Recognition of Free-Form 3D Objects in Range Data Using Global and Local Features

DISSEPTION

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Richard J. Campbell, BSEE, MSEE

* * * * *

The Ohio State University

2001

Dissertation Committee:

Dr. Patrick Flynn, Adviser
Dr. Kim Boyer
Dr. Stan Ahalt
Dr. Rick Parent

Approved by

________________________________________
Adviser
Department of Electrical Engineering
ABSTRACT

In this dissertation we will survey the current state of the art in building, representing and recognizing free-form objects from range data. We will also be introduce two novel and effective methods for identifying objects with these sculpted surface types. The techniques are built to exploit discovered shape structure from CAD models of objects to be identified. The first technique uses realistic rendering to generate synthetic data to train the system to identify the objects in real data. This view-based recognition technique obtains a 97% recognition rate for a 10 object database on real data tests, while a 20 object database obtained 99% recognition rate for synthetic image tests. The second object-centered technique builds a hypothesis on object identify and location based on local surface information and their relationships. This second system was able to correctly identify sculpted objects from images where the objects occluded one another from the sensor. Based on only partial information the system was able to recover the objects and their locations in the image.
Recognition of Free-Form 3D Objects in Range Data Using Global and Local Features

By

Richard J. Campbell, Ph.D.

The Ohio State University, 2001

Dr. Patrick Flynn, Adviser

In this dissertation we will survey the current state of the art in building, representing and recognizing free-form objects from range data. We will also be introduce two novel and effective methods for identifying objects with these sculpted surface types. The techniques are built to exploit discovered shape structure from CAD models of objects to be identified. The first technique uses realistic rendering to generate synthetic data to train the system to identify the objects in real data. This view-based recognition technique obtains a 97% recognition rate for a 10 object database on real data tests, while a 20 object database obtained 99% recognition rate for synthetic image tests. The second object-centered technique builds a hypothesis on object identify and location based on local surface information and their relationships. This second system was able to correctly identify sculpted objects from images where the objects occluded one another from the sensor. Based on only partial information the system was able to recover the objects and their locations in the image.
For clear fields, green forests, snow capped peaks, and clear streams that inspire me;

For the God who breathed life into me;

For the Parents that hold me;

and the Friends that comfort me.

I dedicate this my work and my life.
ACKNOWLEDGMENTS

This work has been supported by the National Science Foundation under grants IRI-9209212, IRI-9506414, IIS-9996004, CDA-9422044, and EIA-9818212, by the department of Electrical Engineering at The Ohio State University and by the School of Electrical Engineering and Computer Science at Washington State University.
VITA


1996 .............................. Pre-Doctoral Lecturer. School of Electrical Engineering and Computer Science, Washington State University. Taught a beginning C programming course for non-computer science majors for one semester. I had complete responsibility for this course.

1995 ........................................ M.S. Electrical Engineering.
School of Electrical Engineering and Computer Science, Washington State
University. Advisor: Dr. Patrick J. Flynn. Thesis title: "Range Image
Segmentation Via Presegmentation and Natural Quadric Surface Identifi-
cation".

1993 ........................................ B.S. Electrical Engineering. School
of Electrical Engineering and Computer Science, Washington State Uni-
versity. Summa Cum Laude.

PUBLICATIONS

Research Publications

R. Campbell and P. Flynn, "Model and Range Image Features for Free-Form Object
pp 173-180.

R. Campbell and P. Flynn, "Eigenshapes for 3D Object Recognition in Range Data,"
Proceedings Computer Vision and Pattern Recognition 1999, Fort Collins, CO, June

R. Campbell and J. Krumm, "Object Recognition for an Intelligent Room," Proceed-
ings Computer Vision and Pattern Recognition 2000, Hilton Head Island, SC June

R. Campbell and P. Flynn, "A Survey Of Free-Form Object Representation and
Recognition Techniques," Computer Vision and Image Understanding, in press.

R. Campbell and P. Flynn, "A WWW-Accessible Database for 3D Vision Research,"
Workshop on Empirical Evaluation Methods in Computer Vision, Santa Barbara, CA,
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td></td>
<td>ii</td>
</tr>
<tr>
<td>Dedication</td>
<td></td>
<td>iii</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td></td>
<td>iv</td>
</tr>
<tr>
<td>Vita</td>
<td></td>
<td>v</td>
</tr>
<tr>
<td>List of Tables</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>List of Figures</td>
<td></td>
<td>xii</td>
</tr>
<tr>
<td>1. Introduction</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Range Images</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>1.2 Dictionary of terms</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>1.3 Outline of Dissertation Topics</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>1.4 What is unique about our work</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>1.5 Applications</td>
<td></td>
<td>12</td>
</tr>
<tr>
<td>2. Survey</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>2.2 Free-Form Objects and Surfaces: Definitions, Properties and Issues</td>
<td></td>
<td>14</td>
</tr>
<tr>
<td>2.3 Geometric Descriptions of 3D Objects</td>
<td></td>
<td>18</td>
</tr>
<tr>
<td>2.3.1 Complete Mathematical Forms</td>
<td></td>
<td>18</td>
</tr>
<tr>
<td>2.3.2 Dynamic Objects</td>
<td></td>
<td>27</td>
</tr>
<tr>
<td>2.4 3D Model Construction and Refinement</td>
<td></td>
<td>29</td>
</tr>
<tr>
<td>2.4.1 Registration</td>
<td></td>
<td>31</td>
</tr>
</tbody>
</table>
2.4.2 View Integration .............................................. 45
2.4.3 Mesh Refinement and Optimization ..................... 51
2.5 Free-form Object Recognition Systems .................... 56
  2.5.1 Appearance-Based Recognition ......................... 56
  2.5.2 Recognition from 2D Silhouettes ....................... 65
  2.5.3 Free-Form Object Recognition in Range Data ........ 69
2.6 Emerging Themes, Conclusions, and Commentary .......... 89

3. Object Recognition Using Global Features .................. 93
  3.1 Introduction .................................................. 93
  3.2 Background Material and Notation ......................... 95
  3.3 Scanning and Modeling Environment ....................... 101
    3.3.1 Synthetic Image Generation ............................ 105
  3.4 Training The Database ..................................... 110
  3.5 Object Manifolds .......................................... 114
  3.6 Recognition of Objects in Range Images .................. 119
  3.7 Preliminary Synthetic Model And Image Experiments .... 120
  3.8 View Planning .............................................. 129
    3.8.1 View Planning Results ................................ 132
  3.9 Real Model And Image Experiments ......................... 138
  3.10 Conclusions .............................................. 142

4. Local Feature Object Recognition In The Presence Of Weak Correspondence 150
  4.1 Introduction .................................................. 150
  4.2 Segmentation ................................................ 150
    4.2.1 Curvature .............................................. 152
    4.2.2 Segmentation Technique ................................ 157
  4.3 Hypothesis Generation .................................... 166
  4.4 Verification ................................................. 179
  4.5 Experiments ................................................ 184
  4.6 Conclusion .................................................. 190

5. Range Image and Object Model Database ..................... 199
  5.1 Introduction .................................................. 199
  5.2 Database Contents and Organization ....................... 201
    5.2.1 Object Model Database .................................. 201
    5.2.2 Range Image Database ................................. 203
    5.2.3 Range Image Rendering Software ....................... 204
  5.3 Impact and Lessons Learned ............................... 205

viii
5.4 Access and Citation ................................................. 206
5.5 Conclusions ....................................................... 206

6. Conclusion ............................................................ 209

Appendices:

A. Karhunen-Loève expansion for images ........................................... 213

B. Differential Geometry ...................................................... 215
   B.1 Space Curves ...................................................... 215
   B.2 Surface Curvature ................................................. 218
      B.2.1 First Fundamental Form ...................................... 219
      B.2.2 Second Fundamental Form ................................. 220

Bibliography ............................................................... 224
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Properties of various 3D object representations.</td>
<td>29</td>
</tr>
<tr>
<td>2.2</td>
<td>Summary of appearance-based recognition techniques.</td>
<td>66</td>
</tr>
<tr>
<td>2.3</td>
<td>Summary of recognition techniques for range data.</td>
<td>88</td>
</tr>
<tr>
<td>3.1</td>
<td>Notation used in this chapter.</td>
<td>97</td>
</tr>
<tr>
<td>3.2</td>
<td>Icosahedron Subdivision Parameters. The angle between neighboring viewpoints for the specified subdivision frequency gives a measure of the density of the sampling with respect to the degree of separation between viewpoints.</td>
<td>110</td>
</tr>
<tr>
<td>3.3</td>
<td>Table of parameters used in generation of subspaces for Database A</td>
<td>125</td>
</tr>
<tr>
<td>3.4</td>
<td>Table of parameters used in generation of subspaces for Database B</td>
<td>127</td>
</tr>
<tr>
<td>3.5</td>
<td>View planning statistics and recognition results: AEL is average edge length, ELV is edge length variance, APA is average patch area and PAV is patch area variance.</td>
<td>134</td>
</tr>
<tr>
<td>3.6</td>
<td>The number of vertices, edges and polygons for the models in the 10 object database.</td>
<td>138</td>
</tr>
<tr>
<td>4.1</td>
<td>Classification of surface shape using Shape Index SI quantizer.</td>
<td>168</td>
</tr>
<tr>
<td>4.2</td>
<td>Similarity mapping between range and model segments for shape index quantization value.</td>
<td>169</td>
</tr>
<tr>
<td>4.3</td>
<td>Recognition results for 10 range image test. Recognition requires correct identification of the model and its pose.</td>
<td>185</td>
</tr>
</tbody>
</table>
4.4 Timing result averages for the various stages of the hypothesis generation process. .......................... 187

4.5 Timing for the hypothesis generation and verification for each of the 10 range images test set. .................... 188

B.1 Classifications of surface shape using Mean $H$ and Gaussian $K$ curvature. 222

B.2 Classification of surface shape using $Shape Index SI$. .................... 223
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Range image scanning environment</td>
<td>3</td>
</tr>
<tr>
<td>1.2</td>
<td>A shaded visualizations of rubber duck model showing measurement error resulting from inter-reflection of near-infra laser light in a crevice oriented parallel to the structure light source (on left) and the CAD model of object (on right).</td>
<td>4</td>
</tr>
<tr>
<td>1.3</td>
<td>A shaded visualizations of toy rubber duck and crocodile model showing object self-occlusion measurement errors in the range image (on left) and the CAD model of the toy rubber duck (on right).</td>
<td>6</td>
</tr>
<tr>
<td>2.1</td>
<td>A Curved NURBS patch depicted as a mesh and as a shaded surface. The control polyhedron is visible in both images.</td>
<td>20</td>
</tr>
<tr>
<td>2.2</td>
<td>Implicit model defined by algebraic equation [ f(x, y, z) = 2x^4 - 3x^2y^2 + 3y^4 - 3y^2z^2 - 2x^3z + 6z^4 - 1 = 0. ]</td>
<td>22</td>
</tr>
<tr>
<td>2.3</td>
<td>Deformed Superquadric.</td>
<td>23</td>
</tr>
<tr>
<td>2.4</td>
<td>Generalized Cylinder.</td>
<td>25</td>
</tr>
<tr>
<td>2.5</td>
<td>A coarse polygonal model of the Calcaneus foot bone. Left side is the underlining wire mesh, while the right side shows a Phong shaded visualization of the model.</td>
<td>26</td>
</tr>
<tr>
<td>2.6</td>
<td>Flow diagram showing the various steps in the model building process.</td>
<td>31</td>
</tr>
</tbody>
</table>
2.7 Top: 3D scans and registered color images of a _papier mâché_ brain, taken 90 degrees apart using a Minolta Vivid 700 non-contact 3D digitizer. Bottom: the registered and merged 3D model with one view's texture mapped on to the merged mesh. The images were generated using Minolta's Vivid software package. ........................................ 32

2.8 Iterative Closest Point (ICP) algorithm. ................................. 37

2.9 Two common network topologies used in Multiple View Registration. 40

2.10 A seam resulting from a registration error. .............................. 49

2.11 In the first pair of images the whale mesh contains 6844 vertices. The second set of images shows a the result of reducing the mesh to contain only 3042 vertices. The right image in both pairs is a shaded version of the mesh to the left. The images were generated using Minolta's Vivid software package. ................................. 53

2.12 Example of 4th order operator on range data sampled from a rubber duck. ................................................................. 73

2.13 The splash representation. The angular differences between the normals $N\rho(\theta)$ near the central point and the normal $N$ at the central point are encoded. ......................................................... 75

2.14 Orientation coordinates for splash features. ............................ 76

2.15 Point Signature. .................................................................. 78

2.16 Spin Image. ...................................................................... 80

3.1 Centering an object footprint to minimize the effects of image plane translation. ................................................................. 102

3.2 Segmentation statistics used to remove the effect of translation in $z$. 103

3.3 Using the segment’s bounding box to redefine the object’s sampling density for a given image resolution .......................... 104
3.4 Image and model geometry for a lamb-shaped toy. (a): range image. (b): model. (c): mesh support for a single range image in an area on the nose. (d): mesh support for the model in the same area.

3.5 Summary of the processing steps performed in the generation of the image templates $\mathbf{T}$.

3.6 Discrete Viewsphere and canonicalizing transformation

3.7 Top: an mechanical part and an elephant and their respective manifolds. Bottom: both manifolds are rendered together in the universal eigenspace $\mathcal{U}$

3.8 System Diagram

3.9 Some Examples of Objects From Database A

3.10 Some Examples of Objects From Database B

3.11 Recognition Rates for Database A

3.12 Recognition Rates for Database B

3.13 One Case of Mistaken Identity

3.14 System Diagram (View Planning)

3.15 Two Objects used in some initial tests of view planning

3.16 Manifold Patch area density for each step of view planning

3.17 The density of manifold patch area over viewsphere face area for each step of view planning

3.18 Canonicalizing transformation pose instabilities

3.19 Examples of ‘eigenshapes’

3.20 10 objects used to train and test recognition methods
3.21 Synthetic experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represent a database trained using different positional statistics and canonicalizing transformations. ......................................................... 144

3.22 Real range data experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represent a database trained using different positional statistics and canonicalizing transformations. ......................................................... 145

3.23 Synthetic experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represent a database trained at different template resolutions (image resolution 16,32,64,128). ......................................................... 146

3.24 Real range data experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represent a database trained at different template resolutions (image resolution 16,32,64,128). ......................................................... 146

3.25 Synthetic experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represent a database trained using the different densities of view sampling Table 3.2. .... 147

3.26 Real range data experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represent a database trained using the different densities of view sampling Table 3.2.148

4.1 Cow object model and it's curvature visualizations. (a): Toy Cow. (b): Gaussian Curvature. (c): Mean Curvature. (d): Curvedness. (e): Shape Index. ......................................................... 155

4.2 Cow object model segmentation results. (a): Quickly changing regions of the cow model (radius of curvature \( \leq 10 \text{ mm} \)). (b): Segments. (c): Segment elements. ......................................................... 159

4.3 Segmentation results. (a): Toy Cow object model. (b): Range image segments of both toy cow and apple. ......................................................... 163

4.4 Flow diagram for verification of the hypotheses. ............................................. 181
4.5  Segmentation results for four of the range images used to test the system. (a) Range image of a toy lamb and duck rattle. (b) Range image of toy Po doll and apple. (c) Range image of toy Po doll and whale. (d) Range image of toy lamb and duck rattle. .............. 191

4.6  Recognition results-The range image surfaces are shaded gray, while the recognized models are shaded green and placed in the range image with their recovered pose. (a) Range image of a toy rubber duck and whale. (b) Recognition results for toy rubber duck and whale range image. (c) Range image of a toy red dinosaur and crocodile. (d) Recognition results for toy red dinosaur and crocodile range image. .............. 192

4.7  Recognition results-The range image surfaces are shaded light gray, while the recognized models are shaded dark gray and placed in the range image with their recovered pose. (a) Range image of a toy cow and orange dinosaur. (b) Recognition results for toy cow and orange dinosaur range image. (c) Range image of a toy lamb and duck rattle. (d) Recognition results for toy lamb and duck rattle range image. .............. 193

4.8  Recognition results-The range image surfaces are shaded light gray, while the recognized models are shaded dark gray and placed in the range image with their recovered pose. (a) Range image of a toy Po doll and apple. (b) Recognition results for toy Po doll and apple range image. (c) Range image of a toy Po doll and whale. (d) Recognition results for toy Po doll and whale range image. .............. 194

4.9  Recognition results-The range image surfaces are shaded light gray, while the recognized models are shaded dark gray and placed in the range image with their recovered pose. (a) Range image of a toy rubber duck and crocodile. (b) Recognition results for toy rubber duck and crocodile range image. (c) Range image of a toy cow and apple. (d) Recognition results for toy cow and apple range image. .............. 195

4.10 Recognition results-The range image surfaces are shaded light gray, while the recognized models are shaded dark gray and placed in the range image with their recovered pose. (a) Range image of a toy lamb and duck rattle. (b) Recognition results for toy lamb and duck rattle range image. (c) Range image of a toy red dinosaur and orange dinosaur. (d) Recognition results for toy red dinosaur and orange dinosaur range image. .............. 196
5.1 Synthetic range images of objects from the four databases in the object model database. (a): MSU-Ideas database. (b): USF database. (c): NETLIB database. (d): WSU-PRO/E database. . . . . . . . . . . . . . . . . . . . . . . 207

CHAPTER 1

Introduction

Think for a moment about all the applications pictures and video have been used for. Can you imagine our life without them? Even in antiquity paintings, petroglyphs and stone carvings have been used to archive information and tell stories. Their uncanny ability to capture an instant in time is one of the reasons why they have been such a popular method to entertain and record our collective history. Since the industrial revolution the decrease in cost in obtaining and recording pictures have made this ability available to most people, not just artisans and historians.

One potential source for the continued growth in the field of image understanding is the applicability of photography for battlefield analysis in the great world wars of the 20th century. Here experts would pour over photographs taken from reconnaissance planes to monitor the opposing side’s military and manufacturing capability. Initially the analysis of the aerial photography could be accomplished by human experts, but as the world moved into the Cold War more and more surface area of the world was being monitored this caused an explosion in the amount of images. To handle this increased demand, the field of Image Understanding nourished. This field desired to develop automated methods to prescreen the aerial photographs to detect or highlight areas of interest that a human operator should examine. This could
be a new construction site on or near military installations or an expansion to the companies making military hardware.

Other areas and applications that fueled the growth in image understanding are: improved methods for detecting and diagnosing disease and ailments, accurate mappings of the earth surface, more accurate predictions of the weather, improved quality control in manufacturing facilities, and better maps and charts of the planets and galaxies.

Some of these accomplishments utilized and captured electromagnetic radiation outside the frequencies is visible to the naked eye. These new and novel sensing methods open up our ability to peer into place previously unknown to us and continue to allow us to look at life and matter in new ways.

1.1 Range Images

One such new way to capture information about the world around us is the ability to directly measure the 3D geometry (shape) of objects in a sensor’s field of view. Utilizing various sensor technologies measurements can be taken to determine how far a point on a surface is away from a sensor. A sequence of such range (or distance) measurements then can be used (or re-sampled) to form an image where the pixels values represent distance and the row and column indices indicate position in the scene. Using knowledge about the sensor, scene and/or the measurement plan, the true 3D position for each point represented by a pixel in the image can be computed. For ease of use most modern sensors will return this estimate. These range images provide a discrete sampling of the surfaces in the scene. In particular, dense sampling of the surfaces gives the practitioner a good approximation to surface properties
(position, normal, and curvature). This information can be used to reverse engineer a model for objects of interest. One possible application for these sensors is to provide models for realistic visual effects in the entertainment industry.

Let a range image be denoted by

\[
R = \left\{ \mathbf{r}(i,j) = \begin{pmatrix} x(i,j) \\ y(i,j) \\ z(i,j) \end{pmatrix} : i = 1, \ldots, M, j = 1, \ldots, N \right\}
\]

---

**Figure 1.1: Range image scanning environment**

---

An image of 3D surface measurements in the sensors coordinate system, where \(i, j\) denotes image raster position and \(x, y, z\) represent the three components of the 3D measurement in a predefined coordinate frame.
Figure 1.2: A shaded visualization of rubber duck model showing measurement error resulting from inter-reflection of near-infra laser light in a crevice oriented parallel to the structure light source (on left) and the CAD model of object (on right).

In this thesis the images will be captured using Minolta’s Vivid 700 non-contact 3D digitizer [89]. Minolta’s sensor utilizes a motorized mirror to scan a near infrared laser line across the object. This structured light source and triangulation geometry are used to recover a dense range image. Figure 1.1 gives the basic scanner setup where $D_{sep}$ is the distance between the camera and the structured light source and $D_{obj}$ is the distance between the camera and the center of the turntable. The field of view of the CCD camera can be changed using a zoom lens attached to the sensor. By narrowing the field of view the number of range samples in a given area of space is increased. This allows the sensor and turntable distance $D_{obj}$ to remain fixed while still obtaining dense range samples for even the smallest objects in our database.
The range images that are produced using this sensor may to contain two types of structured measurement errors.

- The first type is shown in Figure 1.2 in the left image. In this range image the measurement of the surfaces in the neck crease on the toy rubber duck are incorrectly lifted toward the camera. This measurement error is given the name blooming for its lifting of the surfaces. The error is produced from errors in recovering the object’s geometry when the structured infra-red laser source is scattered via inter-reflections between the surfaces in the crease.

- The second type is shown in Figure 1.3 in the left image. In this range image the measurement of some of the object’s surfaces are missing because of object self-shadowing [44] (Figure 1.1). For triangulation sensors (like the Minolta) the locations on the object that are occluded from structured light source or to the CCD sensor cannot be measured. To measure these missing surface locations the object viewpoint has to be changed so that both the structured light source and the CCD array can receive light reflected from this point.

1.2 Dictionary of terms

**Illumination image:** An image that was generated by capturing electromagnetic energy in the visual spectrum (for humans). Unless referred to otherwise the illumination image only contains information about light intensity. These images are often referred to as grayscale images or black and white images.

**Range image:** An image whose pixels contain information about the distances between camera and surfaces of the object in-front of the camera.
Figure 1.3: A shaded visualizations of toy rubber duck and crocodile model showing object self-occlusion measurement errors in the range image (on left) and the CAD model of the toy rubber duck (on right).

Salient: noticeable or striking.

Saliency: A measure of how noticeable or striking.

Occlusion: When part of the object is not visible due to an object blocking the detectable electromagnetic energy from reaching sensor.

Self-Occlusion: When part of an object is not visible because its shape blocks the detectable electromagnetic energy from some of the object’s surfaces from reaching the sensor.

Mathematically Complete Object Model: The model is described using an explicit geometric description, the entirety of a surface or object is described, and hence that synthetic images of the object in an arbitrary pose can be generated when the geometry description is coupled with a view specification.
**Discriminatory Object Model:** The object representation is descriptive but not necessarily complete. These models are typically used for specialized applications to summarize and compare features for specific tasks (i.e. object recognition) in a compact form that is more computationally efficient than complete forms.

**Feature:** A distinctive part or location on an object. In computer vision a feature is often a derived from a series of operations to identify or bring out distinctive descriptions of regions in images. These features help the system to understand and extract information contained in the images.

**Object Centered Representation:** Used to describe a discriminatory object model/representation that is built upon a known fixed and recoverable object coordinate system. This system can be recovered from any view of the object.

**View Centered Representation:** Used to describe a discriminatory object model/representation that is built upon information gathered from observations taken from various viewpoints. The distinction between is representation and object centered is that view centered representations are depended the camera’s coordinate system not the object’s.

**Global Feature:** A distinctive part or measure based on all information visible in a view of an object.

**Local Feature:** A distinctive part or measure based only on a local region of support for an image.

**Rendering:** Graphics term used to indicate the generation of a synthetic image from and model of a scene.

**Z-buffer:** A two dimensional array used in rendering to determine what surfaces are visible to the camera.
**View-point:** Used to describe the rotational pose of an object before rendering. It is derived from surrounding the object by a sphere where the center of the object and sphere coincide. The points on the spheres surface indicate possible viewing directions of the object. These points are given the name view-points. Note: the viewing direction coincides with the optical axis of the camera capturing the image or scene.

**Curvature:** For curves it is the rate of change of the tangent to the curve w.r.t arc length of curve. In the limit this rate of change converges to the inverse of the radius of a circle that matches the curves shape in the little differential region. For more information about curvature and its extension for surfaces please see Appendix B.

**Principal Curvature** ($\kappa_1, \kappa_2$): Refers to the two extreme values of normal surface curvature. Normal curvature has some special properties that are highlighted in Appendix B.

**Gaussian Curvature:** The product of the principal curvature ($K = \kappa_1 \kappa_2$) Appendix B.

**Mean Curvature:** The mean of the principal curvature ($H = \frac{\kappa_1 + \kappa_2}{2}$ Appendix B.

**Free-Form Surface:** Besl [13]: “a free-form surface has a well defined surface normal that is continuous almost everywhere except at vertices, edges and cusps”.

**Free-Form Object:** An object that needs to be described with one or more free-form surfaces.

**KLT Karhunen-Loève expansion or transformation:** Used in image compression and vision to optimally (Mean Squared Error sense) describe a set of data
with the fewest number of coefficients Appendix A. Also provides an inherent significance of each coefficients in the reconstruction of the original data. This allows for intelligent truncation to remove coefficients that are not as important to accurately represent original data with fewest number of coefficients.

**Appearance:** Used to describe how an object looks to the sensor (i.e. sensed values of the pixels in the captured image). Appearance can depend on the lighting environment and pose of the object.

**Appearance-based recognition:** A view-based recognition system built upon a set of training images containing the appearance of the object at various view-points Section 2.5.1.

### 1.3 Outline of Dissertation Topics

This dissertation contains an extensive review of current representation, building and recognition strategies (Chapter 2). The material is presented first to introduce representation methods for the complex free-flowing surfaces, then the reader is lead through the issues surrounding the building of free-form objects with range imagery. These issues provide a good framework for understanding complexity in reverse engineering object models and familiarize the reader with specifications made to construct the CAD models that will be used to train our recognition methods.

The survey also introduces the reader to appearance-based recognition techniques. These methods have gained in popularity for object recognition in intensity imagery and have shown some ability to recognize free-form objects. This portion of the survey is also important in introducing the reader to material that our global feature recognition technique in Chapter 3 is based on. Finally the survey covers the current
state of the art in free-form object recognition from range data. These shape based methods try to capture some inherent characteristic about the objects and their relative merit are compared.

After the survey the dissertation begins to cover research the author generated throughout his time as a Ph.D. candidate at Washington and Ohio State Universities. The first research chapter covers research on the recognition of free-form objects using a view-based recognition technique (Chapter 3. Here the object recognition system is trained on synthetic training range images of the objects in our database. The training data is examined for an inherent set of characteristic basis functions (images) that contain the relevant information in differentiating the objects from one another. Presented in the chapter are the practical issues in getting the synthetically trained recognition system to work on range images of the real objects using the Minolta sensor [89].

In Chapter 4 a local feature based technique is demonstrated to find the same set of free-form objects in range images that can contain more than one object and in cases where the objects can occlude portions of the object from being viewed. These scanning conditions require that the recognition strategy make decisions and build up an hypothesis for only partial information about an object. Presented in this chapter is a unique method that utilizes curvature based segmentation to identify homogeneous regions of high curvature. These high curved regions are then used to recover object identity and location.

Chapter 5 contains a discussion about building and maintaining a database of range images and 3D models. These databases are important for comparison of various recognition strategies and for the vary basic reason that range sensors can be
expensive and time consuming to use. Finally a conclusion about the authors current work and proposed directions for future work (Chapter 6).

1.4 What is unique about our work

The contributions of this dissertation is first and foremost as a valuable resource to the author and others in the area of free-form object recognition. This task is accomplished first by the extensive survey contain in Chapter 2 and then in the commentary contained in the other various chapters and sections.

The next primary contribution of this dissertation is two new recognition strategies for identifying object’s with sculpted surfaces. The first strategy is built upon a popular intensity image technique and applied to a new data type as well as object database (Chapter 3). The commentary on this technique highlights the problems in applying the recognition system to a new data type and also shows the applicability of using CAD models and realistic rendering techniques to train the recognition system. This make practical for the system to operate in environments where object pose is uncertain.

The second strategy builds upon a unique feature extraction technique that exploits the structure of the high curved regions for sculpted objects. The use of surface curvature for segmentation is used in a novel way in conjunction with the definition and construction of a segment element. These basic elements are then used in a hypothesize-and-test architecture that exploits the structure of the high curved regions to limit the search required to generate a likely object hypothesis. The system has shown effective in recognizing objects even in the presence of occlusion and structured measurement errors.
1.5 Applications

Potential applications for the work contained in this dissertation include manufacturing object inspection, automated manufacturing of object’s containing sculpted surfaces and CAD/CAM database object indexing for database searching for arbitrary shaped models. All these potential applications mentioned have been in the general area of industrial manufacturing. The general flavor of our work is geared toward this area. In our work we always worked from CAD models of objects, with the aim being in a modern design and manufacturing environment the CAD models of the objects to be manufactured and inspected are already available. In this field the ability to automate the design, manufacturing, and inspection process is vary important in-terms of cost to produce the object in consideration. Imagine if you will the case where in a modern environment with unified CAD model databases are used for the design, manufacturing and inspection process. In this case if a design change needs to be made on a product in the late stages of the product cycle the cost for implementing this change has a potential to be minimize because the change is reflected in all three stages of process for producing the final product.

Our research is aimed at improving a manufactures ability to automate the design and implementation of the inspection process. During inspection a flexible manufacturing environment needs to be able to recognize the object and its pose so the manufactured model’s can be inspect to see if it is with in the acceptable tolerance specification. Using the CAD model to design and implement a recognition and pose recovery algorithm is one step to automating the design of the inspection process.
CHAPTER 2

Survey

2.1 Introduction

Computer models of *free-form* objects are a key element of many interesting applications involving graphics and visualization (*e.g.*, virtual world design, environment modeling for semi-autonomous navigation [70], *etc*.). In particular, techniques for the automatic recognition of 3D objects have become increasingly sophisticated and the use of free-form object models in such systems is now seeing focused attention from researchers in the computer vision community. Moreover, recent years have seen work in the graphics and vision communities intersect in the area of free-form object modeling. Graphics practitioners, faced with the labor-intensive process of redesigning complex objects on CAD systems, would prefer to use vision techniques to assemble multiple views of a complex object together into a model that can be fine-tuned and completed interactively. Vision practitioners, faced with the limited scope of simple geometric object representations, see free-form representations as a *lingua franca*, a universal language, for future object recognition systems.

1This chapter is based on the paper "A Survey Of Free-Form Object Representation and Recognition Techniques" by Campbell and Flynn to appear in CVIU in 2001.
In this section we will survey the recent work done to represent and recognize free-form objects. Included in the discussion about representation is a definition of free-form objects (Section 2.2), coverage of methods used to represent object geometry (Section 2.3), and a description of some current techniques proposed in the computer vision and graphics literature to develop a model of the object from 3D (range) image data (Section 2.4). Specific techniques covered include image data registration (Section 2.4.1), integration (Section 2.4.2), and model optimization (Section 2.4.3). The second major portion of the survey addresses techniques used to identify free-form objects present in data gathered from intensity or range image scanners (Section 2.5). This includes appearance-based (Section 2.5.1), intensity contour based (Section 2.5.2) and surface feature based techniques (Section 2.5.3). The survey concludes with some speculation on areas ripe for further research (Section 2.6).

2.2 Free-Form Objects and Surfaces: Definitions, Properties and Issues

Definitions of free-form surfaces and objects are often intuitive rather than formal. Synonymous adjectives include ‘sculpted’, ‘free-flowing’, ‘piecewise-smooth’, or ‘piecewise-$C^n$’ for some desired degree of continuity $n$. Often, ‘free-form’ is a general characterization of an object whose surfaces are not of a more easily recognized class such as planar and/or natural quadric surfaces. Hence, a free-form object is often assumed to be composed of one or more non-planar, non-quadric surfaces (‘free-form surface’). A roughly equivalent characterization was provided by Besl [13]: “a free-form surface has a well defined surface normal that is continuous almost everywhere except at vertices, edges and cusps”. Dorai and Jain [36], Besl [13], and Stein and Medioni [131] cite sculptures, car bodies, ship hulls, airplanes, human faces, organs,
and terrain maps as being typical examples of free-form objects. Specifically excluded from this class of object by these authors are statistically defined shapes like textures and foams, infinitely detailed objects possessing self-similarity that are best described using fractal models, and non-orientable surfaces such as Möbius strips and Klein bottles.

Despite the different application contexts of free-form object models in vision and graphics, some criteria apply (or should apply) to representations regardless of domain. In an early survey on object representation, Brown [19] lists ambiguity, conciseness, and uniqueness as some mathematical properties of object representation. *Ambiguity* measures the representation’s ability to completely define the object in the model space; this is sometimes referred to as *completeness* of the model description in the computer vision literature. *Conciseness* represents how efficiently (compactly) the description defines the object. Finally, *uniqueness* is used to measure if there is more than one way to represent the same object given the construction methods of the representation. If the representation is unambiguous and unique then there is a one-to-one mapping from the object to the representation.

The importance of these mathematical properties to the object representation strategy depends on the application context. In the case of object recognition applications, completeness and compactness are often sacrificed in favor of efficiency [42]. The pragmatic issue of performance often makes such compromises appropriate. This highlights the application dependence on the use of complete vs. discriminatory models. Discriminatory models are most often used in object recognition because they can be designed to capture the details that differentiate objects from one another.
efficiently. These representations are designed to be efficient for the task of matching, and not as a complete description of an object’s geometry.

By contrast, some computer graphics applications require complete models to accurately render a realistic synthetic image of the objects described. When rendering, the key attributes relate to realism (visual fidelity). One common method to improve realism of 3D rendering is to increase polygon density and the quality of the accompany texture maps for the models used in rendering.

In a computer vision context, the object model is designed for use in the specific vision application (such as navigation, recognition, or tracking), and visual fidelity may not be a criterion of interest. Efficient execution of the application is almost always of interest, and this may favor multiple redundant representations. In object recognition, saliency is an important feature of object representations; the qualities (geometric or otherwise) that allow objects to be discriminated from one another must be easily obtained. It is tempting to assume that salient features are generic, i.e. that saliency is captured by the cardinality or parameters of a particular pre-defined geometric or photometric feature. As the mix of models changes in a dynamic database, the saliency of some features and models will change requiring new features to be found or more advanced techniques to be developed that deal with the lack of uniqueness.

The locality of an object representation may be of interest in applications of recognition in the presence of occlusion. A representation that explicitly reveals local geometrical structure may be characterized as “occlusion tolerant”, hence better suited to such applications. Unfortunately, representations that are chiefly local are generally verbose. This further motivates the use of multiple representations in applications
where requirements conflict; this adds a burden, namely the need to maintain consistency between representations.

With the foregoing definitions of free-form surfaces and objects as background, some object recognition systems working with intensity imagery have employed (or assumed) such surfaces and objects for some time. Indeed, some of these systems make no explicit assumption about a class of allowable object shapes. In particular, image-based recognition systems (that employ no specific object model representation; appearance-based recognition is an example) process images of free-form objects just as they would process an image of a geometrically simpler object. Recognition decisions in such systems are based on the distribution of intensities in prototypical views of the object rather than on an object-centered representation \textit{per se}.

The history of model-based 3D object recognition techniques, by contrast, shows a progression in the complexity of object models employed, from simple polyhedra to natural quadrics to various free-form-like object representations such as superquadrics. The choice of a representation must be accompanied by robust techniques for extracting compatible features from both object models and input images, so that recognition can be performed. The potential lack of simple features like crease edges or linear silhouette edges makes this feature identification task difficult. Nonetheless, a variety of creative approaches to this problem have been proposed and implemented, and a goal of this survey is to highlight many of these techniques.
2.3 Geometric Descriptions of 3D Objects

This section provides an overview of the popular techniques for representing the geometry (i.e., the 3D shape) of 3D surfaces and objects, with a focus on those representations that are general enough to be labeled ‘free-form’.

