USCIPI REPORT #1030

Shape Matching and Image Segmentation Using Stochastic Labeling

by

Bir Bhanu

August 1981

Signal and Image Processing Institute
UNIVERSITY OF SOUTHERN CALIFORNIA
Department of Electrical Engineering-Systems
Electrical Engineering Building
University Park/MC-2564
Los Angeles, CA 90089 U.S.A.

ACKNOWLEDGEMENTS

I am grateful to my advisor, Professor Olivier D. Faugeras, for his guidance, suggestions, encouragement and friendliness throughout the course of this work.

I want to express my sincere appreciation to the other committee members, Professors Alexander A. Sawchuk, Keith Price and E.K. Blum for their assistance and contributions. I wish to thank Professors Donald Marsh and Zultan Tokes of USC Medical School for their help in the acquisition of cancer cell images. Discussions with Dr. Thomas Henderson and the laboratory assistance of Jean Merlet, during my stay in France while visiting INRIA-Laboria, are also appreciated.

Thanks are due to the entire staff of the Image Processing Institute for their help in the preparation of this work. Finally, I thank Ms. Amy Yiu for the excellent typing of the manuscript.

TABLE OF CONTENTS

	Page
ACKNOWLEDGEMENTS	ii
TABLE OF CONTENTS	iii
LIST OF FIGURES	vi
LIST OF TABLES	xiii
ABSTRACT	xvii
CHAPTER	
1. INTRODUCTION	1
1.1 Introduction	1
1.2 Overview and Summary of Contributions	4
2. SHAPE MATCHING OF TWO-DIMENSIONAL OBJECTS	9
2.1 Introduction	9
2.2 Shape Matching Using Hierarchical Stochastic Labeling	14
2.3 Details of the Shape Matching Algorithm	25
2.4 Examples and Comments	39
2.5 Summary	86
3. SHAPE MATCHING OF TWO-DIMENSIONAL OCCLUDED OBJECTS	88

3.1 Introduction 88
3.2 Problem Formulation 93
3.3 Occlusion Algorithm 98
3.4 Examples and Comments 103
3.5 Summary 143

4. REPRESENTATION, MODELLING AND THREE-DIMENSIONAL SCENE ANALYSIS 145
4.1 Introduction 145
4.2 Three-Dimensional Data Acquisition 146
4.3 Representation and Modelling of 3-D Objects 152
4.4 Algorithm for Surface Approximation by Polygons 161
4.5 Experiments and Discussion 174
4.6 Summary 192

5. SHAPE MATCHING OF THREE-DIMENSIONAL OBJECTS 193
5.1 Introduction 193
5.2 Shape Matching Algorithm 198
5.3 Examples and Comments 208
5.4 Summary 231

6. SEGMENTATION OF IMAGES 233
6.1 Introduction 233
6.2 Segmentation Schemes 235
6.3 Segmentation Using Stochastic Labeling 239

6.4 Conclusion 266

7. CONCLUSIONS 269

7.1 Summary 269

7.2 Suggestions for Further Research 273

APPENDICES

A. Projection Operator 278

B. System Used to Obtain Sequence of Cell Images 281

REFERENCES 285

LIST OF FIGURES

Figure		Page
2.1	Block diagram of the shape matching algorithm using 2 stages of hierarchical stochastic labeling technique	15
2.2	Explanation of notations	24
2.3	Illustration of the definition of exangle	28
2.4	Matching distance error = AB + CD	32
2.5	(a) Model, Perimeter = 21, Number of vertices = 6	49
	(b) Object, Perimeter = 38, Number of vertices = 10	49
2.6	(a) Model 1, Perimeter = 34, Number of vertices = 9	56
	(b) Model 2, Perimeter = 35, Number of vertices = 9	56
	(c) Object, Perimeter = 67, Number of vertices = 18	57
2.7	Two aerial images of San Francisco taken at different times	62
2.8	Regions obtained using a recursive region splitting technique of segmentation. Regions in figs. 2.8(a) and 2.8(b) are obtained from fig. 2.7(a) with slightly different parameters in the segmentation scheme. Region in fig. 2.8(c) is obtained from fig. 2.7(b) using the same parameters used to obtain figs. 2.8(a) and 2.8(b). Regions shown are at different scales. (a) Size = 299 Rows by 258 Columns (b) Size = 297 Rows by 261 Columns (c) Size = 381 Rows by 253 Columns	63
2.9	Polygonal approximation of the regions shown in fig. 2.8 after reducing them by 14 times. A smoothing factor of 4 is used to obtain the	

polygonal approximations. Figs. 2.9(a), (b) and (c) correspond to figs. 2.8(a), (b) and (c) respectively.
(a) Object 1, Number of segments = 23
(b) Object 2, Number of segments = 23
(c) Model, Number of segments = 29 65

2.10 (a) An automobile piece
(b) Superposition of two such pieces 72

2.11 (a) Model, Number of segments = 28
(b) Object, Number of segments = 27 74

2.12 (a) An automobile piece
(b) Superposition of two such pieces 76

2.13 (a) Model, Number of segments = 19
(b) Object, Number of segments = 28 77

2.14 (a) An automobile piece
(b) Superposition of two such pieces 80

2.15 (a) Object, Number of segments = 29
(b) Model, Number of segments = 38 81

3.1 Block diagram of the occlusion algorithm for the shape description of 2 occluding models using 2 stages of the coordinated hierarchical stochastic labeing technique based on the gradient projection method and penalty function approach 92

3.2 (a) Model X_1, Perimeter = 34, Number of segments = 9 105
(b) Model X_2, Perimeter = 35, Number of segments = 9 105
(c) Apparent object, Perimeter = 67, Number of segments = 18 106

3.3 (a) Model X_1, Number of segments = 6 113
(b) Model X_2, Number of segments = 7 113
(c) model X_3, Number of segments = 9 114
(d) Apparent object, Number of segments = 14 114

3.4 (a) An industrial piece
(b) An industrial piece
(c) Partial occlusion of industrial pieces in (a) and (b) 121

3.5 (a) Model X_1, Number of segments = 26
(b) Model X_2, Number of segments = 14 123

(c) Apparent object, Number of segments = 24

3.6 (a) An industrial piece
(b) Partial occlusion of industrial pieces in figs. 3.4(a), 3.4(b) and 3.6(a) 127

3.7 (a) Model X_1, Number of segments = 26
(b) Model X_2, Number of segments = 14
(c) Model X_3, Number of segments = 10
(d) Apparent object, Number of segments = 28 129

3.8 Mitosis of cancer cells
(a) The cell undergoing mitosis
(b) The cell in fig. 3.8(a) is divided into 2 cells after 15 minutes 135

3.9 Images obtained at the third iteration when the segmentation technique described in Chapter 6 is applied to the images in Fig. 3.8. Parameters used are FACT = .9, α_1 = .5, α_2 = .1.
(a) Segmentation of the image in fig. 3.8(a)
(b) Segmentation of the image in fig. 3.8(b) 137

3.10 Polygonal approximation of the two cells in fig. 3.8(b), which resulted from the mitosis of the cell in fig. 3.8(a).
(a) Model X_1, Number of segments = 21
(b) Model X_2, Number of segments = 20 138

3.11 Polygonal approximation of the cell undergoing mitosis (fig. 3.8(a))
Apparent object, Number of segments = 28 139

4.1 The schematic diagram of 3-D scene analysis system 147

4.2 Laser ranging system 151

4.3 Two step process to approximate surface by polygons 162

4.4 Face 1 and 2 lie in the same plane, but are distinct; face 3 is non-convex. 164

4.5 Convexity and Narrowness conditions.
(a) A three point seed (the circled x's) which straddles an edge produces a plane which cross-sections the object. The convexity condition reduces the set of points (x's) to those shown in fig. 4.5(b). 164

(b) The narrowness condition excludes faces like this whose points all lie very near the same line.

4.6 Perpendicular distance from a point m_0 to the line A_1A_2 in 3-D 170

4.7 Automobile piece analyzed 175

4.8 The 14 range data views of the object; (a)-(l) correspond to orientations 0^o to 330^o; (m) is the top view and (n) is the bottom view. 176-178

4.9 The 14 range data views of the object shown as gray scale images. The lighter points are away from the observer and the darker ones are closer. 179

4.10 Faces found in the 0^o view (fig. 4.8(a)). There are 22 faces in this view. The rejected points and the points common to two or more faces are shown in blue and khaki color respectively. 183

4.11 Faces found in the 90^o view (fig. 4.8(d)). There are 14 faces in this view. The rejected points and the points common to two or more faces are shown in blue and khaki color respectively. 186

5.1 Block Diagram of the 3-D Shape Matching Algorithm 199

5.2 Faces found in the view shown in fig. 4.8(a). There are 22 faces in this view and they are labeled in the order they are found using the algorithm described in chapter 4. The rejected points and the points common to two or more faces are shown in brown and white color respectively. 213

5.3 Faces found in the view shown in fig. 4.8(b). There are 24 faces in this view and they are labeled in the order they are found using the algorithm described in chapter 4. The rejected points and the points common to two or more faces are shown in brown and white color respectively. 222

5.4 Faces found in the view shown in fig. 4.8(l). There are 24 faces in this view and they are

labeled in the order they are found using the algorithm described in chapter 4. The rejected points and the points common to two or more faces are shown in brown and white color respectively. 226

6.1 Two typical 128×128, 8 bit images and their gray level histograms.
(a) Cell image
(b) An aerial image
(c) Gray level histogram of the cell image. Mean = 163.76
(d) Gray level histogram of the aerial image. Mean = 26.47 236

6.2 Laplacian of the cell image and the corresponding gray level histogram.
(a) Laplacian of the cell image
(b) Histogram of the image in fig. 6.2(a) 238

6.3 Variation of the criterion (6.1) with the iteration number for various values of α_1 and α_2 for the cell image.
(a) α_1/α_2 = FACT = 1 in all cases 245
(b) $\alpha_1/\alpha_2 > 1$ and FACT = 1 in all cases 246
(c) $\alpha_1/\alpha_2 < 1$ and FACT = 1 in all cases 247

6.4 Effect of different values of α_1 and α_2 on the cell image such that $\alpha_1/\alpha_2 = 1$. In each figure the first row contains images for the first 4 iterations and the second row for the next 4 iterations.
(a) $\alpha_1 = \alpha_2 = 0.2$, FACT = 1.0
(b) $\alpha_1 = \alpha_2 = 0.8$, FACT = 1.0 249

6.5 Effect of biasing of class λ_1 on the cell image such that $\alpha_1/\alpha_2 = 2$.
(a) $\alpha_1 = 0.2$, $\alpha_2 = 0.1$, FACT = 1.0
(b) $\alpha_1 = 0.4$, $\alpha_2 = 0.2$, FACT = 1.0 250

6.6 Effect of biasing class λ_2 on the cell image such that $\alpha_1/\alpha_2 = 1/2$.
(a) $\alpha_1 = 0.1$, $\alpha_2 = 0.2$, FACT = 1.0
(b) $\alpha_1 = 0.2$, $\alpha_2 = 0.4$, FACT = 1.0 251

6.7 Results showing that for a fixed ratio of α_1 and α_2 increasing both of them by a constant factor increases the speed of convergence. In each figure FACT = 1 is taken.
(a) $\alpha_1 = \alpha_2 = 0.2$, iteration 8
(b) $\alpha_1 = \alpha_2 = 0.5$, iteration 4 252

