shortest path between two points, where the path and the points are constrained to lie on a given surface.
- Homeomorphism - A mapping between points on different objects that is continuous, one-to-one, and onto, and the inverse of which is also continuous.
- Manifold - A surface that is locally Euclidean.
- Mesh - A representation of a 3-D surface consisting of vertices, which represent points on the surface of the object, and faces, which define the connectivity between vertices.

- Modality - A specific technique for generating data from a 3-D object, such as computed tomography (CT), magnetic resonance imaging (MRI), etc.
- Multi-modal - Integrating information from two or more modalities applied to the same subject.
- Parameterization - A homeomorphism from a surface to a canonical shape of the same genus, or a continuous, 1-to-1 mapping from a piece of a surface to the plane.
- Pose Estimation - Correlating data from an image which is a view of an object with a model of the object to determine the orientation of the image viewpoint relative to the object.
- Projective Transformation - Transformation from an  $n$  dimensional space to an  $n - 1$  dimensional space.
- Registration - Process of determining the transformation that orients one object in space with respect to another object.
- Similarity - A comparison between two points that measures how much the shape in the vicinity of the two points is alike.
- Template Matching - Classification of unknown samples by comparing to known prototypes or templates (e.g., detection of  $n \times n$  subimage within an  $N \times N$  search area that best matches  $n \times n$  template  $f$ ).



# Chapter 1

## Introduction

Modeling the shape of three-dimensional (3-D) objects is an important capability for computer-based modeling and simulation, graphics, and computer vision. Shape is a fundamental property of objects. Other properties, like texture and color, may often vary considerably among objects of the same type. Complex shapes can be found in naturally occurring structures, such as bones or organs within the human anatomy, as well as man-made structures, such as automotive or aircraft designs. Examples of organic and man-made shapes are shown in Figures 1.1 and 1.2 respectively. Shape modeling applications include design and manufacturing, medical diagnostic imaging and treatment, terrain mapping, and automated surveillance.

The ability to compare 3-D shapes is important for many applications. There is a continuing need for improved medical diagnosis, where for example, automated tracking of tumor growth or bone deterioration could support early detection and treatment planning. Comparison of data from healthy subjects would provide a statistical understanding of what a ‘normal’ shape is. The ability to provide a detailed assessment of the deviation from the norm would support treatment decisions and reconstruction planning. Shape comparison techniques could also be used to search for similar shaped objects in a database for classification based on known shapes. The role of automated shape comparison will grow with the availability of 3D surface data, and will be a major benefit to the domain knowledge expert.

Inter-patient variation and positioning issues make it difficult to accurately align data from different patients, data from the same patient taken at different times, or data captured using different modalities. Solutions to the 3-D object alignment



Figure 1.1: Bone structure in a knee joint. From “Accurate Measurement of Three-dimensional Natural Knee Kinematics Using Single-Plane Fluoroscopy” by Rahman et al., 2003 Summer Bioengineering Conference.



Figure 1.2: Machined parts. From Stamping Press - Preferred Tool and Die, Inc., URL=<http://www.preferredtool.com/oemmarkets.htm>.

problem have generally relied on manual input to identify corresponding features. However, manual selection of corresponding features and subjective determination of the difference between objects is a time-consuming process requiring a high level of expertise. Automatic shape comparison techniques can be used to reduce the need for manual labor by assisting in the automated alignment of 3-D models.

Comparing shapes and tracking shape changes requires identifying similarities and quantifying the differences between objects based on their shapes. In order to extract information from shape models, metrics are needed that can capture and quantify properties of the object surface. Shape comparison can also incorporate or be used in a number of related tasks:

- *Object recognition* is primarily concerned with finding properties that distinguish objects, without the need to identify in what way, or to what extent, the objects are different. The goal is to determine if a particular object is present, or to search through a database to find the objects which match most closely to a given query object. An emphasis is placed on efficiency, leading to approaches which reduce the dimensionality of the comparison, and focus on the gross shape.
- *Computer vision* compares two-dimensional (2-D) images of a scene to either a 3-D model or other 2-D image patterns to determine if a particular object is present in the image, and if possible, identify its orientation.
- *Feature detection* finds points or higher level structures with distinguishing properties that can be used to establish correspondences between objects.
- *Correspondence generation* finds a mapping between points in different images or models. Correspondence generation can also be applied to features, but not all features of an object will necessarily have a corresponding feature in the other object.
- *Registration* addresses the issue of relative orientation. Various methods have been used to find the best alignment of objects. Often these have been applied to register different views of the same object. The views also may have been made

using different medical imaging modalities such as X-ray, computed tomography (CT), magnetic resonance imaging (MRI), positron emission tomography (PET), etc.

- *Pose estimation* is similar to registration. Typically, pose estimation compares an image to a 3-D model or to an indexed set of views to determine the viewpoint of the image relative to the object.
- *Similarity measures* quantify the similarity or dissimilarity between objects.

## 1.1 Detailed Shape Comparison Goal

The objective of this research is to develop a methodology that supports detailed comparison of 3-D objects. This object comparison problem can be stated as follows:

Input data and conditions:

- Representations for two (or more) similar objects.
- The objects can be oriented arbitrarily.
- The mesh resolution of the representations can be different.
- Different portions of the surface of the object may be represented.
- Objects should have some corresponding features, but both objects may have additional features not present in the other object.

Goals:

- Find the best alignment of the two objects.
- Find a mapping between corresponding features occurring in both objects.
- Quantify the difference between the corresponding features in the two objects.
- Identify features that show up in only one of the two objects.

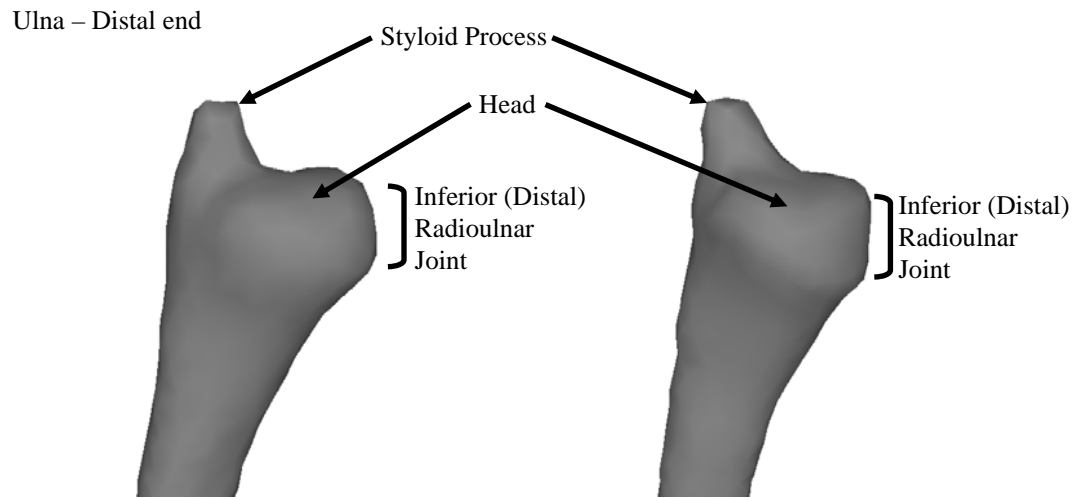


Figure 1.3: Two similar bones that are candidates for detailed shape comparison.

- Classify the nature and magnitude of these features.

The initial application targeted by this research is comparison of bone surface shape. An example of two similar bone shapes is shown in Figure 1.3. The bone surfaces are represented by a mesh created from CT scan data of two different subjects. The mesh consists of points sampled from the surface of the object, plus connectivity information between points used to form triangular faces approximating the surface shape. We assume no other information about the interior of the object, or about other properties on the surface.

In our case, these meshes represent naturally occurring, organic, surfaces. Therefore, it can be inferred that there are few sharply delineated features, such as might occur in a machined part. Because the surfaces are generally smooth, local features are less distinguishable from one another. The existence of extra features in one or both objects, along with noise in the sampled data, can make it more difficult to compare the shape of the objects.

## 1.2 A New Curvature-Based Shape Matching Technique

To facilitate shape comparison, we develop a shape representation that is sensitive to subtle shape differences but relatively insensitive to noise, and use this representation to detect features of the objects. The features are used, along with a measure for shape similarity, to compute a correspondence between the objects, which then allows shape comparison based on the shape properties at corresponding points. An advantage of this approach is that the final comparisons depend on the similarity-based correspondence and not on a physical three-dimensional alignment.

Our novel shape representation, the curvature map, describes shape as a function of surface curvature in the region around a point. Surface curvature is one of the fundamental properties of the surface. The size of the curvature map region is only restricted by practical limitations determined by the size and shape of the object.

Feature extraction is another key ingredient in the shape matching process. Features may be global or local, and may represent fine or gross properties of the object. We generate feature regions on the surface by applying the min-cut/max-flow graph cut algorithm to a local shape property derived from the curvature map. To capture features of different scales, a multi-pass approach is used, where the local shape property represents differently sized subsets of the underlying curvature map. This allows feature detection to proceed without requiring user input or parameter tuning.

We use features ordered by strength, the similarity of pairs of features, and pruning based on geometric consistency to efficiently determine key corresponding locations on the objects. For genus zero objects, the corresponding locations are used to generate a consistent spherical parameterization that defines the point-to-point correspondence used for the final shape comparison. Various shape difference measures are calculated at corresponding points in order to evaluate the differences between the objects.

## 1.3 Organization of the Dissertation

Chapter 2 describes related topics from the fields of shape modeling, object recognition, computer vision, image registration, etc. Chapter 3 presents a survey of existing techniques for estimating curvature and provides a suite of test cases for assessing curvature estimation techniques. The *curvature map* is developed as a new local shape similarity measure in Chapter 4. In Chapter 5, a multi-scale framework for feature detection is presented. This framework combines a local shape property based on the curvature map with an efficient graph cut algorithm. Chapter 6 illustrates feature-based object alignment, and shape similarity assessment of the aligned objects is described in Chapter 7. Finally, in Chapter 8 summarizes this research and presents areas for further study.

## Chapter 2

# Background: Reasoning About Object Shape

Analysis and comparison of three-dimensional (3-D) objects is important for applications such as medical imaging, comparative anatomy, computer vision, facial recognition, and forensic identification. Recent developments in imaging technology have provided extensive detailed data for internal organs as well as the exterior shape of the human anatomy. The desire to use this data for diagnosis and guidance during medical procedures has fueled the search for automated analysis techniques. In computer vision, object recognition is needed to identify 3-D objects present in a scene. Also, the expanding capability to store and access large databases has produced a need for automating comparison of objects for search and retrieval operations. For example, efficient access to the large number of existing machine parts, distributed across manufacturers' web sites, could streamline the design process for the CAD/CAM (computer-aided design/computer-aided manufacturing) community. Comparison and matching of 3-D objects based on shape relies on shape similarity measures, feature detection, correspondence, and registration techniques.

Section 2.1 presents some general shape properties, while shape representations and comparison techniques are presented in Sections 2.2 and 2.3 respectively.



## 2.1 Shape

Shape is a fundamental property of objects, but is difficult to describe concisely in the general case. This has fostered much research in ways to describe and compare the shapes of objects. This is distinct from the way we represent or model the object. We will distinguish between the object representation, which gives us a definition of the object, and the shape representation, which tells us something about the shape properties of the object. The shape representation will generally be created from the object representation, but the two may be very different. While there are several ways to represent an object, the shape of an object is considered to be independent of the object's representation.

Objects may be decomposed into basic components that may then be used for comparing components and reasoning about the object. Whether at the object or component level, a way to measure the similarity of shapes is required. We now describe some common approaches to represent and describe shape and their strengths and limitations.

### 2.1.1 Object Representation

Information about an object comes from some sampling of that object. This can be one or more images of the object, a scene containing the object, range data, or full 3-D models. Representations of 3-D models can employ a boundary representation or a volume representation.

The volume of an object can be represented by dividing the space into volume elements called voxels, and identifying all of the voxels contained within the object. Generally, the voxels of most interest are those on the boundary of the object. Another volumetric representation of an object is a boundary representation solid. It is defined by a set of construction surfaces that each carve away at the 3-D space, eliminating anything outside the surface. The object definition is then everything that is left. Recovering the surface of the remaining volume is problematic, as the surface

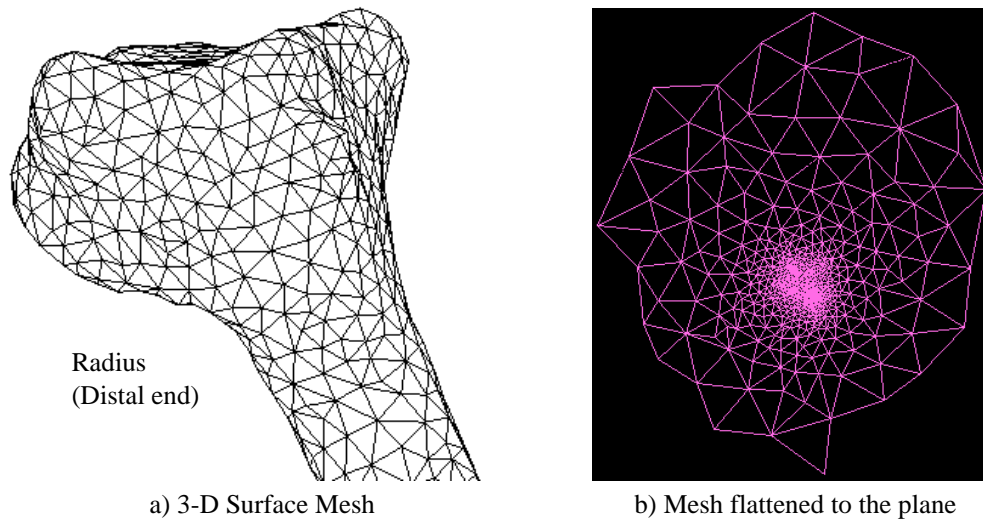


Figure 2.1: A mesh representation of the surface of a bone. (a) 3-D surface mesh. (b) Surface mesh flattened to the plane. Note the distortion of the triangle areas in the flattened mesh compared to the 3-D surface mesh.

may consist of many pieces that are defined implicitly as sections of the construction surfaces. Finding the bounding edges of these pieces is a complex and tedious task.

Nielson [124] discusses several aspects of volume modeling, including different sources and representations of volume data, and multi-resolution models based on wavelets. Wavelets are a common approach to generating different levels of detail. Southern et al. [161] attempt to apply wavelets to triangular meshes as a multi-resolution analysis tool. While theoretically useful for refinement, compression, and generating multi-resolution models, a general mesh seldom meets their connectivity requirements, limiting the use of this approach in practice.

Bonneau and Gerussi [17] generate different levels of detail by removing vertices using a greedy algorithm. Local neighborhoods around each point are used to reduce the problem to a local one. As vertices are removed, the basis functions at adjacent vertices are updated. Both 2-neighborhoods and 3-neighborhoods provide reasonable approximations, even though they may not be optimal. This gets around the mesh requirements to generate the wavelet representation, but still does not decompose the data based on the identified features.

Pang and Furman [131] discuss data quality issues in visualizing sparse data sets. They look at various interpolation techniques, particularly Shepard's methods and Hardy's multiquadric methods. It is possible to get a reasonably smooth fit through the data with an appropriate selection of the free parameter, but the sparse data assumption does not generally hold for the problems we are trying to solve.

Other representations define the surface of the object directly. One of the simplest representations for 3-D shapes is a surface mesh. Meshes are composed of a set of vertices and a set of faces that define the connectivity between the vertices. Mesh vertices can come from surface coordinates generated by a scanning device, points located on an analytical definition of the surface, or any number of other techniques. The mesh faces are defined as polygons connecting mesh vertices. While the face polygons can have an arbitrary number of sides, triangular faces are the simplest and most common. A mesh representation for a bone is shown in Figure 2.1a.

Farin [49] deals with how to quantify shape using principles in computer-aided geometric design (CAGD). He describes the development of B-spline curves and surfaces, and the use of curvature plots to highlight the detailed shape properties. Loncaric [104] gives an overview of shape analysis techniques including evaluation criteria for shape representation methods, and issues related to visual perception.

Operating on splines, planes, or analytic functions requires a detailed understanding of the underlying representation. However, such surfaces can always be used to generate a mesh. There is extensive literature on methods for constructing a mesh from a set of scattered data points [103, 76, 3, 11], splines, and implicit surfaces [105], and these methods will not be covered here.

A related topic is the representation of 3-D objects using 2-D range images. A range image consists of an  $M \times N$  array of pixels, with image intensity indicating depth. For the subset of the image associated with a particular object, this depth can be used to infer shape. In the remainder of this document, the focus is on the general 3-D mesh representations. In general, meshes lack the regular array of pixels provided in range data, however, we still consider methods developed for range data to see if they can be extended to the more general mesh case.

### 2.1.2 Shape Decomposition

It is often useful to decompose shape into its basic components. Decomposition schemes range from hierarchical schemes that essentially form compact representations, to heuristic schemes that subdivide an object based on some local criteria. There are also a number of general discussions of the aspects of shape.

Kim et al. [88] propose recursive shape decomposition based on convexity and the morphological operators dilation, erosion, opening, and closing. The shape is broken into body and branches. Over-segmentation is corrected by merging branches.

Mortara [121] decomposes an object based on topology changes of the intersection of the object with spheres located at each vertex, as the radius of the spheres is increased.

Shape decomposition methods for 3-D volumes have been developed based on their topology [40], and morphological tools [110]. However, volume decomposition provides volumetric features rather than surface features, and also is only applicable to closed objects.

Decomposition schemes can also generate a graph representing the structure of an object. Graph representations and comparison techniques will be discussed later in this chapter. We also look at two ways in which the surface of an object can be decomposed; segmentation and feature detection. Segmentation separates the entire surface into regions, while feature detection identifies regions that typically cover a subset of the entire surface.

### 2.1.3 Shape Similarity

Similarity measures quantify the similarity or dissimilarity between objects by computing distances between shape representations, such as sets of points, feature vectors, histograms, signatures, or graph representations. A number of these shape representations have grown out of image analysis for range data [57, 72] or medical images [46]. Similarity measures can be global, applying to the entire object, or local, applying to

a point or local subset of the object. They provide data to answer questions such as: ‘How much do two objects,  $A$  and  $B$  resemble each other?’, ‘Does  $B$  resemble  $A$  more than some other object resembles  $A$ ?’, or, ‘Does some part of  $A$  resemble some part of  $B$ ?’ There are other variations of the problem, for example, can we simplify an object representation, but still match within some tolerance, or can we find a series of shape transformations to morph from  $A$  to  $B$ .

Two approaches to measuring similarity are (1) directly computing a similarity value for a pair of objects, or (2) compute a descriptor for each object and then take the difference between their descriptors. If the descriptor is a single value, computing the difference is straight forward. If the descriptor has more than one value, the difference can be calculated using a distance measure such as the Minkowski distance,

$$L_p(x, y) = (\sum_{i=0}^k |x_i - y_i|^p)^{1/p}$$

where  $x$  and  $y$  are two shape descriptors, and  $x_i$  and  $y_i$  represent the values of the  $i^{th}$  dimension for the  $N$ -dimensional shape descriptor. For  $p = 2$ , this is just the standard Euclidean distance  $L_2$ .

A significant area of research is retrieving objects from a database via a shape-based search. The goal of efficient retrieval from large image or 3-D object databases dictates the use of lower dimensional signatures for searching. A smaller signature translates to more efficient comparison of the query object signature to the signatures of a large number of candidates in the database. However, the signature must be discriminating enough to correctly retrieve similar items while minimizing false positives. Complicating factors are missing and mis-oriented polygons, and overlapping or self-intersecting surface sections common in existing models.

Two approaches for object recognition are to measure the similarity of a shape signature or a feature graph. Signatures provide a simpler representation of the object, which is used to assess the similarity of objects. Feature graphs embed the topological relationships between features of the objects. The similarity of the graphs or sub-graphs is used to establish correspondence or similarity.

The basic object recognition problem involves determining which objects appear in a scene, and where they appear. The input to the problem is an image of the scene

and knowledge of various objects and how they may appear. The knowledge of objects may be derived from a 3-D model of the object, or may be embedded in a collection of views of the object. Pope [138] surveys various model-based object recognition methods along with the associated shape representation schemes that underlie these methods. The general steps for object recognition include detecting features, organizing features into stable groups, using those features to select a likely model from a set of models, finding the best match between the image and model features, and finally deciding if there is reasonable evidence that the object is present. It concludes that most methods are dependent on specific classes of objects, however, it does not address how features are detected. Most of these same steps show up in our shape comparison approach, and so it is instructive to see how the object recognition problem is being addressed.

Attempts have been made to define signatures for shape matching. Signatures may be global or local, and provide a compact representation that results in more efficient comparison at the expense of their ability to discriminate shape. Several methods employ histograms that represent the distribution of some property relative to a point on the object surface. Global similarity measures are applicable to coarse shape matching for shape retrieval, but generally provide limited discrimination between similar shapes. Moreover, in general, methods based on distances between points, such as Hausdorff distance, multi-resolution Reeb graphs [74], shape distributions [129] [130], and spin images [84], are sensitive to the distribution of the points. There have been a few attempts to create local signatures.

### 2.1.4 Medical Imaging

Medical imaging is just one of the applications for shape matching, but it has been an important source of data. Medical imaging operates with data from a number of different imaging techniques. Van den Elsen et al. [46] describe the challenges of matching medical images from different modalities, such as SPECT (single photon emission computed tomography), PET, MRS (magnetic resonance spectroscopy), MRI, ultrasound, X-ray, and CT, dealing with differences in patient position, and the effect of image acquisition parameters. They define criteria for classifying registration

methods according to dimensionality, whether the properties are extrinsic or intrinsic, global or local, the elasticity of the transformation (rigid, affine, projective or curved), whether it is interpolating or approximating, the way parameters are determined, and how much interaction is required. Current methods are then classified according to these criteria.

Much progress has been made in object matching, particularly in the medical field. The use of non-rigid transformations, segmentation, surface curvature data, and graph structures has already proved useful for a variety of medical imaging applications. The steps used in matching medical images can be applied to other object matching applications as well.

Of most relevance to shape matching are the fully automatic methods based on intrinsic properties. Most commonly, 3-D medical images come from a set of image slices. The resolution within a slice is generally much finer than the resolution between slices. Some matching methods are applied to the full set of 3-D data, while others are more suited to an individual 2-D slice. Alternatively, this data may be used to generate a 3-D surface representation.

## 2.2 Shape Representations

This section highlights different ways to represent shape. Several of these are listed in Tables 2.1 and 2.2. Recurring themes include graph-based methods, feature vectors, multi-resolution schemes, and the use of invariants.

Several signatures have been defined for shape matching. Global signatures result in more efficient comparison at the expense of their ability to discriminate fine shape differences. Global signatures are generally used to recognize complete objects, while matching objects with occlusion requires recognition based on a part of the object, implying a local signature.

Table 2.1: Shape Representation - Descriptors

<i>Shape Representation - Descriptors</i>			
Examples	Paper	Strength	Weakness
<i>Global Descriptors</i>			
Feature vector	Shams et al.(2001)	Efficient to compute and compare	No shape difference information
Moments	Zhang & Chen(2001)		Not distinguish between similar objects
Invariants	Subrahmonian et al.(1996)		
Coarse metrics	Corney et al.(2002)		
Shape Distribution	Osada et al.(2001)	Database search for similar objects	No shape difference information, Not distinguish between similar objects
Shape Context	Mori et al.(2001)		
Spherical Harmonics	Kazdhan & Funkhouser(2002)		
Volume	Kim(2003)	Intuitive subdivision of volume	Volume rather than surface features
Decomposition	Mortara et al.(2003)		Sensitive to relative size of components
	Maintz et al.(1997)		
	Dey et al.(2003)		
<i>Local Descriptors</i>			
Spin Image	Johnson & Hebert(1997)	Represents local shape	Requires specification of local region size
Local feature histogram	Hetzal et al.(2001)		Computing geodesic curves
Geodesic fans	Zelinka & Garland(2004)		Only points of interest
Point fingerprint	Sum et al.(2003)		



Table 2.2: Shape Representation - Graphs, Segmentation, and Features

<i>Shape Representation</i>			
Examples	Paper	Strength	Weakness
<i>Graph Representation</i>			
Medial axis	Bloomenthal & Lim(1999)	Can represent large deformation	Global Sensitive to point distribution Not represent local shape differences
Shock scaffold	Leymarie & Kimia(2001)		
Reeb graph	Hilaga et al.(2001) Chen & Ouhyoung(2002) Mortara & Patane(2002)		
<i>Segmentation</i>			
Watershed	Mangan &Whitaker(1999) Steger(1999) Pulla et al.(2001)	Based on curvature plus	Oversegmentation Noisy Sensitive to threshold
Sign of curvature	McIvor et al.(1997) Wilson & Hancock(1999) Vivodtzev et al.(2003)	Detects primitive shape Multi-resolution representation	Oversegmentation Noisy Best for coarse segmentation
<i>Feature Detection</i>			
Ridge & Valley	Interrante et al.(1995) Ma & Interrante(1997)	Based on curvature	Not for smooth surfaces
Scale Invariant Feature Transform (SIFT)	Ke & Sukthankar(2004) Lowe(2004)	Large number of features	No control of feature type

### 2.2.1 Global Descriptors

Lin and Perry [101] present methods to calculate shape descriptors such as the genus, area, and Gaussian curvature. This information is useful; however, it is not detailed enough for detailed shape matching of similar objects. For example, in addition to overall area, we also would like area broken down according to regions associated with specific features.

Kumar et al. [94] extract the principal components of an image to use as the optimal filter. For matching, they look at the correlation of the peak values and signal-to-noise ratios.

Shams et al [153] generates feature vectors from an image using the response to Gabor wavelets. They present an extension to the typical graph matching similarity measure by including similarity of topologically adjacent nodes in addition to the usual similarity of corresponding nodes.

Ohbuchi et al. [127] define a complex feature vector as a concatenation of nine simpler feature vectors. The nine feature vectors are formed by computing three statistics, (1) moment of inertia, (2) average distance of surface from the axis, and (3) variance of distance of the surface from the axis, for each of the three principal axes of inertia of the model. Each of these vectors consists of values defined parametrically at discrete locations along the associated axis. Their results showed this method to work best for models with some form of rotational symmetry, but the key concept is the ability to combine multiple properties into a complex feature vector.

Corney et al. [34] examine four coarse shape metrics for filtering a search for similar shapes. One is based on properties of the bounding box aligned with the principal axes of the object, and the other three are ratios between properties of the object or its convex hull. The measures relating object volume to convex hull volume and convex hull area cubed to convex hull volume squared were the most discriminating without eliminating many good match candidates.

Zhang and Chen [187] present efficient algorithms to calculate features directly from a mesh representation, whenever the feature can be written as a signed sum of features

of an elementary shape. The features they use are area (for 2-D object recognition) or volume (for 3-D object recognition), moments, and the Fourier transform.

Subrahmonia et al. [166] describe a recognition technique for 2-D and 3-D objects using invariants of higher degree implicit polynomial functions. This method requires that the data set be segmented so that each segment corresponds with one object in the database. Then an implicit polynomial is fit to the data and invariants are computed for the polynomial coefficients. A probability of error based on a weighted distance measure is used to compare these invariants with those of the objects in the database.

Olver et al. [128] consider two affine invariant edge detection schemes. One is based on weighted differences of images at different resolution within an affine invariant scale-space. The other method involves developing an affine invariant gradient function. These edge detectors are then extended to define affine invariant active contours and affine smoothing functions. They apply these methods to 2-D images, but state that extension to 3-D can be accomplished using a corresponding volume functional.

Zhang and Fiume [188] perform shape matching based on normalized Fourier descriptors. They use the  $L_2$  distance between normalized weighted Fourier descriptor coefficients to determine similarity. The Fourier descriptors come from decomposition using the eigenvectors of a mid-point smoothing operator.

Starting with a voxel representation, Kazhdan and Funkhouser [86] split the model into  $M$  concentric spheres, centered at the center of mass. Each sphere is then decomposed into  $N$  harmonic components, producing an  $M \times N$  signature. Since the harmonic components do not depend on the orientation of the concentric sphere, the resulting signature is invariant to rotation. They apply this method to shape retrieval from large model databases.

Similarly, Novotni and Klein [126] develop 3-D Zernike functions in terms of harmonic polynomials. Vectors that are invariant to rotation are derived by applying Zernike functions to a voxel representation of the object. The distance between these vectors is used for matching in the shape retrieval process.

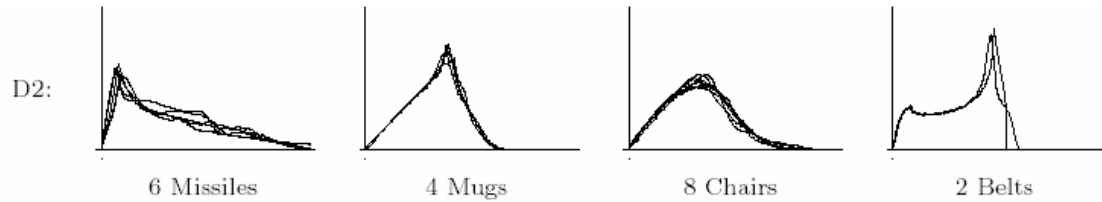


Figure 2.2: Shape distributions for four object classes. Adapted from “Shape Distributions”, Osada et al., ACM Transactions on Graphics, 21(4), 2002.

Tenenbaum [174] presents an isometric feature mapping procedure that maps a set of images to points in a low-dimensional Euclidean feature space. This low-dimensional Euclidean embedding captures the intrinsic similarities of the images. The crux of the method is computing a geodesic distance between observations. These distances are used to compute a global geometry-preserving map using multi-dimensional scaling.

Mokhtarian, Yuen, and Khalili [118, 185] propose a multi-resolution scheme utilizing surface curvature. Different levels of the multi-scale description are generated by parameterizing the surface locally by semi-geodesic or geodesic polar coordinates and then smoothing with a 2-D Gaussian convolution. Curvatures are calculated at each level by estimating derivatives with respect to the geodesic coordinates by convolution with the partial derivatives of the Gaussian function, and then computing mean and Gaussian curvature from these derivatives.

Osada et al. [129, 130] develop *shape distributions*, a signature based on the probability distribution of a selected metric. This signature is represented as a geometric histogram. For one particular metric, which they call D2, they create a distribution from the distances between randomly selected pairs of points. Figure 2.2 shows the D2 signatures for several objects in various classes. The  $L_1$  distance is the preferred way to measure the similarity of shape distributions. By using a random sampling, the method is not severely impacted by local problems or missing sections in the object model. Shape distributions are invariant to rigid motions. Figure 2.3 shows the best five matches returned from a database in response to select query objects. The ability to classify shapes with 66% accuracy is reasonable for coarse matching, but is not sufficient for discriminating fine details.





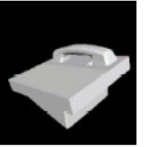


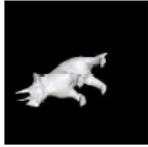

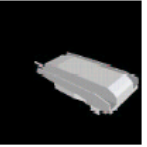
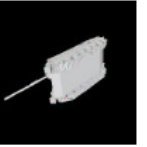

Query	Five Top Matches				
	 0.060	 0.066	 0.078	 0.079	 0.086
	 0.051	 0.052	 0.073	 0.086	 0.101

Figure 2.3: Best matches returned from a database for select query objects. Adapted from “Shape Distributions”, Osada et al., ACM Transactions on Graphics, 21(4), 2002.

Mori et al. [119] apply *shape contexts* to the shape retrieval problem. Shape contexts represent the shape of an object, with respect to a particular point on the object, as a 2-D histogram of the relative coordinates of other points sampled from the surface. Shape contexts can be computed for representative points on an object and stored in a database. Shape retrieval is reduced to computing a number of shape contexts for a query object, and searching for similar shape contexts in the database. Shape contexts are useful for coarse shape matching. Because they use a sampled set of points from the surface, shape contexts are relatively insensitive to moderate levels of occlusion, but this sampling of points also limits their usefulness for detailed shape matching.

## 2.2.2 Local Descriptors

Johnson and Hebert [84] propose the *spin image* as a way to encode the global shape of the object with respect to any oriented point on the surface. Oriented points consist of a point on the surface and a normal direction. Spin images can be calculated at each point on the surface and also for points in a scene. The spin image is stored as a discrete 2-D array. Finding similar spin images in a scene and in the 3-D model establishes a correspondence between them. Spin images can take into account all or

a subset of the points on the surface of the object. However, errors can be introduced if the distribution of points on the surface of the scene is not proportional to the distribution of points on the model.

Geodesic fans [186] represent a local surface resampling that provides a uniform structure in the neighborhood around a vertex. In particular, a geodesic fan consists of a set of spokes, and a set of samples on each spoke. The spokes are geodesic curves marching out across the surface from a central point, equally spaced in the conformal plane of the point’s local neighborhood. With the samples equally spaced along each spoke, they form a local geodesic polar map around the point. Zelinka and Garland use interpolated normal geodesics [15] where possible, reverting to straightest geodesics [137] if the smoothness criterion for interpolated normal geodesics is not met. Different properties can be represented as the signals at the points of the polar map. Zelinka and Garland use distance from the tangent plane and non-geometric properties such as texture to perform similarity based editing, however any other property could be used as a signal in the geodesic fan construct.

The point fingerprint [167] is a signature for local shape matching. The point fingerprint at a point consists of a set of concentric geodesic circles projected to a tangent plane. Points of interest are selected by applying a threshold to an irregularity measure on one of these contours. Shape similarity is computed by comparing corresponding normals and contour radius along each contour. This is analogous to the similarity calculation for geodesic fans [186], with the normal and projected radius used as the signal at the fan points. However, this similarity is only computed for the subset of ‘interesting’ points. Unlike these approaches, we are looking for subtle shape differences that require more than signatures just at ‘interesting’ points.

### 2.2.3 Graph Representations

Skeletal methods represent the shape of an object by capturing the structure of the object, along with properties at the points on the skeleton that define the distance from the skeleton to the surface. Bloomenthal and Lim [16] discuss animation of an object by defining motions of its skeleton, in addition to issues in generating the skeleton and reconstructing the object surface from the skeleton. Siddiqi et al. [160]

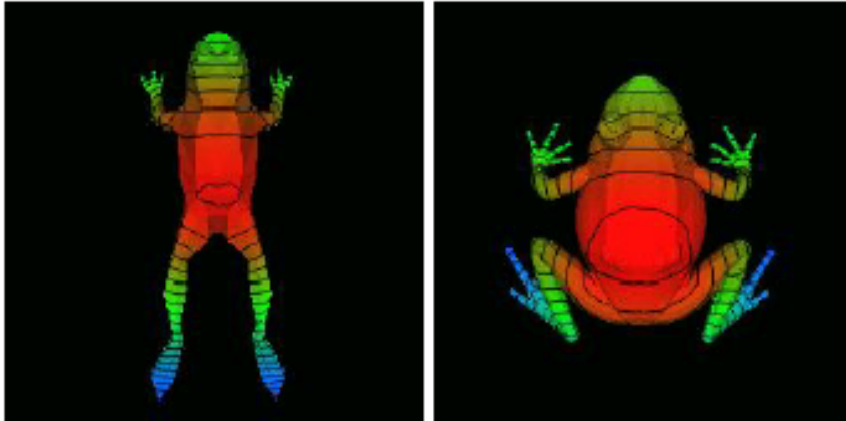


Figure 2.4: Geodesic distance function for a frog in two positions. From “Topology Matching for Fully Automatic Similarity Estimation of 3D shapes”, Hilaga et al., SIGGRAPH 2001.

match shock graphs by converting the shock graph to a tree structure and recursively matching rooted subtrees.

While the medial axis of a 2-D shape is a graph structure, for a general 3-D object the medial axis is represented as a surface. The shock scaffold, proposed in Leymarie and Kimia [99], is a way to represent the medial axis of a 3-D object as a graph. Points on the medial axis are classified as one of five types, based on the type of contact with an osculating sphere. Two of these point types become isolated points in the graph, while two other point types form curves. The direction of increasing distance to the boundary is used to add the notion of flow along curves.

Given the skeletal representation for the shape of an object, the structure of the skeleton and the properties represented at points on the skeleton can be used as features to compare objects. Mortara and Patane [120] uses a Reeb graph based on curvature properties as a skeletal representation. Regions of high curvature are treated as surface features. A graph is constructed by growing from these curvature regions to cover the surface. The starting point is a surface mesh, and smoothness of the skeleton relies on a uniform mesh. An advantage of this representation is that it is invariant under affine transformations.

Hilaga et al. [74] presents a matching technique using a multi-resolution Reeb graph. They define a function at each point based on the geodesic distance to all of the other points on the surface. The algorithm then generates the Reeb graph as a skeleton representation of the segmentation of the surface based on the values of this function. Different resolutions are generated from different levels of segmentation. Matching is performed by comparing the graph structures at the different levels. This method can match objects with significant relative deformation, as long as the geodesic distances do not change too much. An example of the geodesic distance function for a frog in two different positions is shown in Figure 2.4. The Reeb graph is primarily global rather than local. Chen and Ouhyoung [26] improve on the method of Hilaga et al. [74] by additional preprocessing, speeding up re-sampling to break edges greater than some threshold, and calculation of geodesic distances.

Bespalov et al. [13] present a framework for 3-D shape matching using scale-space decomposition. The first step is to decompose the model  $M$  into its  $k$  most significant features using singular value decomposition (SVD) clustering techniques. The graph nodes are the features created by recursive decomposition of the model, and edges relate a feature to its sub-features. The graphs can be interpreted as a tree structure, and graph similarity measured in terms of the similarity of the subtrees.

Graph representations, such as skeletons [16, 90] and multi-resolution Reeb graphs [74], like algorithms based on point sets [12, 146, 29, 4, 23, 107], can be useful for computing similarity and registration. But these methods are primarily global rather than local and do not identify local features of interest. Often, they are also sensitive to the distribution of the mesh points.

### 2.2.4 Segmentation

Segmentation is the process of dividing a surface into regions based on some surface property. These methods can be based on the signs of the Gaussian and mean curvatures [114] [184], isosurfaces and extreme curvatures [182], or watersheds of a curvature function [111] [112] [142]. Related methods compute critical points using principal directions.



One of the benefits of segmentation is a reduction in the complexity of the matching problem. This occurs because the smaller number of regions is typically more compact than the original object representation. Because these regions are based on intrinsic properties of the surface, they are also less dependent on the resolution of the surface representation. In the methods above, curvature is the most common property upon which the segmentation is based. Unfortunately, splitting the surface into regions still gives only coarse information about the differences between local regions, and small changes to the shape can cause large changes in the segmentation.

Pohle et al. [136] propose a two-level segmentation process in which an initial coarse segmentation is followed by adaptation of an active surface to the object boundary. This method operates on a 3-D voxel representation of the object. Applied to liver segmentation from 3-D CT images, the method shows promise for automatic segmentation as part of the feature detection process.

Maintz et al. [110] describe the use of simple morphological tools for voxel-based registration of 3-D medical images. They employ the morphological operators of erosion, dilation, opening, and closing to enhance contrast and simplify images. They apply these techniques to images of different modalities. These methods are limited to rigid transformations and are currently too time consuming for real-time applications.

Vivodtzev et al. [182] perform segmentation of the brain surface based on mean and Gaussian curvature. An aligned set of images is used to provide volume data and the surface is obtained via isosurface extraction. Key details of their technique include preprocessing of the data with a low-pass smoothing filter, ensuring a single surface by using a surface growing algorithm, and use of a multi-resolution representation to determine the most significant features. The cerebral cortex consists of concave (sulci) and convex (gyri) folds. The most convex and most concave locations are used as seed points for automatic generation of a topology graph. Because of the complexity of the surface, the coarse segmentation is more useful for capturing the basic topology of the object.

Wilson and Hancock [184] label regions as one of eight possible types, based on the mean and Gaussian curvature values. The curvatures are calculated using the eigenvalues of the Hessian matrix. Additional constraints, specifying which region types

can be adjacent to other types, ensure that the labels are consistent with physical reality. If labels do not satisfy the consistency constraints, which should be changed? The authors solve this by defining label probabilities and employing a relaxation technique. This improves the structure of curvature regions. The location of the region boundaries is generally close, although they may still vary slightly from the corresponding boundaries on the physical surface.

McIvor et al. [114] present methods for identifying simple geometric shapes from 3-D range data. First, the local curvature is estimated at each visible point. Then the signs of the Gaussian and mean curvatures are used to distinguish between planar, spherical, cylindrical, and ruled surfaces. Boundaries of these regions are identified where the sign of the curvature becomes inconsistent with the region shape. This method depends on a curvature calculation method that is robust to noise. Also, some surface shapes, such as a torus, are not included in the basic shapes. One issue was over-segmentation produced by the unsupervised Bayesian classification approach used for the spherical and cylindrical surfaces.

Mangan and Whitaker [111, 112] describe a method for partitioning 3-D surfaces based on watersheds. This is an extension of work applied to segmentation of images. Areas of high total curvature determine the boundaries between regions. Small fluctuations in curvature can cause over-segmentation, which is repaired by merging regions having low watershed depth with one of its neighbors. This method shows sensitivity to both noise and the user-specified watershed depth threshold.

Pulla et al. [142] modify the watershed segmentation scheme of Mangan and Whitaker by considering segmentation using mean curvature, root mean squared curvature, and absolute curvature. In their tests, absolute curvature was preferred, with the fitting methods providing better accuracy than the discrete curvature calculations. The fitting methods also have advantages at boundaries. Unfortunately, the method also requires user selection of a threshold for merging watersheds, which varies from case to case.

Steger [163] proposes a method to compute watersheds and watercourses from digital terrain models with sub-pixel precision. The digital terrain model is represented as a range image. The method is based on extracting the critical points and accurate

principal directions. Smoothing is required to remove noise, and the image is convolved with the partial derivatives of a Gaussian kernel to get the partial derivatives of the image. The eigenvalues of the Hessian matrix are used to classify the critical points and compute principal directions. Integration along the principal directions then locates ridge lines and valley lines. The graph formed from critical points and these ridge and valley lines is used to extract hills and valleys.

The Connolly function is a function for which the extrema represent knobs and depressions which can be matched to model docking of molecular surfaces. Cazals et al. [25] investigate the Connolly function for triangular meshes and construct a discrete Morse-Smale decomposition to segment the surface into regions where the flow of the Connolly function is uniform.

Graph cut algorithms have been used to segment images [20] and medical datasets [19, 89]. They are effective at assigning the vertices of a graph to either a feature (foreground) or background set, based on graph properties such as the gradient of the image intensity. Some of these methods employ an interactive step, where the user identifies feature and background seed points, to guide the algorithm to the objects that are to be separated. We treat our mesh as a graph and apply the graph cut algorithm described in Section 5.3, and identify features based on the resulting segmentation.

Katz [85] segments a mesh into visually meaningful sub-meshes following the minima rule. The minima rule states that boundaries of parts lie along contours of negative curvature minima. This approach also uses mesh coarsening, multi-dimensional scaling, and feature points, and finds a core, min-cut for final region definition.

### 2.2.5 Features

Features are characteristics of objects that can be used to compare one object to another. A feature may be defined at a point or over a region of a surface. Features may simply exist or not, or may have an associated value or vector of values that can be compared numerically. Features are highly case dependent.

The quality of the features has a direct bearing on the ability to solve the correspondence problem. This quality can be affected by the choice of parameter values and used-defined thresholds. Several techniques have been used to either avoid the need to select parameter values and thresholds, or to automate the selection process.

Falcidieno and Spagnuolo [48] describe reasoning to extract curvature-based surface properties such as convex, concave, planar, and saddle shaped regions. Cohn and Hazarika [31] consider qualitative spatial reasoning as a way to describe the world, independent of any quantitative model. They look at ontology (theories about the nature of space), topology, and difficulties representing shape and spatial changes qualitatively.

Interrante et al. [82] enhance the visualization of surfaces by highlighting features such as ridge and valley lines. Because they are concerned primarily with visualization, a stable approximation is preferred to a more complex calculation of the ridge and valley lines. These ridges and valleys are identified from the local minima and local maxima of the first principal curvature.

Ma and Interrante [108] identify key edges by looking at the angle between normals of adjacent triangles. If this angle exceeds a global threshold, or a local threshold based on the angles between other nearby triangle pairs, that edge is used to support the visualization task. This specific definition of features is not so useful for smooth surfaces, but the general concept may still be appropriate for aligning objects.

A volume skeleton tree (VST) [169, 170, 171] is used to partition terrain surfaces and volume datasets based on critical points such as maxima, saddles, and minima. These features are applied to view morphing and transfer function design, and can be based on surface properties such as curvature.

The Scale-Invariant Feature Transform (SIFT) [106, 87] produces a large number of distinctive key points as scale space extrema. These key points are generated from a difference of Gaussian functions at multiple levels. This is a common technique for detecting features in images, but can also be applied to functions on the surface of an object.

## 2.3 Shape Comparison

This section presents a variety of shape comparison methods. Selected methods are listed in Table 2.3. Alt and Guibas [2] and Veltkamp and Hagedoorn [181, 179, 180] survey a variety of methods to compute a quantitative difference between objects.

Several methods compute an alignment between objects directly. A few methods pick the best alignment from the set of all possible alignments. The correspondence does not need to be calculated as part of the alignment process. The method of moments [73] uses global properties, namely the center of mass and the principal axes, to align data sets.

Registration involves determining the relative transformation that will best align two objects. Simple transformations typically include rigid (translation and rotation), or affine (translation, rotation, and shear) transformations. More advanced transformations may be projective, nonlinear elastic, or piecewise. These more sophisticated transformations have the potential to transform one object so that it conforms more closely with the other. This may make the correspondence problem easier, at the expense of a more complex transformation. However, the non-rigid effects of the transformation may also mask the differences between the objects. In order to compute the differences between the objects, we may prefer a rigid transformation to get the best alignment of only the parts of the objects that really do match.

Correspondence is the process of mapping features of one object to features of the other. This is highly dependent on the features identified. Features can be points, global properties, local properties, or vectors of properties.

Registration and correspondence are closely tied in most shape matching methods. Determining how to align *ObjectA* with *ObjectB* depends on the corresponding features in *A* and *B*. Furthermore, which features correspond may be based on the choices that yield the best alignment and match between the objects. Again, the key concept in these methods is combining or alternating between the alignment and correspondence steps. This is a fundamental but powerful observation, which is straightforward to translate into practical applications.

Chui and Rangarajan [29] point out that “solving for either the registration or the correspondence once the other is known is much simpler than solving the original, coupled problem.” We follow their example in distinguishing between methods that primarily deal with registration, correspondence, or combine both.