2.3.1 Complete Mathematical Forms

The representations discussed here are complete in that the geometric description is explicit, the entirety of a surface or object is described, and hence that synthetic images of the object in an arbitrary pose can be generated when the geometry description is coupled with a view specification.

Parametric Forms

Parametric surfaces are widely used in computer-aided design and manufacturing (CADM) and other object modeling systems for the following reasons:

1. They are mathematically complete.
2. They are easily sampled.
3. They facilitate design (objects can be designed in terms of patches whose continuity can be controlled at the joins).
4. Their representational power is strong (they can be used to represent complex object geometries).
5. They can be used to generate realistic views.
6. Methods for generating parametric representations for data points taken from existing objects have been developed [35, 41].
A generic parametric form for a 3D surface (a 2D manifold embedded in 3D) is

\[
S(u, v) = \begin{bmatrix}
  x = f(u, v) \\
  y = g(u, v) \\
  z = h(u, v)
\end{bmatrix}.
\]

The three functions \( f(u, v), \ g(u, v), \) and \( h(u, v) \) have as arguments the two parametric variables \( (u, v) \). Without loss of generality the domain of \( (u, v) \) can be restricted to be the unit square \([0, 1] \times [0, 1]\).

The most common parametric formulation is the Non-Uniform Rational B-Spline (NURBS) Figure 2.1, which are defined by the specialized parametric form

\[
S(u, v) = \sum_i \sum_j B^h_{i,j} N_{i,k}(u) M_{i,l}(v),
\]

where \( N_{i,k}(u) \) and \( M_{i,l}(v) \) are the B-spline basis functions of order \( k \) and \( l \), and \( B^h_{i,j} \) are the homogeneous coordinates of the control points, respectively [82]. Hence, NURBS are a tensor-product surface form. It has been shown that natural quadrics (such as spheres, cylinders, and cones) admit exact representation as NURBS; the available homogeneous coordinate makes this representation quite flexible [41]. Figure 2.1 shows a curved NURBS patch as a parametric mesh and a shaded surface, including the positions of vertices in its control point mesh.

This common descriptive form has been considered for use in computer vision systems, but Besl [13] explains why parametric representations are not widely used. In particular, it is difficult to make a surface defined on a parametric rectangle fit an arbitrary region on the surface of an object; this necessitates the use of trimming curves, which are not always unique and not generally detectable in imagery. Moreover, the homogeneous control points are also not easily detectable, nor unique. The completeness of parametric forms makes them useful as a source of an initial object
Figure 2.1: A Curved NURBS patch depicted as a mesh and as a shaded surface. The control polyhedron is visible in both images.

specification, from which a polygonal mesh or other representations can be generated and employed in a vision system.

**Algebraic Implicit Surfaces**

A surface can be defined implicitly as the zero-set of an arbitrary function $f$:

$$\mathcal{S} = \{(x, y, z) | f(x, y, z) = 0\}$$

Figure 2.2 depicts an implicit quartic surface. If $f$ is polynomial and a set of samples from a single surface of this form is available, well-established fitting procedures can be used to estimate the polynomial coefficients [135]. These coefficients are not generally invariant to pose transformations, although mathematical invariants can be obtained from some forms [132]. Recent work in this area has included
1. finding the exact distance between an implicit surface and 3D points sampled from it [133],

2. using bounded versions of algebraic curves and surfaces to control the fitting process [136], and

3. preserving a known topology in the fitted surface [133] (i.e. if the data is assumed to be from a surface of genus zero then the fitted surface will be of genus zero).

Surface fitting techniques have been used to segment image scenes [135] and have also been used as a preprocessing step in the recognition of objects [132].

The class of objects that can be described by a single algebraic surface is limited by the stability of the fitting process. In practice, fitting a surface to an order greater than four can produce surfaces whose shape matches poorly to the object from which the data was obtained. In order to model more complex objects, patches of implicit surfaces must often be used [5, 133].

**Superquadrics**

A superquadric \( S(\eta, \omega) \) is a volumetric primitive [7] [125] with the following simple implicit and parametric forms:

\[
S(\eta, \omega) = \begin{bmatrix} x(\eta, \omega) \\ y(\eta, \omega) \\ z(\eta, \omega) \end{bmatrix} = \begin{bmatrix} a_1 \cos^{s_1}(\eta) \cos^{s_2}(\omega) \\ a_2 \cos^{s_1}(\eta) \sin^{s_2}(\omega) \\ a_3 \sin^{s_1}(\eta) \end{bmatrix}, \quad -\frac{\pi}{2} \leq \eta \leq \frac{\pi}{2}, -\pi \leq \omega \leq \pi,
\]

\[
S(x, y, z) = \left[ \left( \frac{x}{a_1} \right)^{\frac{2}{s_1}} + \left( \frac{y}{a_2} \right)^{\frac{2}{s_2}} \right]^{\frac{s_2}{s_1}} + \left( \frac{z}{a_3} \right)^{\frac{2}{s_1}} = 0.
\]

Pentland [110] and Solina and Bajcsy [125] showed that with the addition of tapering, twisting, and bending deformations to the superquadric models, a variety
of ‘free-form’ objects can be modeled (See Figure 2.3.1 for an example). Such models capture only with great difficulty the fine detail that differentiates similar objects like human faces from one another. The ability of the superquadric to model the coarse shape of an object has been used to determine geometric classes (or Geons) [115, 40].

Any object recognition system employing superquadric models as a primary representation must solve two key problems in its design. First, a mechanism for segmenting the input image data (typically range data) into patches, each of which lies within a single superquadric primitive, must be developed. Secondly, a robust technique for fitting superquadric models to the sensed data must be available. Gupta
Figure 2.3: Deformed Superquadric.

The superquadric \((a_1 = a_2 = a_3 = 1 \text{ and } \epsilon_1 = 1.0, \epsilon_2 = 0.3)\) is tapered along the \(z\) axis \[
\begin{bmatrix}
x_t(\eta, \omega) \\
y_t(\eta, \omega)
\end{bmatrix} = \left(\frac{z(\eta, \omega) + 1}{2}\right) \begin{bmatrix}
x(\eta, \omega) \\
y(\eta, \omega)
\end{bmatrix}
\]
then twisted about the \(z\) axis using \(\theta = \pi(1 - z(\eta, \omega))\), \[
\begin{bmatrix}
x_t(\eta, \omega) \\
y_t(\eta, \omega)
\end{bmatrix} = \begin{bmatrix}
\cos(\theta) & -\sin(\theta) \\
\sin(\theta) & \cos(\theta)
\end{bmatrix} \begin{bmatrix}
x(\eta, \omega) \\
y(\eta, \omega)
\end{bmatrix}.
\]
Pentland [110], Solina and Bajcsy [125], Raja and Jain [115], and Dickinson et al. [40] have all addressed facets of this problem.

Superquadrics have been used to recognize a large class of objects and have even been used to build composite parts (unions of several superquadrics), but they lack the descriptive ability to effectively capture local surface shape changes on free-form objects. Ayong-Chee et al. [4] demonstrated a hybrid approach to recognition using the parameters of a superquadric to define object shape classes. These classes are used to remove possible object matches based on their gross shape. Then more computationally expensive techniques can be used to match the fine details of objects within a class.

**Generalized Cylinders**

Generalized cylinders are defined by a space curve \( A(s) \) (representing the axis of the primitive) and a cross-section contour \( C(s, \theta) \) defined in the plane normal to the axis at \( s \) and defining the boundary of the primitive along the axis[1] [104]. Figure 2.4 shows an example of a free-form object represented by a generalized cylinder. To construct more complex objects (e.g., hierarchies such as animals), multiple generalized cylinders are used to represent their individual parts.

Generalized cylinders are particularly attractive for representing elongated shapes where an axis is easy to define. In this case the axis of the primitive often provides an intuitive method to conceptualize the design of a object and a method of reliably recovering useful statistics about the shape of the object.

On the other hand, some shapes are not easily described using generalized cylinders. In such cases, it may be difficult (or impossible) to define an axis whose cross-sections contain only one closed contour. It is possible to extend the representation
to handle these and other cases that appear, but it may not prove to be the most useful representation of the object with respect to design and discrimination.

**Polygonal Meshes**

A popular representation for 3D objects is the polygonal mesh. A shared vertex-list notation is common for such representations. Accordingly, an object is defined by a pair of ordered lists:

\[ \mathcal{O} = \langle \mathcal{P}, \mathcal{V} \rangle, \]

where \( \mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_{N_v}\} \) is a list of \( N_v \) three-dimensional vertices \( \mathbf{v}_i = (x_i, y_i, z_i)^T \), and \( \mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_{N_p}\} \) is a list of polygons, each specified as a list of vertex indices: \( \mathbf{p}_i = \{v_{i,1}, \ldots, v_{i,ne_i}\} \).
If \( nv_i = 3 \) for all \( i \), the mesh consists strictly of triangles. The guaranteed convexity of triangles allows simpler rendering algorithms to be used for the generation of synthetic images of models [48]. A variety of techniques (commonly called polygonization methods) exist for generating polygonal mesh approximations from other geometric primitives (such as implicit surfaces[105], parametric surfaces [83], and isosurfaces in volume data [88]).

![Image](image_url)

Figure 2.5: A coarse polygonal model of the Calcaneus foot bone. Left side is the underlining wire mesh, while the right side shows a Phong shaded visualization of the model.

Polygonal meshes have a long history in computer graphics, but have also become increasingly popular as an object representation in computer vision. This increase in popularity is due to several factors including advances in computer storage capacity and processing power and a modest increase in the popularity of dense range sensors, which produce rectangular arrays of 3D points that can easily be triangulated into meshes. Meshes can faithfully approximate complex free-form objects to any desired accuracy given sufficient space to store the representation. Figure 2.5 shows
(in wireframe and shaded renderings) a polygonal mesh representing a human foot bone. With the decreasing cost of computer memory even very dense meshes are becoming practical. The expanding market for virtual environments in entertainment and geometric design has also played a role in promoting the mesh as a universal language for object representation.

Polygonal meshes have limitations. While they are a faithful representation, they are also approximate and scale-dependent. Any higher-level surface characterization must be explicitly maintained along with the mesh. The required resolution (density) of the mesh may vary between (or within) applications. A strong sub-area of research in computer vision and graphics is the study of approaches to coarsen dense meshes, or refine coarse meshes, in response to application requirements. This topic, as well as a variety of approaches to the construction of models from multiple views, is discussed in Section 2.4.

2.3.2 Dynamic Objects

Machine vision and/or object modeling systems must sometimes accommodate objects with time-varying shape. In this survey we mention only a few relevant papers on nonrigid shape, limiting the discussion to objects with articulated or deformable shapes. A comprehensive survey of either of these areas is outside the scope of this paper, but would be a valuable service to the community.

Articulated Objects

In many cases dynamic objects can be decomposed into a set of rigid parts connected together by joints. This mechanical perspective on both machines and biological forms can closely approximate a large number of dynamic objects. For biological
forms whose support structure (bone, cartridge, exoskeleton) is rigid, the primary contributor to change in the shape and appearance of the object is the relative motion at the joints between parts of the form (body, limbs or appendages).

Articulated objects are typically studied by identifying the rigid parts and parametrizing the relationships between them. As mentioned in the previous sections, superquadric models and generalized cylinders have been used to recover and identify parts of articulated objects. Another reason to study articulated objects is to determine the purpose (function) of the assembly of parts [54]. In the computer vision literature, articulated objects have been studied in a modeling context by Ashbrook et al. [3]. Sallam and Bowyer [119] investigated the extension of the aspect graph representation to articulated assemblies.

**Deformable Objects**

Other animals, plants and machines move in a fluid free-flowing way. Examples of non-rigid motion of object are: a swimming jelly fish, wheat stalks blowing in the wind, and a beating heart. These objects surfaces deform as they move and interact with their environment.

To understand and model the changes in a deformable object’s surfaces the properties of object materials become more important. One method of doing this is to utilize Finite Element Models (FEM) to discretize the object into small connected elements [122] whose properties and environmental pressures can be used to predict shape. Another aspect of deformable modeling is to quantify regular motions like those of a beating heart [109]. Analyzing and comparing these regular motions is an important tool for diagnosing and treating defects in heart rhythm and pumping
<table>
<thead>
<tr>
<th>Representation</th>
<th>Global</th>
<th>Compact</th>
<th>Local Control</th>
<th>Complete</th>
<th>Easily sampled</th>
<th>Easily fit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parametric</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Implicit</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Superquadric</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Gen. Cylinder</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Mesh</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

Table 2.1: Properties of various 3D object representations.

action. Delingette et al. [32] studied deformable object shape modeling and related work is addressed in Section 2.5.3.

Table 2.1 summarizes some important dimensions of 3D modeling technique as discussed above. Each strategy is characterized in terms of the accessibility of local shape information, compactness, controllability, completeness, and ease of sampling, and ease of construction from sampled data.

### 2.4 3D Model Construction and Refinement

The history of computer-aided 3D design has, until recently, focused on CAD-like approaches to object design. A design engineer would work from a product concept or a physical prototype, and encode its geometry using the particular representation tools available in the software. If the geometry of the object did not match the modeling primitives available, cumbersome workarounds and approximations were needed. Recently, a number of experimental and commercial systems composed of versatile range sensors and software have aimed to automatically produce geometric models from physical prototypes of objects. Figure 2.4 depicts range/texture views and their integration within a particular product of this sort. Depending on the
sensor, simple and small mechanical parts, complex assemblies, human bodies, and even environments (e.g., a hallway with doors, obstacles, pipes along the ceiling, etc.) can be scanned and represented. Reverse Engineering is a popular term for this type of technique.

As noted above, the polyhedral mesh representation has recently become a popular representation for 3D objects. The meshes can be produced through direct polygonization of structured data (e.g., range imagery), polygonization from scattered 3D points, and inflation, deflation, or deformation of an initial spherical mesh to approximate a 3D point set. The polygonization is often simplified to triangles in order to speed up the visualization of reconstructed model and reduce the complexity of the processing algorithms.

The process of reverse engineering the geometry of a 3D model for a part is typically decomposed into the following two or three subtasks, as depicted in Figure 2.6:

- The first subtask is the registration of all available sets of 3D data to a common (object) coordinate system. This step may be accomplished automatically or semi-automatically.

- The second subtask is the integration of the data to form a single 3D mesh. By ‘integration’, we refer to the merging of independent (but registered) meshes to form a single polygonal mesh without defects such as dangling edges or topology violations.

- The third (optional) task is to optimize the mesh representation for a specific application or use. This often results in a coarsening of the mesh (i.e., reduction of the vertex and polygon count) in response to application demands or other
constraints. This also could mean changing the object representation to one that is more suited for an application. An example of this is the inclusion of texture map data with the polygonal representation to provide a much more realistic rendering of the object even when the application requires the use of a coarse polygonal mesh to achieve interactivity.

![Flow diagram showing the various steps in the model building process.](image)

**Figure 2.6**: Flow diagram showing the various steps in the model building process.

### 2.4.1 Registration

Most 3D scanning methods can only capture part of the object at a time. This could be due to limitations in the physical design and/or the technology the scanner is
Figure 2.7: Top: 3D scans and registered color images of a *papier mâché* brain, taken 90 degrees apart using a Minolta Vivid 700 non-contact 3D digitizer. Bottom: the registered and merged 3D model with one view’s texture mapped on to the merged mesh. The images were generated using Minolta’s Vivid software package.
built upon. An example is a structured laser light scanner that works on the principle of triangulation to recover position of surface points on the object. Using this scanner technology parts of an object’s surface may not be recoverable because the object itself shadows the structured light from the detector in the sensor. Figure 2.4 shows an example of data taken from a Minolta Vivid 700 structured light range sensor from Minolta. In the first four sub-figures the result of self shadowing produces holes in the mesh produced by the scanner.

The construction of a complete object model will require multiple object scans. Since each scan is generally represented in an independent sensor coordinate system, the scan data must be transformed from several different sensor coordinate systems to a single object-centered coordinate system. This step is typically termed registration and has received sustained attention in the research community over the past ten years [20]. During this time a distinction has been drawn between coarse registration (wherein the set of interframe registrations are determined to within a few degrees of rotation and a similarly small translational tolerance) and fine registration wherein the transformation parameters are fine-tuned. This distinction has been drawn because different techniques are often employed in these two phases.

Most commercial products for model construction employ a rotating turntable upon which the object is placed for sensing. Such systems do not need to solve a coarse registration problem (although pose refinement is still performed to compensate for mechanical variability in the turntable mechanism and calibration errors).

In the model building process a sequence of views \( \{V_1, \ldots, V_m\} \) is taken of an object \( O \) at different viewpoints. The data for each view lies in a local coordinate

\(^2\text{http://www.minolta3D.com/}\)
frame. The goal is a set of transformations \( \{T^0_i(\tilde{p}) \ldots T^0_m(\tilde{p})\} \) that will transform the data from the views to a common coordinate frame and minimize an error function.

Let

\[
T^j_i(\tilde{p}) = R\tilde{p} + \tilde{d}
\]

denote a linear transformation of a point \( \tilde{p} \) in coordinate frame \( i \) to coordinate frame \( j \), where

\[
R = \begin{pmatrix}
a_{1,1} & a_{1,2} & a_{1,3} \\
a_{2,1} & a_{2,2} & a_{2,3} \\
a_{3,1} & a_{3,2} & a_{3,3}
\end{pmatrix}
\]

\[
\tilde{d} = \begin{pmatrix}
d_1 \\
d_2 \\
d_3
\end{pmatrix}.
\]

If the rotation matrix \( R \) is orthonormal then \( T^j_i(\tilde{p}) \) is a rigid transformation.

**Coarse Registration**

The goal of a coarse registration process is to compute efficiently a set of approximate registration transformations \( \{T^0_i(\tilde{p}) \ldots T^0_m(\tilde{p})\} \) to be applied to the data obtained from different views. These transformations are intended to align the data closely enough for a subsequently applied ‘fine’ registration procedure. The coarse registration procedures are designed to improve the chances of the fine registration procedure’s convergence to the global optimal solution.

Often the coarse registration is obtained by using a controlled scanning environment. In this case the relationships between the views’ coordinate frames \( \{T^0_i(\tilde{p}) \ldots T^0_m(\tilde{p})\} \) are known *a priori* from the calibration of the scanner with the mechanics of the sensing environment. The problems with using such well controlled and accurate scanning systems are: objects need to be placed in precise locations in the scene and the automation of the scanner or multiple scanners must be precisely calibrated. These
constraints result in more expensive equipment and limitations on the types of objects that can be scanned.

Human interaction can also be used to obtain coarse registration between object views. This a popular method used in some commercial systems. Here, the users are asked to select corresponding points in each of the views \( V_1, \ldots, V_m \). This manual matching of point correspondence gives the system an initial set of transformations \( \{ T_1^0(\mathbf{p}) \ldots T_m^0(\mathbf{p}) \} \) that can be refined further using optimization techniques.

A more automated approach uses matching techniques (on local surface or albedo data) to recover sets of point correspondences between the views without user interaction. The views \( \{ V_1, \ldots, V_m \} \) contain regions of overlap where for any pair of neighboring views \( V_i \) and \( V_j \) there are surfaces that are visible in both views. In these regions of overlap point correspondences can be found by matching similar points in the adjacent views. Let \( C_k = (\mathbf{p}_{i,k}, \mathbf{p}_{j,k}) \) be a pair of such corresponding points where \( \mathbf{p}_{i,k} \) lies on view \( V_i \) and \( \mathbf{p}_{j,k} \) lies on view \( V_j \). Many techniques have been developed to match points on arbitrary surface shapes [131, 28, 29, 73, 70, 69]. These methods encode surface geometry around points of interest which are used to identify and match “similar” points. The specifics of the features used and the matching techniques are found in Section 2.5.3.

For automated registration of multiple views the sets of corresponding points between views are often not sufficient. Since the characteristics that the matching technique uses are not unique, there exists a real possibility that the sets of corresponding points contain mistaken matches [29, 69]. To counteract this, each of the corresponding points between the two views are grouped based on their ability to
help form a consistent hypothesis for the registration of the views. Some common consistency tests between pairs of corresponding points \( C_k \) and \( C_l \) are:

- the inter-point distance \( \| \vec{p}_{i,k} - \vec{p}_{i,l} \| - \| \vec{p}_{j,k} - \vec{p}_{j,l} \| < \epsilon_1 \),
- angle difference \( \| \vec{p}_{i,k} \cdot \vec{p}_{i,l} - \vec{p}_{j,k} \cdot \vec{p}_{j,l} \| < \epsilon_2 \),
- and orientation change \( \| \vec{n}_{i,k} \times \vec{n}_{i,l} - \vec{n}_{j,k} \times \vec{n}_{j,l} \| < \epsilon_3 \) (where \( \vec{n}_{i,k} \) is the estimated surface normal at point \( \vec{p}_{i,k} \)).

The geometrically consistent corresponding points in two viewpoints are then used to produce the transformation \( T^j_i(\hat{p}) \) relating views \( V_j \) and \( V_i \). If more than one consistent hypothesis results from the grouping, then each group is used to produce a transformation \( T^j_i(\hat{p}) \) and these are compared based on their ability to register the two views \( V_j \) and \( V_i \) with the least error.

**Fine Registration**

Given a set of coarse registrations \( \{ T^0_1(\hat{p}) \ldots T^0_m(\hat{p}) \} \) relating all the views \( \{ V_1, \ldots, V_m \} \), the fine registration task aims to make small changes to the individual transformations \( \{ T^0_1(\hat{p}) \ldots T^0_m(\hat{p}) \} \) to improve the overall “quality” of the registration. The quality criterion typically rewards a small distance between points in the corresponding surfaces of two ‘overlapping’ views.

**Two View Optimization**

The Iterative Closest Point (ICP) algorithm (Figure 2.8) developed by Besl and McKay [12] is widely used for the fine registration task, and uses a nonlinear optimization procedure to further align the data sets. Let \( \mathcal{P}_1 \) be the set of points from the first data set \( \{ p_1^1 \ldots p_{Np_1}^1 \} \), where \( Np_1 \) is the number of points in the set. Let \( \mathcal{P}_2 \)
be the set of points from the second data set \( \{p_1^2 \ldots p_{Np_2}^2\} \), where \( Np_2 \) is the number of points in the second data set. Since ICP is an iterative algorithm, define an incremental transformation \( T(l) \) such that \( \mathcal{P}_2(l + 1) = T(l) \cdot \mathcal{P}_2(l) \); \( T(l) \) reduces the registration error between \( \mathcal{P}_1 \) and the new point set \( \mathcal{P}_2(l + 1) \) by moving the points \( \mathcal{P}_2(l) \). \( \mathcal{P}_2(l) = \mathcal{P}_2 \) initializes the system.

\[
\mathcal{P}_2(0) = \mathcal{P}_2 \\
\text{DO} \\
\quad \text{FOR every point in } \mathcal{P}_2(l) \\
\quad \quad \text{Find the closest point in } \mathcal{P}_1. \\
\quad \text{END FOR} \\
\text{The closest points form a new point set } \mathcal{Y}(l) \text{ where the pairs of points} \\
\quad \{ (p_1^2, y_1), \ldots, (p_{Np_2}^2, y_{Np_2}) \} \text{ form the correspondences between } \mathcal{P}_1 \text{ and } \mathcal{P}_2(l). \\
\text{IF registration error between } \mathcal{P}_1 \text{ and } \mathcal{P}_2(l) \text{ is too large} \\
\quad \text{Compute registration } T(l) \text{ between } (\mathcal{P}_2(l), \mathcal{Y}(l)). \\
\quad \text{Apply registration } \mathcal{P}_2(l + 1) = T(l) \cdot \mathcal{P}_2(l). \\
\text{ELSE} \\
\quad \text{STOP} \\
\text{END IF} \\
\text{WHILE } \| \mathcal{P}_2(l + 1) - \mathcal{P}_2(l) \| > \text{threshold}
\]

Figure 2.8: Iterative Closest Point(ICP) algorithm.

The algorithm summarized in Figure 2.8 starts by establishing correspondences between the two data set \( \mathcal{P}_1 \) and \( \mathcal{P}_2(0) \). Then the registration error between the two data sets is computed and tested to see if it is within a specified tolerance. In Besl and McKay [12] the registration error was defined with respect to the correspondences between points in the data sets. For each point in \( \mathcal{P}_2(l) \) the closest point in \( \mathcal{P}_1 \) is found. This forms a new point set \( \mathcal{Y}(l) \) where point \( y_k \) denotes the closest point in
$P_1$ to the point $p^2_k(l)$ in $P_2(l)$. Then the registration error between $P_1$ and $P_2(l)$ is

$$E = \frac{1}{N_{p_2}} \sum_{k} \|y_k - p^2_k(l)\|^2.$$ 

If the registration error is too large then the transformation between the views is updated based on a traditional non-linear optimization technique like Newton’s method. The new registration is applied to $P_2(l)$’s points and tested to see if the optimization has reached a local minimum. This process continues until the registration error falls below a desired quality threshold or sufficiently small changes in the motion between $P_2(l)$ and $P_2(l + 1)$ are detected indicating convergence of the solution to a local minima.

The parameter space explored by ICP can contain many local minima. In order to converge to the global minima the initial parameter estimate must be reasonably close to the true value to avoid converging to a non-optimal solution. This issue motivated the development of the coarse registration procedures discussed above. An alternative to the ICP approach would be to use a different non-linear optimization algorithm (such as simulated annealing or evolutionary programming) capable of climbing out of local minima.

Zhang [146] has added elements to the general ICP algorithm to handle outliers and occlusion, and to perform general subset-subset matching. The modifications changed the ICP algorithm’s (Figure 2.8) routines to select closest points between two point sets $P_1$ and $P_2(l)$. Zhang improved the original ICP algorithm by adding some heuristics for whether a particular pair of closest points should be included in the estimation of the new transformation. The new system showed better tolerance for larger uncertainties in the initial guess of pose, erroneous data, and missing data while still managing to converge to the optimal solution.
Chen and Medioni [26] also use an iterative refinement of initial coarse registrations between views to perform fine registration. Instead of minimizing the distance between the closest points in the two point sets, Chen and Medioni employ orientation information. They devised a new least squares problem where the energy function being minimized is the sum of the distances from points on one view’s surface to the tangent planes of another view’s surface. This tends to not only minimize the distance between the two surfaces but also match their local shape characteristics [8]. Because of the expense of computing intersections between lines and the discrete surface (generated from the tangent planes) they subsample the original data to obtain a set of “control” points. The set of control points are points located on the surface conforming to a regular grid, furthermore the points are then selected from this subsampling of the surface (regularized grid) that lie in smooth regions on the surface. In these areas of smoothness the surface normal can be calculated more reliably.

Dorai et al. [21] proposed the use of an “optimal” minimum variance estimate (MVE) of the registration between two views assuming uniform i.i.d Gaussian noise in the sensor’s measurement of the depth coordinate; the \((x, y)\) measurement in each input image is assumed to be noise free. In their discussion of the results they show that for both synthetic and real data the MVE estimate produces lower registration error than the least squares estimation.

**Multiple View Optimization**

In the process of forming complete 3D models from multiple images, often more than two viewpoints of data have to be combined to form a complete object model. Shum *et al.* [124], Gagnon *et al.* [51, 8], Neugebauer [103], Blais and Levine [16], and Chen and Medioni [26] have examined the propagation of registration error when
multiple pairwise estimates of registration transformation are concatenated (e.g. \( T_0^4 = T_3^1 \cdot T_2^3 \cdot T_1^2 \cdot T_0^1 \)). Figure 2.9(a) shows graphically, for a series of views \( \{ V_0, \ldots, V_4 \} \), one possible way of calculating the pairwise estimates in the registration process. This sequential estimation of the multiple view registrations can lead to large registration errors between views \( V_0 \) and \( V_4 \). To minimize this error the registration process can be performed simultaneously for multiple viewpoints, thus minimizing the registration error for the entire object.

![Network Topologies](image)

Figure 2.9: Two common network topologies used in Multiple View Registration.

The star network topology in Figure 2.9(b) shows the most commonly used relationships between view coordinate frames. In the network the nodes represent each view, while the links represent transformations from one view’s coordinate frame to another. The central node (node 0 in the figure) represents the object’s coordinate frame. In this topology every view’s points \( V_i \) can be transformed into any other view’s coordinate frame \( f_j \) by only two view transformations \( T_0^j \cdot T_i^0 \). This topology reduces the propagation of registration error between any two views \( V_i \) and \( V_j \) by decreasing the number of transforms that have to be chained together.
The star topology simultaneously “minimizes” the distance between any two nodes [51, 8]. By using this topology (Figure 2.9(b)) at most only two pairwise transformations are needed to project data from one viewpoint’s coordinate frame to any other viewpoint’s coordinate frame. Gagnon et al. [51, 8] also show that the calculation of each transformation cannot be decoupled and done sequentially. View \( V_i \)’s data may overlap more than one neighboring view. Therefore, the transformation \( T_i^0 \) taking points from \( V_i \) and the object coordinate frame not only affects the registration error between \( V_i \) and \( V_j \) but also any other view whose data overlaps view \( V_i \). Finding the “best transformation” between any particular view \( V_i \) and the common object coordinate system uses all the correspondences between all the views simultaneously. This will not only minimize the registration error between view \( V_i \) and \( V_j \), but also minimize the registration error between view \( V_i \) and any other overlapping view \( V_k \), where \( k \neq j \).

It is possible (due to self shadowing and sensing errors) that even within the region of overlap there will be missing or unreliable data. With this in mind, Gagnon et al. [51, 8] built on Chen and Medioni’s work [26] by adding an additional heuristic designed to discard unreliable control points used in the registration process. The heuristic tests the control points and their corresponding tangent planes in the overlapping views for consistency in the orientation of the tangent plane and the normal at the control point. Normally, if there is only a small error in the registration, the normal of the control point and the normal to the tangent plane of an overlapping view should be nearly parallel. But, due to larger than normal registration errors and errors in the scanning process, the difference can become quite large. This is a
strong indication that the control point should not be used to refine the estimate of
the registration between the views.

Neugebauer[103] used a graph topology to represent the relationships between
views in the multiple registration problem. The nodes of the network is the set of
3D data in each view and the links are the transformation matrices needed to map
the data from one view to the coordinate system of a neighboring view. The goal
is to obtain a well-balanced network where the registration error is similar for all
pairs of views (Figure 2.9(b)). A ‘central’ coordinate system is chosen and initial
transformation matrices are found to align each view with respect to this central co-
ordinate system. These transformation matrices are then incrementally refined (using
the Levenberg-Marquardt method) by minimizing the distance between points in one
view and all corresponding points in the other views. This nonlinear optimization is
guaranteed to reduce the registration error between the views. A statistical termina-
tion criterion is used based on an assumption that the residuals between views are
zero-mean Gaussian random variables. To speed up the refinement process, only a
few points from each view are used in the registration computation initially; as the
refinement process continues, more and more points from the scene are incorporated
into the estimate. This resolution hierarchy was shown to speed up the process of
registering the views.

Pulli [114] documented a multiview registration technique that begins with pair-
wise registrations, with a subsequent global alignment step. The latter step attempts
to balance registration errors across all views. Of note in this technique is its ability
to handle a large data set size (i.e. a large number of scans and/or very dense scans),
and a careful study of alternative alignment and optimization processes.
Alternative coordinate systems have proven useful in some registration contexts. Chen and Medioni [26] found it useful to convert all view data into spherical coordinates. With all the view data referenced to a single spherical coordinate system, they found it easier to combine the overlapping view data to a single object model. In regions of overlap the corresponding points between views can be determined using the elevation and azimuth angular coordinates of points in the overlapping regions. These corresponding points are used to refine each view’s transformation \( T_i^0 \) to the object’s coordinate system. The resulting registered data was then transformed back to Cartesian coordinates.

Shum et al. [124] used a re-sampling mesh to represent each view and determine the transformation between them. The goal of the approach is to transform all the local coordinates to a global coordinate system that will represent the complete model, thus yielding the transformations for each view to the global coordinate system (Figure 2.9(b)). The solution is found by generalizing a principal component analysis problem to a weighted least squares problem. The approach shows noticeable improvements over sequential registration of the views. One drawback of this method is the use of re-sampling, which may cause some of the fine detail of the object to be lost.

In both Besl and McKay’s [12] and Chen and Medioni’s [26] work the process of finding the points or tangent surfaces that match in the error minimization process can be quite expensive. In ICP linear search is often employed while Chen and Medioni use a method similar to Newton’s root finding technique. To reduce the search complexity Blais and Levine [16] calibrate the sensor so camera parameters are known. With the camera parameters available, 3D coordinates can be accurately
projected into the sensor’s image plane. This allows range pixels from view $V_i$ of an object to be quickly matched with range pixels of another view $V_j$ by projecting pixels from the $i^{th}$ view into the image plane of view $V_j$. This technique uses an initial estimate of the rigid transformation between the views then refines it using corresponding points found by projecting pixels from one view into another views image planes. Another difference between the work of Blais and Levine [16] and other multiple view registration techniques is their use of simulated annealing to minimize the registration error instead of least squares or MVE estimators.

Eggert et al. [39] extended the traditional pairwise ICP algorithm [12] to solve the simultaneous registration of multiple views to achieve a global optimal registration for all views. In this new system all the data is transformed using coarse registration techniques into a single coordinate system. The data from each view is first smoothed using a median filter to remove outliers, then smoothed again using a Gaussian filter before estimating the surface normal at every point. The Gaussian filtered data is only used for surface normal estimation and the median filtered data is used in all other portions of their algorithm. At each iteration of their algorithm point position and surface normal orientation are used to establish correspondence between points. During correspondence all the view data is search to find the best match. Since this can be allot of data they used hierarchical search along with space carving methods (k-D trees) to speed up the identification of the closest points. The refinement of the registration of the view data is accomplished by simulating a damped spring mass mechanical system. The corresponding points are linked with a hypothetical spring and incremental motions are calculated based on the forces and torques produced by the network of springs connecting the data sets together. Since the calculation of
point correspondence is still expensive the force based registration is allowed to iterate until convergence before new point correspondences are identified. Convergence is identified when the motion of the view data is sufficiently small. A drawback of this approach is that it requires that every portion of the surface be present in at least two views. The authors have presented some heuristics to allow them to work with and register data where this assumption is violated, but the authors still seek a more complete solution that does not require a heuristic based on thresholds.

2.4.2 View Integration

Once the registration process has been completed, the data contained in each view can then be transformed into a single object coordinate system. Depending on the type of sensor that captured the data and the method of registration, the points on the surface can be in one of two basic forms. In the first case, all the points form a cloud where (at least initially) no connectivity information is known. The second case, the data is structured via relationships (e.g., image-plane connectivity) known a priori in each view. The method of integrating the data depends on the form of the data (structured or unstructured) and the final representation required by the application.

Unstructured Points

A polyhedral mesh constructed from the unstructured point cloud is useful for visualization and reasoning about the object’s geometric properties. The potential difficulties with generating this approximation are as follows: recovering the correct genus of the object, dealing with missing data (undersampling of the surface), and search.
Since the data is initially unorganized, the task of simply identifying the set of points closest to a particular point in the cloud can be expensive computationally. There exist space carving methods to speed up the search process (e.g. the k-D tree and uniform space partitioning) [49, 121]. With the emergence of connectivity information relating points in the cloud, some sense of the object’s surface takes form. In particular, the orientation of the surface can be estimated more reliably from points within some neighborhood. Both Hoppe et al. [63] and Oblonsek and Guid[106] use an approximation of the surface normal to begin reasoning on the reconstruction of the surface shape.