(c) $\alpha_1 = \alpha_2 = 0.9$, iteration 3

6.8 Results of Gradient Relaxation method at various iterations and corresponding histograms for the cell image. FACT = 0.9, $\alpha_1 = 0.2$, $\alpha_2 = 0.1$. 253-255
(a) Iteration 1
(b) Histogram of fig. 6.8(a)
(c) Iteration 3
(d) Histogram of fig. 6.8(c)
(e) Iteration 4
(f) Histogram of fig. 6.8(e)
(g) Iteration 5
(h) Histogram of fig. 6.8(g)
(i) Iteration 7
(j) Histogram of fig. 6.8(i)
(k) Iteration 9
(l) Histogram of fig. 6.8(k)

6.9 Results of Gradient Relaxation method at various iterations and corresponding histograms for the aerial image. FACT = 1, $\alpha_1 = 0.1$, $\alpha_2 = 0.5$. 256-257
(a) Iteration 1
(b) Histogram of fig. 6.9(a)
(c) Iteration 3
(d) Histogram of fig. 6.9(c)
(e) Iteration 5
(f) Histogram of fig. 6.9(e)

6.10 Variation of the criterion (6.1) with the iteration number for various values of FACT for the cell image. $\alpha_1 = \alpha_2 = 0.50$ in all cases. 259

6.11 Gradient relaxation results at various iterations for different values of FACT for the cell image. $\alpha_1 = \alpha_2 = 0.5$ in all the 3 cases. 260
(a) FACT = 0.1
(b) FACT = 0.5
(c) FACT = 1.0

6.12 Results of Nonlinear relaxation method at various iterations and corresponding histograms for the cell image. FACT = 0.9. 261-263
(a) Iteration 1
(b) Histogram of fig. 6.12(a)
(c) Iteration 3
(d) Histogram of fig. 6.12(c)
(e) Iteration 4

(f) Histogram of fig. 6.12(e)
(g) Iteration 5
(h) Histogram of fig. 6.12(g)
(i) Iteration 7
(j) Histogram of fig. 6.12(i)
(k) Iteration 9
(l) Histogram of fig. 6.12(k)

6.13 Results of Nonlinear relaxation method at various iterations and corresponding histograms for the aerial image. FACT = 1. 264-265
(a) Iteration 1
(b) Histogram of fig. 6.13(a)
(c) Iteration 3
(d) Histogram of fig. 6.13(c)
(e) Iteration 5
(f) Histogram of fig. 6.13(e)

7.1 A system for the analysis of image sequences 276

B.1 Real time video acquisition and digital display system 282

LIST OF TABLES

Table		Page
2.1	Expected assignments for the units of the model in Example 2.1.	50
2.2	Label of units of the model. Reduced compatibility and gradient computation (Example 2.1).	51
2.3	Labels of units of the model. Full compatibility and gradient computation (Example 2.1). Parameters same as in Table 2.2 except no. of neighbors = 11.	52
2.4	Label of units of the model. Parameters same as in Table 2.2 except ITER1 = 6 and ITER2 = 0.	53
2.5	Label of units of the model. Parameters same as in Table 2.3 except ITER1 = 6, ITER2 = 0.	54
2.6	Expected assignments of the units of Model 1 and Model 2 in Example 2.2.	59
2.7	Label of units of the model 1. Example 2.2.	60
2.8	Label of units of the model 2. Example 2.2. Parameters same as in Table 2.7.	61
2.9	Label of segments of the object 1. Example 2.3.	67
2.10	Label of segments of the object 2. Example 2.3. Parameters same as in Table 2.9 except ITER1 = 6, ITER2 = 6.	68
2.11	Computation of relative rotation between the object 1 and the model. Example 2.3.	70
2.12	Computation of the relative rotation between the object 2 and the model. Example 2.3.	71
2.13	Label of units of the model. Example 2.4.	75

2.14 Label of units of the model. Example 2.5. 79

2.15 Label of units of the model. Example 2.6. 82

3.1 Expected assignments of the units of models X_1 and X_2, Example 3.1. 107

3.2 Assignment of the units of model X_1, Example 3.1. 108

3.3 Assignment of the units of model X_2. Example 3.1. Parameters same as in Table 3.2. 109

3.4 Assignment of the units of model X_1. Example 3.1. Parameters same as in Table 3.2 except ITER2 = 11 and PERCEN = 90%. 110

3.5 Assignment of the units of model X_2, Example 3.1. Parameters same as in Table 3.4. 111

3.6 Expected assignments of the units of models X_1, X_2 and X_3. Example 3.2. 116

3.7 Results of labeling of the models X_1, X_2 and X_3 when they are matched to the apparent object by themselves (i.e., no coordination). Example 3.2. 117

3.8 Results of labeling for the model X_1 when models X_1, X_2 and X_3 are matched to the apparent object using the Occlusion Algorithm. Examle 3.2. Parameters same as in Table 3.7. PERCEN = 30%. 118

3.9 Results of labeling for the model X_2 when models X_1, X_2 and X_3 are matched to the apparent object by using the Occlusion Algorithm. Example 3.2. Parameters same as in Table 3.7. PERCEN = 30%. 119

3.10 Results of labeling for the model X_3 when models X_1, X_2 and X_3 are matched to the apparent object using the Occlusion Algorithm. Example 3.2. Parameters same as in Table 3.7. PERCEN = 30%. 120

3.11 Results of labeling of the models X_1 and X_2 when they are matched to the apparent object by themselves (i.e., no coordination). Example 3.3. 124

3.12 Results of labeling for the model X_1 when models X_1 and X_2 are matched to the apparent object using the Occlusion Algorithm. Example 3.3. Parameters same as in Table 3.11 except ITER2 = 5. PERCEN = 10%. 125

3.13 Results of labeling for the model X_2 when models X_1 and X_2 are matched to the apparent object using the Occlusion Algorithm. Example 3.3. Parameters same as in Table 3.11 except ITER2 = 5, PERCEN = 10%. 126

3.14 Results of labeling of the models X_1, X_2 and X_3 when they are matched to the apparent object by themselves (i.e., no coordination). Example 3.4. 130

3.15 Results of labeling for the model X_1 when models X_1, X_2 and X_3 are matched to the apparent object using the Occlusion Algorithm. Example 3.4. Parameters same as in Table 3.14 except ITER2 = 8, PERCEN = 40%. 131

3.16 Results of labeling for the model X_2 when models X_1, X_2 and X_3 are matched to the apparent object using the Occlusion Algorithm. Example 3.4. Parameters same as in Table 3.14 except ITER2 = 8, PERCEN = 40%. 132

3.17 Results of labeling for the model X_3 when models X_1, X_2 and X_3 are matched to the apparent object using the Occlusion Algorithm. Example 3.4. Parameters same as in Table 3.14 except ITER2 = 8, PERCEN = 40%. 133

3.18 Results of labeling of the models X_1 and X_2 when they are matched to the apparent object by themselves (i.e., no coordination). Example 3.5. 140

3.19 Results of labeling for the model X_1 when models X_1 and X_2 are matched to the apparent object using the Occlusion Algorithm. Example 3.5. Parametes same as in Table 3.18 except ITER2 = 7, PERCEN = 80%. 141

3.20 Results of labeling for the model X_2 when models X_1 and X_2 are matched to the apparent object using the Occlusion Algorithm. Example 3.5. Parameters same as in Table 3.18 except 142

ITER2 = 7, PERCEN = 80%.

4.1 List of faces in the 0° view (fig. 4.8(a)). I1, I2 and I3 are the indices in the list (of points for this view) which make up a face. 185

4.2 List of faces in the 90° view (fig. 4.8(d)). I1, I2 and I3 are the indices in the list (of points for this view) which make up a face. 187

4.3 Neighbors of a face in 0° and 90° views. These neighbors are arranged in the descending order of their size.
(a) 0° view (fig. 4.8(a))
(b) 90° view (fig. 4.8(d)) 188

5.1 Neighbors of the faces shown in fig. 5.2. Neighbors are arranged in the descending order of their size. 214

5.2 Labels of the faces shown in fig. 5.2. Example 5.1. 215

5.3 Labels of the faces shown in fig. 5.2. Example 5.1. Parameters same as in Table 5.2 except ITER2 = 8 and No. of neighbors = (1,2). 219

5.4 Labels of the faces shown in fig. 5.2. Example 5.1. Parameters same as in Table 5.2 except p_i(nil) = 0.1, No. of neighbors = (1,2) and Inverse weights = (10, 8, 5, 5, 10, 25, 25, 25). 220

5.5 Neighbors of the faces shown in fig. 5.3. Neighbors are arranged in the descending order of their size. 223

5.6 Labels of the faces shown in fig. 5.3. Example 5.2. 224

5.7 Neighbors of the faces shown in fig. 5.4. Neighbors are arranged in the descending order of their size. 227

5.8 Labels of the faces shown in fig. 5.4. Example 5.3. 228

ABSTRACT

New results are presented in the areas of shape matching of nonoccluded and occluded objects in two dimensions, surface approximation by polygons, shape matching of objects in three dimensions, and segmentation of images having unimodal distributions. The same stochastic labeling technique is used in both shape matching and segmentation with various extensions.

Shape matching is viewed as a segment matching problem. Unlike the previous work in shape matching of 2-D objects, the technique is based on a stochastic labeling procedure which explicitly maximizes a criterion function based on the ambiguity and inconsistency of classification. To reduce the computation time, the technique is hierarchical and uses results obtained at low levels to speed up and improve the accuracy of results at higher levels. This basic technique has been extended to the situation where various objects partially occlude each other to form an apparent object and our interest is to find all the objects participating in the occlusion. In such a case several hierarchical processes are executed in

parallel for every participating object in the occlusion and are coordinated in such a way that the same segment of the apparent object is not matched to the segments of different actual objects. These techniques have been applied to two-dimensional shapes represented by polygons and the power of the techniques is demonstrated by the examples taken from synthetic, aerial, industrial and microscope images, where the matching is done after using the actual segmentation methods.

In three dimensional scene analysis, a method based on a laser triangulation principle to acquire 3-D data is described. The problems related with the 3-D data acquisition and geometric processing are addressed. A method is given for representing three dimensional objects, such as complicated automobile castings, by a set of convex planar faces. The major advantage of this method is that it is applicable to a composite object and not restricted to single range view which was the limitation of the previous work in 3-D scene analysis. This method is used to obtain the surface representation of a 3-D object in terms of faces approximated by polygons. The hierarchical stochastic labeling technique used in 2-D shape matching is applied to do the shape matching of 3-D objects based on matching the faces of the unknown view against the faces of the model. Thus the

segment matching problem of 2-D becomes face matching problem in 3-D. The objective here is to match the individual views of the object taken from any vantage point. The results of partial shape recognition can be used to determine the orientation of the object in 3-D space.

Further the stochastic labeling technique used for shape matching has been applied for the segmentation of images having unimodal gray level distributions. Unimodal histograms are typically obtained when the image consists mostly of a large background region with other small but significant regions. This is true for most biological and aerial images. The technique provides the control over the relaxation process by choosing three parameters which can be tuned to obtain the desired segmentation results at a faster rate compared to the classical nonlinear relaxation method. Aerial and biological examples are presented.

The techniques developed here have the potential of being useful in industrial automation, navigation and military applications.