In order to reduce the size of the correspondence space, some methods group feature points into higher level structures using object parameterization. Fitting curves and/or surfaces to the extracted features produces a simplified correspondence space. However, fitting and feature extraction do not extend easily to the complex general case. This is further complicated if the data includes noise.

The coupling between registration and correspondence makes it very difficult to solve this problem directly. However, alternating between solving for the transformation and solving for the correspondence is a practical approach.

These signatures for shape retrieval are applicable to coarse shape matching, but generally are less discriminating between similar shapes. On the other hand, the use of angle and distance measures and oriented point functions can provide insight into additional feature detection approaches.

A focus on accurate registration and correspondence is required in order to provide detailed comparisons between 3-D objects. This goal of detailed comparison will affect our choice of features and the methods we use to align objects.

Golland [66] uses a distance transform as a shape descriptor, and employs support vector machines (SVMs) with Gaussian kernel functions for learning with small sample size. Cross validation is used to analyze performance.

Gelfand et al. [62] presents an example which develops a coarse matching followed by an ICP alignment step. They propose an integral volume descriptor used to detect feature points which are persistent at multiple scales. A voting scheme is used along with a pairwise distance matrix and rigidity constraints which filter out incorrect correspondences.

One can also check the quality of the match [122] to pick the best threshold. A similar multi-scale approach [6] has been applied to medical image analysis, with

Table 2.3: Shape Comparisons - Points and Models

<i>Shape Comparison</i>			
Examples	Paper	Strength	Weakness
<i>Point Representations</i>			
Hausdorff distance	Huttenlocher et al.(1993)		Sensitive to noise
Iterative closest point (ICP)	Besl & McKay(1992)	Local minimum	Need close starting point
Exhaustive	Huttenlocher & Ullman(1990)	Best match	Expensive
<i>Models</i>			
Thin-plate spline (TPS)	Bookstein(1989) Chui & Rangarajan(2000)	Non-rigid	Requires correspondence to define warp
Statistical	Davies et al.(2001)	Provides correspondence	Requires training set
Active shape	Cootes et al.(1994)	Captures variability	Manual landmark selection
Viscous fluid	Fookes & Maeder(2003)	Recover large mismatch	Assumes similar intensity
Energy minimization	Lai & Fang(1999)	Robust to occlusion	Need to define energy terms
	Shelton(2000)	Mesh simplification to increase efficiency	
<i>Graph Matching</i>			
Fuzzy homomorphism	Perchant & Bloch(1999)	Handling deformation	Global
Edit distance method	Klein et al.(2000) paper	Cost to morph plus	Difficult for 3-D minus
<i>Parameterization</i>			
Spherical	Shum et al.(1996) Asirvatham et al.(2005) Praum & Hoppe(2003)	Consistent over multiple objects Minimize stretch	Genus 0 only User picks landmarks

classification based on the sign of mean and Gaussian curvature. In addition, multi-step algorithms have been used successfully with curvature-based registration, where an initial coarse computation is followed by a second refinement step.

Biber and Strasser [14] propose a step between feature extraction and establishing correspondence, in which each feature is assigned a uniqueness with respect to other features. This enables automatic determination of matching thresholds that minimize the likelihood of incorrect or ambiguous matches. Correspondence is assigned between features only if the distance between the objects is less than the uniqueness of the object relative to other objects in the domain. This method has limited applicability in cases where uniqueness cannot be assumed, for example, where several features may be very similar or even identical.

Another alternative to the uniqueness of features is to compute the saliency of features. Saliency is a representation of the relevance, but does not require that the features be unique. Curvature-based saliency detection has been proposed by building high-level features from local surface descriptors [58], and by creating features from non-trivial local shapes and center-surround filters with Gaussian weighted curvatures [97]. However, in these cases, the features generated tend to be too localized for detailed shape matching.

Li and Guskov [100] use smoothing of a point cloud to generate a scale-space representation. From this multi-scale representation, salient features are computed as the extrema of the change in the normal between adjacent levels. Then a signature is computed as an  $M \times N$  array by sampling a disc around the feature point, projecting normals onto the direction connecting the feature point to the sample point, and applying discrete Cosine and Fourier transforms. This approach yields a suitable number of feature points for shape matching, and has been used to align multiple scans of objects.

Liu and Heidrich [102] looks at the problem of constructing a 3-D model from multiple views of an object. Hardware acceleration is used to efficiently register partial 3-D volumetric models with each other. The method is a variation of the iterative closest point algorithm with the graphics hardware used to evaluate the error between partial models. The off line version of the algorithm can be used to produce a highly accurate



registration. The alignment metric is the sum of absolute pairwise distances in the region where the models overlap, normalized by the number of overlapping pixels.

Maillot et al [109] present one of the first papers to use the concept of an atlas, or collection of parameterizations. They “cut up” the mesh into regions based on curvature. Each of these regions is flattened out using an optimization procedure, with a parameter that allows distortion to avoid flipping. However, this method is interactive, requiring user input to define the atlas structure.

### 2.3.1 Point-based Comparison

There are several methods developed for sets of points. The Bottleneck distance [45] is the minimum of the maximum distances for all possible one-to-one correspondences between two point sets. This requires that the point sets have the same number of points.

The Hausdorff distance [79] finds for each point in one set, the distance to the closest point in the other set, and takes the maximum of these distances. It is susceptible to noise in the point locations. Variations of the Hausdorff distance [7] have attempted to reduce the noise sensitivity. Both of these methods are primarily global rather than local.

The Hough transform [8] divides the transformation parameter space into bins and uses a voting scheme to select a best transformation. By restricting the problem to a finite set of possible alignments, methods such as the Hausdorff distance [79], the alignment method [178], and geometric hashing [79] can try all possible alignments to select a best alignment. These methods can align objects under rigid, affine, and projective transformations, but are not easily extended to more general transformations.

The iterative closest point (ICP) algorithm [12] uses a nearest-neighbor relationship for the correspondence step, and the correspondence is used to refine the transformation. It iterates to converge to a local minimum, but assumptions break down for non-rigid transformations and outliers.

The ICP method requires a reasonably close initial alignment. In a number of techniques, a coarse alignment is generated prior to applying the ICP or a modified ICP algorithm. Planitz [134] looks at a number of methods and proposes a correspondence framework. This framework breaks down correspondence calculation into five tasks: (1) region definition, (2) feature extraction, (3) feature representation, (4) local matching, and (5) global matching. Different techniques can be characterized by how they perform the five tasks, and new approaches can be generated by taking different combinations of the candidate techniques for each task.

Ichimura [81] addresses the issue of tracking feature points from frame to frame in a sequence of frames. A key issue is deciding when new features appear or disappear during a sequence of frames. Feature points are extracted by a corner detector. Normalized cross-correlation is used as the similarity measure. The search region for a feature point is based on the location in the previous frame. In this context, the transformations are generally constrained, such that the solution of the correspondence problem can make use of frame-to-frame similarity and final registrations can be done after the fact, based on the correspondence.

Planitz et al. [135] looks at the problem of automatically generating a correspondence between two 3-D models. They propose a signature based on a local region around select vertices. A distance measure,  $D1$ , is defined from the distances between a selected vertex and other points in the local support region. An angle measure,  $A1$ , is based on the angle between normals at the selected vertex and at other points in the local support region. The signature is a combination of  $D1$  and  $A1$ . First, all potential matches are found by locally matching vertices. Then the number of matches is reduced by checking combinations of matches for geometric consistency. The final correspondence is chosen based on the best alignment of the models. However, the use of distances and angles between normals for points in a local support region makes this method sensitive to point distributions.

Huttenlocher [80] presents an exhaustive method for point sets that uses two (2-D) or three (3-D) reference points in each set to align the point sets to a reference frame. The measure of similarity is the number of points in one set that have a corresponding point in the other set, within some tolerance. After trying all combinations of points, the reference frame with the highest matching score is selected.

Geometric hashing [96] tries to speed up the matching process through the use of preprocessing. Versions of each model based on each possible reference frame are stored in a global hash table using all coordinates as the hash key. Then, to match an object, similar hash table entries are calculated for several reference frames, and hashing into the table produces votes for the (model, frame) pair stored at that hash table entry. This method can just as easily be applied to features extracted from the object.

### 2.3.2 Models and Morphing

Another topic of interest, which is supported by parameterization methods, is the morphing of one shape into another. While such morphing can be applied directly to a point set, it is often more useful if the parameters can be associated with scientific principles. Studies in anthropology often assess the differences in bone or fossil structure to infer information about the subjects. For example Niewoehner infers changes in behavior between Neanderthals and the Skhul/Qafzeh hominids [125], based on differences in hand structure. They generate landmark coordinates photogrammetrically and then perform a Procrustes alignment followed by principle component analysis (PCA) to capture isometric and overall shape variation. The PCA scores for different known classes can be used to develop a classifier that can be applied to unknown classes, such as the Skhul/Qafzeh hominids, to determine where they fall relative to the known classes. The constrained spherical parameterizations above [5] can also be combined with PCA, so that morphs can be generated from different weights on the principle components.

Geometric morphometrics represents objects in shape spaces, and performs a statistical analysis of the transformation between these shape spaces. This approach has been used to model and visualize the a current evolutionary hypothesis concerning the shape changes over time between extant species and their ancestors [183]. Similar to the approach above, the method relies on a set of manually generated landmark points. The associated parameters represent placement on an evolutionary tree.

All of the above methods still rely on manually generated landmark points, although some tools are provided to assist the user [183]. A more automatic approach is to

minimize an energy function that represents the quality of the correspondence between objects [158]. Given correspondence between the objects, a model can be built as combinations of corresponding points. This automation of the point correspondence still leaves us with limited meaning associated with the shape parameters.

Some representations define the boundary of an object by splines, planes, or other analytic functions. Turk and O'Brien [177] model the transition of morphing of one object into another as an implicit function in one higher dimension. They generate this function using radial basis functions to fit between the two surface representations modeled as scattered data points from the surface of each object. This higher dimensional surface, or properties derived from it, may be useful to describe the changes in the object as it transforms from one shape to another. However, this function still does not take into account the correspondence between features, and is not based on any specific information about how one object should deform into the other.

Bookstein [18] decomposes the deformation of a set of landmark points into bending modes, called principal warps, of a thin-plate spline interpolated through the points. The modes are derived from the eigenvectors of the bending energy matrix. These warps can be used as features for comparing objects, for describing specific deformations, or to align a specific object to a standard atlas. Bookstein's work is particularly interesting because decomposing deformations is closely related to describing the differences between similar objects.

Chui and Rangarajan [28] use a thin-plate spline to generate a non-rigid mapping for 3-D brain MRI sulcal point matching. The approach alternates between solving the correspondence problem and the registration problem. The *softassign* technique is used to relax the binary correspondence to a continuous valued matrix, so that the correspondences do not approach binary values until the transformation begins to converge. In addition, deterministic annealing adds an additional entropy term that is gradually reduced as the minimization process proceeds. They refer to the general point matching algorithm as robust point matching (RPM). One strength of this method is that it is relatively insensitive to points that are outliers.

Another approach to shape comparison is through modeling. Davies et al. [37, 38] construct a statistical shape model that defines modes of variation. This shape model

encodes correspondence acquired from parameterization of a training set, and is represented by a minimum description length. Cootes et al. [33, 32] define active shape models which capture natural variability of an object. Points are placed on the training set manually, and then principal component analysis (PCA) is used to find independent parameters and principal modes of variation. These active shape models are applied to medical datasets.

Styner et al. [165] evaluate four statistical models for analyzing anatomical objects from medical datasets with manually selected landmarks. The evaluation criteria included generalization outside the training set, compactness, and specificity. The spherical harmonics method, tied to a Procrustes alignment, was the least accurate, while the minimum description length and determinant of the covariance matrix methods performed best.

Haker et al. [68] propose an elastic registration method based on optimal mass transport. They apply the method to a 3-D brain deformation sequence and to surface warping of a colon surface. The 3-D deformation case uses voxel data. The colon surface case is flattened to a plane using a conformal mapping technique. An area correction is applied to the initial mapping to preserve the size of surface structures. This technique is parameter free and optimizes mass transport, as calculated from the movement of the surface, weighted by an assumed density.

Fookes and Maeder [56] propose a hybrid non-rigid registration scheme combining the viscous fluid algorithm with mutual information (MI). The viscous fluid algorithm has been used to recover large local mis-registrations between two images, but assumes similar intensity values between images. This method produces results similar to the thin-plate spline warp, but is not as good as Gaussian convolution.

The Monge-Kantorovich Metric [144], also known as the transport metric or earth-mover's distance, measures a cost needed to transform  $A$  into  $B$ . This cost is related to the energy necessary to move surface elements represented by a corresponding mass.

Other methods assign attributes to points based on the way they can move. Sclaroff and Pentland [151] develop a modal matching method based on a mass and stiffness

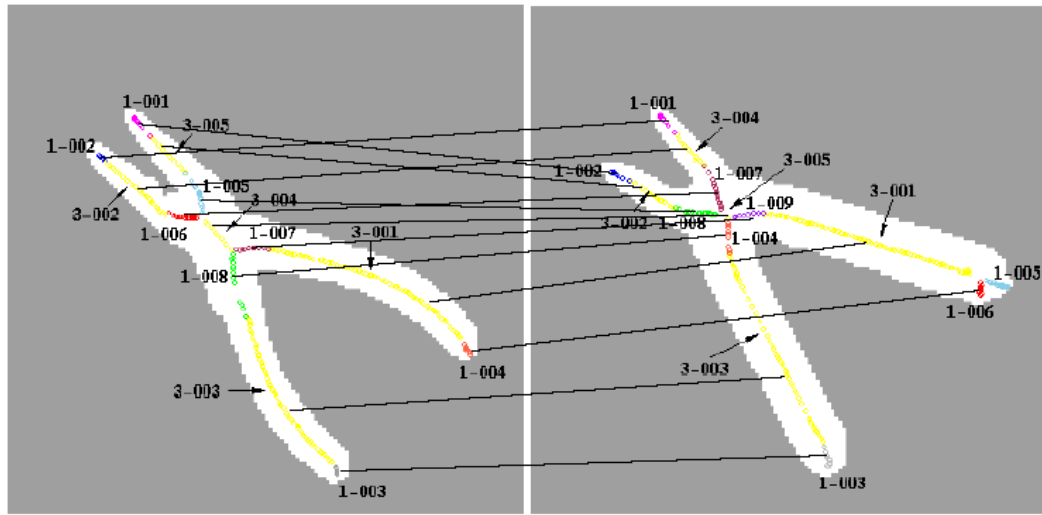


Figure 2.5: Correspondence between two shock graphs. From “Shock Graphs and Shape Matching”, Siddiqi et al., ICCV 1998.

matrix. Correspondence is determined by each point’s participation in the eigen-modes calculated from the decoupled dynamic equilibrium equation. The main issues with these methods is intolerance of outliers and accuracy limitations from just comparing eigen-modes.

### 2.3.3 Graph Matching

Graph matching is another tool frequently applied to object recognition. Labeled graph matching represents a pattern by a graph where nodes are labeled with feature information and links indicate topological relationships between features. Features may be based on invariants or other properties such as geodesic distances. More complex features may be represented as a vector of features or using a multi-resolution approach.

One challenging topic of research in medical imaging is brain imaging. The complexity of the brain structure (see Figure 2.6) and the availability of MRI data make this a popular area of research. One of the key problems is atlas-based labeling of a brain MRI. Perchant and Bloch [132, 133] apply fuzzy graph homomorphism to the brain

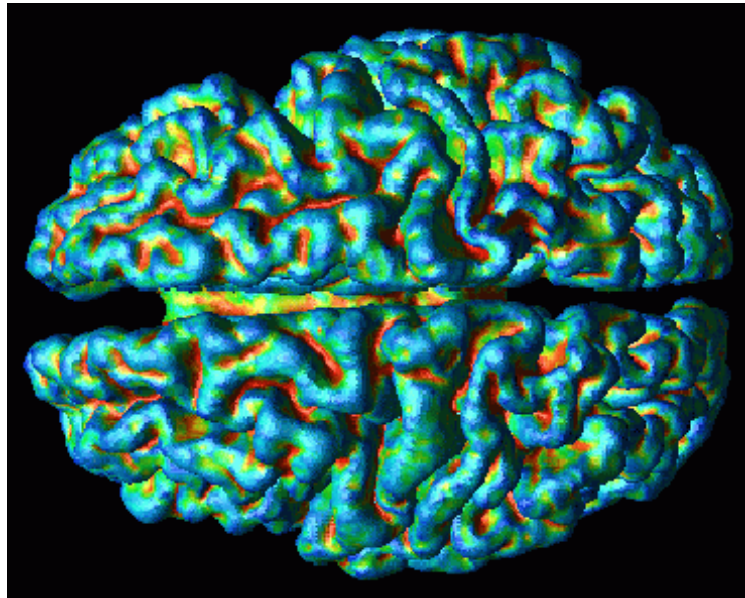


Figure 2.6: Brain surface segmentation. From “Segmentation, Modeling, and Registration - Introduction to Computer-Integrated Surgery” by Russell Taylor.

structure recognition problem. They note that in order to capture all the boundaries of the anatomical brain structures, when comparing segmentation of a brain image with segmentation of a reference brain atlas, over-segmentation is required. As a result, they introduce “graph fuzzy morphism” which relaxes the notion of a binary relation between nodes to a fuzzy relation based on the degree of correspondence between corresponding nodes.

Recent methods have used entropy-based measures to align images, rather than trying to identify specific correspondences. Neemuchwala et al. [123] investigate a graph matching scheme using higher dimensional image component analysis (ICA) feature vectors and a minimal graph entropy estimator. The method is applied to the registration of a pair of ultrasound images.

Klein et al. [91, 90] compares 2-D shapes by computing shock graphs for each shape, and then computing a cost to convert from one shock graph to the other. The shock graph is the medial axis of the shape, along with information at each point about the distance to the boundary. The shock graphs for two fish are shown in Figure 2.7. Note the differences in the structure of these graphs. The edit-distance, which represents

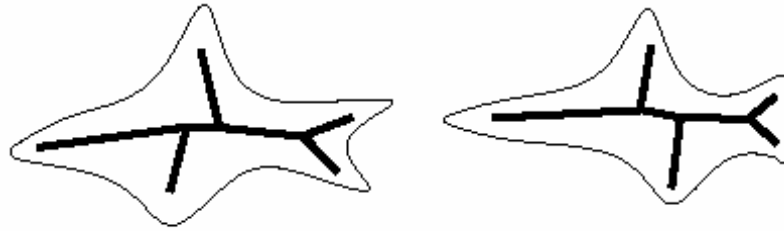


Figure 2.7: Shock graphs for two fish. From “Shape matching using edit-distance: an implementation”, Klein et al., Symposium on Discrete Algorithms, 2001.

the cost, is computed using dynamic programming from the costs to deform edges of one shock graph into any edge of another shock graph.

In addition to the cost, matching algorithms can also find corresponding edges of the graphs. The correspondence between the graphs for two tools is shown in Figure 2.5. However, this is most easily accomplished if the graphs have the same structure. Another approach is inexact weighted graph matching. Shapiro and Haralick [154] use the spatial relationships between points to constrain the search for correspondences between point sets.

Bengoetxea et al [10] develop a graph matching algorithm using a stochastic heuristic search strategy called estimation of distribution algorithms, or EDAs. In applications to the human brain, node attributes were the gray level (intensity) and region size in the image, and edge attributes indicate spatial relationships such as relative position and distance.

Gold and Rangarajan [63] develop a “graduated assignment” algorithm for the problem of weighted graph matching. This algorithm uses a permutation matrix to identify the correspondences between graphs, but relaxes the constraints on this matrix to allow real values, representing probabilities of particular correspondences. As the solution progresses, changes to a control parameter force the probabilities to approach integer values, representing true permutations.

For solving the subgraph isomorphism problem, Eppstein [47] looks at algorithms and their time complexity. He identifies improvements that can be achieved when the



graphs are represented as tree structures and the tree-width of the subgraphs can be limited.

### 2.3.4 Energy Functions

Lai and Fang [95] formulate image alignment as an energy minimization problem to estimate an affine transformation. The method is intensity-based and the energy function is a weighted sum of squared differences. An iterative re-weighted least-square approach is used to minimize the energy function using a robust  $\rho$ -function. Robust estimation allows this method to be robust to partial occlusion.

Shelton [158] also uses an energy function to represent alignment quality. The method includes terms for similarity, structure (directional springs), and prior information. Weights on the terms vary during the iterative solution. Mesh simplification is used to increase efficiency.

### 2.3.5 Template Matching

Template matching applied to images entails classifying samples of the image by comparing them to known templates. Cox [35] discusses template matching in the context of detecting an  $n \times n$  sub-image  $g$  within an  $N \times N$  search area  $s$  that best matches an  $n \times n$  template  $f$ . The key issue is determining the similarity (or dissimilarity) between two  $n \times n$  sub-images. He presents several measures including correlation measures such as normalized cross-correlation, intensity difference measures such as root mean square distance and sum of the absolute valued differences, sign change criteria, and distortion measures such as weighted squared error. He also formulates similarity detection as a filtering process, and proposes filters aimed at optimizing different signal-to-noise ratios. Cox also presents techniques such as fast fourier transforms and different template matching schemes to improve efficiency.

De Souza and Montenegro [162] describe two methods developed to align noisy images of protein structures, one based on similarity and one using templates. The

similarity-based method uses an exhaustive search after creating a discrete orientation space. After aligning the centers of mass, the best match is chosen by testing all possible rotations within this orientation space. For the template-based method, edges are first extracted from the image. Then versions of the template for all possible orientations and sizes are compared with the image, choosing the best match between the template and the image edges. Efficiency is achieved by creating templates in different orientations, and by comparing the edges of the template instead of the entire template. This allows the template-based approach to run faster than the similarity-based method.

Sebe and Chen [152] develop a one-dimensional method for template matching. A 2-D template matching using the sum of square differences and an affine motion model is slow due to the size of the least-squares problem. As an alternative, they propose summing rows and columns to reduce complexity. This produces a new template which is  $2N$  instead of  $N^2$ . However, it is also an approximation of the original template image. This method is applied to the real-time tracking problem and compared to corner tracking and a standard 2-D template matching method.

Template matching is most appropriate when there is prior knowledge of the object being matched. Unfortunately, this is not the case for general shape comparison. In addition, representing the possible deformations becomes prohibitive, limiting this method's usefulness for non-rigid transformations.

### 2.3.6 Parameterization

A surface parameterization defines a mapping from points on the surface of an object to a canonical shape, such as a plane, sphere, or  $N$ -holed tori. For each surface point mapped to a plane, the 2-D coordinates in the plane become the parametric coordinates for the surface. The mapping from the surface to the plane forms a parameter space.

Closed surfaces cannot be mapped to a plane without cutting. However, the spherical topology of genus zero surfaces naturally maps to a spherical, and higher genus surfaces can be mapped to  $N$ -holed tori. Except for the torus, closed surfaces do not

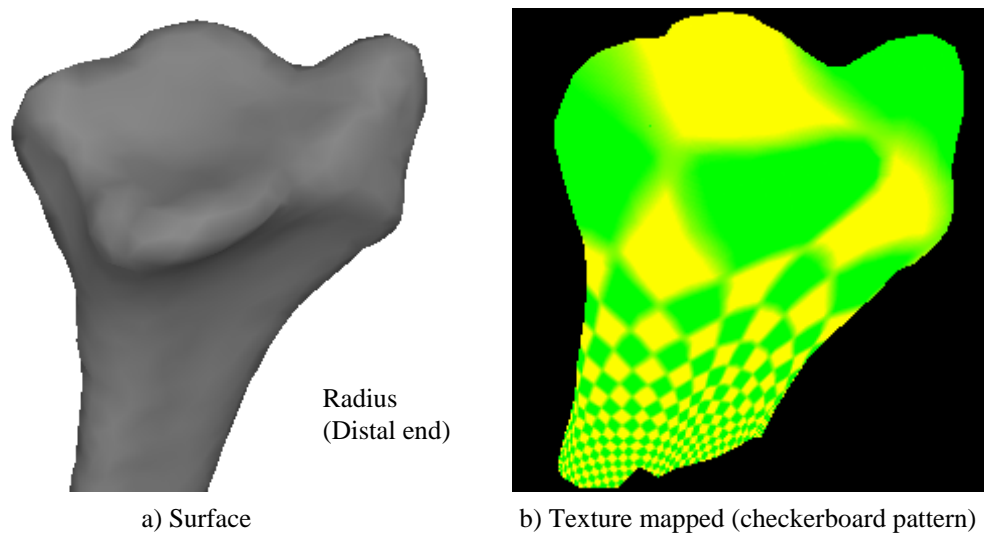


Figure 2.8: Texture mapped bone surface. (a) Surface. (b) Checkerboard texture map applied to the surface.

admit a global 2-D parameterization. These surfaces may be segmented into multiple regions, each of which can have a local mapping to the plane. This mapping can be generalized using a manifold, where multiple charts, each having its own 2-D parameterization, are laid out to cover the surface of the canonical shape. The mapping through the canonical shape then provides a parameterization of the original surface.

Parameterizations are useful for distributing and interpolating data on a surface. For example, one might measure the pressure at several locations on the surface of the heart, then interpolate (using the parameter space to determine nearby points) to get data at other locations on the surface of the heart.

Another major application of parameterization is texture mapping, where the goal is to map 2-D images or textures onto 3-D surfaces without introducing visually displeasing distortion. Figure 2.8a shows an example of a radius (distal end), and Figure 2.8b illustrates texture mapping applied to this bone. Note the distortion of the areas of the squares mapped to the surface.

Only in rare cases can a mapping be generated that preserves both relative distances between points on the surface, and the angles between intersecting lines on the surface. Therefore a parameterization introduces distortion. A more severe issue is folding.

Folding occurs when different points on the surface map to the same points in the plane, and can make the parameterization useless. A number of techniques have been developed to try to eliminate folding and minimize distortion.

Techniques for parameterizing meshes range from ad-hoc to fairly well principled. Most parameterization techniques attempt to create a mapping from the surface to the canonical shape that maintains some property, such as area or angles. They may also place constraints on the input data, for example requiring that the object be convex, to ensure that the parameterization does not fold.

In this work, we use techniques that flatten local regions, which are homeomorphic to a disc, onto the plane, as well as techniques that map genus zero surfaces to a sphere.

### **Mapping 3-D Surfaces to 2-D**

The simplest parameterization technique is to find a suitable plane, and project each point onto the plane in the direction normal to the plane. A pair of orthogonal directions in the plane can be chosen arbitrarily to define parametric coordinates in the plane. Often, the plane is defined to be tangent to the surface at some point, or may represent a least squares fit to the points on the surface. This may be acceptable for parameterizing local regions or surfaces that are nearly planar, but is generally unsuitable for most surfaces. The more curved the surface, the more distortion is introduced, and often, folding cannot be avoided.

Floater [54] presents a technique for mapping a disk of triangles to the plane, using a linear least-squares solution. The boundary of the mesh is mapped to the boundary of the desired parameterization (usually a square). Each interior vertex tries to map to the centroid of its neighbors. This mapping is guaranteed not to fold, if the boundary is convex. He extends this work to meshes with  $n$ -sided polygons [52] (as opposed to triangles), and to data points without a mesh [53, 51]. Once the points are projected onto the plane, a Delaunay triangulation yields a triangular mesh, if desired. Techniques for ordering the boundary and picking weights for the interior vertices are also presented. Unfortunately, the use of a prescribed boundary shape

for the projection to the parametric space may not be consistent with shape of the 3-D region being parameterized.

Levy et al [98] develop a technique for producing a conformal mapping of a set of polygons. No guarantees are given concerning folding. This is very similar to Desbrun’s [39] technique for conformal mapping, but more expensive to compute. It also contains methods to split up the mesh into sections and arrange them for use in texture mapping.

Praun and Hoppe [140] use a regular polyhedron as the domain and map the mesh to this regular polyhedron, minimizing a measure of conformality plus stretch.

Praun et al [141] present an approach to “lining up” two parameterizations of two different models. It requires user intervention to identify the corresponding points in different meshes at the coarsest level. A conjugate gradient approach is used to solve a linear system with weights based on Floater’s method [54]. This approach is hierarchical, lining up the lowest level of the mapping, then the next level, and so on.

Sheffer and de Sturler [155, 156] propose an angle-based flattening approach. The basic idea is that every interior vertex of a flattened mesh must have a sum of angles equal to  $2\pi$ . The flattened mesh does not fold (although it may overlap). To improve on the area distortion, Sheffer and de Sturler [157] use Laplacian flow on a grid to alter the initial parameterization. This reduces size distortion while relaxing the angle distortion requirement. The fundamental concept is to lay a grid across the parameterized mesh and grow or shrink the edges in proportion to the ratio of the physical area to the flattened area of the triangles within the grid cell.

Desbrun et al [39] develop two mappings, one based on angle preservation and one based on area preservation, and propose a linear combination of these two mappings to trade-off angle versus area distortion. The method can use a specified boundary, or compute the boundary (given a specified orientation and scale) as part of the solution process. The flattened mesh tends not to fold in the angle preserving case, although no guarantees are given. Figure 2.1b shows the flattened mesh for the bone from Figure 2.1a.

Meyer et al. [116] extend the concept of barycentric coordinates to irregular N-sided polygons. This could allow for the extension of parameterization methods for triangular meshes to general polygonal meshes, but in this dissertation we will concentrate on triangular meshes.

The approaches taken by Sheffer and de Sturler, and Desbrun et al. are most promising because they do not use a prescribed boundary, and tend to minimize the distortion of the angles, with options to balance angle optimization with area optimization. Initial indications show that minimizing area distortion is important for viewing the relationships between features, and may also be important for detecting features, in order to improve the correspondence based on the area associated with different features.

## Surface Parameterization

As surfaces become topologically complex, a single parameterization gives way to different representations for local surface regions. A manifold surface model is locally parameterized by charts that overlap at their boundaries. These charts preserve useful properties, such as continuity and smoothness, across regions [67].

Instead of explicitly mapping surface points to a plane, a parameterization can be applied directly to the surface of the object. For example, families of geodesic curves placed on a surface can be used to define a parameterization. First, one geodesic curve is created on the surface of the object. This is analogous to taking a marker and drawing a curve on the surface, making it as straight as possible. Then construct another geodesic curve on the surface that crosses the first curve and is perpendicular at their intersection point. Finally, construct a family of geodesic curves which cross the first curve and are approximately parallel to the second curve but successively offset by a given distance. Then for any point in the region covered by the curves, the parametric coordinates are defined based on the cross curves it is between, and the (signed) distance from the initial curve. Geodesics can also be used to define a local polar parameterization [186].

An approach that simplifies the parameterization of complex objects is to first map the object to a reference object of the same genus, and then use a standard pre-defined parameterization approach for that reference object. This is discussed in the next section for genus zero surfaces mapped to a sphere.

## Spherical Parameterization

Several techniques have been used to map genus zero objects onto a sphere. Brechbühler [21] uses a continuous one-to-one mapping to the unit sphere. Constrained optimization is used to improve the uniformity of the parameterization. The parameterization enables generation of a series of spherical harmonic functions. By first determining a canonical position, this produces an invariant shape description. Quicken [143] presents a similar technique, with the addition of a multi-resolution approach to handle large meshes. Quaternions [77] can be used to describe the absolute orientations on the sphere.

Praun and Hoppe [140] employ a sequential approach using a spherical parameterization as an intermediate step in mapping from a mesh to a flat image. The mesh is first mapped to a sphere, and then the sphere is mapped to a tetrahedron, cube, or octahedron. Each of these regular polyhedra can be unfolded into the plane by cutting along select edges. A metric is presented that measures the stretching required to transform from the mesh to the sphere, and from the sphere to the regular polyhedra. Asirvatham et al. [5] extend this approach by manually defining corresponding feature points on multiple meshes, and creating the spherical parameterizations that align the parametric locations of these points. Constraining these feature points to map to the same locations on the spherical parameterization generates a parametric alignment of multiple objects.

Shum et al. [159] use the  $L_p$  distance between the local curvature functions of two 3-D surfaces. The curvature functions are mapped to a special mesh, which is the dual of a semi-regular triangulation of the unit sphere. This technique is only applicable to surfaces that are topologically spherical.

Eck et al [44] discusses more than just parameterization, but they do introduce a harmonic map parameterization as a sub-problem. Hence, this has become the paper to cite for a harmonic mapping of a mesh to the plane. This is essentially an edge-based method, which tries to minimize the square of the norm of the gradient of change in the surface parameters.

## 2.4 Chapter Summary

One can see that there are numerous approaches discussed in the literature, but none of them adequately address our shape matching problem. For our work, we assume that the objects to be compared are represented by a mesh consisting only of triangles, i.e., a triangulation. Our approach makes use of several of the techniques presented, particularly, the graph cut method of Boykov and Kolmogorov [20], and the flattening of Desbrun et al. [39] and Sheffer and De Sturler [157]. Our similarity measure can be represented as a variation of the geodesic fans [186], and exhibits properties similar to the method of Gelfand [62]. We also rely on spherical parameterization similar to Brechbühler et al. [21].



# Chapter 3

## Curvature Calculation on Meshes

As mentioned in previous chapters, *curvature* is a surface property with potential uses in object comparison. This chapter presents a new technique for evaluating the impact of mesh effects on various curvature calculation methods. This technique is used to evaluate several existing curvature calculation methods along with a few new variations. As will be seen, each curvature calculation method responds differently to factors such as noise in the mesh, irregularities in the triangulation, and overall resolution.

### 3.1 Overview

Curvature is an intrinsic property of a surface and can be calculated by a variety of techniques. Curvature metrics include scalar properties such as maximum and minimum principal curvatures, mean and Gaussian curvatures, and vector quantities such as principal curvature directions.

Several decomposition methods use surface curvature properties directly to identify features such as ridges and valleys, and planar, convex, concave, or saddle shapes [184, 176, 48]. Surfaces are segmented into regions [182, 6] based on these curvature features, and the segments and features are then used for object recognition and registration. A measure of total curvature can be used to distinguish flat regions from regions of small curvature. There are also other feature detection and registration methods and shape signatures which utilize surface curvature.

Meshes support wide variations in complexity and resolution for local regions of an object. They use a relatively simple representation consisting of vertices (points sampled from the surface), and polygonal faces defining connectivity between vertices. Today's visualization tools are extremely compatible with this mesh data structure. However, tools for extracting surface properties from meshes, for example, smoothness, have not yet progressed to match the state-of-the-art for more traditional representations such as those used in the Computer-Aided Design (CAD) environment.

The ability to compute curvature from meshes is complicated by the lack of an analytic definition for the surface shape. Meshes are defined at discrete vertices, while curvature is a function of how the surface behaves in a local region around the vertex. This is evident since curvature is based upon 1<sup>st</sup> and 2<sup>nd</sup> derivatives, which are themselves defined as a limit function. Thus, some assumptions on the behavior of the surface are required to estimate curvature for a localized set of vertices, such as a given vertex and its neighbors.

Past experience indicates that curvature metrics tend to be very sensitive to noise [83, 71]. Scanners and sensors typically introduce some noise into the data. Small amounts of noise may be compensated for by smoothing, while large amounts may render the data unusable. Besides noise, the mesh resolution, i.e., how finely the surface is sampled, and regularity, i.e., the uniformity in size and shape of the mesh faces, also affect the accuracy of curvature estimates.

Curvature calculation methods applicable to triangular meshes fall into one of three categories: (1) fitting methods, (2) discrete estimation of curvature and curvature directions, and (3) estimation of a curvature tensor from which curvature and curvature directions can be found. We have developed a process for evaluating the accuracy and stability of such methods using a suite of test cases that highlight the effects of mesh properties in addition to noise. These mesh properties include factors such as valence (the number of vertices adjacent to a given vertex) and the regularity of the mesh. This suite is applied to several existing algorithms to examine how reliably different algorithms predict the curvature values. This evaluation process compares the error in mean, Gaussian, and principal curvatures, and the normal and principal curvature directions.

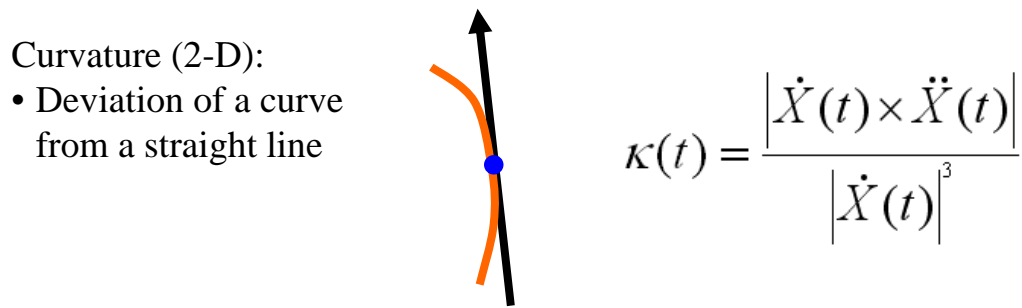


Figure 3.1: Definition of curvature in two-dimensions.

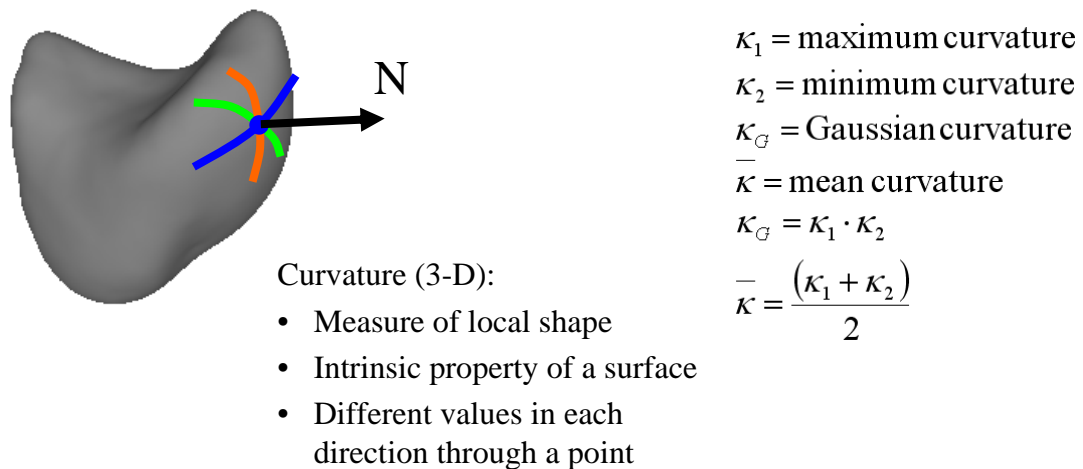


Figure 3.2: Definition of curvature in three-dimensions.

Knowledge of the accuracy and sources of error allows selection of algorithms that are robust and reliable for tasks such as shape matching and registration. An understanding of the errors in the curvature calculations can be combined with techniques from the Bayesian community to add confidence levels to the data, and to develop an understanding of when and why a method might break down.

### 3.1.1 Definitions

The curvature at a point on a planar curve is the inverse of the radius of curvature of the osculating circle at that point. An illustration and alternate definition for curvature of a planar curve is shown in Figure 3.1. For a point on a surface, there are an infinite number of planes that contain the normal to the surface. The intersection of each of these planes with the surface will form a planar curve through the point. Three sample curves are shown in Figure 3.2. For the non-degenerate case, there will be some curve where the curvature at the point is a minimum, and some curve where the curvature will be a maximum. These curves will be orthogonal. The magnitudes of these extreme curvatures are the principle curvatures,  $k_1$  and  $k_2$ , and the tangents to these curves at the point are called the principal curvature directions. Two additional quantities of interest are the Gaussian curvature and the mean curvature. The Gaussian curvature is defined as the product of the principal curvatures,  $k_1 \times k_2$ , while the mean curvature is the average of the principal curvatures,  $(k_1 + k_2)/2$ . The Gaussian curvature indicates the amount of deformation of the surface required to flatten the surface onto a plane, and is an intrinsic property of the surface. For a more formal definition of curvature, see do Carmo [41].

For smooth surfaces, the Gauss-Bonnet theorem and Critical Point theorem relate geometric surface properties to the surface topology, while the Theorema Egregium of Gauss relates the intrinsic and extrinsic curvatures. Banchoff [9] proves the analogues to these theorems for the polyhedral surface case. The Gauss-Bonnet theorem for polyhedral surfaces, which expresses the total Gaussian curvature over a region of a surface in terms of the properties at the boundary of the region, is the basis for several of the curvature estimation methods applied to meshes. Brehm and Kühnel [22] demonstrate the approximation of polyhedral surfaces by smooth surfaces, such that the smooth and polyhedral surfaces have the same topology, and the curvature and absolute curvature of the smooth approximation converge to that of the polyhedral surface.

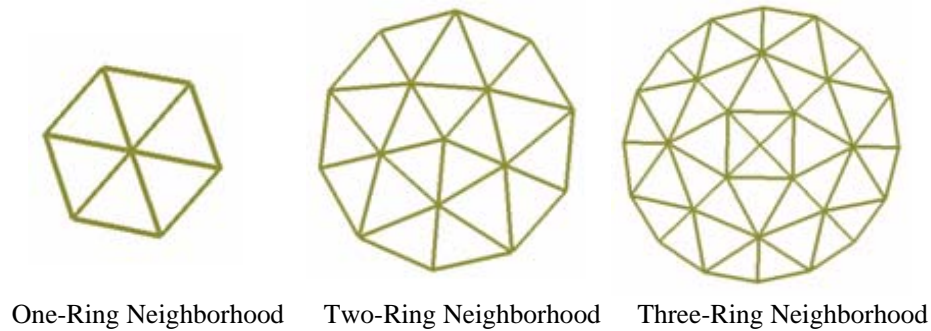


Figure 3.3: Sample test case meshes. Left: 1-ring neighborhood (valence=6), Middle: 2-ring neighborhood (valence=5), Right: 3-ring neighborhood (valence=4).

### 3.1.2 Related Work

A number of researchers [55, 149, 1, 27, 75, 70, 42] have looked at curvature estimation from 3-D range images for computer vision applications. Range data provides a rectangular array of sample data, usually in the form of pixels. Many of the methods operate on an  $N \times N$  window centered at a point, where  $N$  is an odd integer, typically 5 or 7. This window provides a natural orthogonal parameterization and well-defined diagonals. Mean and Gaussian curvature can be computed from first and second partial derivatives with respect to these preferred directions, or directly from the array of sample data. Methods that rely on this regular organization of data are not directly applicable to a general mesh.

Curvature estimation methods have also been developed specifically for meshes. Meshes have a more general structure than range images. Mesh representations have adjacency information embedded in the mesh connectivity, but without any regular organization or preferred direction.

Two vertices are defined to be neighbors if there any face that contains both vertices. All of the vertices that are neighbors to, i.e., share a common face with, a given vertex constitute its one-ring neighborhood. This is extended to a two-ring neighborhood by adding all of the neighbors of the one-ring vertices, and so on. Sample one-ring, two-ring, and three-ring meshes are shown in Figure 3.3. A given vertex of the mesh

can have an arbitrary number of neighbors. These vertices need not be equidistant from the given vertex or equally spaced around the one-ring neighborhood.

Methods for 3-D range images that rely on the regular array structure, natural orthogonal parameterization, or preferred directions, are not readily adapted to mesh representations. However, methods that rely primarily on adjacency, such as surface fitting, may be adapted to mesh representations if a suitable set of vertices can be found. This set of vertices is typically an  $N$ -ring neighborhood, where commonly  $N = 1$ .

Past evaluations have compared specific methods, generally for very regular meshes, and have looked at the effect of noise and the benefits of smoothing [173, 117, 65]. The impact of other mesh factors has often been ignored. A few studies have compared a selection of curvature estimation methods, for example on range images [164, 113], or meshes [168, 65]. Others have focused on a particular method and then varied factors such as mesh resolution [173, 24], the amount of noise added to the data [65], or the shape of the surface [69]. Most of these studies evaluate these methods for a very regular mesh. A few studies [175, 65, 145] also apply the methods to irregular meshes, but do not address the impact of the mesh irregularity.

A few papers have performed theoretical evaluations or experimental comparisons of selected curvature estimation methods. Meek and Walton [115] perform asymptotic analysis for several methods using both regular data (as in range data) and irregular data (as in meshes). The asymptotic behavior is important to insure that the methods would converge to the correct value, but as they state, the results may not be suitable for comparing different methods for fixed size meshes. While this asymptotic analysis was applied only to discretization and interpolation methods, Cazals and Pouget [24] note that ‘interpolation fitting is always more ill-conditioned than approximation’, so one might expect similar results for approximation techniques, such as least-squares fitting methods.

McIvor and Valkenburg [113], in comparing fitting methods for range data, note that there is bias in the curvature estimates since cylindrical and spherical patches cannot be represented exactly by a quadric. They also observe that for quadric fitting of surfaces with large curvature magnitude or with large sampling noise, the eigenvector

associated with the surface normal may not have the smallest corresponding eigenvalue, causing the principal curvature and curvature direction estimation to break down. Overall, their results show the quadric fitting method performs better than the finite difference methods. Surazhsky et al. [168] compare several curvature estimation methods for meshes and conclude that the Gauss-Bonnet scheme (angle deficit) provides the best Gaussian curvature estimate, and paraboloid (quadric) fitting is best for mean curvature estimations and second best for Gaussian curvature.

The focus of this work is on computation of surface curvature on polyhedral surfaces, specifically triangular meshes. Polthier and Schmieß [137] have looked at geodesic curves on polyhedral surfaces, and define a geodesic curvature for curves on these surfaces. Minimizing this geodesic curvature of the curve yields the straightest geodesics on polyhedral surfaces.

## 3.2 Curvature Estimation

This section describes the methods that have been developed to calculate curvature on meshes. There are three basic approaches. The first is surface fitting, which involves finding an analytic function that fits the mesh locally. The curvature of the analytic function is well-defined [78]. The second approach creates discrete curvature equations from the continuous equations by approximating integrals as a summation of contributions attributed to each face or edge adjacent to a vertex. The third approach develops an approximation of the curvature tensor, from which curvature and curvature directions can be calculated.

### 3.2.1 Fitting Methods

The primary discriminator between fitting methods is the function chosen to model the local surface shape. Functions may be parametric, requiring a local parameterization of the surface near each vertex, or implicit. The chosen function is fit separately at each vertex of the mesh, with the method solving for the coefficients of the function.