The surface normal at a point in an unstructured cloud is easily calculated, but its orientation has a 180 degree ambiguity since the inside and outside of the object are not explicitly known. This information is important for recovering the correct genus of the object. To insure that the correct genus is recovered, Hoppe et al.[63] used a graph optimization method to ensure neighboring points retain an consistent normal direction (neighboring normals point in the same general direction). Oblonsek and Guid[106] started with the point in the cloud with the largest z value and assumed that its normal was oriented in the +z direction; subsequently, local consistency of the normal vector was enforced.

The normal direction attached to each point in the cloud give a notion (at least locally) of which part of the tangent plane’s half space is inside or outside the object. This is used by Hoppe et al.[63] along with an isosurface to identify intersections of the object’s surface with the cubes of the isosurface. Then by using an marching cube algorithm [88] they were able to recover a polygonal approximation of the object’s surface.
Oblonsek and Guid[106] developed another method to form a polygonal approximation to the surface represented in the point cloud. Their technique starts a seed point in the cloud, then grows the polygonal mesh using neighboring information and the surface normal estimates. Their growing technique avoids the initial test of all the point’s normal consistencies used by Hoppe et al.[63], while still allowing the system to recover objects of genus greater than zero.

**Structured Points**

With structured input data (*e.g.* range images), the connectivity of the points in each view is already known. What remains is to integrate the data from separate views to complete a single representation of the object. The different methods for combining view data all address the following problems arising from the capture and registration of the data.

1. The sensor used to capture data produces errors in the measurements of the surface. This is especially present for some sensors along the silhouette of object in each view (where depth estimation techniques are least reliable).

2. The view data overlaps in regions. The redundancy in the view data aids in finding the registration, but also insures that accurate measurements of the object’s surface exist in at least one view. These overlapping measurements must be integrated.

3. Self occlusion and sensor errors result in holes in the data. The solutions to these problems, and to the general mesh integration problem, can be classified as mesh-based or volume-based.
In mesh-based techniques, the problem is to merge multiple polygonal meshes into a singular polygonal mesh that accurately represents the composite geometry. In the process of ‘stitching’ together the views, seams can appear clearly showing the boundary between two separate views (Figure 2.10). These seams result from several factors; registration error between views and sensor errors in the regions near the object silhouette. Both Soucy and Laurendeau [128] and Turk and Levoy [141] used an integration technique coupled with a geometric averaging technique to combine redundant data in the regions of overlap. They differ in the order in which they perform the operations. Soucy and Laurendeau [128] combine the redundant information from different views first to form a point set that is then stitched together. Turk and Levoy [141] first stitch together the meshes (removing redundant polygons in the overlapping regions) to form the final mesh, then use the vertices of the removed polygons to form a consensus of the surface shape in the regions of redundancy by moving the vertices of the final mesh. Both methods result in a local smoothing of the data in the regions of redundancy. Furthermore, neither method deals with holes in the data resulting from errors in the sensor measurement or due to the sensor’s geometry. To fix the latter problem in their techniques, the holes were fixed by a operator that formed polygons in these regions, filling in the holes.

Soucy and Laurendeau [127] address more practical issues of surface integration for a Computer Aided Graphic Design program. In these applications a designer will most-likely take a series of views of the object to build an initial model of the object being modeled. Then because of self-occlusion and sensor errors the designer often has to take scans of additional views of the object to fill in large holes or recapture fine detail in the object. In this case repeating the integration of all the
Figure 2.10: A seam resulting from a registration error.
views simultaneously for every additional view may not be the best solution especially for an interactive model building program. Instead Soucy and Laurendeau have developed a dynamic integration algorithm that adds additional data to a model by modifying only the necessary data structures. The new algorithm maintains the advantages of simultaneous integration techniques while maintaining the ability to sequentially integrate new data to a model.

The volume-based set of techniques limit the reconstructions to objects with a genus of zero. Under this assumption, a representation can be formed without worrying about the difference between samples resulting from an actual hole in the object and a hole resulting from sensor errors.

Under the genus zero assumption, a deformable surface can be used conform to the object’s data points. These deformable surfaces can be thought of as a balloon being inserted inside the object then inflated until the surface of the balloon conforms the data (Chen and Medioni [27]). Alternatively, a filled balloon can be placed around the object then slowly the air can be released until the balloon forms a mold of the data (Shum et al.[124]). These techniques solve several problems related to the registered data. First the deformable model implicitly handles redundancy in the data, since the deformable model contains its own vertices which are moved based on an energy minimization between the model and the data. Second, the deformable model will fill in the holes in the data by stretching the deformable model’s surface between the boundaries of the hole. However the fine details are often lost due to the smoothing effect of the energy minimization and the fixed set of vertices in the deformable mesh. Because of this, Chen and Medioni [27] added adaptive mesh refinement of the

50
deformable surface that allows the model to adaptively subdivide in order to expand into regions of fine detail that may have been lost otherwise.

A novel volume-based technique developed by Reed et al. [118, 117] used a constructive solid geometry (CSG) technique to form a solid model of the object from multiple registered range views. Each view forms a solid by sweeping the range data from each object away from the scanner filling in the space self occluded by the object. The surfaces of this new object are labeled as being either being visible (part of the sensed data) or occluded by the object view. This labeling can be used for view planning since it labels portions of the viewspace that have not been covered by the previous view. For each view, solid objects are formed using similar sweeping operations. Once all the visible portions of the object have been covered by one or more views, the view solid models are intersected to form a singular solid object model.

2.4.3 Mesh Refinement and Optimization

Due to the discrete nature of the data being used to model the object, the result will only be an approximation. The accuracy of that approximation depends on the ability of the sensor to capture data and ability of the system to register multiple data sets. With the availability of high resolution range scanners and better methods of registration, the quality of the reconstructed model can be quite good. Still, there may exist a need to refine the model based on application requirements [58].

In graphics applications the important properties of a refined model are that it occupy roughly the same shape, that its reconstructed surface normals be close to those of the original model (important for shading), and finally that the object's color texture can be mapped to the reconstructed model. All these factors improve
the model’s ability to yield synthetic images that appear very similar to the true scene and the initial unrefined model.

The neighboring vertices in a good quality polygonal mesh should be geodesically the same distance apart (Welch and Witkin [144]). For highly curved areas on an object, two vertices might be close geometrically, but very far apart geodesically (due to surface curvature changes). In these regions, the mesh representing the surface should be fine, while in regions where the surface is largely planar the density of the mesh can be coarse. This adaptive mesh density results in very good approximation of the object’s surface normal and curvature, and is a key goal of mesh refinement procedures.

The speed of the rendering depends directly on the number of polygonal faces in the mesh. Because of this, mesh optimization strategies have been studied to reduce the number of polygons needed to represent the model for the object [58, 140, 62, 60, 87, 30, 52, 61, 129, 126], while still maintaining a good approximation to the surfaces. Figure 2.11 shows an example of using mesh optimization strategies to reduce the number of polygons in the representation of a toy whale.

In the process of refining a model, care has to be taken to avoid simplifications that yield a topological change (e.g. creation of holes, surfaces smoothed to a line, etc.). This can happen when vertices are removed from the mesh and the mesh is then re-polygonized. To take care of this problem, most systems use heuristics to verify that the model’s topology does not change [140] during the refinement process.

During decimation, the process of removal of vertices, edges and faces can result in a smoothing and/or shrinking of the object’s surface. Care has to be taken while refining the object’s polygonal representation to avoid smoothing out the original model’s
Figure 2.11: In the first pair of images the whale mesh contains 6844 vertices. The second set of images shows a the result of reducing the mesh to contain only 3042 vertices. The right image in both pairs is a shaded version of the mesh to the left. The images were generated using Minolta’s Vivid software package.

sharp discontinuities and therefore shrinking of the object. Automated techniques tend to utilize a penalty function that measures the error incurred by iteratively refining the number of polygons in the mesh [58, 140, 62, 60, 87, 30, 52, 61, 129]. These functions guide the choices of which entities (vertices, edges and polygons) should be removed and where new entities should be placed.
In addition to decimation, another approach to reducing the amount of space needed to store a model is to change the representation of the model to something that is less space consuming than a dense polygonal mesh. Krishnamurthy and Levoy [83] studied the conversion of a dense polygon mesh to a B-spline surface patch representation for objects. An appropriately designed B-spline patch representation uses less computer storage space than a polygonal mesh that represents the surface with the same accuracy. In addition to being more memory efficient, the B-spline is differentiable and has been extensively used in the CAD/CAM industry for representation and design [41].

For the area of computer vision the criteria for a good representation of an object model may be different from the criteria employed in computer graphics. For example, Johnson and Hebert [71, 74] required that the models of the objects in their database be represented in polygonal mesh format with the density of the mesh as regular as possible. The regular mesh ensures that their “spin” image features (used to encode and match the shape of each object) approximates the true range image features scanned from the real object’s surface. The use of spin images for recognition is discussed at length in Section 2.5.3.

Another example is the identification of curvature extrema from polygonal mesh representations of surfaces. Curvature is often used in computer vision as a descriptive feature because of its detectability in a wide variety of imagery. Campbell and Flynn [22] studied some of the issues in identifying contours of curvature extrema on polygonal mesh surfaces. They found that identification of the curvature extrema on an object surface depended on the properties of the polygonal mesh used to represent it. Polygonal mesh representations with vertices guaranteed to be within an error
tolerance of the true surface typically will contain polygons with varying surface area depending on the magnitude of the the surface curvature. In regions where the surface is rapidly changing shape the polygonal mesh model must used small polygons to insure the error between true surface and polygonal model is within tolerance. This difference in area can then be used to aid in the identification of the curvature extrema on the mesh representation of the model.

Another concern is identifying ridges of curvature extrema. Campbell and Flynn [22] developed a curvature ridge/valley detector that used hysteresis and non-maximum suppression to build contours of curvature extrema. They found that following the true contour was often difficult since the edges of the polygonal mesh representation can be a poor polygonal approximation to the contour of curvature extrema. This results from poor approximation of the ridges of the object surface by the edges in the polygonal mesh. Often the edges of the mesh cut through the true contour instead of following it. This haphazard placement of edges in the polygonal mesh causes trouble in reconstructing contours from the representation.

The refinement of the model in large part depends on the application and the desired accuracy of the representation. A resampling of the data is often performed to improve the ability to reconstruct approximations to the surface normal and curvature on the model [22] and to reduce the redundancy in the representation. The result is a model that more accurately represents the model and all of its properties while still being efficient in time and space.
2.5 Free-form Object Recognition Systems

The preceding sections have surveyed the elements and techniques for representing free-form 3D objects, with a focus on mesh representations and their construction from range images. This section discusses several approaches to the recognition of such objects in intensity or range imagery. Additional surveys of this area (or of object recognition in general) include Besl and Jain [10], Brady et al. [18], Flynn and Jain [47], and Pope [113]. Our goal is to survey those systems that are relatively recent, are designed to recognize either single or multiple objects in the presence of occlusion, and that do not restrict the 3D shape of objects to be identified to simple shapes such as planes or cylinders. While some attention is given to recognition in intensity imagery to provide context, the focus here is techniques employing 3D (range) imagery.

2.5.1 Appearance-Based Recognition

Appearance-based approaches have become a popular method for object identification and pose location in intensity images, and has also been used to recognize free-form objects in range data, as noted below. This popularity is in part due to these methods' ability to handle the effects of shape, pose, reflectance, and illumination variations for large databases of general objects. An appearance-based recognition system encodes individual object views as points in one or more multidimensional spaces. The bases for these spaces are obtained from a statistical analysis of the ensemble of training images. Recognition of an unknown view is typically performed by projecting that view into the space(s) along the stored basis vectors and finding the nearest projected view of a training image.
The literature contains many variations on this simple and powerful basic idea. Early work in the area of appearance-based recognition included as a study of efficient representation of pictures of faces. Kirby and Sirovich [80] used principal component analysis to determine the best coordinate system (in terms of compression) for their training data. The basis vectors for the new coordinate system were termed \textit{eigen-pictures}. In later publications [143, 95] they are sometimes given equivalent terms: \textit{eigenfaces} or \textit{eigenimages} depending on the application. The term \textit{eigenfaces} is used when the training set is limited to face data because certain basis vectors, when viewed, have a facelike appearance.

The appearances of objects are encoded using principal component analysis on a set of training image data

\[ X = \{ \bar{x}_1, \ldots, \bar{x}_M \} \]

Each training image \( \bar{x}_i \) can be considered a column vector obtained from the original image \( [x_{i;jk}] \) by unraveling:

\[ \bar{x}_i = [x_{i1}, \ldots, x_{iN}]^T, \]

where \( N \) is the number of pixels in the image. Each image is typically normalized to unit energy \( |\bar{x}_i| = 1 \). For Lambertian objects the normalization factors out any variation in global illumination that might occur between trials. To accurately represent each object’s appearance, the object must be viewed in a set of poses and under a set of illumination conditions that captures those poses and conditions expected to be present in the recognition environment.
Principal component analysis is used to determine a set of orthogonal basis vectors for the space spanned by the training data $X$. To do this the average

$$\bar{c} = \frac{1}{M} \sum_{i=1}^{M} \bar{x}_i$$

is subtracted from every image $\bar{x}_i$ in $X$ yielding a centered set $X_c$. $X_c$ is then used to form an $N \times N$ covariance matrix:

$$Q = X_cX_c^T.$$  

Then the eigenvectors and corresponding eigenvalues $\{ (\bar{e}_i, \lambda_i), i = 1 \ldots N \}$ for $Q$ are determined. The set of $N$ eigenvectors is an orthogonal basis for the space spanned by $X$. This principal components representation can be used to approximate the training data in a low-dimensional subspace that captures the gross appearance of the objects. This enables appearance-based approaches to characterize the objects with the eigenvectors corresponding with the largest eigenvalues. Murase and Nayar [95] found that a subspace of 20 dimensions or less was sufficient to capture object appearance for pose estimation and object recognition purposes.

Let $\{ \bar{e}_1, \ldots \bar{e}_k \}$ be the set of eigenvectors corresponding with the $k$ largest eigenvalues. The centered vectors ($\bar{x}_i - \bar{c}$) in the training data are projected along $\{ \bar{e}_1, \ldots \bar{e}_k \}$ to form $k$-dimensional prototypes $\tilde{g}_i$. An unknown image $\tilde{s}$ that maps close to one of these prototypes is similar in appearance to the corresponding training image $\bar{x}_i$. This enables the system to identify object identity and pose.

An approximation to $\tilde{x}_j$ can be constructed using the basis vectors $\bar{e}_i$ and the projected point $\tilde{g}_j$ in the eigenspace. The reconstructed image is given by

$$\tilde{x}_j = (\bar{e}_1, \ldots, \bar{e}_k)\tilde{g}_j + \bar{c}.$$
and is only an approximation to $\bar{x}_j$ since only $k$ eigenvectors are used instead of $\min(N, M)$ (the rank of the covariance matrix $Q$), but the gross details of the object’s appearance are typically preserved.

Turk and Pentland [143, 142] noted that in using appearance-based recognition strategy a small set of eigenpictures could be used to describe and reconstruct faces from a large segment of the population. Their ideas were tested on a database of 16 subjects whose faces were captured under varying illumination, pose, and scale (2500 images in all). They found that the system had a higher sensitivity to changes in the size of the subject face in the digitized images than to changes in illumination and pose. They showed that appearance-based methods could be used to recognize complex sculpted surfaces (faces) in under one second.

Murase and Nayar [95] developed the use of appearance-based recognition and pose recovery for general objects. Using two eigenspaces they were able to recognize and locate pose for objects under varying lighting conditions. The universal eigenspace $U$ is formed from all training images $X$ and is used to determine the identity of an object in a scene image, while an object eigenspace $O_j$ is formed for each model $j$ from all training images $X_i$ containing that model, and used to determine the object’s pose and the illumination conditions. For each object a bivariate manifold is created, allowing interpolation between discrete points $g_i$ in the eigenspaces. The manifolds were constructed using a cubic-spline interpolation algorithm. The two parameters of the manifold were the rotation of the imaging turntable and the illumination direction. In general, three parameters are required to represent the rotational pose of an object in 3D and an additional two dimensions are needed to describe varying illumination positions (for a single non-directional light source). Thus the practicality of this sort
of system depends on the number of pose and illumination degrees of freedom as well as the complexity of the object shape and appearance under pose and illumination variations.

To identify an object in an image \( \mathbf{s} \), it is first projected

\[
\mathbf{g} = \left( \begin{array}{c} \mathbf{e}_1^T \\ \vdots \\ \mathbf{e}_k^T \end{array} \right) (\mathbf{s} - \mathbf{c})
\]

into the universal eigenspace, then matched with the closest object’s manifold. Naively, matching can be done by uniformly sampling the manifolds, but this is both inefficient in memory and time. Nene and Nayar [102, 101] developed a structured binary search technique to quickly search through a multi-dimensional eigenspace for the best match.

Other notable work in the appearance-based recognition of objects in intensity scenes includes the following.

- Turk and Pentland [142] discussed the utility of using trained neural networks to match \( \mathbf{g} \) with a person’s identity.

- Mukherjee and Nayar [91] experimented with radial basis function (RBF) neural networks to match \( \mathbf{g} \) with a point on an object’s manifold.

- Murase and Nayar [94] followed their earlier work on appearance-based matching of object with a study of illumination. The goal of the study was to determine illumination parameters to maximize the difference in appearance between objects. The study produced graphs of minimum distance between object curves in eigenspace as a function of light source direction, recognition rate as a function of SNR, and recognition rate as a function of the amount of segmentation
error. In all cases the experimentally determined optimal source was found to have a higher recognition rate in the presence of noise and segmentation errors.

- Mundy et al. [92] compared Murase and Nayar's [95] appearance-based approach with two geometric model based recognition approaches. The study concluded that the appearance-based system had the highest recognition rate, but it also had the highest number of false positives. This was due to the lack of a verification procedure, and to the sensitivity of current appearance-based methods to occlusion, outliers, and segmentation errors. The geometric model based approaches, being structured around the hypothesize and test architecture, had a smaller number of false positives.

- Black and Jepson [15] showed that the projection \( \mathbf{g} = [\mathbf{e}_1, \ldots, \mathbf{e}_k]^T (\mathbf{x} - \mathbf{c}) \) results in a least squares estimate when used for reconstruction (i.e., the calculation minimizes the squared error between image \( \mathbf{x} \) and its reconstruction \( \mathbf{g} \)). To compensate for the well known sensitivity of least squares techniques to outliers and gross errors, Black and Jepson used robust statistics in the calculation of the projection \( \mathbf{g} \). The addition of robust estimation improved the appearance-based method's ability to handle input images for which good segmentations were not available. A notable application was the development of a system to track and recognize hand gestures.

- A major problem with the appearance-based approaches is their lack of ability to handle more than one object in the scene with the possibility of occlusion. Huang et al. [65] proposed segmenting the input images into parts and using the appearance of the parts and their relationships to identify objects in the
scene as well as their pose. Later work by Camps et al. [24], following Huang et al. [65], enhanced and extended the earlier work, yielding recognition system that includes hierarchical databases and Bayesian matching.

- One drawback of an appearance-based part representation is the input image must be segmented at runtime before recognition can occur. This limits the class of objects that can be recognized to objects who can be segmented reliably. Most free-form objects do not lend themselves to easy repeatable segmentations. Ohba and Ikeuchi [107] and Krumm [84] avoided this problem by creating an appearance-based technique using local windows of the objects appearance. Ohba and Ikeuchi [107] were able to handle translation and occlusion of an object using eigenwindows. The eigenwindows encode information about an object’s appearance for only a small section of its view. Measures of detectability, uniqueness, and reliability were developed for the eigenwindows. These measures were used to omit eigenwindows from the training set if they are hard to detect (detectability), have poor saliency (uniqueness), or are sensitive to noise (reliability). Using the local eigenwindows they were able to identify multiple objects in cluttered scenes. Krumm [84] independently and roughly simultaneously devised a eigenwindow approach similar to that of Ohba and Ikeuchi [107]. The differences between these two approaches deal with the model construction and recognition procedure. Krumm employs an interest operator (points and corners) to identify possible non-overlapping eigenwindows.

- Edwards and Murase [38] addressed the occlusion problem inherent in appearance-based methods using a mask to block out part of the basis eigenimages $\mathbf{\tilde{e}}_i$ and
the input image $g_i$. The masks were used with a resolution hierarchy to search for an initial mask and object identity at low resolution then adaptively refine them through higher resolutions.

- Leonardis and Bischof [85] handled occlusion by randomly selecting image points from the scene and their corresponding points in the basis eigenvectors $[\mathbf{e}_1, \ldots \mathbf{e}_k]$. Their method uses a hypothesize-and-test paradigm, where a hypothesis is a set of image point locations (initially randomly generated) and the eigenspace prototype $g$. This method shows the ability to reconstruct unseen portions of the objects in the scene. Bischof and Leonardis [14] extended this earlier work to handle scaling and translation of objects in the scene.

- Rao [116] applied the adaptive learning of eigenspace basis vectors $[\mathbf{e}_1, \ldots \mathbf{e}_k]$ in appearance-based methods. The dynamic appearance-based approach is used to predict spatial and temporal changes in the appearance of sequence of images. The prediction modifies the eigenvectors by minimizing an optimization function based on Minimum Description Length (MDL) principles. MDL is used because it balances the desire to learn the input data without memorizing its details. This is important when the system needs to be flexible and not overspecialized to the training data.

- Appearance-based recognition has also been applied to range or depth imagery by Campbell and Flynn [23]. The $2\frac{1}{2}D$ depth data captures the shape appearance of a view. The principal component analysis of this data results in a set of characteristic shapes. These eigenshapes are used in much the same way as eigenpictures or eigenfaces to form a low dimensional subspace that is useful
for matching and pose recovery. An advantage to using range data in place
of albedo is that in most cases the illumination conditions at the time of the
scan does not affect the measurements of the depth data in the range image.
Traditionally appearance-based recognition systems trained on images captured
by rotating the object on a turntable and varying the elevation of the light with
respect to turntable. This resulted in two degrees: of pose freedom the first for
the rotation of the turntable, and the second for the light position that could
be recovered by the appearance-based recognition system. Since illumination
was not a factor in the shape appearance, the authors addressed the recovery
of 3D rotational pose.

The training of the system consisted of uniformly sampling the view sphere by
a pseudo-regular tessellation of the sphere. At each vertex of the discrete view
sphere defines a viewpoint for which a set of training images are taken. The
training images were then used to build, along with Nene and Nayar’s [102]
efficient search technique in high dimensional spaces, an efficient object recog-
nition system for free-form shapes. This system was tested on two databases the
first contain 20 free-form objects and the second contain 54 mechanical parts
(most of the surfaces here could be described by the natural quadrics). Both
databases were tested to identify the influence that; density of pose sampling,
dimension of the matching subspace, and size of training images had on the
probability that the correct object would be detected. The authors found that
image size was not nearly as important as the number of training images taken
for each object and the dimension of the subspace. Campbell and Flynn also
noted that an appearance subspace of 20 dimensions was sufficient for accurate
object recognition. Their system was able to identify the objects using shape
with 91% accuracy in about a second on a Pentium class PC.

Table 2.2 summarizes the various appearance-based recognition techniques sur-
veyed above. For each technique, the table indicates whether the technique can
handle multiple object scenes with occlusion and changes in the size of the objects in
the database. It also reports the largest documented database size and the recogni-
tion rate obtained \( \frac{N_c}{N_{\text{trials}}} \) where \( N_c \) was the number times the object’s were correctly
identified and \( N_{\text{trials}} \) was the total number of trials.\(^3\) In some of the papers surveyed
the number of objects or the recognition rate was not reported and are marked in the
table as NR. The entry for the system of Black et al. \cite{15} reports N/A for database
size and rate because it was designed to track objects using recognition and multiple
object databases were not explored.

\subsection*{2.5.2 Recognition from 2D Silhouettes}

In addition to appearance-based techniques, object silhouettes have been used to
characterize free-form objects. In a controlled environment an object’s silhouette can
be quite useful to determine an object’s identity and pose. This subsection reviews a
few representative techniques employing free-form contours in intensity imagery for
recognition.

Mokhtarian \cite{90} developed a complete object recognition system based on closed
object silhouettes. The system is designed to recognize free-form objects that have
only a few stable views in an environment where only one object will be present.
To do this, a light-box is used to illuminate the object and make the boundary

\(^3\)In the experiments of Nayar et al. \cite{95, 98}, tests using a 100 object database reported 100%
correct recognition for a database subset of 20 objects that do not contain self similar viewpoints.
<table>
<thead>
<tr>
<th>Technique</th>
<th>Occlusion</th>
<th>Scale Changes</th>
<th>Largest Database</th>
<th>Recognition Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kirby et al.[80]</td>
<td>No</td>
<td>Yes</td>
<td>NR</td>
<td>NR</td>
</tr>
<tr>
<td>Turk et al.[142]</td>
<td>No</td>
<td>No</td>
<td>16</td>
<td>1.0</td>
</tr>
<tr>
<td>Nayar et al.[95, 98]</td>
<td>No</td>
<td>Yes</td>
<td>100</td>
<td>1.0</td>
</tr>
<tr>
<td>Black et al.[15]</td>
<td>No</td>
<td>No</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Camps et al.[24]</td>
<td>Yes</td>
<td>No</td>
<td>24</td>
<td>1.0</td>
</tr>
<tr>
<td>Ohba et al.[107]</td>
<td>Yes</td>
<td>No</td>
<td>NR</td>
<td>NR</td>
</tr>
<tr>
<td>Krumm et al.[84]</td>
<td>Yes</td>
<td>No</td>
<td>2</td>
<td>0.8</td>
</tr>
<tr>
<td>Edwards et al.[38]</td>
<td>Yes</td>
<td>Yes</td>
<td>6</td>
<td>0.89</td>
</tr>
<tr>
<td>Leonardis et al.[14]</td>
<td>Yes</td>
<td>Yes</td>
<td>20</td>
<td>NR</td>
</tr>
<tr>
<td>Rao et al.[116]</td>
<td>Yes</td>
<td>No</td>
<td>NR</td>
<td>NR</td>
</tr>
<tr>
<td>Campbell et al.[23]</td>
<td>No</td>
<td>Yes</td>
<td>54</td>
<td>0.91</td>
</tr>
</tbody>
</table>

Table 2.2: Summary of appearance-based recognition techniques.
between background and object easier to detect, and objects are isolated by simple thresholding. Boundary curves (extracted by contour following) are then represent by calculating its Curvature Scale Space (CSS) representation. The matching of CSS curves is done based on the location of the maxima of the curvature zero-crossings. For convex boundary curves the CSS representation is smoothed until only four maxima points are remaining, while concave curves utilize all maxima obtained at a given scale. During recognition the aspect ratio of the object’s silhouette is used to pre-filter possible scene/model matches before the silhouettes are matched. This technique provides a fast way of matching the coarse features of a scene silhouette with an object’s silhouette. The best silhouette matches are then verified by registering the two curves and measuring the error. The system correctly identified all 19 objects in its database.

Ponce and Kriegman [112] used contours of discontinuous surface normal and occluding contours to recognize 3D objects. Parametric surface models of the objects yield candidate model contours. Using elimination theory, implicit equations of the parametric patches are found and the intersections between the patches are used to construct an implicit equation of the contour. A weak perspective projection model is used to correlate 3D contours with 2D contours found in the intensity images. Image contours are manually pruned and grouped into clusters corresponding to a single object. The system was only used to model surfaces of revolution, but could handle multiple object scenes with moderate amounts of occlusion. Their results showed quick recognition times ($\approx$ 30 secs) with good location accuracy (average error between 0.4-1.4 pixels).
Joshi et al. [78, 79] used HOT (high order tangent) curves to model a smooth object for recognition. These curves can be identified from points on an object's silhouette in an intensity image. Sequences of these points can be tracked to recover the 3D curve. The angles between tangent lines and ratio of distances between points on contour are useful to form scale and pose independent features of the object. This in turn can be used to index a recognition table. Their experiments tested the system on four objects (a squash, a pear, a banana and a duck decoy) and showed that the HOT curve representation was able to recover the objects identity and pose for every test, but only after a verification step pruned out false matches.

Internal edges have also been used to help recognized free-form objects in intensity images. Chen and Stockman [25] used both silhouette and internal edges to recover pose and identity of 20 free-form object models from intensity imagery. The silhouette curve was first used with a invariant feature indexing system to build candidate model and pose hypotheses. A part-based contour representation was incorporated to tolerate occlusion. The invariant attributes of these curve segments were then used to index into a hash table. This results in matches between possible model parts and observed parts. These matches are then grouped into consistent hypotheses based on model identity (associated with each part) and a rough pose estimate. The hypotheses are then verified by matching model edge maps to the observed edge maps. The verification step also produces a refined estimate of the object's pose. During verification both the silhouette and internal edges are used. The inclusion of the internal edges improved the performance of the verification step in rejecting false hypotheses of object identity and pose.
2.5.3 Free-Form Object Recognition in Range Data

Besl [13] reviewed the difficulties in matching free-form objects in range data using point, curve, and surface features. The computational complexity of such matching procedures can quickly become prohibitive. For example, brute-force matching of 3D point sets was shown to have exponential computational complexity. Because of this, Besl and McKay [12], Stein and Medioni [131], Johnson and Hebert [73, 70, 72], and Chua and Jarvis [29] all have developed techniques to reduce the amount of computation required. These methods often group corresponding model and scene point pairs into sets that can be used to determine object pose and verify the existence of the model in the scene.

Range images allow the computer vision practitioner more complete information about object geometry. Their ability to capture 3D points on the viewable surfaces can be exploited to compare geometric relations between range images and training data or 3D models. While their utility in construction of 3D models has been a topic of relatively recent interest (as surveyed above), the recognition of objects in depth maps has a longer history and systems developed for this purpose have demonstrated a variety of approaches. The purpose of this section is to describe recent systems that were developed for free-form object recognition using range images.

A technique formulated by Nevatia and Binford [104] used symbolic descriptions derived from a generalized cone part segmentation of range images to recognize free-form articulated objects (doll, horse, snake, glove and a ring) in the presence of occlusion. The parts (generalized cones) are used to form a symbolic description of the scene built up using part properties (e.g. axis length, average cross-section width) connectivity relations (e.g. number parts connected) and global properties
(e.g. number parts, number of elongated parts). In an effort to eliminate some of the objects, distinguished parts are employed to index into the database of possible objects. Distinguished parts are generalized cones that are much wider than other parts in the database. These parts are more likely to be segmented properly in the presence of occlusion. The scene and model descriptions are then compared, starting with matching distinguished parts. These matches are then grown by matching other distinguished parts whose properties and connections are consistent. This is equivalent to matching two graph descriptions of the objects. The resulting scene graph of connected parts is allowed to be smaller than the model graph because of the presence of occlusion and segmentation errors. The quality of the graph match is then evaluated by the quality of the individual part matches. This representation works well for objects who have very different symbolic descriptions.

Raja and Jain [115] developed a system for fitting and classifying deformable superquadrics (a subset of the set of all free-form objects). They fit a deformable superquadric to range data where the deformations considered were tapering and bending. The parameters that define the fit superquadric are used to classify the shape into one of twelve different geon classes. The geon classes are used to discriminate between ‘qualitatively different’ shapes. Qualitative differences include straight vs. curved axes, straight vs. curved cross-section, and increasing, decreasing, or increasing-then-decreasing cross-sectional area along the primary axis of the superquadric. Superquadrics recovered from noisy or rough range images caused problems with classification of the superquadric with the correct geon class. With real range images their classification method correctly identified the geon class 77% of the time and with synthetic imagery the correct identification rate was 87%.
Surface curvatures has also been used to describe an object’s surface. For a surface the curvature at a point is characterized by finding the directions in which the surface normal changes the most and the least rapidly. These principal curvatures $\mathcal{K}_1$ and $\mathcal{K}_2$ respectively encode the rate of change of surface orientation in the two extremal directions. Typically the principal curvatures $\mathcal{K}_1$ and $\mathcal{K}_2$ are used to characterize and encode discriminatory information about an objects.

Thirion [137] used surface curvature to define a global representation of free-form objects. The method uses curvature extrema on the surface to find critical points and contours. The extremal point/contours are defined as zero crossings between maxima and minima of Gaussian curvature ($\mathcal{K} = \mathcal{K}_1 \ast \mathcal{K}_2$). The representation has been used on real range data as well as synthetic models to find extremal meshes of the object. Thirion notes that the representation is still too complex to be practical for object identification or pose localization, because the description does not lend itself easily to a quick matching technique.

Surface curvature has also been used to classify local surface shape into a small set of representative shapes [10, 36]. Besl and Jain used Gaussian curvature ($\mathcal{K} = \mathcal{K}_1 \ast \mathcal{K}_2$) and mean curvature ($\mathcal{H} = (\mathcal{K}_1 + \mathcal{K}_2)/2$) to classify local surface shape into eight basic categories. Dorai and Jain [36] extended this earlier work by defining two new curvature measures: the shape index ($\mathcal{S} = 1/2 - \frac{1}{\pi} \arctan \frac{K_1 + K_2}{K_1 - K_2}$) and curvedness ($\mathcal{R} = \sqrt{(K_1^2 + K_2^2)/2}$), where the shape index $\mathcal{S}$ now defines (classifies) the local shape into nine shape types (very similar to Besl and Jain’s work [10]) and the curvedness measures the magnitude of the curvature change. Dorai and Jain [36] use these new measures along with a spectral extension of the shape measure to build a view-dependent representation of free-form objects. Their system (named COSMOS, for
‘Curvedness-Orientation-Shape Map On Sphere’) uses a histogram of shape index values to characterize the surface curvature of a view. The histogram bins store the amount of area on a view that lies within a range of shape index values. These histograms (called shape spectra) can be quickly matched using moments and are invariant to rotations about the optical axis. The system builds up an object database made up of many views of each object to be recognized. To reduce the complexity of the search, views of an object are grouped based on their similarity into clusters. Then for each cluster a prototype shape spectral histogram is found using averaging. The process to match a scene shape spectra histogram with the database first matches the scene with each cluster’s prototype. Then the top \( n \) clusters that match well with the scene are examined to find which views in the clusters best match the scene. The view that best matches the scene identifies the object and pose. The translation and rotation parameters have yet to be determined. The view clustering scheme for the recognition database reduces the amount of time it takes to produce a match. On average only 20% of the views had to be matched from a database with a total of 20 objects. This recognition scheme works well for single object scenes not containing polyhedral objects.

In addition to curvature based features, deformable polyhedral meshes have been used to encode the local surface shape. Pipitone and Adams [111] used a connected set of equilateral triangles (Figure 2.12) to characterize an object shape by deforming it to the shape of the surface. The crease angle (Besl [9]) between pairs of triangles in the mat are used to build a pose invariant feature vector \( \Theta \) that characterizes the local surface orientation change. The number of angles \( N_a \) resulting from the mat of
Figure 2.12: Example of 4th order operator on range data sampled from a rubber duck.
triangles gives the order of the operator $\Theta$, where

$$
\Theta = \begin{pmatrix}
\theta_1 \\
\vdots \\
\theta_{N_a}
\end{pmatrix}.
$$

The length of the sides of the equilateral triangles can be used to control the operator’s ability to capture fine detail or reject noise in the surface shape of the object. To completely encode an object, the tripod operator is randomly placed on its surface enough times to insure sufficient coverage of all the surfaces of the object. Recognition in this context was performed using maximum a posteriori (MAP) estimation of the object identity given the observed tripod features. Nonparametric density estimates were employed in the MAP estimator. By using ten tripod operators and picking the most likely object, a correct recognition rate of 92% was achieved for a four object database.