CHAPTER 1

INTRODUCTION

1.1 Introduction

The problem of assigning names or labels to a set of units/objects is the key problem in computer vision, image analysis and pattern recognition. Since all the labels are not possible for a given unit, constraints based on contextual information, called the world model, are used to obtain a consistent and unambiguous valid assignment of the units. Local parallel processes are a very efficient way of assigning labels. The features of such algorithms include the propagation of local contextual information in a paradigm of competition and cooperation, locality and speed. In general the task of assigning names to units only on the basis of features of the units is very difficult since any segmentation based on low-level analysis is bound to contain errors and the computed features are noisy. The solution to this problem is to delay any firm commitment until all the contextual information has been used. Depending upon the type of constraints embodying the world model, the problem can be

attacked by discrete methods (discrete relaxation) or continuous methods (continuous relaxation, also called probabilistic or stochastic labeling). Recent surveys on these algorithms applied to low level vision and symbolic matching can be found in [1-1, 1-2].

Waltz [1-3] used the notion of constraints introduced by the world model for the description of scenes made up of complex polyhedra. Rosenfeld, Hummel and Zucker [1-4] proposed a parallel version of this algorithm. Barrow and Tenenbaum [1-5] and Rosenfeld et al. [1-4] introduced the idea of stochastic labeling. The nonlinear algorithm proposed in [1-4] has been extensively used in various applications [1-2]. Marr et al. [1-6] have used the similar ideas. Theoretical analysis of convergence and stability properties of this algorithm have proven to be difficult as shown by Zucker et al. [1-7, 1-8]. Faugeras and Berthod [1-9 to 1-11] reformulated the stochastic labeling problem by explicitly maximizing a criterion function based on the inconsistency and ambiguity of classification. This criterion is maximized using a gradient projection method. A similar idea has been proposed by Ullman [1-12].

In this dissertation we extend the stochastic labeling technique of Faugeras and Berthod [1-10] to do

shape matching of two and three dimensional objects in a hierarchical manner and to perform the segmentation of images. We shall present new results in the areas of shape matching of nonoccluded and occluded objects in two-dimensions, three-dimensional data acquisition, surface approximation by polygons, shape matching of three-dimensional objects and the segmentation of images having unimodal gray level distributions. The power of the techniques in 2-D is demonstrated by the examples taken from synthetic, aerial, industrial parts and microscope images where the matching is done after using the actual segmentation methods. In 3-D the complexity of the objects viewed is typified by a complicated casting of an automobile. In shape matching our concern will be with the shape properties of an object only. Other cues such as color and surface texture etc. have not been used. For the segmentation of images aerial and biological gray level images have been considered. The related previous work in the field of image analysis will be described in the corresponding chapters. In the next section, we present an overview of the dissertation and summary of contributions.

1.2 Overview and Summary of Contributions

Chapter 1 consists of the introduction to the dissertation, broadly outlines the research problem, objectives and contributions of the research work.

Chapter 2 presents an extension of the stochastic labeling technique developed by Faugeras and Berthod [1-10] to do shape matching of two-dimensional objects in a hierarchical manner. The technique explicitly maximizes a criterion function based on the ambiguity and inconsistency of classification. Shape matching is viewed as a segment matching problem, where a piece of a shape is recognized as an approximate match to a part of a larger shape. The hierarchical nature of the algorithm reduces the computation time and uses results at low levels to speed up and improve the accuracy of results obtained at higher levels. The two-dimensional shapes are represented by their polygonal approximation by finding the points of maximum curvature on their boundary. As compared to the previous studies in 2-D shape matching, the hierarchical stochastic labeling technique developed here, has been applied not only to the synthetic images, but also to the real images taken from the aerial, industrial parts and biological fields. The technique allows for the changes in scale, rotation, translation and significant changes in

shape. The results of shape matching are used to compute the rotation. Various strategies that lead to faster computation are described. Although the technique is computationally more costly than the other feature based or correlation techniques, it is more robust and flexible. It allows for the parallel implementation in hardware. This method is also used when the the objects partially occlude, but our objective is to match only the object of interest.

Chapter 3 extends the hierarchical stochastic labeling technique of chapter 2 to do shape matching of two-dimensional occluded objects. For each of the objects participating in the occlusion, there is a hierarchical process. These processes are executed in parallel and are coordinated in such a way that the same segment of the apparent object, formed as a result of occlusion of two or more actual objects, is not matched to the segments of different actual objects. This problem is solved by combining the gradient projection method and penalty function approach. Results are presented on synthetic objects, a sequence of microscope images and industrial images when two or three objects partially occlude.

Chapter 4 describes the three-dimensional scene analysis system implemented in this dissertation for the

shape matching of real world 3-D objects. Techniques used for acquiring 3-D data and geometric processing are presented. Issues related to representation and modelling of 3-D objects are discussed. A new technique for the approximation of 3-D objects by a set of planar faces is discussed in detail. This is used to generate a 3-D model of an object in terms of faces approximated by polygons. The technique is a sequential region growing algorithm. It is not applied to range images, but rather to a set of 3-D points. It is not directly related to how the 3-D surface points were acquired, but tied to the sampling distances between the points on the object. It is not restricted to single view range data images, but applicable to a composite object and does not require the ordering of points. It finds the convex faces of the object, but the information exists to merge convex parts of nonconvex faces . The 3-D model of an object is obtained by combining the object points from a sequence of range data images corresponding to various views of the object, applying the necessary transformations and then approximating the surface by polygons. Such a representation should be of use not only in 3-D scene analysis, but also in computer graphics and reduction of terrain data obtained by synthetic aperture radar etc.

Chapter 5 uses the basic hierarchical stochastic labeling technique developed in Chapter 2 for the shape matching of real world 3-D objects. In 2-D the matching is "segment matching", but in 3-D it is "face matching". We match the faces of the unknown view against the faces of the 3-D model. Details of the algorithm are presented and the results are shown on several unknown views of a complicated automobile casting. The results can be used to obtain the orientation of the object in three-space.

Chapter 6 presents results about the segmentation of images having unimodal gray level distributions. It uses the stochastic labeling technique used for shape matching. Unimodal histograms are typically obtained when the image consists of a large background region with other small but significant regions. For example, in the biomedical area the extraction of the boundaries of various types of cells is complicated by the fact that cells are very close together, their boundaries are poorly defined and the gray level histogram is unimodal. Similarly unimodal histograms are obtained for aerial pictures because the range of intensities of the many different objects overlap. The technique developed here provides the control over the relaxation process by allowing the user to choose three parameters which can be tuned to obtain the desired segmentation results at a faster rate. This

is the major advantage of the technique compared to the classical nonlinear relaxation method used in segmentation [1-13]. Aerial and biological cell examples are presented.

Chapter 7 summarizes the results of the research and offers suggestions for future research.

Appendices describe the projection operator needed in the gradient projection method of optimization and the system used to obtain the sequences of cancer cells and lymphocytes images used in this work.

CHAPTER 2

SHAPE MATCHING OF TWO DIMENSIONAL OBJECTS

2.1 Introduction

The problem of shape matching mainly consists of two parts : One shape is recognized to be an approximate match to another shape or a piece of a shape is recognized as an approximate match to a part of larger shape [2-1]. The second problem is also known as the "segment matching" problem [2-2]. In this chapter we address the "segment matching" problem of shape matching to analyze a shape using a hierarchical stochastic labeling technique. The class of shapes that we consider are represented by simple closed curves and are two dimensional in nature such as the boundary of a region in an aerial image, outlines of biological cells and industrial parts etc. These shapes are approximated by polygons. The shapes are allowed to have undergone translation, rotation, scale and significant changes in the shape. These variations are essential if one is concerned with the real images, because in practice the results of a segmentation technique will be different when it is applied to the

images of the same scene taken under different conditions.

Past techniques used in the shape matching of two-dimensional objects are: chain code cross-correlation, Fourier descriptors and moments, statistical pattern recognition techniques, symbolic matching, syntactic and relaxation methods.

The usefulness of chain code cross-correlation [2-3] as a solution of the segment matching problem is very much limited by the fact that it is not rotation invariant, very sensitive to local changes in the number of chain links and quite sensitive to small global changes in the shape. Fourier descriptors [2-4 to 2-7] and moments [2-8,2-9] use global features. They have proven effective in attacking both two and three dimensional shape matching problems. Also there exist a large number of statistical pattern recognition techniques for shape matching [2-10]. However, these global features based approaches cannot be used for the segment matching problem because they suffer from the fact that the descriptors of a segment of a shape do not ordinarily bear any simple relationship to the descriptors of the entire shape. Price and Reddy [2-11] use a symbolic matching technique to locate corresponding segments in two views of a natural scene using the machine generated segmentations rather than perfect operator

generated segmentations. However use of symbolic matching as the solution of the segment matching problem is limited since it depends upon the global features. The symbolic matching technique has been improved by Faugeras and Price [2-12] by using a stochastic labeling technique and good results have been obtained in the semantic description of aerial images.

For the syntactic approaches to work, shapes to be analyzed must be segmented appropriately into pieces which correspond to the terminal symbols of some grammar, and these pieces must subsequently be analyzed by a parsing mechanism. Generally, it is assumed that pieces can be easily found, but this is not true when dealing with real images. Davis and Rosenfeld [2-13, 2-14] use iterative methods for recognizing upright squares on a noisy background and hierarchical relaxation for waveform parsing. Henderson [2-15] extends the hierarchical relaxation process in [2-14]. He describes syntactic shape analysis technique which combines the constraint propagation and syntactic representation techniques. The design and debugging of shape grammars specific to the object of interest is a major difficulty with this method and an interactive approach may be required to decompose a shape into natural pieces. Syntactic techniques are awkward in handling pattern noise and distortion [2-16].

Davis [2-1, 2-17] considers segment matching using discrete relaxation methods which carry strong syntactic flavor. However, neither he defines a hierarchical relaxation network [2-1] nor studies its usefulness and computational properties. Using a syntactic shape analyzer for contour matching Pavlidis [2-18] shows that the islands examples of Davis [2-1] are completely trivial. Kitchen [2-19, 2-20] applies discrete and fuzzy logic approaches of relaxation for matching relational structures. Shapiro's [2-21] approach is similar to Pavlidis [2-22] in which the shape is decomposed into primitives and a model of shape consisting of these primitives, their properties and relationships is obtained. Since the syntactic methods can only perform discrete symbolic analysis and are not very useful when we deal with real pictures, Tsai and Fu [2-16] present a syntactic-statistical approach for the recognition of industrial parts. Such a combination is made possible through the use of attributed grammars. The effectiveness of this approach to more complicated shapes is yet to be explored. Other techniques used in shape matching are: analyzing shape at the level of boundary chains (ordered sets of connected edge points) [2-23], transform boundary chains into concurves (ordered sets of curve segments) and matching shapes at the level of such concurve, [2-24],

converting edge-point data into straight lines and arcs of ellipses which are used in matching [2-25] and transforming a shape boundary into a set of polar coordinates [2-26]. Pavlidis [2-27] presents a review of algorithms for shape analysis.