Table 3.1: Curvature Calculation Taxonomy - Fitting Methods

<i>Curvature Calculation Taxonomy - Fitting Methods</i>							
Fit	Param	Paper	Data	Gauss	Mean	Princ Crv	Crv Dir Req Norm
<i>Range image methods</i>							
Quadric	Grid	Flynn & Jain(1989)	NxN	X	X		
		Abdelmalek(1989)	NxN	X	X		
		Stokely & Wu(1992)	Voxels	X	X		
		McIvor & Valkenburg(1997)	Voxels	X	X		X
<i>Mesh methods</i>							
Quadric	Planar Proj.	Hamann(1993)	1-Ring			X	X
		Meek & Walton(2000)	1-Ring	X	X		
		Goldfeather & Interrante(2004)	1-Ring			X	X
		Gatzke & Grimm(2003)	N-Ring			X	X
Quadric	Natural	Gatzke & Grimm(2003)	N-Ring			X	
Cubic	Planar	Goldfeather & Interrante(2004)	1-Ring			X	X
Arb. Order	Planar	Cazals & Pouget(2003)	N Pts			X	X
Conic	Implicit	Douros & Buxton(2002)	N Pts	X	X	X	X
Radial Basis	Natural	Gatzke & Grimm(2003)	N-Ring			X	X



A local 3D coordinate frame, centered at the vertex, is useful for parameterization, and may also simplify estimation when using implicit functions.

The second discriminator is the number of vertices fit by the function. If too few vertices are fit, the problem is under-constrained. Therefore, a minimum number of vertices, based on the number of function coefficients, should be supplied. Fitting the minimum number of vertices defines an interpolating function where the function goes through each vertex. Fitting more than the minimum number of vertices leads to an approximating function, which minimizes some measure of distance from the function to the vertices, for example, a least-squares minimization. Fitting methods are listed in Table 3.1. This table indicates the type of data on which the algorithms operate, the parameterization method used, whether they require surface normals as input, and whether they compute Gaussian, mean, or principal curvature estimates, or principal curvature directions.

### **Parameterization and Local Coordinates**

Many parameterization methods utilize a local 3D coordinate frame with its origin at the vertex. The normal vector at the vertex is frequently chosen as one axis of this frame. The vertex normal can be computed as the average of the face normals for the faces adjacent to the vertex, with various weightings applied, or as the normal to the plane that best fits the vertex and some number of nearby vertices. For methods that fit a surface to the data near the vertex, the normal can be replaced with the normal calculated from the surface fit. A local coordinate system is formed by the normal vector and two arbitrary orthogonal axes in a plane perpendicular to this vector. Transforming to such a local coordinate system does not restrict the curvature calculation, but does simplify the solution of the equations defining the surface representation.

One class of fitting methods represents the surface as a function of two parametric variables  $u$  and  $v$  in the form:

$$F(u, v) = (x(u, v), y(u, v), z(u, v))$$

The simplest representation is a height function, also referred to as a Monge patch. The height function is oriented relative to the local tangent plane, so that

$$F(u, v) = (u, v, f(u, v))$$

The parametric coordinates of the vertices are found by projecting the vertices onto the tangent plane. This projection can cause distortion in the relative distances between points, and the projection of complex regions can even cause folding. As an alternative, a mapping is computed that transforms the vertices to the plane while minimizing some measure of distortion. Several algorithms [54, 39, 156] have been developed to generate such mappings for a mesh that better preserve relationships and avoid folding.

### Quadric Fitting

A popular choice for  $f(u, v)$  is a quadratic function. Various forms of quadratic function have been fitted to range data [55, 1, 164, 113] and to mesh representations [69, 115, 168]. For a general second-order polynomial with six coefficients, applied to a height function, the equation is:

$$z_i = f(u_i, v_i) = Au_i^2 + Bu_iv_i + Cv_i^2 + Du_i + Ev_i + F$$

where  $(u_i, v_i)$  is the parametric location of the  $i^{th}$  point in the tangent plane, and  $z_i$  is the height of the point above (or below) the tangent plane. Here,  $i$  runs from 1 to  $N$ , where  $N$  is the number of vertices being fit. The coefficients  $A$  through  $F$  are determined by solving a least-squares [139, 50] problem. Two factors distinguish variations of this approach. First, the constant term, or the constant and linear terms, can be dropped. Dropping the constant term forces the fit to go through the vertex, while dropping the linear terms forces the normal to line up with the  $z$  axis of the local reference frame. The second factor is the number of vertices to include in the least-squares fit. One approach is to use just the vertices of the one-ring neighborhood. Alternatively, the neighborhood can be expanded to include a specified number of vertices in the least-squares fit. This larger number of vertices may be required based on the number of coefficients or to improve the stability of the solution. Cazals and

Pouget [24] extend fitting to differential quantities of arbitrary order, using higher degree truncated Taylor expansions, called osculating jets.

### **Cubic Fitting with Normals**

Goldfeather et al. [65] expand the quadric method by using a cubic fit of a system of equations formed from the coordinates and normal vectors at vertices on the one-ring neighborhood. Their focus is on calculation of principal curvature directions rather than the curvature magnitudes.

### **Implicit Conic Functions**

Implicit functions provide an alternative fitting method that does not require a parameterization of the surface. Conic surfaces, particularly ellipsoids, have been used for surface fitting in applications such as medical imaging. Douros and Buxton [42] extend this approach to a general conic:

$$ax^2 + by^2 + cz^2 + dxy + exz + fyz + gx + hy + iz + j = 0$$

### **Fitting Variations**

A variation of the quadric fitting method can be developed by providing an alternate fitting function. In this variation of quadric fitting, the projection to the tangent plane is replaced with a parameterization using the flattening algorithms of Desbrun et al.[39] and Sheffer[157]. This parameterization is intended to reduce distortion, and is less likely to produce folding. We evaluate such a parameterization where the boundary of the flattened mesh is a natural outcome of the flattening process. In the experimental results, this technique will be referred to as the natural parameterization, and will be compared to the planar parameterization based on a projection to a reference plane. Replacing the quadratic equation fitting function with radial basis functions was also explored. This variant used radial basis function with a uniformly weighted Gaussian, which has well-behaved derivatives at the data points.

### 3.2.2 Discrete Methods

One of the main motivations for discrete methods is to avoid the computational costs associated with fitting algorithms. These methods do not involve solving a least square problem and are very fast. However, many of these methods only provide a subset of Gaussian, mean, and principal curvature directions (unlike a surface fit, from which any of these data can be calculated). Table 3.2 lists several common discrete curvature estimation methods.

#### Spherical Image

The spherical image method [115] uses the unit normals of the one-ring vertices, translated to a common origin, to define a region of a unit sphere, and approximates Gaussian curvature as the ratio of the spherical area to the one-ring area.

#### Angle Deficit

The angle deficit method [164, 115, 117], based on the Gauss-Bonnet theorem, approximates Gaussian curvature as  $2\pi$  minus the sum of the angles for the faces at a vertex, divided by an area associated with the vertex. Cohen-Steiner and Morvan [30] combine the angle deficit method with the integral absolute mean curvature form (below) from the theory of normal cycles, and present a bound on the error for a restricted Delaunay triangulation.

#### Angle Excess

The angle excess or turtle-walking method [164] is similar to the angle deficit method, but approximates Gaussian curvature as  $2\pi$  minus the total turning angle for a path around a vertex divided by the area enclosed by the path. The path is taken as the boundary of a one-ring neighborhood.

Table 3.2: Curvature Calculation Taxonomy - Discrete Operators

<i>Curvature Calculation Taxonomy - Discrete Operators</i>							
Type	Paper	Data	Gauss	Mean	Princ Crv	Crv Dir	Req Norm
<i>Range image methods</i>							
Finite Diff.	McIvor & Valkenburg(1997)	NxN	X	X		X	
Srf Norm. Change	Flynn & Jain(1989)	NxN			X	X	X
Cross Patch	Stokely & Wu(1992)	NxN	X	X			
<i>Mesh methods</i>							
Meusnier-Euler	Chen & Schmitt(1992)	N Pairs			X	X	
	Hameiri & Shimshoni(2002)	N Pts			X	X	
Angle Deficit	Stokely & Wu(1992)	1-Ring	X				
	Meek & Walton(2000)	1-Ring	X				
	Meyer et al.(2002)	1-Ring	X				
Angle Excess	Stokely & Wu(1992)	1-Ring	X				
Integ. Abs. Mean	Dyn et al.(2001)	1-Ring		X			
Norm. Crv. Vec.	Meyer et al.(2002)	1-Ring		X		X	
Spherical Image	Meek & Walton(2000)	1-Ring	X				X

## Integral of Absolute Mean Curvature

Falcidieno and Spagnuolo [48] employ a reasoning approach to extract curvature information, including a discrete measure of mean curvature. This measure assigns a measure of curvature to each edge as a function of the angle between the faces incident on the edge, and then sums the contributions for each edge incident on a vertex to determine the curvature at that vertex. They also convert the curvature to a curvature density by dividing by an associated area based on Voronoi neighborhoods. Dyn et al. [43] instead normalize this sum by the edge length divided by four times the area associated with the vertex. This method is then paired with the angle deficit method to use as a cost function when optimizing the triangulation of a cloud of points.

## Meusnier and Euler Theorem

Chen and Schmitt [27] estimate normal curvature and principal curvature directions by solving for the coefficients of the Dupin indicatrix [78] using three or more circular fits through a vertex and two of its neighbors. A normal section is the intersection of the surface with a plane containing the normal vector. Since there are many triples of points that can be used to create circular fits, the ones forming curves closest to a normal section are used. Hameiri and Shimshoni [70] use quadratic curves to estimate the normal section between the vertex with its normal and neighboring vertices.

## Curvature Normal Operator

Meyer et al. [117] compute mean curvature by using a summation to approximate the integral of the Laplacian over the area associated with a vertex, and normalize by this area. This area can be a mixture of Voronoi and Barycentric area, depending on whether or not triangles are obtuse. They assume mild smoothness conditions and incorporate local operators to denoise arbitrary meshes while preserving features. The mean curvature is combined with Gaussian curvature computed using the angle deficit method to derive principal curvatures, and a least-squares method is employed to calculate principal directions.

## Derivative Calculation

Csakany and Wallace [36] use a simplified approach to compute the second derivatives at a vertex of a mesh. They first compute the surface normal by averaging adjacent face normals. The normal defines the first partial derivatives. A substitution scheme is used to directly compute the second partial derivatives, which can be used to estimate curvature. Their scheme is considered a simplification of an auto-correlation method and a Hessian matrix method, and has been applied to both range images and tessellated data.

## Other

Tang and Medioni [172] compute the sign and direction of curvature, without fitting or derivative calculation, using a voting scheme with weighting based on proximity. Their technique does not provide a curvature magnitude estimate.

### 3.2.3 Estimating the Curvature Tensor

Curvature tensor estimation is similar to the discrete methods, except that instead of estimating the curvature directly, a discrete estimation of the curvature tensor is created, and the curvatures and principal directions are calculated from the curvature tensor. These methods tend to have computational complexity lower than the fitting methods, but slightly higher than the discrete methods. Table 3.3 lists several curvature tensor estimation methods.

## Integral Formulation

Taubin[173] proposes a method that estimates the tensor of curvature from the eigenvalues and eigenvectors of a  $3 \times 3$  matrix, which approximates an integral as a summation around a one-ring neighborhood. He also incorporates a smoothing step for noisy meshes. A key benefit of his method is its simplicity, with the complexity being linear in both time and space. Hamieri and Shimshoni [70] propose modifications

Table 3.3: Curvature Calculation Taxonomy - Curvature Tensor Estimation

<i>Curvature Calculation Taxonomy - Curvature Tensor</i>									
Type	Paper	Data	Gauss	Mean	Princ Crv	Crv Dir	Req Norm		
<i>Range image methods</i>									
Integral Form.	Taubin(1995) Hamieri & Shimshoni(2002)	1-Ring N-Ring			X X	X X		X	X
<i>Mesh methods</i>									
Integral Form.	Taubin(1995) Hamieri & Shimshoni(2002)	1-Ring 1-Ring			X X	X X		X	X
Per Face	Theisel et al.(2004) Rusinkiewicz(2004)	1-Ring 1-Ring			X X	X X		X	X



of Taubin’s method by expanding to more points (primarily for range data), and weighting based on distance rather than triangle area, while Surazhsky et al. [168] proposed weighting based on angle. In the case of a general mesh, it is not clear whether variation in distance or angle will dominate.

### **Per Face Tensor Calculation**

A recent approach calculates the curvature tensor separately for each face [175, 145]. Given a face and the normal vectors at each vertex of the face, this curvature tensor is well-defined. To get the curvature tensor at a vertex, the tensors for each face adjacent to that vertex are averaged.

## **3.3 Evaluations**

Previous studies do not provide an understanding of the differences between mesh size, regularity, and noise issues. We develop a small number of tests that highlight both the detailed behavior of curvature estimation methods and a statistical analysis of errors.

Our detailed behavior test case defines mesh parameters that distinguish between noise (perturbation normal to the surface) and triangulation effects (number, size, and regularity of triangles). We track the error measures as we change parameter values. For example, we can empirically determine if the estimated curvature converges to the known value as the mesh cell size approaches zero. The detailed behavior test case uses an idealized (extremely regular) mesh, except for specific mesh parameter variations. This isolates the effect that specific mesh factors have on the curvature estimation, and provides insight into how sensitive different methods are to these factors.

The statistical analysis test case creates meshes containing vertices for a range of valences, with both regular and irregular mesh regions, and analyzes the errors with

respect to the properties of these meshes. In practice, all of the detailed mesh parameters are likely to vary in a mesh. The statistical analysis helps us determine how the detailed behavior affects the overall behavior on a realistic mesh.

A suite of surface shapes, for which the exact curvature is known, is used to avoid the bias of methods that may be optimal for one particular surface shape. These tests illustrate the behavior of several of the curvature estimation methods discussed above.

### 3.3.1 Curvature Estimation Test Cases

Test cases are constructed by first generating a mesh in the  $X-Y$  plane, and then projecting the mesh in the  $Z$  direction onto surfaces of different shapes, represented by the following equations, as used previously by Hamann [69] and Cazals and Pouget [24]:

$$\textit{Sphere} : x^2 + y^2 + z^2 = 4$$

$$\textit{Cylinder} : x^2 + z^2 = 4$$

$$\textit{Ellipsoid} : (x/3)^2 + (y/2)^2 + (z/4)^2 = 1$$

$$\textit{EllipticParaboloid} : z = 2x^2 + y^2$$

$$\textit{Hyperboloid} : z = 0.4(x^2 - y^2)$$

$$\textit{MonkeySaddle} : z = 0.2(x^3 - 3xy^2)$$

$$\textit{CubicPolynomial} : z = 0.15(x^3 + 2x^2y - xy + 2y^2)$$

$$\textit{TrigonometricFunction} : z = 0.1[\cos(\pi x) + \cos(\pi y)]$$

$$\textit{ExponentialFunction} : z = 0.1e^{2x+y-y^2}$$

Figure 3.4 shows the mesh geometry for these shapes.

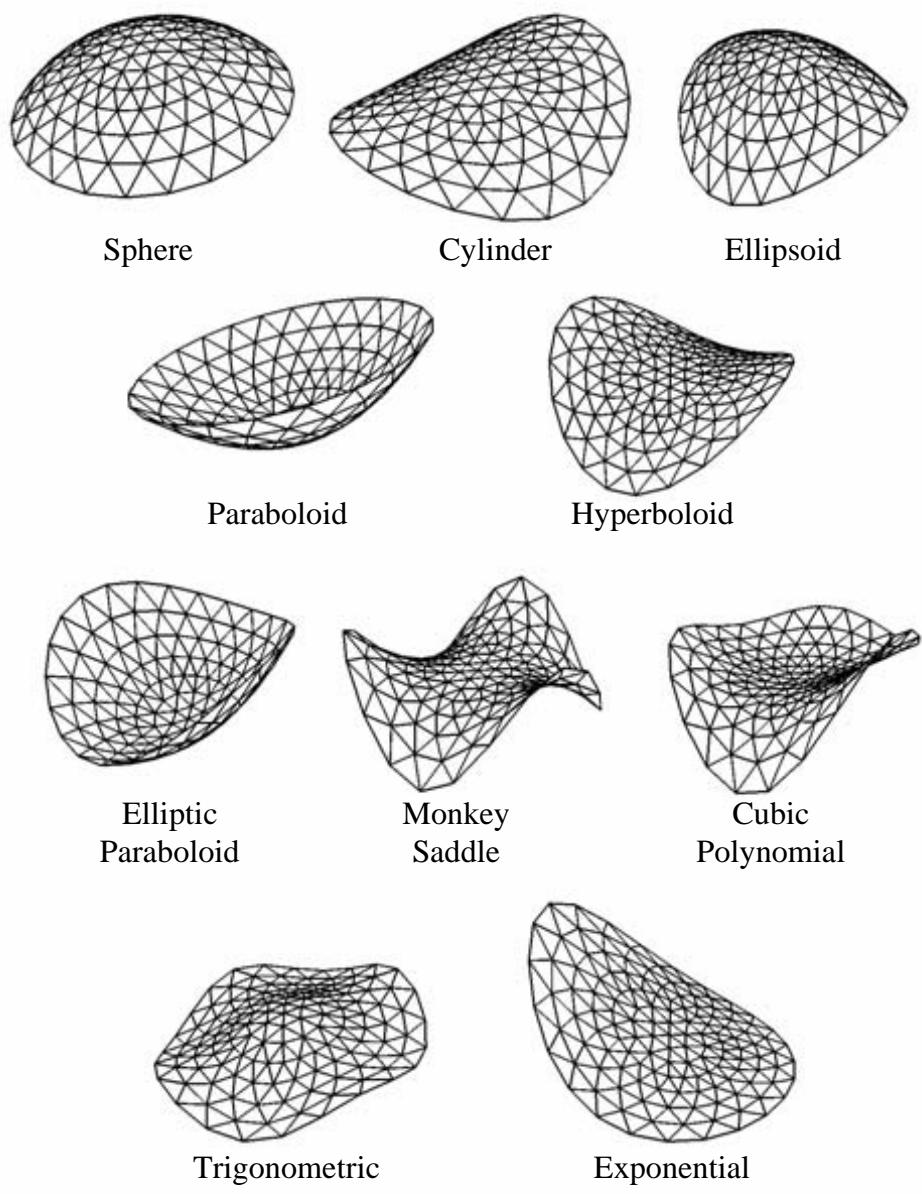


Figure 3.4: Test case geometric shapes.

## Detailed Mesh Parameters

In order to create a mesh that represents our test surfaces, a planar triangular mesh is projected onto the surface, centering the mesh at the target point. The target point is the vertex at which we are calculating curvature, and is always the center point of the mesh. The mesh is a regular  $N$ -ring neighborhood, where  $N \in \{1,2,3\}$ . Sample two-ring meshes are shown in Figure 3.3. The vertices are equally spaced along the rings around the target vertex, except for variations of one of the seven parameters that are used to control the qualities of the mesh:

- $n$ , the number of vertices (valence) in the first (adjacent) ring, with the second ring containing twice as many vertices,
- $\phi$ , the cell size (a relative distance from the target vertex to the first ring of adjacent vertices, and between successive rings of vertices on the test surface),
- $dR_T$ , the displacement of the target vertex normal to the surface,
- $dR_A$ , the displacement of an adjacent vertex normal to the surface,
- $d\phi_T$ , the displacement of the target vertex along the surface toward an adjacent vertex,
- $d\phi_A$ , the displacement of an adjacent vertex along the surface toward or away from the target vertex, and
- $d\theta$ , the displacement of an adjacent vertex along the surface toward a neighboring adjacent vertex.

The normal displacements,  $dR_T$  and  $dR_A$ , represent noise, *i.e.*, true deviation from the actual surface geometry, and are applied after the mesh is projected to the surface. This noise is synonymous with measurement error.  $d\phi_T$ ,  $d\phi_A$ , and  $d\theta$  represent perturbations of the triangulation. Examples of perturbations normal to and along the surface are shown in Figure 3.5. Moving the target point radially toward a point on the first ring, or moving a point of the first ring radially or circumferentially along the surface, reduces the regularity of the mesh.

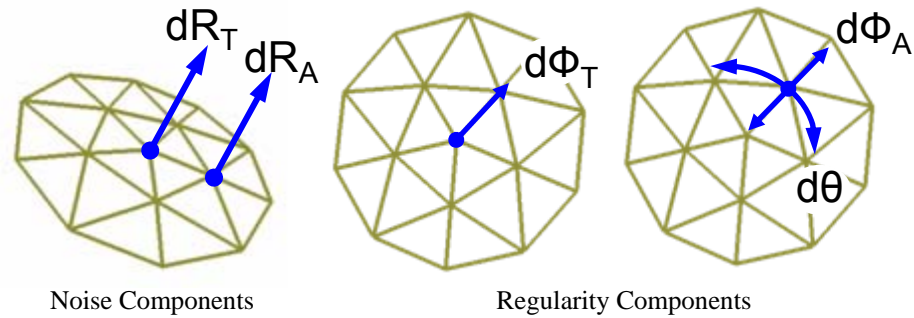


Figure 3.5: Detailed behavior test cases for a mesh projected onto a sphere. Left: Noise component normal to the surface at the target vertex ( $dR_T$ ) and at an adjacent vertex ( $dR_A$ ). Center: Mesh regularity component based on moving the target vertex away from center ( $d\phi_T$ ). Right: Mesh regularity components based on moving an adjacent vertex toward/away from the target ( $d\phi_A$ ) or moving the adjacent vertex along the ring toward a neighboring adjacent vertex ( $d\theta$ ).

To consider different target points and mesh orientations on the surface, an offset and rotation are added. This avoids bias that could occur from looking only at special points, such as the points on the major and minor axes of an ellipsoid, or due to alignment of the mesh with the coordinate axes. For several of the algorithms tested, the accuracy at these special points was better than the accuracy of the method at a generic point on the surface.

The exact curvatures, normals, and principal directions are computed when the mesh is projected to the surface. For methods requiring surface normals, exact normals can be used or approximate normal vectors can be calculated.

### Statistical Analysis Case

For statistical analysis, a mesh containing 72 interior vertices (112 total) is created. This mesh has valence ranging from three to ten, and contains both obtuse and non-obtuse triangles. This mesh is again created in the  $X - Y$  plane and projected onto one of our surface shapes. Figure 3.6 shows an example of the statistical analysis mesh projected to the exponential surface. Statistics for curvature estimation can be broken down by (a) valence, (b) the presence or absence of obtuse angles at the

### Statistical Analysis Test Case

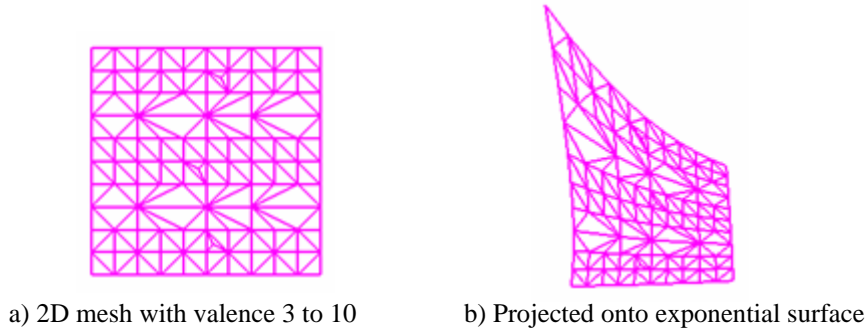


Figure 3.6: Statistical analysis test case. Left: The vertex layout in the  $X - Y$  plane has valence ranging from three to ten (for interior vertices), and contains a mixture of obtuse and non-obtuse triangles. Right: The mesh projected to an exponential surface.

vertex, or (c) the sign of the actual curvature. This breakdown is used to determine the impact of these three factors, and trends associated with the error in the curvature estimation.

### 3.3.2 Experimental Results

The following sections present selected results for the three categories of curvature estimation methods.

#### Curvature Estimation based On Fitting

Various mesh parameters are adjusted to determine their effect on the error in the curvature estimate. The first factor considered is valence (i.e., the number of vertices making up the one-ring neighborhood around the target vertex) and its impact on the Gaussian curvature estimate. The asymptotic behavior of the error for each method is plotted versus cell size, as the cell size decreases. This will be referred to as the convergence of the method. Convergence will be considered to be good if the curve approaches the exact curvature value as the cell size approaches zero.

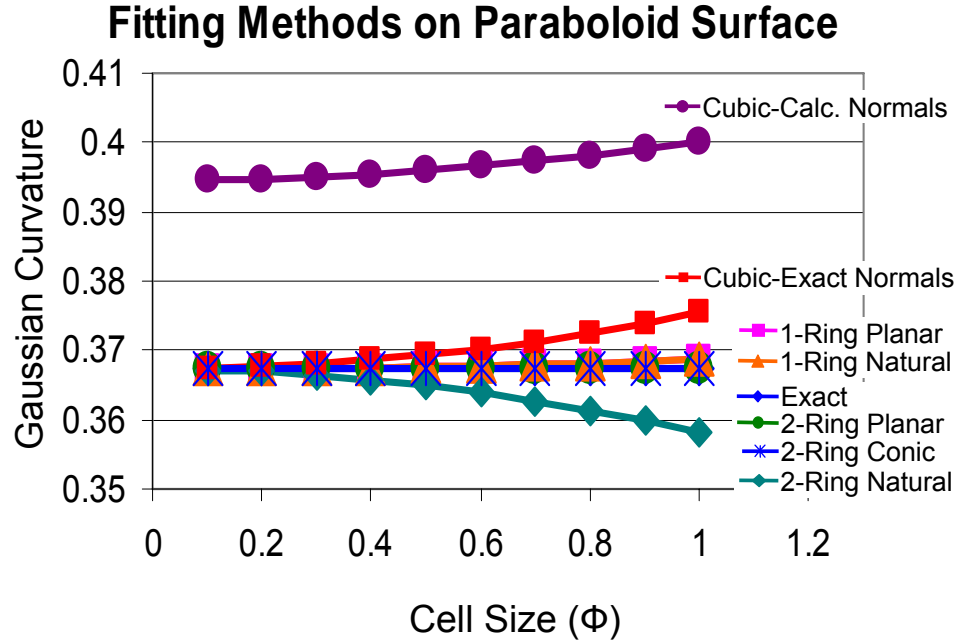


Figure 3.7: Comparison of fitting methods applied to a paraboloid surface. A valence of six was used for the data shown. The cubic fit with computed normals does not converge to the exact curvature of 0.367. The one-ring and two-ring fits behaved similarly, with the one-ring fits being a little more accurate than the two-ring fits.

For one-ring neighborhoods with valences of three or four, the problem may be under-constrained, depending on the number of coefficients in the particular equation being fit. With enough vertices in the one-ring, or using multiple rings, the fitting methods are relatively insensitive to the valence. The cubic fit based on vertex locations and normals converges for all valences when using the exact surface normals, but has poor convergence when using normals calculated as the weighted average of the adjacent face normals.

As the cell size is decreased, corresponding to finer resolution, all of the fitting methods, except the cubic fit with calculated normals, converge to the correct value. Figure 3.7 illustrates the convergence for various fitting methods as a function of mesh resolution on a paraboloid. The conic fit performs well for several surface shapes, and as would be expected, is exact for the ellipsoid. However, the conic fit does not perform as well for some other surface types such as the exponential surface.

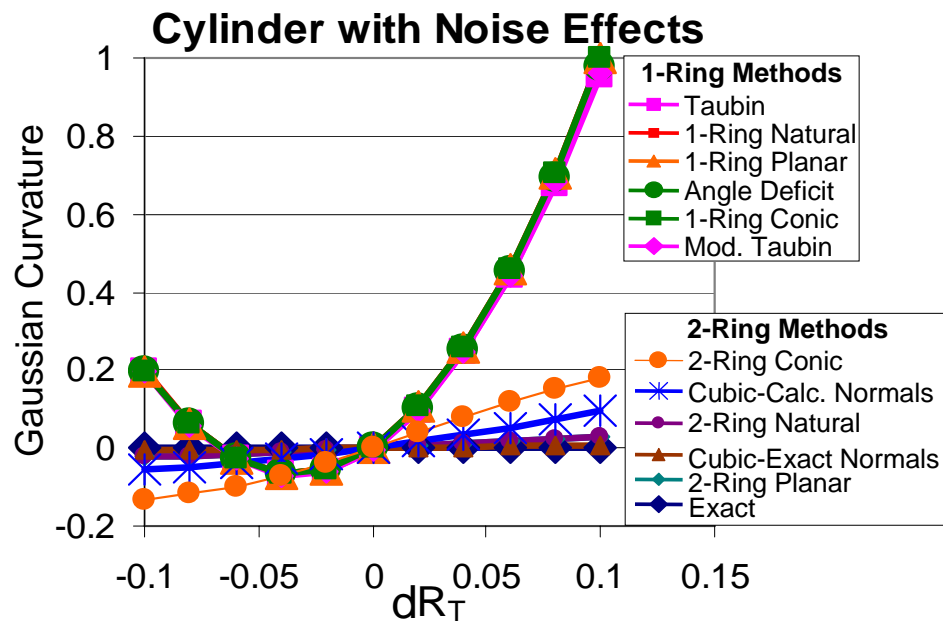


Figure 3.8: Impact of noise normal to a cylindrical surface on the discrete and fitting methods. A valence of six was used for the data shown. The discrete methods and one-ring fitting methods exhibit extreme sensitivity to noise. The cubic fit behaves as a two-ring method and, along with the two-ring quadric fitting methods, shows the least sensitivity to noise.



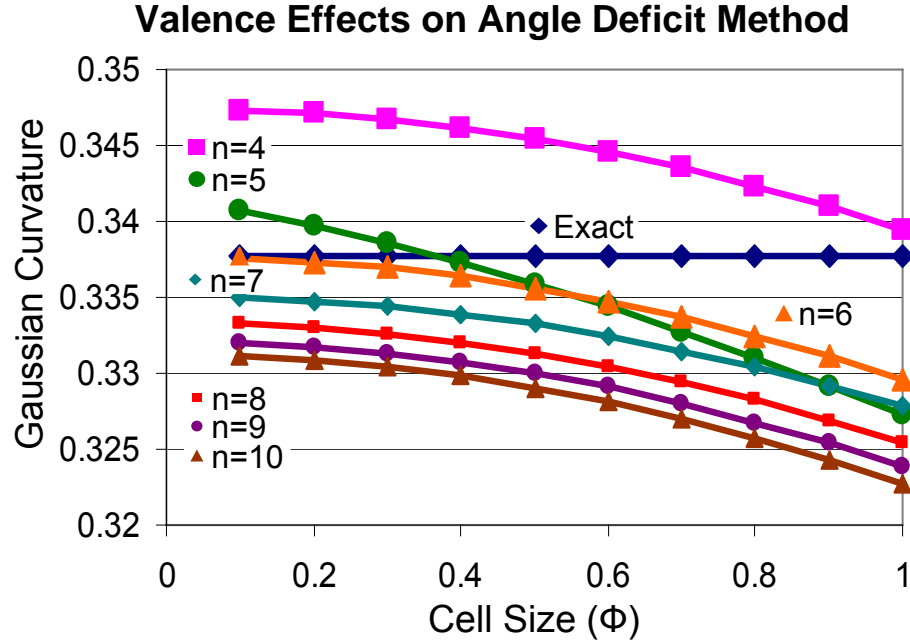


Figure 3.9: Impact of valence ( $n$ ) on the accuracy of the angle deficit method, applied to a cubic polynomial. Increasing Cell Size ( $\phi$ ) represents decreasing mesh resolution. Only valence six converges to the actual Gaussian curvature (0.338). This method is extremely inaccurate for a valence of three, probably due to the effect of obtuse triangles.

The biggest factor distinguishing performance for the fitting methods is the effect of noise in the direction normal to the surface, as shown in Figure 3.8. The quadric and conic fitting methods based on one-ring neighborhoods are extremely sensitive to this type of noise. The normals used with the cubic method effectively provide information from a second ring, and this was the most accurate fitting method in this situation. The fits based on two and three rings also performed well in the presence of noise normal to the surface, with a three-ring fit having no clear advantage over the two-ring fit. The Gaussian curvature estimates from the fitting methods were not particularly sensitive to varying the vertex location along the surface.

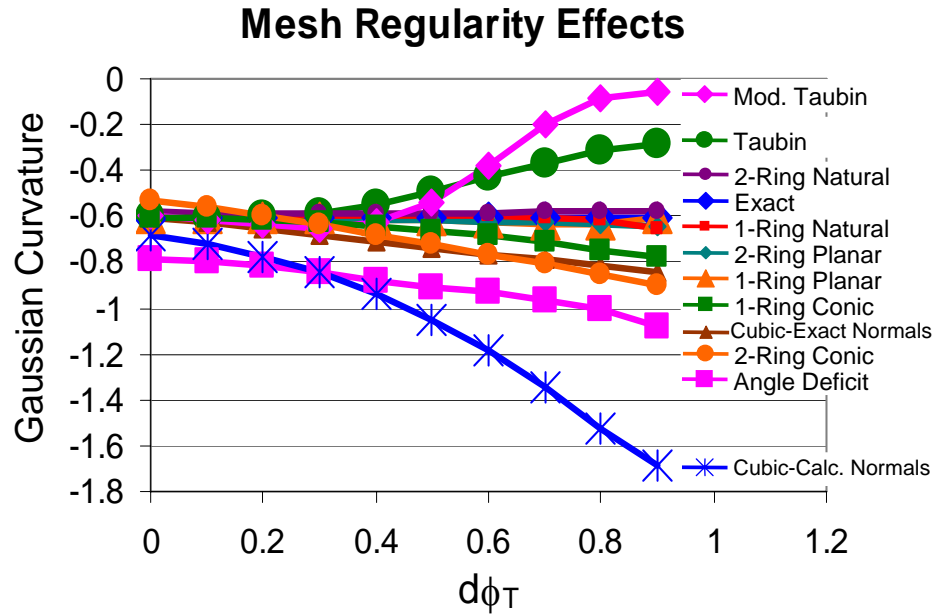


Figure 3.10: Impact of moving the target vertex along the surface toward a one-ring point ( $d\phi_T$ ). The valence is six for all methods. The discrete methods and the cubic fit with computed normals are very susceptible to this mesh quality issue, while the other methods show little sensitivity.

### Curvature Estimation Using Discrete Methods

The impact of valence is most pronounced for the angle deficit method, as shown in Figure 3.9. This method converges to the exact value only for valence six. Note also the distinction between point curvature, which represents our ground truth, and the integral of curvature over a region, upon which the angle deficit method is based. These methods will produce similar results if the curvature is relatively constant over the integration area, but may vary significantly in areas of rapidly changing curvature. There may be applications where one or the other type of curvature information is preferred, and this may lead to a different choice of methods. However, these valence plots show significant variations for essentially the same curvature region.

Like the one-ring fitting methods, the discrete curvature estimation methods [117] suffer from severe sensitivity to noise normal to the surface, as shown in Figure 3.8. But they are also very sensitive to perturbations of the mesh vertices along the surface,

as compared to the fitting methods, as shown in Figure 3.10. This is likely caused by the reliance on angles and areas of the mesh faces, which do not enter directly into the fitting methods.

### Curvature Tensor Results

Figure 3.11 illustrates Taubin’s integral eigenvalue method [173] and Hameiri and Shimshoni’s [70] modification of it on a sphere, calculated on a mesh centered at the origin, and on a mesh offset from the origin. Both meshes are projected in the  $Z$  direction to the same surface. Both methods match the exact curvature of the sphere for the mesh centered at the origin, but are less accurate for the offset mesh. The main difference is that the projection of the offset mesh is not normal to the sphere, which degrades the mesh regularity. This effect is confirmed in Figure 3.12, where starting with a regular mesh, one of the adjacent points is moved along the surface, generating larger error with both methods. For the movement toward or away from the target vertex, the modified method performs better, but the modified method is more sensitive to movement around the ring, as shown in Figure 3.13. Being based on a one-ring neighborhood, they still suffer from severe sensitivity to noise normal to the surface.

### Statistical Analysis Results

Results from the statistical analysis test case were generated for several surface shapes. They confirmed that the variations for some methods were very dependent on the type of surface. Overall, considering both Gaussian and mean curvature, the five most accurate methods were:

1. The cubic fit with exact normals,
2. The two-ring quadric planar fit,
3. The two-ring conic fit,
4. The two-ring quadric natural fit, and

### 5. The cubic fit with calculated normals,

The other methods had consistently larger error. The cubic fit using exact surface normals was best, with the two-ring fits (planar, natural, and conic) having similar errors. However, the two-ring conic fit and the cubic fit using calculated normals had much larger standard deviation than the other top methods. The cubic fit with exact normals and the planar and natural two-ring fitting methods were also most consistent across differences in valence, triangle shape, and curvature sign. This shows that just looking at the overall errors can be deceptive, and mask deficiencies that are only uncovered by more detailed statistical analysis, or through the use of the specific noise analysis test cases. Results for the Gaussian and mean curvature calculations were similar. Mean curvature tends to be better behaved since it is an average rather than the product of the principal curvatures.

In order to place bounds on the accuracy of curvature estimates, it would be useful if methods could be identified that consistently over- or under-predict curvature magnitudes. In our evaluation, the cubic fit with calculated normals under-predicted the curvature magnitudes in most cases, and the two-ring conic fit predictions were larger (more positive or less negative) than the actual Gaussian curvature. However, as discussed above, these methods had other problems that limit the application of these trends. The cubic fit using exact normals and the two-ring quadric fitting methods had smaller error magnitudes, but the sign of the error did not exhibit a consistent trend across the set of test shapes.

### 3.3.3 Discussion of Results

The accuracy for the conic fitting method was very dependent on the type of surface being fit. This points to the importance of comparing methods for more than one type of surface. If an evaluation case is based on the same equation as the fitting method, the results of the evaluation will not necessarily reflect performance for other surfaces to which the method will be applied.

The accuracy of fitting methods and the angle deficit method have been demonstrated in previous studies, as mentioned in Section 2. Our analysis confirms the benefits of

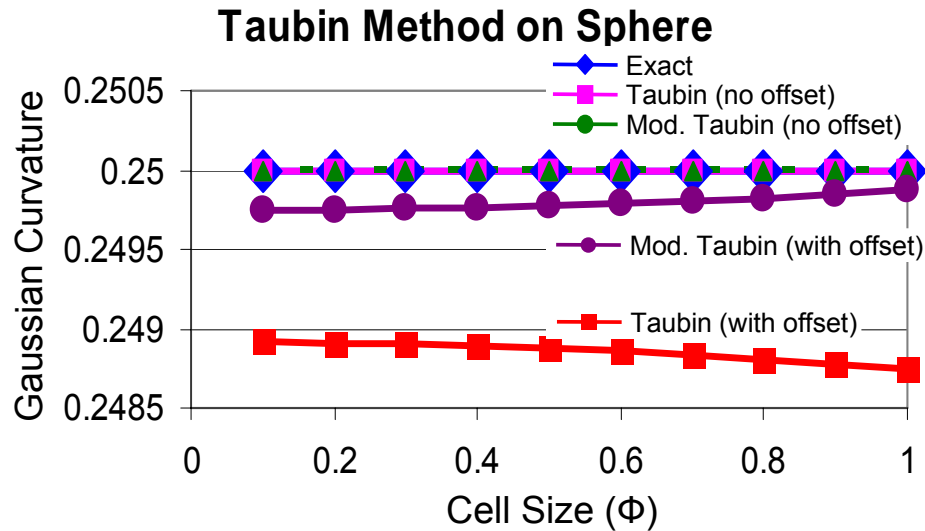


Figure 3.11: Integral Eigenvalue method applied to a sphere as a function of cell size. A valence of six is used for the data shown. The methods match well for a regular mesh centered at the origin, but degrade due to the minor distortion from projection of an offset mesh.

fitting methods, but identifies deficiencies in the angle deficit method, and conic fitting methods. Even without noise, the statistical analysis indicates that the two-ring fitting is superior to the one-ring fitting methods. The detailed behavior suggests that noise normal to the surface severely degrades the one-ring methods, which have higher noise sensitivity.

The cubic fit appears promising, but the sensitivity to the calculation of the normals is a severe drawback. The principal curvature direction calculations appeared more stable and less sensitive to mesh regularity than the curvature magnitudes. See Goldfeather[65] for further discussion comparing calculation of principal directions.

The discrete curvature methods are appealing because of their speed. Fitting is by its nature a more expensive computation. However, the sensitivity to valence, noise, and mesh regularity limit the usefulness of the discrete curvature estimates to very regular meshes for which either noise is absent or smoothing has been applied. The authors of these methods have proposed applying smoothing algorithms for cases with noise. But smoothing can also mask surface detail if not applied judiciously.

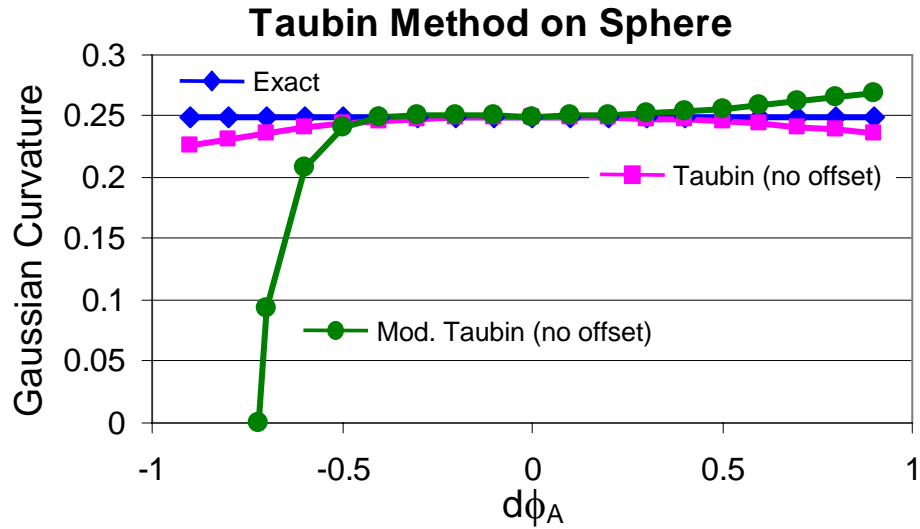


Figure 3.12: Effect of perturbation of an adjacent vertex along the surface toward or away from the target vertex. Both variation of the Integral Eigenvalue method show sensitivity to the mesh regularity.

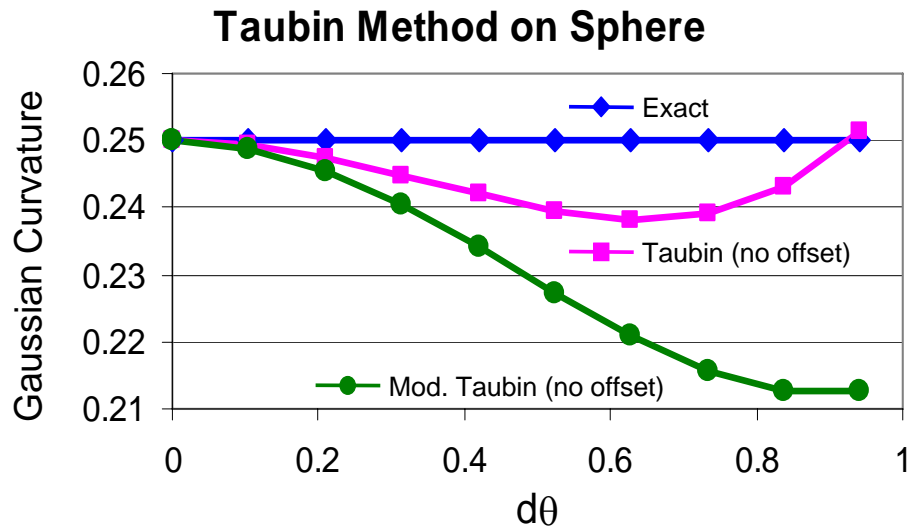


Figure 3.13: Perturbation introducing non-uniform spacing around the ring. Again, both methods exhibit sensitivity to this mesh regularity parameter.

The fitting methods based on two or more rings have better overall performance, albeit at a greater computational cost. In our tests, accuracies for three ring neighborhoods did not warrant the increased cost due to the size of the fitting problem, so a surface fit based on a two ring neighborhood is recommended.

Conic fitting is usually phrased as a least-squares solution that minimizes  $F(x, y, z)^2$ . Scaling the conic equation by a constant value does not change the zero set, but it does change the value of  $F(x, y, z)$ . For this reason, fitting is generally more stable if the points are first transformed to a local coordinate system centered around the origin, with the normal pointing in the  $y$  direction.

Fitting using radial basis functions did not yield suitable curvature estimates. However, there are a variety of possible formulations that may be worth investigating.

Generating the parametric coordinates used for fitting by projecting to the tangent plane is very fast. Alternatively, generating parametric coordinates based on a flattening [39] of the local mesh avoids potential problems that can occur due to folding or distortion when the mesh is projected to a plane. These techniques require more work and do not provide much accuracy improvement. The behavior for a two-ring fit parameterized by a natural flattening technique was similar to the two-ring planar fit. For smooth meshes, the projection of a two-ring neighborhood is not likely to fold, but the overhead of the flattening technique may be worthwhile for meshes with sharper features.

These results demonstrate the value of our analysis methods to uncover the detailed behavior of curvature calculation methods on triangular meshes, and an approach to statistical analysis that can provide practical assessment of new or existing methods. It is important to recognize that looking at surfaces colored by the calculated curvature values is not very useful for comparing methods. Noise, shape details of the surface, and the surface triangulation affect the accuracy of the curvature estimate and these effects are hard to detect by curvature visualization.

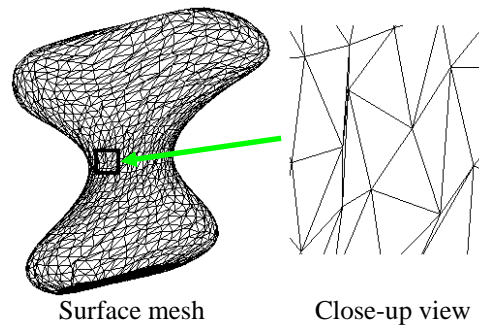


Figure 3.14: Goldfeather's general surface test case. Left: Overall mesh. Right: Expanded view of local mesh.

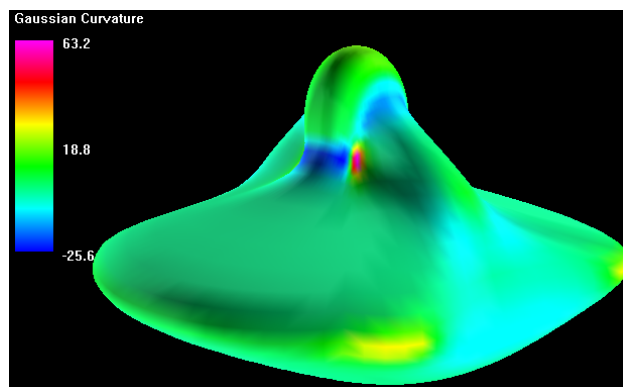


Figure 3.15: Surface plot of exact Gaussian curvature for Grimm's general surface test case with complex curvature.