Stein and Medioni [131] used changes in surface orientation to match local patches of surfaces. The local nature of the matching technique allowed them to find multiple free-form objects in a cluttered scene. To provide good matches between corresponding scene points and known model points they devised a novel method to measure the difference between two relative surface normal distributions (Figures 2.13, 2.14). For a given point $P$ on a surface the normals a distance $\rho$ away from $P$ contain some structural information about the surface around $P$ (Figure 2.13). The distribution of all the normals $N_\rho(\theta)$ on the surface around $P$ are called a ‘splash’, because its appearance can be strikingly similar to a splash in water. Then to encode relative information about the normals $N_\rho(\theta)$ a spherical coordinate system is used (Figure 2.14), where the angles $\phi(\theta)$ and $\psi(\theta)$ give the relative orientation of $N_\rho(\theta)$ with respect to $P$’s normal $N$ and the $X(\theta)$ axis. $X(\theta)$ is perpendicular to $N$ and lies in the
plane containing $P,N$ and the point $\rho$ distance away from $P$ and angle $\theta$ from where the encoding started. As $\theta$ is varied from 0 to $2\pi$ the values of $\phi(\theta)$ and $\psi(\theta)$ form a 3D curve $\mathbf{v}(\theta) = \left( \begin{array}{c} \phi(\theta) \\ \psi(\theta) \end{array} \right)$. To allow for quicker matching techniques between pairs of curves $\mathbf{v}_i(\theta)$ and $\mathbf{v}_j(\theta)$ the curves are polygonized starting at the value of $\theta$ where $\phi(\theta)$ is maximum. This starting point is chosen to provide one method of insuring some rotational invariance to encoding the curve. Then finally the polygonal curve is encoded into a representation called the 3D super segment (Figure 2.14). The 3D super segment stores the curvature angles between links $\kappa_i$ and the torsion angles $\tau_j$ shown in Figure 2.14 from the polygonization of the curve $\mathbf{v}(\theta)$.

For recognition, the best places to encode splash features are in the areas of high curvature. In these areas the variations between the normal at $P$ and the normals $N\rho(\theta)$ gives splashes a rich structural description of the local surface shape of the object.
Figure 2.14: Orientation coordinates for splash features.
Stein and Medioni [131] employed a ‘Structural Indexing’ approach to matching. This recognition method is a variant on hashing where the indices to the hash table are related to structures formed from the features. In this case the code for the hash table is found from the 3D super segment. The curvature $\kappa_i$ and torsion angles $\tau_j$ between the segments of the polygonal curve along with the number of segments, the maximal orientation difference between point of interest and the contour $\max \phi(\theta)$ and the radius $\rho$ are used to index the hash table. This is termed structural indexing because the table is indexed by structural elements of the features. For a given interest point a entry is encoded for various radii and number of segments in the polygon approximation of the splash. At recognition time scene interest points are determined and used to index the hash table. The matches are used to generate hypotheses. These are first grouped by the model they correspond to. Then for each model a set of geometric consistent hypotheses are formed. The geometric consistency heuristic checks to see if pairs of point matches are approximately the same distance apart and have the same relative orientation differences. The sets of consistent hypotheses then are used to find a the pose for verification. Finally the hypothesized model and pose is verified with the scene data. They have shown that in the best case, where only one object is present in the scene, the complexity can be as low as $O(n)$, but in the worst case where multiple instances of the the object are present in the scene with partial occlusion then the complexity can be as high as $O(n^2 m^3)$, where $n$ is the number of scene feature points and $m$ is the number of models. They have shown through experimentation that the system works fairly quickly and can handle complex scenes with moderate amounts of noise.
Chua and Jarvis [29] formulated a new representation (the point signature), which follows along the same lines as Stein and Medioni’s work [131]. The point signature is different in that it does not encode information about the normals around the points of interest (Figure 2.13); rather it encodes the minimum distances of points on a 3D contour to a reference plane (Figure 2.15). The contour is constructed by intersecting the surface with a sphere centered on $P$ and with a fixed radius $r$. A principal component analysis of the 3D contour defines a plane where the distance from points on the contour to the plane are at a minimum. The normal of the plane can be thought of as approximating the surface normal around the point of interest. The plane is then translated until it contains $P$. A signed distance from the 3D contour forms a 1D parametric curve $d(\theta)$ as shown in Figure 2.15. The final representation is a discretized version $d[n] = d(n \cdot \Delta \theta)$, where $\Delta \theta$ is 15 degrees. This provides a compact way to store information about the structure of the surface around a point that is pose invariant. For their final system they found that it was better to encode two signatures at every point of interest on the object where the two signature were created with spheres of different radii. This improved the selectivity of the matches.

![Diagram of point signature](image)

**Figure 2.15: Point Signature.**

78
To sufficiently cover the object, a discrete parametric curve \( d[n] \) is obtained at every model point. These point signatures are then placed into an index table. Each bin in the table contains a list of model point signatures whose \( \min d[n] \) and \( \max d[n] \) values are similar. During recognition, point signatures are calculated on a grid evenly spaced over the scene. These point signatures are used to index the table and compare with the model signatures contained in the appropriate bin. Signatures whose are similar generate possible model scene correspondences. The hypotheses are grouped by model and models are ordered by the number of hypotheses they received. The models with the most correspondences are verified. The rotation and translation transformation between the scene point and model are found using partial correspondence search [28, 29]. The worst case computational cost of the verification is \( O(N_sN_mH^3) \), where \( N_s \) is the number of scene points in the hypotheses, \( N_m \) is the number of model points in the correspondence, and \( H \) is the maximum number of hypotheses a scene point is mapped to model points (since a scene point can map to more than one model point). The system shows quick recognition times for multi-object scenes from a database of fifteen models with high accuracy.

Johnson and Hebert [73, 70, 72, 69, 75, 76] also employed point features for object recognition. Their ‘spin images’ are 2D histograms of the surface locations around a point. The spin image is generated using the normal to the point and rotating a cutting plane around the point using the normal as the axis of rotation (see Figure 2.16). As the plane spins around the point the intersections between the plane and the surface are used to index a 2D histogram. The bins in the histogram represent the amount of surface (or the number of times) a particular patch of the cutting plane intersects the object. Spin images are generated from models using a polygonal
approximation to the surface. The vertices in the polygonal model are approximately uniformly distributed across the object. When spin images are generated from real data taken from a range scanner, a similar criterion is required. The uniformity is required so the bins in the spin images approximate the surface area cut by the plane.

Figure 2.16: Spin Image.

Johnson and Hebert's [72, 69] spin images are used to establish correspondence between scene and model points for 3D data. During recognition the scene is randomly sampled to find points where spin images are to be found. This is done until 10% of the scene points have been sampled. For each scene spin image is compared with every model spin image. The comparisons produces a similarity measure histogram. The histogram is analyzed to find the upper outliers. These outliers are
used to generate hypothetical matches between a scene point and possibly more than one model point. Once all the scene points have been used to generate possible correspondences, they are again filtered to remove 50% of the worst matches. The correspondences are then grouped by model and checked for geometric consistency. The measure insures consistency in both geometric position and orientation. The measure is weighted by distance between the sets of correspondences. This is used to promote grouping of matches that are geometrically consistent and are separated by some distance. Groups of geometrically consistent matches between model and scene points are used to calculate a rotation and translation transformation between model and scene points. A modified ICP algorithm [12] is used to refine the transformation before a final verification step confirms or rejects the model/scene match.

The recognition scheme can be used to identify partially occluded free-form objects. The system is shown to work on an object where only 20% of the surface is visible. The space requirements of the spin image stack is $O(MI)$ where $M$ is the number of model points (this includes all models in the database) and $I$ is the number of pixels (bins) in the spin images. The computational complexity for establishing correspondences is $O(SMI + SM \log(M))$, where $S$ is the number of randomly sampled scene points, and the term $M \log(M)$ identifies time needed to sort the similarity measures between a scene point and all the model points in the spin image stack. The complexity of grouping the correspondences is not given, but the ICP refinement of the transformation is at worst $O(M)$ for each iteration.

Johnson and Hebert [75] have also adapted the appearance-based techniques of Murase and Nayar [95] for use with the spin images. The images are used to find a low dimensional subspace where scene spin images can be match quickly to model

81
spin images. New appearance-based representation for the discriminatory database of spin images significantly reduced the amount of storage space, while enabling the system to more effectively recognize multiple objects in one scene range image. But their results have shown a slight decrease in the recognition rate when compared with previous recognition scheme [72, 69]. Their testing further shows that the amount of clutter in the scene does not seem to effect the recognition rate, while the amount of occlusion of the object does. It seems that if a sufficient amount (more than 30%) of the object surface is visible in the scene, it has a high probability of being recognized.

Spherical representations for representing 3D surfaces for object recognition and pose localization have a rich history in the vision community [86, 31, 145, 123, 57, 67, 33]. Ikeuchi and Hebert [67] provides an excellent overview of the use of these representations from the early 1980’s to the present. Delingette, Hebert, and Ikeuchi’s recently developed spherical attribute image (SAI) [57, 33] to address many of the shortcomings of the earlier spherical representations (EGI, DEGI, and CEGI [67]). Those problems are: ability to handle non-convex objects (representation has a many to one mapping), and the ability to handle occlusion (partial surface matching).

The SAI representation maps points on an object surface to vertices on a quasi-regular tessellated sphere. By regular it is implied that the triangle faces are similar in surface area and equilateral. Local surface characteristics are stored at the vertices of the tessellated sphere that correspond with the surface point. The surface point to vertex mapping is determined by deforming/shrinking an ellipsoidal version of the same tessellated sphere to the object’s surface. The deformation is guided by forces that try to preserve the regularity, while shrinking the mesh to the surface. This
regularity condition gives the SAI representation rotational invariance and the ability
to be extended to match occluded objects.

Delingette, Hebert, and Ikeuchi [57, 33] point out that the rotational invariance
is only true as the number of nodes in the mesh becomes very large, because the
SAI is discrete and the nodes of the mesh are mapped to the object's surface. They
overcome this problem by averaging SAI values at vertices to approximate values at
any point in between the mesh nodes.

Oclusion is handled by assuming the mesh faces have equal area and using a scale
factor to enlarge or shrink the spherical attribute image (SAI) deformed to a partial
surface. In the generation of the SAI, a closed surface mesh is deformed to the partial
surface; this produces some areas on the SAI where the mesh is interpolating the
space in between surface data from the partial data of the object. These interpolated
regions will not be matched with another SAI. Typically, the SAI generated from only
part of an object surface will not map to the same number of mesh nodes as one fit
to the full object. To allow such spherical attribute images to be matched, one mesh
has to be shrunk or enlarged so that, when matched, they represent the same amount
of area on the objects they model.

Matching two objects reduces to finding the best rotation between scene and
model SAI's. The simplest strategy is to sample the space of all possible rotations of
the sphere and to evaluate the distance measure between SAI's. The rotation that
minimizes the error is a candidate match. For more than one object in the database,
each object SAI must be matched with the scene. To reduce the search space for the
best rotation the geometry of the tessellated sphere is used to limit the number of
possible rotations. The best match can be determined in a few seconds on a modern
workstation. This approach has been extended to include occluded objects. The mesh resulting from a partial view of an object is scaled and the interpolated nodes do not count in the distance measures between SAI. A shape similarity measure has been introduced between SAI so classification of similar object can be used or a hierarchical database. Sets of similar objects can be classified together under a prototype for the class. Similarity is done at different scales of smoothing of the SAI. A multi-scale version of an object’s SAI then can be matched with the prototype to determine if the objects in the class should be matched against the scene SAI. This can reduce the number of times the scene SAI has to be matched against database objects.

In the work summarized above, objects must have the same topology as a sphere. Delingette [31] uses a representation similar to the SAI to represent more complex topologies. Instead of maintaining the relationship with a sphere, parts of an object can be fit using the deformable surface mesh and can be connected to another mesh. The use of this representation for object recognition and localization of pose has not yet been explored.

Shum, Hebert, and Ikeuchi [123] developed a similarity measure for spherical attribute images. The similarity measure is used to show the SAI’s ability to differentiate objects and how the number of nodes in the tessellation effects matching of objects. Zhang and Hebert [145] also use the similarity measure to classify an object’s SAI at different scales. They have also added the notion of smoothing an object’s surface through its SAI representation without shrinkage. This has allowed the matching of objects at different levels of detail.

Barequet and Sharir [6] formulated a novel approach for partial surface and volume matching by finding registration parameters. The registration technique was inspired
by a intensity image technique for matching and pose localization. The method uses a footprint that is invariant to rotation and translation to differentiate objects. The footprints change depending on the application. One example of a footprint is an approximation of surface curvature at a point, which is invariant to pose. Using the footprints, correspondences are established between scene and model points. The list of correspondences are passed through a series of discrete rotations and used to score the quality of the match. This produces a voting table where the entries record the quality of a rotation. The entry in the table with the best score is used as an initial rotation estimate for the object and fed to an iterative refinement algorithm. Once the best rotation has been determined, the translation transformation is found. The complexity of finding the match is \( O(n + k + s^3) \), where \( n \) is number of points in the input sets, \( k \) is the number of correspondences, and \( s \) is the maximum size of the voting table.

Greenspan [55] used a hypothesis-test-verify methodology to avoid the typical feature extraction techniques associated with 3D recognition. The method's objective is to determine all occurrences of a model in the image. This is accomplished by using image seed points as starting points for a sequential hypothesis-and-test algorithm. The seed point is hypothesized to lie on the surface of the model. The set of possible poses that maintain the truth of this hypothesis is large since the point can be any point on the object. To reduce the number of possible poses, successive image points are queried and their hypotheses intersected with the current set until only a small number of hypotheses are left. At each stage the new point is hypothesized to lie on the surface on the model and tested. This is accomplished efficiently by generating a sample tree from the model and using it to guide the choice of points around the seed
point. Since most seed point combinations will not match well to a model, the tree is
designed to be efficient at refuting incorrect the hypotheses. In the implementation
of the system, the image is used to generate a coarse voxel representation of the data.
This is done for two reasons: first it is easy to determine if a point in space is close
to the surface of the object (voxels are labeled as surface, occluded, free, unknown)
and it effectively limits the number of possible model poses. The time complexity to
traverse the sample tree with $n$ leaf nodes for $s$ image seeds is $O(s \log^2 n)$. This is the
time it takes to generate a set of initial pose hypotheses for an object in a scene image.
The hypotheses are checked using template correlation between model and scene data
for a given pose. The surviving hypotheses are further eliminated by looking at the
amount of overlapping surface area and the volume between the surfaces in that area.
Finally the few hypotheses left are sent to an ICP algorithm [12] to refine the pose and
tested again for the overlap area and the volume between the surfaces. The sample
trees took several days to generate on a modern workstation, while recognition was
achieved in about a minute. The system was shown to handle recognition and pose
determination of free-form objects in complex scenes with occlusion.

Complex objects with similar gross shape but different fine shape detail present
new and interesting problems to object recognition practice. One example of this is
the human brain which has similar overall shape but unique convolution detail. For a
certain class of problems, a general shape descriptor would allow studies of common
properties of the object without being adversely affected by individual variations of
single entities. This is in contrast to another class of problems where one might want
to identify differences between an individual and other members in its population.
In the first class of problems the general shape and large structures are important,
while in the second class of problems the individual variations are important. To handle both of these cases, an object must represent both local and global details. Naf et al. [97] proposed using 3D Voronoi skeletons to represent medical volumetric data. A Voronoi skeleton is derived by first finding the 3D Voronoi diagram of the volumetric data then iteratively removing the Voronoi faces that are closest to the object’s boundary and whose removal will not change the topology of the diagram. When no more faces can be removed, what remains is a 3D Voronoi skeleton of the object. At this level, the diagram represents the general shape of the object, compared to the original Voronoi diagram of the data which contains the individual characteristics. The Voronoi skeletons where used to find the thickest part of a hip bone from a 3D radiological scan of the hip and to analyze abnormalities in the temporal lobe of a brain MRI.

Table 2.3 summarizes and compares the various techniques described above. The table includes information about the geometric feature the technique is built upon (e.g. surface normal or curvature, geons, generalized cones), and whether or not the technique can handle object occlusion. The table also includes the largest documented database size and the recognition rate and time complexity of the algorithm. Not all the papers surveyed reported recognition rate or the complexity of the recognition algorithm. In these cases the table is marked with a NR (not reported). In some papers the method was mostly described as a surface or volume matching technique. In these cases often what was described was the ability of the system to accurately match two data sets by reporting the recovered pose vs. known pose. For these papers the largest database size was listed as N/A to indicate the tests were not designed to
<table>
<thead>
<tr>
<th>Technique</th>
<th>Features</th>
<th>Occlusion Local</th>
<th>Largest Database</th>
<th>Recognition Rate</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nevatia et al. [104]</td>
<td>Generalized Cones</td>
<td>Yes</td>
<td>5</td>
<td>NR</td>
<td>NR</td>
</tr>
<tr>
<td>Raja et al. [115]</td>
<td>Geons</td>
<td>No</td>
<td>36 synthetic and 12 real</td>
<td>0.77</td>
<td>NR</td>
</tr>
<tr>
<td>Extremal Mesh [137]</td>
<td>Curvature</td>
<td>No</td>
<td>N/A</td>
<td>NR</td>
<td>NR</td>
</tr>
<tr>
<td>COSMOS [36]</td>
<td>Curvature shape spectra</td>
<td>No</td>
<td>20 synthetic and 10 real</td>
<td>0.97</td>
<td>NR</td>
</tr>
<tr>
<td>Tripod [111]</td>
<td>Crease Angle</td>
<td>Yes</td>
<td>4</td>
<td>0.92</td>
<td>NR</td>
</tr>
<tr>
<td>Splash [131]</td>
<td>Structural indexing</td>
<td>Yes</td>
<td>9</td>
<td>NR</td>
<td>$O(n)$-$O(n^2m^r)$</td>
</tr>
<tr>
<td>Point Signature [29]</td>
<td>Distance</td>
<td>Yes</td>
<td>15</td>
<td>1.0</td>
<td>Worst Case $O(N_sN_mH^3)$</td>
</tr>
<tr>
<td>Spin Image [76]</td>
<td>Surface histogram</td>
<td>Yes</td>
<td>4</td>
<td>1.0</td>
<td>$O(k\log_2(n))$</td>
</tr>
<tr>
<td>SAI [57]</td>
<td>Angle</td>
<td>Yes</td>
<td>N/A</td>
<td>NR</td>
<td>NR</td>
</tr>
<tr>
<td>Barequet et al. [6]</td>
<td>Curvature</td>
<td>Yes</td>
<td>N/A</td>
<td>NR</td>
<td>$O(n + k + s^3)$</td>
</tr>
<tr>
<td>Greenspan [55]</td>
<td>Sample Tree</td>
<td>Yes</td>
<td>5</td>
<td>NR</td>
<td>$O(s \log^2(n))$</td>
</tr>
<tr>
<td>Naf et al. [97]</td>
<td>Voronoi skeleton</td>
<td>No</td>
<td>N/A</td>
<td>NR</td>
<td>NR</td>
</tr>
</tbody>
</table>

Table 2.3: Summary of recognition techniques for range data.
discriminate between objects, but rather to recover correct pose of a known object in the scene.

2.6 Emerging Themes, Conclusions, and Commentary

The focus of this survey has been the construction and recognition of three-dimensional object models, with primary emphasis on range data but with some mention of intensity techniques as context. Toward that end, modeling and recognition are both surveyed in detail.

Improvements in modeling technique will increase the accuracy and speed of reverse engineering complex 3D models from examples. The quality of these models presently depends on the skill of the person monitoring and manipulating the software packages used to form the model. The designer must currently be constantly on the watch for erroneous data (errors in measurement by the sensor), errors in the registration process, and the integrity of the mesh after integration. Each one of these tasks may and often does fail when there are errors made in the measurement of the object and/or the position and amount of overlap between surfaces. The methods developed for model building have only been tested on small sets of objects. This is in part due to the large effort required to train an algorithm to work on a set of objects. Often the heuristics used to produce good results on one model will fail on another. In the future the methods for registration, integration and optimization of polygonal meshes need to be tested on a large number of standardized objects with varying geometry so the strengths and weakness can be cataloged for each technique. The research community would benefit from an “Open Source” architecture and implementation of the standard techniques used in range image registration and integration. This
would be a valuable resource that allows the community to build on the knowledge of the previous researchers without wasting time and effort in rediscovering what has already been done.

The object recognition problem involves a study of salient features and their identification, as well as a study of control and data structures that yield efficient recognition techniques. The problem of finding and identifying objects in single object scenes with no occlusion has been well studied and many systems designed that show good results [4, 15, 36, 67, 95, 98, 142]. In these systems a fair amount of information is present about the object. This allows for the design of systems which are able to identify objects under noisy conditions except in cases where views of the objects are too similar. In these cases noise can dominate the differences between the very similar object views and cause the reliability of the recognizer to decrease. In the future these ambiguous views need to be determined and documented for each model and system so that a designer can determine whether these difficult views are significant for an application. This is especially important as more and more objects are added to a system and the chances of similarity increase.

The problem of finding and identifying multiple objects in scenes with the possibility of occlusion and background clutter is a much harder problem [14, 24, 29, 55, 57, 75, 84, 107, 131]. In general, the partial information about an objects makes the recognition less reliable and more complex because the information can be incomplete and disjoint. This is in part due to the decreased saliency of local measures over global measures. Locally many regions of a surface or many regions of surfaces on different objects may appear similar. Using the information in these regions by themselves is a poor choice for recovering the identity and location, but together by using
the relationships between features the identity and location of an object may still be obtained. In the future the measures of saliency for features in the illuminance, color, and range image domains need to be qualified in terms of their sensitivity to noise so that their discriminatory properties can be more clearly specified and compared.

For both single object and multiple object scenes large database studies need to be applied. As the size of the database increases the importance of a systems method for quickly and accurately indexing to the correct model become more important. Large multimedia databases and flexible manufacturing inspection and assembly systems are examples of applications where quick indexing of image/model databases are becoming important.

An emerging area of study for new object recognition systems is to combine multiple imaging modalities to determine object identity and location. One example is to use both depth (range) and color information from many modern 3D digitizers common to the computer graphics and computer vision fields. The textural information from the color images may allow for discrimination between objects in cases where their shape is similar while their texture is different, likewise cases where the textural information does not discriminate between objects well, their shapes may.

Another area likely to increase in importance is deformable object modeling and understanding. Techniques to deal with non-rigid object matching and motion are an important field especially in the areas of medical imaging. Here the individuality of a patient may cause ridge techniques problems because the surface geometry of an organ or other parts varies from person to person. Even in the case of maintaining a health history of a single patient the parts can change due to swelling or aging.
Therefore what may be more interesting for non-rigid objects is to study their defining characteristics so that time-dependent shapes may be represented and recovered faithfully.
CHAPTER 3

Object Recognition Using Global Features

3.1 Introduction

A free-form object’s surface characteristics are often hard to quantify into a concise descriptive representation useful for object recognition [13]. The techniques surveyed in Chapter 2 used differential geometry (e.g., [36]), part decomposition (e.g., [104]), and point centered surface descriptors (e.g., [72],[29]) as features for model-based 3D object recognition. Each of these types of descriptive representations has its advantages and disadvantages, namely in the saliency of the descriptors, the types of objects and environments employed, and the methods used to search through the descriptions to identify the correct match. A common problem with most of these techniques is the scalability of the recognition process to large databases (100 or more objects). Examination of Table 2.3 (page 88) indicates that the largest database for synthetic and real experimentation with free-form object recognition from range images is 36 and 15 models respectively. When this is compared this to a 100 object database used for object recognition from intensity image (Table 2.2, page 66), it is noted that the scalability and saliency issues have been approached and studied for larger databases in the context of intensity data. In particular, appearance-based
object recognition has shown great success in recognizing objects using an image-based training technique to build a large combined database of objects (Nayar et al. [95, 98]). One failing of these intensity image techniques is they rely heavily on the reflectance of the surfaces of the object rather than on the object shape alone. This may become a problem especially when the concept is applied in a manufacturing environment where parts may be similar in surface color.

In this chapter we will introduce a technique that applies appearance-based recognition to shape instead of reflectance of the object surfaces. The work was inspired by the work of Nayar and colleagues discussed in Section 2.5.1. The differences between this work and that of Nayar’s team are as follows.

1. We are applying the technique to a different problem domain (range images) studying the effect of surface shape instead of reflectance.

2. We have increased the dimension of possible rotational pose variations of the object’s position in the scanning environment from a 1D rotation (e.g., rotation on a turntable) to a full 3D rotation of the object (this can be important for sculpted objects whose pose maybe unstable when the object is placed on a flat surface prior to scanning).

3. Our system uses CAD models and synthetic image generation techniques to generate the large amount of training data required to build the recognition database using appearance-based techniques instead of using a labor intensive image acquisition and training procedure with a real camera. Our system could be trained on real range image data as well, providing a superset of the capabilities of the prior work.
More detail on appearance-based recognition techniques and relevant background material can be obtained in Section 2.5.1 of this dissertation, Trucco and Verri’s discussion of appearance-based recognition [139], and the seminal papers by Turk and Pentland [142], Nayar’s group at Columbia [95, 102] and Murakami and Kumar [93]. A brief review appears in Section 3.2.

This chapter is organized as follows. Section 3.2 presents background material and notation used in our solution to the object recognition problem. Section 3.3 describes the range image capturing environment that will be used for experimentation. Section 3.4 describes our technique for training an appearance-based recognition technique to identify objects in a range image. Section 3.5 describes the properties of object manifolds in our appearance space. Section 3.6 discusses the steps used to identify objects. Section 3.7 describes early results from experimentation on a large synthetic model database. Section 3.9 describes recent and more comprehensive experiments using both synthetic and real range data. Section 3.10 summarizes the contribution of this chapter.

### 3.2 Background Material and Notation

Appearance-based techniques are a popular class of view-centered recognition methods. The technique relies on gathering information about object’s appearance (reflectance) under all allowable viewing conditions (e.g., pose, illumination, articulation). Depending on the dimension of these appearance variation parameters and the resolution with which they are sampled, the number of training images required can be extremely large. Critical to appearance-based recognition techniques is an efficient
way to compare an observed image \( S \) with large sets of training data \( (X) \) in order to
determine which training view best matches the observed image.

Image correlation (or template matching) can be used as a simple way to compare
diplies of images \( (I_1, I_2) \) or image regions. Assume that \( I_1 \) and \( I_2 \) are both \( N \times M \)
images; their correlation is defined as the sum of the pixel products:

\[
C(I_1, I_2) = \sum_{i=1}^{N} \sum_{j=1}^{M} I_1(i,j)I_2(i,j). \tag{3.1}
\]

In practice, the images \( (I_1, I_2) \) are assumed to be normalized to unit energy:

\[
\sum_{i=1}^{N} \sum_{j=1}^{M} I_1(i,j)^2 = 1 \quad \text{and} \quad \sum_{i=1}^{N} \sum_{j=1}^{M} I_2(i,j)^2 = 1. \tag{3.2}
\]

With these sets of assumptions the value of the correlation \( C(I_i, I_j) \) lies in \([0, 1]\) if
both \( I_i \) and \( I_j \) lie in \( \mathbb{Z}^{N \times M} \); a value of one corresponds to a perfect match and zero is
the worst match (\( I_i \) and \( I_2 \) are uncorrelated). Similarly, if \( (I_i \) and \( I_j \) lie in \( \mathbb{R}^{N \times M} \), the
image correlation \( C(I_i, I_j) \) lies between -1 and 1, where +1 corresponds to a perfect
match and -1 is the worst match (perfect anticorrelation).

Image correlation can be rewritten in terms of a vector dot product by unraveling
the 2D images \( I_k \) into 1D vectors

\[
\mathbf{x}_k = [I_k(1,1), I_k(1,2), \ldots, I_k(1,N), I_k(2,1), \ldots, I_k(M,N)]^T.
\]

The correlation of \( \mathbf{x}_i \) and \( \mathbf{y}_j \) then becomes

\[
C(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j. \tag{3.3}
\]

The correlation between two vectors is related to the Euclidean distance between two
images by the relationship:

\[
\| \mathbf{x}_i - \mathbf{x}_j \|^2 = 2(1 - \| \mathbf{x}_i^T \mathbf{x}_j \|). \tag{3.4}
\]
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbf{I}$</td>
<td>an image (e.g., $\mathbf{I}_1, \mathbf{I}_j$, etc.)</td>
</tr>
<tr>
<td>$N$</td>
<td>number of rows in $\mathbf{I}$</td>
</tr>
<tr>
<td>$M$</td>
<td>number of columns in $\mathbf{I}$</td>
</tr>
<tr>
<td>$N \cdot M$</td>
<td>number of pixels in $\mathbf{I}$</td>
</tr>
<tr>
<td>$\mathcal{I}$</td>
<td>the image subspace</td>
</tr>
<tr>
<td>$\mathbf{\tilde{x}}$</td>
<td>1D vector containing all pixels in $\mathbf{I}$</td>
</tr>
<tr>
<td>$\mathbf{X}$</td>
<td>matrix containing all training images</td>
</tr>
<tr>
<td>$M$</td>
<td>the number of training images in $\mathbf{X}$ (the number of columns)</td>
</tr>
<tr>
<td>$\mathbf{\bar{c}}$</td>
<td>the mean image of $\mathbf{X}$ ($\mathbf{\bar{c}} = \frac{1}{M} \sum_{i=1}^{M} \mathbf{\tilde{x}}_i$)</td>
</tr>
<tr>
<td>$\Sigma$</td>
<td>Covariance matrix of all training images in $\mathbf{X}$</td>
</tr>
<tr>
<td>$L$</td>
<td>Rank of $\Sigma$ ($L \leq \min(M, N)$)</td>
</tr>
<tr>
<td>$\mathcal{U}$</td>
<td>the eigenspace defined by the Karhunen-Loève expansion</td>
</tr>
<tr>
<td>$K$</td>
<td>the truncation dimension of subspace $\mathcal{U}$ used for approximating correlation</td>
</tr>
<tr>
<td>$\mathbf{\bar{g}}$</td>
<td>$K$ dimensional prototypes formed from projecting ($\mathbf{\tilde{x}} - \mathbf{\bar{c}}$) into $\mathcal{U}$</td>
</tr>
</tbody>
</table>

Table 3.1: Notation used in this chapter.

Hence, minimizing the Euclidean distance between two images is equivalent to maximizing the image correlation between the same pair of images.

Let $\mathbf{X} = \{\mathbf{\tilde{x}}_1, \ldots, \mathbf{\tilde{x}}_M\}$ be a set of $M$ training images taken of objects we desire to identify and assume $\mathbf{s}$ is an available image containing an object whose identity is unknown. A simple method to hypothesize the identity of the object in $\mathbf{s}$ is to use image correlation as an indicator of the similarity between the unknown image $\mathbf{s}$ and each training image $\mathbf{\tilde{x}}_i$. The problem with this approach is that the time required to calculate the correlation between $\mathbf{s}$ and every training image in $\mathbf{X}$ can be quite large (the complexity of a naive nearest neighbor match process is in $O(NM)$). Nearest neighbor search $\max_i (C(\mathbf{\tilde{x}}_i, \mathbf{s})$ in this case identifies the training vector $\mathbf{\tilde{x}}_i \in \mathbf{X}$ with the maximum correlation with $\mathbf{s}$, the image containing the unknown object.
In order to make an object recognition system based on image correlation practical for large numbers of objects and/or training views, the time required to search the training data must be reduced. Appearance-based recognition techniques accomplish this by approximating image correlation in a lower dimensional subspace \( \mathcal{U} \) of the image space \( \mathcal{I} \) and by utilizing more efficient search techniques [102].

The subspace \( \mathcal{U} \) is built by applying a Karhunen-Loève transformation [50, 93] to the training set of images \( \mathbf{X} \). The details of the Karhunen-Loève expansion are given in Appendix A, and summary notation appears in Table 3.1. The sample covariance matrix \( \hat{\Sigma} \) (constructed from the training data \( \mathbf{X} \)) has eigenvectors and associated eigenvalues \( \{ (\mathbf{e}_i, \lambda_i), i = 1 \ldots L \} \), assumed sorted in descending order of eigenvalue magnitude. The eigenvectors span a subspace \( \mathcal{U} \) of rank \( L = \min(N \cdot M, N_I) \) of the original image measurement space, where \( N_I \) is the number of columns in \( \mathbf{X} \) (i.e., the number of images used in training). The training data \( \mathbf{X} \) can then be projected along the eigenvectors into \( \mathcal{U} \) forming \( L \) dimensional prototypes
\[
\mathbf{g}_i = \begin{pmatrix} \mathbf{e}_i^T \\ \vdots \\ \mathbf{e}_L^T \end{pmatrix} (\mathbf{x}_i - \bar{\mathbf{c}}).
\]
of all the training data. Similarly
\[
\mathbf{x}_i = ([\mathbf{e}_1, \ldots, \mathbf{e}_L]) + \bar{\mathbf{c}}
\]
can be used to reconstruct the image training vectors.

The Karhunen-Loève expansion has two very useful properties:

1. The prototypes from any two training vectors \( (\mathbf{x}_i, \mathbf{x}_j), i \neq j \) are decorrelated:
\[
\mathbf{g}_i^T \mathbf{g}_j = 0.
\]
2. If the eigenvectors associated with the $K$ largest eigenvalues ($K < L$) are used to form the subspace $\mathcal{U}$ then the reconstruction from the truncated prototypes $\tilde{\mathbf{g}}_i$ minimizes the mean squared error between the training data and the reconstruction ($\|\mathbf{x}_i - \tilde{x}\|$, $\tilde{x} = ([\mathbf{e}_1, \ldots, \mathbf{e}_K]) + \mathbf{c}$) with respect to the number of dimensions ($K$) used to approximate the training data.

It has also been shown [95, 102, 139] that the Euclidean distance between two images is equivalent to the Euclidean distance between two prototypes:

$$
\|\mathbf{x}_i - \mathbf{x}_j\|^2 = \|\tilde{\mathbf{g}}_i - \tilde{\mathbf{g}}_j\|^2
$$

, where the dimension of the prototypes is $L$. When only the coordinates associated with the $K$ largest eigenvalues ($\lambda_1 < \cdots < \lambda_K$) are used, the Euclidean distance between two prototypes only approximates the distance between the two corresponding images. By utilizing the relationship in Equation 3.4, the Euclidean distance between the prototypes can be used to calculate the image correlation between the training images

$$
\|\tilde{\mathbf{g}}_i - \tilde{\mathbf{g}}_j\|^2 = 2(1 - \|\mathbf{x}_i^T \cdot \mathbf{x}_j\|). \quad (3.5)
$$

One advantage of appearance-based techniques is that the calculation of $\|\tilde{\mathbf{g}}_i - \tilde{\mathbf{g}}_j\|^2$ is less expensive ($O(K)$) than the calculation of $2(1 - \|\mathbf{x}_i^T \cdot \mathbf{x}_j\|)$ directly ($O(N \cdot M)$). Often a projection dimension of $K \approx 20$ is sufficient [95] compared to the rather large dimensionality of the original image space (e.g., $N \cdot M = 16384$ for $128 \times 128$ images). Doing a single correlation calculation this way does not necessarily pay off due to the overhead cost of projecting the data into the subspace $\mathcal{U}$ ($O(K \cdot N \cdot M)$). But if correlation is used in conjunction with nearest neighbor search, the overhead of projecting the data pays off due to the large number of image correlations that will
be calculated using the single projected prototype. Several well known improvements
(like the k-d tree [120]) have been used to reduce the computational complexity of
nearest neighbor search from $O(KM)$ to $O(K\log_2(M))$. One such new technique,
developed by Nene and Nayar [102]) to specifically address nearest neighbor search
for appearance-based object recognition, has been shown to perform slightly better
than k-d trees (for $K \approx 20$) while reducing the complexity of structuring data to
obtain the efficient search.

A strong link exists between appearance-based recognition and image compression,
and some discussion of this link is appropriate. In image compression, the goal is to
reduce the number of bits required to store or transmit an image. To accomplish, this
most compression techniques take advantage of an image’s inherent self similarity,
as measured by the amount of correlation between neighboring pixels [138]. This
redundancy can be exploited to reduce the number of bits needed to represent the
image. Furthermore, in an image sequence the redundancy present between pairs of
images adds an additional realm of exploitation.