We solve the segment matching problem by extending the stochastic labeling technique of Faugeras and Berthod [2-28 to 2-32]. As compared to the probabilistic method of Rosenfeld, Hummel and Zucker [2-33] and Kitchen [2-20], this technique explicitly maximizes a criterion function based on the ambiguity and inconsistency of labeling of a set of units/objects. The criterion is maximized using a gradient projection method. Rosenfeld's method fails to completely account for the coupling between units and the notion of ambiguity. It has a tendency to converge towards ambiguous solutions [2-12,2-36]. Kitchen's method has also been shown as unsuitable when we deal with many features, many segments, noisy data and similar objects [2-36]. The basic technique of Faugeras and Berthod [2-28] with some variations has been successfully applied to semantic description of aerial images, segmentation and edge detection [2-12, 2-34 to 2-36]. We extend this technique to do shape matching of objects in a hierarchical manner. The hierarchical nature of the shape matching algorithm allows the reduction of computation

time and improvement in the accuracy of results obtained at higher levels. We call the shape matching algorithm as hierarchical because at the higher levels of hierarchy, there are more constraints and world knowledge. This is different from the hierarchical relaxation of Davis [2-17], which is based on deriving local constrains at many levels of the description from a stratified context free system of relational models.

In this chapter we formulate the shape matching problem using the hierarchical stochastic labeling (also called the hierarchical gradient relaxation) in section 2-2 and discuss the details of the algorithm in section 2-3. Section 2-4 presents examples from the domains of synthetic, aerial and industrial parts images. Finally, section 2-5 summarizes the chapter.

2.2 Shape Matching Using Hierarchical Stochastic Labeling

In this section we present a two stage hierarchical stochastic labeling method (fig. 2-1) for matching the segments of a template/model against the segments of an observed object. Let $T = (T_1, T_2, \ldots, T_N)$ and $O = (O_1, O_2, \ldots, O_{L-1})$ be the polygonal path representation of the model and the object respectively, where T_i and O_j are line segments, $i = 1, \ldots, N$ and $j = 1, \ldots, L-1$. In general

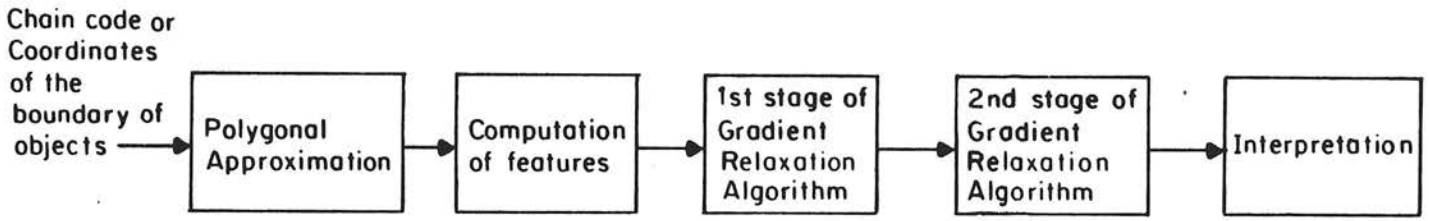


Fig. 2.1 Block diagram of the shape matching algorithm using 2 stages of hierarchical stochastic labeling technique

L may be greater, equal to or less than N. Model elements will be referred to as units and object elements as classes. We are trying to identify part of the model T within the observation O. We are therefore, trying to label each of the segments T_i (i = 1,...,N) either as a segment O_j (j = 1,...,L-1) or as not belonging to O (label O_L = Nil). Each segment T_i therefore has L possible labels.

Using a technique described in section 2.3 we compute for every segment T_i a set of L positive numbers $p_i(\ell)$, ℓ = 1,...,L forming a vector $\vec{p}_i = [p_i(1),...,p_i(L)]^T$. $p_i(\ell)$ can be thought of as the probability of labeling the segment T_i as O_ℓ. It satisfies the constraints

$$\sum_{\ell=1}^{L} p_i(\ell) = 1 \qquad (2\text{-}1)$$

and

$$p_i(\ell) \geq 0 \qquad (2\text{-}2)$$

The set of all vectors $\vec{p}_i$ (i = 1,...,N) is called a stochastic labeling of the set of units.

Initially the stochastic labeling is ambiguous (except in some very special cases) and we make it evolve toward a less ambiguous labeling by comparing the local

structure of T and O. From now on the indexes i are taken modulo N. To every segment T_i, we associate the two neighboring segments T_{i-1} and T_{i+1}. In order to compare the local structures of T and O we define two compatibility functions C_1 and C_2 of

$$S_2 \times O^2 \text{ and } S_3 \times O^3 \text{ into } [0,1] \text{ where,}$$

S_2 and S_3 are two subsets of T^2 and T^3 defined by,

$$S_2 = \{(T_i, T_j)\},\ i = 1,\ldots,N,\ j = i-1 \text{ or } i+1$$

$$S_3 = \{(T_i, T_{i-1}, T_{i+1})\},\ i = 1,\ldots,N$$

The compatibility function $C_1(T_i, O_k, T_j, O_\ell)$ ($j = i-1$ or $i+1$) and $C_2(T_i, O_k, T_{i-1}, O_\ell, T_{i+1}, O_m)$ will be denoted more simply by $C_1(i,k,j,\ell)$ and $C_2(i,k,i-1,\ell,i+1,m)$. C_1 and C_2 take values between 0 and 1. $C_1(i,k,i-1,\ell)$ measures the resemblance of the set $\{T_i, T_{i-1}\}$ with the set $\{O_k, O_\ell\}$. A good (bad) match means that the value of C_1 is close to 1, (0). As described in [2-29] we can associate to every segment T_i a compatibility vector $\vec{q}_i = [q_i(1),\ldots,q_i(L)]^T$. Intuitively this vector represents what the neighbors of segment T_i (that is to say segments T_{i-1} and T_{i+1}) "think" about the way it should be labeled whereas $\vec{p}_i$ represents what the segment T_i "thinks" about its own labeling.

Mathematically speaking we compute

$$Q_{ij}(k) = \sum_{\ell=1}^{L} C_1(i,k,j,\ell)p_j(\ell), \qquad \begin{matrix} j = i-1,\ i+1 \\ i = 1,\ldots,N \\ k = 1,\ldots,L \end{matrix} \tag{2-3}$$

$$Q_i^{(1)}(k) = \frac{1}{2}(Q_{i\ i-1}(k) + Q_{i\ i+1}(k)) \tag{2-4}$$

$$Q_i^{(2)}(k) = \sum_{\ell_1,\ell_2=1}^{L} C_2(i,k,i-1,\ell_1,i+1,\ell_2)p_{i-1}(\ell_1)\, p_{i+1}(\ell_2) \tag{2-5}$$

The numbers $Q_i^{(1)}(k)$ and $Q_i^{(2)}(k)$, $k = 1,\ldots,L$ are positive. The idea is that they are large when the probabilities of the labels of the neighbors of T_i compatible with label O_k are large and small otherwise. The numbers $Q_i^{(1)}(k)$ and $Q_i^{(2)}(k)$ are normalized so that they add up to 1 yielding two vectors $\vec{q}_i^{(1)}$ and $\vec{q}_i^{(2)}$ such that,

$$q_i^{(j)}(k) = \frac{Q_i^{(j)}(k)}{\sum_{\ell=1}^{L} Q_i^{(j)}(\ell)}, \qquad \begin{matrix} j = 1,2 \\ k = 1,\ldots,L \end{matrix} \tag{2-6}$$

It is desired to decrease the discrepancy between what every segment T_i thinks about its own labeling ($\vec{p}_i$) and what its neighbors think about it ($\vec{q}_i^{(j)}, j = 1,2$). We can therefore define local inconsistency as the amount of difference between $\vec{p}_i$ and $\vec{q}_i^{(j)}$ ($j = 1,2$). A good measure

of it is the angle between these two vectors,

$$C_i^{(j)} = -\cos\theta_i = \frac{\vec{p}_i \cdot \vec{q}_i^{(j)}}{\|\vec{p}_i\|_2 \|\vec{q}_i^{(j)}\|_2} \quad , \qquad j = 1,2 \tag{2-7}$$

when $\vec{p}_i = \vec{q}_i^{(j)}$, $-\cos\theta_i = -1$ and is larger than -1 if $\vec{p}_i$ and $\vec{q}_i^{(j)}$ are different with a maximal value of 0 (because $\vec{p}_i$ and $\vec{q}_i^{(j)}$ are probability vectors). Similarly, a local measure of ambiguity can be defined as the quadratic entropy,

$$H_i = \sum_{\ell=1}^{L} p_i(\ell)(1-p_i(\ell)) = 1-\|\vec{p}_i\|_2^2 \tag{2-8}$$

Since H_i is large when $\|\vec{p}_i\|$ is small and vice versa, we can use

$$H_i' = \|\vec{p}_i\| \tag{2-9}$$

as a local measure of ambiguity. Since we want to minimize both local inconsistency and ambiguity, we can use the product of eqs. (2-7) and (2-9) as a reasonable candidate for a measure of both quantities i.e.,

$$C_i^{(j)} H_i' = -\frac{\vec{p}_i \cdot \vec{q}_i^{(j)}}{\|\vec{q}_i^{(j)}\|_2} \quad , \qquad j = 1,2 \tag{2-10}$$

It is minimal when $\vec{p}_i = \vec{q}_i^{(j)}$ (minimum inconsistency) and $\vec{p}_i$ is the unit vector (minimum ambiguity). We can define,

$$J_i'^{(j)} = \|\vec{q}_i^{(j)}\| C_i^{(j)} H_i' = -\vec{p}_i \cdot \vec{q}_i^{(j)}, \quad j = 1,2 \qquad (2\text{-}11)$$

as a local measure of ambiguity and inconsistency. Note that $J_i'^{(j)}$ is minimum for $\vec{p}_i = \vec{q}_i^{(j)}$ and $\vec{p}_i$ = unit vector. Therefore, a good "local" measure of ambiguity and inconsistency is the inner product $\vec{p}_i \cdot \vec{q}_i^{(j)}$, j = 1,2. By computing the average over the set T of these local measures we obtain two global criteria:

$$J^{(j)} = \sum_{i=1}^{N} \vec{p}_i \cdot \vec{q}_i^{(j)}, \quad j = 1,2 \qquad (2\text{-}12)$$

The problem of labeling the segments T_i is therefore equivalent to an optimization problem: given an initial labeling $\vec{p}_i^{(0)}$, i = 1,...,N, find a local maximum of the criteria $J^{(j)}$ (j = 1,2) closest to the original labeling $\vec{p}_i^{(0)}$ subject to the constraints that $\vec{p}_i$'s are probability vectors. Since C_2 is a better measure than C_1 of the local match between T and O we are actually interested in finding local maxima of the criterion $J^{(2)}$. On the other hand maximizing $J^{(1)}$ is easier from the computational standpoint. We therefore use the following hierarchical approach: starting with an initial labeling $\vec{p}_i^{(0)}$, we look for a local maximum $\vec{p}_i^{(1)}$ of the criterion $J^{(1)}$. This labeling is less ambiguous than $\vec{p}_i^{(0)}$ in the sense that many labels have been dropped (their probabilities $p_i(k)$

are equal to zero). We then use the labeling $\vec{p}_i^{(1)}$ as an initial labeling to find a local maximum of the criterion $J^{(2)}$. The computational saving comes from the fact that the values $C_2(i,k,i-1,\ell_1,i+1,\ell_2)$ corresponding to probabilities $p_{i-1}(\ell_1)$ or $p_{i+1}(\ell_2)$ equal to zero are not computed.