### 3.3.4 Extension to General Surfaces

How do the results above, derived from analyses of specific behavior for vertices with positive and negative Gaussian curvature, apply to general surfaces? These methods are applied to a torus (with meshes having three different levels of resolution) and to two general surface test cases. All three meshes for the torus represent very regular triangulations. The first general surface test case, supplied by Goldfeather [64], is shown in Figure 3.14, and contains a range of valences, angles, and edge lengths, with 56.8% of the triangles being obtuse. An example of the irregularity of this mesh is shown in the close-up view on the right. Figure 3.15 shows the second general surface test case, which was built from rational polynomials [59]. The Gaussian curvature ranges from -25 to 63. The mesh tessellation is very regular (squares split into triangles) for this test case.

For the torus, although the overall mean error is lower for cubic fitting with normals, the standard deviation is much larger. This is due to the much larger mean error for the positive and negative Gaussian curvature vertices. The cubic fitting with normals method estimates positive curvature regions as more positive and negative curvature regions as more negative. The discrete angle deficit and curvature normal operator method also estimates positive curvature regions as more positive, but estimate negative curvature regions as less negative. The fit with normals method has about the same magnitude mean error for positive and negative regions, but the standard deviation is much larger for negative curvature regions. The discrete operator error is about a factor of eight to ten lower than the fit with normals method for positive curvature regions, but only about three to four times lower for negative curvature regions. The cubic fit with normals method using computed normals is only slightly worse than the cubic fit with normals method using exact normals, probably due to the high quality of the mesh.

For the first general surface test case, the fit with normals method using exact normals has better mean and standard deviation for negative curvature regions than for positive curvature regions. When using computed normals, the mean and standard deviations are comparable for positive and negative curvature regions, with the mean error slightly larger for positive curvature regions, and the standard deviation slightly larger for negative curvature regions. The discrete operator method has bigger mean

error for negative curvature regions, and slightly bigger standard deviation for positive curvature regions. But the bottom line is that the cubic fitting with normals method using exact normals performs well, with the other three methods doing much worse. Of these, the discrete angle deficit and curvature normal operator methods have lower mean error, but somewhat larger standard deviation.

For the second general surface test case, the discrete operator, polynomial, radial basis functions, integral formulation of the curvature tensor, and conic methods exhibit the same relative behavior as they did on the torus, although the maximum error is higher. The cubic fitting with normals, using exact or computed normals, exhibited excessive curvature ( $> 1000$ ) for a small number of points.

### 3.4 Chapter Summary

This chapter has described a suite of test cases that model mesh variations to assess the impact of mesh resolution, regularity, valence, and noise on the accuracy of curvature calculation algorithms for triangular meshes. In addition to fundamental mesh issues, this suite includes statistical analysis that also addresses different aspects of curvature estimation error. Along with a summary of existing curvature estimation methods, evaluation results for the most common surface fitting and discrete methods have been generated to produce guidelines for choosing an algorithm. Using the behavior of curvature estimation methods to place bounds on the error in the curvature estimates based on mesh resolution and other factors is an area for further research.

We use surface curvature in developing our shape representation and similarity analysis described in the next chapter. Due to the potential for noise in the surface meshes we are using, we choose a fitting technique based on a two-ring neighborhood. We employ the natural parameterization based on flattening to avoid the possibility of folding. So the curvatures used in the remainder of this dissertation are calculated using a quadric fit on a natural parameterization of the two-ring neighborhood.

## Chapter 4

# Curvature Map Similarity Measure

This chapter addresses the problem of local shape similarity, *i.e.*, is a region of a surface the same “shape” as another region? Similarity information is useful for several tasks in our shape matching process. For example, identifying corresponding regions of two surfaces, based on the similarity of the regions, is a necessary first step toward alignment and registration of those surfaces. Similarity is also used to identify corresponding points and to evaluate the final similarity of objects based on their corresponding points.

There are several goals for a shape similarity measure. The priority for these goals is dependent on the intended use. For detailed shape matching, the similarity measure needs to be capable of discriminating between fine shape variations. To compare subtle shape differences, it must be relatively insensitive to noise, for example, due to measurement tolerances. A coarse similarity measure may be used to rapidly identify significantly dissimilar points, with a higher fidelity, but likely slower, method applied only to points that pass some initial similarity threshold. Since objects may be sampled very differently, the shape similarity measure should also be independent of the point distribution on the object’s mesh representation.

Many previous approaches to shape similarity have focused on object recognition. For efficiency, these methods try to match objects using a minimal number of key points or with a single shape signature. Because objects of different classes are not very similar, coarse methods are often adequate.

Point-based similarity measures, such as Hausdorff distance [79], multi-resolution Reeb graphs [74], shape distributions [130], and spin images [84], can be sensitive to

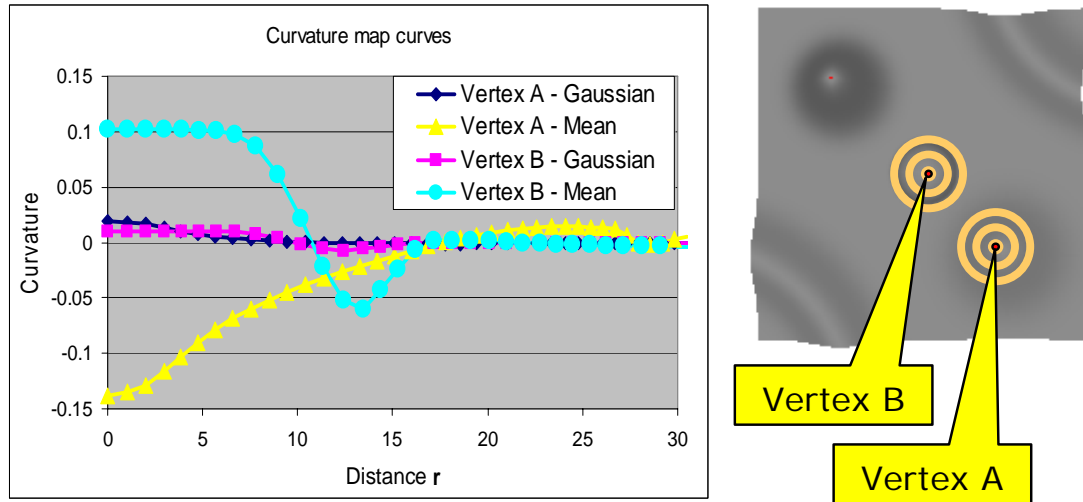


Figure 4.1: Conceptual view of a 1-D curvature map. The curves on the left represent mean and a Gaussian curvature functions for two sample vertices,  $A$  and  $B$ . The distance can be considered as expanding concentric rings (first five rings shown on the right), with the curvature value found by averaging the curvature values in the associated ring.

the distribution of points on the surface of the object. Many point-based methods are also global in nature, so that shape changes anywhere on the object can effect the signature. The statistical sampling of shape contexts [119, 92, 57] adds uncertainty beyond the point distribution sensitivity and global nature.

Local methods, such as local feature histograms [72] and distance and angle descriptors [135] work well for coarse matching, but are less suitable for fine matching of similar shapes. The point fingerprint [167] does represent local shapes, but is limited to ‘interesting’ points.

Geodesic fans [186] can provide even more local information about the shape in the region around a point. Like the point fingerprint, geodesic fans can also carry other information such as curvature.

The *curvature map* builds on the geodesic fan construct. We start with a basic geodesic fan, but use both mean and Gaussian curvature as signals. Because curvature is a point metric, it does not provide information about the region around the point.

However, the curvature values at the geodesic fan points create a local 2-D curvature map. A dimensionality reduction to 1-D increases efficiency for coarse matching. The basic concept for the 1-D curvature map is illustrated in Figure 4.1. An alternate formulation generates the 1-D map directly from the object’s mesh without resampling at the geodesic fan points. Curvature maps can be generated at any point on the mesh. Note that using just the curvature at a point is the 0-D form of the curvature map function.

This chapter develops the curvature map and comparison functions for local shape similarity. In Section 4.1, we investigate various methods for building curvature maps from both mean and Gaussian curvature, and the effect of the size of the region. We then define a similarity function that compares two curvature maps in Section 4.2. Curvature maps are robust with respect to grid resolution and mesh regularity. Both the 1-D and 2-D comparison functions yield a high degree of discrimination for local shapes, compared to the 0-D (point curvature) methods which have been used previously. Curvature calculation on discrete meshes is often noisy [71] and not always accurate [60]. Because curvature maps combine curvature information over a region, they are less susceptible to these issues.

## 4.1 Defining Local Surface Shape

This section describes the construction of the curvature map, and how it is used to identify regions of similar shape. Two methods are defined for creating samples around the point, one based on the mesh topology (Section 4.1.1) and one based on geodesic sampling (Section 4.1.2). Next, we describe how curvature is calculated on the mesh. Finally, we define the function for evaluating shape similarity.

### 4.1.1 Defining Rings of a Mesh

Given a specified vertex of the mesh, a set of “rings” around the vertex is defined using the existing mesh structure. The  $i^{th}$  ring around vertex  $v_0$  is defined as the

set of vertices  $v \in V$  such that there exists a shortest path from  $v_0$  to  $v$  containing  $i$  edges. The set of rings  $R_i, i \leq N$  defines the  $N$ -ring neighborhood about  $v_0$ .

Figure 4.2 shows the first nine rings around a selected vertex of the mesh. The ring structure can be extended an arbitrary distance from any point; however, as the distance increases, the shape of the ring may become irregular.

### 4.1.2 Geodesic Fans

Geodesic fans [186] represent a local surface resampling that provides a uniform neighborhood structure around a vertex. In particular, a geodesic fan consists of a set of spokes, and a set of samples on each spoke. The spokes are geodesics marched out across the surface from the neighborhood center, equally spaced in the conformal plane of the neighborhood's 1-ring. With the samples equally spaced along each spoke, they form a local geodesic polar map around the vertex. Each set of points equi-distant from the neighborhood center is treated as a ring. Following Zelinka and Garland [186], we use interpolated normal geodesics [15] where possible, reverting to straightest geodesics [137] if the smoothness criterion for interpolated normal geodesics is not met.

We use this procedure to generate fans at each vertex of the mesh. Sample fans at two vertices are shown in Figure 4.3. Each fan point is defined in terms of the Barycentric coordinates in some triangular face in the original mesh. These Barycentric coordinates are used to interpolate curvature values defined on the mesh to the fan point. This forms a uniform sampling of curvature data around each vertex. As the sampling increases, more overhead is required to store the fan data.

The regularity of geodesic fans can break down as the distance from the point increases, due to a) stretching of the circumferential spacing while the radial spacing remains uniform, and b) issues in constructing geodesics over longer distances. As a result, the fan resolution may be locally finer, coarser, or both, when compared to the mesh resolution. If the sampling is coarser than the mesh triangle size, then the geodesic fan will not incorporate all of the curvature data available.

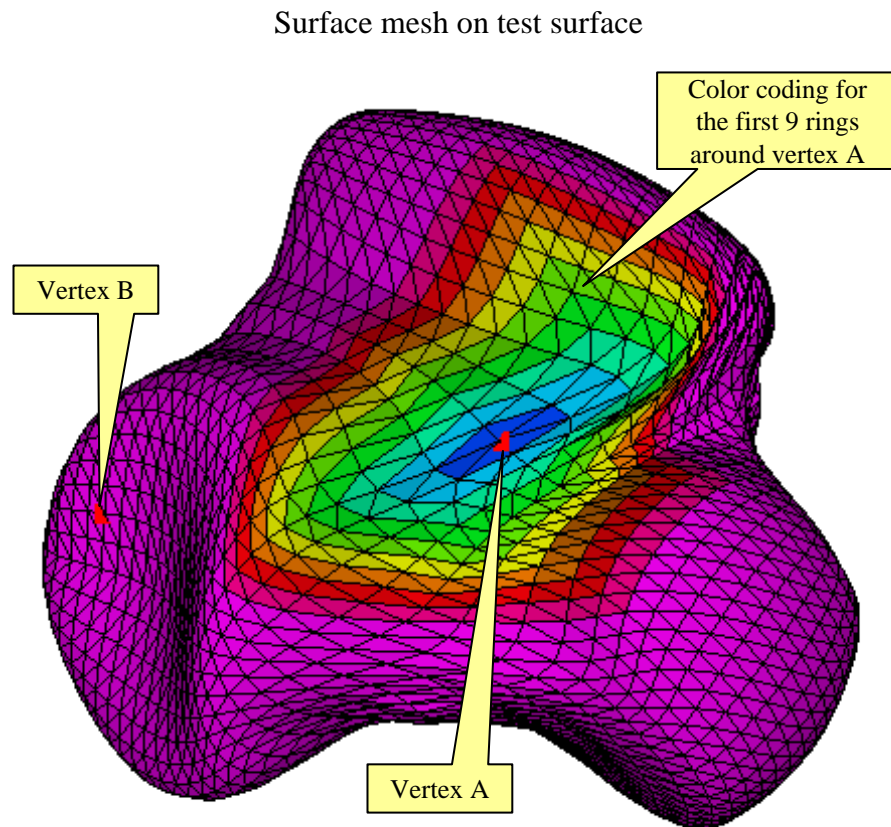


Figure 4.2: Test surface with vertices A and B highlighted. The first nine rings defined around vertex A are color coded. The mesh is fairly uniform except for blending between sections. Note that the ring structure is still well-defined in spite of the skewness near its right edge.

Geodesic fans for vertices A and B  
(20 spokes, 11 points per spoke)

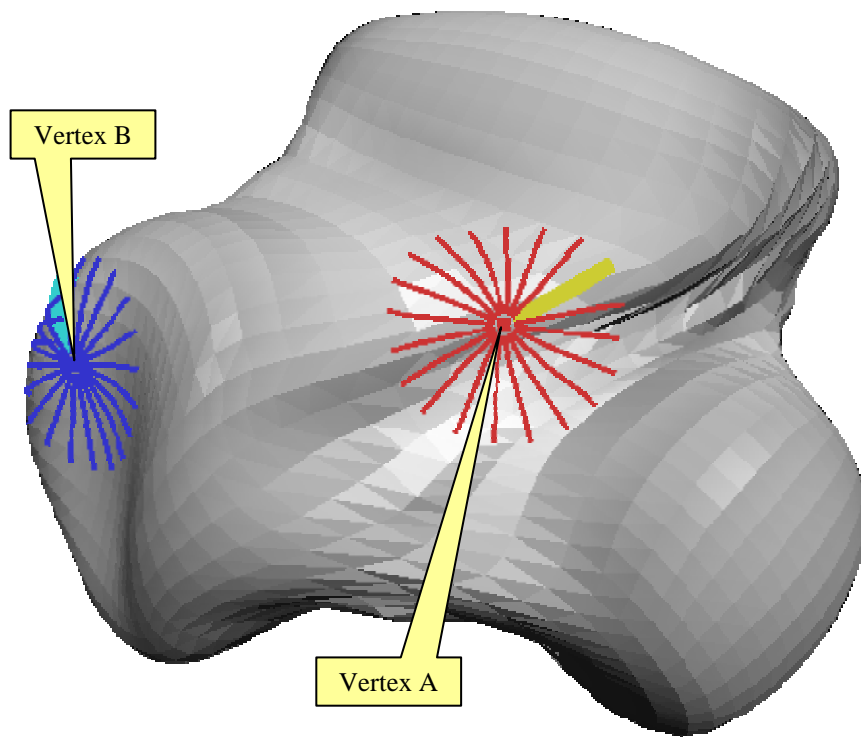


Figure 4.3: Geodesic fans at two vertices. The first spoke of each fan is highlighted and used to track the relative orientation for 2-D fan comparisons. Fan parameters include the number and length of spokes, and the number of points per spoke.



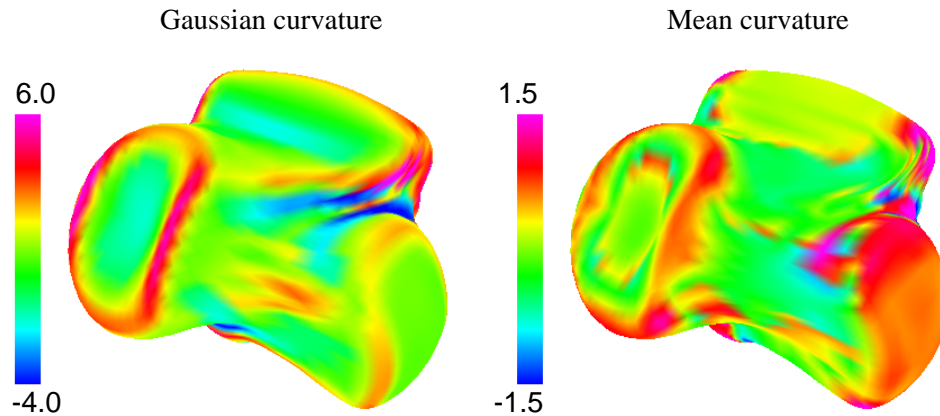


Figure 4.4: Gaussian curvature (left) and mean curvature (right). Note that the Gaussian curvature ranges over  $[-4, 6]$ , while the mean curvature ranges over  $[-1.5, 1.5]$ .

### 4.1.3 Estimating Curvature

Based on the evaluation of curvature estimation methods for triangular meshes in Chapter 3, we calculate curvature by fitting a 2-ring neighborhood using a natural parameterization of the input mesh [39]. This method is reasonably robust with respect to noise as well as mesh irregularity, and provides consistent accuracy of the curvature values. Gaussian curvature and mean curvature are plotted as scalar properties on the surface of the test shape in Figure 4.4.

### 4.1.4 1-D Curvature Maps

The 1-D form of the curvature map is defined over  $M$  rings, where the rings come from either the mesh structure or the geodesic fan structure. Each point  $p_i$  in the map is constructed from data accumulated along the ring  $R_i$ . The point  $p_i$  can have one or more data values; this allows us to compare, for example, both the Gaussian

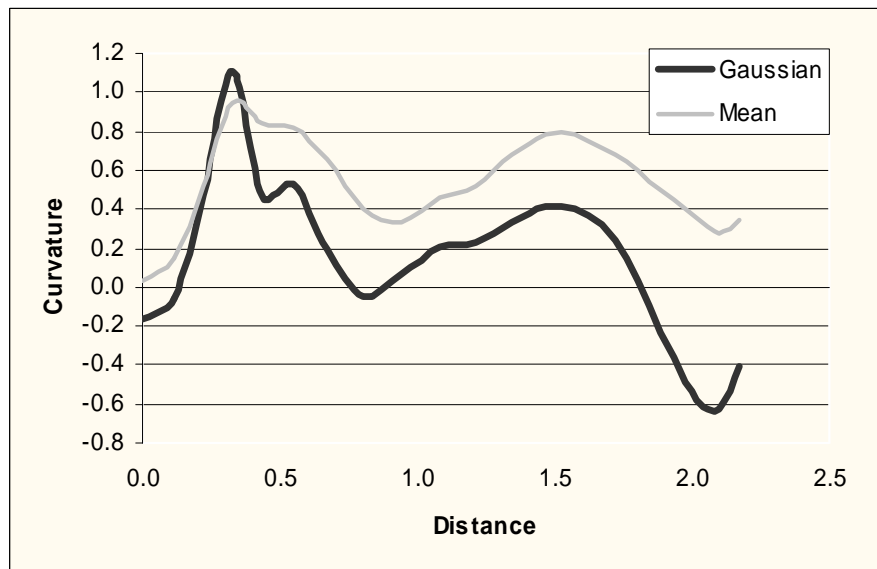


Figure 4.5: Maps of Gaussian curvature and mean curvature as a function of distance from the point. Peaks and valleys tend to be more pronounced for the Gaussian curvature curve, which is the product of the principal curvatures ( $\kappa_g = \kappa_1 \kappa_2$ ), compared to the mean curvature, which is an average ( $\kappa_m = \frac{\kappa_1 + \kappa_2}{2}$ ).

and the mean curvature simultaneously (see Figure 4.5). Each element of  $p_i$  generates a curve as a function of the ring distance <sup>1</sup>.

Because the Gaussian curvature is a product of the principal curvatures and the mean curvature is an average, the Gaussian curvature magnitudes will be roughly proportional to the square of the mean curvatures. A square root function applied to the Gaussian curvatures gives a more equal weighting. Similarly, we use a logarithmic function to reduce the effect of large variations in the peak curvature values, which tend to dominate over areas with lower curvature magnitudes. The emphasis here is to match the shape of curves rather than just the magnitude of the peaks.

More formally, the curvature map  $\kappa_{map}$  at a vertex  $v$  is a set of  $N$  piecewise linear functions defined over the rings  $R_i$ :

$$\kappa_{map} = \{f^j : r_i \rightarrow \mathfrak{R}\}_{0 < j < N}, \quad 0 < i < M \quad (4.1)$$

$$r_i = \sqrt{A_i/\pi} \quad (4.2)$$

$$g(\kappa) = \begin{cases} \frac{1}{N_i} \sum_{w \in R_i} \kappa(w) & \text{or} \\ \max_{w \in R_i} \kappa(w) & \text{or} \\ \min_{w \in R_i} \kappa(w) \end{cases} \quad (4.3)$$

$$h_1(x) = \begin{cases} x & \text{or} \\ \text{sign}(x) \text{ sqrt} \|x\| \end{cases} \quad (4.4)$$

$$h_2(x) = \begin{cases} x & \text{or} \\ \text{sign}(x) \log(1 + \|x\|) \end{cases} \quad (4.5)$$

$$f^j(\kappa) = h_2 \circ h_1 \circ g(\kappa) \quad (4.6)$$

where  $A_i$  the area of the  $i$ -ring neighborhood. The functions  $f^j$  can be applied to Gaussian  $\kappa_g$  or mean  $\kappa_m$  curvature.  $r_i$  is used to normalize the parameterization of the  $f^j$  curves with respect to the area covered by the region.

---

<sup>1</sup>The curvature map is formulated for a discrete mesh, but the same concept can be applied to an analytic surface, where the curve values for discrete increments would be replaced by a continuous function on the surface.

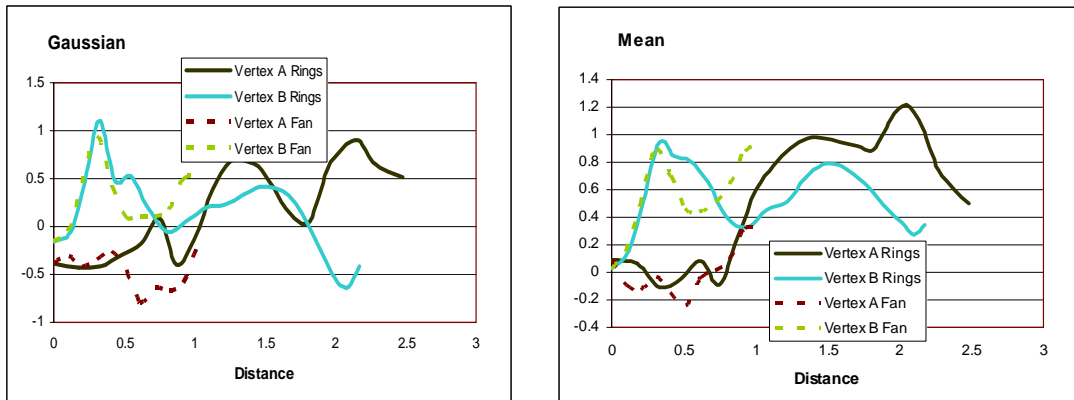


Figure 4.6: Map of Gaussian (left) and mean (right) curvature as a function of distance from the selected vertex. The ring-based and fan-based curves for a particular point start out at the same value and initially have similar shape, but diverge due to a) non-uniformity of the rings, and b) fans sampling only a subset of the data, as the distance from the center increases. In this case, the fans cover a smaller area than the rings.

To compare the shape at two points, such as those shown in Figure 4.2, we compare the corresponding curvature map functions (see Figure 4.6). The shape similarity,  $S_1$  is a function of the difference between the individual curves. Let  $f_A$  be the set of curves for one point, and  $f_B$  the curves for the second point.

$$S_1 = \sum_j \int_0^R (\|(f^j)_A(r) - (f^j)_B(r)\|) dr \quad (4.7)$$

Note that the difference we compute is actually a dissimilarity measure, with zero indicating high similarity and positive values indicating the relative difference between shapes. The user can also specify the radial distance over which the curvature maps are compared. This provides a parameter to control the size of the region used to compute similarity between points.

### 4.1.5 2-D Curvature Maps

The 2-D curvature map is similar to the 1-D map, except that we maintain the angular ( $\theta_k$ ) information and accumulate data only along a single spoke (*i.e.*, there is one  $f^j$  per spoke). Let  $N_s$  be the number of spokes:

$$\kappa_{map} = \{(f_k^j)\}_{0 < j < N, 0 < k < N_s} \quad (4.8)$$

The comparison metric sums up the curve differences along each spoke. There are  $N_s$  possible alignments between two fans; we calculate  $S_2$  for each alignment and choose the smallest value.

$$S_2 = \sum_j \sum_k \int_0^R (\|(f_k^j)_A(r) - (f_k^j)_B(r)\|) dr \quad (4.9)$$

It is important that the fans are generated with the same number of spokes. By checking all possible relative orientations of the fans, the 2-D form can also provide information about the relative orientation of the points. As with the 1-D curvature map, the user can choose the size of the region to compare over by selecting  $R$ .

## 4.2 Shape Similarity Evaluations

Figure 4.7 shows points A, B, and C on two Ulna samples. Points A and B are similar to each other and dissimilar with respect to Point C. This example gives an intuitive sense that the similarity measure does the right thing.

Next, to evaluate our metrics we created a test shape with known curvature properties (see Figure 4.8). Because this manifold surface is defined parametrically, we can easily generate a range of cases for testing that cover curvatures found in realistic

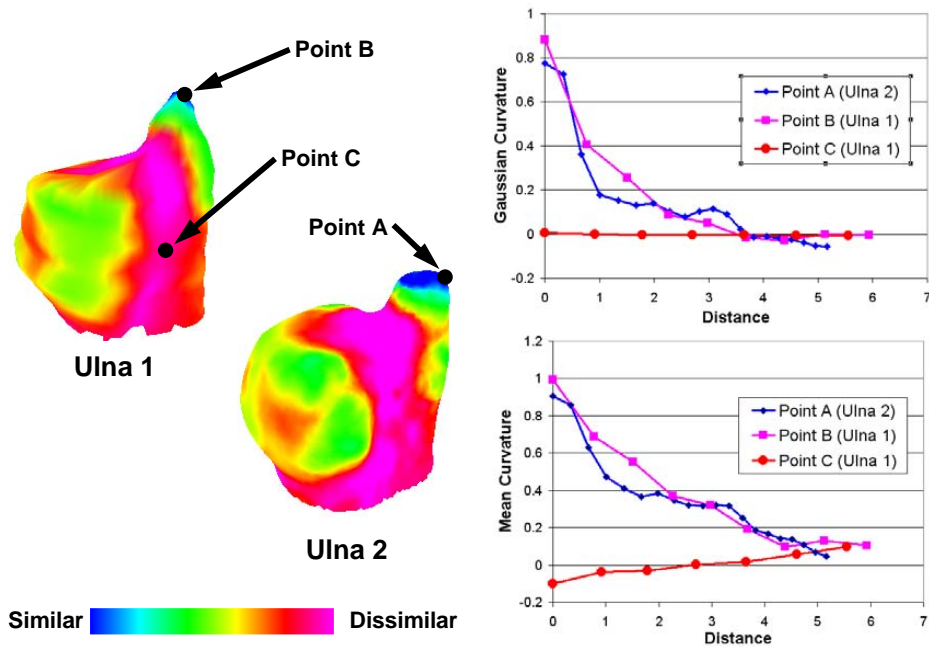


Figure 4.7: Similarity for points on two Ulna samples. Point A and B are similar, and Point C differs from Points A and B.

applications. We also applied the curvature map to standard meshes such as the Stanford Bunny mesh.

We first look at the discrimination power of the 0,1, and 2-D curvature maps, using the “best”  $f^j$  functions for each case. Next, we describe our study to determine which  $f^j$  functions have the best discrimination power. Finally, we look at evaluation times for each of the techniques.

#### 4.2.1 Comparing 0-,1-, and 2-D Curvature Maps

We compare the 0-, 1-, and 2-D curvature maps for our three-lobed test shape and the bunny. The top (Vertex A) and bottom (Vertex B) rows of images in Figure 4.9 show which points on the surface are most similar to the selected vertex. For all of these images, we apply the square root and logarithmic functions to the average Gaussian curvature, and the logarithmic function to the average of the mean curvature. As

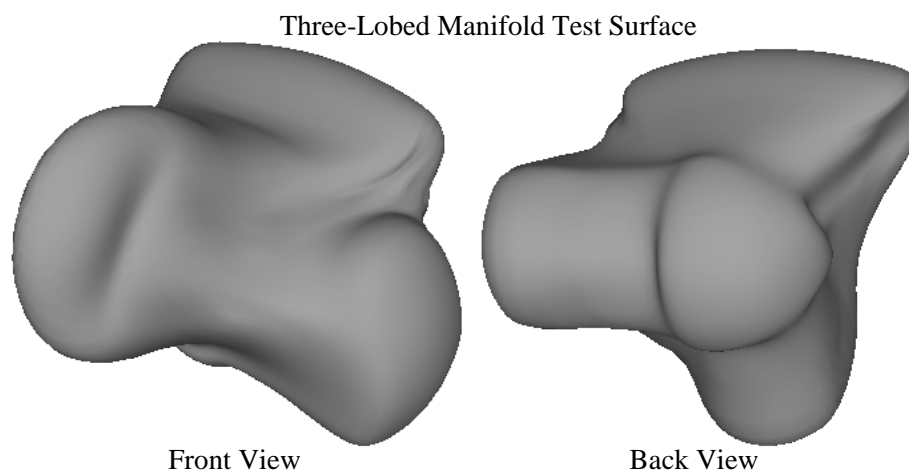


Figure 4.8: Two views of the test surface used for shape comparison. The left and right lobes in the front view are the same except for the addition of a dent (concave region) in the end of the left lobe.

expected, the number of similar points decreases as we increase the dimension of the curvature map.

The ring and fan-based 1-D methods are similar in discriminatory power, but differ slightly in which points they mark as similar. Small differences may be due to differences in the size and shape of the regions covered by the rings and fans. We also varied the size of the region covered by the fans, keeping the same number of spokes and number of points along each spoke. The results remained similar as long as we adjusted the number of rings to match the approximate region sizes.

### Choice of Comparison Functions

The visualization of similarity as a scalar function plotted on the surface of the object gives an indication of the improved ability to differentiate based on shape, but is not as useful in determining which of our 1-D curvature map functions, and associated comparison functions works best. To test these options, we identify groups of points that we expect to be similar, based on our intuition. The similarity for each pair of points is used to form a distance grid. Distance grids for 0-D, 1-D ring-based, 1-D

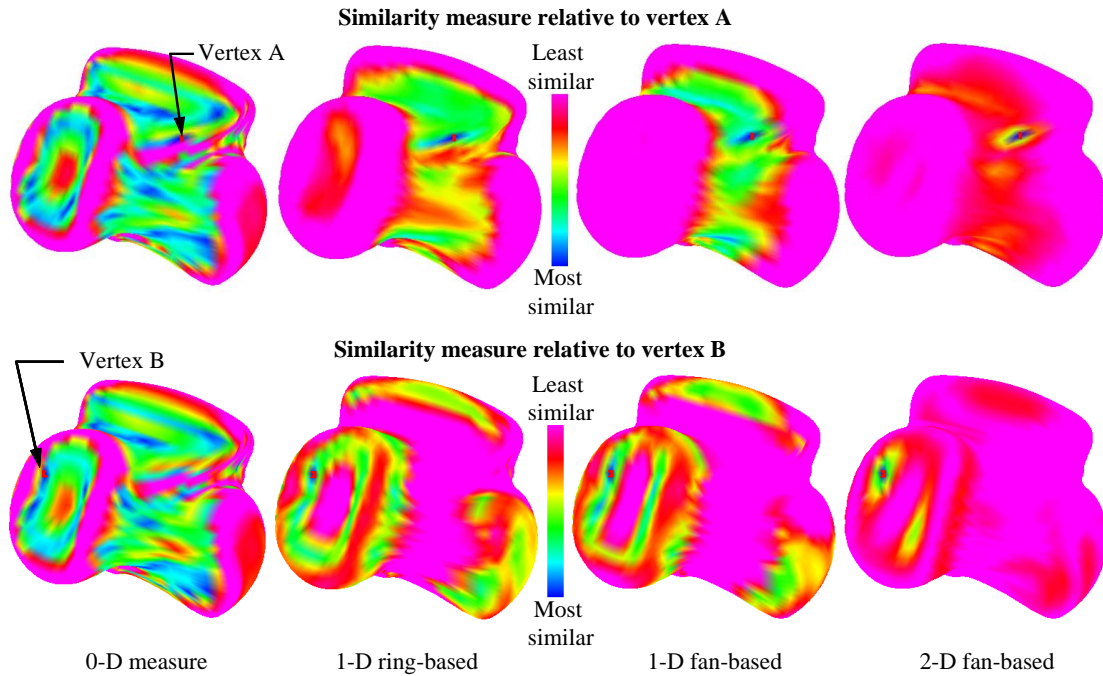


Figure 4.9: Top: Similarity measure relative to vertex A plotted on the test surface. Bottom: vertex B. The color scale ranges from blue (high similarity to the selected point) to magenta (most dissimilar). Nine rings were used in the ring-based calculation. 20 spokes, 11 samples per spoke, were used in the 1- and 2-D fan-based calculation; the surface area is approximately the same as the ring version. Note that the 0-D measure (far left) is very noisy compared to the 1-D ring-based (center left) and fan-based (center right) measures. The 2-D measure (far right) shows few points with similarity to the selected vertex.



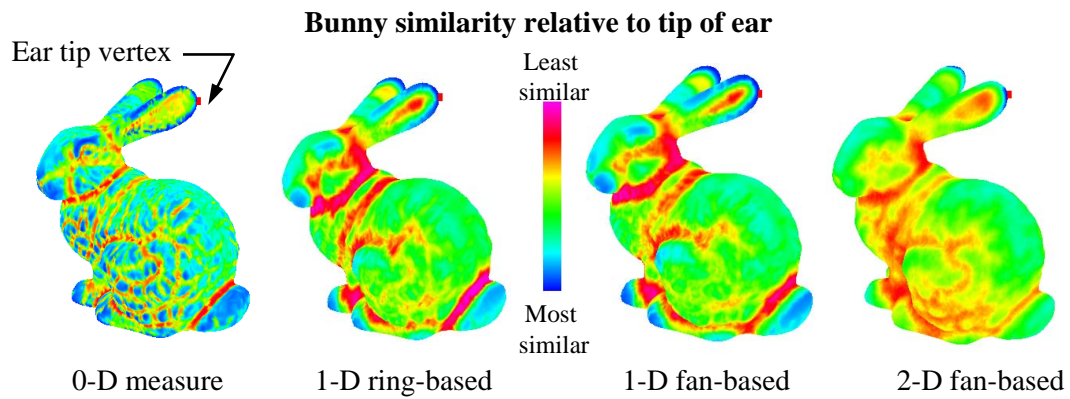


Figure 4.10: Similarity measure relative to a vertex on the tip of the ear of the Stanford Bunny. The color scale ranges from blue (high similarity to the selected point) to magenta (most dissimilar). The 0-D similarity has significant noise, while the 1-D methods isolate the tips of the ears much more cleanly. The 2-D method is even more discriminating, with similarity limited to the tip of the other ear.

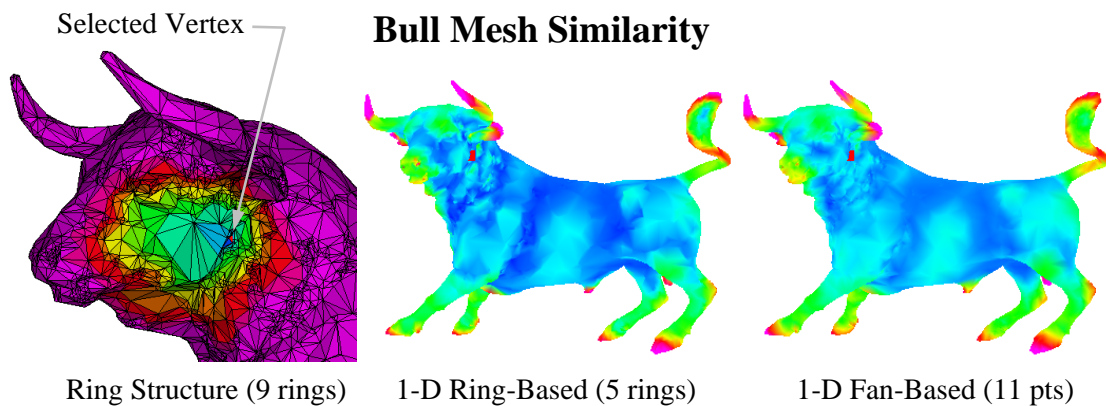


Figure 4.11: Similarity measure for the Bull mesh. The color scale ranges from blue (high similarity to the selected point) to magenta (most dissimilar). The view on the left shows that the ring structure is very non-symmetric about the selected vertex, due to the irregularity of the bull mesh. Even so, the ring-based and fan-based 1-D methods provide comparable similarity measures.

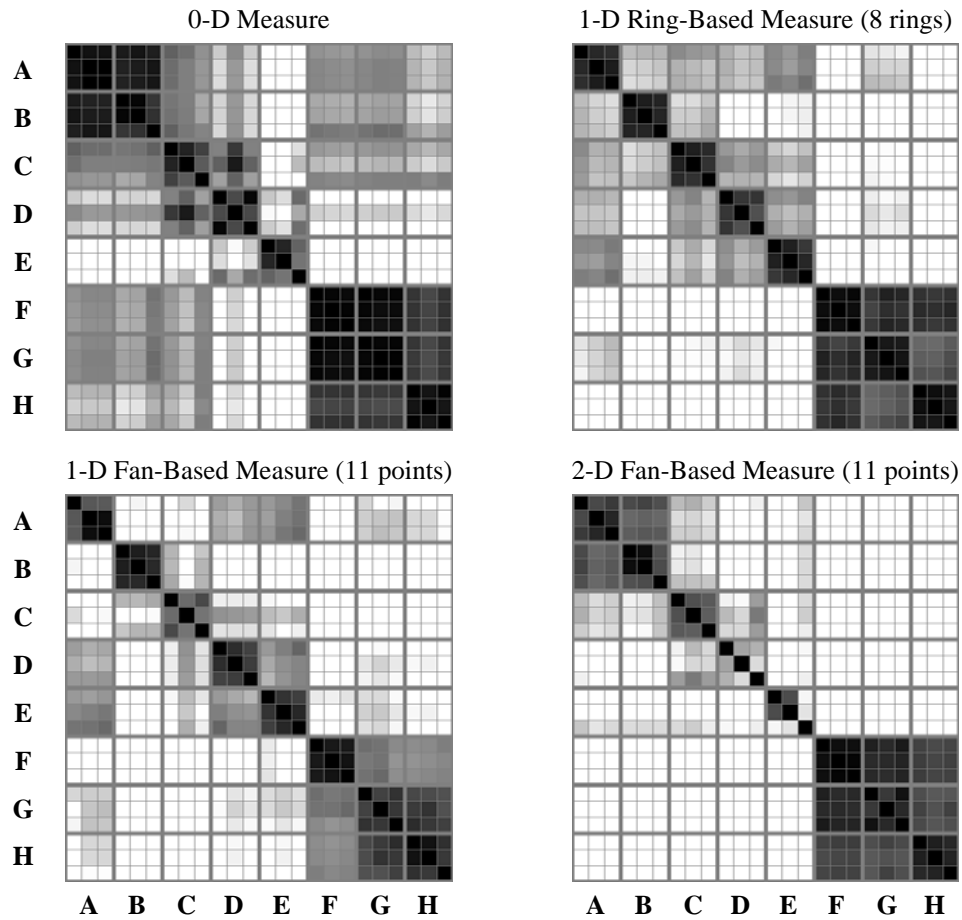
fan-based, and 2-D methods are shown in Figure 4.12. We chose eight groups on our three-lobed test surface, where each group contains three vertices. Group A is located in the concave region of one lobe. Groups B, C, and D are in three saddle regions occurring between pairs of lobes. Group E is in the crease along the rounded back of the main body. Groups F, G, and H are on convex regions of the body and two lobes respectively.

Comparing the distance grids allows us to evaluate various combinations of comparison functions. The comparison function having the most similarity between points of the same group (darkest  $3 \times 3$  boxes along the diagonal), with much less similarity (lighter) for dissimilar groups, was deemed best. The first five groups include concave regions, while the last three are primarily convex, so similarity between certain groups is expected.

Average mean curvature with the square root function applied to the average Gaussian curvature gave the best discrimination in our tests. The logarithmic function has a less significant effect, but this importance may depend on the nature of the curvature peaks. We varied the number of rings over a wide range, but for our test case, there was little change after about eight rings. Using fewer rings caused more degradation as we approach the 0-D curvature map. Using the minimum curvature or maximum curvature, instead of the average over the ring, performed poorly. Using a vector of both the minimum and maximum curvature in a ring did much better, but was not quite as effective as the average.

The 1-D ring-based method generates the highest degree of self-similarity within the groups. The 1-D fan-based method does not do quite as well within groups, but is good at distinguishing between the groups. The 0-D method does not differentiate between Groups A and B, and has poor self-similarity for Groups C through E. All three methods have just subtle differences for the last three groups. Overall, the ring-based 1-D method most consistently indicates more similarity within the group than between groups.

**Test Surface Similarity Comparisons for Vertex Groups (3 Vertices per Group)**



Group A – Concave region at end of a lobe  
 Group B, C, D – Saddle regions between  
 pairs of lobes

Group E – Crease along edge of rounded back  
 Group F, G, H – Convex regions of the back  
 and two lobes

Figure 4.12: Distance grids for select points. The similarity within groups, indicated by the darkest  $3 \times 3$  boxes along the diagonal, and dissimilarity between groups, based on lighter off-diagonal squares, was most consistent for the 1-D ring-based measure.

### 4.2.2 Applying Curvature Maps to Other Objects

To test how our new similarity measure works in practice, we apply it to a mesh of the Stanford Bunny. The bunny has a much more irregular surface, with regions of similar curvature, but quite a bit of local curvature variation. As Figure 4.10 (far left) shows, the 0-D (point curvature) similarity is very noisy due to these local curvature variations. The 1-D ring-based similarity measure (second from left) was generated from Gaussian curvature with log and square root functions and mean curvature with a log function applied, compared over eight rings. The same functions applied to the 1-D similarity based on eleven fan points and 2-D similarity are shown in the second from the right and far right images of Figure 4.10 respectively. The results are consistent with our test surface, i.e., the 0-D method is extremely noisy, both 1-D methods identify much smaller and more consistent regions of similarity. The 2-D method has even more differentiation between ear tip points and points not on the ear tip, with similarity indicated only for the tip of the other ear.

We also apply curvature maps to the mesh of a bull. This mesh is highly irregular, causing the ring structure to be asymmetric about the selected vertex, as shown in Figure 4.11. However, the ring-based and fan-based 1-D methods still provide similar results.

### 4.2.3 Efficiency Comparison

We also made comparisons of the speed of the methods for the test shape and the bunny mesh. Table 4.1 contains pre-processing times for computing curvature on the mesh, creating a ring-based curvature map, and creating a fan-based curvature map. All times are per mesh vertex. Identifying the ring structure around each vertex is included in the ring-based map times, and fan generation time is added to the map creation time for the fan-based maps. Table 4.2 shows the times for computing the similarity of each point of the mesh relative to a selected point, normalized by the number of vertices. The 1-D and 2-D methods were timed for four, eight, and eleven rings/points. All times were computed on a 1.7 GHz Pentium M processor. Some inaccuracy in the smaller times for the test shape is due to approaching the

Table 4.1: Preprocessing Times (milliseconds per Vertex) for 1.7 GHz Pentium M Processor

<i>Preprocessing Times - msec/vertex</i>		
	Test Shape	Bunny
Compute Curvature	1.5	1.6
Ring-based Map	1.8	30.0
Fan-based Map	3.4 to 10.4†	5.2 to 26.1†

† Time is proportional to the physical length of fan spokes

Table 4.2: Comparison Times (microseconds per Vertex) for 1.7 GHz Pentium M Processor

<i>Comparison Times - <math>\mu</math>sec/vertex</i>		
Comparison Method	Test Shape	Bunny
0-D (Point curvature)	2.9	1.1
1-D Ring-based Map (4 pts)	6.4	4.6
1-D Ring-based Map (8 pts)	10.1	8.6
1-D Ring-based Map (11 pts)	12.1	12.1
1-D Fan-based Map (4 pts)	7.5	4.4
1-D Fan-based Map (8 pts)	11.0	8.5
1-D Fan-based Map (11 pts)	12.4	11.5
2-D Map (4 pts)	672	671
2-D Map (8 pts)	1283	1287
2-D Map (11 pts)	1584	1597

resolution of our timing algorithm. The comparison functions are much faster than the pre-processing step, with the 0-D and 1-D methods a few orders of magnitude faster than the 2-D comparisons.