One way to reduce redundancy between the elements of any set of data is to use
the Karhunen-Loève expansion to de-correlate the data. The expansion forms the
smallest subspace with respect to the number of dimensions where all the data is
de-correlated. If loss can be tolerated, the expansion can be truncated to further
reduce the number of dimensions. The smaller the number of dimensions used to
describe the data, the fewer numbers needed to represent the data. In appearance-
based recognition, by contrast, the goal may not be the smallest number of bits,
but efficient search times. The fewer numbers used to represent the data in nearest
neighbor search, the faster the search times using spatial search structures like k-d
trees [120] or high dimensional search techniques [102].

3.3 Scanning and Modeling Environment

The scanner setup shown in Figure 1.1 illustrates the environment in which
range images are captured. Included in the figure is the location of the scanner
with respect to the object and the scanner’s coordinate systems. The origin of the
scanner’s coordinate system is located on the color CCD camera’s optical axis and
near the front of the camera optics (its exact position is unknown). The range image

\[ \mathbf{R} = \{(f_{ij}, x_{ij}, y_{ij}, z_{ij}), i = 1, \ldots M, j = 1, \ldots N\} \]

has four scalar measurements per pixel: a flag \( f_{ij} \) which is FALSE if the image measurement was not available due
to shadowing and TRUE otherwise, and (conditionally) a 3D point measurement
\((x_{ij}, y_{ij}, z_{ij})\) in the sensor’s coordinate system. The \( x \) coordinate is a linear function
of the pixel’s column index in the raster; likewise, the \( y \) coordinate is a linear function
of the row number. In this environment, the objects are placed on turntable without
any special fixturing or calibration to precisely position the object before each scan.

The lack of precise positioning control is significant considering image correlation
calculations are not shift invariant. It is well known that correlation based image
matching is sensitive to 2D translation of either image template. The sensitivity ex-
ists in all three dimensions in the system described here. Past work with intensity
data has employed normalization techniques such as alignment within 2D bounding
boxes to remedy the problem. In our work, a simple foreground-background seg-
mentation enables alignment in the image plane (Figure 3.1). This transformation
of the data allows the system to handle object translation in the image plane but

101
induces sensitivity to object segmentation. Segmentation is a well known problem for appearance-based techniques [95, 15, 142] and solutions have been proposed in [15, 107, 84, 95, 14, 116] which typically involve adding a search or adaptive method to find a solution. We will not specifically address this problem and solutions in this paper other than introducing a more intelligent choice for the background pixel value (Section 3.3.1).

Figure 3.1: Centering an object footprint to minimize the effects of image plane translation.

Alignment in the z (depth) coordinate was examined more selectively. Specifically, we developed three approaches to normalizing the z coordinate of the object’s depth distribution. In order to handle different z measurements for the same object points (e.g., an object moved closer to the scanner between scans) the system will translate all object pixels so that a datum z value is set equal to a prespecified constant $Z_{obj}$. Three choices for the datum z value were explored:

- $z_{min}$, the z-coordinate of the range pixel closest to the range scanner origin;
• \(z_{\text{average}}\), the average \(z\)-value for range pixels within the object; and

• \(z_{\text{median}}\), the median \(z\)-value of range pixels within the object.

These statistics are illustrated in Figure 3.2. They were chosen for their ease in calculation and for their different noise sensitivity properties. Once the statistic is recovered the object’s pixels are translated in \(z\) so that the datum becomes equal to \(Z_{\text{obj}}\). The closest point \(z_{\text{min}}\) utilizes only the nearest point to the scanner and is therefore sensitive to any noise that may be present in that single sample. The average depth \(z_{\text{average}}\) utilizes more depth values and is a better estimate when the noise present in the range image is Gaussian. The median estimate \(z_{\text{median}}\) is better suited for more impulsive noise. By translating images so their statistic now lies on a common point the effect of the translation in \(z\) is removed.

Figure 3.2: Segmentation statistics used to remove the effect of translation in \(z\)
Before the two images can be compared (using image correlation) the scale (or image plane resolution) of the images must also be made consistent. Inconsistencies result because the objects scanned vary in size, the distance between scanner and object changes, and the range sensor has a zoom setting to change the density of samples. To handle these scale changes, we re-sample the segmented object region at a prespecified resolution (Figure 3.3). This is done by rendering the segmented object in a virtual image plane of the specified resolution. Since the range image is easily converted to a polygonal mesh, standard polygon Z-buffering rendering techniques are easily adapted to range synthesis (we employ routines in the OpenGL library [100] (Section 3.3.1).

![Figure 3.3: Using the segment’s bounding box to redefine the object’s sampling density for a given image resolution](image)

Range imaging sensors provide reliable methods for direct capture of object geometry, but they can require an experienced operator to monitor the image capture process in order to produce accurate measurements of the objects and their surfaces.
The nuances of each range sensor depend on the technique used to recover the measurements. Examples of the possible measurement errors are discussed in Section 1.1.

In addition to problems obtaining accurate range data, the objects scanned in this thesis are free-form. These types of objects many have many stable poses where objects in previous appearance-based recognition techniques typically only had a few such poses. This requires extensive training on 3D rotational pose variations (Section 3.4 contains more details of this training process). The potential need to generate vast amounts of training data and the speed and problems in gathering that data using our scanning system led us to develop synthetic range image generation techniques to train appearance-based range image recognition system from an existing solid model (specified as polygonal meshes).

3.3.1 Synthetic Image Generation

Range scanners have been used by visual effect artists and industrial designers to generate realistic computer models of physical objects. These geometric models can be used to generate realistic visual effects simulating reality. In our case the range scanner allows us to build accurate CAD models (represented as a polygonal mesh) of the objects we wish to recognize. In Section 2.4, we covered in more detail the topics of building 3D solid models from range data.

As noted in Section 2.3.1, a polygonal mesh can be defined by a pair of ordered lists:

\[ \mathcal{O} = \langle \mathcal{P}, \mathcal{V} \rangle, \]
where $\mathcal{V} = \{v_1, \ldots v_{N_v}\}$ is a list of $N_v$ three-dimensional vertices $v_i = [x_i y_i z_i]^T$, and $\mathcal{P} = \{p_1, \ldots p_{N_p}\}$ is a list of polygons, each specified as a list of vertex indices: $p_i = \{v_{i,1}, \ldots v_{i,nv_i}\}$.

One way to compare the range image geometry with a stored CAD model of the object is to view both objects as polygonal meshes. Denote the polygonal mesh of the segmented object in the range image as $\mathcal{R}$ and the polygonal mesh of the object’s CAD model as $\mathcal{O}$. Figures 3.4 (a) and (b) show a flat shaded visualizations of those meshes for a view and a model of a toy lamb. In the ideal case, the range image polygonal mesh is a subset of the CAD model’s polygonal mesh ($\mathcal{R} \subset \mathcal{O}$), but more realistically the range image and the polygonal mesh will sample the object’s surface at different locations and with different sampling densities. Still, both representations should be good approximations the true surfaces of the object. Since our recognition technique involves sampling models at resolutions comparable to those in the range image and the underlying geometry is not identical between image and model, a correlation of unity is unlikely to be attained. Since this effect is likely to be negligible provided care is taken during the modeling process, this slight reduction in matrych quality is ignored henceforth.

In order to use polygonal models in an image based recognition system (as reviewed in Section 3.2, the image templates generated from the model and range image must be similar to allow image correlation to return the expected results and match corresponding views. To do this we use an OpenGL [100] based rendering system to generate $2^\frac{1}{2}D$ image templates from both the range image mesh and model mesh. To ensure that templates obtained from images and models are similar in 2D footprint and location in 3D, we use the techniques described in the previous section.
1. A common image template resolution is defined for rendering of model meshes \( O \) and resampling of input range image meshes \( R \).

2. The range image mesh and object mesh are centered using bounding rectangles for the \( x \) and \( y \) coordinates, and one of the three statistical datums \( z_{\text{min}} \), \( z_{\text{average}} \), or \( z_{\text{max}} \) as noted above. All \( x \) and \( y \) coordinates are translated by the middle of their bounding interval values (hence the midpoint of the bounding rectangle becomes the origin) and the depth values are translated to make the reference datum value \( Z_{\text{obj}} \). This effectively removes the relative translations between templates taken of the same object and view.

3. The transformation that takes vertices in \( R \) or \( O \) from the scanner’s coordinate system to the rendering coordinate system is obtained. This transformation takes each 3D vertex and transforms it into a pixel location and depth value (for hidden surface removal). To define this transformation we use an orthographic projection of the model or range image mesh. Once the rendering volume has been set OpenGL calculates the orthographic projection. This projection takes the vertices given in scanner coordinates to screen coordinates.

4. Finally, the scene is rendered, yielding a \( 2\frac{1}{2}D \) range image template \( T \) from the depth buffer of the renderer.

Ideally, the above method would work well for synthesizing range image templates similar to those produced from real range data, but as shown in Figure 3.4, the range image mesh \( R \) often contains holes and eroded boundaries. These artifacts are a byproduct of the sensing technology. Recall that the Minolta sensor used to capture the range image is based upon the principle of triangulation via an active
illumination source and CCD camera (Figure 1.1). Since the illumination source and capture device are not co-located, there can be locations on the surface of the object visible to the camera but occluded from the illumination source or vice versa [44]. In these self-shadowing regions, range data can not be recovered. An example of the effects of self-shadowing can be seen in Figure 3.4(a) under the chin of the lamb. Dark object colors can absorb active illumination to a point where range cannot be calculated (for example, in the eyes of the lamb object in Figure 3.4). Geometry is not present at such locations, yielding a hole in the mesh. To avoid this, we could paint the object but that would prevent us from capturing the original color texture of the model and using this additional information in future research. In addition to the object’s color affecting the recovery of depth values, depth near the image plane silhouette of the object is often not sensed reliably, resulting in an eroded mesh. The above sensing errors yield a range image that is missing regions of data when compared with the model.

So far in the discussion, we have avoided specifying the value of background pixels in the generated image templates \( T \) (i.e., pixels which do not lie within valid geometry). The \( z \) value of this background pixel becomes important when the range image mesh \( R \) contain holes or its boundary is eroded; in such cases a valid model template pixel will be compared to an invalid image template pixel. To reduce the effect of these differences on image correlation calculations, we modify the generation of the templates \( T \) from range \( R \) and model \( O \) meshes. The image differences noted above appear to the system as segmentation errors. It is well documented that segmentation errors can cause problems with appearance-based recognition systems because of the large differences between the object’s boundary and the background value [95]. To
minimize the effects of these missing pixels, background pixel depth values are set to the sample mean of the valid pixels' z coordinates. This will minimize correlation bias due to comparisons between valid and invalid pixels.

In addition to setting the background pixel value to minimize the error accrued when missing data, more realistic synthetic range rendering techniques could be used [44]. These techniques check to make sure the surface point represented by the synthetic range pixel are visible from both the camera and the active light source. At present, we are not using a more realistic rendering technique because the simpler technique described above has proven to be effective in minimizing the error between range T_R and object T_O templates.

In intensity image appearance-based recognition systems [95], the background pixel values are set to zero. Hence, when two images \( \bar{x}_i \) and \( \bar{x}_j \) are compared using correlation or image differencing, the overlapping background pixels do not contribute to either measure. Also, during the normalization procedure (Equation 3.2), the background pixels do not contribute to the computation of the image energy, so only the energy contained in the object is normalized. In our technique, however, we have changed the image's background pixel depth value to a nonzero number. If these images are then normalized and compared, the contributions of overlapping background pixels could bias the correlation measure. To remove this artifact, the entire set of training images is translated so that the background pixel depth value becomes zero. This is done by subtracting every pixel in the training set by the background pixel depth value.

Figure 3.5 contains a summary of the steps applied to the real range data and synthetic data to generate image templates \( T \). The box labeled "Set Background
Pixel Value” refers both to setting all the background pixels’ depths to the expected object training pixel value and to translating the entire image by that value so the new background pixel value is zero.

3.4 Training The Database

The range image templates $\mathbf{X} = \{\mathbf{T}_1, \ldots, \mathbf{T}_M\}$ used to train our database were synthetically generated from polyhedral mesh models $\mathcal{O}$ of objects found in the WSU/OSU 3D database $^4$. A set of rigid rotations was applied to the canonical model to obtain coverage of the 2D pose space (coverage of rotation about the ‘optical’ axis of the range sensor is not necessary in this synthesis step, as discussed below). The output of this step, for an input model $j$, is a set of $m_j$ training views $\{\mathbf{T}_i^j, i = 1 \ldots m_j\}$.

<table>
<thead>
<tr>
<th>Subdivision frequency</th>
<th>Number of viewpoints</th>
<th>Angle between neighboring viewpoints</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12</td>
<td>64.3°</td>
</tr>
<tr>
<td>2</td>
<td>42</td>
<td>32.1°</td>
</tr>
<tr>
<td>3</td>
<td>92</td>
<td>21.4°</td>
</tr>
<tr>
<td>4</td>
<td>162</td>
<td>16.1°</td>
</tr>
<tr>
<td>5</td>
<td>252</td>
<td>12.8°</td>
</tr>
<tr>
<td>6</td>
<td>362</td>
<td>10.7°</td>
</tr>
</tbody>
</table>

Table 3.2: Icosahedron Subdivision Parameters. The angle between neighboring viewpoints for the specified subdivision frequency gives a measure of the density of the sampling with respect to the degree of separation between viewpoints.

$^4$http://sampl.eng.ohio-state.edu/~sampl/data/3DDB
Figure 3.4: Image and model geometry for a lamb-shaped toy. (a): range image. (b): model. (c): mesh support for a single range image in an area on the nose. (d): mesh support for the model in the same area.
Figure 3.5: Summary of the processing steps performed in the generation of the image templates $\mathbf{T}$.

Figure 3.6: Discrete Viewsphere and canonicalizing transformation
Most work in appearance-based recognition [139, 142, 95, 102] has assumed a one-dimensional pose space, typically allowing objects to be rotated on a turntable in the sensor’s field of view. We tessellated the surface of the viewsphere to identify view angles with reasonably even angular spacing. The vertices (normalized to lie on the unit sphere) of the $l$-frequency subdivision of the icosahedron were used to define viewpoints around the object in its canonical position. Table 3.2 enumerates the number of viewpoints and the minimum angle between viewing directions for a range of values of the subdivision parameter $l$ and Figure 3.6 shows the discrete viewsphere for $l = 2$. The choice of $l$ allows the number of viewpoints (hence the angular spacing between viewpoints) to be tuned by the user. A training image $\mathbf{T}_i^j$ was obtained for each of these pose coordinates.

The rotation of an object to an arbitrary viewpoint (as accomplished above) will fix two degrees of freedom in its rotational pose. There still is an additional degree of freedom left in the rotation of the object about the viewpoint (i.e., rotation about the optical axis of the sensor). Rather than generate a set of views for each such rotation, we developed two canonicalizing transformations (rotations in the image plane) to align the axis of elongation with the $x$-axis of the image. The first method uses the 2D ‘footprint’ of each view (as shown in Figure 3.6) to find the elongation direction for the footprint, and then applies a planar rotation to the data to align the direction of elongation with the $x$-axis. In some cases the 2D ‘footprint’ does not contain a unique (or dominant) direction of elongation. In these cases, a principal components analysis [77] to the 3D range data yields a dominant direction of elongation of the 3D point cloud that represents the object in the range image. This direction then can be projected on the the image plane and the image rotated until it is aligned with
the $x$-axis. Either of these transformations produces a canonical training image for that viewpoint. Since zenith and nadir views of the object will produce mirror image footprints (or point clouds), we actually generate two training views per viewpoint, one a mirror image of the other. Both canonicalizing transformations will be tested in experiments to compare their ability to produce characteristic training views at each viewpoint.

The number of viewpoints for a particular experiment is denoted $N_{vp}$. The value of $l$, hence the value of $N_{vp}$, is fixed on an experiment-by-experiment basis, and is used to generate training views \( \{ \tilde{T}_{i}^{j} : i = 1, \ldots, 2 \cdot N_{vp} \} \) for each model $j$.

### 3.5 Object Manifolds

In Murase and Nayar's [95] initial work, bicubic interpolation was used to define a parametric bivariate manifold where the parametric variables correspond to the pose orientation and the lighting position. They used a set of discrete object orientations and lighting positions to define a set of prototypes in the eigenspace, and then interpolated between the prototypes to obtain the bivariate manifold. The bivariate manifold then provides an arbitrarily dense distribution of prototype points for their recognition scheme that uses a modified nearest neighbor search. Since we have the luxury of generating any synthetic view it seemed more appropriate to use more training views to fill in the distribution of the prototype points rather than using interpolation. This view planning idea is developed in Section 3.8.

The manifolds for shape recognition in the eigenspace $\mathcal{E}$ can be parameterized by the pose and location. The variation of these parameters changes the shape appearance of the object to the scanning device. In this work we will only consider the
parameterization of an object shape appearance with respect to the rotational pose of the object. For single object scenes it is reasonable to assume that a reliable translation of the object to a specified coordinate can be found for every possible rotational pose. This can be accomplished by locating the bounding box of the object in the scene using simple background subtraction techniques. The bounding boxes center can then be translated to a specified coordinate using the same methods described in Section 3.3.1.

The shape appearance of a general 3D object changes as the object is rotated in front of the observer. In the subspace $E$, the shape appearance forms a manifold whose surface can be parametrized by the three degrees of rotational freedom in the pose of the object. To reduce some of the complexity of the manifold as well as decrease the number of training views taken for each object, only two of the three rotational degrees of freedom are varied. The object is studied by varying the viewpoint from which the object is observed. As noted above, the additional degree of freedom is handled by using a canonicalizing transformation to align a unique elongation axis of the object with the $x$ axis. Both 2D and 3D alignment transformations were explored and the particular technique in use will be specified where appropriate. For each viewpoint $v_i$ and type of alignment strategy (2D or 3D), there exist two mirror canonicalizing transformations $R_{1i}$ and $R_{2i}$, differing by $pi$ radians. These rotations fix the third rotational degree of freedom of the object at each viewpoint. However, the canonicalizing transformations for neighboring viewpoints on the viewsphere may be quite different; this is particularly true when the two viewpoints lie across a ‘visual event boundary’. An example of a visual event is the appearance of a previously occluded object face. Such aspect changes have the
potential to change the 2D footprint of the object in drastic ways thus changing the elongation axis of the object.

For notational purposes, let the parameter $\theta$ specify the rotation about a viewpoint of the object, while $(\gamma, \eta)$ specifies the viewpoint. Then $\theta, \gamma,$ and $\eta$ parameterize the shape manifold produced from the object in the subspace $\mathcal{E}$ by projecting the shape appearance of the object at each selected value of $\theta, \gamma,$ and $\eta$. Furthermore, since the canonicalizing transformations $R1_i$ and $R2_i$ are used, only two values for $\theta$ are possible for a choice of $(\gamma, \eta)$. The parameter $\theta$ will be fixed for a given viewpoint to the values $\theta_1$ and $\theta_2$.

The shape manifold of an object summarizes information on the object’s shape. In particular the first derivative of the manifold’s surface relates to the change in shape of the object with the change in orientation about a specific point. This can be useful in detecting critical aspect changes that can dramatically vary the object’s shape appearance to the observer. Similarly, viewpoints on the viewsphere can be analyzed to find regions of the viewsphere where the shape appearance of the object is largely similar. This can be used for object surface segmentation and to measure of how reliable an object’s pose estimate is in the presence of noise.

In Section 3.4, the eigenspace $\mathcal{E}$ is derived from training data $\{\mathbf{T}^j_i, \mathbf{T}^j_2 : i = 1, \ldots, 2 \cdot N_{VP}\}$ for each model $j$ using a discretized viewsphere and the canonicalizing transformations $R1_i$ and $R2_i$ at each viewpoint. The training data $\mathbf{T}^j_i, \mathbf{T}^j_2$ is projected into $\mathcal{E}$ along the $k$ eigenvectors corresponding to the $k$ largest eigenvalues, producing prototypes $g1_i$ and $g2_i$ in the manifolds $M(\theta_1, \gamma, \eta)_j$ and $M(\theta_2, \gamma, \eta)_j$, where $(\{\theta_1|\theta_2\}, \gamma, \eta)$ are the parameters of the manifolds. That is, there are two manifolds produced, one employing $\theta_1$ as its third parameter and one employing $\theta_2.$

116
In the subspace $\mathcal{E}$, the discrete samples of the manifolds obtained from training data can be polygonized to obtain a connected surface approximating to the true manifold. The connectivity of the viewpoints $v_i$ is imposed on prototypes in the eigenspace $\mathcal{E}$, where the vertices of the viewsphere are replaced by the high dimensional prototypes $\tilde{g}_1^i$ or $\tilde{g}_2^i$ to form a mesh that approximates the manifold for an object.

Given the method of generating training data $\{\mathbf{T}_1^i, \mathbf{T}_2^i: i = 1, \ldots, 2 \cdot N_{VP}\}$ each planar facet of the viewsphere is associated with 6 prototypes $\{g_1^i, g_2^i, g_1^m, g_2^m, g_1^n, g_2^n\}$ where $l$, $m$, $n$ are the indices of the viewpoints designated by vertices of the facet. If the canonicalizing transformations $R_{1l}$, $R_{2l}$, $R_{1m}$, $R_{2m}$, $R_{1n}$, $R_{2n}$ yield $\theta_{1l} = \theta_{1m} = \theta_{1n}$ and $\theta_{2l} = \theta_{2m} = \theta_{2n}$ then the facet approximates the manifold resulting from fixing the rotation about the viewpoint parameter. Due to aspect changes and numerical considerations, however, the rotational parameter will typically vary between neighboring viewpoints: $\theta_{1l} \neq \theta_{1m} \neq \theta_{1n}$ and $\theta_{2l} \neq \theta_{2m} \neq \theta_{2n}$. With the appropriate sampling (which varies between objects due to different aspect changes) of each object’s viewsphere, the localized variation in $\theta_1$ and $\theta_2$ will be small.

Hence, two polygonal mesh manifolds will be formed from the training data $M(\theta_1, \gamma, \eta)$ and $M(\theta_2, \gamma, \eta)$ for each object (Figure 3.7). The symmetry of the manifolds results from the symmetry of the training data which is used to build the subspace $\mathcal{E}$. Figure 3.7 was generated by displaying the manifolds in a three dimensional subspace of $\mathcal{U}$. The visualized dimensions are chosen by finding the the coordinates in the universal eigenspace $\mathcal{U}$ that have the largest variations.
Figure 3.7: Top: an mechanical part and an elephant and their respective manifolds. Bottom: both manifolds are rendered together in the universal eigenspace $\mathcal{U}$.
3.6 Recognition of Objects in Range Images

In appearance-based object recognition (Section 3.2), the training images of all database objects \( X = \{\overline{T}_1, \ldots, \overline{T}_M\} \) are projected into the universal eigenspace \( U \) along the eigenvectors \( \overline{e}_j \) to produce \( k \)-dimensional prototypes \( \overline{g}_i \).

During recognition, the range images \( R \) are first preprocessed using techniques described in Sections 3.3 and 3.3.1 to produce the image template \( T_R \). The template \( \overline{T}_R \) is then projected into the same subspace \( U \) using the the eigenvectors \( \overline{e}_j \) to produce a prototype \( \overline{g}_R \) (Section 3.5 and [95]), but finding the distance between the complex surface of the manifold or even the polygonal approximation described above is much more complex computationally than using nearest-neighbor search on the point prototypes \( \overline{g}_i \). The result produces a training prototype \( \overline{g}_i \) that is closest to the prototype \( \overline{g}_R \). During training, the object identity and pose are stored along with the training prototype \( \overline{g}_i \), in order to provide the system with knowledge about the identity and pose of the object in the training view \( T_i \). Figure 3.8 contains a block system diagram of this technique.

To improve the efficiency of the nearest-neighbor search through the training prototypes we use Nene and Nayar’s technique [101, 98, 102] for efficient high dimensional search. This technique reduces the complexity of the search to approximately \( O(k \cdot \log_2(n)) \) where \( k \) is the dimension of the subspace and \( n \) is the total number of prototypes. The technique orders the coordinate vectors of the \( n \) prototypes so that all the prototypes \( \overline{g}_i \) within a hypercube centered at \( \overline{g}_R \) are found using binary searches on the coordinates. If the size of the hypercube is small the technique
is very close to $O(k \cdot \log_2(n))$ complexity. Nene and Nayar show [101] show that their recognition times are about twice as fast as the k-d tree algorithm [120] on real systems.

3.7 Preliminary Synthetic Model And Image Experiments

The results presented in this section were documented in a conference publication [23]. These preliminary results show the utility of using appearance-based recognition system for range data and were extensively tested on two object databases. One database contained 20 free-form objects and the other containing 54 mechanical parts. This initial work differs slightly from the system described in the prior sections. The main differences are:
1. Both model and range images where synthetically generated so precise locations of the object's could be controlled.

2. Because the locations of object was known, the translation normalization steps in Section 3.3.1 were unnecessary, as they are intended to deal with factoring out uncertainty in the location of object with respect to the scanner.

As a result of these differences, the system tested here would not able to recognize objects in range data captured from our scanner unless precise positions of the objects are known before recognition.

The system was synthetically tested on two object databases of 3D models. The first object database (database A) contains twenty 3D models (Figure 3.9). The database contains objects from human bones, mechanical parts (designed on a mechanical CAD package), to reverse-engineered 3D models constructed from range imagery.\(^5\) All models are stored in a polyhedral mesh format approximating the 3D smooth surfaces. The second database (database B) contains 54 mechanical parts designed using CAD packages (Figure 3.10).

For each database, a series of tests was run to determine the dependence of recognition accuracy on the number of viewpoints \(N_v\), the size of the image \(M \cdot N\), and the dimension of the subspace \(k\). A structured approach was used to generate test views. Since the training views were generated at the vertices of the subdivided icosahedron, we felt that viewpoints (for the test views) chosen as far as possible from the vertices (training views) would provide a worst-case test set. Therefore, test views

\(^5\)We are grateful to Marc Soucy of Innovmetric Corp., Marc Levoy of Stanford University, and Kari Pulli of the University of Washington/Nokia for making their models available.
Figure 3.9: Some Examples of Objects From Database A
Figure 3.10: Some Examples of Objects From Database B
were chosen from viewpoints corresponding to the centers of the triangular faces in the \( l \)-frequency subdivision of the icosahedron.

For database A, eleven subspaces were generated for testing, corresponding to different values of \( N_{\text{VP}}, l, \) and \( M \cdot N \). The results of recognition rate vs. \( k \) are shown in Figure 3.11 for the various eigenspaces defined in Table 3.3.

![Average Recognition Rate for Smooth Object Database](image)

Figure 3.11: Recognition Rates for Database A

For database B, nine subspaces were generated. The results of the trials are shown in Figure 3.12 for the eigenspaces defined in Table 3.4.

¿From Figures 3.11 and 3.12, we see that the most important parameters are the number of views \( N_{\text{VP}} \) and the dimension of the subspace \( k \). For database A a subspace \( E \) with dimension greater than 13 does not improve the recognition rate. The larger database B requires \( k > 20 \) before the recognition rate approaches its maximum. A
<table>
<thead>
<tr>
<th>Subspace</th>
<th>$N_{VP}$</th>
<th>$M \cdot N$ (image size)</th>
<th>Best Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{E}_1$</td>
<td>12</td>
<td>1024 (32x32)</td>
<td>75%</td>
</tr>
<tr>
<td>$\mathcal{E}_2$</td>
<td>12</td>
<td>4096 (64x64)</td>
<td>74.5%</td>
</tr>
<tr>
<td>$\mathcal{E}_3$</td>
<td>12</td>
<td>16384 (128x128)</td>
<td>75%</td>
</tr>
<tr>
<td>$\mathcal{E}_4$</td>
<td>12</td>
<td>65536 (256x256)</td>
<td>74.7%</td>
</tr>
<tr>
<td>$\mathcal{E}_5$</td>
<td>42</td>
<td>1024</td>
<td>91%</td>
</tr>
<tr>
<td>$\mathcal{E}_6$</td>
<td>42</td>
<td>4096</td>
<td>92.3%</td>
</tr>
<tr>
<td>$\mathcal{E}_7$</td>
<td>42</td>
<td>16384</td>
<td>92.8%</td>
</tr>
<tr>
<td>$\mathcal{E}_8$</td>
<td>92</td>
<td>1024</td>
<td>97.4%</td>
</tr>
<tr>
<td>$\mathcal{E}_9$</td>
<td>92</td>
<td>4096</td>
<td>97.3%</td>
</tr>
<tr>
<td>$\mathcal{E}_{10}$</td>
<td>162</td>
<td>1024</td>
<td>99%</td>
</tr>
<tr>
<td>$\mathcal{E}_{11}$</td>
<td>252</td>
<td>1024</td>
<td>99.7%</td>
</tr>
</tbody>
</table>

Table 3.3: Table of parameters used in generation of subspaces for Database A

A significant result found in both Figures 3.11 and 3.12 is that the size of the images $M \cdot N$ does not significantly change the recognition rate; even when a 16 x 16 image template is used, the recognition rate only drops a few percent. Apparently, coarse shape matching is all that is needed to distinguish the objects in these databases.

The number of training views does have a dramatic effect on the recognition rate. For database A, increasing the number of training views from 12 to 92 increases the recognition rate from 75% to 97%. For database B, the rate increases from 47% to 80% when this change is made. For 252 views the eigenspace calculated for database A almost obtains perfect recognition, while database B only obtains a 91% recognition rate.

The results from the larger database B of mechanical objects consistently demonstrate much lower recognition rates. This is in part due to some object similarity under certain viewing positions. In cases where a particular view of an object changes in shape appearance enough from its neighboring training views, the view may be
matched with another object of similar shape in a similar pose. Figure 3.13 shows an example where the view to be recognized $\mathbf{T}_R$ from object ‘bigwye’ is incorrectly identified as another cylindrical object with handles ‘331c’. In these particular views of ‘bigwye’ and ‘331c’, the large discriminatory features have disappeared and the neighboring views of the ‘bigwye’ object are more dissimilar than that of the ‘331c’ training view. This is due to the change of aspect caused by the appearance of the wye feature on the ‘bigwye’. Inbetween two neighboring viewpoints in the training data, the appearance of the wye causes enough change in the object appearance that matching errors occur because the test surface is more similar to the prototypes taken from the ‘331c’ object.
<table>
<thead>
<tr>
<th>Subspace</th>
<th>$N_{VP}$</th>
<th>$M \cdot N$ (image size)</th>
<th>Best Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{E}_1$</td>
<td>12</td>
<td>1024 (32x32)</td>
<td>44.8 %</td>
</tr>
<tr>
<td>$\mathcal{E}_2$</td>
<td>12</td>
<td>4096 (64x64)</td>
<td>47.5%</td>
</tr>
<tr>
<td>$\mathcal{E}_3$</td>
<td>12</td>
<td>16384 (128x128)</td>
<td>47.1%</td>
</tr>
<tr>
<td>$\mathcal{E}_4$</td>
<td>42</td>
<td>256 (16x16)</td>
<td>65%</td>
</tr>
<tr>
<td>$\mathcal{E}_5$</td>
<td>42</td>
<td>1024</td>
<td>68.3%</td>
</tr>
<tr>
<td>$\mathcal{E}_6$</td>
<td>42</td>
<td>4096</td>
<td>70%</td>
</tr>
<tr>
<td>$\mathcal{E}_7$</td>
<td>92</td>
<td>256</td>
<td>80%</td>
</tr>
<tr>
<td>$\mathcal{E}_8$</td>
<td>92</td>
<td>1024</td>
<td>81%</td>
</tr>
<tr>
<td>$\mathcal{E}_9$</td>
<td>162</td>
<td>1024</td>
<td>88%</td>
</tr>
<tr>
<td>$\mathcal{E}_{10}$</td>
<td>252</td>
<td>1024</td>
<td>91%</td>
</tr>
</tbody>
</table>

Table 3.4: Table of parameters used in generation of subspaces for Database B

In all these experiments, the angular sampling of the possible views is coarser than that used in current appearance-based methods. In Murase and Nayar’s seminal work on 3-D object recognition [95] an angular separation of 4° was used in the 1D orientation space. They used a coarser sampling of 7.5° in their work on recognition of a 100 object database [98]. Since our system samples a 2D pose manifold embedded in 3D, a larger angular spacing was needed because of memory and computation requirements. There are undoubtedly additional optimizations that can be made to increase the sampling frequency in the 2D pose space; such optimizations are the topic of current research and are presented in Section 3.8.

The results above demonstrate the usefulness of an appearance-based approach to recognize and determine the pose of objects in synthetic $2_{2D}^D$ shape data. These results show some interesting properties of using appearance-based methods for range image object recognition under ideal conditions.
1. Image size does not dramatically affect the ability to distinguish objects in the universal eigenspace $\mathcal{U}$.

2. A subspace with dimension $k \approx 20$ is sufficient to archive a good compromise between good recognition rates for database A and B and space complexity of the database.

3. Decreasing the angle between training views increases the recognition rate.

4. The change of some aspects of an object can cause misclassification during recognition.

5. The representation has shown its ability to recognize both free-form objects and manufactured parts.
6. Using a simple 2D ‘footprint’ to align each view with the coordinate axes is a useful way to resolve the third degree of freedom in the pose.

3.8 View Planning

Previous recognition results (Section 3.7) and the discussion in Section 3.5 suggest that sampling of the pose space of an object must be adaptive if the number of views taken for each object is to be minimized. As a focus of future research, a density of sampling that will produce accurate pose results, improve discriminatory power in regions of similarity and handle critical aspects (where large surface changes occur over relatively small changes in view direction) must be determined.

The result of uniform sampling of the pose space on the viewsphere produces irregular polygonal meshes that approximate the manifold in the eigenspaces $U$ or $O_j$. The facets of the mesh clearly show that some neighboring views are dissimilar enough that the prototypes are far apart in the eigenspace. This can cause a problem with the correct recognition and pose estimation when the object is viewed in these orientations, as in the case of the ‘bigwye’ object and ‘331a’ (Figure 3.13). In this case, the closest match to test image $\tilde{g}^*$ was ‘331a’ because the neighboring viewpoints from the ‘bigwye’ object were farther away from $\tilde{g}^*$ in the subspace $U$ than prototypes from part ‘331a’.

The irregular mesh formed from the prototypes $\tilde{g}_i$ in the subspaces $U$ and $O_j$ allows for uncertainties in the matches for regions where neighboring poses project to prototypes with large separations. We would prefer to have a more regular mesh, so we would be equally as confident of matches in all regions of the pose space.
Figure 3.14: System Diagram (View Planning)
At present it is not essential to make the object’s manifold regular, just to eliminate the facets with large areas and those that are long and thin. This will be accomplished by modifying the viewsphere to produce new views that will more densely sample the manifold in such regions.

The view planning process uses the polygonal approximation to the object’s manifold to decide where to take more views of the object. Those polygons on the manifold with areas that are ‘too large’ are split by subdividing them to produce new viewpoints. Recall that a patch on the manifold is associated with a face on the viewsphere. The associated face on the viewsphere is split to produce a new viewpoint at the center of the face. This face will provide two new training images for the system and thus a pair of new prototypes for the manifolds of the object. Likewise, edges between prototypes on the manifold are tested to see if the distance between prototypes is too large. If this occurs, the edge is split to produce a new viewpoint and another pair of prototypes.

The view planning process for the universal eigenspace may be different from that of view planning for individual object eigenspaces. In the case of the universal subspace $\mathcal{U}$, the goal is to maximize recognition rate. For the object’s subspace $\mathcal{O}_j$ the goal is to minimize the pose error.