The problem of maximizing (2-12) can be efficiently solved using the gradient projection method [2-29, 2-37]. The gradient of the criterion at the first stage of hierarchy is given by,

$$\frac{\partial J^{(1)}}{\partial p_i(k)} = q_i^{(1)}(k) + \sum_j \frac{1}{D_j} \sum_{\ell=1}^{L} C_1(j,\ell,i,k)\,(p_j(\ell) - \vec{p}_j \cdot \vec{q}_j^{(1)}) \qquad k = 1,\ldots,L \quad j = i-1,i+1 \qquad (2\text{-}13)$$

where

$$D_j = \sum_{\ell=1}^{L} Q_j^{(1)}(\ell) \qquad (2\text{-}14)$$

The first term in (2-13) corresponds to the simple maximization of the product $\vec{p}_i \cdot \vec{q}_i^{(1)}$ in the global criterion $J^{(1)}$, and the second term corresponds to the coupling between units through the compatibility function C_1. Note that in general $C_1(j,\ell,i,k) \neq C_1(i,k,j,\ell)$ since it depends upon the manner in which the compatibility is computed. More about this is explained in the discussion

of the compatibility computation.

At the second stage of hierarchy the gradient of the criterion $J^{(2)}$ is obtained as,

$$\frac{\partial J^{(2)}}{\partial p_i(k)} = q_i^{(2)}(k) + \frac{1}{D_{i_1}}[\tilde{Q}_{i_1}(k) - D'_{i_1}\vec{p}_{i_1}\cdot\vec{q}_{i_1}^{(2)}] \quad (2\text{-}15)$$
$$+ \frac{1}{D_{i_2}}[\tilde{Q}_{i_2}(k) - D'_{i_2}\vec{p}_{i_2}\cdot\vec{q}_{i_2}^{(2)}]$$

where,

$$\tilde{Q}_{i_1}(k) = \sum_{\ell_1,\ell_3=1}^{L} C_2(i_1,\ell_1,i_3,\ell_3,i,k)p_{i_1}(\ell_1)p_{i_3}(\ell_3) \quad (2\text{-}16)$$

$$\tilde{Q}_{i_2}(k) = \sum_{\ell_2,\ell_4=1}^{L} C_2(i_2,\ell_2,i_4,\ell_4,i,k)p_{i_2}(\ell_2)p_{i_4}(\ell_4) \quad (2\text{-}17)$$

$$D'_{i_1} = \frac{\partial D_{i_1}}{\partial p_i(k)} = \sum_{\ell_1,\ell_3=1}^{L} C_2(i_1,\ell_1,i_3,\ell_3,i,k)p_{i_3}(\ell_3) \quad (2\text{-}18)$$

$$D'_{i_2} = \frac{\partial D_{i_2}}{\partial p_i(k)} = \sum_{\ell_2,\ell_4=1}^{L} C_2(i_2,\ell_2,i_4,\ell_4,i,k)p_{i_4}(\ell_4) \quad (2\text{-}19)$$

$$D_{i_1} = \sum_{\ell=1}^{L} Q_{i_1}^{(2)}(\ell) \quad (2\text{-}20)$$

and

$$D_{i_2} = \sum_{\ell=1}^{L} Q_{i_2}^{(2)}(\ell) \qquad (2\text{-}21)$$

These notations are made clear in Fig. 2-2. Again it is to be noted that compatibilities like $C_2(i_1,\ell_1,i_3,\ell_3,i,k)$ need not be equal to $C_2(i,k,i_1,\ell_1,i_3,\ell_3)$, since it depends upon the manner of computation. Now the iteration of p_i's is given by,

$$p_i^{(n+1)}(k) = p_i^{(n)}(k) + \rho_i^{(n)} P_i^{(n)}\left(\frac{\partial J^{(j)}}{\partial p_i(k)}\right) \qquad (2\text{-}22)$$

$$k = 1,\ldots,L;\ i = 1,\ldots,N \text{ and } j = 1,2$$

where the projection of the gradient is given by

$$P_i^{(n)}\left(\frac{\partial J^{(j)}}{\partial p_i(k)}\right) = \frac{\partial J^{(j)}}{\partial p_i(k)} - \frac{1}{L}\sum_{k=1}^{L}\frac{\partial J^{(j)}}{\partial p_i(k)} = A_1^{(j)} \qquad (2\text{-}23)$$

$$j = 1,2$$

if none of the components of $\vec{p}_i$ are zero. Appendix A describes the computation of the projection when some of the components of $\vec{p}_i$ are zero. Normally, $\rho_i^{(n)}$ is kept constant for all units during each iteration and is determined to have the largest possible value such that p's at the (n+1)th iteration still lie in the bounded convex region of LN dimensional Euclidean space defined by $\sum_{k=1}^{L} p_i(k) = 1$ and $p_i(k) \geq 0$, $i = 1,\ldots,N$. However, to obtain a faster convergence rate $\rho_i^{(n)}$ is obtained as

$$\rho_i^{(n)} = \alpha \cdot \min_i [\max_k \rho_i^{(n)}(k)] \qquad (2\text{-}24)$$

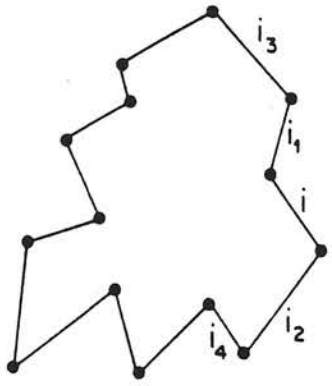


Fig. 2.2 Explanation of notations

where α is a constant between 0 and 1 and can be used to control the rate of convergence and

$$\rho_i^{(n)}(k) = \begin{cases} \dfrac{1-p_i^{(n)}(k)}{A_1^{(j)}}, & A_1^{(j)} > 0 \\ \dfrac{p_i^{(n)}(k)}{A_1^{(j)}}, & A_1^{(j)} < 0 \end{cases} \qquad (2\text{-}25)$$

A side effect of computing $\rho_i^{(n)}$ for every unit is that we may not be following the gradient exactly. However, it can be expected that we are approximately in the direction of the gradient and the criterion (2-12) is still maximized. Although the criterion normally increases, sometimes it decreases slightly at some iteration and then it continually increases. This is because $\rho_i^{(n)}$ in (2-24) is too large for some of the units.

2.3 Details of the Shape Matching Algorithm

In this section we present details of the shape matching algorithm whose block diagram is shown in Fig. 2-1.

Polygonal Approximation

Polygonal approximation for the model and object is obtained by using the algorithm proposed by Rosenfeld and

Johnston [2-38], which detects the points of high curvature. The algorithm works as follows. Let $R = \{(X_i, Y_i)\}_{i=1}^{n}$ be the sequence of points describing a closed curve so that $(X_1, Y_1) = (X_n, Y_n)$. At every point of the sequence, smoothed k-curvature is evaluated as,

$$C_{ik} = (\vec{a}_{ik} \cdot \vec{b}_{ik}) / |\vec{a}_{ik}| \; |\vec{b}_{ik}|$$

where

$$\vec{a}_{ik} = (X_i - X_{i+k}, Y_i - Y_{i+k})$$
$$\vec{b}_{ik} = (X_i - X_{i-k}, Y_i - Y_{i-k})$$

C_{ik} is the cosine of the angle between the vectors $\vec{a}_{ik}$ and $\vec{b}_{ik}$ so that $-1 \leq C_{ik} \leq 1$, and $C_{ik} = -1$ for a straight line and +1 for the sharpest angle of 0 degrees. Thus, the local maxima of C_{ik} or the local minima of the angle corresponds to the points of maximum curvature, which serve as the vertices in the polygonal approximation. Although this method has certain shortcomings [2-39], it works well if the largest k is smaller than the distances between successive angles along the curve. Normally smoothing factor k is taken to be between 1/5 and 1/20 of the perimeter. It is possible to use other methods for polygonal approximation such as split and merge technique of Pavlidis and Horowitz [2-40].

Features Derived from Polygonal Approximation

Some of the features that can be derived from the polygonal approximation of the boundary of an object are-

1) Length of a segment.

2) Intervertices distance.

3) Slope of a segment.

4) Angle between the two segments called the interior angle, and

5) Angle between the two segments as shown in Fig. 2-3. This angle corresponding to a vertex is equal to the angle between the two straight lines, where one line is obtained by extending the line joining this vertex and its neighboring counter clockwise vertex and the other line is obtained by extending the line joining the two clockwise neighboring vertices. This angle is named the exangle.

When scale invariant features are desired, slope, interior angle and exangle features can be used. For rotation invariance interior angle, exangle, length of a segment and intervertices distance can be used. For rotation as well as scale invariance interior angle or exangle or a combination of them can be used in the initial assignment of probabilities.

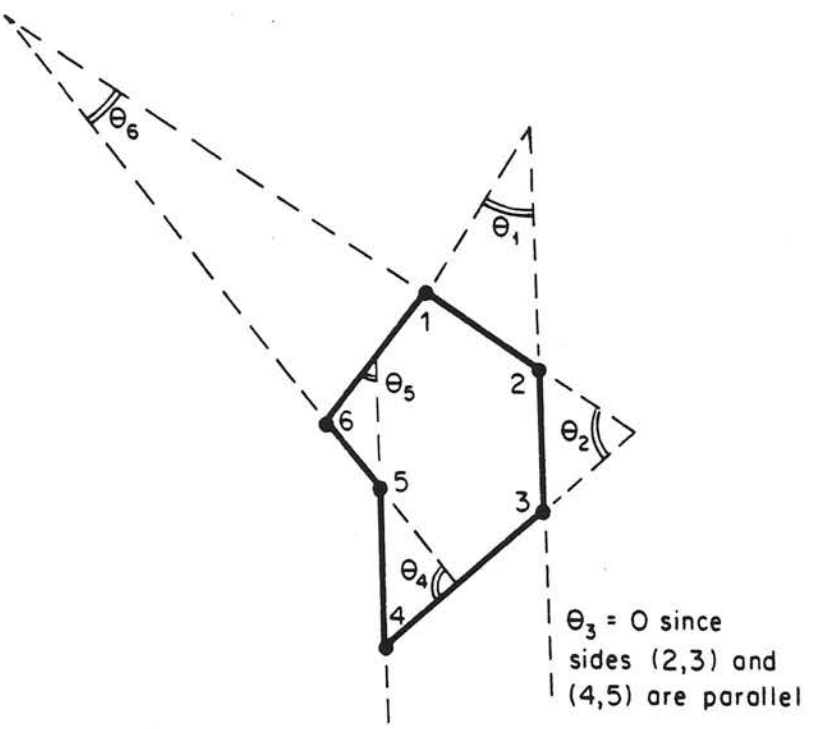


Fig. 2.3 Illustration of the definition of exangle

Initial Assignment of Probabilities

The initial assignment of probabilities for a unit is obtained by comparing its feature values with the feature values of all the segments of the object. Depending upon the type of invariance desired, we may need the weight of a particular feature. In general, the quality of correspondence of a unit i to an object segment k is given by,

$$M(T_i,O_k) = \sum_{p=1}^{P} |f_{tp} - f_{op}|W_p \qquad (2\text{-}26)$$

where P is the total number of features

f_{tp} = pth feature value for the model segment

f_{op} = pth feature value for the object segment

W_p = weight factor for the pth feature

Note that for a perfect match $M(T_i,O_k) = 0$ and for a poor match $M(T_i,O_k)$ will be large. The initial probabilities chosen proportional to

$$\frac{1}{1+M(T_i,O_k)}, \quad k = 1,\ldots,L-1$$

are normalized so that they sum to 1. We may not need weight factors if only one type of features is used. However, if we combine length and angles, then we need the weights to account for their importance and the different range of values. Note that the initial assignment of

probabilities involve unary relations.