#### 4.2.4 Finding Unique Features with Curvature Maps

In order to look for key features in the mesh, we look for the groups of points that are least similar to the remaining points. For each point, we compute its similarity with respect to all other points, and then sort these by decreasing similarity. A Gaussian

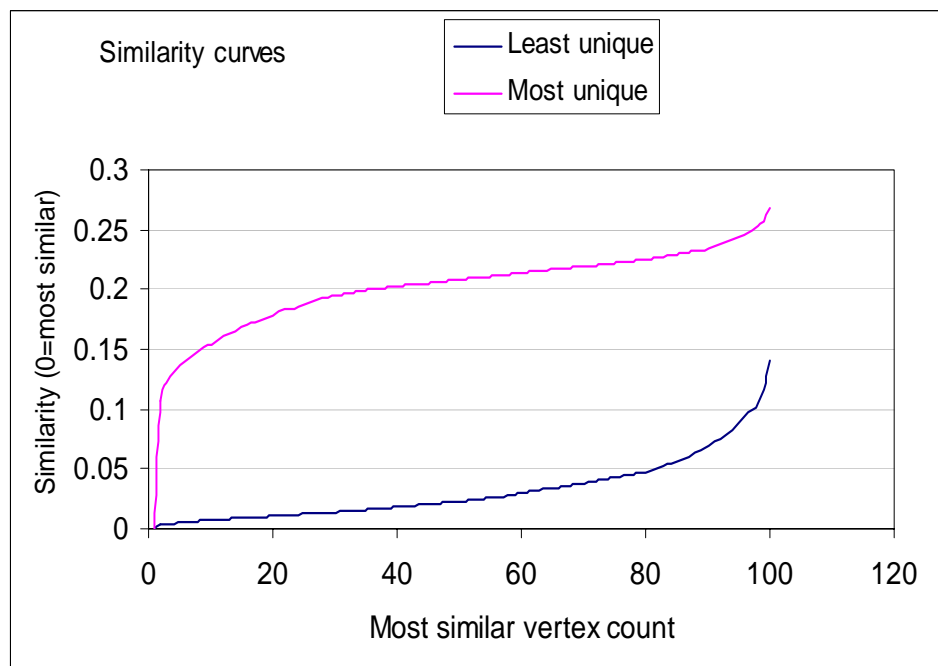


Figure 4.13: The similarity curves for the most and least similar vertices.

function is applied to the sorted similarity curves, and the resulting contribution, which represents a non-parametric kernel density estimate, quantifies how many other points the given point is similar to. The smallest values indicate the points most different from the general population. The sorted similarity curves for the highest and lowest similarity density are shown in Figure 4.13 (the kernel value was set to be 0.05 at the 100th point). The three-hundred most unique points are highlighted for the 0-D (left) and 1-D (right) methods in Figure 4.14. The 1-D method picks up more consistent point groupings than the 0-D method. This is apparent in both the neck region and on the tail.

### 4.2.5 Guidelines for Computing Similarity

The ring-based and fan-based 1-D methods are comparable in ability to discriminate, comparison times, and setup times. Ring-based methods are more appropriate for larger regions, provided the mesh is fairly uniform. If storage space is not an issue,

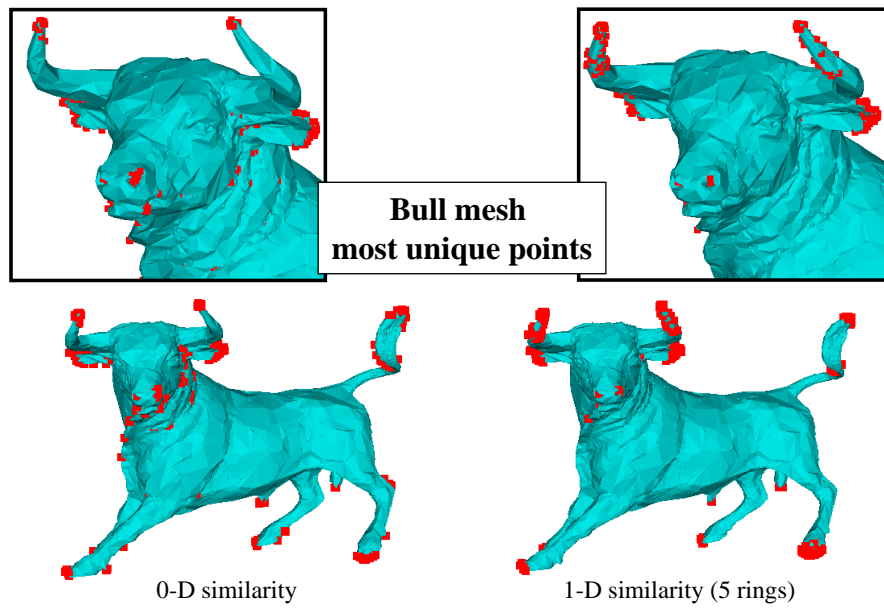


Figure 4.14: The three-hundred most unique points based on similarity to to all other points. The 0-D method (left) picks up most of the peak curvatures, but finds a lot of isolated points in the neck and face region. The 1-D method (right) finds consistent groups of points reflecting key features in the mesh.

the fan-based 1-D method provides a more consistent comparison for smaller local regions.

The 1-D ring-based method can also be used to pre-process a mesh to identify regions that are similar. The more expensive, but more exact, 2-D method can then be applied just to these regions.

In summary, it is preferable to use the ring-based 1-D method for comparing larger local regions, as long as the mesh quality is reasonable. The slower 2-D methods can be reserved for the final stage when exact matching is required.

### 4.3 Chapter Summary

The curvature map is a new method for comparing local shape based on surface curvature. It has been applied as a 1-D method on N-ring neighborhoods and as a 1-D or 2-D method on Geodesic fans. Point curvature (0-D) methods do a poor job of distinguishing between local regions. Curvature maps demonstrate improved capability to discriminate shape as compared to these 0-D methods.

The radius to use when generating the curvature map is case dependent, but a general guideline is to use a radius slightly larger (ten to twenty percent) than the largest expected feature size. Similarity can still be evaluated for any size up to this radius. The only drawback due to using a larger radius is an increase in the required storage and computation time. Determining the size over which to compare similarity is still an open issue, but a multi-scale approach, such as the one described in the next chapter, is one way to overcome this issue.

Curvature maps offer a valuable capability to differentiate local shapes. These methods will be applied to the shape matching problem to identify corresponding points based on shape similarity. These comparison methods could also be extended to account for shape similarity when objects or portions of objects are scaled differently.



## Chapter 5

# Feature Detection Using Multi-Scale Graph Cuts

Previous object matching approaches often have required some type of user interaction to select features [183]. Manual selection of corresponding features and subjective determination of the difference between objects are time consuming processes requiring a high level of expertise. As advances in 3-D scanning capability increase access to 3-D shape data, automatic detection of features is necessary to analyze and compare shapes effectively. But this brings us to two problems. The first is the fundamental question: ‘What constitutes a feature?’ Once that question is answered, the next question is ‘How do we detect features automatically?’

Man-made objects often have well-defined features such as edges, but features of natural shapes, such as the wrist bones shown in Figure 5.1, are more subjective. Such shapes can have subtle variations, the importance of which may not be obvious. We expect peaks, pits, ridges, and valleys to be useful features for shape matching applications. Furthermore, important features may be of various sizes within one object. These features may or may not be unique, as long as there are enough features to resolve any ambiguities during shape matching.

Our goal is to detect subtle shape features in a robust way with a fully automated process. In our shape matching process we match regions, and then determine selected points within the matched regions for point-to-point matching. Therefore, the desired output is a set of geometrically interesting regions that are sufficient for

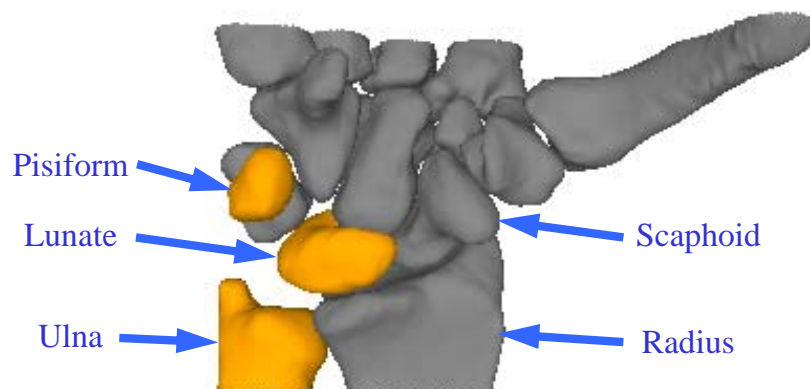


Figure 5.1: Bones making up the human wrist. Natural objects have subtle shape variations that are challenging to characterize.

shape matching. The feature detection process should be independent of the mesh resolution, relatively insensitive to noise, and should not depend on parameter tuning.

## 5.1 Feature Detection Process

The underlying concept for our feature detection algorithm is to use the sensitivity of the curvature map, combined with a robust segmentation approach in a multi-scale framework. For shape matching, detection of every feature is not required (indeed, we cannot even define every feature). Since our algorithm produces features that are ordered by strength, we will show that we can sort them and use them for our shape matching application in Chapter 6.

The curvature map at a point represents shape information for the point and its surrounding region. A min-cut/max-flow graph cut algorithm, popular for image segmentation tasks, is employed to identify features at various scales. Results from multiple cuts are combined in a novel manner to produce a final feature set. The multi-scale approach eliminates the need for user interaction, and for tuning parameters based on a particular application.

The proposed feature detection algorithm is robust to noise and mesh variations. The process is automatic, with no user controlled parameters. We demonstrate the

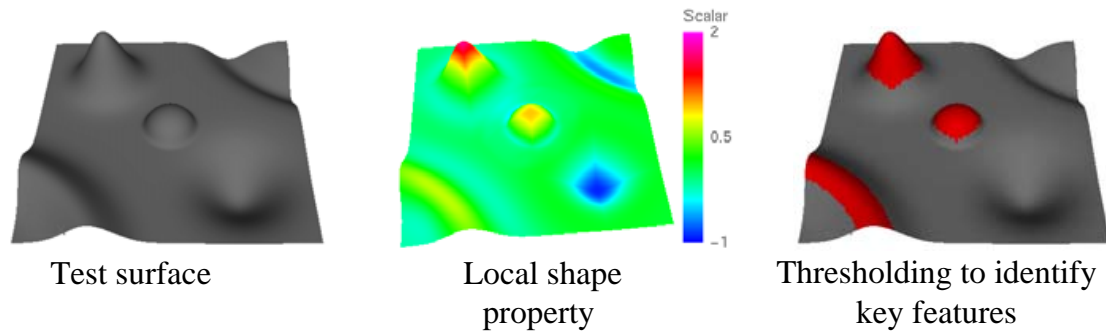


Figure 5.2: Feature detection test surface. Left: Surface shape with peaks, pit, ridge, and valley. Center: Mean curvature scalar function. Right: Features highlighted by selecting a function threshold. With the proper threshold, this function can highlight useful features, however, the threshold must be found by experimentation.

algorithm on several shapes represented by triangular meshes. These shapes include a test shape with and without noise, data from face scans, and bone data.

## 5.2 Local Shape Property

Basic feature shapes include the peak, pit, ridge, and valley. The common link between these features is the dependence on the magnitude of the mean curvature. In order to identify these features, we need some measure of the likelihood that a vertex should be classified as belonging to one of these features. This measure needs to incorporate information about the neighborhood around the vertex, as well as at the vertex itself. The curvature map [61] provides this context.

Our local shape property  $S$  is

$$S(p) = \int_0^R \text{Mean}(Kmap(p))(r) dr$$

where  $R$  represents the radius corresponding to the maximum feature size.  $Kmap(p)$  represents the 1-D curvature map, and  $\text{Mean}(Kmap(p))$  and  $\text{Gauss}(Kmap(p))$  indicate the mean and Gaussian functions of the curvature map respectively. A test

surface, colored by the local shape property, and features resulting from applying a threshold to  $S$  are shown in Figure 5.2.

Functions based on the Gaussian curvature component of the curvature map and combinations of mean and Gaussian curvature were also considered, but given a suitable threshold, the mean curvature function gave the most consistent identification of the features in the test case. This is due to the primary relationship of these features to the mean curvature. Experimenting with the range over which the curves were integrated by finding the sign changes in the function value did not improve the ability to detect features.

Although this local shape property often highlights the expected features, finding an appropriate threshold requires manual adjustment, and the results still depend on the curvature map radius  $R$ . In addition, no single threshold could extract both the positive curvature features (peak and ridge) and the negative curvature features (pit and valley). These factors motivated our search for an improved feature detection approach.

### 5.3 Multi-Scale Feature Detection

Our local shape property was combined with the min-cut/max-flow graph cutting technique of Boykov and Kolmogorov [20] to create a multi-scale approach for feature detection. The primary benefit of the graph cut algorithm is its efficiency, and the compact boundary produced. We use two parameters,  $R$  and  $\alpha$ , where  $R$  is the radius for our local shape property and  $\alpha$  is a weighting factor. We note that when the graph cut algorithm is executed with different values for these parameters, different features may be identified. This motivates an approach which runs the graph cut algorithm multiple times, varying these parameters, and extracts the most significant features overall.

As discussed previously, we are interested in features that correlate most strongly with the magnitude of mean curvature, so we first run the graph cut algorithm on the absolute value of the shape property. Applying the default graph cut weight detects only the most prominent features. To detect less prominent features, a range

---

**Algorithm 1** Multi-Scale Feature Detection
 

---

```

Read Curvature Map ( $K_{map}$ ) for Mesh  $M$ 
for  $K_{map}$  radius  $R$  from  $R_{min}$  to  $R_{max}$  do
  Compute  $S$  as the integral of the  $K_{map}$  mean curvature component from 0 to  $R$ 
  for a range of weight factor  $\alpha$  do
    Create graph cuts  $C_{abs}, C_{pos}, C_{neg}$  on the positive, negative, and absolute value
    of  $S$ 
    Identify the features in  $C_{abs}, C_{pos}, C_{neg}$ 
    for each vertex  $v$  in Mesh  $M$  do
      Count feature occurrences  $N_{abs}, N_{pos}, N_{neg}$  in  $C_{abs}, C_{pos}, C_{neg}$ 
    end for
    for each edge do
      Count how many times both endpoints occur in the same region
      Note: Used to generate edge weights for the later max-flow/min-cut runs
    end for
  end for
end for
for a range of weight factor  $\alpha$  do
  Create graph cuts  $C_{abs}, C_{pos}, C_{neg}$  from normalized counts  $N_{abs}, N_{pos}, N_{neg}$ 
  Identify and merge features from  $C_{abs}, C_{pos}, C_{neg}$  into composite feature sets
   $G_{abs}, G_{pos}, G_{neg}$ 
end for
Merge  $G_{neg}$  and  $G_{pos}$  into  $G_{abs}$  to create the Master Feature Set  $G$ 

```

---

of weighting factors (values of  $\alpha$ ) are applied. Since the larger of the positive or negative shape property magnitudes may dominate the absolute value graph cuts, the graph cut algorithm is also applied separately to the positive and negative values of the local shape property. So the graph cut technique is applied three times for each combination of parameter values ( $R$  and  $\alpha$ ) in order to ensure capture of key positive and negative curvature features. For  $N_R$  values of  $R$  and  $N_\alpha$  values of  $\alpha$ , this results in  $N_R \times N_\alpha$  sets of features for each of the three categories of graph cuts: absolute value, positive, and negative.

The variations of curvature map radii and scale factors for the three graph cut categories generate a large number of possible feature sets. In order to simplify the process of extracting a master feature set from this data, we first count the number of times each vertex is identified as part of a feature in each of these categories. Then we run the graph cut algorithm on the normalized frequency counts, again varying

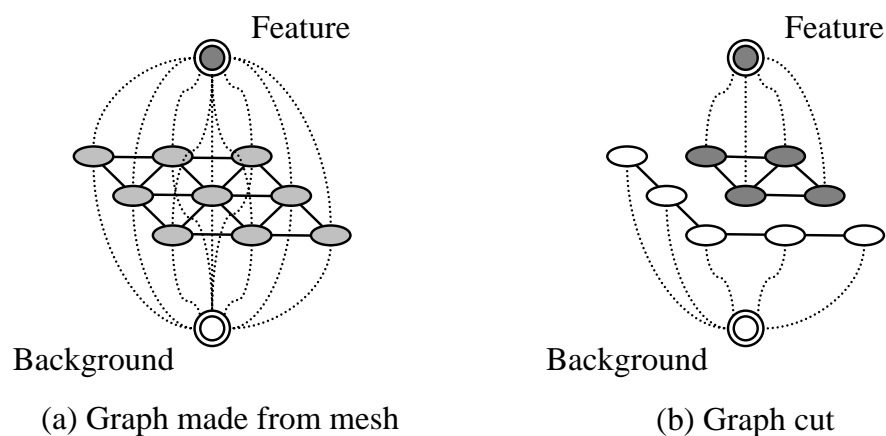


Figure 5.3: The min-cut/max-flow graph cutting algorithm finds an optimal separation of the vertices of a mesh into a feature group and a background group. The cut is based on weights assigned to the mesh edges (solid lines) and to edges connecting the graph vertices to the feature and background nodes (dotted lines).

the scale factor. This yields a smaller set of features that are then merged to create the master feature set. This process is shown in Algorithm 1.

### 5.3.1 Graph Cut Parameters

In order to run the graph cut algorithm, weights need to be assigned to the mesh edges and to connections from the mesh vertices to a ‘feature’ node and a ‘background’ node, as shown in Figure 5.3. These weights, given in Table 5.1, represent the cost of breaking the edge in order to separate the graph vertices into the feature and background sets. Note that the vertices within a set need not form a single contiguous region of the graph.

Once a graph cut has been created, contiguous groups of vertices are extracted from all vertices associated with the feature node of the graph cut. These contiguous groups of vertices are our features. Figure 5.4 shows features extracted from selected graph cuts of a test surface.

Table 5.1: Graph Cut Weights

	<i>Shape Property Based</i>		<i>Feature Frequency Based</i>	
<i>Edge</i>	<i>Weight(cost)</i>	<i>for</i>	<i>Weight(cost)</i>	<i>for</i>
$\{p, q\}$	$E_1\{p, q\}$	$\{p, q\} \in Edges$	$E_2\{p, q\}$	$\{p, q\} \in Edges$
$\{p, F\}$	$-\log(1 - S(p))\sqrt{\alpha}$	$\forall p$	$-\log(1 - N(p))\sqrt{\alpha}$	$\forall p$
$\{p, B\}$	$-\log(S(p))/\sqrt{\alpha}$	$\forall p$	$-\log(N(p))/\sqrt{\alpha}$	$\forall p$

$$E_1 = \exp\left(-\frac{(S(p)-S(q))^2}{2(dist(p,q)\sigma)^2}\right) \frac{1}{dist(p,q)} \text{ if } S(p)S(q) < 0, \frac{1}{dist(p,q)} \text{ otherwise.}$$

$$E_2 = \exp\left(-\frac{N_T - N_S}{N_T}\right) \text{ where } N_T \text{ is the number of cuts, and } N_S \text{ is the number for which } Feature_p = Feature_q.$$

$F$  and  $B$  are the feature and background nodes respectively.

$p, q$  are mesh vertices.

$\alpha$  is the scale factor for the feature node weights.

$S(p)$  is the local shape property value at  $p$ , limited to  $\epsilon \leq S(p) \leq 1 - \epsilon$ .

$N(p)$  is the normalized frequency count at  $p$ .

### 5.3.2 Multi-Scale Parameters

The two parameters that are varied are the curvature map radius  $R$  and the weighting factor  $\alpha$ .  $R$  is varied from small to large, with the size of the largest region based on the radius  $R_{max}$  used for the original curvature map calculation.  $R_{max}$  is assumed to be large enough to capture the largest desired feature. For example, on our human face scans, we use a maximum radius of about two inches. Smaller radii are defined by successively scaling by  $1/\sqrt{2}$ . For our cases, using eight levels was sufficient to make the minimum  $R$  comparable to the shortest edge of the mesh.

The weights for the connections to the feature node are scaled by  $\sqrt{\alpha}$ , while the connections to the background node are scaled by  $1/\sqrt{\alpha}$ . We determine  $\alpha$  by trial and error. We first decrease  $\alpha$  until we get only one group. Then we increase  $\alpha$  until the number of groups reaches a peak. We then take uniformly spaced values for  $\alpha$  in this range. For our examples, we use ten divisions. Thus, the 8  $K_{map}$  radii cross the 10 scale factors results in 80 graph cuts for each category, for a total of 240 graph cuts. Fortunately, the graph cut algorithm is very efficient, with the 240 graph cuts on a 10,000 vertex mesh taking less than 40 seconds on a 2.8GHz Pentium 4 processor.

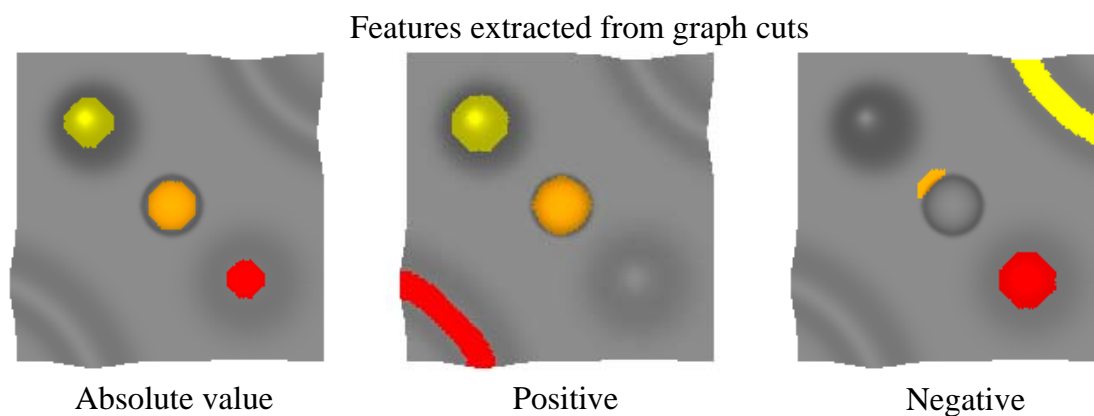


Figure 5.4: Graph cuts generated by the min-cut/max-flow algorithm on the local shape property for three graph cut categories: absolute value (left), negative (center), and positive (right). The absolute value graph cut picks up the peak and pit features, while the valley feature is only found in the negative graph cut and the ridge feature is only found in the positive graph cut.

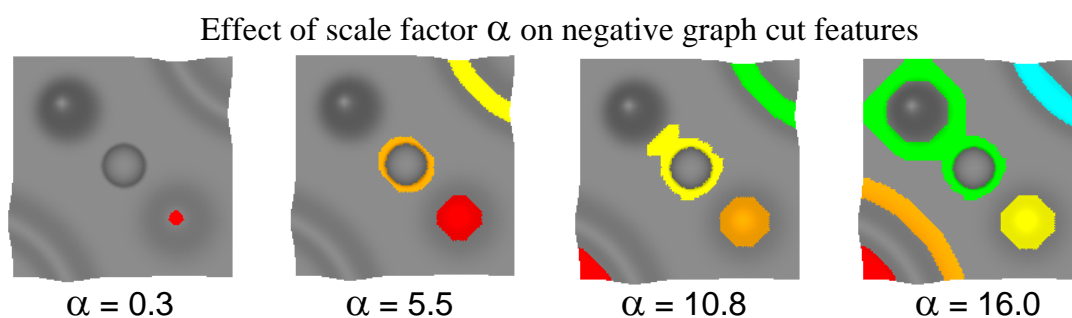


Figure 5.5: Effect of the scale factor  $\alpha$  on features identified using the min-cut/max-flow graph cutting algorithm. Representative cuts from the negative of the local shape property are shown. As  $\alpha$  increases, more features are detected, and existing features become larger. At larger  $\alpha$  the saddle region at the base of the peaks is detected.



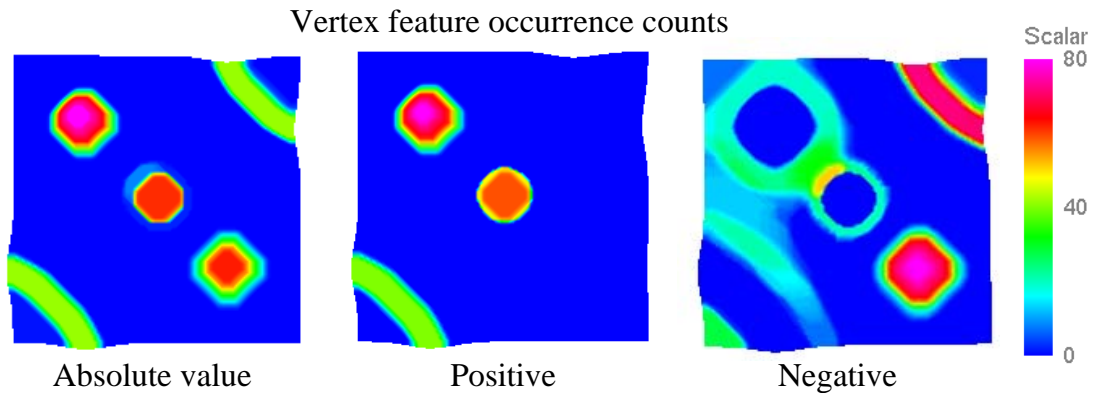


Figure 5.6: Feature counts for the absolute value (left), negative (center), and positive (right) graph cut categories. Maintaining separate frequency counts for the three graph cut categories allows extraction of more well-defined features.

Figure 5.5 shows the groups produced for selected scale factors for the negative graph cuts of our local shape property with a curvature map radius of 3.8. As the scale factor is increased, individual features tend to get larger, and new features may show up. The feature frequency counts are shown in Figure 5.6.

### 5.3.3 Group Merging Criteria

---

**Algorithm 2** Merging Feature Set  $A$  into Feature Set  $B$

---

**Require:** Feature Sets  $A$  and  $B$  on Mesh  $M$

```

for each feature  $F_i$  in Set  $A$  do
    Determine how many features  $n$  in Set  $B$  overlap  $F_i$ 
    if  $n = 0$  then
        Add  $F_i$  as an additional feature in Set  $B$ 
    else if  $n = 1$  then
        Take the union of  $F_i$  with its overlapping feature in Set  $B$ 
    else
        Ignore the feature  $F_i$ 
    end if
end for

```

---

A simple greedy approach, as shown in Algorithm 2, is used to merge feature groups together. When combining cuts from progressively larger source weights to form

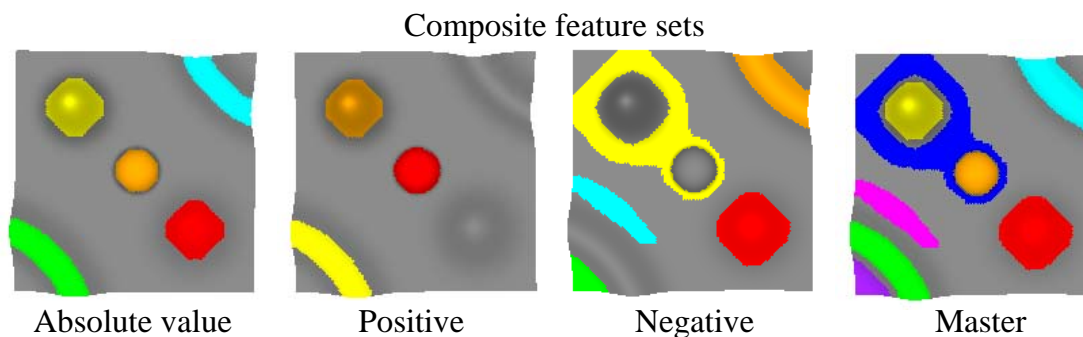


Figure 5.7: Composite feature sets for the absolute value, negative, and positive graph cuts, and the master feature set made by merging them.

composite feature sets, the groups tend to grow, but without allowing neighboring groups to merge. This makes sure all of the features do not get merged together, as might occur for a very large scale factor. The same algorithm is applied when merging the composite feature sets for the absolute value, negative, and positive portions of the function. The composite feature sets for each of the function variations, and the final feature set made by combining them, are shown in Figure 5.7.

## 5.4 Results

This approach has been applied to several different types of meshes. Figure 5.8 shows the previous test surface with the addition of Gaussian noise. In spite of the noise, the feature structure is very similar to that of the case without noise shown in Figure 5.7, especially for the main features.

Figure 5.9 shows feature detection applied to a low resolution scan of a human face. The coarseness of the mesh has a smoothing effect that eliminates many details. It also highlights the benefit of running the absolute value, positive, and negative graph cuts to identify features for the master set that would be missed otherwise. In Figure 5.10 we compare our feature detection method with segmentation based on the signs of the mean and Gaussian curvature for a higher resolution human face. Even after smoothing the curvature data, the segmentation on the left shows quite

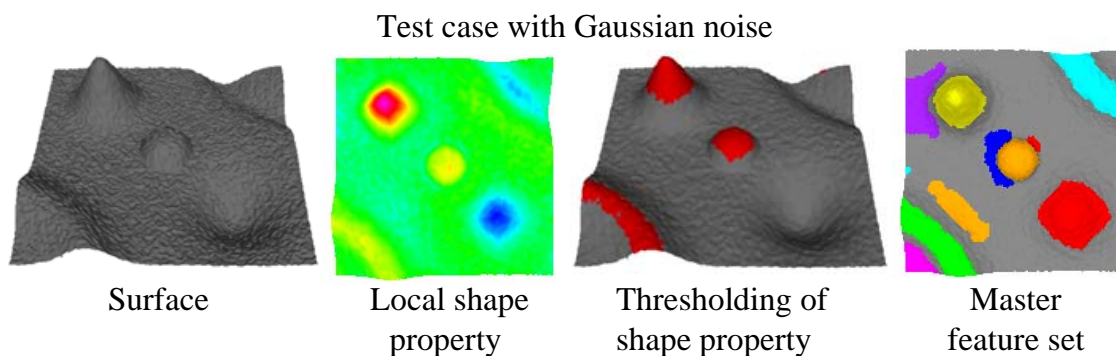


Figure 5.8: Test case with Gaussian noise added. The function and final feature set are similar to the test case without noise.

a bit of noise. This is improved by setting a zero threshold so that large regions of low curvature are separated from the higher curvature features, as shown in the center segmentation. However, the resulting features depend strongly on the amount of smoothing applied and the zero threshold, and are still less well-defined than the master feature set shown on the right.

Features for the Stanford bunny are presented in Figure 5.11. While this case produced a number of very small features, the larger feature regions, such as in the ears, face, feet, and tail, seem to be features that could be useful for shape matching.

The features for several bone meshes are shown in Figure 5.12. These bones generally have fairly subtle features. Note the similarity of the feature layout for Ulna A (View 2) and Ulna B in spite of a significant difference in mesh resolution and being from different subjects.

## 5.5 Chapter Summary

This chapter presented a two-step multi-scale feature detection approach that uses a local shape function based on the curvature map. This feature detection approach employs an efficient min-cut/max-flow graph cutting algorithm and greedy algorithm to merge feature sets. The method is robust with respect to noise, and consistently

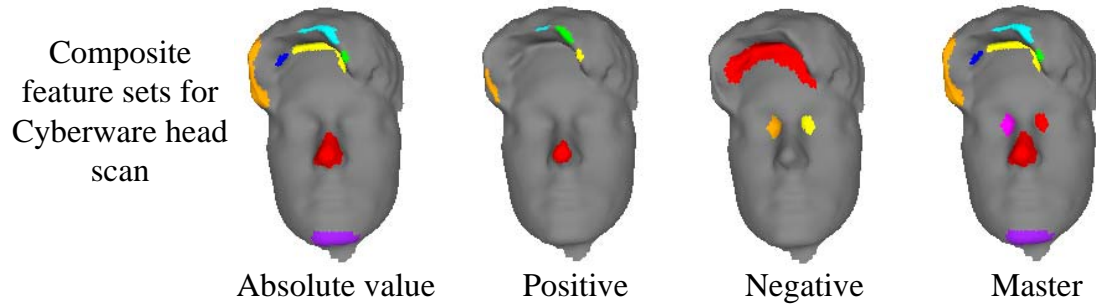


Figure 5.9: Features detected on a Cyberware low resolution female face scan. The absolute value graph cuts pick up the nose chin and hair features, while the negative cuts detect the eyes. In spite of the smoothness of the mesh, master feature set captures the prominent features of the face.

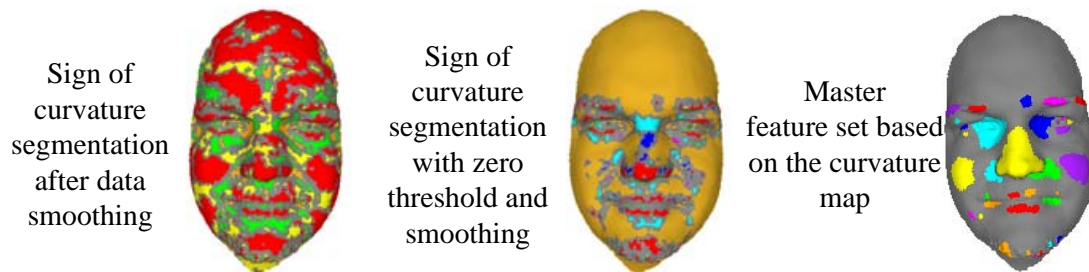


Figure 5.10: Comparison of graph cut feature detection with sign of curvature segmentation for a high resolution Cyberware face scan. Before coloring by the sign of Gaussian and mean curvature, the curvature values were smoothed. The segmentation in the center uses a zero threshold to separate low curvature regions from higher curvature features. However, the master feature set provides more well-defined features.

Shape features for the Stanford bunny (two views)

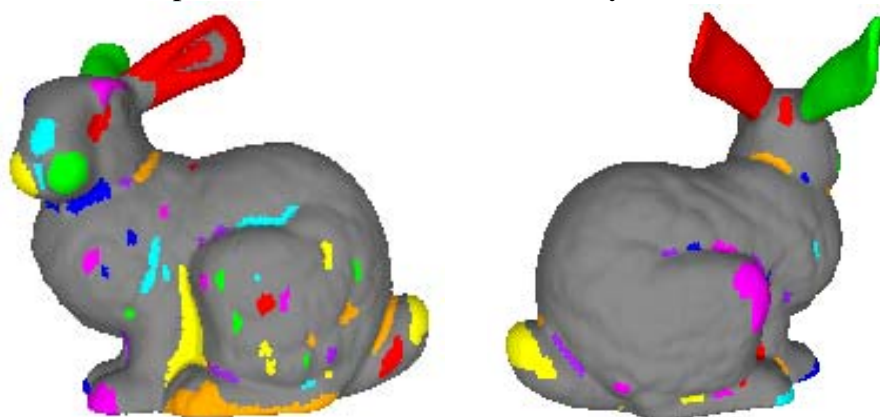


Figure 5.11: Features detected for the Stanford bunny. Several features, such as the large sections of the ears and the features in the face region, are very intuitive.

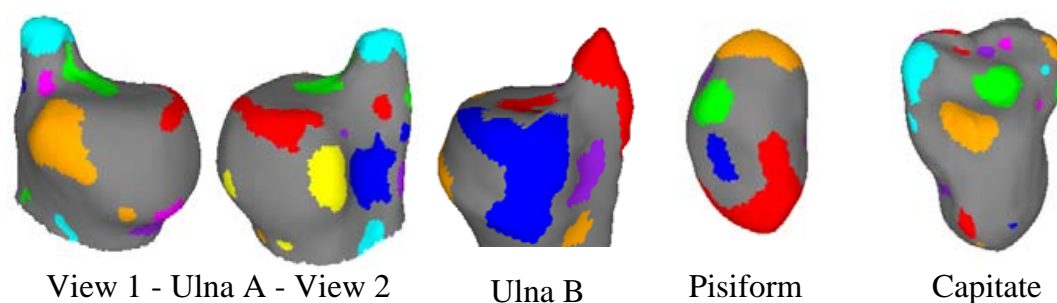


Figure 5.12: Master feature sets for selected bone meshes. The Ulna is challenging due to the limited number of pronounced features and the significant difference between the scales of the features. Similar features were detected for cases A and B even though the resolution of the meshes is very different. Reasonable features were also identified for the Pisiform (second from right) and Capitate (far right).

yields a reasonable set of features. Most importantly, there is no user interaction or parameter tuning required.

The method could benefit from alternate algorithms for merging feature sets. The greedy approach works fairly well, but may cause some over-segmentation, since it does not allow two features to coalesce into one, which might be desirable in some instances.

Because the local shape property is based on the integral of mean curvature, it detects primarily higher curvature features. While less useful for identifying shape similarity, the capability to detect flat or nearly flat regions might further reduce the search space for feature detection and shape matching tasks.

## Chapter 6

# Shape Matching

The previous chapter presented a robust feature extraction process. This chapter develops a shape matching approach that uses extracted features for both an initial alignment of 3-D surfaces, and for refining the correspondence between those surfaces.

In our target applications, differences between the objects can range from subtle changes in fine details to missing or added elements. Our shape matching method finds corresponding features and aligns the objects based on those features. The alignment is used to geometrically check the feature correspondence. Then feature point correspondence is extended to the entire object by generating a common parameterization. This produces a point-to-point correspondence between the objects. This correspondence can be used to analyze the relative size and location of the features.

The overall shape matching process is shown in Figure 6.1. The work to date has been limited to pair-wise comparison of objects, but is easily extended to compare more than two objects. For each object, a set of features is identified using the techniques of Chapter 5. The similarity of these features is compared to generate a candidate list of corresponding feature pairs. The top candidates from this list are used to generate an initial alignment, which is then refined using the feature data. The initial alignment produces a correspondence that is again refined using feature similarity and surface parameterizations are used to define the final correspondence.

Creating the feature pairs and finding the best initial alignment is fully automatic. The user currently chooses which of the refinement options to apply, but there are no parameters for the refinement process. The user chooses how many additional

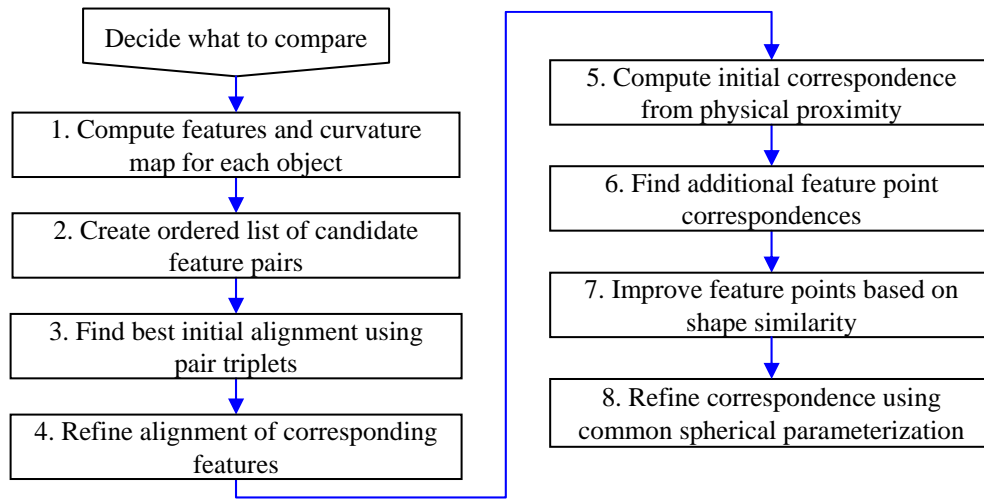


Figure 6.1: Shape matching process steps.

feature points are needed and there are parameters that limit the point movement for similarity improvement.

## 6.1 Shape Comparison Goals

Our primary goal is to find the point-to-point correspondence between the objects. This correspondence will be used to classify the nature and magnitude of shape differences, so there are several desirable properties.

- The correspondence should be 1-to-1 and should also be continuous. This is a fundamental requirement for a valid common parameterization.
- It is also desirable that the correspondence be smooth. In general, this means that relative distances between pairs of points on one object should be similar to the relative distances between their corresponding points on the other object. Locations where the correspondence is not smooth should identify differences between the objects.
- The correspondence should reflect shape similarity as much as practical. This is based on the intuition that because the objects being compared are similar,



some corresponding portions of the objects should be similar, and for these portions, corresponding points should reflect this similarity. This is important to ensure that lack of similarity at corresponding points represents differences between the objects, and not just a poor definition of corresponding points.

### 6.1.1 Technical Challenges

One critical issue for automated shape matching is the quantity of data to be processed and the difficulty in describing shape in a meaningful way. The possible variations in shape are unlimited. By extracting a small number of features, ordering features by strength, and ordering potential feature pairs based on similarity, the scope of the problem can be drastically reduced to enable very efficient matching.

Another critical issue affecting automated shape matching is noise. Noise can introduce error into the calculation of properties such as curvature, as presented in Chapter 3. Typically, smoothing is applied to reduce noise, but smoothing can also obscure small magnitude or very localized features. As a result, it can affect the quality of the alignment calculation. This issue is addressed by developing features, based on improved underlying curvature calculation, that are less susceptible to noise.

Outliers can also impact shape matching. An outlier is a point or feature very different from its neighbors or from the corresponding locations in the other object. For example, outliers may be features that occur in only one of the two objects. Outliers may also be caused by a severe instance of local noise or sampling error. Such outliers may produce large distances that are likely to throw off shape matching methods that try to minimize a distance measure.

## 6.2 Determining Object Correspondence

The correspondence between objects determines for each point on one object, what the corresponding point is on the other object. One approach for generating this correspondence is to physically align the objects and identify corresponding points

based on their proximity. Correspondence based on the best physical alignment can be very good if the object shapes are very similar. The more the shapes differ, the higher the likelihood that incorrect correspondences may be found. Examples of incorrect correspondences are multiple points on one object mapping to the same corresponding point on the other object, or discontinuities where points ‘close together’ on one object map to points not close together on the other object.

The approach used in this dissertation generates a correspondence between objects based on a mapping of the objects to a common parametric domain. This common parametric domain for the objects reduces the reliance on their physical alignment. By definition, a point in the common parametric domain maps to two corresponding points, one on each object. Using manifold representations avoids discontinuous mappings. The challenge is to generate mappings that preserve certain key correspondences, and provide reasonable correspondences between other points on the objects. The mapping to the common parametric domain is based on alignment of some number of reference point pairs.

### 6.2.1 Feature Correspondence

To define the common parametric domain, we identify reference point pairs by establishing correspondence between features of the objects. Each mesh vertex could represent a feature. Even with the same number of vertices on each object, there may still not be a good one-to-one correspondence between the vertices. A more practical alternative is to extract features from the surface of the object. The set of features is generally much smaller than the number of vertices in the object representation, and features also have the benefit that they can represent some meaningful region or property of the object surface. It should be noted that in general there is no guarantee that there will be a one-to-one correspondence between features.

The features that we generate are described in Chapter 5. Our approach produces a relatively small number of features that are a function of the surface shape and do not depend heavily on the resolution of the mesh.

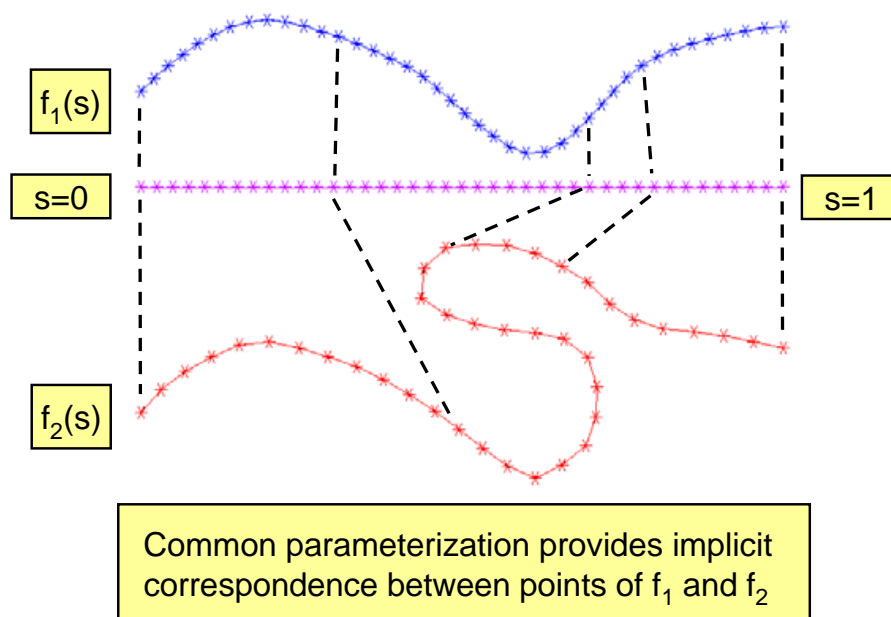


Figure 6.2: Values of  $s$  in  $[0, 1]$  define a correspondence between a point of  $f_1$  (blue) and a point of  $f_2$  (red).

## 6.2.2 Surface Parameterization

Surface parameterization establishes a correspondence between the surface of a 3-D object and a parametric domain of the same genus. Surface parameterization is an extension of curve parameterization from 1-D to 2-D.

Figure 6.2 shows how the common parameter  $s$ , represented by the points along the purple line, generates a correspondence between the points on  $f_1$  (blue curve) and  $f_2$  (red curve). The dashed lines connect the end points and three intermediate points of  $f_1$  and  $f_2$  to their common parameter values.

The mappings from the objects to the common parametric domain define a chain of transformations that take points of one object to their corresponding points on the other object. For example, Praun and Hoppe [140] use a mapping from a mesh to a sphere, and from the sphere to a regular polyhedra (tetrahedron, cube, octahedron, etc.) for use in texture mapping. To be useful, the mappings must be one-to-one and onto (i.e., a bijection) so that the inverse mappings can be found.

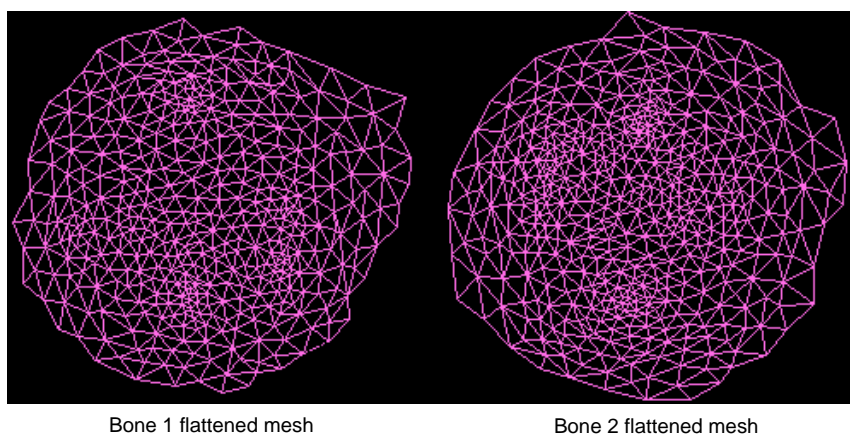


Figure 6.3: Surface meshes flattened into the plane for two bones.

There may be more than one possible parameterization approach for a given object. For example, a bone represented by a genus zero surface may be mapped to a sphere, or a subset the bone, such as one end, may be mapped to a plane. The following sections look at some of the issues for planar and spherical parameterizations.

### Planar Mapping

For a surface that is homeomorphic to a disc, a parameterization can be generating by flattening the surface to a plane using the techniques described in Section 2.3. The flattened mesh has the same connectivity as the original mesh, and represents a parameterization of the original mesh. Then corresponding feature points can be aligned in the plane. One motivation for operating in the plane is that mapping functions, and routines that adjust the parameterization are simpler to implement in two-dimensions.

When the mesh is mapped to the plane, there will be distortion of the edge lengths and areas of the triangular faces. Techniques such as overlay smoothing [157] can be used to minimize area distortion, as shown in Figure 6.3. However, distortion cannot generally be eliminated. This is also true for other types of parameterization.