The minimization of pose error depends on the method for recovering the pose. Currently we only use the pose recovered from the closest training prototype. A group of neighboring training prototypes might be used to extrapolate a better estimate of the pose. In our current method it is most important that the test prototype match to the correct closest training prototype. If this happens the pose error can only be as large as half the distance between any two views on the view sphere.
3.8.1 View Planning Results

To explore some view planning ideas we used the two objects ‘bigwy’ and ‘331c’ shown in Figure 3.15 (which exhibited a case of ‘mistaken identity in Section 3.7, motivating this investigation). We hope that with some view planning we will be able to differentiate between these two objects better and thus increase recognition rate. We begin by training an appearance-based recognition system using 42 uniformly distributed view points of each object, then we will use view planning to refine the number of views.

Figure 3.15: Two Objects used in some initial tests of view planning

For each object, two manifolds were created: $M(\theta_1, \gamma, \eta)$ and $M(\theta_2, \gamma, \eta)$, where the parameters $\theta_1$ and $\theta_2$ were fixed from the canonicalizing transformation and $\gamma, \eta$ parameterize the viewpoint. Rewrite the manifold’s notation as $M(\theta_1, \bar{v})$ and $M(\theta_2, \bar{v})$ to express the parameterization of the manifold surface by the view points.
corresponding to object’s training data $\mathbf{T}_i$, where \( \{ \mathbf{v}_i : i = 1 \cdots N_{nv} \} \). Consider a set of neighboring view points on the discrete viewsphere \( \{ \mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_l \} \) that form a triangular facet on the viewsphere’s surface. The corresponding training views \( \{ \mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_1, \mathbf{T}_2 \} \) are projected to the prototypes \( \{ \mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_1, \mathbf{g}_2 \} \) in \( \mathcal{U} \). The prototypes \( \{ \mathbf{g}_1, \mathbf{g}_1, \mathbf{g}_1 \} \) form a triangular facet that approximates a surface patch on \( M(\theta_1, \mathbf{v}) \) while \( \{ \mathbf{g}_2, \mathbf{g}_2, \mathbf{g}_2 \} \) approximates a patch on \( M(\theta_2, \mathbf{v}) \). The distance between neighboring prototypes measures the similarity between the shape appearance for neighboring views on the viewsphere. Likewise, the area of a triangular face also gives an indication of the distances between three neighboring object viewpoints.

The length of the edges and the area of the triangular facets \( \{ \mathbf{g}_1, \mathbf{g}_1, \mathbf{g}_1 \}, \{ \mathbf{g}_2, \mathbf{g}_2, \mathbf{g}_2 \} \) measure the similarity of each patch and edge on the manifolds \( M(\theta_1, \mathbf{v}) \) and \( M(\theta_2, \mathbf{v}) \). As a first approach to view planning we wish to decrease the number of large facets on the manifolds as well as the number of large edges. To do this we will subdivide the facets whose area is larger than one standard deviation above the average facet area. For each such subdivided facet, new viewpoints were found and used to produce new training data. The new training data is used to reform the subspace and the manifolds.

Using these methods for determining where new viewpoints should be generated we iteratively refined a uniform sampling of the viewsphere, generated new universal eigenspaces \( \mathcal{U}_t \) and reformed the manifolds for a database of two objects. Table 3.5 shows that as view planning progresses using this method, the recognition rate increases, while the manifold facet area and the manifold facet area variance decreases. However, we also note that uniform sampling of the viewsphere at the next highest
<table>
<thead>
<tr>
<th>Subspace</th>
<th>$N_v$</th>
<th>AEL</th>
<th>ELV</th>
<th>APA</th>
<th>PAV</th>
<th>Recognition Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uniform 42</td>
<td>84</td>
<td>6.692</td>
<td>1.955</td>
<td>19.2758</td>
<td>39.581</td>
<td>95 %</td>
</tr>
<tr>
<td>1&lt;sup&gt;st&lt;/sup&gt; Iteration</td>
<td>111</td>
<td>6.573</td>
<td>2.129</td>
<td>18.164</td>
<td>33.238</td>
<td>96.25 %</td>
</tr>
<tr>
<td>2&lt;sup&gt;nd&lt;/sup&gt; Iteration</td>
<td>149</td>
<td>6.3915</td>
<td>2.529</td>
<td>16.66</td>
<td>30.60</td>
<td>96.4 %</td>
</tr>
<tr>
<td>3&lt;sup&gt;rd&lt;/sup&gt; Iteration</td>
<td>190</td>
<td>6.2123</td>
<td>3.025</td>
<td>15.195</td>
<td>30.32</td>
<td>97.4 %</td>
</tr>
<tr>
<td>Uniform 92</td>
<td>184</td>
<td>5.592</td>
<td>2.4146</td>
<td>13.56</td>
<td>33.263</td>
<td>99.3 %</td>
</tr>
</tbody>
</table>

Table 3.5: View planning statistics and recognition results: AEL is average edge length, ELV is edge length variance, APA is average patch area and PAV is patch area variance.

Subdivision yields a better recognition rate with less training data. One of the goals of view planning is to decrease the total number of training views used to build $U$ and still obtain high recognition rates. In the case of our initial view planning the planned system contains more view points and and lower recognition rate than a system generated from uniformly sampling the viewsphere.

In Figure 3.16, we show the density of the facets on the manifold for each iteration of view planning verses the area of the facets. The density plots show that in general the area of the larger facets is decreasing as more and more views are added, but there exist some facets with areas that do not decrease despite the subdivision process. This is even more notable in Figure 3.17, which lists an area-normalized facet density. This ratio gives a measure of an object’s shape appearance change between neighboring viewpoints normalized by the distance (in view space) for which that change occurred. The new figure indicates that the larger differences between neighboring object shape appearance occurs over a similarly small change in pose of the objects.

To further analyze these results, we examined the new discrete viewspheres for the planned systems. We determined that subdivisions tended to occur near viewpoints
Figure 3.16: Manifold Patch area density for each step of view planning

where the in-plane canonicalizing rotations are unstable. Because of changes in the 2D footprint of the object between neighboring views the canonicalizing transformation radically oriented object differently between views (Figure 3.18).

These results highlight some problems with view planning that need to be addressed in future research. Because of the way we generate training data, some objects may occur large differences between neighboring views due to sensitivity of the view to canonicalizing transformation for that object. This enhances the need to more intelligently plan the views for each object. Because of canonicalizing transformation instabilities or critical aspect changes, there may exist some regions on an object viewsphere were refining the viewpoints does not improve an appearance-based object recognition systems performance. It would be more efficient to refine other regions of the viewsphere for the object. At least recognition performance in
Figure 3.17: The density of manifold patch area over viewsphere face area for each step of view planning
Figure 3.18: Canonicalizing transformation pose instabilities
these regions would improve and thus improve the performance of the system, while still minimizing the number of viewpoints used to train the system.

3.9 Real Model And Image Experiments

<table>
<thead>
<tr>
<th>Model</th>
<th>Number of Vertices</th>
<th>Number of Edges</th>
<th>Number of Polygons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple</td>
<td>7070</td>
<td>2191</td>
<td>14126</td>
</tr>
<tr>
<td>Cow</td>
<td>10032</td>
<td>30090</td>
<td>20060</td>
</tr>
<tr>
<td>Duck Rattle</td>
<td>10731</td>
<td>32181</td>
<td>21451</td>
</tr>
<tr>
<td>Toy Lamb</td>
<td>10383</td>
<td>31143</td>
<td>20762</td>
</tr>
<tr>
<td>Orange Dino</td>
<td>10191</td>
<td>30561</td>
<td>20374</td>
</tr>
<tr>
<td>Po</td>
<td>10645</td>
<td>31923</td>
<td>21282</td>
</tr>
<tr>
<td>Red Dino</td>
<td>12262</td>
<td>36774</td>
<td>24516</td>
</tr>
<tr>
<td>Rubber Duck</td>
<td>10413</td>
<td>31233</td>
<td>20822</td>
</tr>
<tr>
<td>Whale</td>
<td>4853</td>
<td>14502</td>
<td>9652</td>
</tr>
<tr>
<td>Yellow Croc</td>
<td>9875</td>
<td>29601</td>
<td>19734</td>
</tr>
</tbody>
</table>

Table 3.6: The number of vertices, edges and polygons for the models in the 10 object database.

The complete system for free-form 3D object recognition in depth data (Figure 3.8) was tested on the 10 object database shown in Figure 3.20 and whose statistics are given in Table 3.6. The 10 object models were generated from toys using reverse engineering techniques to build a polygonal mesh models of the objects (Section 2.4). The CAD models are used with synthetic range rendering techniques to build the various eigenspaces used to test the proposed recognition technique. Figure 3.19 shows examples of typical ‘eigenshapes’ in the object database look like. The first eight ‘eigenshapes’ represent the eigenvectors that correspond to the largest eight eigenvalues (the upper left eigenshape is associated with the largest eigenvalue). The
last image in Figure 3.19 shows the eigenvector with the 100th largest eigenvalue. The shapes of the eigenvectors indicate that most of each view’s shape can be reconstructed from a characteristic set of smooth slowly changing shapes (given by eight largest eigenvalues). Conversely, the reconstruction of the object’s views from the low-variance basis vectors are less important.

Figure 3.19: Examples of ‘eigenshapes’
For the 10 object database, a series of tests was run to determine the dependence of recognition accuracy on the number of viewpoints $N_{vp}$, the size of the image $n$, the dimension of the subspace $k$, the sensitivity to the various view position statistics (closest point, average, and median), and the two different canonicalizing transformations (2D and 3D). The trials were divided into two different cases:

1. A series of trials in which training and testing images are both synthetic.

2. A series of trials in which synthetic images are used to training and real images are used to test.

In all the tests, the dimension of the subspace $k$ is varied. For the synthetic test image trials, the system is tested with 500 independent trials for each dimension of the subspace being tested. Each trial is of a random object and viewpoint. The real image tests were carried out on a database of 150 range images (15 images of each object) for each dimension of the subspace being tested.

In Section 3.3.1 we introduce three different view-based image statistics (nearest object point, average object point and median object point). We used these image based statistics to translate the model $O$ or range image mesh $\mathcal{R}$ to a predetermined location. This removes (reduces) the affect of uncertainty in object position along the scanner’s optical axis (Section 3.3). Also recall (Section 3.4) that when training our recognition system we mentioned using two different in plane rotations around the optical axis to align the object’s direction of orientation with the $x$ axis. This factored out an additional degree of freedom in the rotational pose of the object without having to train for that variability. We wish to study the effects of training
the system using the above techniques with respect to recognition rate on synthetic and real test images.

The following notation is used in the tables:

- **T1** denotes a system that uses the nearest point statistic to establish a $z$ reference.
- **T2** denotes a system that uses the average object point statistic to establish a $z$ reference.
- **T3** denotes a system that uses the median object point statistic to establish a $z$ reference.
- **C2D** denotes a system that uses the elongation direction of the '2D footprint' to orient $R$ or $O$ meshes about the optical axis,
- **C3D** denotes a system that uses the elongation direction of the 3D point cloud to orient $R$ or $O$ meshes about the optical axis.

In the first two experiments, the appearance-based recognition database was trained by fixing the number of viewpoints at 92 for each object and using a $32 \times 32$ template image resolution for $T$. Each curve in Figure 3.21 and 3.22 represents recognition accuracy for the various appearance-based recognition system where the method for handling object location uncertainty in $z \ t_2$ and third degree of pose variability changes. The six different trials were:

1. train using $T1$ and $C2D$,
2. train using $T1$ and $C3D$,
3. train using $T2$ and $C2D$,

4. train using $T2$ and $C3D$,

5. train using $T3$ and $C2D$,

6. train using $T3$ and $C3D$.

In the next two experiments, the appearance-based recognition database was trained using 92 view points per object, translated based on the median point statistic $T3$ and oriented using 3D PCA $C3D$. Each curve in Figure 3.23 and 3.24 represents the recognition results where the resolution of the image templates are changed from $16 \times 16$ to $128 \times 128$.

The final two experiments show appearance-based recognition systems trained using $32 \times 32$ images, translated based on median point statistic $T3$ and oriented using 3D PCA $C3D$. Each curve in Figure 3.25 and 3.26 represents the recognition results for appearance-based recognition systems where the number of viewpoints are changed from 12 to 362.

3.10 Conclusions

In this chapter we have shown the utility of an appearance-based approach to represent and recognize 3D objects from $2\frac{1}{2}$D range data. It has shown some interesting properties in using appearance-based methods for range image object recognition under ideal conditions for our two databases.

1. Image size does not dramatically effect the ability to distinguish between objects in the eigenspaces.
Figure 3.20: 10 objects used to train and test recognition methods
Figure 3.21: Synthetic experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represent a database trained using different positional statistics and canonicalizing transformations.
Figure 3.22: Real range data experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represent a database trained using different positional statistics and canonicalizing transformations.
Figure 3.23: Synthetic experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represents a database trained at different template resolutions (image resolution 16, 32, 64, 128).

Figure 3.24: Real range data experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represents a database trained at different template resolutions (image resolution 16, 32, 64, 128).
Figure 3.25: Synthetic experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represents a database trained using the different densities of view sampling Table 3.2.
Figure 3.26: Real range data experiment testing the 10 object database with respect to dimension of the matching subspace. Each curve represent a database trained using the different densities of view sampling Table 3.2.
2. Recognition rates in our experiments did not increase significantly for dimension 
k > 20.

3. A decrease in the angle between training views increases recognition rate.

4. The change of some aspects of an object can cause misclassification during recognition (Section 3.7 and 3.8).

5. This representation is useful for discriminating free-form objects as well as manufactured parts.

6. Using a simple 2D ‘footprint’ C2D or 3D PCA C3D algorithm to align each view with the coordinate axes are useful ways to fix the third degree of freedom in pose (Section 3.4).

The results suggest that best sampling the views of an object will have to be adaptive if the number of training views taken for each object is to be minimized. To this end a viewplanning process needs to be developed to produce accurate pose results, improve discriminatory power in regions of similarity and handle critical aspects where large surface changes occur over relatively small changes in view direction.
CHAPTER 4

Local Feature Object Recognition In The Presence Of Weak Correspondence

4.1 Introduction

1. Local feature object recognition in general

   (a) The Saliency Problem

   (b) The Search Problem

2. Andrew Johnson’s work

3. Chua and Jarvis’s work

4.2 Segmentation

   In order to address the search problem, we use surface segmentation to limit search regions and to capture the geometric structure of the objects in our database. For our problem, we wish to exploit features based on the free-form shape of our objects to differentiate the models from one another. In this case, the distribution of the high curved regions defines salient features on the surface of the objects. These valleys,
ridges, pits and peaks define landmarks on the surface that can be used to capture some of the structural information about an object.

In the past, it has been claimed that applying segmentation to the free-form matching problem is problematic at best, because of the lack of clearly defined and stable boundaries to the segments [13]; (also see the discussion surrounding Figure 4.1). In recent work by Srikantiah and Boyer [130] it has been noted that by using homogeneity criteria based on consistency in mean or Gaussian curvature, a reliable range image segmentation can be found. This technique does not utilize predetermined decision regions on the mean and Gaussian curvature values; rather, it allows the algorithm to search through the data and determine connected regions containing pixels whose mean or Gaussian curvature largely agrees within some criteria of homogeneity. In addition to the homogeneity criteria, they ranked the recovered segments based on reliability and surface area to produce a hierarchy used in region merging routine to absorb small (noise) segments into the much more reliable neighboring segments. The method has proven through experimentation to be fairly invariant to noise and surface quantization errors inherent in surface digitization.

For our work the criticism of using segmentation for free-form recognition is addressed by considering a clearly defined criteria for segmentation with an understanding that the segmentation results are likely to be imperfect. The segments and sub-segments we use to prune the search in the hypothesis generation and verification routines in our prototype recognition system and are not used with interpretation trees or relational graphs to directly determine object identity. The homogeneity criteria for our segments will find connected regions where the surface is changing shape rapidly. These high curved regions will be further divided based on a coarse shape
classification into three basic classes (convex, saddle, or concave). This will produce three types of high curvature segments.

4.2.1 Curvature

To find the regions of interest for our segments we will be using surface curvature. The fundamental concepts of surface curvature in its various forms is summarized in Appendix B. The Appendix includes notation and an overview of the mathematics behind curvature.

The estimation of surface curvature from range data is prone to errors due to the amplification of noise in the application of the derivative operator. To combat the noise problem two techniques are typically applied in the literature [45]:

1. the data is smoothed to lessen the effect of noise in the measurement,

2. or the data is fit to a known surface model whose derivatives can be found using a closed form solution.

Another problem in estimating surface curvature is the loss of locality of the surface measure, due to the larger region of support created in the smoothing or fitting process.

In this document we will be using surface fitting to estimate surface curvature for both the range images and for the polyhedral mesh CAD models that form our object database. The method developed by Flynn and Jain [43, 45] uses the following steps to estimate curvature:

- gather a neighborhood of data $\Omega_p$ around point of interest $p$,

- center the data in $\Omega_p$ to place $p$ at the origin,
• estimate a tangent plane $T_p$ to $\Omega_p$ and a local coordinate system defined by the normal to the tangent plane and two orthogonal vectors in the plane,

• fit a bicubic surface to the neighborhood of points in $\Omega_p$, and

• estimate the surface curvature at the point $p$ using coefficients of the fitted surface.

The tangent plane $T_p$ is defined as the plane that best fits the neighborhood of points $\Omega_p$ (minimizes least squares perpendicular error). The planar fit is calculated using principal component analysis to find the directions of least and most variation in the data [77]. The direction with the least variance defines the normal to the tangent plane $T_p$ for the data. The other principal components lie in the tangent plane and define an orthonormal coordinate system for $p$. Denote the coordinate system by $\{\mathbf{v}_x, \mathbf{v}_y, \mathbf{N}\}$ where $\mathbf{v}_x$ and $\mathbf{v}_y$ lie in the tangent plane and $\mathbf{N}$ estimates the surface normal direction at $p$. The normal estimated from principal component analysis could be pointing inside the object depending on the location of the data with respect to the object or image coordinate system. To resolve the ambiguity we will be using the vertex normal estimates obtained from the polygonal meshes representing the range image or the model. In the polygonal mesh representation the vertices of the mesh are ordered to indicate which side of the polygonal faces are inside the object and which are outside. This allows us to flip the inward facing normal estimates to obtain outward facing surface normals.

The points in $\Omega_p$ are then projected into the new coordinate system $\{\mathbf{v}_x, \mathbf{v}_y, \mathbf{N}\}$ where the new $z$ direction is distance along the surface normal at the point. Denote the projected points as $(x', y', z')$. Then the projected neighborhood of points are fit
to a bicubic surface

\[ z' = f(x', y') = a_1 x'^3 + a_2 x'^2 y + a_3 x' y'^2 + a_4 y'^3 + a_5 x'^2 + a_6 x' y' + a_7 y'^2 + a_8 x' + a_9 y' + a_{10} \]

using a least squares solution. To calculate the least squares solution we use linear algebra routines from the CLapack software library \[37\].

The coefficients of the fit are used to estimate the principal curvatures \( \kappa_1 \) and \( \kappa_2 \) and their corresponding directions \( \vec{\kappa}_1 \) and \( \vec{\kappa}_2 \):

\[
\begin{align*}
\kappa_1(p) &= H + \sqrt{H^2 - K}, \\
\kappa_2(p) &= H - \sqrt{H^2 - K} \\
\vec{\kappa}_1(p) &= \begin{cases} 
(a_5 - a_7 + \sqrt{(a_5 - a_7)^2 + a_6^2})\vec{v}_x + a_6\vec{v}_y, & \text{if } a_5 \geq a_7, \\
 a_6\vec{v}_x + (a_7 - a_5 + \sqrt{(a_5 - a_7)^2 + a_6^2})\vec{v}_y, & \text{otherwise}
\end{cases} \\
\vec{\kappa}_2(p) &= \begin{cases} 
-a_6\vec{v}_x + (a_5 - a_7 + \sqrt{(a_5 - a_7)^2 + a_6^2})\vec{v}_y, & \text{if } a_5 \geq a_7, \\
 (a_5 - a_7 - \sqrt{(a_5 - a_7)^2 + a_6^2})\vec{v}_x + a_6\vec{v}_y, & \text{otherwise}
\end{cases}
\end{align*}
\]

where the formulas for mean and Gaussian curvature are:

**Gaussian Curvature** \( K(p) = 4a_5a_7 - a_6^2 \),

**Mean Curvature** \( H(p) = a_5 + a_7 \).

The principal curvatures and directions are reordered based on the magnitude of the curvature so that \( |\kappa_1| \geq |\kappa_2| \). Then the shape index \( S \) and curvedness \( C \) (Koenderink \[81\] and Appendix B) are calculated for the point of interest \( p \) using:

\[
\begin{align*}
S(p) &= \frac{2}{\pi} \arctan\left( \frac{\kappa_1(p) + \kappa_2(p)}{\kappa_1(p) - \kappa_2(p)} \right), \\
C(p) &= \sqrt{\frac{\kappa_1^2(p) + \kappa_2^2(p)}{2}}.
\end{align*}
\]

Figure 4.1 shows the output of this curvature estimation technique on the cow polygonal mesh model. In Figure 4.1(b) the *Gaussian* curvature image of the cow,
Figure 4.1: Cow object model and its curvature visualizations. (a): Toy Cow. (b): Gaussian Curvature. (c): Mean Curvature. (d): Curvedness. (e): Shape Index.
the red shade identifies points where the principal curvatures are of the same sign (indicating a convex or concave surface shape) and the blue shade identifies points where the principal curvatures are of different sign (indicating a saddle surface shape), while the intensity of the color is a function of the magnitude of the curvature value. Figure 4.1(c) displays the Mean curvature image of the cow; the red shade indicates positive mean curvature (convex shape) and the blue shade indicates negative mean curvature (concave shape), while the intensity of the colors indicate the magnitude of the mean curvature. In Figure 4.1(d) the Curvedness image of the cow, the intensity of the color indicates the strength of the surface surface change. In Figure 4.1(e) the Shape Index visualization of the cow is more complex. Here the colors identify surface class. The classes are listed in Table B.2 and the corresponding colors are:

- red: convex ellipsoid,
- yellow: convex cylinder,
- green: saddle,
- light blue: concave cylinder,
- blue color: concave ellipsoid.

For the CAD models in our database curvature is estimated at every vertex. At each vertex, concentric "rings" of vertices surrounding it are gathered until a sufficient amount of the model’s surface has been captured. In our experiments and for our models we will fit a bicubic surface patch to the set of points with in 7mm of the points of interest. For range images, the data in the image is converted to a polygonal mesh then the same routines are used. For the range image mesh the
pixels (vertices) near the borders are considered unreliable locations for curvature estimation. These locations represent boundaries between objects or between an object and its background. Because of this, the estimates of curvature in these areas could be quite different from the curvature that may be calculated for the same region on the object models. In an effort to minimize the difference between curvature values on the range image verses the model for the same portion of the object, we will use an ‘erosion’ morphological operator [53] to remove the estimates of curvature for the range image vertices near boundary between objects and the background.

4.2.2 Segmentation Technique

The shape index depiction (Figure 4.1 (e)) of the cow shows some of the reasons why segmentation has not been used as description for free-form surfaces. In the figure, many small spurious regions can be seen; these arise from errors in the estimation of surface curvature and noise in the placement of the vertices in the mesh model. The boundaries of the regions are also rather jagged (due to the discrete nature of the mesh model) and small variations in shape index value occur near the decision boundary between classes. If a precise surface model of the cow were known, the location of these boundaries could be quite different.

We have noticed that reliability of the shape classification decreases in regions of slow surface change. In these regions the noise in the surface measurements can dramatically influence the estimation of the type of the surface through curvature. Consider for example utilizing a range sensor to image a plane. Conceptually the curvature of the surface should be zero and the shape index should be undefined, but
in practice the curvature values are small (as a result of the noise) and the shape index is defined but unreliable.

For recognition, these slowly changing surface regions (at least locally) are not very interesting. This results from the low saliency of generally flat regions on a free-form object. The relationship between these regions can be a strong descriptor of the object though [130], depending on the database of objects and the saliency of these relationships. Often the size and shape of these slowly changing regions can be unreliable and affect the ability to correctly hypothesize the most likely objects.

We propose a segmentation system that avoids these slow changing surface regions and focuses on the quickly changing regions (Figure 4.2(a)). In these regions the classification of surface shape is less affected by noise, while structurally interesting regions are captured and exploited for recognition.

Another aspect of any segmentation system in addition to its reliability is whether the segments are useful to extract and quantify information about the object for the task at hand. In our case, we wish to use the segmentation to aid in the determination of the identity of the objects in an range image. Furthermore the distribution and structure of these landmarks provide a descriptive model for each object. Consider the case were we approximate each ridge segment by a curve and the peaks and pits by a point; the distance and orientation between these skeletal representations of the segments provides a useful representation of the object. This representational strategy is limited by the ability of these landmarks to differentiate models in the database of interest. One example where this skeletal representation might not be useful is if the two objects contain the same ridge, peak, and pit relationships like two dolls whose articulation and joints are the same but the facial and body features
Figure 4.2: Cow object model segmentation results. (a): Quickly changing regions of the cow model (radius of curvature $\leq 10$ mm). (b): Segments. (c): Segment elements.
are slightly different. In our database, the objects largely contain vary different ridge, peak, and pit relationships.

In practice, we have found that extracting a skeletal representation and relationships too unreliable to use as a representational strategy. Instead we will only utilize the region based ridge, peak, pit detection as a representational method to generate our object identity and pose hypotheses.

The segmentation system we employ utilizes the basic surface classifications given by Koenderink [81] (Table B.2). The five basic classes shown in Figure 4.1(e) divide the surface into regions of similar surface shape. The shape classes we are most interested in are convex ellipsoid, concave ellipsoid, and saddle regions. The curvature estimation and classifications for these regions are detectable and reliable [107], considering we are utilizing high curvature regions and requiring that both principal curvatures be significant. Upon inspection of curvature results of several models and range images, it was noticed that several significant ridges or creases in the object briefly transition through convex or concave cylinder classified surface regions. In order to obtain a surface segment that an observer would classify as a connected region, we are going to allow convex ellipsoid, concave ellipsoid and saddle regions contain the less reliable convex, concave cylindrical surface areas. Hence, segments that start out describing a homogeneous convex ellipsoid regions will be allowed to grow into and include convex cylindrical regions, while concave ellipsoidal segments will be allowed to grow into concave cylindrical regions. For segments who describe homogeneous saddle regions will be allowed to grow into both convex and concave cylindrical regions.

The following gives an outline of the segmentation algorithm.
• Mark vertices whose curvedness $C$ is greater than a predetermined threshold $Th$. Then classify these high curvature vertices by Koenderink’s [81] shape classes given by the shape index value $S$ using the decision regions given in Table B.2.

• Sort the high curvature vertices by the curvedness $C$ value in descending order. Remove the vertices whose shape class is convex or concave cylinder.

• Starting from the highest curved vertex grow connected segments of uniform surface class (convex, concave ellipsoid and saddle). The segments are allowed to grow into regions of convex and concave cylinder vertices. As an example a segment that was seeded by a convex ellipsoid vertex are allowed to grow into convex cylinder vertices, while concave ellipsoid seeded segments can grow into concave cylinder vertices, and saddle seeded segments can grow into both concave and convex cylinder vertices.

• Sort the segments by surface area in descending order. The larger segments typically have a better chance of showing up in an range view.

• Remove any segments whose surface area does not exceed $25mm^2$. Small segments can occur due to noise in the measurements of the object’s surface.

Figure 4.2(b) shows an example of this segmentation algorithm applied to the toy cow CAD model.

Recall that for the model and range image the density of the samples for a given region on the object’s surface can vary widely (Figure 3.4). Because of this, the calculation of statistics for the segments on either the range image mesh or the model
mesh are always tied to the surface area. Take for example the approximation of the location of the object. One could use the average vertex value to estimate the location of the segment, but this estimate will vary based on the distribution of the vertex samples for the segment. A better estimate would be to take into consideration the amount of surface area the vertex represents on the object’s surface. This can be calculated using Voronoi diagrams [108] to divide up the surface into cells that represent all the locations on the surface closest to the given vertex. The surface area of the cells gives a good estimate of the amount of surface area represented by a given vertex on the polygonal mesh.

Several statistics are calculated to summarize and prioritize the segments. First, the surface area is used to rank the largest segments with the highest priority considering they describe a rather large connected surface region of high curvature. These segments are more likely to appear in a range view of the same object even in the presence of occlusion because of their size. Then, the area-normalized shape index and area-normalized curvedness are calculated. Let \( S_i \) be a segment and \( \mathcal{V} = \{ \mathbf{v}_k : k = 1 \ldots N \} \) be a list of vertices in the segment, where \( N \) is the number of vertices in the segment. Each vertex in the segment has associated with it a region of the surface of the object it approximates using the Voronoi diagrams mentioned above and the surface area of that region for each vertex is denoted as \( \{ a_k : k = 1 \ldots N \} \). Then the area-normalized shape index for the segment is:

\[
SI_i = \frac{\sum_{k=1}^{N} a_k SI(\mathbf{v}_k)}{\sum_{k=1}^{N} a_k} \tag{4.4}
\]
and the area-normalized curvedness is given by

\[ C_i = \frac{\sum_{k=1}^{N} a_k C(\mathbf{v}_k)}{\sum_{k=1}^{N} a_k}. \]  

These measures will be used in hypothesis generation to determine the similarity between segments in an effort to prune the search for valid hypotheses.

---

**Figure 4.3:** Segmentation results. (a): Toy Cow object model. (b): Range image segments of both toy cow and apple.

---

During recognition, we wish to hypothesize the identity and pose of objects in a range image. To do this, we will be utilizing a discriminatory model database of polygonal CAD models of the objects we expect to see. Part of building this discriminatory model database is to segment the polygonal objects to mark the regions...
of high curvature that have uniform surface shape. When a similar segmentation is obtained from a range image, the segments appearing in the image will be likely subsets of the segments found on the polygonal model. This is a result of object self-occlusion and object-to-object occlusion present in the range scan. In addition to the absence of segments, the range image segments may also be smaller than the original segments on the model. Figure 4.3 shows the segments found on the toy cow model and the segments found on a range image of the toy cow. Because of object self-occlusion or object-to-object occlusion only portions of some of the segments present on the polygonal CAD model may be visible to the range scanner. For example, the large light green segment between the front legs of the toy cow (Figure 4.3 (a)) appears as only a small blue segment in the range image segmentation (Figure 4.3 (b)). The higher level of noise present in the range images can also cause changes between range image and model surface segments. An example of this can be seen on the tail of the toy cow. In Figure 4.3 (b), we can see the segment covering the tail is split into three segments; compared to the single segment in shown in Figure 4.3 (a) or 4.2(b). The error was cause by a classification of the crease in the tail as a strict saddle region instead of a convex cylinder as in the model.

Due to the presence of occlusion and noise, any recognition algorithm that utilizes this segmentation technique cannot rely on recovering exactly the same segments. To handle this uncertainty we will be splitting up our segments up into smaller sub-segments we will be calling segment elements (Denoted as \( S\mathcal{E}_j \)). To do this we will again rank the vertices by their curvedness, but this time we will do it on a segment by segment basis. So for each segment we will sort the vertices in descending curvedness order and grow segment elements.
The following gives an outline of the segment element subdivision algorithm.

- For each segment sort the high curvature vertices by the curvedness $C$ value in descending order.

- Starting from the highest curved vertex grow connected segments element until the segment element covers at least $25mm^2$ surface area on the object or cannot be grown any more. Once a segment element has covered the minimum surface area its growth is stopped and another segment element is started at the next highest curved vertex that has not been included in a segment element. This is continued until all the vertices that are apart of a segment have been assigned to a segment element.

- The segment elements $S_E_j$ in a segment $S_i$ are ordered based on their surface area. The segments that have a surface area below $25mm^2$ are merged into a neighboring segment element. Merging priority is given to the smallest segment elements to avoid a single segment from getting too large.

Figure 4.2(c) shows the result of running the segment element subdivision algorithm on the segments in Figure 4.2(b).

The segment elements obtained from the subdivision algorithm can vary between model and range image for the same object, but in general they are very similar due to the structure of the high segment regions. When we utilize the segment elements in the following sections, we will be mindful of the differences that can be present in the subdivision of the segments (between range image and model). Due to the locality of the segment elements, we will calculate location and orientation measures
to provide some more geometric information about the relationship between the segment elements. These measure are in addition to the area-normalized shape index and curvedness described for segments and calculated in the same way for segment elements. The orientation and location are again calculated using a normalization based on surface area associated with each vertex in a segment element:

\[
\text{Location density } L_j = \frac{\sum_{k=1}^{N} a_k \mathbf{v}_k}{\sum_{k=1}^{N} a_k} \quad (4.6)
\]

\[
\text{Orientation density } O_j = \frac{\sum_{k=1}^{N} a_k \mathbf{N}(\mathbf{v}_k)}{\sum_{k=1}^{N} a_k} \quad (4.7)
\]

\(N\) is the number of vertices in the segment element and \(\mathbf{N}(\mathbf{v}_k)\) the normal at vertex \(\mathbf{v}_k\).

### 4.3 Hypothesis Generation

The hypothesis generation procedure determines likely model pose estimates to explain the observed data in the range image. Our procedure for the generation of the hypotheses uses the segments’ shape indices \(SI\) and areas to determine segment similarity. Then, using segment to segment correspondences to limit search, sets of consistent segment element pairs and triples are formed. The set of segment element triples are used with a pose clustering technique to produce sets of pose consistent segment element triples. These sets enumerate the hypotheses to be verified.
Before we enter into a discussion about hypothesis generation we should define and clarify some notation. First denote:

\[
S = \{ S_i : i = 1, \ldots, Ns \} \quad (4.8)
\]

\[
S(m) = \{ S_j(m) : j = 1, \ldots, Ns(m), m = 1, \ldots, M \} \quad (4.9)
\]

\[
SE = \{ SE_a(i) : a = 1, \ldots, Nse(i), i = 1, \ldots, Ns \} \quad (4.10)
\]

\[
SE(m) = \{ SE_a(j, m) : u = 1, \ldots, Nse(j, m), j = 1, \ldots, Ns(m) \} \quad (4.11)
\]

as the set of range segments, model segments, range segment elements, and model segments elements respectively, where \( Ns \) is the number of range segments, \( Ns(m) \) is the number of model segments for model \( m \) and \( M \) is the number of models, \( Nse(i) \) is the number of range segment elements for segment \( S_i \) and \( Nse(j, m) \) is the number of model segment elements for model segment \( S_j(m) \). In the previous section we defined several properties for the segments and segment elements. These segment properties are denoted as follows:

- \( S_i.C \) the curvedness density as given by Equation 4.5,

- \( S_i.SI \) the shape index density as given by Equation 4.4,

- and \( S_i.area \) the area of the segment,

comparable attributes for the segment elements are:

- \( SE_a.C \) the area-normalized curvedness,

- \( SE_a.SI \) the area-normalized shape index,

- \( SE_a.area \) the area of the segment element,
<table>
<thead>
<tr>
<th>Quantization</th>
<th>Index and Shape Description</th>
<th>SI Index Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Convex Ellipsoid</td>
<td>$SI \in [-1, -5/8]$</td>
</tr>
<tr>
<td>1</td>
<td>Convex Cylinder</td>
<td>$SI \in [-5/8, -3/8]$</td>
</tr>
<tr>
<td>2</td>
<td>Weak Convex Saddle</td>
<td>$SI \in [-3/8, -1/8]$</td>
</tr>
<tr>
<td>3</td>
<td>Saddle</td>
<td>$SI \in [-1/8, 1/8]$</td>
</tr>
<tr>
<td>4</td>
<td>Weak Concave Saddle</td>
<td>$SI \in [1/8, 3/8]$</td>
</tr>
<tr>
<td>5</td>
<td>Concave Cylinder</td>
<td>$SI \in [3/8, 5/8]$</td>
</tr>
<tr>
<td>6</td>
<td>Concave Ellipsoid</td>
<td>$SI \in [5/8, 1]$</td>
</tr>
</tbody>
</table>

Table 4.1: Classification of surface shape using Shape Index SI quantizer.