Computation of Compatibilities

The compatibility function determines the degree by which the assignments of two or three neighboring units are compatible with each other. There are at least 4 ways of computing the compatibilities at the first and second stage of hierarchy. At the first stage computation of $C_1(i,k,j,\ell)$ involves binary relations and at the second stage $C_2(i,k,i-1,\ell_1,i+1,\ell_2)$ involves a subset of ternary relations.

First Method

At the first stage, we compute a transformation TR from a unit T_i to label O_k, i.e., TR: $T_i \rightarrow O_k$. TR consists of scale, rotation and translation in the X and Y directions. This transformation is applied to the unit T_j ($j = i-1$ or $i+1$) and the error between the transformed T_j and O_ℓ is computed as,

$$M(TR(T_j),O_\ell) = \sum_{p=1}^{P} |f_{t'p} - f_{op}| W_p \qquad (2\text{-}27)$$

where $f_{t'p}$ = pth feature value for the transformed unit, and the other quantities are similar to those defined in (2-26).

Note that here the features may be slope and length of a segment, so we shall need the weights for these features, if the matching error is based on them. However, it is possible to avoid these parameters in the computation of compatibilities, if we use only the distance between the ends of O_ℓ and transformed T_j as the matching error between two segments i.e., matching error $M(TR(T_j),O_\ell)$ = AB + CD (refer fig. 2-4). Now the compatibility at the first stage is given by

$$C_1(i,k,j,\ell) = \frac{1}{1+M(TR(T_j),O_\ell)}$$

The problem with this method of computing the compatibilities is that they are not symmetric i.e., $C_1(i,k,j,\ell) \neq C_1(j,\ell,i,k)$. As we have seen, the computations of gradient requires $C_1(j,\ell,i,k)$, so if this method is used we shall also require the computation of $C_1(j,\ell,i,k)$. Moreover, since we are using only one transformation, compatibilities so obtained will not be very accurate compared to the other three methods described below.

At the second stage to compute $C_2(i,k,i_1,\ell_1,i_2,\ell_2)$ using the first method we still find one transformation as described above and use it to compute the error between the transformed i_1 and ℓ_1 (Error 1) and transformed i_2 and ℓ_2 (Error 2). Then

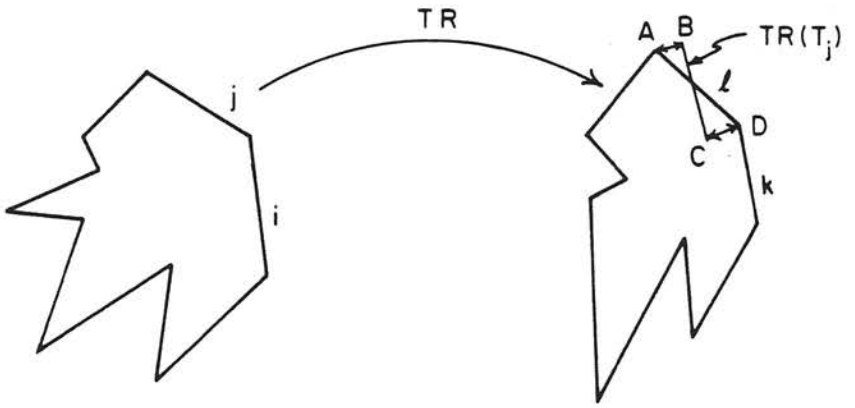


Fig. 2.4 Matching distance error = AB + CD

$$C_2(i,k,i_1,\ell_1,i_2,\ell_2) = \frac{1}{1+\text{Error } 1+\text{Error } 2} \quad .$$

Because of the problems of asymmetry and inaccuracy, this method is not used..

Second Method

Unlike the first method, here we find two transformations TR1 and TR2 such that

$$\text{TR1:} \quad T_i \rightarrow O_k$$

and

$$\text{TR2:} \quad T_j \rightarrow O_\ell$$

Now the average rotation, average scale and average translation (in the X and Y directions) of these two transformations are computed. The transformation associated with these parameters called TV, is now applied to the unit i and unit j and the matching errors between the transformed units and the segments O_k and O_ℓ are computed as in the first method and finally,

$$C_1(i,k,j,\ell) = \frac{1}{1 + \text{Total Error}}$$

At the second stage instead of finding two transformations, we find three transformations and take the average of these values. This average transformation is then applied to units i, i_1, i_2 and the total error between the transformed units and object segments k, ℓ_1,

and ℓ_2 is computed to get $C_2(i,k,i_1,\ell_1,i_2,\ell_2)$ as in the first stage compatibility computation.

Third Method

This method is similar to the second method in that we compute two transformations TR1 and TR2. Now TR1 is applied to T_j giving matching error $M(TR1(T_j),O_\ell)$ and TR2 is applied to T_i giving matching error $M(TR2(T_i),O_k)$. Average of this error is taken and

$$C_1(i,k,j,\ell) = \frac{1}{1 + \text{Average Error}}$$

At the second stage, we will find three transformations and the average error will be the average of six error terms and the compatibility

$$C_2(i,k,i_1,\ell_1,i_2,\ell_2) = \frac{1}{1 + \text{Average Error}}$$

Fourth Method

In this method we compute mathematically the best transformation from units i and j such that the sum of the squares of the error between the transformed units and the object segments is minimum. Here we can use only distance for the computation of matching error (unlike the first three methods where in principle we could use a combination of slope and length) so that the error criterion is linear and linear least squares techniques can be applied. For example, let the beginning and end

coordinates of the segments i, j, k and ℓ be given by (X1,Y1), (X2,Y2), (DX1,DY1), (DX2,DY2), (R1,S1), (R2,S2), (U1,V1) and (U2,V2) respectively then

$$MX = b$$

where

$$M = \begin{bmatrix} U1 & -V1 & 1 & 0 \\ V1 & U1 & 0 & 1 \\ U2 & -V2 & 1 & 0 \\ V2 & U2 & 0 & 1 \\ R1 & -S1 & 1 & 0 \\ S1 & R1 & 0 & 1 \\ R2 & -S2 & 1 & 0 \\ S2 & R2 & 0 & 1 \end{bmatrix}, \quad X = \begin{bmatrix} \lambda\cos\theta \\ \lambda\sin\theta \\ X_0 \\ Y_0 \end{bmatrix}, \quad b = \begin{bmatrix} DX1 \\ DY1 \\ DX2 \\ DY2 \\ X1 \\ Y1 \\ X2 \\ Y2 \end{bmatrix}$$

and λ is a scaling factor, θ is the rotation and X_0 and Y_0 are the translations in the X and Y directions respectively. The above set of equations is an overdetermined system. It can be transformed as

$$M^T MX = M^T b$$

and now it will be 4x4 system, which can be uniquely solved for λ, θ, X_0, Y_0. This computed transformation can then be applied to obtain compatibilities at the first stage by using the second method described above.

At the second stage we will have M as a 12x4 matrix and b a column vector of size 12x1. It can be solved exactly as in the first stage case to obtain the best transformation and compatibilities at the second stage. This method requires more time for the computation of compatibilities, than any of the other methods. Note that

the transformations used in this method allows polygons to undergo rotation, translation and scaling deformations only. However, a more general transformation can be easily computed. Also note that the second, third, and fourth method of compatibility computation lead to symmetric compatibilities.

Initial Probability and Compatibility for the Nil Class

The assignment of initial probability and compatibility to the nil class is very important, since for some of the units there may not be any corresponding object segment. Note that O_L is the nil class.

Assignment of Initial Probability to the Nil Class $p_i(L)$

There are at least three ways for the initial assignment of the probability to the nil class.

1. All the classes are assumed to be equally likely, then

$$p_i(\text{nil}) = p_i(0) = p_i(1) \ldots = p_i(L-1) = 1/L;$$

where L is the total number of classes. However, it is not a good assignment since the results of the stochastic labeling scheme depend on the initial assignment and it is better to use the available feature information rather than not using such information.

2. $p_i(nil)$ is assigned a small constant value, depending upon the a priori information that we may have about the possible number of matches. Normally, we have taken $p_i(L)$ between 0.05 to 0.25. The actual value is not critical, however, it affects the convergence of probabilities, hence the number of iterations required to achieve the desired result.

3. If the number of units is not very large, then $p_i(L)$ can be taken as

$$p_i(L) = 1 - \max_k p_i(k), \qquad k = 1,\ldots,L-1$$

After the assignment of probability to the nil class in the above two cases, probabilities are again normalized so that they sum to 1 for L classes.

Assignment of compatibilities Involving Nil Class

At the first stage of hierarchy the compatibility is $C_1(i,k,j,\ell)$. The compatibility involving nil class are assigned as follows. (Note that O_L is the nil class).

$$C(i,k,j,nil) = p_i(k),$$

and, $C_1(i,nil,j,\ell)$ = small constant usually between 0.05 and 0.25 where, ℓ varies over all classes. Normally this small constant is taken equal to $p_i(L)$.

At the second stage of hierarchy the compatibility is given by $C_2(i,k,i_1,\ell_1,i_2,\ell_2)$. Compatibilities involving nil classes are assigned as follows.

$C_2(i,k,i_1,nil,i_2,\ell_2) = C\ (i,k,i_2,\ell_2)$

$C_2(i,k,i_1,\ell_1,i_2,nil) = C\ (i,k,i_1,\ell_1)$

$C_2(i,k,i_1,nil,i_2,nil) = p_i(k)$

$C_2(i,nil,i_1,\ell_1,i_2,\ell_2)$ = small constant usually between 0.05 and 0.25. Normally it is taken equal to the $p_i(L)$.

where ℓ_1 and ℓ_2 vary over all classes.

Strategies That Lead to Faster Computation

Following are some of the strategies that have been used to obtain faster convergence of the probabilities.

1) We set a probability value $p_i(k)$ to zero if it is less than a certain threshold such as 10^{-4} or if it is less than a specified percentage of the largest component of $\vec{p}_i$. When one or more of the components of $\vec{p}_i$ becomes zero, we don't compute the gradients and compatibilities for them and suitably take care of it in the computation of the projection of the gradient.

2) We set a probability vector $\vec{p}_i$ to the unit vector if any of the components of $\vec{p}_i$ becomes greater than a certain

threshold, say 80%. The compatibilities and gradients are not subsequently computed for this unit.

3) We compute compatibility and gradient at the first and second stage only for a limited number of most likely assignments of neighbors for a given unit. Usually this number is taken to be 1 or 2.

4) We use the shape matching algorithm in a hierarchical manner.

All of these features have been incorporated in the shape matching program and lead to a marked reduction in the computation time. Effect of these features will be discussed in the next section, where we present shape matching examples from synthetic, aerial and industrial domains.

2.4 Examples and Comments

In this section we present several examples for the shape matching of two-dimensional objects using the hierarchical stochastic labeling technique developed in the previous sections. We discuss the computational aspects of the technique. The results of the labeling are used to compute the relative rotation between the observed object and the model. We are seeking only the approximate matches as the exact matches do not exist when we deal

with the real data.