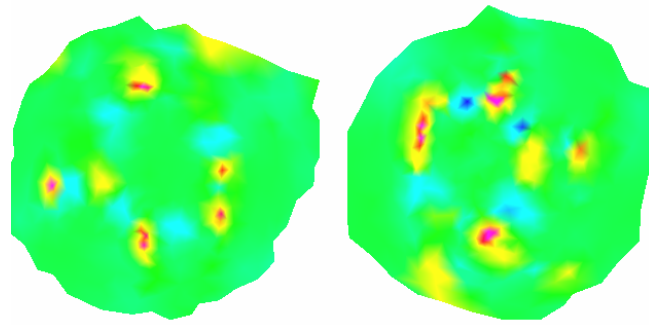


Figure 6.4: Flattened mesh with Gaussian curvature for the radius bones.

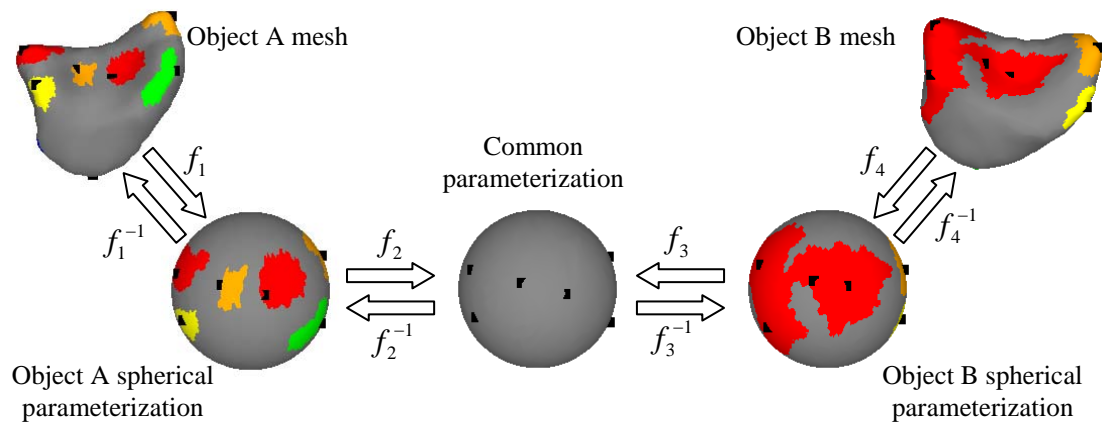


Figure 6.5: Mapping from mesh to spherical parameterizations to common parameterization.

Figure 6.4 shows the curvature from the 3-D object mapped to the flattened meshes. Curvature is just one example of a property that can be represented on the flattened mesh. Features or any other vertex or face based properties could be represented as well.

## Mapping to a Sphere

Figure 6.5 relates two objects through the mapping between their spherical parameterizations. The function  $f_1$  maps locations from *ObjectA* to its spherical parameterization, while  $f_1^{-1}$  (the inverse of  $f_1$ ) maps locations from the spherical parameterization back to *ObjectA*. Function  $f_4$  and its inverse perform similar mappings between *ObjectB* and its spherical parameterization. Functions  $f_2$  and  $f_3$  map between the common parameterization and the spherical parameterizations of *ObjectA* and *ObjectB* respectively. For the points  $P \in A$  and  $Q \in B$ , the correspondence between  $P$  and  $Q$  is defined by the composite mappings  $P = f_1^{-1}(f_2^{-1}(f_3(f_4(Q))))$ , and  $Q = f_4^{-1}(f_3^{-1}(f_2(f_1(P))))$ .

### 6.2.3 Beyond Rigid Body Transformations

In this research, we use rigid alignment only to generate coarse alignment and test for the best triple of feature pairs. From this coarse alignment and shape similarity, we compute correspondence directly via parameterization and avoid more complex 3-D transformations.

We have also incorporated an iterative closest point algorithm as an additional physical alignment option. The iterative closest point (ICP) [12] algorithm is a standard algorithm for aligning point sets using a rigid transformation. The ICP algorithm that we obtained from Rusinkiewicz includes options to compute the best rigid transformation, the best rigid plus scaling transformation, or the best affine transformation. The ICP algorithm is not formally part of our shape matching process, but using the more general scaling or affine transformation options can extend our matching process by providing a coarse alignment based on a more complex transformation.

## 6.3 Implementing a Feature-Based Matching Process

Our feature-based matching process is shown in Algorithm 3. The four phases of the process are (1) preprocessing, (2) generating a coarse alignment, (3) generating reference point pairs, and (4) determining the final correspondence.

### 6.3.1 Curvature, Curvature Map, and Feature Generation

The first three steps of Algorithm 3 represent preprocessing applied to the mesh representations of individual objects. Surface curvature is calculated using the quadric fitting method based on a two-ring natural parameterization as described in Chapter 3. This method was chosen primarily because it is robust to noise.

Curvature maps are then generated for each vertex as described in Chapter 4. Both 1-D and 2-D maps can be generated, with the 1-D maps generated using rings and the 2-D maps generated from geodesic fans. We use the 1-D map generated using rings, which is the fastest to generate. The curvature map radius for the bone meshes was chosen to be between one-fourth and one-half of the average dimension of the objects bounding box. For the face scans, the radius was chosen to be slightly larger than the largest significant feature, typically the nose.

Once the curvature maps have been generated, features are detected in a fully automated process as discussed in Chapter 5. The range of feature sizes follows from the choice of the curvature map radius.

### 6.3.2 Computing the Best Feature Pairs

The process for determining candidate feature pairings is shown in Figure 6.6. This is a more detailed view of Step 2 in Figure 6.1. In addition to being used to compute features, the curvature maps are also used to compare the similarity of the features. For each feature, the average curvature map is computed from the strongest points in the feature. The average Gaussian curve is shown in Figure 6.7. The mean curvature curve is calculated similarly. Strongest points are so designated in the feature detection step as feature points which occur most frequently during the multiple graph cutting runs. The similarity is then computed for every point in the feature relative

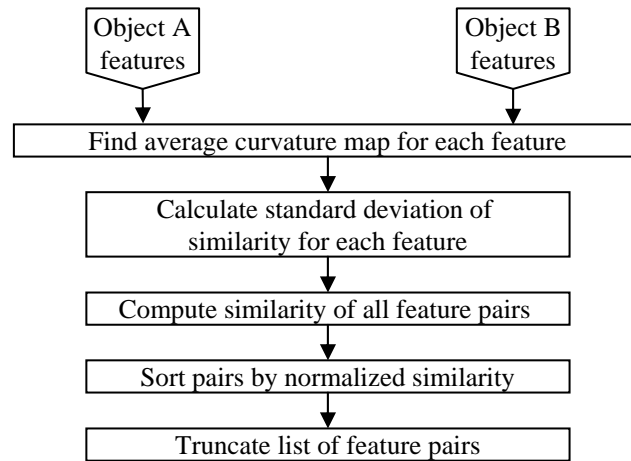


Figure 6.6: Steps in the process for determining candidate feature pairings.

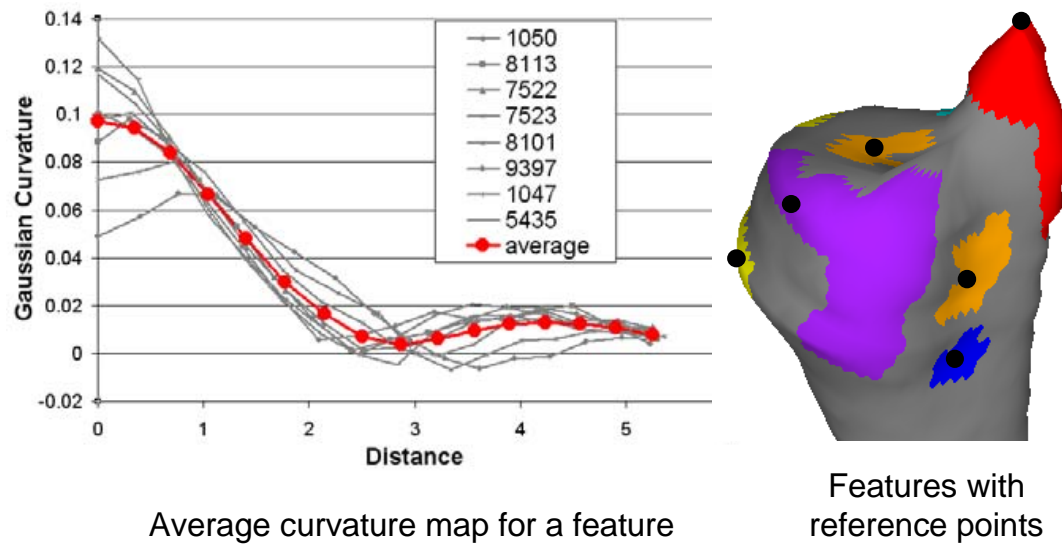


Figure 6.7: Average curvature map for a feature (left) and feature reference points (right). Note: only the Gaussian curve is shown.



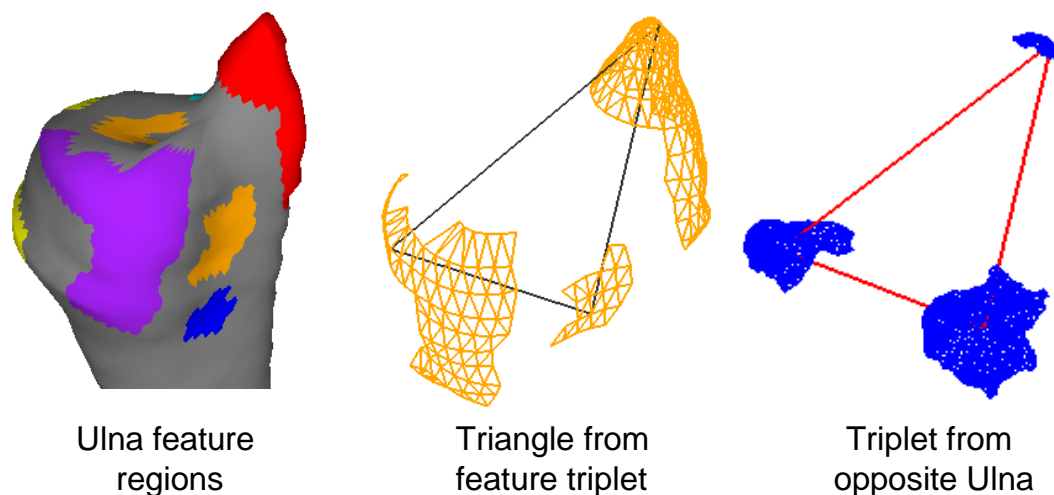


Figure 6.8: Triangles formed by three feature reference point pairs on the surface of two ulna bones.

to the average to get the standard deviation of the similarity. The radius used for the similarity calculation is one-half of the maximum radius of the curvature map. Then each feature of Object  $A$  is compared to every feature of Object  $B$ , by computing the similarity of their average curvature maps, and normalizing by the product of the standard deviations. The pairs are then ranked by this similarity score, and a threshold is used to truncate this list. Currently a simple approach is used which just keeps a fixed fraction of the list of corresponding pairs. We conservatively choose a fraction of 25% in our tests.

A reference point is also calculated for each feature. This is accomplished by finding the boundary of the region containing the strongest feature points for each feature and growing inward to the center of the feature. One of the points within the region, with the largest distance from the boundary of this region in terms of edge count, is selected as the reference point for the feature. This point is used for the initial alignment and may be replaced by a better choice during refinement of the alignment.

Triples of corresponding feature reference point pairs are used to generate candidate initial alignments, starting with the most similar pairs. Using the reference point

for each feature, the triplet for each object defines a triangle connecting these three features, as shown in Figure 6.8. The rigid transformation that best aligns these triangles is computed and then tested to see how well the object is aligned by this transformation. The metric used for the alignment quality is the sum of the distances from each reference point to the closest point (not necessarily a reference point) on the other object.

### **Pruning with Geometric Constraints**

The cost of comparing every possible three-point correspondence of two objects is prohibitive. Even with the identification of a limited number of features, additional pruning of the correspondence space is required. For example, with fifteen features on each object, there are over  $10^6$  possible correspondences. We apply a common pruning method which uses a geometric constraint to eliminate triplets when the distances between the features of one object is inconsistent with the distances between the corresponding features of the other object, indicating that the transformation the triplet would produce is physically unrealistic. Consistency is determined by calculating the ratio of the edge lengths for each pair of corresponding edges and discarding the triple of feature pairs if any of the edge length ratios is outside the range  $(\alpha, 1.0/\alpha)$ . For our work, we use  $\alpha = 0.8$ .

### **Benefits of Ordered Features**

Two other constraints can be used to prune the list of correspondences to be tested. First, the strength of the features can be used to place the highest priority on testing correspondences involving strong features. Secondly, the similarity measure can be used to order the possible feature pairs based on their similarity. Applying a threshold to the list of features or feature pairs can significantly reduce the number of correspondences to be tested, while retaining a high likelihood that the best alignments will not be eliminated prematurely. By sorting the feature pairs by similarity, the most likely triples are tested early. In our experiments, the best match was generally

- Physical alignment
  - Depends on choice of reference points
  - Determines three feature pairs
- Refinement
  - Test different possible reference points within the same feature pairs
    - Selected at random from all of the points in a feature

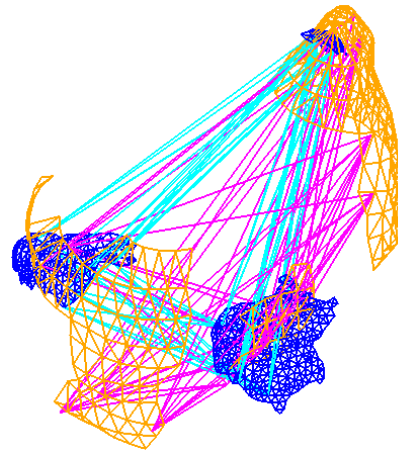


Figure 6.9: Refinement of the feature reference points for the selected feature pairs on the surface of two ulna bones.

found in the first two hundred triples tested. As a result, we chose to limit the maximum number of triples tested, setting the limit to 1000 or 2000 depending on the case.

### 6.3.3 Refining the Feature-to-Feature Fit

The previous step generates a coarse alignment based on a subset of the set of reference points. The reference points were chosen in a heuristic manner from a much larger set of candidate points. This drastically reduces running time, but with a sacrifice of alignment quality. While only a coarse alignment is needed, this initial alignment may be quite far off due to the initial choice of reference points.

A refinement step is used to improve upon this initial alignment. It maintains the correspondence between the three feature pairs, but finds more suitable choices for the reference points within the features. This refinement is illustrated in Figure 6.9. Three different algorithms have been investigated.

The first refinement algorithm revisits the selection of the reference points, this time selecting a reference point from the set of all of the feature points instead of just the strongest feature points. This new set of reference points produces a new rigid

alignment. The resulting quality is calculated, as represented by the sum of the reference point distances from the other object. The new alignment replaces the original alignment if the quality improves. Quality is measured as the sum of the distances of all reference points from the other object.

The next two refinement algorithms apply a randomized process to improve alignment quality. They each randomly select several alternate reference points for each feature pair. One algorithm tests the different combinations of these candidate reference points and chooses the best quality alignment produced. This can be somewhat time consuming since for  $m$  candidates at each of the six features, there are  $m^6$  combinations to test.

The final method improves on this by adjusting one reference point at a time, resulting in  $O(6m)$  combinations. Several passes can be performed efficiently, and the alignment updated whenever the new reference point improves the alignment quality.

Several methods in the literature apply ICP once a coarse alignment has been found. ICP looks at distances from all points to the other surface instead of distances only at the reference points. While this does make the overall distance between the objects smaller, it can also make the distances from the feature reference points to the surface of the other object larger. ICP does not weight the feature points any higher than other points, and since the number of reference points is small compared to the total number of points, ICP devalues the feature data. For this work, it is generally more important for the features to match than for the distances for all other points to be small. Therefore, while we have incorporated an ICP algorithm into our tools, we do not currently employ it for the shape matching process. As mentioned earlier, it can be used if the scaling or affine alignments are required.

### 6.3.4 Physical Correspondence and Area of Overlap

At this stage, only three corresponding feature pairs have been identified. So the next step is to calculate a rough correspondence from the refined coarse alignment. This rough correspondence is used only to determine potential overlaps between features of different objects. To create this correspondence, a brute force approach is used to

find, for each point on one object, the closest point on the other object. In our work, this brute force approach has been sufficiently fast. A pair of features is considered to overlap if a point in a feature of one object corresponds to a point that lies in the region of a feature on the other object.

Nothing so far has required that objects be genus zero. For example, the two objects may be bone samples whose spherical topology meshes have been sliced at possibly different places, and so there may be missing parts. This results in a boundary for the mesh. This trimming and the opening created do not affect the coarse alignment process, however they can affect the quality of the rough correspondence. By detecting points that correspond to the boundary points of the other object and checking the distance between them, we can apply a threshold to flag points that do not have valid correspondence in the other object. We apply a filter to remove such points and the opening in the remaining mesh can be triangulated to convert the open mesh to a genus zero surface.

### 6.3.5 Generating Additional Feature Pin Point Pairs

It is possible for a feature of one object to overlap more than one feature of the other object. In this case, the feature that overlaps multiple features can be split. This is done by labeling feature points by the corresponding feature they overlap, and growing rings out from the overlap region until the entire feature is labeled. The feature is split into subregions according to the labels. It is also possible that a feature of one object may not overlap any feature in the other object. These features are ignored.

For each of these additional feature pairs, a new pair of pin points is found. First, the point of each feature region with the largest absolute mean curvature is found. Next, the two features are extracted and aligned using ICP. From this alignment, a corresponding point in the other feature is found. Suppose  $C$  and  $D$  are features in objects  $A$  and  $B$  respectively. This yields two point pairs; the maximum mean curvature point in  $C$  and its corresponding point in  $D$ , and the maximum mean curvature point in  $D$  and its corresponding point in feature  $C$ . One of these pairs is chosen as pin points for this feature pair using the following symmetry test.

Every point in  $A$  has a corresponding point in  $B$  and every point in  $B$  has a corresponding point in  $A$ . Suppose  $P_1 \in A$  has corresponding point  $P_2 \in B$ ,  $P_2$  has corresponding point  $P_3 \in A$ ,  $Q_1 \in B$  has corresponding point  $Q_2 \in A$ , and  $Q_2$  has corresponding point  $Q_3 \in B$ . Since the rough correspondence mapping may not be a one-to-one mapping,  $P_1$  can be different from  $P_3$  and  $Q_1$  can be different from  $Q_3$ . In the symmetry test the distance from  $P_1$  to  $P_3$  is compared with the distance from  $Q_1$  to  $Q_3$ . The pair with the smaller distance is considered more symmetric, and this pair is chosen as the new pin point pair.

### 6.3.6 Adding Non-Feature-Based Pin Point Pairs

The number of pin point pairs based on overlapping features can vary greatly, depending on the number and layout of features identified on the individual objects. A small number of pin point pairs can limit control of the correspondence between the objects. To improve control, additional pin point pairs that are independent of the features can be added. To add a point, each vertex is labeled with its distance from the nearest pin point in terms of edge count. The vertex with the largest count in  $A$  or  $B$  is chosen. Its corresponding point in the other object is chosen as the other point of the new pin point pair.

### 6.3.7 Similarity-Based Adjustment of Point Pair Locations

The pairs of pin points come either from the best alignment of select features, overlapping features, or the addition of a point and its corresponding point based on the rough correspondence. To improve the correspondence, the pin points can be adjusted using the curvature map similarity measure in a greedy approach. Consider a pin point pair  $P_A \in A$  and  $P_B \in B$ . First the similarity of  $P_A$  to  $P_B$  and to each of  $P_B$ 's neighbors is checked. If  $P_A$  is more similar to one of the neighbors than to  $P_B$ , the neighbor is a candidate to replace  $P_B$ . Similarly, if  $P_B$  is more similar to a neighbor of  $P_A$  than to  $P_A$ , the neighbor is a candidate to replace  $P_A$ .

To avoid large distortions of the mesh, the pin point adjustment is limited by tracking the change in geodesic distances between a pin point and its neighboring pin points. A

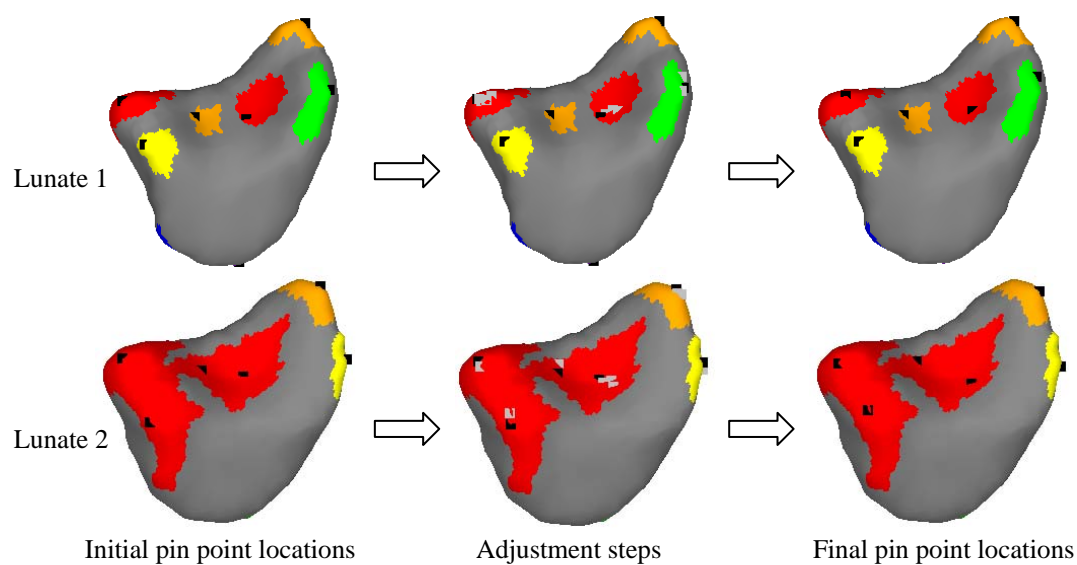


Figure 6.10: Adjustment of pin point locations based on similarity. The original pin point locations (left) are moved in an iterative process (center) to their final locations (right).

cost function is constructed with two terms, a similarity term based on the amount of improvement in the similarity, and a stretching term based on the change in geodesic distances. Weights are applied to balance the magnitudes of the terms. Pin point adjustment proceeds only so far as the resultant cost function decreases.

The weights can be varied to control the magnitude of the adjustment. The goal is to adjust the weights so that where the change in similarity between a point and its neighbors is small, the stretching term will limit the pin point movement, while a large similarity improvement by movement from a pin point to its neighbor will be allowed. Currently, setting the weights is a manual process, since we cannot know a priori what the distances between the pin points on the objects, or the difference in similarity values will be. Figure 6.10 shows the adjustment of pin point locations based on the shape similarity measure. The original pin points are shown on the left. The middle view shows the original pin points in black and locations during the iteration process in gray. Note that not all of the pin points move. The final pin point locations are shown on the right.

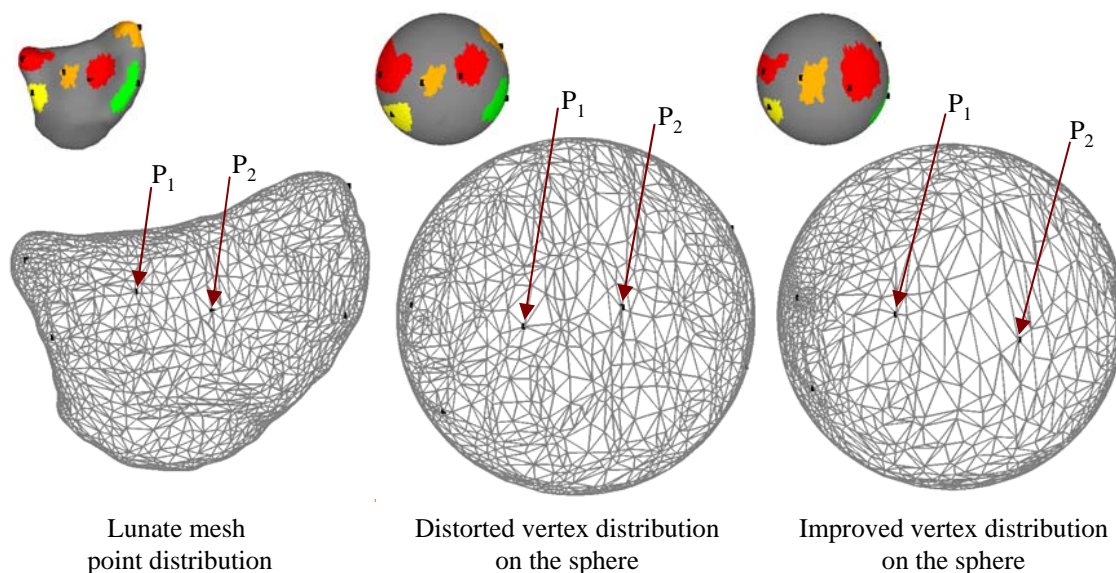


Figure 6.11: Iterative smoothing with a proper choice of weights can enable the spherical parameterization to better represent the relationships on the original mesh.

### 6.3.8 Correspondence and Parameterization

Once the final pin point pairs have been determined, the final correspondence can be generated by mapping both objects to a common spherical parameterization. An initial spherical parameterization [147] is made for each object by partitioning the mesh into two halves, and using an isomap technique to parameterize the equator. Each half is mapped to a hemisphere and the hemispheres are joined at the equator. Weighted smoothing techniques are used to adjust the points on the sphere to control the vertex distribution and to avoid folding. Different weights may be used to control different aspects of the distribution, such as equalizing area of the mesh triangles, or making the distribution of the mesh triangles on the sphere approximate the distribution of triangles on the original mesh. For example, using Floater's method [51] to calculate weights from the original mesh, and using those weights to smooth the mesh on the sphere, leads to better similarity between the mesh triangles and the corresponding triangles of the spherical parameterization.

Why is this distribution important? Because only the pin point pairs will be controlled explicitly in the common spherical parameterization, the correspondence between



other points of the original meshes relies on the mapping being ‘well-behaved’. The more distortion in the mapping from the mesh to the parameterization, the more pin points that are required to represent the mapping accurately.

Several metrics have been proposed to measure the distortion associated with mappings between triangular meshes [148, 140, 93, 150]. We implemented two stretch measures, one specific to spherical mapping, similar to the method of Praun and Hoppe [140], and one for comparing general meshes similar to the method of Schreiner et al. [150]. These methods show consistent trends and were used to determine the choice of weights to generate reasonable similarity between the original mesh and its spherical parameterization. Figure 6.11 shows an initial and an improved spherical parameterizations for a coarse mesh of a lunate bone. The first spherical parameterization (center) had stretch measure of 0.19 where 1.0 is ideal. The second parameterization had stretch measure of 0.43 indicating a significant improvement. For these cases, the best values obtained with our smoothing techniques are in the range from 0.4 to 0.6. Note that the point distribution between the center pin points ( $P_1$  and  $P_2$ ) is more consistent with the original mesh distribution in the right view compared to the center view.

### **Aligning Spherical Parameterizations**

In order to establish the correspondence between the objects, they need to have a common spherical parameterization. Mapping each object to a common spherical parameterization involves using the pin point pairs to adjust one or both of the individual spherical parameterizations. The steps in this process are shown in Phase 4 of Algorithm 3. The first step is to find the best rigid rotation of the spherical domain to get an approximate alignment of the pin point locations on the sphere. This rigid rotation is applied and then the pin point locations are averaged to get target pin point locations on the common spherical parameterization. Alternately, the set of pin points from one of the objects can be selected as the target pin point locations.

Next, we create a common base parameterization from the pin point pairs. The construction of the base parameterizations will ensure that the pin points are aligned.

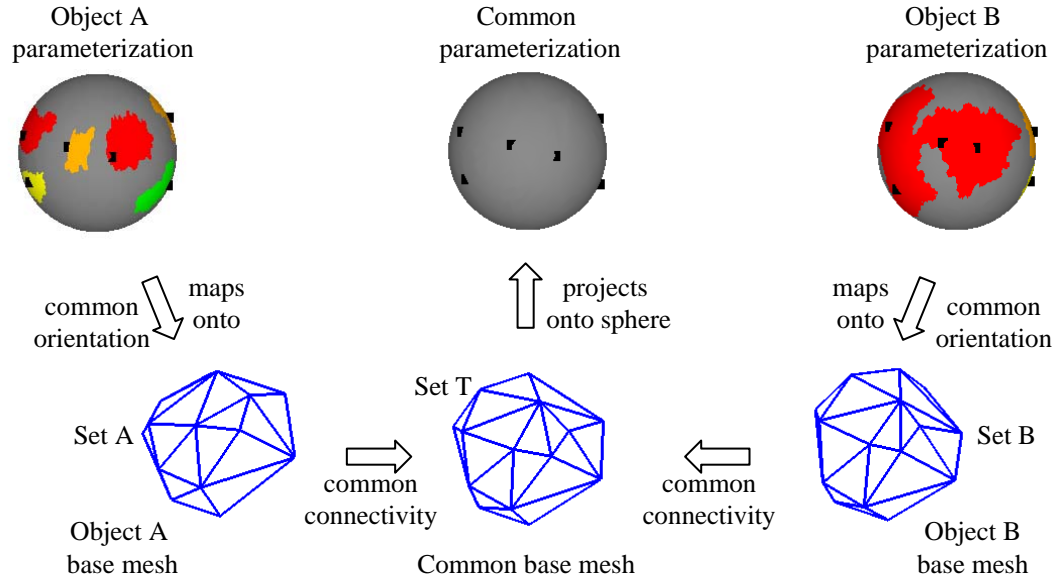


Figure 6.12: Creation of the common spherical parameterization. A common base mesh is created as an intermediate step in the alignment process.

There are now three sets of canonical sphere points defined; *SetA* which is the set of pin point locations on the spherical parameterization for *ObjectA*, *SetB* which is the set of pin point locations on the spherical parameterization for *ObjectB*, and *SetT* which is the set of target pin point locations. A base mesh is generated from these three sets of points. First, a convex hull triangulation is generated for *SetA* and *SetB*. The connectivity for each of these triangulations can also be applied to the *SetT* and compared with a convex hull triangulation of *SetT*. Differences in the connectivity can be resolved by generating the great circle arcs associated with edges representing differences in connectivity, and adding points where these arcs intersect on the sphere. Points are also added at the corresponding locations of *SetA* and *SetB* and the sets are re-triangulated.

Figure 6.12 illustrates the process of mapping the points on the individual spherical parameterizations onto the common parameterization. The individual spherical parameterizations are mapped onto versions of this base mesh adapted to their specific pin point locations. The spherical parameterization for *ObjectA* is mapped onto the base mesh representation on the *SetA* points. This same mapping applied to the

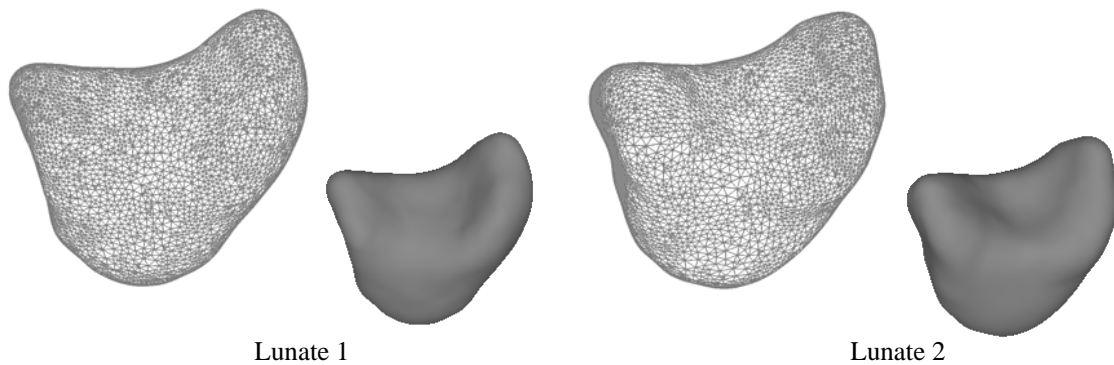


Figure 6.13: Mesh and surface views for two lunate bones.

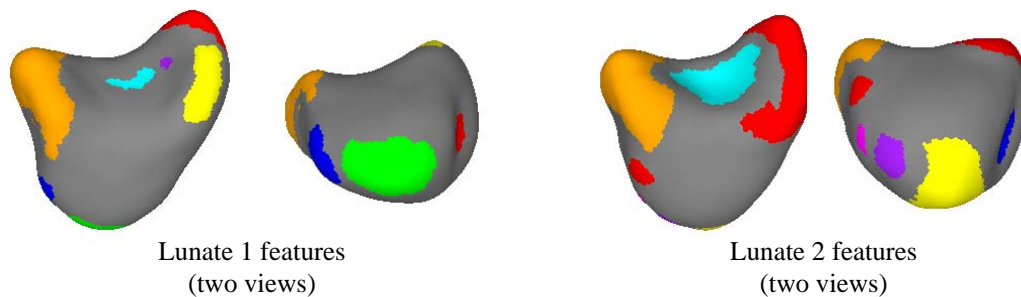


Figure 6.14: Features identified for two lunate bones.

base mesh on *SetT* projects the points onto the common spherical parameterization. A similar mapping is generated for *ObjectB*. The common spherical parameterization produces a point-to-point correspondence between *ObjectA* and *ObjectB*. The construction ensures that the pin points are aligned.

## 6.4 Bone Shape Comparisons

### 6.4.1 Lunate Bone Data Set

Figure 6.13 shows the surface mesh and shaded views for two lunate bones. Lunate 2 is from a right wrist, and lunate 1 is from a left wrist and has been mirrored for

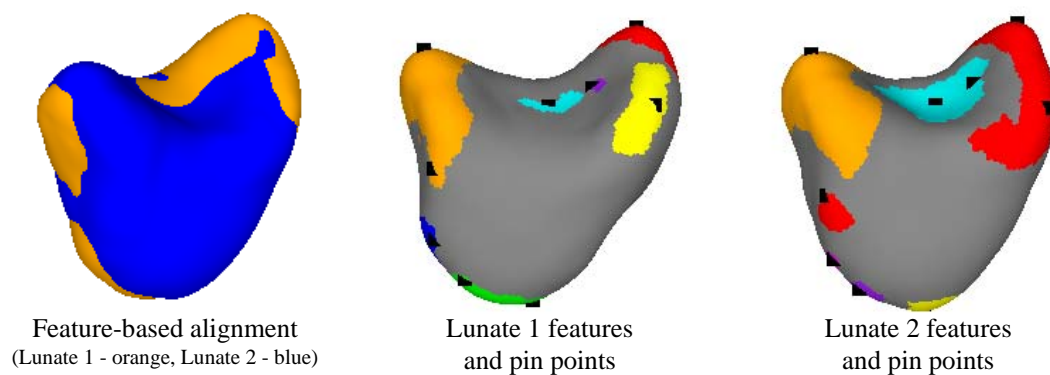


Figure 6.15: Alignment of the lunate bones based on feature correspondence, and the pin points generated.

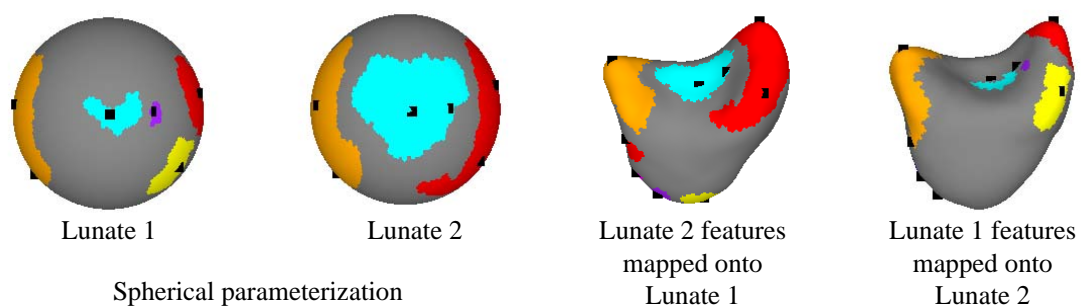


Figure 6.16: Spherical parameterizations used to determine correspondence (two left views). The two right views show the features from each lunate mapped onto the other lunate.

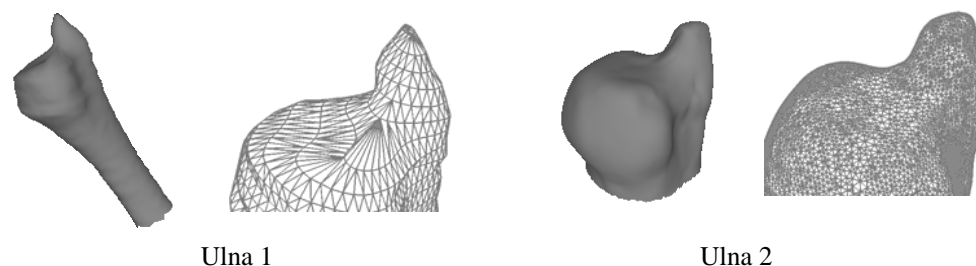


Figure 6.17: Geometry and mesh representation for two ulna samples.

comparison with the right lunate. The shapes are very smooth and are represented by a dense mesh. Each mesh contains 7800 vertices and 15596 faces. Figure 6.14 shows the features generated for these bones. Ten features were found for each object, although there is not a one-to-one match of the features.

The left view in Figure 6.15 shows the coarse alignment using the features identified. The second alignment tested was the best, with a total of 136 alignments tested and 1632 possible alignments pruned away. The center and right views show the pin points that were selected for the overlapping features.

The first two views in Figure 6.16 show the spherical parameterizations for lunate 1 and 2 after aligning the parameterizations using the pin point pairs. These aligned spherical parameterizations define the correspondence between these bones. The second view from the right shows the features from lunate 2 mapped onto lunate 1 using this correspondence. The right view maps the features from the lunate 1 onto lunate 2. Comparison with Figure 6.15 shows a high degree of similarity between the original and mapped features. This gives an indication of the consistency of the correspondence.

### 6.4.2 Ulna Bone Data Set

Figure 6.17 shows the surface mesh and shaded views for two ulna samples from different subjects. There are three issues complicating this case. First, there is a significant difference in resolution with ulna 1 containing 2108 vertices and 4171

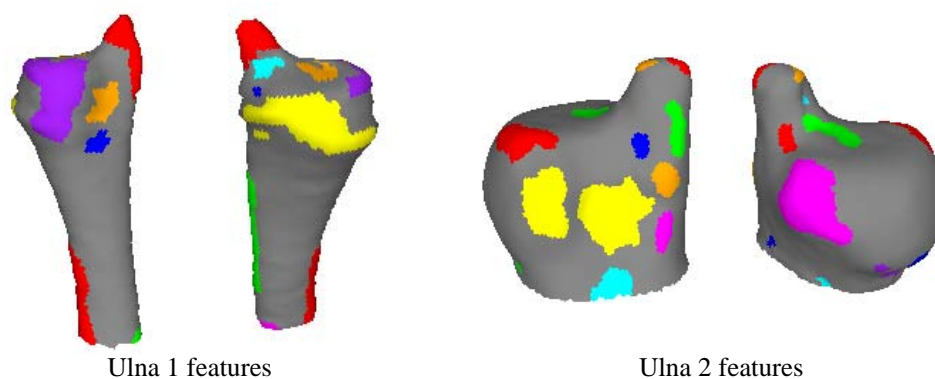


Figure 6.18: Features detected for two ulna samples.

faces, while ulna 2 has 9645 vertices and 19133 faces. The second issue is that ulna 1 includes a larger portion of the shaft, while ulna 2 is mainly just the head of the ulna. The third issue is that there is a hole in the surface where the shaft would continue so that the surfaces are not genus zero. Features can still be detected, as shown in Figure 6.18. In spite of the difference in resolution, the features for the two ulna samples are similar.

The left view of Figure 6.19 shows the alignment based on the 14 and 22 features for ulna 1 and 2 respectively. The 22nd out of 1000 matches tested was the best, with pruning removing 20838 potential matches. The dark gray in the center view indicates points of ulna 1 that do not have valid correspondence with ulna 2 and can be trimmed. The trimming step is combined with closing the hole to generate genus zero versions of ulna 1 and 2. Features for the trimmed version of ulna 1 are shown in Figure 6.19 (right).

Figure 6.20 shows the feature-based alignment of the trimmed data set (left), and the pin points generated for ulna 1 (center) and ulna 2 (right).

The aligned spherical parameterizations for the two ulna samples are shown on the left in Figure 6.21. The right side of the figure shows the cross-mapping of features from one sample onto the other using the correspondence produced by the common spherical parameterization. The cross-mapped features can be compared to Figure 6.20.

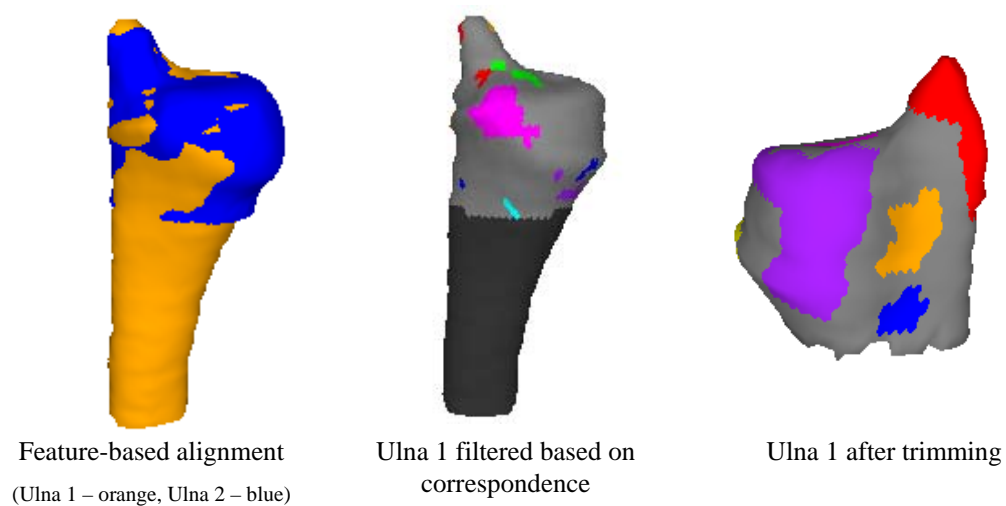


Figure 6.19: Alignment and trimming of the samples to a common portion of the ulna. Dark gray (center) indicates regions for which there is no corresponding point in the other object.

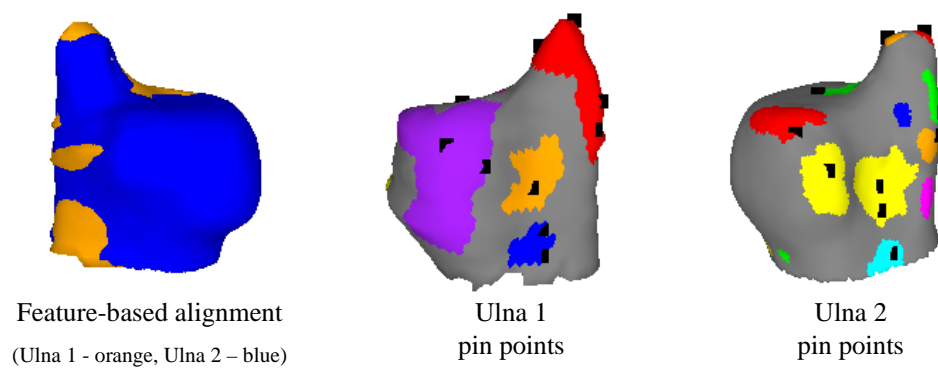


Figure 6.20: Alignment of the trimmed samples and the pin points generated.

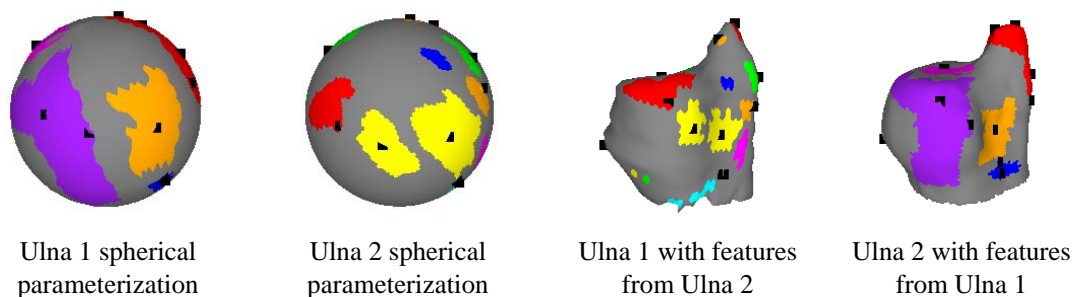


Figure 6.21: Spherical parameterizations aligned using the pin points for the ulna samples. The views on the right show the features of one ulna mapped onto the other using the correspondence produced by the common spherical parameterization.

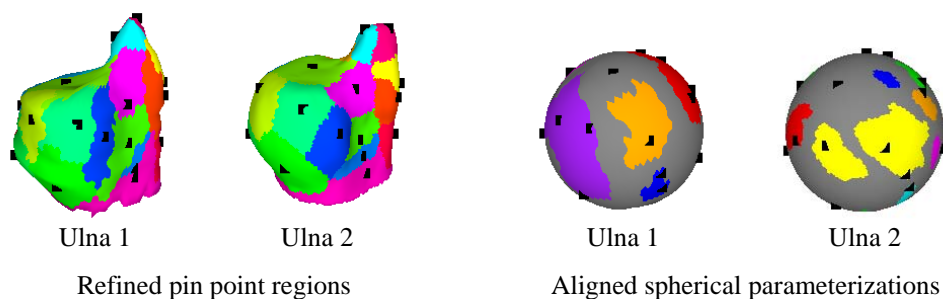


Figure 6.22: Additional pin points with adjustment based on similarity. Voronoi-like regions associated with each pin point are shown on the left. The adjusted spherical parameterizations are shown on the right.

Some issues can be seen near the base where the trimming occurred, and near the tip of the styloid process where the shapes are noticeably different.

In order to improve the common parameterization, we automatically add additional pin points and adjust these pin points to improve the similarity. Figure 6.22 shows Voronoi-like regions associated with the pin points. These regions are based on edge count rather than distance, and support the process of adding pin points at locations farthest from any current pin point. These pin points are adjusted to improve the similarity of the pin point pair, which improves the quality of the overall parameterization. The adjusted spherical parameterizations are shown on the right side of Figure 6.22.