- $SE_a \mathbf{L}$ the area-normalized location as given by Equation 4.6,

- and $SE_a \mathbf{O}$ the area-normalized orientation as given by Equation 4.7.

The segments often represent large areas in the image and on the model; thus, if a region in the range image does not correspond to a large region in a model it will prune a fair amount of the model surface area from being matched in a hypothesis. This is done by first quantizing the segment’s shape index value into a discrete set of classes. Experimentally we preferred to use more classes than listed in Table B.2. For each segment we quantized into seven classes given by Table 4.1.

The quantized shape index value is denoted by $S_i .SIQ$ for the range segments and $S_j(m).SIQ$ for the model segments. Correspondence between $S_i .SIQ$ and $S_j(m).SIQ$ is determined by a similarity map given in Table 4.2. The table implies that for range segments whose shape index quantization value is $S_i .SIQ = 1$, then it is similar to model segments with quantized shape indices $S_j(M).SIQ = 0, 1,$ or $2$. The "fuzziness" in the similarity criteria takes into consideration the noise present in the range images and the imprecision of the segmentation algorithm.

168
<table>
<thead>
<tr>
<th>$s_i.SIQ$</th>
<th>$s_j(m).SIQ$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$\rightarrow 0,1$</td>
</tr>
<tr>
<td>1</td>
<td>$\rightarrow 0,1,2$</td>
</tr>
<tr>
<td>2</td>
<td>$\rightarrow 1,2,3$</td>
</tr>
<tr>
<td>3</td>
<td>$\rightarrow 2,3,4$</td>
</tr>
<tr>
<td>4</td>
<td>$\rightarrow 3,4,5$</td>
</tr>
<tr>
<td>5</td>
<td>$\rightarrow 4,5,6$</td>
</tr>
<tr>
<td>6</td>
<td>$\rightarrow 5,6$</td>
</tr>
</tbody>
</table>

Table 4.2: Similarity mapping between range and model segments for shape index quantization value.
For a range segment $S_i$ and model segment $S_j(m)$ to be considered similar, $S_i.SIQ$ must map to $S_j(m).SIQ$ and the area of the range segment must be less than $S_i.Area < S_j(m).Area + 25mm^2$, where $25mm^2$ is a under-segmentation area bound on the range image segments. Typically, the range image segments include only part of the surface area covered by the corresponding model segments because of occlusion and noise. Occasionally we did notice that some range segments were bigger than their corresponding model segments. This resulted from errors in the calculation of the curvature near object borders.

For each range segment, we obtain a set of corresponding model segments

$$SC_i(m) = \{S_j(m) : j = 1, \ldots, N\text{match}_i(m)\}.$$  

The corresponding sets

$$SC = \{SC_i(m) : i = 1, \ldots, N_s; m = 1, \ldots, M\}$$

define all possible range to model segment correspondence matches for the hypothesis generation procedures.

Once the list of corresponding sets has been found $SC$, further analysis is applied to the segment elements of the corresponding sets to determine a geometrically consistent set of segment element matches. The first step is to determine a good set of segment element pairs in the range image. These seed pairs are defined as a pair of segment elements in a segment that are the farthest distance apart

$$SP_i = \{(SE_a(i), SE_b(i))|\max_{a,b}\|SE_a(i).L - SE_b(i).L\|\}.$$  

This is done for every segment to produce a set of seed pairs

$$SP = \{SP_i : i = 1, \ldots, N_s\}.$$
The set of maximum separated segment elements where chosen to limit the number of similar pairs that are found in the object models (thus improving the pair’s saliency), and to ensure more stable estimates of object pose. Both of these reasons are important for the correct identification and location of the object models in the range image. For a seed pair, we compute

$$SP_i.D = \|SE_a(i).L - SE_b(i).L\|$$

the distance between the pairs and

$$SP_i.O = \cos^{-1}(SE_a(i).\bar{O} \cdot SE_b(i).\bar{O}),$$

the angle between the orientation of the segment elements.

The search for similar model pairs will use the list of $SC_i(m)$ corresponding range to model segments to limit the number of model segment elements. For each $SP_i$, the algorithm searches through the model segment elements $SE_u(j,m)$ where $S_j(m)$ is a segment in $SC_i(m)$. During the search, the algorithm finds all model segment pairs denoted by

$$PM_i = \{P_i(j,m) : t = 1, \ldots, nMatchP(i)\}$$

$$P_i(j,m) = (SE_u(j,m), SE_v(j,m)) : S_j(m) \in SC_i(m)$$

where the pair’s distance and orientation between segment elements agree $\|P_i(j,m).D - SP_i.D\| < Dp$ and $\|P_i(j,m).O - SP_i.O\| < Op$ within a tolerance threshold $(Dp, Op)$ to the seed pair. Currently for our database the thresholds are $Dp = 5mm$ and $Op = \frac{x}{12}$. This results in a list of possible seed to model pair matches

$$PM = \{PM_i : i = 1, \ldots, Ns; m = 1, \ldots, M\}.$$
Each segment element pair can be used to define a local coordinate system using the line between the segment elements and the orientation directions. The coordinate system for a seed pair is given by \( \{SP_i, \vec{x}, SP_i, \vec{y}, SP_i, \vec{z}\} \) where:

\[
SP_i, \vec{x} = \frac{SE_a(i).L - SE_b(i).L}{\|SE_a(i).L - SE_b(i).L\|}
\]

is the vector connecting the two segment elements in the pair normalized to unit length. Let

\[
\vec{b} = \frac{SE_a(i).area * SE_a(i).\vec{o} + SE_b(i).area * SE_b(i).\vec{o}}{SE_a(i).area + SE_b(i).area}
\]

be the area-normalized the orientation of the two segment elements. The area-normalized orientation and the line connecting the two segments are not necessarily orthogonal. To obtain orthogonal coordinate system we will use the Gram-Schmidt orthonormalization procedure, yielding:

\[
SP_i, \vec{z} = \frac{\vec{b} - (\vec{b} \cdot SP_i, \vec{x})SP_i, \vec{x}}{\|\vec{b} - (\vec{b} \cdot SP_i, \vec{x})SP_i, \vec{x}\|}.
\]

The final component of the local coordinate system is obtained from the cross-product:

\[
SP_i, \vec{y} = SP_i, \vec{z} \times SP_i, \vec{x}.
\]

The location of origin of this coordinate system is defined as the midpoint of the line connecting the segment elements:

\[
SP_i, \text{Cen} = \frac{SE_a(i).L + SE_b(i).L}{2}
\]

This procedure is duplicated for each of the seed pairs \( SP_i \in SP \) and model pairs \( P_l(j, m) \in PM \).

Each \( P_l(j, m) \) model pair in \( PM \) gives two possible pose transformations to align the local coordinate system given by the seed pair \( SP_i \), because it is unknown whether
$SE_a(i)$ corresponds to $SE_u(j, m)$ or $SE_v(j, m)$. Without a strong correspondence measure, it is unknown if $SP_i \bar{x}$ corresponds to $P_i(j, m) \bar{x}$ or $-P_i(j, m) \bar{x}$ in the model coordinate system.

The reader should stop and take a moment to reflect on what has been done up to this point. So far we have been using a weak shape and area based similarity measure to correspond range and model segments. Then we defined a set of seed pairs $SP$ using the range image segments and searched through the model database for similar model pairs based on binary distance and orientation constrains. Each pair contains enough information to recover a local coordinate system that could be used to estimate the pose transformation between the object models and the range image. Unfortunately locally many surfaces and segments on surfaces look very similar to one another in our database, so the number of hypotheses that need to be examined is large. To begin to differentiate between the models we will examine the relationships between surface segments. This will be done by forming a triple of segment elements from the seed pairs and a segment element from another segment in the range image.

For scenes that contain only one object this third segment will build a triple of segment elements that can be searched for in our model database. If the model contains a triple with the same distribution of point locations and surface normal distributions then this is a strong indicator that the range to model triple match may be a good estimate of the pose of the object in the image. If the triple is not found, then the range to model pair match is rejected as a possible pose hypothesis. The search for the third segment element in the model data will prune the list of possible seed (range) to model pairs $PM$. If our scene can only contain one object at a time, we could pick one such range image triple to prune our search and identify object
model and pose. This approach was taken by Chua and Jarvis [28]. When more than one object model is present in the range with we do not know if a particular pair of segments $S_i$ and $S_k$ and their corresponding segment elements that form the triple $(SE_a(i), SE_b(i), SE_c(k))$ will lie on the same object without prior knowledge of the scene. To handle this we will encode all possible range segment pairs for each segment $S_i$, some of which will describe pairs of segments on the same object and some that will bridge between two objects.

For a pair of range segments $S_i$ and $S_k$ the seed pair $SP_i$ defines two of the segment elements for a triple. The third segment element $SE_c(k)$ is chosen to maximize the area of the triangle formed by the triple $ST_{i,j} = (SE_a(i), SE_b(i), SE_c(k))$. The larger triangles will minimize the error in the registration estimation method. Denote the seed triples for the range image by:

$$ST = \{ST_{i,k} : i = 1, \ldots, Ns; k = 1, \ldots, Ns; i \neq k\}$$

$$ST_{i,k} = (SE_a(i), SE_b(i), SE_c(k))$$

where $SP_i = (SE_a(i), SE_b(i))$ is the seed pair for segment $S_i$.

Next we wish to search the model database for similar triples. To do this we will be using predictions for the location third model segment element $\hat{L}$ based on the distribution of the triples in the range image $(SE_a(j), SE_b(j), SE_c(k))$ and the local coordinate systems defined by the seed pair $\{SP_i, \bar{x}, SP_i, \bar{y}, SP_i, \bar{z}\}$ and the corresponding model pairs $\{P_l(j, m), \bar{x}, P_l(j, m), \bar{y}, P_l(j, m), \bar{z}\}$.

The prediction of $\hat{L}$ and $\hat{L}'$ for a seed triple $(SE_a(i), SE_b(i), SE_c(k))$ and a model pair $P_l(j, m)$ are given by the following formulas defining the transformation between
coordinate systems. Let
\[
\Gamma_r = [SP, \bar{x}, SP, \bar{y}, SP, \bar{z}]
\]
and
\[
\Gamma_m = [P_l(j, m), \bar{x}, P_l(j, m), \bar{y}, P_l(j, m), \bar{z}]
\]
be matrices containing the axis vectors of the coordinate system from the seed pair and the model pair and
\[
\Gamma'_m = [-P_l(j, m), \bar{x}, -P_l(j, m), \bar{y}, P_l(j, m), \bar{z}]
\]
be the other possible coordinate system for the model pair match, recalling that their is some ambiguity in the correspondence between the segmentation elements. Then the matrices
\[
R = \Gamma_m \Gamma_r,
\]
\[
R' = \Gamma'_m \Gamma_r
\]
and define the possible rotation components of the transformation between the range image to the model coordinate system. The complete transformations of the predicted segment element locations are:
\[
\hat{L} = R \cdot SE_c(k) \cdot L + P_l(j, m) \cdot \text{Cen} - R \cdot SP_i \cdot \text{Cen}
\]
or
\[
\hat{L}' = R' \cdot SE_c(k) \cdot L + P_l(j, m) \cdot \text{Cen} - R' \cdot SP_i \cdot \text{Cen}
\]
in the model coordinate system.

For a seed triple \(ST_{i,k}\) the set of model pairs are searched to see if a set of model triples \(T_{j,m} = (SE_u(j, m), SE_v(j, m), SE_w(j, m))\) can be found such that the distance constraint \(\|SE_w(l, m) \cdot L - \hat{L}\| < Dt\) or \(\|SE_w(l, m) \cdot L - \hat{L}'\| < Dt\) and the
orientation similarity constraint

\[ \| \cos^{-1}(SE_w(l,m), \tilde{O} \cdot P_i(j,m), \tilde{z}) - \cos^{-1}(SE_c(k), \tilde{O} \cdot SP_t, \tilde{z}) \| < Ot \]

are satisfied. The similarity thresholds \( Dt = 7mm \) and \( Ot = \frac{\pi}{6} \) are larger than \((Dp, Op)\) due to some compounding of error from the pair matches. The set

\[ T_M_{i,k} = \{(ST_{i,k}, T_{j,l})\} \]

of model triple segment elements form a set of triple matches with the seed triple \( ST_{i,k} \). The set of all possible triple matches are denoted by

\[ T_M = \{T_M_{i,k} : i = 1, \ldots, N_s; k = 1, \ldots, N_s; i \neq k\}. \]

Each element in \( T_M \) contains a possible pose hypothesis for an object in the range image. The enumeration of \( T_M \) hypothesis are created under the assumption that occlusion and multiple objects are present in the image there will be pose hypotheses in the set that are redundant \((i.e., \) they hypothesize the same object and pose). When an object contains multiple segments on its surface the seed triples that utilize those segments should find similar model triples on the correct model all producing essentially the same model pose estimate for the range image.

To identify these redundancies we will use pose clustering to determine sets of consistent pose triples. The clustering is first done on the rotational pose then on the translation. For each triple match \((ST_{i,k}, T_{j,l}(m))\) in \( T_M \) the transformation between the model coordinate system to the range image is given by:

\[ R_p = \Gamma_r \Gamma_m \]

\[ T_p = ST_{i,k} \cdot \text{Cen} - R_p \cdot T_{j,l}(m) \cdot \text{Cen} \]
\[ST_{i,k}.\text{Cen} = \frac{SE_a(i).L + SE_b(i).L + SE_c(k).L}{3}\]
\[T_{j,l}(m).\text{Cen} = \frac{SE_a(j,m).L + SE_a(j,m).L + SE_a(l,m).L}{3}\]

where the centers of the triples define the location used to define the local coordinate system origins. For each hypothesis, the model’s viewpoint can be determined by:

\[\text{VP} = R_p[0,0,1]^T\]

the product of the rotation matrix for the hypothesis and the z vector. This value gives the location of the rotational pose of the model in the model’s viewspace as defined by the viewsphere. For hypotheses whose pose is consistent, their viewpoints must be close to one another indicating the hypotheses are estimating similar model viewing directions. The viewpoint does not quantify the entire rotational pose of the hypothesis but it should be sufficient to determine within a rotation about the viewing direction if hypotheses are rotationally consistent. To cluster the rotational pose space we will use an \(l = 2\) subdivision of the icosahedron (Table 3.2) to generate a discrete sphere whose vertices are approximately 30 degrees apart. The Voronoi tessellation of the sphere defines the bins we will use to cluster the hypothesis viewpoints \(\text{VP}\). This can be done by using a simple nearest point clustering algorithm to determine which vertex on the discrete sphere is closest to the hypothesis viewpoint. The discrete sphere in effect quantizes are viewspace into coarse viewing directions separated approximately 30 degrees. In order to handle the case where a consistent set of hypotheses are grouping near the Voronoi tessellation cell boundaries, we use the vertices of the Voronoi tessellation to define a new set of bins. The vertices of the Voronoi tessellation are located at the center of the faces defined by the \(l = 2\) subdivision discrete viewsphere. The Voronoi tessellation bins define a redundant
quantization of the viewspace whose decision boundaries are the original tessellation (a Delaunay triangulation) of the viewsphere.

Each hypothesis in $\mathcal{T}_M$ associated with model $m$ has its pose estimate quantized using both the Delaunay and Voronoi vertex quantizers. The set of triples matches that fall into a single bin are considered consistent with respect to the model viewpoint. Once the pose hypotheses have been tested for consistency we will then search through the viewpoint consistent hypotheses to determine yet another set of translational pose consistent hypotheses. This is done by hierarchically clustering the viewpoint consistent hypotheses until the clusters average translation centers are more than $D_{\text{cluster}}$ distance apart. Currently $D_{\text{cluster}} = 12\text{mm}$ for our tests and database. The hierarchical clustering technique results in a group of consistent hypotheses that are all in the same model viewpoint quantizer bin and whose pose translation estimates are within at least $12\text{mm}$ of one another. Denote a set of consistent triple matches as

$$\mathcal{PC}_g(m) = \{(SE_a(i), SE_b(i), SE_c(k)), (SE_a(j, m), SE_v(j, m), SE_w(l, m)), \ldots\}.$$ 

Each set of pose consistent triple matches $\mathcal{PC}_g(m)$ form a hypothesis for object location and orientation in the range image. For each hypothesis the list of range segments used in the hypothesis are found denoted by

$$\mathcal{S}(\mathcal{PC}_g(m)) = \{S_i : SE_a(i) \text{ or } SE_b(i) \text{ or } SE_c(i) \in \mathcal{PC}_g(m)\}.$$ 

This list is then used to determine how much range image surface area the segments used in the hypothesis represent. The pose consistent triples are then prioritized by their total range segment surface area. The hypothesis with the most range segment surface area will be verified first, considering it is trying to explain the most area

178
in the range image and is likely to be an object rather than a random patchwork of segments from several objects.

To review, the hypothesis generation procedure contains the following steps:

1. Generate a set of consistent range image segment to model segment list \( SC \).

2. Generate a set of seed range image segment element pairs \( SP \), one for each range image segment \( S_i \).

3. Generate a set of consistent seed to model pair matches \( PM \).

4. Generate a set of seed triples \( ST \).

5. Generate a set of consistent seed to model triples \( TM \).

6. Generate a set of pose consistent seed to model triples \( PC \). A set of pose consistent seed to model triples \( PC_y(m) \) forms a model pose hypothesis to be verified.

7. Prioritize the hypotheses in \( PC \) by range segment surface area. Each hypothesis \( PC_y(m) \) utilizes a subset of the total range segments and their area indicates how much of the scene will be verified if correct.

### 4.4 Verification

The list of pose consistent hypotheses

\[
PC = \{PC_y(m) : g = 1, \ldots, nC(m), m = 1, \ldots, M \}
\]

contains possible matches between range image segments and model segments and estimates on a model pose to align those segments. The list is ordered based on
the range image segment surface area covered by each hypothesis which gives a good indication on the strength of the hypothesis in explaining the data. In our preliminary experimentation, we found that in most cases the correct hypotheses were in the top of the list of prioritized list \( \mathcal{PC} \). In single object cases often the highest prioritized hypothesis was correct. This is comforting, considering the verification of a hypothesis is a computationally expensive procedure.

The hypothesis \( \mathcal{PC}_g(m) \) may contain multiple sets of corresponding triples \((ST_{i,k}, T_{j,i}(m))\) each of which contains a pose estimate. A combined pose estimate is found using Horn’s closed form [64] quaternion solution to the corresponding point set registration problem. For a triple match in a hypothesis the segment elements

\[
((SE_a(i).L, SE_u(j, m).L), (SE_b(i).L, SE_v(j, m).L), (SE_c(k).L, SE_w(l, m).L))
\]

define the point correspondences between the range image and the model. Then by using all the triple matches a list of point matches are formed

\[
\text{CORR}_{Points} = \{(x_i, p_i) : i = 1, \ldots, N_p\}
\]

where \( N_p \) is the number of corresponding points and \( x_i \) represents segment element locations \( SE_a(i).L \) from the range image and \( p_i \) represents the corresponding segment element location \( SE_u(j, m).L \) on the model.

Using Besl’s notation [12] for Horn’s method [64] the transformation \((R, T)\) that minimizes the error

\[
\text{Err} = \frac{1}{N_p} \sum_{i=1}^{N_p} \| x_i - Rp_i - T \|^2
\]

between corresponding points \( x_i \) and \( p_i \) is given by the following formalization. Let

\[
\mu_p = \frac{1}{N_p} \sum_{i=1}^{N_p} p_i
\]

180
Figure 4.4: Flow diagram for verification of the hypotheses.
\[ \mu_x = \frac{1}{N_p} \sum_{i=1}^{N_p} x_i \]
denote the sample average for the respective point sets. The covariance matrix of the points sets is denoted by

\[ \Sigma_{px} = \frac{1}{N_p} \sum_{i=1}^{N_p} [(p_i - \mu_p)(x_i - \mu_x)^T]. \]

Define a column vector

\[ \Delta = \begin{pmatrix} \Sigma_{px}(1,2) - \Sigma_{px}(2,1) \\ \Sigma_{px}(2,0) - \Sigma_{px}(0,2) \\ \Sigma_{px}(0,1) - \Sigma_{px}(1,0) \end{pmatrix}. \]

The covariance matrix is used to form a 4 × 4 symmetric matrix

\[ Q(\Sigma_{px}) = \begin{pmatrix} tr(\Sigma_{px}) & \Delta^T \\ \Delta & \Sigma_{px} + \Sigma_{px}^T - tr(\Sigma_{px})I \end{pmatrix} \]

where I is a 3 × 3 identity matrix. The eigenvector \( q_R = (q_0, q_1, q_2, q_3)^T \) corresponding to the minimum eigenvector of \( Q(\Sigma_{px}) \) defines the optimal rotation. The 3×3 rotation matrix is given by

\[ R(q_R) = \begin{pmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 + q_2^2 - q_1^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 + q_3^2 - q_1^2 - q_2^2 \end{pmatrix}. \]

Finally the optimal translation vector is found by

\[ T = \mu_x - R(q_R)\mu_p. \]

This least squares solution to the corresponding point registration problem forms the backbone of Besl and McKay's iterated closest point (ICP) registration algorithm summarized in Section 2.4.1 and Figure 2.8.

For a hypothesis \( \mathcal{P}C_g(m) \) the locations of the corresponding segment elements \( \mathcal{C}_O \mathcal{R}_R \mathcal{P}_{\text{point}s} \) are used to define an initial pose transformation \((R, T)\) between the
model $m$ and the range image. After the initial pose transformation has been applied for the model, the Besl and McKay’s ICP algorithm is used to refine the registration between model and range image. During the ICP refinement only the range pixels a part of the hypothesis range segments $\mathcal{S}(\mathcal{PC}_g(m))$ are used to correspond with the model $m$ vertices. These pixels are used because we are hypothesizing that they are apart of the model. The other pixels not part of the hypothesis at this point cannot be used because we are unsure which object they may belong to. The ICP process continues until the difference in registration error $\text{RegErr} = \frac{1}{N_p} \sum_{i=1}^{N_p} ||\mathbf{x}_i - \mathbf{Rp}_i - \mathbf{T}||$ between successive steps of the ICP algorithm falls below 0.05$mm$ or 11 iterations have been tried.

If the registration error between the range image and the model falls bellow an acceptance threshold $\text{RegErr} < T_{\text{accept}}$ the hypothesis is accepted, otherwise it is rejected. Currently the acceptance threshold is $T_{\text{accept}} = 2mm$. When a hypothesis has been accepted, the range image segments are checked to see how well they fit on the accepted model’s surface. If the error between the range segments vertices and the model $m$ is small then the segment is considered part of the model. Most of these range segments are given by the list $\mathcal{S}(\mathcal{PC}_g(m))$ of range segments for the hypothesis, but due to noise and imperfections in segmentation some of them may of been excluded from the hypothesis. The set of range segments that fit well to the model are removed from all the hypotheses in $\mathcal{PC}$. The removal of the range segments may leave a hypothesis with no triples indicating that the hypothesis was trying to provide an alternate explanation for data already explained. The hypotheses still containing some range segments are re-prioritized based on their new total range segment surface areas. Then the next highest priority hypothesis is verified. This

183
continues until all the range segments have been verified by a hypothesis or their are no more hypotheses left to be verified (Figure 4.4).

4.5 Experiments

Our test database of objects consisted of 10 3D polygonal mesh models of toys (Figure 3.20). These object models were generated using the Minolta Vivid 700 non-contact 3D digitizer and Minolta’s model building software [89] from physical prototypes. The models’ polygon counts are listed in Table 3.6. The same sensor is used to test the recognition methods outlined in this chapter. Section 1.1 contains more information about the sensor and range images in general.

The system was tested on a set of 10 range images with each range image containing two objects from our database. The test images are shown in the (a) and (c) sub-figures for Figures 4.6, 4.7, 4.8, 4.9, and 4.10. To capture the images shown, the objects were placed on a turntable and imaged using the Minolta sensor. The images are then stored and loaded into a range editing tool to remove the turntable surfaces from the image (in the future, this step could be automated considering the turntable’s surfaces are at a known location and are planar).

The turntable surfaces were removed to eliminate the surface interaction between the object boundaries and the turntable. The creases formed from these two separate surfaces (object and turntable) introduce a high curved region that does not lie on a single object, rather it results form the interaction between two objects. These highly curved regions (that ultimately result in no object matches) will only add to the computational burden of identifying the real objects. So we chose to pre-mark
the turntable pixels as background in order to remove them from consideration as part of an object in our database.

<table>
<thead>
<tr>
<th>Image In Image</th>
<th>Objects In Image</th>
<th>Correct Detection</th>
<th>RegErr</th>
<th>False Alarms</th>
<th>Missed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scene 1</td>
<td>Rubber Duck</td>
<td>Rubber Duck</td>
<td>1.1mm</td>
<td>Duck Rattle</td>
<td>Whale</td>
</tr>
<tr>
<td></td>
<td>Whale</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene 2</td>
<td>Red Dino</td>
<td>Crocodile</td>
<td>1.6mm</td>
<td>Red Dino</td>
<td>Red Dino</td>
</tr>
<tr>
<td></td>
<td>Crocodile</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene 3</td>
<td>Cow</td>
<td>Cow</td>
<td>0.8mm</td>
<td>Orange Dino</td>
<td>Po</td>
</tr>
<tr>
<td></td>
<td>Orange Dino</td>
<td>Orange Dino</td>
<td>1.1mm</td>
<td></td>
<td>Apple</td>
</tr>
<tr>
<td>Scene 4</td>
<td>Lamb</td>
<td>Lamb</td>
<td>1.7mm</td>
<td>Duck Rattle</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Duck Rattle</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene 5</td>
<td>Po</td>
<td></td>
<td></td>
<td>Orange Dino</td>
<td>Po</td>
</tr>
<tr>
<td></td>
<td>Apple</td>
<td>Orange Dino</td>
<td></td>
<td></td>
<td>Apple</td>
</tr>
<tr>
<td>Scene 6</td>
<td>Po</td>
<td>Po</td>
<td>1.2mm</td>
<td>Orange Dino</td>
<td>Whale</td>
</tr>
<tr>
<td></td>
<td>Whale</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene 7</td>
<td>Rubber Duck</td>
<td>Crocodile</td>
<td>1.1mm</td>
<td>Duck Rattle</td>
<td>Rubber Duck</td>
</tr>
<tr>
<td></td>
<td>Crocodile</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene 8</td>
<td>Cow</td>
<td>Cow</td>
<td>0.87mm</td>
<td></td>
<td>Apple</td>
</tr>
<tr>
<td></td>
<td>Apple</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene 9</td>
<td>Lamb</td>
<td>Duck Rattle</td>
<td>0.83mm</td>
<td>Orange Dino</td>
<td>Lamb</td>
</tr>
<tr>
<td></td>
<td>Duck Rattle</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene 10</td>
<td>Red Dino</td>
<td>Red Dino</td>
<td>1.22mm</td>
<td>Orange Dino</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Orange Dino</td>
<td>Orange Dino</td>
<td>1.58mm</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.3: Recognition results for 10 range image test. Recognition requires correct identification of the model and its pose.

The object scenes contain both object-to-object occlusion and object self occlusion. An example of object-to-object occlusion is shown in Figure 4.6 (c) where the crocodile occludes the right leg and part of the belly of the dinosaur. The effects of self-shadowing can also be seen in almost every image as white gaps between the surfaces of an object. In addition to self-shadowing, another source of structured error
in the range data is the blooming of an object surface at concave ridges. This can be clearly seen on the range image of the rubber duck around the neck (Figure 4.6 (c)). These errors distort the surface information available to our recognition technique and introduce false curvature segment types that diverge from the true types in the database. Even in the presence of such missing and erroneous data, our system has shown success at matching objects.

Images depicting the recognition results can be found in the (b) and (d) sub-figures of the range images shown in Figures 4.6, 4.7, 4.8, 4.9, and 4.10. In the (b) and (d) sub-figures the original image is displayed with the detected objects in their recovered pose. The original objects are shown in light gray while the recovered models are shown in dark gray. The images depict the ability of the recognition system to recover both the object’s identity and its pose in the range image’s coordinate system. The objects whose surfaces seem sort of a patch work of gray and dark gray surfaces typically show a good quality registration result. These patchwork surfaces result from a intermittent classification of the nearest surface in the rendering software. Since both the range image representation of the model and the model’s surfaces are close together they will both appear at various locations due to noise and minor misregistration error. The registration error is also given for each correct detection and is listed as a column in Table 4.3.

The results listed in the Figures 4.6, 4.7, 4.8, 4.9, and 4.10 are obtained using a high curvature detection threshold of \( Th = 0.1 \); this corresponds to a radius of curvature of \( 10 \text{mm} \). All the other thresholds and values are as given in the discussion of the techniques (Section 4.2, 4.3, 4.4). The choice the the high curvature region detection threshold was made based on examination of objects with rich shape variations.
<table>
<thead>
<tr>
<th>Stage</th>
<th>Average time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range to Model Segment Correspondence $\mathcal{SC}$</td>
<td>0.011 sec</td>
</tr>
<tr>
<td>Matching seed pairs to corresponding model pairs $\mathcal{PM}$</td>
<td>0.151 sec</td>
</tr>
<tr>
<td>Matching seed triples to corresponding model triples $\mathcal{T_3}$</td>
<td>9.397 sec</td>
</tr>
<tr>
<td>Pose clustering to determine pose consistent triples $\mathcal{PC}$</td>
<td>3.10 sec</td>
</tr>
</tbody>
</table>

Table 4.4: Timing result averages for the various stages of the hypothesis generation process.

(cow, dinosaurs, and crocodile). These objects contain a rich set of ridges, bumps and pits that were successfully detected using the curvature threshold set at 0.1. As the threshold is moved beyond this point, more and more of the object’s surfaces are included. The more surface area that is considered to be highly curved, the slower our algorithm will run. The slower run times occur because of the increased number of segment elements that have to be searched in the hypothesis generation process. The larger number of segment elements may not necessarily mean an increased chance of detection for an object which already contains a rich segmentation. Only in one instance did the recognition technique fail for these objects at this curvature threshold. This case is shown in Figure 4.6 (b) where the red dinosaur is incorrectly recognized as an orange dinosaur and a red dinosaur with incorrect pose. For the other objects in the database, the curvature threshold may not bring out enough descriptive regions on the object to correctly identify them.

To highlight the subjectiveness of the high curvature threshold we have included segmentations from four of the range images (Figure 4.5). The images show the various curvature segments in different colors. The planar regions in the images are marked by a dark blue and are clearly visible on the side of the lamb which is
<table>
<thead>
<tr>
<th>Image</th>
<th>Hypothesis Generation Time</th>
<th>Verification Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scene 1</td>
<td>13.28 sec</td>
<td>35.86 sec</td>
</tr>
<tr>
<td>Scene 2</td>
<td>16.37 sec</td>
<td>169.48 sec</td>
</tr>
<tr>
<td>Scene 3</td>
<td>22.06 sec</td>
<td>21.25 sec</td>
</tr>
<tr>
<td>Scene 4</td>
<td>17.04 sec</td>
<td>435.45 sec</td>
</tr>
<tr>
<td>Scene 5</td>
<td>4.92 sec</td>
<td>74.96 sec</td>
</tr>
<tr>
<td>Scene 6</td>
<td>15.68 sec</td>
<td>11.44 sec</td>
</tr>
<tr>
<td>Scene 7</td>
<td>6.15 sec</td>
<td>12.88 sec</td>
</tr>
<tr>
<td>Scene 8</td>
<td>6.99 sec</td>
<td>7.86 sec</td>
</tr>
<tr>
<td>Scene 9</td>
<td>2.67 sec</td>
<td>68.6 sec</td>
</tr>
<tr>
<td>Scene 10</td>
<td>22.55 sec</td>
<td>15.04 sec</td>
</tr>
<tr>
<td>Average</td>
<td>12.77 sec</td>
<td>85.28 sec</td>
</tr>
</tbody>
</table>

Table 4.5: Timing for the hypothesis generation and verification for each of the 10 range images test set.

planar. These regions also show up near the border of the object since the curvature values in these regions have been removed with the erosion operation mentioned in Section 4.2. The images show that for some of the objects (like the apple and the whale), very little surface area is labeled as highly curved. This yields in only a few descriptive features for the hypothesis generation technique to utilize and determine its uniqueness. Currently our algorithm needs at least two segments to generate a triple and a possible hypothesis for the object. In order to better handle these less curved objects the threshold for high curvature regions should adaptively change to the curvature content of the object or a fixed set of interesting curvature scales should be applied. We prefer the latter due to the high scales being selective for some of the objects and less computationally expensive to match, while the lower curvature scales could be used to recover the objects with lower curvature content after removal of the more highly curved objects.
Some timing results for the various stages of our recognition system are shown in Tables 4.4 and 4.5. These tests were run on a Micron Pentium II based PC under the Windows NT 4.0 operating system. In Table 4.4, we show the completion time for the various stages of the hypothesis generation procedures averaged over all the test images. From the table, we can see that matching of the seed triples to consistent model triples took the most time (9.39 seconds). Table 4.5, shows the time taken to complete the hypothesis generation and object verification stages for every range image. The large variation in the verification stage completion time results from the differing number of hypotheses generated from the images and the priority of the correct hypotheses in $\mathcal{P}C$. The image of the toy lamb and duck rattle (Figure 4.7 (c)) took the most time to complete the recognition procedure. In this case, the verification stage iterated through 18 hypotheses before the lamb was selected, and then iterated through the rest of the remaining hypotheses before determining that none of them explained the toy duck rattle. This exhaustive search required before rejecting all the explanations is very time consuming. When we examine the hypotheses rejected before the correct lamb hypothesis is examined, we find that the list contained many hypotheses of the lamb model, but for similar poses to the one that was finally accepted. These other lamb hypotheses were rejected because the registration error $RegErr$ was too high to yield high confidence in the match. The priorities of all of these lamb hypotheses are essentially the same since they cover the same high curved range image surface area. It might be possible to speed up the time to recognize and verify objects by ranking the hypotheses for an object based on an initial range image to model error. The initial error measure should place the correct hypothesis before the others even in the presence of similar range image surface area.
The recognition results shown in Figures 4.6 (b), (d), 4.8 (b), (d), 4.9 (b), 4.10 (b) all show cases of false alarms. In these cases an incorrect object was matched to portions of a range image and accepted based on the registration error $RegErr$ between the range image highly curved segments and the model. These highly curved segments are selected based on the hypothesis generation process. In all these cases, the highly curved regions selected in the hypothesis locally matches to the range data, but their less curved regions do not. These results seem to indicate that a major failing of our technique is the criteria of acceptance in the hypothesis verification stages of our algorithm.

We propose using rendering techniques to re-render the object at the hypothesized location and orientation. Then the surfaces of the rendered model will be compared with the range image. This process should point out the large discrepancies between the proposed model and range image matches (For example, the toy orange dinosaur hypothesis for the whale range data in Figure 4.8 (d)). In this case, the rendering-based verification technique would point out the absence of a large potion of the orange dinosaur’s surfaces that should be visible if the object is truly at this location. This improved verification scheme should decrease the methods false alarm rate.