The technique requires a number of parameters. Before presenting the examples we provide a discussion of these parameters and their effects on the labeling.

Discussion of Parameters

1. Smoothing factor (k) used in the polygonal approximation - largest k should be smaller than the distances between successive angles along the boundary. Normally it is taken between 1/5 to 1/20 of the perimeter.

2. Weights of features - we may require the weights of features (length and slope of a segment, interior angle and exangle) to account for their importance and range of values. Any a priori knowledge about the type of deformations and changes in the shape of an object can be incorporated into these weights. For the examples presented in this work, weights are needed only in the initial assignment of probabilities as the compatibility computation is based on the matching distance error (fig. 2.4). Interior angle and slope vary between 0 and 360 degrees and exangle between 0 and 90 degrees. Normally, when we have used all the features, slope is assigned the least weight,

interior angle and exangle about the same weight and length the maximum weight. The idea is that for a unit change in length how much variation in the other features can take place so that their contribution to the matching function of (2-26) is about the same. When only rotation invariant features (length, interior angle and exangle) are used, we have used the different weights for length, interior angle and exangle. Since the segmentation results are not perfect (usually we have missing or extra segments) and the polygonal approximation is rough, features are noisy but the exact values of the weights are not critical to the success of the stochastic labeling, although the results of labeling will be slightly different with the changes in the weights. The inverse of weights for the interior angle, exangle, slope and length will be denoted as a 4-tuple in the tables associated with various examples. The presence of a dash as one of the tuples indicate that the particular feature is not used in the initial probability assignment.

3. Compatibility computation - In the previous sections we presented four methods of computing the compatibilities. The third method based on the average error is normally used in the examples

presented in this work. This method is chosen because it provides symmetric compatibilities and its computation is faster than the fourth method which computes the exact transformation. Also the third method is a better choice over the second method because it does not use the average transformation. (Results obtained with the second and fourth method were similar to those reported here). Since we have used the matching distance error in the compatibility computation, weights of the features are not required in the computation of compatibilities.

4. Parameters used to control the stochastic labeling process - there are two critical questions related to any relaxation process in general [2-41]. They are -
(a) How do we evaluate the progress of the labeling process?
(b) How many iterations do we need and how do we stop?

The answer to the first question is provided by the formulation of the hierarchical stochastic labeling technique which explicitly maximizes the global criteria and finds the local maxima closest to the original labeling. It guarantees that consistency will increase and ambiguity will decrease as the

process progresses. This is not true for many relaxation schemes which often converge to results which are quite poor although the first few iterations provide significant improvement, (Peleg [2-42] suggests methods for evaluating the labelings based on the initial stochastic labeling and the probabilistic model).

To answer the second question, we need four parameters, the number of iterations at the first stage (ITER1), the number of iterations at the second stage (ITER2), the upper threshold on probabilities (UPTHLD) and the lower thresholding factor on probabilities (LFACT).

In the examples presented in this work on the shape matching of two-dimensional objects, we have allowed the maximum number of units and classes to be 42 and 31 respectively. Normally, in these examples ITER1 is taken about 3 and ITER2 about 5 or more until all the units are firmly assigned, i.e., their probability vectors become unit vectors and it is this time when the process is terminated.

The upper threshold on probabilities (UPTHLD) is needed to force a permanent assignment of a unit when any one of the components of the vector corresponding

to it, has a component which is equal to or greater than this threshold. Normally this is taken between 0.75 and 0.9. Changing this threshold by small amounts has very little or no effect on the labeling. Increasing this threshold requires more iterations and thus more computation time and some assignments may be lost or gained. Since in our formulation of the stochastic labeling, we explicitly deal with the nil class (no match class), the units which do not match very well with any of the segments in the object are assigned to this class when the process is terminated. From a practical stand point, use of this threshold is essential, otherwise we are faced with the problems of how many iterations are needed and how to evaluate the results i.e., separate the correct assignments from the incorrect ones on the basis of probability values. Thresholding reduces the computation time and provides us a degree of confidence in the labeling.

For the lower threshold on probabilities we have two options. In the first we have a fixed threshold such as 10^{-4}. If any of the components of a probability vector associated with a unit goes below this threshold, it is set to zero. However, the use of a constant threshold is not fair, since in the beginning of the labeling process, probability values

are small (the range of values is also small) and as the process progresses, the range of values gets larger so at the later iterations, the labeling process is slowed down by the low probabilities values and thus leads to increased computation time. We have chosen to vary this threshold with the probability values at every iteration for every unit. After an iteration we find the maximum value of the components of a probability vector $\vec{p}_i$. Let the maximum component value be $p_i(k)$. Then we compute the threshold $p_i(k)/LFACT$. If any of the components of $\vec{p}_i$ is less than this threshold, it is set to zero. The selection of lower threshold on probabilities in this manner substantially improves the computation time (a factor of 2 to 5 over the constant threshold selection of 10^{-4}), while the effect on the labeling is insignificant. Normally we have varied LFACT between 10 and 30 depending upon our confidence in the initial assignment of probabilities of the units.

5. Number of neighboring labels - In the computation of compatibilities (and gradients) at the first and second stage of hierarchy, normally we have restricted the number of most likely neighboring labels. Considering more alternatives than 1 or 2 substantially increases the computation time while the

performance is not improved. This was not totally obvious without experiments. The reason for this is that the probability value for the second, third and other neighboring alternatives is much less than the most likely one and they contribute very little to the computation. Price [2-36] has also obtained the similar result on the number of neighbors in the matching of aerial images.

6. Value of α in eq. (2-24) - the value of α ($0 < \alpha < 1$) provides the control on how fast we converge to the solution. Normally it is taken as 0.99.

7. Nil class value - It is needed in the initial assignment of probabilities and compatibilities. The assignment of this value is important. Normally, we have varied it between 0.05 and 0.25. The results of labeling are not critically dependent on the actual value used. A higher value requires less computation time. Normally, it is taken on the high side of its range when we expect a small number of matches, otherwise it is taken on the low side.

Examples

In using the shape matching algorithm in principle it does not matter whether we match an object with the model

or vice versa. Generally it may be preferred to label the segments of the object or model, whichever has the smaller number of segments, because in that case the labeling of a smaller number of units is required. At present our 2-D shape matching program can handle 42 units and 31 classes. However, it is possible to deal effectively with 100 or more segments (classes) by considering only a limited number of best classes (this has been implemented in the shape matching of three-dimensional objects described in chapter 5). The larger number of segments will provide a better polygonal approximation of the boundary of the object.

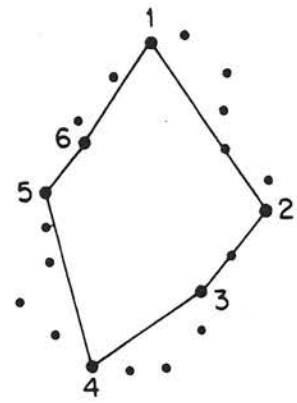
We show the matching results in the form of a table. Only the labels with the highest probability are shown. In the bracket we also indicate the probability of the most likely assignment. Computation time for various examples is also shown in the corresponding tables. In the present implementation of the program we do not store the compatibility values when we compute the consistency vector. We recompute them when the gradient (the most expensive of all the operations) is needed. So the computation time can be reduced if we store compatibility values. Now we present synthetic, aerial and industrial examples of shape matching to illustrate the capabilities and powers of the hierarchical stochastic labeling

technique.

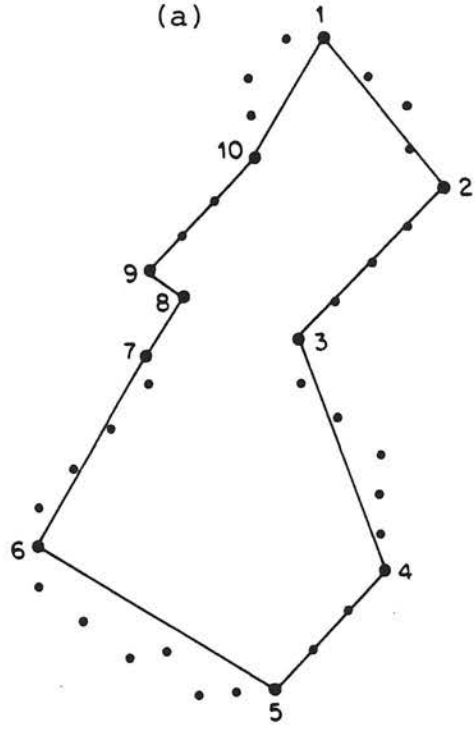
Example 2.1

Fig. 2.5 shows a model and an object. Dotted points show the boundary of the model or the object. The perimeter of the model and object consist of 21 and 38 points respectively. The polygonal approximation is obtained with a smoothing factor of 4. Note that the upper portion of the model is a noisy version of the object so their polygonal approximations are different. This makes the matching problem a little more complicated than in [2-1] where Davis introduces the noise after the polygonal approximation so that the number of segments remains the same before and after the introduction of noise. Table 2.1 shows the expected assignments of the units of the model. Tables 2.2 to 2.5 illustrate various results.

Table 2.2 shows the results of the hierarchical stochastic labeling technique when the number of iterations used at the first and second stage are 3 and 4 respectively. The parameters used in the labeling process are shown in this table. The number of neighboring labels is taken as one in the computation of compatibility and gradient (eqs. 2-3, 2-5, 2-13 and 2-15 to 2-19). Only the interior angle is used in the initial probability



(a)



(b)

Fig. 2.5 (a) Model, Perimeter = 21, Number of vertices = 6

(b) Object, Perimeter = 38, Number of vertices = 10

Table 2.1

Expected assignments for the units of the model in Example 2.1

Units of the Model	Label
1	1
2	2
3	2 or 11
4	9 or 11
5	9 or 10 or 11
6	10

Table 2.2

Label of units of the Model. Reduced compatibility and gradient computation (Example 2.1)

Smoothing factor = 4, $\alpha = 0.99$, $p_i(nil) = 0.15$, ITER1 = 3, ITER2 = 4, UPTHLD = 0.9, LFACT = 20, No. of neighboring labels = 1, Inverse weights = (1, -, -, -)

Units of the Model	Labels at different iterations				
	0	1	3	1	4
1	1(.38)	1(.48)	1(.70)	1(.81)	1(1.0)
2	5(.31)	5(.30)	2(.41)	2(.63)	2(1.0)
3	7(.53)	7(.55)	11(.61)	11(1.0)	11(1.0)
4	1(.44)	1(.42)	1(.60)	11(.61)	9(1.0)
5	4(.32)	4(.29)	11(.33)	11(.46)	10(1.0)
6	10(.75)	10(.83)	10(1.0)	10(1.0)	10(1.0)
Value of Criteria	-	.71	1.42	.93	1.14
		$J^{(1)}$		$J^{(2)}$	

Total Computation Time = 18.27 seconds

Table 2.3

Label of units of the model. Full compatibility and gradient computation (Example 2.1). Parameters same as in Table 2.2 except No. of neighobrs = 11