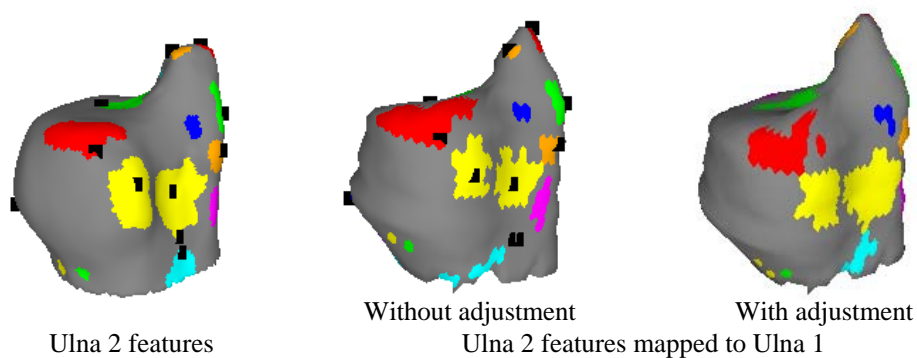


Figure 6.23: The features of ulna 2 are shown on the left and mapped onto ulna 1 without (center) and with (right) additional pin points and pin point adjustment.

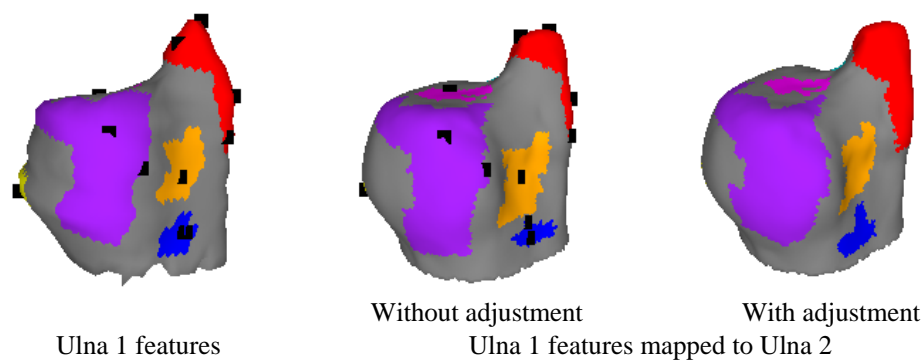


Figure 6.24: The features of ulna 1 are shown on the left and mapped onto ulna 2 without (center) and with (right) additional pin points and pin point adjustment.

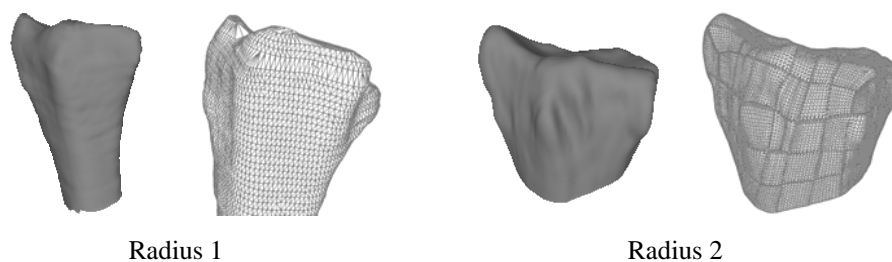


Figure 6.25: Geometry and mesh representation for two radius samples.

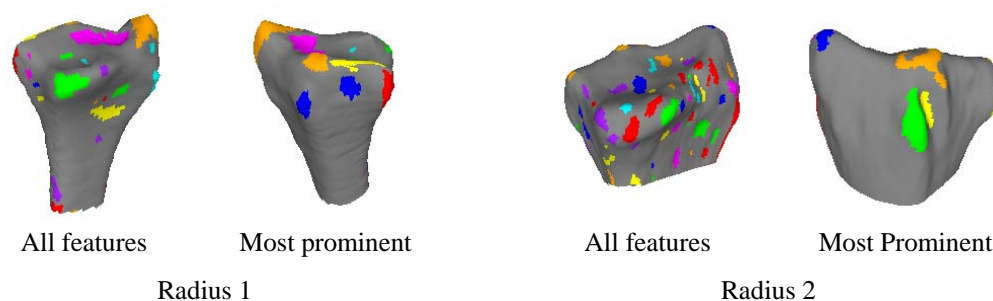


Figure 6.26: Features detected for radius 1 and radius 2 bone samples.

Figures 6.24 and 6.23 show the cross mapping of parameters without and with the additional pin points and pin point adjustment. The adjustment improves the parameterization in several areas, particularly near the base where the trimming occurred.

### 6.4.3 Radius Bone Data Set

Figure 6.25 shows the surface mesh and shaded views for two radius samples from different subjects. Like the ulna dataset, these samples have different resolution, cover different amounts of the bone surface, and are not genus zero. Radius 1 has 3674 vertices and 7286 faces while radius 2 has 17587 vertices and 35006 faces.

Figure 6.26 shows features for the two radius samples. Thirty-four features were found for radius 1 and 100 features were found for radius 2. The left view for each

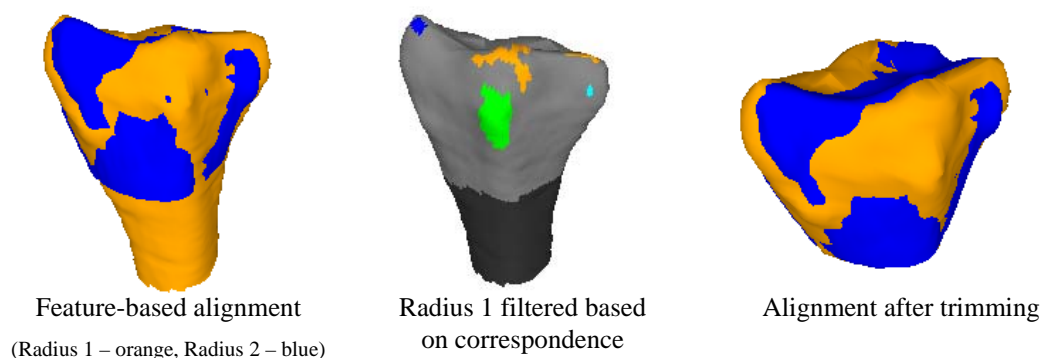


Figure 6.27: Alignment and trimming of the samples to a common portion of the radius. Dark gray (center) indicates regions for which there is no corresponding point in the other object.

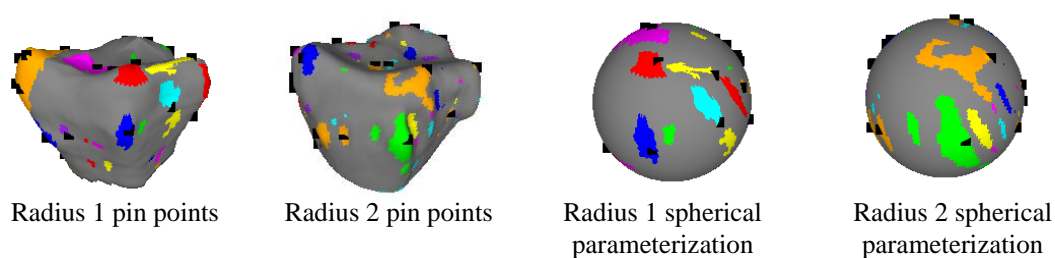


Figure 6.28: Spherical parameterizations aligned using the pin points for the radius samples. The views on the right show the features of one radius mapped onto the other using the correspondence produced by the common spherical parameterization.

sample shows all of the features and the right view shows only the 15 most prominent features.

Figure 6.27 shows the alignment (left), correspondence-based filtering used to trim radius 1 (center) and the trimmed alignment (right). Using all features, the best alignment of the first 2000 tested (29314 pruned) was number 1255. However, if the alignment was limited to the 20 most prominent features of each surface, the best match was number 72 of 1000 with 20311 possible alignments pruned. The value of the alignment quality distance metric (sum of all distances from reference points to the other object) was comparable for both cases.

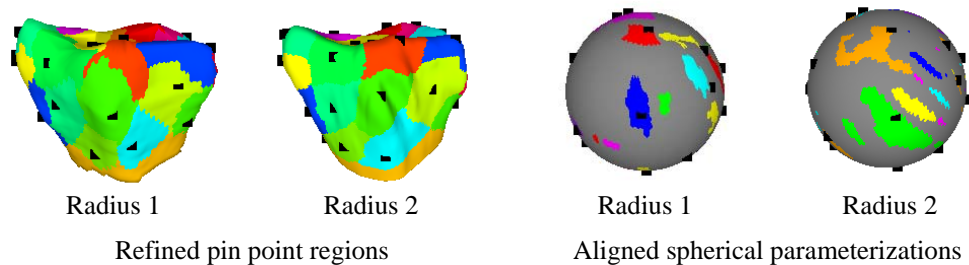


Figure 6.29: Additional pin points with adjustment based on similarity. Voronoi-like regions associated with each pin point are shown on the left. The adjusted spherical parameterizations are shown on the right.

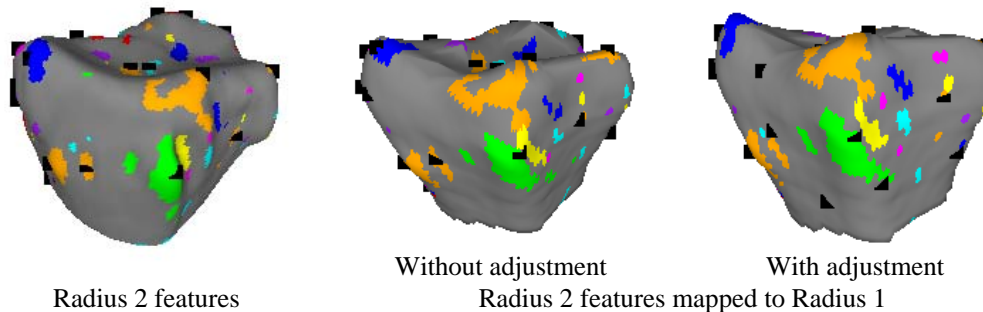


Figure 6.30: The features of radius 2 are shown on the left and mapped onto radius 1 without (center) and with (right) additional pin points and pin point adjustment.

Figures 6.28 and 6.29 show the alignments of the spherical parameterization without and with additional pin points respectively.

The cross-mapping of features is shown in Figures 6.30 and 6.31. Better similarity with additional pin point adjustment is indicated in Figure 6.31 by the orange feature (left edge), red feature (right edge), and the relation of the cyan and yellow features (left-center).

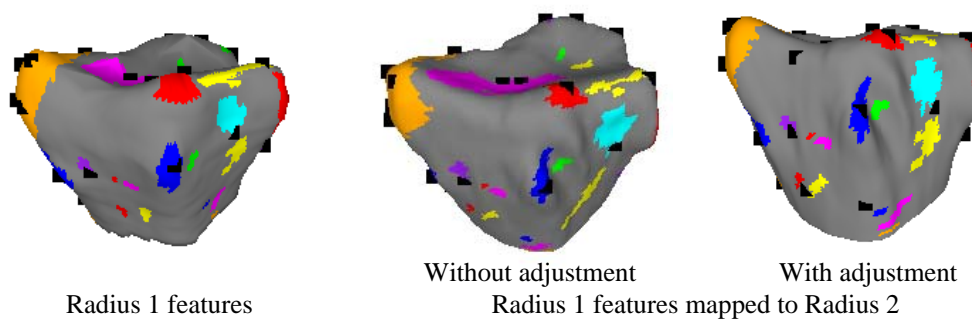


Figure 6.31: The features of radius 1 are shown on the left and mapped onto radius 2 without (center) and with (right) additional pin points and pin point adjustment.

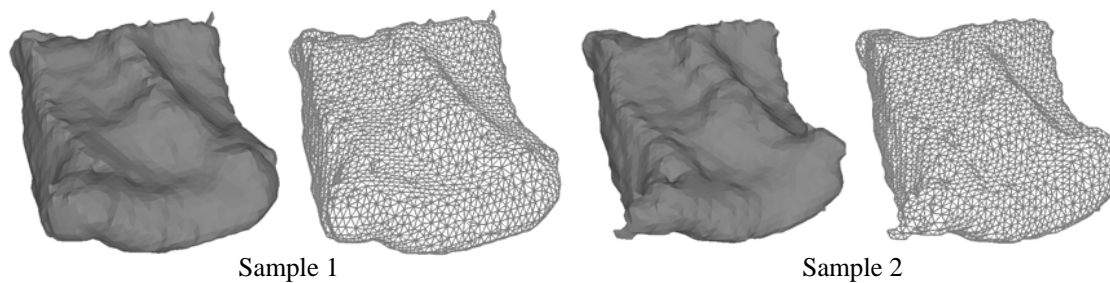


Figure 6.32: Geometry and mesh representation for two intermediate cuneiform samples.

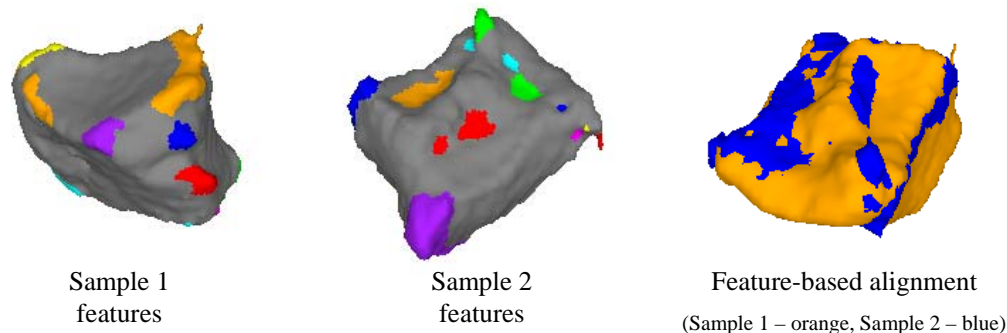


Figure 6.33: Features detected for two intermediate cuneiform bone samples (left and center), and the resulting feature-based alignment (right).

#### 6.4.4 Intermediate Cuneiform Bone Data Set

Figure 6.32 shows the surface mesh and faceted views for two intermediate cuneiform bone samples. Note the roughness of the mesh due to the coarseness of the discrete sampling. A small amount of smoothing was applied to avoid an instability in the curvature calculation when fitting a quadric function to points that fall on the edges of a pyramid. Both meshes contain approximately 4800 vertices and 9700 faces.

Figure 6.33 shows the features detected (left and center) and the alignment based on those features (right). With 15 and 22 features on samples 1 and 2 respectively, the best alignment occurred on the 360th of 1000 tested with 24639 possible alignments pruned.

Figure 6.34 shows the pin points and the aligned spherical parameterizations for the two intermediate cuneiform bone samples.

The cross-mapping of features is shown in Figure 6.35. The left pair of views show the features of sample 2 on sample 2 and mapped onto sample 1. The right pair of views show the features of sample 1 on sample 1 and mapped onto sample 2. The correspondence for this case seems well-behaved.

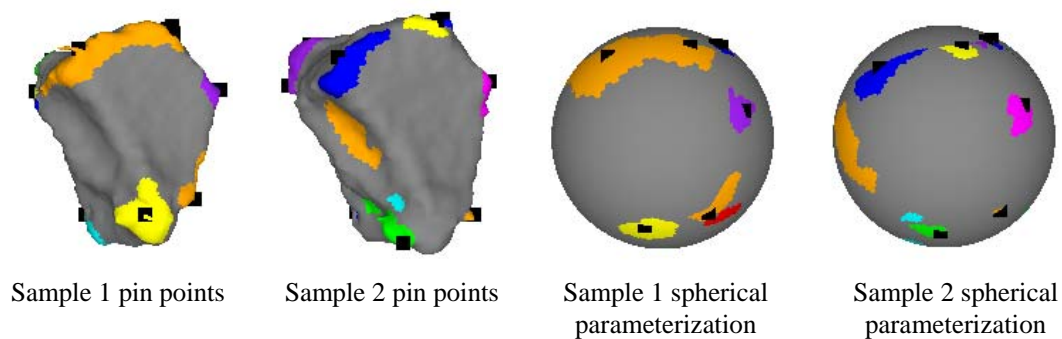


Figure 6.34: Pin points generated for the intermediate cuneiform samples (left two views), and the spherical parameterizations aligned using those pin points.

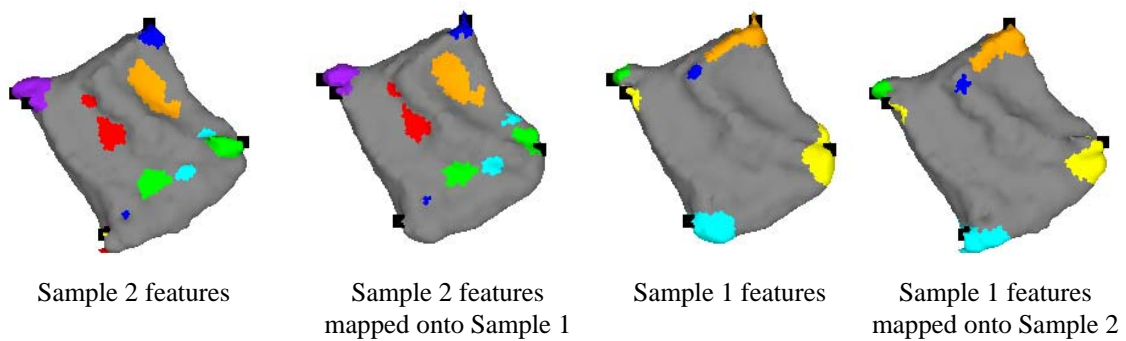


Figure 6.35: Cross mapping features between intermediate cuneiform bone samples.

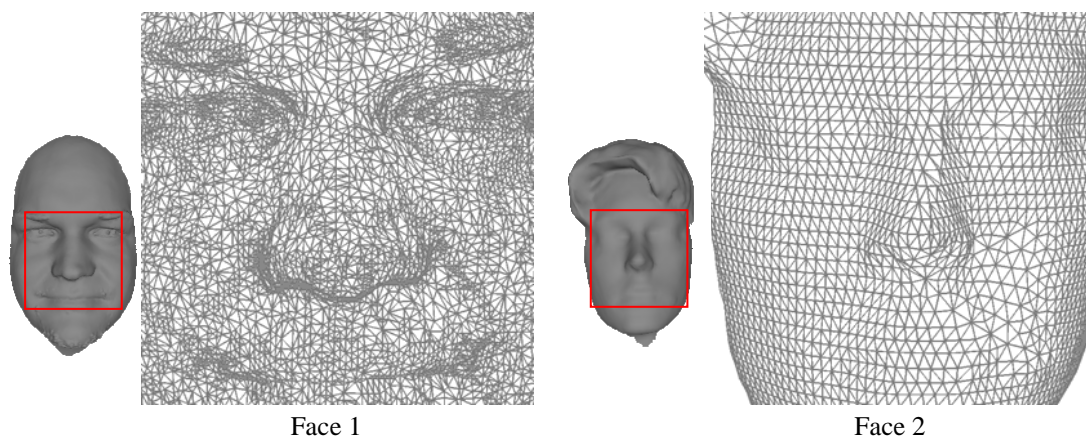


Figure 6.36: Geometry and mesh representation for two face scans.

## 6.5 Face Scan Comparisons

Figure 6.36 shows two face scans with expanded views of the surface mesh. On the left is a high resolution scan with 8863 vertices and 17479 faces. The scan on the right has 3032 vertices and 5920 faces. This results in 27 and 9 features respectively for Face 1 and 2. Figure 6.37 shows these features (left and center) and the alignment they produce (right). The best alignment was number 10 of 744 tested with 35802 possible alignments pruned.

## 6.6 Discussion

Even with only ten to twenty features identified for each object, the number of possible alignments is large. Limiting the number of candidate feature pairs and geometric pruning reduce the number of possible alignments to be tested, but this can still be a large number. Our experience has shown that since the pairs are ordered, the best alignment is always found early in the process. This provides justification for reducing the limit on the maximum number of alignments to be tested. It is statistically likely that if we tested enough other alignments, even if alignments were picked at random, we might find one that better aligns the reference points. However, it is not likely



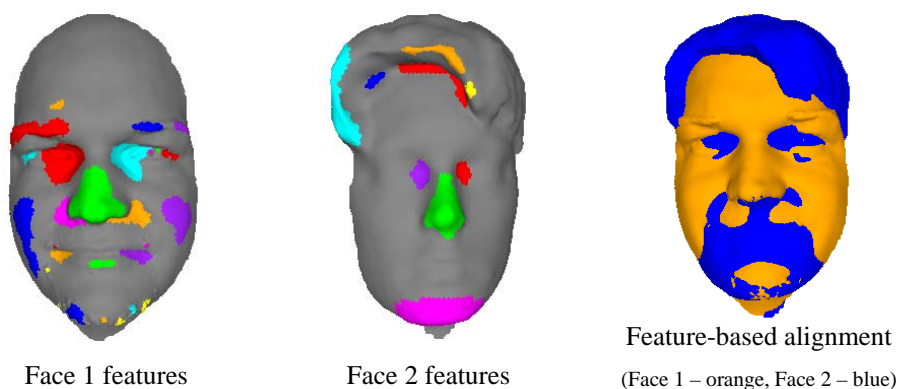


Figure 6.37: Features detected for the two face scans (left and center) and the alignment based on these features (right).

that the alignment would be significantly better than an alignment based on the most prominent features of the objects.

We experienced one case where our method failed to align two coarse bone samples due to an insufficient number of features. This circumstance is easy to detect, and can be overcome by providing additional features. These features could be generated using extrema in scale space to detect salient features [100] or using the scale-invariant feature transform (SIFT) from a difference of Gaussian function [106]. Our similarity measure can be used to order potential feature pairs by their similarity, independent of how the features are identified.

Since the method proposed here aligns based on the best corresponding pairs, it is less susceptible to individual outliers, however more detailed assessment of the impact of outliers is still needed.

### 6.6.1 Timing

Execution times for the steps in the shape matching process applied to several example cases are presented in Table 6.1. The computation of curvature, the curvature map, detecting features, and computing the spherical parameterization can all be performed as a preprocessing step. The total preprocessing time for the example cases ranges

Table 6.1: Shape Matching Execution Times

<i>Shape Matching Example Cases</i>												
Case:	Lunate		Ulna		Radius		Foot		Face			
	1	2	1	2	1	2	1	2	1	2		
<i>Mesh sizes</i>												
Vertices	7800	7800	2108	9646	3674	17578	4842	4890	8863	3032		
Faces	15596	15596	4171	19133	7286	35006	9680	9776	17479	5920		
<i>Preprocessing calculation times (seconds)</i>												
Curvature	13.6	13.1	2.9	14.6	5.3	28.2	8.3	8.4	15.9	4.4		
Curvature map	4.8	4.8	0.1	3.0	0.3	8.2	0.8	0.9	4.8	0.5		
Features	36.3	35.5	8.5	46.0	13.9	88.1	19.9	21.1	46.6	12.1		
Spherical parameterization	30.1	27.4	4.6	37.1	11.1	54.6	16.1	14.3	-	-		
<i>Features</i>												
Number found	10	10	14	24	47	100	15	21	35	9		
<i>Matching calculation times (seconds)</i>												
Initial alignment	0.8		2.3		15.4		2.6		3.1			
Refinement method 1	0.6		0.2		0.2		0.2		0.3			
Refinement method 2	8.1		8.7		65.0		10.7		12.7			
Refinement method 3	0.8		0.5		2.5		0.6		0.7			
Pin points	7.4		5.4		22.4		4.5		-			
Adjusting points	23.7		17.0		94.8		9.5		-			
Adding points (time per point)	3.0		1.9		4.2		1.6		-			
Spherical correspondences	68.8		14.3		56.2		26.9		-			
<i>Shape comparison times (seconds)</i>												
Distance measures	0.2	0.3	0.04	0.2	0.1	0.5	0.2	0.2	0.3	0.1		
Angle measure	0.2	0.2	0.2	0.2	0.3	0.4	0.1	0.2	0.3	0.1		
Similarity	1.0	1.0	0.2	0.8	0.4	1.3	0.5	0.5	1.0	0.5		

\* All execution times based on a 1.7 GHz Intel Pentium M processor with 2.0 GB of RAM

from sixteen seconds to about three minutes on a 1.7 GHz Intel Pentium M processor with 2.0 GB of RAM. The curvature calculation is  $O(n)$ , where  $n$  is the number of vertices. The curvature map calculation is  $O(nk)$ , where  $n$  is the number of vertices and  $k$  is the number of vertices within the curvature map radius, which in turn depends on the size of the curvature map radius and the resolution of the mesh. Feature calculation time is based on the speed of the graph-cut algorithm, which is approximately linear in  $n$ .

The speed of the alignment step depends primarily on the number of features, as seen in the radius case. The initial matching is efficient due to the pruning of the potential alignments. The time is still dependent on the limits set for pruning, the number of features considered, and the number of alignments tested. More aggressive choices would seem reasonable based on our limited experience.

Refinement method 2 is much slower than the other two refinement methods. This is due to the much larger number of alignments to be tested. Since the other refinement methods seem to provide comparable improvement of the initial alignment, use of method 2 is not recommended. The initial alignment combined with the fast refinement techniques generates alignments that compare well with ICP alignment, and overcome the ICP requirement for a coarse alignment as a starting point.

The most time consuming of the matching steps are computing and adjusting the pin points, and generating the point-to-point correspondence from the spherical parameterizations. However, even matching our largest example case required only about three minutes to compute, ignoring refinement method 2 which we recommend skipping. Relative to these execution times, the time required for shape comparison based on the point-to-point correspondence is inconsequential.

### 6.6.2 Scaling

Our method is not formally scale independent. In fact, it can be important to be able to detect changes in scale as well as shape. As a result, we have not set scale independence as a goal.

The relative scale of the objects impacts two areas. First, the magnitude of the mean and Gaussian curvatures is a function of the overall scale of the objects. The curvatures then affect the computation of the curvature map. The curvature map is also a function of the curvature map radius, which will depend on the scale. Similarity, which is a function of the curvature map, will also depend on the scale of the object.

The second area impacted by the relative scale of the objects is the tolerance used for geometric pruning. For small relative differences in scale, the geometric pruning constraint parameter may be sufficient, and matching will not be significantly affected. If necessary, the geometric constraint can be loosened, which decreases the pruning effect, thereby increasing the time searching for the best correspondence. As long as the object sizes are not too different, the position of the feature pairs in the priority list should not change drastically. If the geometric constraint is too restrictive, valid correspondences may be eliminated.

If scale independence is desired, the primary challenge is to compute shape similarity in a scale independent manner. The features that are extracted are not effected by the overall scale. If the relative scale factor is known, the curvature data can be adjusted and matching can proceed using our method. Rigid body scaling can be estimated by using a more general transformation to align features, and decomposing the transformation to get the scaling component. For example, the scaling and affine transformation options of the ICP algorithm [146] could be used to get an initial alignment of the objects. The relative scale could then be extracted from the alignment transformation.

As an alternative, the variation of the curvature map could be constructed using a scale independent curvature metric. For example, the curvature of a triangular face can be estimated from the normal vectors at its three vertices. These normal vectors are independent of the scale of the object. The area and radius associated with the curvature map would also need to be normalized in some manner.

## 6.7 Chapter Summary

This chapter uses the shape similarity measure along with identified features to align objects and compute their point-to-point correspondence. The alignment method benefits from the robustness of the curvature map and the multi-scale feature detection approach. Efficient alignment is the result of geometric pruning of the possible possible triples of feature pairs, features ordered by strength, and the ordering of potential feature pairs based on similarity. Pruning limits testing of alignments to a fraction of the possible alignments.

The alignment process can operate on genus zero surfaces. It can also operate on meshes that have boundaries, having been trimmed at possibly different locations, by repairing the mesh before continuing with shape matching.

From the initial alignment and the identified features, pin points are determined that are used to align the spherical parameterizations for each object. These pin point locations are adjusted to improve the shape similarity of the corresponding locations. Additional pin points can be added and adjusted to promote improved similarity across the correspondence, instead of being controlled only at feature point pairs.

The alignment of the spherical parameterizations defines the point-to-point correspondence. This is demonstrated by mapping the features of one object onto the other. The point-to-point correspondence could also be used to refine the physical alignment, possibly taking into account scaling or affine transformations. The next chapter will look at some other ways to use the correspondence to examine the shape differences.

---

**Algorithm 3** Feature-Based Matching
 

---

*Phase 1: Preprocessing of object meshes*

**for all** Object Meshes **do**

**if** Not genus zero **then**

    Convert mesh to genus zero

**end if**

    Compute curvature at each vertex

    Compute curvature maps (from rings and/or fans)

    Identify features

    Generate a spherical parameterization

**end for**

*Phase 2: Create a coarse alignment of a pair of objects ( $ObjectA, ObjectB$ )*

Compute similarity between features

Select feature reference points

Sort pairs by similarity

$count \leftarrow 0$

**while**  $count \leq maxcount$  **do**

  Form triples of potential pairs from ordered list of pairs

**if** Satisfies geometric constraints **then**

    Compute rigid transformation

    Compute alignment quality as sum of all reference point distances to the nearest surface

    Keep alignment with minimum distance sum

$count \leftarrow count + 1$

**end if**

**end while**

Refine using alternate reference points within the chosen feature pairs (randomized)

*Phase 3: Generate refined list of point pairs*

Compute an approximate correspondence from the alignment

Find other overlapping features and add a corresponding point pair

Refine pair locations via similarity subject to a stretching constraint

Add additional pairs and refine via similarity

*Phase 4: Generate the correspondence between the objects*

Generate reference locations on the spherical parameterization for each point pair

Triangulate the reference locations and point pairs ( $A, B$ ) in the common,  $ObjectA$ , and  $ObjectB$  domains respectively to form a common base mesh and corresponding local base meshes

Map each original spherical parameterization to the common parameterization using its local base mesh

*The common spherical parameterization provides correspondence between points of  $ObjectA$  and  $ObjectB$*

---

## Chapter 7

# Quantifying Shape Differences

This chapter addresses the problem of identifying local and global shape differences. In order to describe the differences between our aligned objects, we need to differentiate between:

- Regions where the shapes are similar,
- Regions where the shapes are different,
- Likely locations of anomalies, such as features that grow/shrink, appear/disappear, or shift in location.

### 7.1 Shape Comparison Goals

We are applying shape comparison to a class of shapes that have few well-defined features. There are many nuances and subtle differences that can be classified only by a domain expert. One of our goals is therefore to help the domain expert find the shape features that are most likely to be of interest.

To be useful, our shape comparison method must include both analysis and visualization components. It should be easy for a user to apply the analysis and see the similarities and differences for the surfaces.

## 7.2 Analysis and Visualization Approach

The approach presented here builds on the initial alignment and the point-to-point correspondence computed in Chapter 6. We define several shape difference measures based on physical distances, local orientation angles, and shape similarity. The point-to-point correspondence enables these shape difference measures to be used to analyze corresponding points on the objects.

The visualization technique that we use is pseudo-coloring to represent a scalar property on the surface, where the scalar property is one of the shape difference measures. Thresholds are used to control the lower and upper bounds of the range over which the shape difference measures are displayed. Automatic detection of a threshold is outside the scope of the current activity, but is a future direction to pursue.

We can also generate statistics, such as maximum and average distance, or vector displays, which indicate the relationship between corresponding points. However, vector displays can be very cluttered and confusing, and pseudo-coloring is better able to convey where the surfaces differ.

## 7.3 Shape Difference Measures

No single measure captures the 3-D shape difference, so a set of measures is used to convey the difference between objects. Each measure may detect different qualities of the shape differences and may have different limitations.

The measures are defined as functions over the surface. Each of the shape difference measures is applied to pairs of corresponding points. We do not establish a mutually consistent mesh triangulation, so vertices in one mesh map to (possibly) a point in a face of the other mesh. For visualization purposes, we evaluate the measures at just the vertices of the mesh.

In addition to using the correspondence between the objects, the distance and angle-based measures also use the physical alignment, and can therefore depend on the rigid



alignment chosen. The similarity-based difference measure uses only the correspondence, and therefore is not impacted by the physical alignment of the objects.

The distance and angle-based measures are based only on the relative location and orientation of the corresponding points in some aligned position. Therefore, the region of comparison is just the point. The similarity measure has an associated radius that represents the size of the region of comparison. This variable region of comparison enhances the ability to assess the difference between shapes.

### 7.3.1 Distances Between Corresponding Points

We look at three distance measures that can be applied to corresponding points. The first is the absolute distance measure. This is the Euclidean distance between a point and its corresponding location, applied to the surfaces in their aligned state. It is most useful for detecting missing or moved features. The primary limitations, as with all of the distance measures, is a dependence on the alignment of the surfaces.

The other two distance measures are derived by breaking the absolute distance into normal and tangential components. The sign of the normal component indicates whether the corresponding point is inside (negative) or outside (positive) the object. This is useful in detecting if features grow or shrink. In addition to dependence on the alignment, its usefulness may be limited if the magnitude of the tangential component is large. The tangential distance is a better indicator of whether features occur in different locations in the objects.

Figure 7.1 shows the alignment and distance measures for the lunate dataset. For reference, the maximum dimension for the lunate is about 17.5. The absolute distance measure (top center) shows four areas with the largest distances. The absolute, normal, and tangential distance measures together help identify where major regions are inside (a), outside (b), or misaligned (c) relative to the other surface.

Figure 7.2 shows distance measures for the ulna. The maximum dimension in this case is about 18. The two views on the left show the absolute distance, while the center right and right views show the normal and tangential distances respectively. In

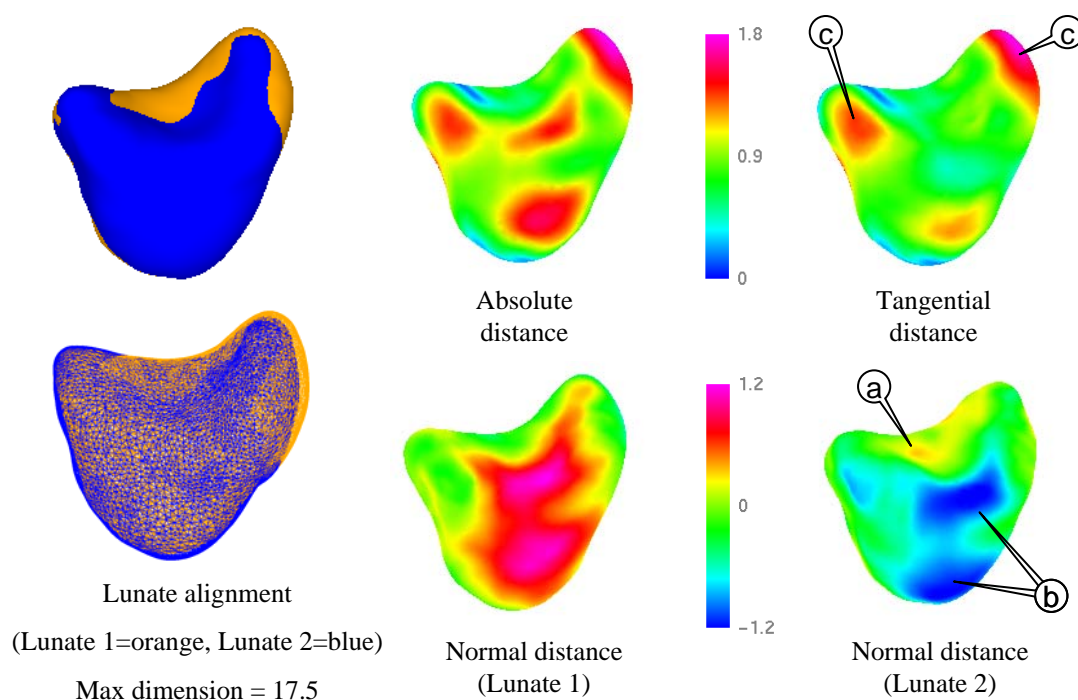


Figure 7.1: Distances between corresponding points for the aligned lunate surfaces. The distance measures can identify where one surface is (a) inside, (b) outside, or (c) misaligned with respect to the other surface.

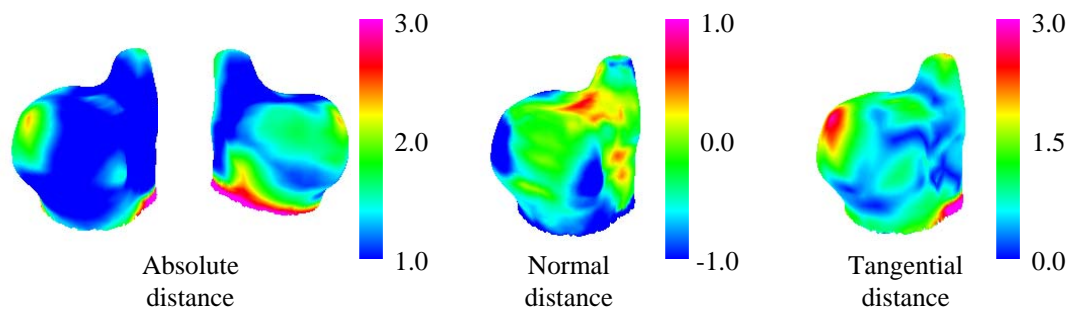


Figure 7.2: Distances between corresponding points for the aligned ulna surfaces. The maximum dimension is approximately 18.

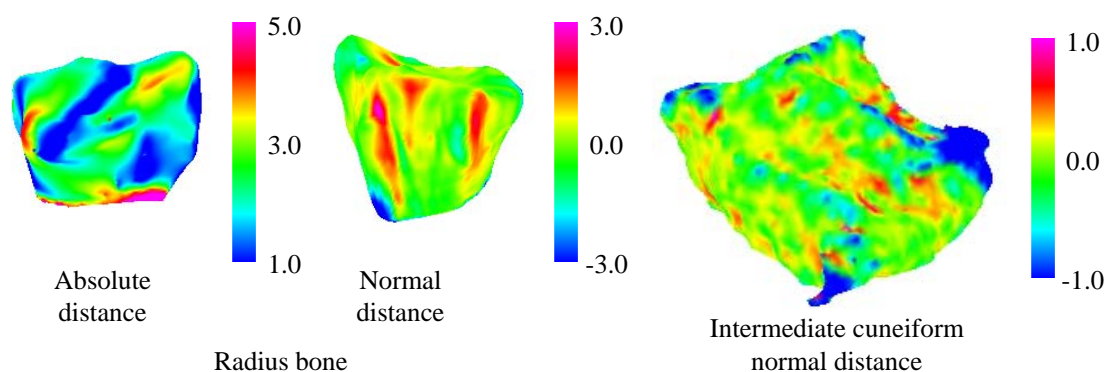


Figure 7.3: Distance measure for corresponding points of the radius and intermediate cuneiform surfaces. The maximum dimensions are approximately 31 for the radius and 37 for the intermediate cuneiform bone.

this case, the normal and tangential distance measures give more specific indication of local differences.

The scale for the absolute distances goes from 1.0 to 3.0 to highlight the differences on the surface. Areas where the absolute distance is less than 1.0 will be blue, and areas greater than 3.0 will be magenta. If the aligned surfaces intersect, the absolute distance is zero, however, there may not be any vertex that falls exactly at a zero location.

Figure 7.3 shows absolute (left) and normal (center) distances for the radius and normal distances for the intermediate cuneiform bone (right). The maximum dimension for the radius and intermediate cuneiform samples are approximately 31 and 37 respectively. The normal distance is best able to detect the change in the ridges of the radius. The negative normal distance for the intermediate cuneiform bone is an indicator of a feature missing from the corresponding object.

### 7.3.2 Change in Surface Normal

Figures 7.4 and 7.5 show the angle between the surface normals at corresponding points for the lunate, radius, and intermediate cuneiform bone samples. This angle highlights where the surface orientation of one object changes with respect to the

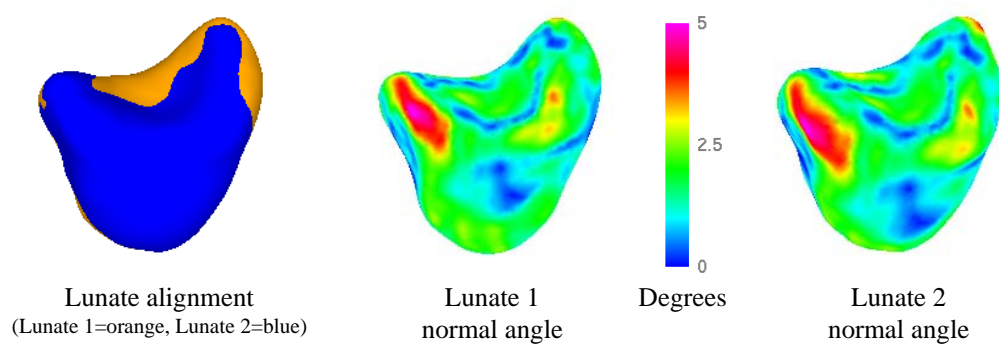


Figure 7.4: Differences in surface normal for corresponding points for the aligned lunate surfaces.

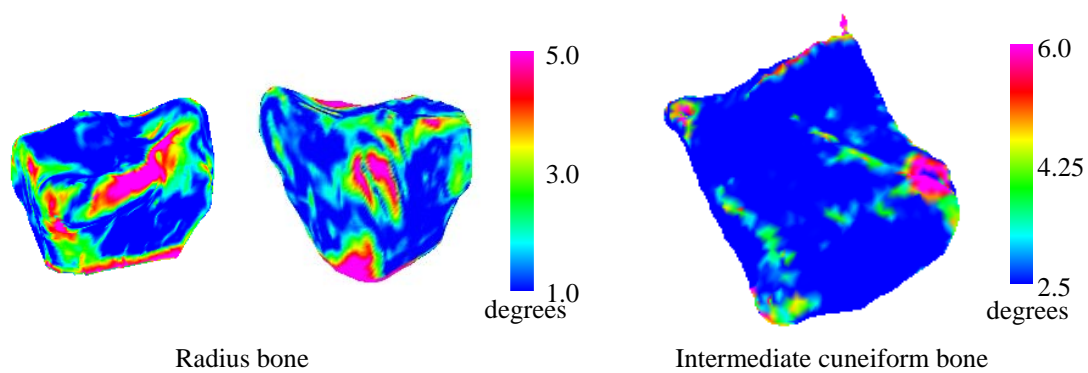


Figure 7.5: Surface normal orientation measure for corresponding points of the radius and intermediate cuneiform surfaces.

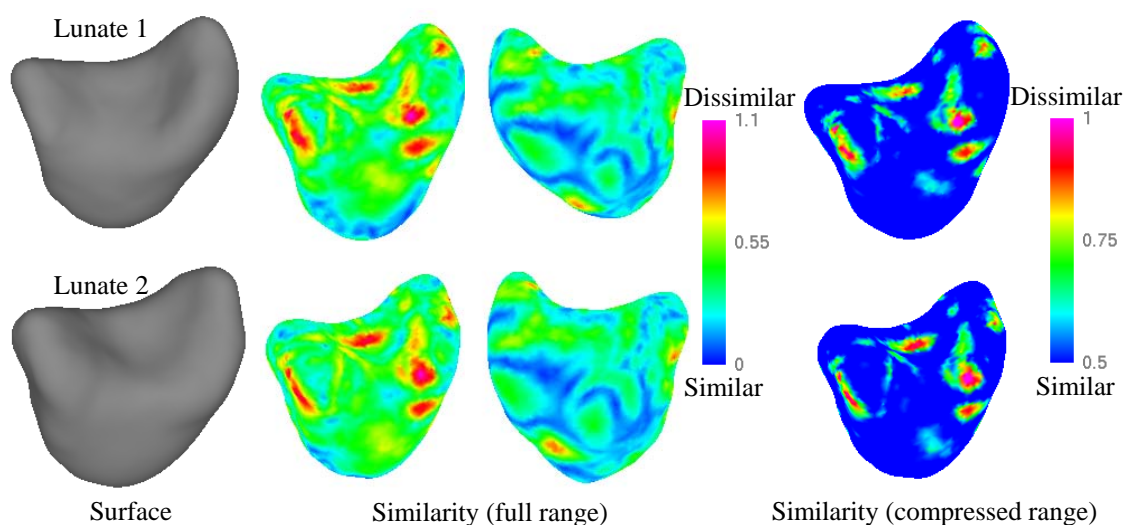


Figure 7.6: Two lunate surfaces and the shape similarity measure for corresponding points plotted for two different scalar ranges.

other object. This is likely to occur when a feature occurs in a different orientation or if a new feature is added.

### 7.3.3 Similarity-based Shape Comparison

Another scalar property used to color the surface is the local shape similarity. Figure 7.6 shows the surfaces and similarity for corresponding points of the lunate surfaces. The similarity shown here is based on the 1-D curvature map and computing the similarity for the range up to one half of the maximum curvature map radius. The maximum curvature map radius is 7 for this case.

The range for the center similarity plots is based on the similarity values for the most similar and the most dissimilar corresponding pairs of points. The similarity plots on the right use a compressed range to bring out the most prominent shape changes.

Figure 7.7, 7.8, and 7.9 show similarity values plotted over a range of curvature map radii for the ulna, radius, and intermediate cuneiform bone surfaces respectively. The

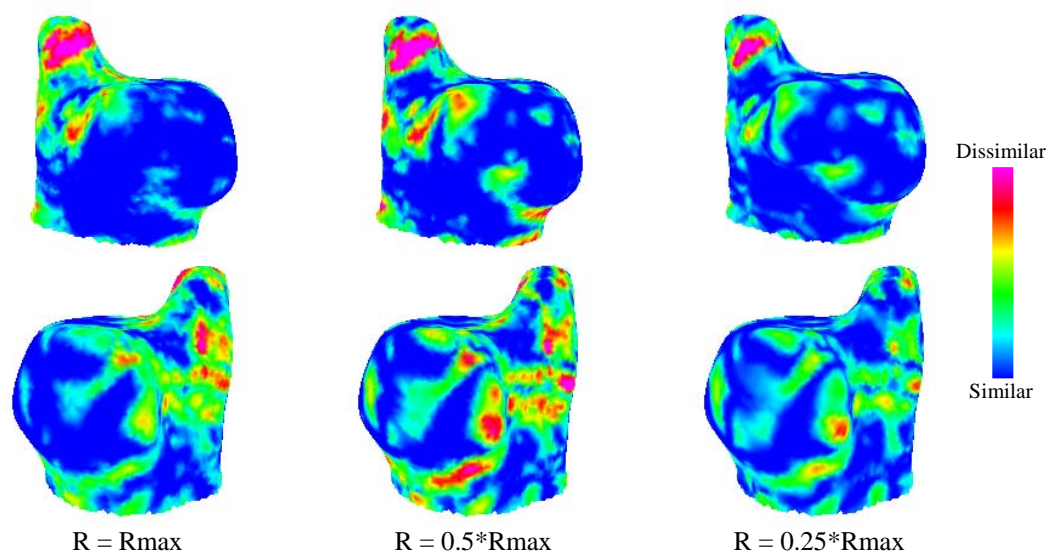


Figure 7.7: Shape similarity measure for corresponding points of the ulna surfaces using different curvature map radii.