4.6 Conclusion

In this chapter, we have suggested and demonstrated a new local feature based recognition technique for free-form objects. This method utilizes the surface structure in the highly curved regions of the objects to determine whether an object is present in an image and, if so, where it is located. The algorithm is built upon an initial surface segmentation to select and classify the high curved regions of the images and
Figure 4.5: Segmentation results for four of the range images used to test the system. (a) Range image of a toy lamb and duck rattle. (b) Range image of toy Po doll and apple. (c) Range image of toy Po doll and whale. (d) Range image of toy lamb and duck rattle.
Figure 4.6: Recognition results-The range image surfaces are shaded gray, while the recognized models are shaded green and placed in the range image with their recovered pose. (a) Range image of a toy rubber duck and whale. (b) Recognition results for toy rubber duck and whale range image. (c) Range image of a toy red dinosaur and crocodile. (d) Recognition results for toy red dinosaur and crocodile range image.
Figure 4.7: Recognition results-The range image surfaces are shaded light gray, while the recognized models are shaded dark gray and placed in the range image with their recovered pose. (a) Range image of a toy cow and orange dinosaur. (b) Recognition results for toy cow and orange dinosaur range image. (c) Range image of a toy lamb and duck rattle. (d) Recognition results for toy lamb and duck rattle range image.
Figure 4.8: Recognition results: The range image surfaces are shaded light gray, while the recognized models are shaded dark gray and placed in the range image with their recovered pose. (a) Range image of a toy Po doll and apple. (b) Recognition results for toy Po doll and apple range image. (c) Range image of a toy Po doll and whale. (d) Recognition results for toy Po doll and whale range image.
Figure 4.9: Recognition results-The range image surfaces are shaded light gray, while the recognized models are shaded dark gray and placed in the range image with their recovered pose. (a) Range image of a toy rubber duck and crocodile. (b) Recognition results for toy rubber duck and crocodile range image. (c) Range image of a toy cow and apple. (d) Recognition results for toy cow and apple range image.
Figure 4.10: Recognition results- The range image surfaces are shaded light gray, while the recognized models are shaded dark gray and placed in the range image with their recovered pose. (a) Range image of a toy lamb and duck rattle. (b) Recognition results for toy lamb and duck rattle range image. (c) Range image of a toy red dinosaur and orange dinosaur. (d) Recognition results for toy red dinosaur and orange dinosaur range image.
object models. These large uniform surface type segments are divided up into much smaller segments called segment elements. The segment elements then define a basic unit of the surface that will be used in the hypothesis generation stages of the method.

The hypothesis generation procedure was created to exploit the structure of the high curved regions of the object’s surfaces and to prune the search process. This method uses a coarse classification of segment type to limit the search through the highly curved regions. Next the system used a pairwise binary constraint to form likely model and pose estimates. These pairwise matches are not descriptive enough and result in far too many equally likely hypotheses. To prune the search and to add some segment to segment structure information a triple is formed between two segments. The triples reduce the number of possible pose hypotheses adding additional surface structure. Pose clustering is then used to discover the redundancy in the pose estimates of the triples. Triples of consistent pose are then ranked based on their range segment surface area. The hypotheses with the highest range image surface area are verified first. A hypothesis is verified if the registration error between the range segments and the model are below a threshold after Besl and McKay’s iterated closest point pose refinement algorithm has been run.

The experiments suggest that the following improvements can be made to the recognition system.

1. A render based verification scheme, that checks for consistency between model and image for more than just the high curved surface areas, should be developed.

2. Once an object has been verified, the pixels that are consistent with the model should be removed from further consideration.
3. A multi-scale technique should be used to improve the recognition of less highly curved objects.

4. The hypotheses $PC$ should be prioritized by an initial registration match score, along with the already present range image surface area. This should increase the likelihood of the correct model being at the top of the verification list.
CHAPTER 5

Range Image and Object Model Database

5.1 Introduction

Model-based three-dimensional object recognition continues to be a popular and productive area of research in the computer vision community. The different assumptions inherent in designing a complete system have spawned many original systems and the subsequent development of variants and improvements [10, 2]. Object recognizers employing dense range maps have been particularly popular systems for experimental computer vision researchers to construct. Recent drops in the prices of range sensors and dramatic improvements in the processing power and memory capacity of research workstations have combined to make experimentation with real (as well as synthetic) range data and historically-slow recognition strategies approachable, even in research labs with significantly limited resources.

At the 1991 NSF-sponsored workshop entitled “Future Directions in Computer Vision Research” [99], computer vision methodology (or lack of same) received focused

---

attention, and a series of articles in CVGIP: Image Understanding [68] has hopefully prompted researchers to standardize those parts of their experiments, systems, and data where such standardization is warranted. While improvement of the situation requires activity on many fronts, one theme that has been sounded repeatedly in recent years is the value of comparisons against standard data sets. Such comparisons allow the strengths and weaknesses of competing techniques to be highlighted, as well as demonstrating the domain of applicability of the methods under study (promoting the ‘where does it break?’ question asked of researchers describing implementations and algorithms by reviewers and conference presentation attendees). There are signs that this need is beginning to be addressed by researchers, as evidenced by the use of standardized databases in applications as diverse as stereo matching [17], character recognition [66], and (most relevant to this work) range image segmentation [59].

In this article, we describe a database of range images and 3D objects constructed over an nine-year period at two institutions, which was designed to facilitate research in model-based 3D object recognition. Almost by accident, this database (freely available over the Internet) has become a popular source of imagery for evaluation of range image segmentation algorithms. Dissemination and use of this database will allow for realistic comparative studies as well as a source of test data for development of new techniques for range image analysis and understanding.

The remainder of this paper is organized as follows. Section 5.2 describes the overall organization of the database, along with some commentary about the history of its construction. Section 5.4 describes two techniques for accessing the database and suggests a standard scheme for crediting the authors in publications employing the database. Section 5.5 presents some final comments.
5.2 Database Contents and Organization

At present, the database has two main components: an archive of over 250 3D object models, and an archive of over 400 real and synthetic range images. New items are added to the database regularly and documented on its home page. Access statistics for the database are incomplete but the available data indicates a minimum of 100 downloads of one or more items from the database each month.

5.2.1 Object Model Database

The object model database contains descriptions of 3D objects in a variety of formats. Different descriptions of the same object are employed for different purposes in our model-based object recognition system. For example, each object in the database is described as a polyhedron, which for objects with curved surfaces is merely an approximation to the true object shape. This polyhedral description is useful for synthesizing images of the object, both to provide data sets for system testing and to provide images of hypotheses for verification [42]. Many objects are described in terms of the types and parameters of their constituent surfaces; this description is useful during recognition as a source of surface primitives to correspond to surfaces extracted from range image segmentations. Each file format is described via example below.

There are presently three sub-archives in the model database:

1. The **MSU-Ideas** database contains descriptions of twenty 3D objects used in testing of the BONSAI and IFI object recognition systems developed by Flynn and Jain [42, 46]. These objects were originally designed in 1989 and 1990
with the I-deas solid modeler sold by Structural Dynamics Research Corporation. A subsequent redesign phase employed the IRIT public-domain solid modeler\(^7\) written by Gershon Elber (of the University of Utah and Technion). Object shapes are generally simple and piecewise-planar, cylindrical, spherical, or conical. Objects in this database are described in four formats:

(a) IRIT solid modeler input.

(b) IGES 3.0 formatted data.

(c) An ASCII polyhedral approximation.

(d) An ASCII file of surface types and parameters.

2. The **USF** database contains descriptions of 81 polyhedral objects designed by students and faculty at the University of South Florida. A custom solid modeler developed at USF was used to fabricate these models. Each object is presented in two formats:

(a) An ASCII polyhedral description.

(b) An ASCII file of surface types and parameters (in this database, all object surfaces are planar).

3. The **NETLIB** database contains descriptions of 119 polyhedral objects obtained from the NETLIB scientific data repository. Each object is presented in two formats:

(a) An ASCII polyhedral description.

\(^7\) [http://www.cs.technion.ac.il/~gershon/iris/home/iris_jhome.html](http://www.cs.technion.ac.il/~gershon/iris/home/iris_jhome.html)
(b) An ASCII file of surface types and parameters (in this database, all object surfaces are planar).

4. The WSU-PRO/E database contains descriptions of seven objects constructed from drawings in mechanical design texts. The models were designed using the Pro/Engineer CAD package sold by Parametric Technologies, Inc. and are currently presented only as polyhedral approximations. We are currently designing more models and making surface descriptions available.

Figure 5.1 shows a synthetic range image of an object from each of the four databases. Figure 5.2 shows excerpts from the four files used to describe the ‘curvblock’ object in the MSU-Ideas database. As mentioned above, the file formats most useful to our object recognizers are the polyhedral approximation (i.e., the .poly format, Figure 5.2(c)) and the neutral surface description (i.e., the .neutral format, Figure 5.2(d)).

5.2.2 Range Image Database

The range image archive consists of several sub-archives.

- The Synthetic sub-archive contains five synthetic range images of each of the models in the MSU-Ideas database described above (for a total of one hundred images). These images were generated from the polyhedral approximations (.poly files) accompanying each of the MSU-Ideas models. Software to generate these synthetic range images is also available in the archive and is described below.
• The **Isolated** sub-archive contains five real range images (taken with a Technical Arts 100x range sensor) of each of the twenty objects whose models are in the MSU-Ideas model database (for a total of 100 images).

• The **Cluttered** sub-archive contains ten real range images (taken with the 100x range sensor), each of a scene containing two of the objects in the WSU-Ideas object database.

• The **Miscellaneous** sub-archive contains 47 images taken with the 100x range sensor, containing one or more miscellaneous objects.

• The **seg-comp** sub-archive contains 80 images used in the segmentation comparison project conducted at the University of South Florida, Washington State University, the University of Bern, and the University of Edinburgh since 1993 [59]. Forty of the images were taken at Bern with a structured light scanner built by ABW Gmbh and forty images were taken at Oak Ridge National Laboratory with an Odetics Perceptron laser scanner.

• The **USF** sub-archive contains 19 range/reflectance image pairs taken with the Odetics Perceptron laser range finder at Oak Ridge.

### 5.2.3 Range Image Rendering Software

The archive also contains the source code and supporting libraries for a program that synthesizes range images from the polyhedral approximations (.poly files) mentioned above. The program currently available is somewhat limited in that it centers the object in the synthetic aperture, scales the coordinates appropriately to fill the image with the object, and specifies the object orientation by an index between 0 and
319. The index identifies one of 320 viewpoints which are drawn from the centers of the triangles comprising a 16-frequency subdivision of the icosahedron. The program requires the Mesa software library (which is a freely available implementation of most of the OpenGL graphics library) and a pointer to the home site for Mesa is provided in the database description.

5.3 Impact and Lessons Learned

The primary impact of a database such as the one described here on its research community is demonstrated by the its frequency of appearance in the literature. A brief examination of recent issues of IEEE Trans. on PAMI revealed several uses of database images, including these two uses.

- Dickinson et al. [34] used Technical Arts 100X images as input to a system to derive part models for the objects in the images.

- Sun and Sherrah [134] used images as input to a procedure that identifies symmetries in objects.

Images in the database have also been used in the literature to evaluate range image segmenters. The object models in the database have not been used to any great extent (this is perhaps due to the proliferation of free and inexpensive databases of CAD models developed for use in computer graphics and animation applications). Therefore, the record of success of this database is mixed. It is reasonably well-known and well-regarded as a source of range imagery, but has had little impact at a level above segmentation.

In retrospect, the design, updating and ‘marketing’ of the database might have been done differently to enhance its value to the community. Inexpensive CD-ROM
writers and media as well as inexpensive high-capacity rewritable disks have impacted and are continuing to impact the for-free and for-profit approaches to selling data. More publicity about databases such as the 3D database also raises their profile in the community and the recent and continuing activities to improve the experimental rigor of computer vision research will help reinforce their value.

5.4 Access and Citation

The ‘top’ of the 3D model/image database is available at the following URL:


The archive is organized as a tree; at the top level the visitor can elect to browse the object models, the range images, or the rendering software. Each leaf directory’s contents (e.g. a group of related object models or images, or the entire rendering program source code) is available as a compressed UNIX tar file, and every individual file may also be browsed or downloaded separately. The several WWW pages associated with this database will be instrumented with programs to keep track of the database’s usage; this will help to guide future development of the archive.

5.5 Conclusions

In this short article, we have described a nine-year-old archive of data used in model-based 3D object recognition. As the archive matures, additional items are begin added and its value to the computer vision research community should continue to increase. Ultimately, the success of this archive depends on the willingness of researchers to use it; this was the basis of our decision to make the data freely available over the Internet and World Wide Web.
Figure 5.1: Synthetic range images of objects from the four databases in the object model database. (a): MSU-Ideas database. (b): USF database. (c): NETLIB database. (d): WSU-PRO/E database.
Figure 5.2: Four descriptions of the MSU-Ideas curvblock object. (a): IRIT (.irt) file. (b): IGES file (excerpts). (c): Polyhedral approximation (.poly file, excerpts). (d): surface descriptions (.neutral file).
CHAPTER 6

Conclusion

This dissertation included an extensive survey (Chapter 2 of the construction and recognition of three-dimensional object models, with primary emphasis on range data but with some mention of intensity techniques as context.

In recent literature we have seen an increased use of computer vision techniques to improve the accuracy and speed of the surface registration problem used in reverse engineering complex 3D models from examples [29],[73]. With out automated registration techniques from local features the quality of these models presently depends on the skill of the person monitoring and manipulating the software packages used to form the model. The designer must currently be constantly on the watch for erroneous data (errors in measurement by the sensor), errors in the registration process, and the integrity of the mesh after integration. Each one of these tasks may and often does fail when there are errors made in the measurement of the object and/or the position and amount of overlap between surfaces. In the future the methods for registration, integration and optimization of polygonal meshes need to be tested on a large number of standardized objects with varying geometry so the strengths and weakness can be cataloged for each technique. One of the purposes of this dissertation work was to provide a web-resource of range data and polygonal mesh models for other researches.
to compare their work (Chapter 5). This will be a valuable resource that allows the community to build on the knowledge of the previous researchers without wasting time and effort in rediscovering what has already been done.

The object recognition problem involves a study of salient features and their identification, as well as a study of control and data structures that yield efficient recognition techniques. The problem of finding and identifying objects in single object scenes with no occlusion has been well studied and many systems designed that show good results [4, 15, 36, 67, 95, 98, 142]. In these systems a fair amount of information is present about the object. This allows for the design of systems which are able to identify objects under noisy conditions except in cases where views of the objects are too similar.

We added to this body of work by demonstrating the applicability of appearance-based recognition techniques to recognize objects in range images (Chapter 3). This work also demonstrated the feasibility of using realistic rendering techniques to train the recognition database. This allowed us great flexibility to planning what views to capture in training the database, which in turn allowed us to study increasing the pose uncertainty from a normally concerted 1D variation to full 3D rotational pose variation. We also introduced a more intelligent selection for the "background pixel" value (not object) which allowed improved our methods ability to recognize object in the presence of moderate amounts of segmentation errors in the range images of our objects. For tests run on a ten object database we were able to correctly identify the objects 97% of the time.

The problem of finding and identifying multiple objects in scenes with the possibility of occlusion and background clutter is a much harder problem [14, 24, 29, 55,
57, 75, 84, 107, 131]. In general, the partial information about an object makes the recognition less reliable and more complex because the information can be incomplete and disjoint. This is in part due to the decreased saliency of local measures over global measures. Locally many regions of a surface or many regions of surfaces on different objects may appear similar. Using the information in these regions by themselves is a poor choice for recovering the identity and location, but together by using the relationships between features the identity and location of an object may still be obtained.

In this dissertation we also presented a technique to recognize free-form objects in images where only partial information about the object is known either because of occlusion or the camera was zoomed into the object to densely sample a section of the object. Our method exploits the structure present in the high curved regions on an object surface. These high curved regions contain important landmarks for differentiating objects from one another. We showed that by using a simple surface classification and segmentation technique in these high curved regions we could reliability extract our areas of interest. The large homogeneous segments are then divided into segment elements each of which contains a much more local estimate of the object structure. These local estimates are required because the presence of occlusion will obstruct portions of the segments from being recovered in the range images, while the segmentation elements have a improved chance of being recovered consistently between model and range image segmentations. Our recognition technique builds upon these base segment and segment elements by using their structure and location to determine likely model pose hypotheses. Redundant pose hypotheses are discovered using a pose clustering technique. These redundant hypotheses are combined to form
a single hypothesis are is run through a pose refinement step before being accepted or rejected based on its registration error.

For both single object and multiple object scenes large database studies need to be applied. As the size of the database increases the importance of a systems method for quickly and accurately indexing to the correct model become more important. Large multimedia databases and flexible manufacturing inspection and assembly systems are examples of applications where quick indexing of image/model databases are becoming important.

An emerging area of study for new object recognition systems is to combine multiple imaging modalities to determine object identity and location. One example is to use both depth (range) and color information from many modern 3D digitizers common to the computer graphics and computer vision fields. The textural information from the color images may allow for discrimination between objects in cases where their shape is similar while their texture is different, likewise cases where the textural information does not discriminate between objects well, their shapes may.

Another area likely to increase in importance is deformable object modeling and understanding. Techniques to deal with non-rigid object matching and motion are an important field especially in the areas of medical imaging. Here the individuality of a patient may cause ridge techniques problems because the surface geometry of an organ or other parts varies from person to person. Even in the case of maintaining a health history of a single patient the parts can change due to swelling or aging. Therefore what may be more interesting for non-rigid objects is to study their defining characteristics so that time-dependent shapes may be represented and recovered faithfully.
APPENDIX A

Karhunen-Loève expansion for images

Consider a training set of $M$ images $\mathbf{X} = \{ \mathbf{x}_1, \ldots, \mathbf{x}_M \}$ where $\mathbf{x}_i$ is an training image whose elements are in raster scan order. Let $\mathbf{x}_i \in \mathbb{R}^N$ where $N$ is the number of pixels in the images. Note since $\mathbb{Z}^N \subset \mathbb{R}^N$ this formalization also applies for grayscale imagery.

The Karhunen-Loève expansion [93, 50, 96] is found by first calculating the covariance matrix of the training data. To do this the average $\mathbf{c} = \frac{1}{M} \sum_{i=1}^{M} \mathbf{x}_i$ image is subtracted from every image $\mathbf{x}_i$ in $\mathbf{X}$ yielding a centered set $\mathbf{X}_c = [\mathbf{x}_1 - \mathbf{c}, \ldots, \mathbf{x}_M - \mathbf{c}]$. Then $\mathbf{X}_c$ is used to form the $N \times N$ covariance matrix: $\Sigma = \mathbf{X}_c \mathbf{X}_c^T$. Next the set of $L$ eigenvalues $\lambda_i$ and eigenvectors $\mathbf{e}_i$ of the covariance matrix $\Sigma$ are calculated where $L \leq \text{min}(M, N)$ the rank of the covariance matrix. The eigenvalues and corresponding eigenvectors are ordered so that $\lambda_1 < \lambda_2 < \cdots < \lambda_L$. The set of eigenvectors $\{ \mathbf{e}_i : i = 1, \ldots, L \}$ forms a new orthonormal basis that can be used to describe the training images $\mathbf{X}$.

Next project the training data onto the new basis:

$$
\mathbf{\tilde{g}_i} = \begin{pmatrix}
\mathbf{e}_1^T \\
\vdots \\
\mathbf{e}_L^T
\end{pmatrix}
(\mathbf{x}_i - \mathbf{c}).
$$

This operation is often referred to as the Karhunen-Loève Transformation.
Finally the Karhunen-Loève expansion of the images is given by

\[ \mathbf{x}_i = ([\mathbf{e}_1, \ldots, \mathbf{e}_L] \mathbf{g}_i) + \bar{c}. \]

Murakami and Kumar [93] showed a more efficient calculation of the eigenvectors \( \mathbf{e}_i \) and eigenvalues \( \lambda_i \) for the covariance matrix \( \Sigma \) for cases where \( M < N \) (the number of training images are less than the number of pixels).

Define a new \( M \times M \) matrix:

\[ \mathbf{A} = \mathbf{X}_c^T \mathbf{X}_c \]

where the elements of \( a_{i,j} = (\mathbf{x}_i - \bar{c})^T(\mathbf{x}_j - \bar{c}) \) are the correlation between pairs of centered training images.

Then find eigenvalues and eigenvectors \( \{\lambda_i, \bar{a}_i : i = 1, \ldots, L\} \) of \( \mathbf{A} \). The eigenvalues for \( \mathbf{A} \) are the same as the eigenvalues for \( \Sigma \) [93] and the eigenvectors \( \mathbf{e}_i \) of \( \Sigma \) are related to the eigenvectors \( \bar{a}_i \) of \( \mathbf{A} \) by:

\[ \mathbf{e}_i = \frac{\mathbf{X}_c \bar{a}_i}{M \lambda_i}. \]
APPENDIX B

Differential Geometry

The material and notation presented here is based on the differential geometry chapters in Gerald Farin’s book on curves and surfaces [41].

B.1 Space Curves

Define a space curve \( \mathbf{r} \) as a vector valued function:

\[
\mathbf{r}(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix}
\]

where \( \mathbf{r}(t) \) is smooth (\( \mathbf{r}'(t) \neq 0 \) and \( \mathbf{r}''(t) \neq 0 \) \( \forall t \in [a, b] \)).

The arc length of the curve from \( a \) to any point \( t \in [a, b] \) is:

\[
L(t) = \int_{a}^{t} \|\mathbf{r}'(t)\| \, dt
\]

\[
= \int_{a}^{t} \sqrt{(x'(t))^2 + (y'(t))^2 + (z'(t))^2} \, dt. \tag{B.1}
\]

But utilizing the Fundamental Theorem of calculus the rate of change of the arc length is given by:

\[
L'(t) = \|\mathbf{r}'(t)\|.
\]

Then we can define the tangent to the curve \( \mathbf{r}(t) \) as:

\[
\mathbf{T}(t) = \frac{\mathbf{r}'(t)}{\|\mathbf{r}'(t)\|} = \frac{\mathbf{r}'(t)}{L'(t)}
\]
which is the rate of change of the curve with respect to \( t \) over the rate of change of the curve’s arc length with respect to \( t \). The magnitude of the tangent vector \( \mathbf{T}(t) \) is unity. The arrow above a vector will be used only to denote unit vectors.

Recall this corollary from calculus: if \( F(t) \) is differentiable and \( \|F(t)\| = c \) where \( c \) is a constant, then \( F(t) \cdot F'(t) = 0 \).

Define the normal to the curve \( \mathbf{r}(t) \) as:

\[
\mathbf{M}(t) = \frac{\mathbf{T}'(t)}{\|\mathbf{T}'(t)\|}.
\]

the normalized rate of change of the tangent with respect to the parameter \( t \). Utilizing the corollary mentioned above it can be shown that \( \mathbf{M}(t) \cdot \mathbf{T}(t) = 0 \).

The set of unit vectors \( \{\mathbf{T}(t), \mathbf{M}(t), \mathbf{T}(t) \times \mathbf{M}(t)\} \) define a local coordinate system for every value of \( t \). The direction of the third coordinate axis is labeled the Binormal vector \( \mathbf{B}(t) = \mathbf{T}(t) \times \mathbf{M}(t) \). This new coordinate system \( \{\mathbf{T}(t), \mathbf{M}(t), \mathbf{B}(t)\} \) is called a Frenet Frame.

The notation above was developed in terms of an arbitrary parameter \( t \) for convenience and generality. To generate a more meaningful result that does not depend on parameterizations, the forms will all be re-parameterized with respect to a new variable \( s \) which corresponds to arc length from \( a \) (the start of the curve) to location on the curve of interest. The formula relating the arc length parameter and \( t \) is given by:

\[
s(t) = \int_a^t \|\mathbf{r}'(t)\|dt.
\]

Using the expressions for \( \{\mathbf{T}(s), \mathbf{M}(s), \mathbf{B}(s)\} \) re-parameterized for curve length and taking their partial derivatives with respect to the length parameter results in
the the Frenet-Serret formulas:

$$\frac{\partial\vec{T}(s)}{\partial s} = \kappa \vec{M}(s) \quad (B.3)$$

$$\frac{\partial\vec{M}(s)}{\partial s} = -\kappa \vec{T}(s) + \tau \vec{B}(s) \quad (B.4)$$

$$\frac{\partial\vec{B}(s)}{\partial s} = -\tau \vec{M}(s). \quad (B.5)$$

$\kappa$ and $\tau$ are the curvature and torsion of the curve $\vec{r}(s)$. Solving for $\kappa$ and $\tau$ we obtain:

$$\kappa(s) = \vec{M}(s) \cdot \frac{\partial\vec{T}(s)}{\partial s} \quad (B.6)$$

$$\tau(s) = -\vec{M}(s) \cdot \frac{\partial\vec{B}(s)}{\partial s}. \quad (B.7)$$

Curvature and torsion have an intuitive geometric meaning related to the rate of change of the tangent $\vec{T}(s)$ and binormal vectors $\vec{B}(s)$. Denote

$$\Delta \alpha = \cos^{-1}(\vec{T}(s) \cdot \vec{T}(s + \Delta s))$$

as the angle between tangent vectors separated by a small arc length segment $\Delta s$. Similarly denote

$$\Delta \beta = \cos^{-1}(\vec{B}(s) \cdot \vec{B}(s + \Delta s))$$

as the angle between binormal vectors separated by a small arc length segment $\Delta s$. Then by using a Taylor’s expansion on $\Delta s$ terms and taking the limit as $\Delta s \to 0$ it can be shown that

$$\kappa = \frac{\partial \alpha}{\partial s} \quad (B.8)$$

$$\tau = -\frac{\partial \beta}{\partial s} \quad (B.9)$$

where now curvature and torsion are now defined as the angular velocities of the tangent and binormal vectors.
Another tool to analyze the curvature of a curve is by considering a circle that intersects the curve at a point and whose first and second derivatives are the same as those of the curve \( \bar{r}(s) \). The circle is called an osculating circle, because it’s center and radius changes as the curve is traced out. Consider the center of the circle tangent to the curve

\[
\bar{c} = \bar{r}(s) + \rho \bar{M}(s)
\]

where \( \rho \) is the radius of the osculating circle and \( \bar{M}(s) \) is the normal to the curve \( \bar{r}(s) \). Using a Taylor expansion on the equation for \( \bar{r}(s) \) and taking the limit as \( \Delta s \to 0 \) it can be shown that the radius of the circle is \( \rho = \frac{1}{\kappa} \). Because of this \( \rho \) is often referred to as the radius of curvature. As a conceptual tool the radius of curvature is useful in giving an intuitive feel for the magnitude and sign of curvature.

### B.2 Surface Curvature

Surface curvature is defined from a generalization of the space curve formulation given above. First define a surface using its parametric form:

\[
\bar{r}(u, v) = \begin{pmatrix} x(u, v) \\ y(u, v) \\ z(u, v) \end{pmatrix},
\]

where \( u, v \in [a, b] \). An arbitrary curve on the surface can be specified by \( \bar{r}(\bar{p}(t)) \), where \( \bar{p}(t) = \begin{pmatrix} u(t) \\ v(t) \end{pmatrix} \). In order to utilize the material presented in the previous section the functions for \( u(t) \) and \( v(t) \) must be smooth so that the corresponding curve defined by \( \bar{r}(p(t)) \) is also smooth.
B.2.1 First Fundamental Form

Using the chain rule, the first partial derivative of the surface curve on the surface with respect to $t$ is:

$$
\vec{r}' = \frac{\partial \vec{r}}{\partial u} \frac{\partial u}{\partial t} + \frac{\partial \vec{r}}{\partial v} \frac{\partial v}{\partial t}.
$$

Using $\vec{r}_u$ as short hand notation for $\frac{\partial \vec{r}}{\partial u}$ the first partial of the curve

$$
\vec{r} = \vec{r}_u u' + \vec{r}_v v'.
$$

Recall that for a curve

$$
\frac{\partial s}{\partial t} = \|\vec{r}'\|
$$

then the squared arc element for the surface curve is:

$$
\partial s^2 = \|\vec{r}'\|^2 \partial t^2
$$

(B.10)

$$
= (\vec{r}_u'^2 + 2\vec{r}_u \vec{r}_v u' v' + \vec{r}_v'^2) \partial t^2
$$

(B.11)

which can be written as

$$
\partial s^2 = E \partial u^2 + 2F \partial u \partial v + G \partial v^2
$$

where $E(u, v) = \vec{r}_u \vec{r}_u, F(u, v) = \vec{r}_u \vec{r}_v, \text{ and } G(u, v) = \vec{r}_v \vec{r}_v$. This equation is referred to as the First Fundamental Form.

The parametric surface differential area element is:

$$
\partial A = \|\vec{r}_u \times \vec{r}_v\| \partial u \partial v.
$$

The key to showing this result employs the form for finding the area of a parallelogram. Terminology used in this context includes the *discriminant* which is denoted as:

$$
D = \|\vec{r}_u \times \vec{r}_v\| = \sqrt{EG - F^2}.
$$

219
Then the area of the surface is:

\[ A = \int \int \sqrt{EG - F^2} \, du \, dv. \]

The tangent plane to the surface is defined by the two tangent vectors \( \vec{r}_u, \vec{r}_v \). Then the normal of the surface at \((u, v)\) is:

\[ \vec{N}(u, v) = \frac{\vec{r}_u \times \vec{r}_v}{||\vec{r}_u \times \vec{r}_v||} = \frac{1}{D} \vec{r}_u \times \vec{r}_v. \]

**B.2.2 Second Fundamental Form**

Recall from Section B.1 that curvature was defined using the Frenet-Serret formula

\[ \frac{\partial \vec{T}(s)}{\partial s} = \kappa \vec{M}(s). \]

Depending on the definition of \( \vec{p}(t) \), the normal \( \vec{M}(s) \) to the curve and the normal to the surface \( \vec{N}(u, v) \) can be different at the same surface location. To bring out the difference, the curve curvature equation and the surface normal direction are multiplied using the dot product

\[ \frac{\partial \vec{T}(s)}{\partial s} \cdot \vec{N} = \kappa \vec{M}(s) \cdot \vec{N}(u, v). \]

Let \( \phi \) be the angle between the curve’s normal and the surface’s normal for the same point on the surface

\[ \cos(\phi) = \vec{M}(s) \cdot \vec{N}(u, v). \]

Then

\[ \frac{\partial \vec{T}(s)}{\partial s} \cdot \vec{N}(u, v) = \kappa \cos(\phi). \]

From this expression the **Second Fundamental Form** can be obtained:

\[ \kappa \cos(\phi) \partial s^2 = L \partial u^2 + 2M \partial u \partial v + N \partial v^2 \]

220
where \( L(u, v) = \mathbf{N}(u, v) \cdot \tilde{r}_{uu}, M(u, v) = \mathbf{N}(u, v) \tilde{r}_{uv}, \) and \( N(u, v) = \mathbf{N}(u, v) \cdot \tilde{r}_{vv}. \)

The Second Fundamental Form relates curvature of a curve \( \tilde{r}(\bar{p}(t)) \) on the surface to the surface normal and the second order partial derivatives of the surface with respect to the parameters \( u, v. \) An interesting sub-case of this formulation is to consider the family of curves on the surface whose curve normal is equal to the surface normal for the same point on the surface. One way to insure curve and surface normal are always the same is to define a cutting plane (hence a curve through the surface) using the normal to the surface. The plane surface intersection results in curves whose normal \( \tilde{M}(s) \) are equivalent to the surface normal \( \tilde{N}(u, v). \) If \( \tilde{M}(s) = \tilde{N}(u, v) \) then \( \cos(\phi) = 1. \)

Let \( \lambda = \frac{\partial v}{\partial u} \) be the ratio of the change in the surface parameterization. For the set of normal curves, \( \lambda \) ratio specifies a unique curve going through a point on the surface. Therefore, the parameter \( \lambda \) is use to parameterize the possible normal curves going through any point on the surface.

By taking the ratio of the fundamental forms and forcing \( \tilde{M}(s) = \tilde{N}(u, v) \) the normal curvature at any point on the surface is given by:

\[
\kappa(\lambda) = \frac{L + 2M\lambda + N\lambda^2}{E + 2F\lambda + G\lambda^2},
\]

where we have dropped the \((u,v)\) parameterization for readability. This equation has two extreme values \( \kappa_1 \) and \( \kappa_2. \) They are found by setting \( \frac{\partial \kappa(\lambda)}{\partial \lambda} = 0 \) and finding the two roots to the equation. The roots are:

\[
\lambda_{1,2} = \frac{(LE - NE) \pm \sqrt{(NE - LE)^2 - 4(NF - MG)(ME - LF)}}{2(NF - MG)}
\]

and \( \kappa_1 = \kappa(\lambda_1), \kappa_2 = \kappa(\lambda_2). \) \( \kappa_1 \) and \( \kappa_2 \) are called principal curvatures. The quantities \( \lambda_1 \) and \( \lambda_2 \) define directions in the \((u,v)\)-plane; the corresponding directions in the
<table>
<thead>
<tr>
<th>$H &lt; 0$</th>
<th>$H = 0$</th>
<th>$H &gt; 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K &lt; 0$</td>
<td>Saddle Ridge</td>
<td>Minimal</td>
</tr>
<tr>
<td>$K = 0$</td>
<td>Convex Cylinder</td>
<td>Plane</td>
</tr>
<tr>
<td>$K &gt; 0$</td>
<td>Convex Ellipsoid</td>
<td>impossible</td>
</tr>
</tbody>
</table>

Table B.1: Classifications of surface shape using Mean $H$ and Gaussian $K$ curvature.

tangent plane are called the principal directions. Denote these principal directions as $\vec{\kappa}_1$ and $\vec{\kappa}_2$.

If the surface is reparameterized so that the $(u, v)$-plane axes are aligned with the $\lambda_1$ and $\lambda_2$ directions, it can be shown that the principal directions $\vec{\kappa}_1$ and $\vec{\kappa}_2$ must be perpendicular to one another. Furthermore Euler showed that using this special parameterization of the surface the principal curvatures are $\kappa_1 = \frac{L}{E}$ and $\kappa_2 = \frac{N}{G}$.

Then the curvature can be rewritten as:

$$\kappa(\lambda) = \kappa_1 \frac{E}{E + G\lambda^2} + \kappa_2 \frac{G\lambda^2}{E + G\lambda^2}.$$ 

This in turn can be rewritten in terms of a angle parameter $\phi$ (the angle between $\vec{\kappa}_1$ and the tangent vector of the normal curve being considered) instead of $\lambda$. Then the curvature parameterized by $\phi$, yielding

$$\kappa(\phi) = \kappa_1 \cos^2(\phi) + \kappa_2 \sin^2(\phi).$$

Gaussian curvature

$$K = \kappa_1 \kappa_2 = \frac{LN - M^2}{EG - F^2}$$

and Mean curvature

$$H = \frac{\kappa_1 + \kappa_2}{2} = \frac{NE - 2MF + LG}{2(EG - F^2)}$$

222
<table>
<thead>
<tr>
<th>Shape</th>
<th>SI Index Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convex Ellipsoid</td>
<td>$SI \in [-1, -5/8)$</td>
</tr>
<tr>
<td>Convex Cylinder</td>
<td>$SI \in [-5/8, -3/8)$</td>
</tr>
<tr>
<td>Saddle</td>
<td>$SI \in [-3/8, 3/8)$</td>
</tr>
<tr>
<td>Concave Cylinder</td>
<td>$SI \in [3/8, 5/8)$</td>
</tr>
<tr>
<td>Concave Ellipsoid</td>
<td>$SI \in [5/8, 1)$</td>
</tr>
</tbody>
</table>

Table B.2: Classification of surface shape using Shape Index $SI$.

are often used in the description of surfaces instead of the principal curvatures. This form has been used to classify surface shape in range image analysis into eight characteristic types shown in Table B.1 and lists Besl’s [11] classification tree.

Another alternative curvature representation defined by Koenderink [81] decouples the shape of the surface with the magnitude of the surface change. They define the $Shape\ Index\ SI = \frac{2}{\pi} \times \arctan\left(\frac{\kappa_1 + \kappa_2}{\kappa_1 - \kappa_2}\right)$ and Curvedness $C = \sqrt{\frac{\kappa_1^2 + \kappa_2^2}{2}}$ of a surface. The shape index can also be used to classify surface type, as listed in Table B.2.
BIBLIOGRAPHY


227


233