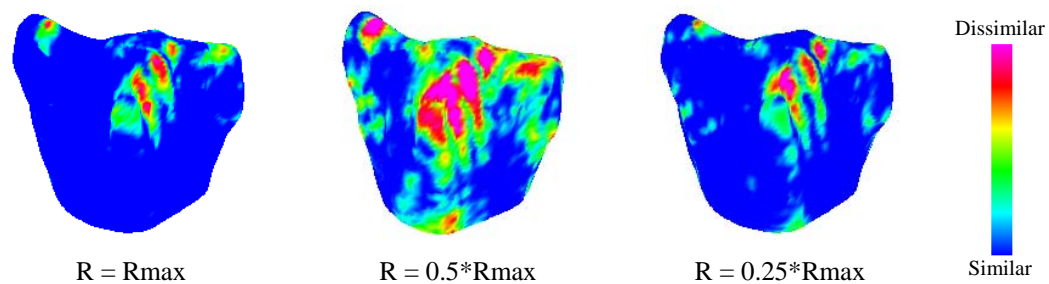


Figure 7.8: Shape similarity measure for corresponding points of the radius surfaces using different curvature map radii.

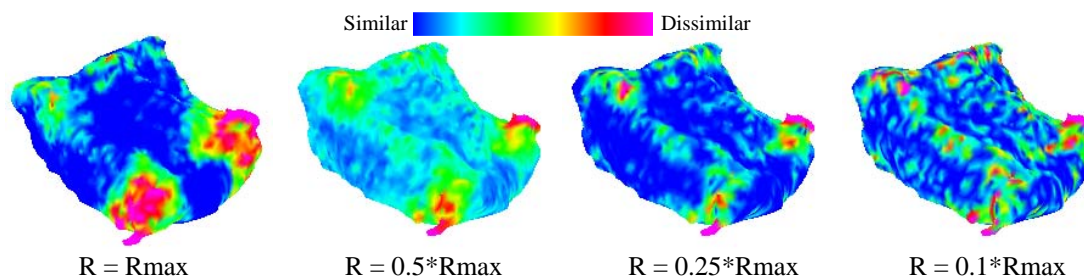


Figure 7.9: Shape similarity measure for corresponding points of the intermediate cuneiform bone surfaces using different curvature map radii.

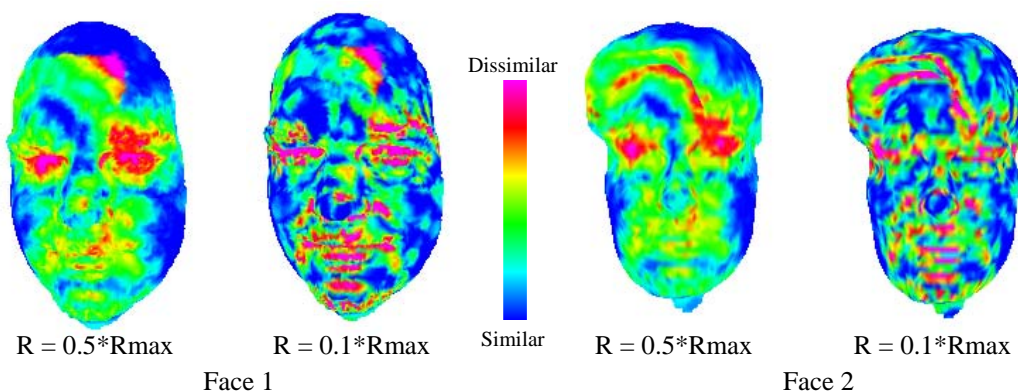


Figure 7.10: Shape similarity measure for corresponding points of two face scans using different curvature map radii.

maximum curvature map radii for these cases were 5 (ulna), 8 (radius), and 10 (intermediate cuneiform). The radii for the similarity plots shown range from 0.1 to 1.0 times the maximum curvature map radius. The larger radii tend to highlight changes over larger regions of the surface, while smaller radii highlight more local shape differences. The two views in the center of Figure 7.9 illustrate how an appropriate radius can make certain shape differences stand out.

Similarity plots for two face scans are shown in Figure 7.10. The maximum curvature map radius is 2 for this case. The correspondence for this case was computed from the physical alignment rather than a spherical parameterization. As a result, more artifacts can be seen, for example in the forehead of Face 1. However, the effect of the different curvature map radii still apply, with the larger radius detecting changes in

larger regions such as the eyes, and a smaller radius highlighting many more smaller shape differences.

## 7.4 Chapter Summary

Several shape difference measures have been applied to the point-to-point correspondence between objects. Visualization of these measures can highlight both small and large regions where the shapes of the objects differ.

The distance measures have a strong dependence on the alignment. As a result, the distance measures are not easy to interpret where regions do not align well. The difference in angle between surface normals at corresponding points appears to be influenced less by the alignment, while the similarity measure is independent of the physical alignment. Naturally, all of the measures rely on the quality of the correspondence provided.

The locations where the shape difference is the greatest tend to dominate. Controlling the curvature map radius for the similarity calculation gives some additional control, but it may also be useful to add clipping to turn off visualization of parts of the surface that are outside the upper and lower thresholds. However, the measures presented are useful for highlighting both similarities and differences for the objects being compared.



# Chapter 8

## Conclusions

### 8.1 Research Summary

This dissertation demonstrates an approach for detailed shape comparison based on establishing a point-to-point correspondence between three-dimensional objects represented by meshes. We develop a new shape representation that is sensitive to subtle shape differences. This shape representation is used to compute shape similarity and to detect features of the objects. The features and similarity measure are used to create a consistent parameterization of the objects that in turn defines the correspondence. The fact that the final correspondence does not rely on a physical alignment of the objects is a key benefit.

The primary contributions of this research are in the areas of curvature estimation, shape similarity measures, feature detection, and alignment and parameterization.

In the area of curvature estimation, this research quantifies the effects of the mesh structure or quality, as well as noise, on the accuracy of existing curvature estimation methods. A suite of cases was developed to evaluate curvature estimation methods, particularly their sensitivity to mesh issues and noise. Evaluating existing curvature methods using this suite yields insight into the issues that are present in real cases. This evaluation process is especially important for investigating the claims of new methods as they are proposed.

Based on the real world issues of noise and mesh quality, we proposed curvature calculation using an N-Ring fit with a natural parameterization. This technique is

robust to noise, and avoids the danger of folding which complicates the choice of the projection direction required for some other fitting methods.

Our contribution in the area of shape similarity measures is the curvature map, a new curvature-based shape descriptor that represents the shape at a point in its local context. The 1-D versions of the curvature map can be compared quickly and require minimal storage. It is easy to vary the size of the region over which the maps are compared. The 2-D version of the curvature map, which is based on geodesic fans, can be used when a more discriminating similarity measure is required. The curvature map has been shown to be useful for feature detection, for assessing the similarity of corresponding points, and for assessing the similarity of pairs of features.

This research provides a feature detection technique well-suited to our shape matching goals. The technique provides multiscale feature detection using curvature maps and the min-cut/max-flow algorithm. It primarily detects strong mean curvature features. Features can be ranked based on a strength associated with each feature. Most importantly, this is a fully automatic method.

The alignment technique presented in this dissertation is based on object features and is independent of the initial orientation of the objects. Pruning based on geometric constraints, feature strength, and the similarity of potential feature pairs makes the process very efficient. Even though many possible alignments are tested, in our experience the best alignment is always found early in the process. Pruning limits testing of alignments to a fraction of the possible alignments.

The alignment also does not depend on the genus of the surfaces. For open surfaces, the alignment and the resulting physical correspondence can be used to extract the common portion of the objects being matched.

We generate the correspondence for genus zero surfaces using a spherical parameterization. From the initial alignment and the identified features, pin points are determined that are used to align the spherical parameterizations for each object. These pin point locations are adjusted to improve the shape similarity of the corresponding locations. Additional pin points can be added and adjusted to promote improved similarity in regions where no feature pairs are located. Using similarity to adjust the parameterization is an important factor in generating a useful correspondence.

Three-dimensional shape matching and analysis is challenging. The new curvature-based shape matching technique presented in this dissertation is able to distinguish and measure differences between similar shaped objects. This shape matching technique has been demonstrated on data from scans of several bone samples. It provides greater detail about the differences between objects, which is not available from existing methods. This technique has also been shown to be robust to resolution differences.

## 8.2 Ideas for Future Work

A logical follow-on to this work would be to extract information about regions from the point-to-point shape similarity data. Here, the graph cut approach used for feature detection might be applied to the shape difference measures to identify regions that can be classified based on the distance, orientation, and similarity properties.

Calculating the correspondence between objects of genus zero relies on the ability to generate the spherical parameterization of each object. Improving the robustness of the spherical parameterization algorithm would increase the overall robustness of this approach. The method could also be expanded to higher genus by parameterizing with an  $n$ -holed tori, or to surfaces homeomorphic to a disk by parameterizing with a subset the plane.

It is generally beneficial to have a small number of ordered features, but occasionally not enough features are present. An alternate technique could be used to supply additional features when necessary, for example the scale invariant feature transform (SIFT) [87].

Our shape comparison approach could also be extended to account for scale variation. There are other measures of curvature, for example based on the angle between surface normals at the three vertices of a triangular face, which are independent of scale. Such curvature measures might be adapted to generate a scale independent curvature map that could be used for shape comparison.

*Acknowledgments*

This work was partially supported by NSF Grant 049856. The wrist bone data was provided through NIH Grant AR44005, PI: J.J. Crisco, Brown Medical School/Rhode Island Hospital. The intermediate cuneiform bone data was provided by Dr. David Sinacore and Paul K. Commean, Mallinckrodt Institute of Radiology, Washington University School of Medicine. Thanks also go to Vladimir Kolmogorov for the min-cut/max-flow code, Cyberware for the human head scans, and the Stanford Scanning Repository for the bunny data set.

# References

- [1] Nabih N. Abdelmalek. Algebraic error analysis for surface curvatures and segmentation of 3-d range images. *Pattern Recognition, Vol. 23*, pages 807–817, September 1989.
- [2] Helmut Alt and Leonidas J. Guibas. Discrete geometric shapes: Matching, interpolation, and approximation A survey. In *Handbook of Computational Geometry*, pages 121–153. North Holland, 1999.
- [3] N. Amenta, S. Choi, T. K. Dey, and N. Leekha. A simple algorithm for homeomorphic surface reconstruction. In *In Proceedings of the 16th Annual ACM Symposium on Computational Geometry (SCG '00)*, pages 213–222, 2000.
- [4] D. Anguelov, D. Koller, P. Srinivasan, S. Thrun, H.-C. Pang, and J. Davis. The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces. In *Advances in Neural Information Processing Systems (NIPS 2004)*, Vancouver, Canada, 2005.
- [5] Arul Asirvatham, Emil Praun, and Hugues Hoppe. Consistent spherical parameterization. *Lecture Notes in Computer Science*, 3515:265–272, 2005.
- [6] Brian B. Avants and James C. Gee. The shape operator for differential analysis of images. In *IPMI*, pages 101–113, 2003.
- [7] A. J. Baddeley. An error metric for binary images. In W. Förstner and H. Ruwiedel, editors, *Robust Computer Vision: Quality of Vision Algorithms*, pages 59–78. Wichmann, Karlsruhe, 1992.
- [8] D. H. Ballard. Generalized hough transform to detect arbitrary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13 (2):111–122, 1981.
- [9] T. F. Banchoff. Critical points and curvature for emmbedded polyhedra. *Differential Geometry*, 1(1):257–268, 1967.
- [10] E. Bengoetxea, P. Larraaga, I. Bloch, and A. Perchant. Image recognition with graph matching using estimation of distribution algorithms. In *Proceedings of the Medical Image Understanding and Analysis (MIUA 2001)*, pages 89–92, Birmingham, UK, 2001.

- [11] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, August 1999.
- [12] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.
- [13] Dmitriy Bespalov, Ali Shokoufandeh, William C. Regli, and Wei Sun. Scale-space representation of 3d models and topological matching. In *SM '03: Proceedings of the eighth ACM symposium on Solid modeling and applications*, pages 208–215, New York, NY, USA, 2003. ACM Press.
- [14] Peter Biber and Wolfgang Straßer. Solving the correspondence problem by finding unique features. In *16th International Conference on Vision Interface*, 2003.
- [15] Henning Biermann, Ioana Martin, Fausto Bernardini, and Denis Zorin. Cut-and-paste editing of multiresolution surfaces. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 312–321. ACM Press, 2002.
- [16] J. Bloomenthal and C. Lim. Skeletal methods of shape manipulation. In *International Conference on Shape Modeling and Applications*, page 44, 1999.
- [17] Georges-Pierre Bonneau and Alexandre Gerussi. Level of detail visualization of scalar data sets on irregular surface meshes. In David Ebert, Hans Hagen, and Holly Rushmeier, editors, *IEEE Visualization '98*, pages 73–78, 1998.
- [18] F.L. Bookstein. Principal warps : Thin-plate splines and the decomposition of deformations. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*. 11, pages 567–585, 1989.
- [19] Yuri Boykov and Marie-Pierre Jolly. Interactive organ segmentation using graph cuts. In *MICCAI '00: Proceedings of the Third International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 276–286, London, UK, 2000. Springer-Verlag.
- [20] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137, 2004.
- [21] C. Brechbühler, G. Gerig, and O. Kübler. Parametrization of closed surfaces for 3-d shape-description. *CVIU*, 61(2):154–170, March 1995.

- [22] Ulrich Brehm and Wolfgang Kühnel. Smooth approximation of polyhedral surfaces regarding curvatures. *Geometriae Dedicata*, 12:435–461, 1982.
- [23] Benedict Brown and Szymon Rusinkiewicz. Non-rigid range-scan alignment using thin-plate splines. In *Symposium on 3D Data Processing, Visualization, and Transmission*, pages 759–765, September 2004.
- [24] F. Cazals and M. Pouget. Estimating differential quantities using polynomial fitting of osculating jets. In *SGP '03: Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 177–187. Eurographics Association, 2003.
- [25] Frédéric Cazals, Frédéric Chazal, and Thomas Lewiner. Molecular shape analysis based upon the morse-smale complex and the connolly function. In *Proc. 19th Annu. ACM Sympos. Comput. Geom.*, pages 237–246, 2003.
- [26] Ding-Yun Chen and Ming Ouhyoung. A 3d object retrieval system based on multi-resolution reeb graph. In *Proc. of Computer Graphics Workshop*, pages 16–, June 2002.
- [27] X. Chen and F. Schmitt. Intrinsic surface properties from surface triangulation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 739–743, 1992.
- [28] H. Chui and A. Rangarajan. A new algorithm for non-rigid point matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 44–51, 2000.
- [29] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. In *Computer Vision and Image Understanding*, volume 89, pages 114–141, 2003.
- [30] David Cohen-Steiner and Jean-Marie Morvan. Restricted delaunay triangulations and normal cycle. In *SCG '03: Proceedings of the nineteenth annual symposium on Computational geometry*, pages 312–321. ACM Press, 2003.
- [31] A G Cohn and S M Hazarika. Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1-2):1–29, 2001.
- [32] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models—their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.
- [33] Timothy F. Cootes, Andrew Hill, Christopher J. Taylor, and J. Haslam. Use of active shape models for locating structures in medical images. *Image Vision Comput.*, 12(6):355–365, 1994.

- [34] J. Corney, D. Clark H. Rea, John Pritchard, M. Breaks, and R. MacLeod. Coarse filters for shape matching. In *IEEE Computer Graphics and Applications*, pages 65–73, May/June 2002.
- [35] G.S. Cox. Template matching and measures of match in image processing. In *Review*. University of Cape Town, South Africa, 1995.
- [36] P. Csakany and A.M. Wallace. Computation of local differential properties on irregular meshes. In *Proc. of IMA Conference on Mathematics of Surfaces (NIPS 2000)*, pages 19–33, Cardiff, 2000.
- [37] Rhodri H. Davies, Tim F. Cootes, and Chris J. Taylor. A minimum description length approach to statistical shape modelling. *Lecture Notes in Computer Science*, 2082:50–63, 2001.
- [38] Rhodri H. Davies, Carole J. Twining, Tim F. Cootes, John C. Waterton, and Chris J. Taylor. 3d statistical shape models using direct optimisation of description length. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision*, volume 3, pages 3–21, London, UK, 2002. Springer-Verlag.
- [39] M. Desbrun, M. Meyer, and P. Alliez. Intrinsic parameterizations of surface meshes. *Computer Graphics Forum*, 21:209–218, 2002.
- [40] T. Dey, J. Giesen, and S. Goswami. Shape segmentation and matching with flow discretization. In Frank K. H. A. Dehne, Jrg-Rdiger Sack, and Michiel H. M. Smid, editors, *Proceedings of the Workshop on Algorithms and Data Structures (WADS)*, Lecture Notes in Computer Science 2748, pages 25–36, 2003.
- [41] M. P. Do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice Hall, Englewood Cliffs, NJ, 1976.
- [42] I. Douros and B. Buxton. Three-dimensional surface curvature estimation using quadric surface patches. *Scanning 2002 Proceedings, Paris*, May 2002.
- [43] N. Dyn, K. Hormann, S.-J. Kim, and D. Levin. Optimizing 3D triangulations using discrete curvature analysis. In T. Lyche and L. L. Schumaker, editors, *Mathematical Methods for Curves and Surfaces: Oslo 2000*, Innovations in Applied Mathematics, pages 135–146. Vanderbilt University Press, Nashville, TN, 2001.
- [44] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, , and W. Stuetzle. Multiresolution analysis of arbitrary meshes. In *Siggraph 1995, Computer Graphics Proceedings*, pages 173–182. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 1995.



- [45] Alon Efrat and Alon Itai. Improvements on bottleneck matching and related problems using geometry. In *Symposium on Computational Geometry*, pages 301–310, 1996.
- [46] P.A. van den Elsen, J.B.A. Maintz, E.J.D. Pol, and M.A. Viergever. Medical image matching - a review with classification. In *IEEE Engineering in Medicine and Biology*, pages 26–39, March 1993.
- [47] D. Eppstein. Subgraph isomorphism in planar graphs and related problems. In *SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms)*, 1995.
- [48] B. Falcidieno and M. Spagnuolo. Geometric reasoning for the extraction of surface shape properties. In *Communicating With Virtual Worlds*. Springer-Verlag, 1993.
- [49] G. Farin. Shape. *Mathematics Unlimited - 2001 and Beyond*, pages 463–477, 2001.
- [50] Andrew W. Fitzgibbon, Maurizio Pilu, and Robert B. Fisher. Direct least square fitting of ellipses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):476–480, 1999.
- [51] M. Floater and M. Reimers. Meshless parameterization and surface reconstruction.
- [52] M. S. Floater. Parametric tilings and scattered data approximation. In *International journal of Shape Modeling 4*, pages 165–187, 1998.
- [53] M. S. Floater. Meshless parameterization and b-spline surface reconstruction. In *The Mathematics of Surfaces IX, 2000*, pages 1–18, 2000.
- [54] Michael S. Floater. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14(4):231–250, 1997.
- [55] P. J. Flynn and A. K. Jain. On reliable curvature estimation. *IEEE Proceedings of Computer Vision and Pattern Recognition*, pages 110–116, June 1989.
- [56] C. Fookes and A. Maeder. Local non-rigid image registration using mutual information. In *16th International Conference on Vision Interface*, 2003.
- [57] Andrea Frome, Daniel Huber, Ravi Kolluri, Thomas Bülow, and Jitendra Malik. Recognizing objects in range data using regional point descriptors. In *ECCV (3)*, pages 224–237, 2004.
- [58] Ran Gal and Daniel Cohen-Or. Salient geometric features for partial shape matching and similarity. *j-TOG*, 25(1):130–150, January 2006.

- [59] T. D. Gatzke and C. M. Grimm. Improved curvature estimation on triangular meshes. Technical Report WUCSE-2004-9, Washington University, St. Louis Missouri, 2004.
- [60] Timothy Gatzke and Cindy Grimm. Estimating curvature on triangular meshes. *International Journal of Shape Modeling*, 12(1):1–29, June 2006.
- [61] Timothy Gatzke, Steve Zelinka, Cindy Grimm, and Michael Garland. Curvature maps for local shape comparison. In *Shape Modeling International*, pages 244–256, June 2005.
- [62] Natasha Gelfand, Niloy J. Mitra, Leonidas J. Guibas, and Helmut Pottmann. Robust global registration. In *Symposium on Geometry Processing*, pages 197–206, 2005.
- [63] Steven Gold and Anand Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377–388, 1996.
- [64] Jack Goldfeather. Understanding errors in approximating principal direction vectors. *preprint*, aug 2001.
- [65] Jack Goldfeather and Victoria Interrante. A novel cubic-order algorithm for approximating principal direction vectors. *ACM Trans. Graph.*, 23(1):45–63, 2004.
- [66] Polina Golland, W. Eric L. Grimson, Martha Elizabeth Shenton, and Ron Kikinis. Small sample size learning for shape analysis of anatomical structures. In *MICCAI*, pages 72–82, 2000.
- [67] Cindy Grimm and John Hughes. Modeling surfaces of arbitrary topology. In *Siggraph*, pages 359–369, July 1995.
- [68] Steven Haker, Allen Tannenbaum, and Ron Kikinis. Mass preserving mappings and image registration. In *MICCAI*, pages 120–127, 2001.
- [69] B. Hamann. Curvature approximation for triangulated surfaces. *Geometric modelling*, pages 139–153, 1993.
- [70] E. Hameiri and I. Shimshoni. Estimating the principal curvatures and the darboux frame from real 3d range data. In *First International Symposium on 3D Data Processing Visualization and Transmission*, pages 258–267, 2002.
- [71] A. Hertzmann and D. Zorin. Illustrating smooth surfaces. In Kurt Akeley, editor, *Siggraph 2000, Computer Graphics Proceedings*, pages 517–526. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

- [72] Günter Hetzel, Bastian Leibe, Paul Levi, and Bernt Schiele. 3d object recognition from range images using local feature histograms. In *CVPR (2)*, pages 394–399, 2001.
- [73] L. S. Hibbard and R. A. Hawkins. Objective image alignment for three-dimensional reconstruction of digital radiograms. *Journal of Neuroscience Methods*, Vol. 26, pages 55–75, 1988.
- [74] M. Hilaga, Y. Schinagawa, T. Kohmura, and T. L. Kuni. Topology matching for fully automatic similarity estimation of 3d shapes. In *Computer Graphics (SIGGRAPH)*, pages 203–212, 2001.
- [75] A. Hilton, J. Illingworth, and T. Windeatt. Statistics of surface curvature estimates. *Pattern Recognition*, Vol 28, pages 1201–1221, February 1995.
- [76] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *Computer Graphics (SIGGRAPH '92 Proceedings)*, Vol. 26, No. 2, pages 71–78, July 1992.
- [77] Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America. A, Optics and image science*, 4(4):629–642, April 1987.
- [78] Josef Hoschek, Dieter Lasser, and Larry L. Schumaker. *Fundamentals of computer aided geometric design*. A. K. Peters, Ltd., 1993.
- [79] D. Huttenlocher, D. Klanderman, and A. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, September 1993.
- [80] Daniel P. Huttenlocher and Shimon Ullman. Recognizing solid objects by alignment with an image. *Int. J. Comput. Vision*, 5(2):195–212, 1990.
- [81] Naoyuki Ichimura. Feature point tracking based on feature point extraction by frame. In *AIST:tech. report AIST02-J-00002-3*, 2002.
- [82] Victoria Interrante, Henry Fuchs, and Stephen M. Pizer. Enhancing transparent skin surfaces with ridge and valley lines. In *IEEE Visualization*, pages 52–, 1995.
- [83] Victoria L. Interrante. Illustrating surface shape in volume data via principal direction-driven 3D line integral convolution. *Computer Graphics*, 31(Annual Conference Series):109–116, 1997.
- [84] A. Johnson and M. Hebert. Recognizing objects by matching oriented points. In *CVPR '97*, pages 684–689, 1997.

- [85] Sagi Katz, George Leifman, and Ayellet Tal. Mesh segmentation using feature point and core extraction. *The Visual Computer*, 21(8-10):649–658, 2005.
- [86] Michael Kazhdan and Thomas Funkhouser. Harmonic 3d shape matching. In *SIGGRAPH 2002 Technical Sketch*, July 2002.
- [87] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2:506–513, 2004.
- [88] Duck Hoon Kim, Il Dong Yun, and Sang Uk Lee. A new shape decomposition scheme for graph-based representation. *Pattern Recognition*, 38(5):673–689, 2005.
- [89] Junhwan Kim and Ramin Zabih. A segmentation algorithm for contrast-enhanced images. In *Ninth IEEE International Conference on Computer Vision (ICCV'03)*, volume 1, pages 502–509, 2003.
- [90] Philip N. Klein, Thomas B. Sebastian, and Benjamin B. Kimia. Shape matching using edit-distance: an implementation. In *Symposium on Discrete Algorithms*, pages 781–790, 2001.
- [91] Philip N. Klein, Srikanta Tirthapura, Daniel Sharvit, and Benjamin B. Kimia. A tree-edit-distance algorithm for comparing simple, closed shapes. In *Symposium on Discrete Algorithms*, pages 696–704, 2000.
- [92] M. Körtgen, G.-J. Park, M. Novotni, and R. Klein. 3d shape matching with 3d shape contexts. In *The 7th Central European Seminar on Computer Graphics*, April 2003.
- [93] Vladislav Kraevoy and Alla Sheffer. Cross-parameterization and compatible remeshing of 3d models. *ACM Trans. Graph.*, 23(3):861–869, 2004.
- [94] B. V. K. Vijaya Kumar, D. Casasent, , and H. Murakami. Principal-component imagery for statistical pattern recognition correlators. In *Optical Engineering*, 21(1):, pages 43–47, 1982.
- [95] Shang-Hong Lai and Ming Fang. Robust and efficient image alignment with spatially varying illumination models. In *CVPR*, volume 2, pages 167–173, 1999.
- [96] Y. Lamdan and H. Wolfson. Geometric hashing: a general and efficient model-based recognition scheme. *Intl. Conf. on Computer Vision (ICCV)*, pages 238–249, 1988.
- [97] Chang Ha Lee, Amitabh Varshney, and David W. Jacobs. Mesh saliency. *Trans. Graph.*, 24(3):659–666, 2005.

- [98] Bruno Levy, Sylvain Petitjean, Nicolas Ray, and Jerome Maillot. Least squares conformal maps for automatic texture atlas generation. In *SIGGRAPH 2002, Computer Graphics Proceedings*. ACM Press / ACM SIGGRAPH, 2002.
- [99] Frederic F. Leymarie and Benjamin B. Kimia. The shock scaffold for representing 3d shape. In *IWVF*, pages 216–228, 2001.
- [100] Xinju Li and Igor Guskov. Multi-scale features for approximate alignment of point-based surfaces. In *Proceedings of 3rd ACM Siggraph/Eurographics Symposium on Geometry Processing*, July 2005.
- [101] C. Lin and M.J. Perry. Shape description using surface triangulation. In *In Workshop on Computer Vision: Representation and Control*, pages 38–43, 1982.
- [102] Y. Liu and W. Heidrich. Interactive 3d model acquisition and registration. In *Pacific Graphics '03*, 2003.
- [103] S. K. Lodha and R. Franke. Scattered data techniques for surfaces. *Scientific Visualization Dagstuhl '97, Hagen, Nielson, and Post (eds)*, pages 181–222, August 1997.
- [104] S. Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31(8):983–1001, 1998.
- [105] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *Computer Graphics (Proceedings of SIGGRAPH '87)*, 21(4):163–169, 1987.
- [106] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.
- [107] Xiaoguang Lu and Anil K. Jain. Deformation analysis for 3d face matching. In *WACV/MOTION*, pages 99–104, 2005.
- [108] Kwan-Liu Ma and Victoria Interrante. Extracting feature lines from 3d unstructured grids. In *IEEE Visualization*, pages 285–292, 1997.
- [109] J. Maillot, H. Yahia, and A. Verroust. Interactive texture mapping. In *SIGGRAPH 1993, Computer Graphics Proceedings*. ACM Press / ACM SIGGRAPH, 1993.
- [110] J. B. Antoine Maintz, Petra A. van den Elsen, and Max A. Viergever. Registration of 3d medical images using simple morphological tools. In *IPMI*, pages 204–217, 1997.

- [111] A. Mangan and R. Whitaker. Surface segmentation using morphological watersheds. In *IEEE Visualization '98: Late Breaking Topics*, pages 29–32, October 1998.
- [112] Alan P. Mangan and Ross T. Whitaker. Partitioning 3d surface meshes using watershed segmentation. In *IEEE Transactions on Visualization and Computer Graphics* 5(4), pages 308–321, 1999.
- [113] Alan McIvor and Robert J. Valkenburg. A comparison of local surface geometry estimation methods. *Machine Vision and Applications*, 10(1):17–26, 1997.
- [114] Alan M. McIvor, David W. Penman, and Peter T. Waltenberg. Simple surface segmentation. In *DICTA/IVCNZ97*, pages 141–146, Massey University, New Zealand, December 1997.
- [115] D. S. Meek and D. J. Walton. On surface normal and gaussian curvature approximations given data sampled from a smooth surface. *Computer Aided Geometric Design* 17, pages 521–543, February 2000.
- [116] M. Meyer, H. Lee, A. H. Barr, and M. Desbrun. Generalized barycentric coordinates for irregular n-gons. In *Journal Of Graphic Tools*, 2002.
- [117] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. *VisMath.*, 2002.
- [118] Farzin Mokhtarian, Nasser Khalili, and Peter Yuen. Multi-scale 3-d free-form surface smoothing. In *BMVC*, 1998.
- [119] G. Mori, S. Belongie, and J. Malik. Shape contexts enable efficient retrieval of similar shapes. In *CVPR 1*, pages 723–730, 2001.
- [120] M. Mortara and G. Patanè. Affine-invariant skeleton of 3d shapes. In *Shape Modeling International 2002 (SMI'02)*, page 245, 2002.
- [121] Michela Mortara, Giuseppe Patanè, Michela Spagnuolo, Bianca Falcidieno, and Jarek Rossignac. Blowing bubbles for multi-scale analysis and decomposition of triangle meshes. *Algorithmica*, 38(1):227–248, 2003.
- [122] P. Mus, F. Sur, F. Cao, and Y. Gousseau. Unsupervised thresholds for shape matching. In *IEEE Intl. Conf. on Image Processing, ICIP, 2003.*, pages 647–650, 2003.
- [123] H. F. Neemuchwala, A. O. Hero, and P. L. Carson. Image registration using entropic graph-matching criteria. In *36th Asilomar Conf. on Signals Systems and Computers*, Pacific Grove, CA, November 2002.

- [124] G. Nielson. Volume modelling. In Arie Kaufman Min Chen and Roni Yagel, editors, *Volume Graphics*, pages 29–50. Springer, 1999.
- [125] Wesley A. Niewoehner. Behavioral inferences from the skhul/qafzeh early modern human hand remains. In *Proceedings of the National Academy of Sciences of the United States of America*, February 2001.
- [126] M. Novotni and R. Klein. 3D Zernike descriptors for content based shape retrieval. In *The 8th ACM Symposium on Solid Modeling and Applications*, June 2003.
- [127] Ryutarou Ohbuchi, Tomo Otagiri, Masatoshi Ibato, and Tsuyoshi Takei. Shape-similarity search of three-dimensional models using parameterized statistics. In *PG '02: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, page 265, Washington, DC, USA, 2002. IEEE Computer Society.
- [128] Peter J. Olver, Guillermo Sapiro, and Allen Tannenbaum. Affine invariant detection: Edges, active contours, and segments. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '96)*, pages 520–525, 1996.
- [129] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Matching 3D models with shape distributions. In *SMI '01: Proceedings of the International Conference on Shape Modeling & Applications*, pages 154–166, Washington, DC, USA, 2001. IEEE Computer Society.
- [130] Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin. Shape distributions. In *ACM Transactions on Graphics*, 21(4), pages 807–832, October 2002.
- [131] A. Pang, J. Furman, and W. Nuss. Data quality issues in visualization. In *SPIE Proceedings on Visual Data Exploration and Analysis*, volume 2178, pages 12–23, 1994.
- [132] A. Perchant and I. Bloch. A new definition for fuzzy attributed graph homomorphism with application to structural shape recognition in brain imaging. In *16th IEEE Instrumentation and Measurement Technology Conference (IMTC'99)*, pages 1801–1806, Venice, Italy, May 1999.
- [133] Aymeric Perchant and Isabelle Bloch. Graph fuzzy homomorphism interpreted as fuzzy association graphs. In *ICPR 2000*, pages 6042–6045, 2000.
- [134] B. M. Planitz, A. J. Maeder, and J. A. Williams. The correspondence framework for 3d surface matching algorithms. *Comput. Vis. Image Underst.*, 97(3):347–383, 2005.

- [135] Birgit M. Planitz, Anthony J. Maeder, and John A. Williams. Intrinsic correspondence using statistical signature-based matching for 3d surfaces. In *Australian Pattern Recognition Society (APRS) Workshop on Digital Image Computing (WDIC)*, 2003.
- [136] R. Pohle, T. Behlau, and K. D. Toennies. Segmentation of 3D medical image data sets with a combination of region-based initial segmentation and active surfaces. In M. Sonka and J. M. Fitzpatrick, editors, *Proceedings of the SPIE Medical Imaging 2003: Image Processing*, volume 5032, pages 1225–1232, May 2003.
- [137] Konrad Polthier and Markus Schmies. Straightest geodesics on polyhedral surfaces. *Mathematical Visualization*, page 391, 1998.
- [138] Arthur R. Pope. Model-based object recognition - a survey of recent research. Technical Report TR-94-04, University of British Columbia, Vancouver, BC, Canada, Canada, 1994.
- [139] Vaughan Pratt. Direct least-squares fitting of algebraic surfaces. *Computer Graphics*, 21(4):145–152, 1987.
- [140] E. Praun and H. Hoppe. Spherical parameterization and remeshing. In *SIGGRAPH 2003, Computer Graphics Proceedings*. ACM Press / ACM SIGGRAPH, 2003.
- [141] Emil Praun, Wim Sweldens, and Peter Schröder. Consistent mesh parameterizations. In Eugene Fiume, editor, *SIGGRAPH 2001, Computer Graphics Proceedings*, pages 179–184. ACM Press / ACM SIGGRAPH, 2001.
- [142] Sandeep Pulla, Anshuman Razdan, and Gerald Farin. Improved curvature estimation for watershed segmentation of 3-dimensional meshes. In *IEEE Transactions on Visualization and Computer Graphics*, 2004. submitted for publication.
- [143] M. Quicken, C. Brechbühler, J. Hug, H. Blattman, and G. Székely. Parameterization of closed surfaces for parametric surface description. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR 2000*, volume 1, pages 354–360. IEEE Computer Society, June 2000.
- [144] S. Rachev. The monge-kantorovich mass transference problem and its stochastic applications. *Theory of Probability and Applications*, 29:647–676, 1985.
- [145] Szymon Rusinkiewicz. Estimating curvatures and their derivatives on triangle meshes. In *2nd International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT 2004)*, pages 486– 493, September 2004.



- [146] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *Third International Conference on 3D Digital Imaging and Modeling (3DIM)*, pages 145–152, June 2001.
- [147] Shadi Saba, Irad Yavneh, Craig Gotsman, and Alla Sheffer. Practical spherical embedding of manifold triangle meshes. In *SMI '05: Proceedings of the International Conference on Shape Modeling and Applications 2005 (SMI' 05)*, pages 258–267, Washington, DC, USA, 2005. IEEE Computer Society.
- [148] Pedro V. Sander, John Snyder, Steven J. Gortler, and Hugues Hoppe. Texture mapping progressive meshes. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 409–416, New York, NY, USA, 2001. ACM Press.
- [149] Peter T. Sander. Generic curvature features from 3-d images. *IEEE Transactions on Systems, Man, and Cybernetics*, pages 1623–1635, November 1989.
- [150] John Schreiner, Arul Asirvatham, Emil Praun, and Hugues Hoppe. Inter-surface mapping. *ACM Trans. Graph.*, 23(3):870–877, 2004.
- [151] S. Sclaroff and A. P. Pentland. Modal matching for correspondence and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6):545–561, 1995.
- [152] I.O. Sebe and G.Q. Chen. A novel affine template matching method and its application to real-time tracking. In *STMicroelectronics Technical Report*, San Diego, October 2002.
- [153] L. Shams, M. J. Brady, and S. Schaal. Graph matching vs mutual information maximization for object detection. In *Neural Networks, 14(3)*, pages 345–354, 2001.
- [154] L. G. Shapiro and R. M. Haralick. Structural descriptions and inexact matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(9):504–519, 1981.
- [155] A. Sheffer and E. de Sturler. Surface parameterization for meshing by triangulation flattening. In *Proc. 9th International Meshing Roundtable*, pages 161–172, 2000.
- [156] A. Sheffer and E. de Sturler. Parameterization of faceted surfaces for meshing using angle based flattening. In *Engineering with Computers, 17 (3)*, pages 326–337, 2001.
- [157] A. Sheffer and E. De Sturler. Smoothing an overlay grid to minimize linear distortion in texture mapping. *ACM Trans. Graph.*, 21(4):874–890, 2002.

- [158] Christian R. Shelton. Morphable surface models. *Int. J. Comput. Vision*, 38(1):75–91, 2000.
- [159] H.-Y. Shum, M. Hebert, and K. Ikeuchi. On 3d shape similarity. In *Proc. IEEE Computer Vision and Pattern Recognition*, pages 526–531, 1996.
- [160] Kaleem Siddiqi, Ali Shokoufandeh, Sven J. Dickinson, and Steven W. Zucker. Shock graphs and shape matching. In *ICCV*, pages 222–229, 1998.
- [161] Richard Southern, Patrick Marais, and Edwin Blake. Wavelets for multi-resolution analysis of triangular surface meshes. Technical Report CS00-11-00, Computer Science Department, University of Cape Town, 2000.
- [162] Lucio De Souza. Similarity-based versus template matching-based methodologies for image alignment of polyhedral-like objects under noisy conditions. In *Proc. X Brazilian Symposium on Computer Graphics and Image Processing*, 1997.
- [163] Carsten Steger. Subpixel-precise extraction of watersheds. In *ICCV (2)*, pages 884–890, 1999.
- [164] E. M. Stokely and S. Y. Wu. Surface parameterization and curvature measurement of arbitrary 3-d objects: Five practical methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 833–839, August 1992.
- [165] Martin Styner, Kumar T. Rajamani, Lutz-Peter Nolte, Gabriel Zsemlye, Gábor Székely, Christopher J. Taylor, and Rhodri H. Davies. Evaluation of 3d correspondence methods for model building. In *IPMI*, pages 63–75, 2003.
- [166] Jayashree Subrahmonia, David B. Cooper, and Daniel Keren. Practical reliable bayesian recognition of 2d and 3d objects using implicit polynomials and algebraic invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(5):505–519, 1996.
- [167] Yiyong Sun, Joon Ki Paik, Andreas Koschan, David L. Page, and Mongi A. Abidi. Point fingerprint: A new 3-d object representation scheme. *IEEE Trans. Systems, Man, and Cybernetics, Part B*, 33(4):712–717, 2003.
- [168] Tatiana Surazhsky, Evgeny Magid, Octavian Soldea, Gershon Elber, and Ehud Rivlin. A comparison of gaussian and mean curvatures triangular meshes. *IEEE International Conference on Robotics and Automation (ICRA2003)*, pages 1021–1026, May 2003.
- [169] S. Takahashi, N. Ohta, H. Nakamura, Y. Takeshima, and I. Fujishiro. Modeling surperspective projection of landscapes for geographical guide-map generation. *Computer Graphics Forum*, 21(3):259–268, 2002.

- [170] S. Takahashi, Y. Takeshima, and I. Fujishiro. Topological volume skeletonization and its application to transfer function design. *Graphical Models*, 66(1):24–49, 2004.
- [171] Y. Takeshima, S. Takahashi, I. Fujishiro, and G. M. Nielson. Introducing topological attributes for objective-based visualization of simulated datasets. In *Eurographics/IEEE VGTC Workshop Proceedings of Fourth International Workshop on Volume Graphics*, pages 137–145, Stony Brook, USA, June 2005. Eurographics Association.
- [172] Chi-Keung Tang and Gerard G. Medioni. Robust estimation of curvature information from noisy 3d data for shape description. In *Intl. Conf. on Computer Vision (ICCV'99)*, pages 426–433, 1999.
- [173] G. Taubin. Estimating the tensor of curvature of a surface from a polyhedral approximation. *5th Intl. Conf. on Computer Vision (ICCV'95)*, pages 902–907, June 1995.
- [174] J. B. Tenenbaum. Mapping a manifold of perceptual observations. In *Advances in Neural Information Processing Systems 10*, 1998.
- [175] Holger Theisel, Christian Rossl, Rhaleb Zayer, and Hans-Peter Seidel. Normal based estimation of the curvature tensor for triangular meshes. In *PG '04: Proceedings of the Computer Graphics and Applications, 12th Pacific Conference on (PG'04)*, pages 288–297. IEEE Computer Society, 2004.
- [176] Emanuele Trucco and Robert B. Fisher. Experiments in curvature-based segmentation of range data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(2):177–182, 1995.
- [177] Greg Turk and J. O'Brien. Variational implicit surfaces. Technical Report GIT-GVU-99-15, Georgia Institute of Technology, May 1999.
- [178] S. Ullman. Aligning pictorial descriptions: An approach to object recognition. *Cognition*, 32(3):193–254, 1989.
- [179] R. Veltkamp and M. Hagedoorn. State-of-the-art in shape matching. Technical Report UU-CS-1999-27, Utrecht University, the Netherlands, 1999.
- [180] R. C. Veltkamp and M. Hagedoorn. Shape similarity measures, properties, and constructions. In *Proc. 4th Int. Conf. Adv. Visual Info. Sys., volume 1929 of Lect. Notes in Comp. Sci.*, pages 467–476. Springer, 2000.
- [181] Remco C. Veltkamp. Shape matching: Similarity measures and algorithms. In *SMI '01: Proceedings of the International Conference on Shape Modeling & Applications*, pages 188–199, Washington, DC, USA, 2001. IEEE Computer Society.

- [182] Fabien Vivodtzev, Lars Linsen, Georges-Pierre Bonneau, Bernd Hamann, Kenneth I. Joy, and Bruno A. Olshausen. Hierarchical isosurface segmentation based on discrete curvature. In *Proceedings of VisSym'03- Data Visualization 2003*, pages 249–258, New York, New York, 2003. ACM Press.
- [183] David F. Wiley, Nina Amenta, Dan A. Alcantara, Deboshmita Ghosh, Yong J Kil, Eric Delson, Will Harcourt-Smith, F. James Rohlf, Katherine St. John, and Bernd Hamann. Evolutionary morphing. In *Proceedings of IEEE Visualization 2005*, 2005.
- [184] Richard C. Wilson and Edwin R. Hancock. Consistent topographic surface labelling. *Pattern Recognition*, 32(7):1211–1223, 1999.
- [185] P. Yuen, N. Khalili P., and F. Mokhtarian. Curvature estimation on smoothed 3-d meshes. In *Proc. British Machine Vision Conference*, pages 133–142, 1999.
- [186] Steve Zelinka and Michael Garland. Similarity-based surface modelling using geodesic fans. In *Proc. Second Eurographics Symposium on Geometry Processing*, pages 209–218, 2004.
- [187] C. Zhang and T. Chen. Efficient Feature Extraction for 2D/3D Objects in Mesh Representation. In *Proceedings of the International Conference on Image Processing (ICIP'01)*, volume 3, pages 935–938, 2001.
- [188] Hao Zhang and Eugene Fiume. Shape matching of 3-d contours using normalized fourier descriptors. In *SMI '02: Proceedings of the Shape Modeling International 2002 (SMI'02)*, page 261, Washington, DC, USA, 2002. IEEE Computer Society.

# Vita

Timothy David Gatzke

- Date of Birth** November 2, 1954
- Place of Birth** Cleveland, Ohio
- Degrees** B.S. Cum Laude, Physics, June 1976  
B.A. Mathematics, June 1976  
B.S.Ed. Secondary Education, June 1978  
M.S. Engineering Science, August 1978  
M.S. Aeronautical and Astronautical Engineering, January 1989
- Professional Societies** American Institute for Aeronautics and Astronautics
- Publications** Gatzke, T. D., and Grimm, C. M., "Feature Detection Using Curvature Maps and the Min-Cut/Max-Flow Algorithm" *Geometric Modeling and Processing*, July 2006.
- Gatzke, T. D., and Grimm, C. M., "Estimating Curvature on Triangular Meshes", *Int. Journal of Shape Modeling*, June 2006.
- Gatzke, T. D., Zelinka, S., Grimm, C. M., and Garland, M., "Curvature Maps for Local Shape Comparison," *Shape Modeling International*, June 2005.
- Gatzke, T. D., "Block-Structured Applications," *CRC Handbook of Grid Generation*, Eds. Thompson, Soni, Weatherill, pp. 13-1 to 13-24, 1999.
- Gatzke, T. D., Dowgwillo, R. M., and Ikeda, Y., "The Design of an Aerodynamic Fairing Using Viscous Computational Fluid Dynamics," *AIAA Paper No. 98-2787*, June 1998.

Gatzke, T. D., and Melson, T. G., "Generating Grids Directly On CAD Database Surfaces Using A Parametric Evaluator Approach," NASA CP 3291, May 1995.

LaBozzetta, W. F., Gatzke, T. D., Ellison, S., Finfrock, G. P., and Fisher, M. S., "MACGS - Toward The Complete Grid Generation System," AIAA Paper No. 94-1923, June 1994.

Gatzke, T. D., and Weatherill, N. P., "Unstructured Grid Generation Using Interactive Three-Dimensional Boundary and Efficient Three-Dimensional Volume Methods," AIAA Paper No. 93-3452, August 1993.

Gatzke, T. D., LaBozzetta, Cooley, J., and W. F., Finfrock, "Geometry Acquisition and Grid Generation - Recent Experiences with Complex Aircraft Configurations," NASA CP 3143, pp. 31-43, April 1992.

Gatzke, T. D., LaBozzetta, W. F., Finfrock, G. P., Johnson, J. A., and Romer, W. W., "MACGS: A Zonal Grid Generation System for Complex Aero-Propulsion Configurations," AIAA Paper No. 91-2156, June 1991.

Short Title: Comparing 3-D Object Models

Gatzke, Ph.D. 2007