Units of the model	Labels at different iterations				
	0	1	3	1	4
1	1(.38)	1(.47)	1(1.0)	1(1.0)	1(1.0)
2	5(.31)	5(.32)	2(.52)	2(.60)	2(1.0)
3	7(.53)	7(.67)	7(1.0)	7(1.0)	7(1.0)
4	1(.44)	1(.48)	1(.70)	11(.73)	9(1.0)
5	4(.32)	4(.32)	4(.26)	10(.30)	9(.50)
6	10(.75)	10(.83)	10(1.0)	10(1.0)	10(1.0)
Value of Criteria	-	.81	1.14	.76	.95
		$J^{(1)}$		$J^{(2)}$	

Total Computation Time = 231.83 seconds

Table 2.4

Label of units of the model. Parameters same as in Table 2.2 except ITER1 = 6 and ITER2 = 0

Units of the model	Labels at different iterations				
	0	1	3	5	6
1	1(.38)	1(.48)	1(.70)	1(1.0)	1(1.0)
2	5(.31)	5(.30)	2(.41)	2(1.0)	2(1.0)
3	7(.53)	7(.55)	11(.61)	11(1.0)	11(1.0)
4	1(.44)	1(.42)	1(.60)	1(1.0)	1(1.0)
5	4(.32)	4(.29)	11(.33)	9(.75)	9(1.0)
6	10(.75)	10(.83)	10(1.0)	10(1.0)	10(1.0)
Value of Criterion	-	.71	1.42	1.70	2.03

$J^{(1)}$

Total Computation Time = 4.24 seconds

Table 2.5

Label of units of the model. Parameters same as in Table 2.3 except ITER1 = 6, ITER2 = 0

Units of the model	Labels at different iterations				
	0	1	3	5	6
1	1(.38)	1(.47)	1(1.0)	1(1.0)	1(1.0)
2	5(.31)	5(.32)	2(.52)	9(1.0)	9(1.0)
3	7(.53)	7(.67)	7(1.0)	7(1.0)	7(1.0)
4	1(.44)	1(.48)	1(.70)	11(1.0)	11(1.0)
5	4(.32)	4(.32)	4(.26)	9(.51)	9(.61)
6	10(.75)	10(.83)	10(1.0)	10(1.0)	10(.10)
Value of Criterion	-	.81	1.14	1.29	1.57

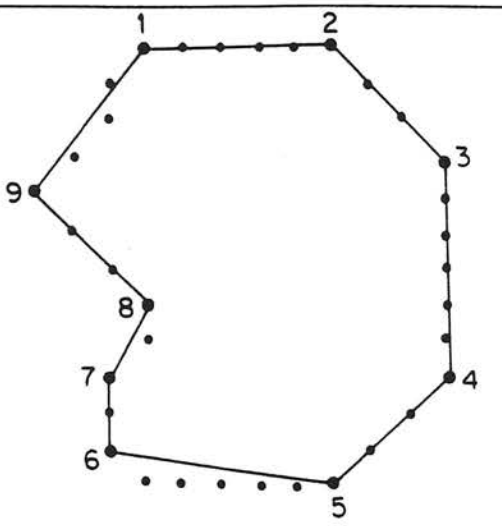
$J^{(1)}$

Total Computation Time = 18.74 seconds

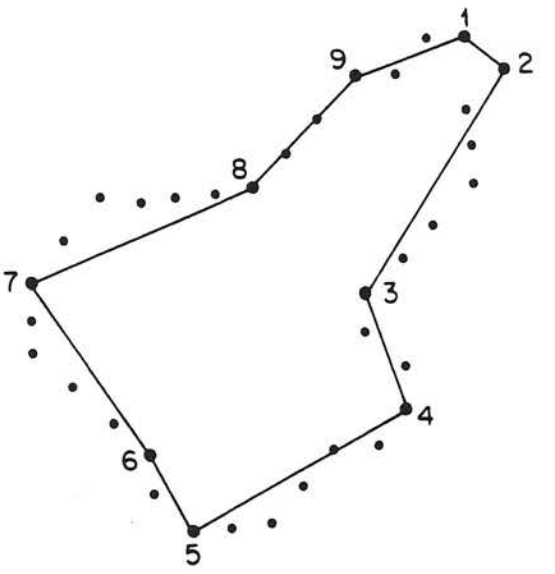
assignment. The computation time is 18.27 seconds. In Table 2.3 we have used all the neighboring labels and other parameters remain the same as in Table 2.2. The computation time for the results shown in table 2.3 is 231.83 seconds. So when we use all the neighboring labels, computation time increases by more than an order of magnitude. Also the label of unit 3 of the model is wrong in table 2.3. Thus not only the computation time but also the labeling results are degraded when we use all the labels. The results of labeling in table 2.2 are good. Label 11 is the nil class. Table 2.4 and 2.5 show the results when only 6 iterations of the first stage are used. Now the label of unit 4 in Table 2.4 and the label of unit 3 in Table 2.5 are wrong. This illustrates the need for the second stage which corrects the mistakes of the first stage, if any. Subsequent examples presented here support this fact.

Example 2.2

Figure 2.6 shows two models and an object. Model 1 and Model 2 occlude each other to form an apparent object shown in fig. 2.6(c). Note that there is a change of scale for the Model 1 in the apparent object; it may be because of segmentation (different lighting conditions) or changes in the model itself even if the camera position



(a)

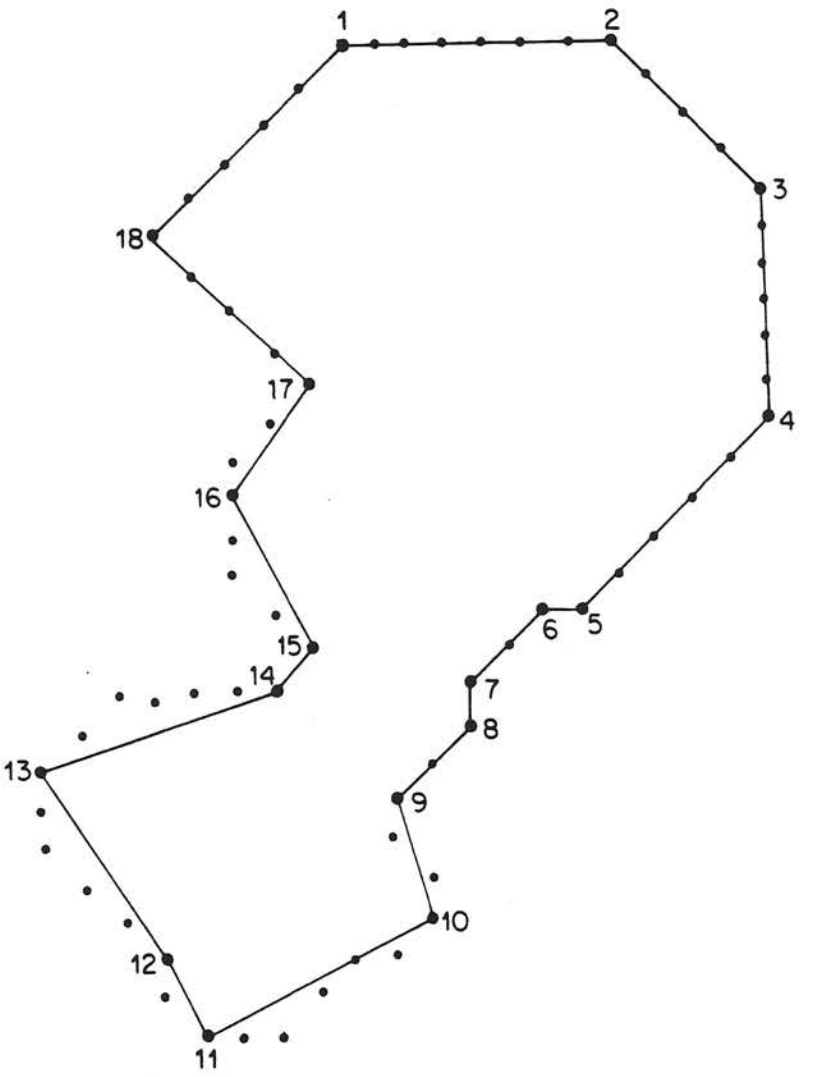


(b)

Fig. 2.6 (a) Model 1, Perimeter = 34, Number of vertices = 9

(b) Model 2, Perimeter = 35, Number of vertices = 9

(c) Object, Perimeter = 67, Number of vertices = 18



(c)

Fig. 2.6 (CONTINUED)

remains fixed. The polygonal approximation is obtained with a smoothing factor of 4. It is given that the changes in the shape are small so a weighting of all the features is used. Table 2.6 shows the expected labels for the units of Model 1 and Model 2. Table 2.7 and 2.8 show the labels of Model 1 and Model 2 at different iterations of the stochastic labeling process. Note that all the assignments of Model 1 are correct and for Model 2 the assignment of unit 1 is wrong. Although the probability of assignment for unit 1 to label 19 increases, label 1 is finally assigned to this unit. Label 19 is the nil class. The parameters used in the labeling process are shown in Table 2.7. This example is also described in Chapter 3 to illustrate the occlusion algorithm. There it will be seen that unit 1 of Model 2 is not assigned to label 1.

Example 2.3

Fig. 2.7 shows two (1024x1024 pixels) aerial images of San Francisco taken at different times. Note that these images are rotated with respect to each other. Fig. 2.8 shows three regions corresponding to the same area in the images (the lower left portion in Fig. 2.7(a) or the upper left portion in Fig. 2.7(b)) consisting of the golden gate park. These regions are obtained by using the recursive region splitting method [2-43, 2-44] of

Table 2.6

Expected assignments of the units of Model 1 and Model 2 in Example 2.2

Model 1		Model 2	
Unit	Label	Unit	Label
1	1	1	19
2	2	2	8 or 19
3	3	3	9
4	4	4	10
5	5 or 19	5	11
6	15 or 19	6	12
7	16	7	13
8	17	8	14 or 19
9	18	9	19

Table 2.7

Label of units of the Model 1. Example 2.2

Smoothing factor = 4, α = 0.99, p_i(nil) = 0.15, ITER1 = 3, ITER2 = 6, UPTHLD = 0.8, LFACT = 20, No. of neighboring labels = 1, Inverse weights = (3, 2, 5, 2)

Units of the Model 1	Labels at different iterations					
	0	1	3	1	3	6
1	1(.20)	1(.33)	1(.50)	1(.53)	1(1.0)	1(1.0)
2	2(.51)	2(.62)	2(1.0)	2(1.0)	2(1.0)	2(1.0)
3	3(.63)	3(.74)	3(1.0)	3(1.0)	3(1.0)	3(1.0)
4	4(.28)	4(.43)	4(.71)	4(1.0)	4(1.0)	4(1.0)
5	19(.15)	19(.24)	19(.30)	19(.33)	19(.37)	5(1.0)
6	19(.15)	19(.21)	19(.27)	19(.31)	19(.45)	19(1.0)
7	19(.15)	19(.26)	16(.39)	16(.51)	16(.74)	16(1.0)
8	17(.30)	17(.43)	17(1.0)	17(1.0)	17(1.0)	17(1.0)
9	18(.29)	18(.46)	18(.68)	18(1.0)	18(1.0)	18(1.0)
Value of Criteria	-	1.03	1.56	1.33	1.44	1.44
		$J^{(1)}$			$J^{(2)}$	

Total Computation Time = 52.79 seconds

Table 2.8

Label of units of the Model 2. Example 2.2. Parameters same as in Table 2.7

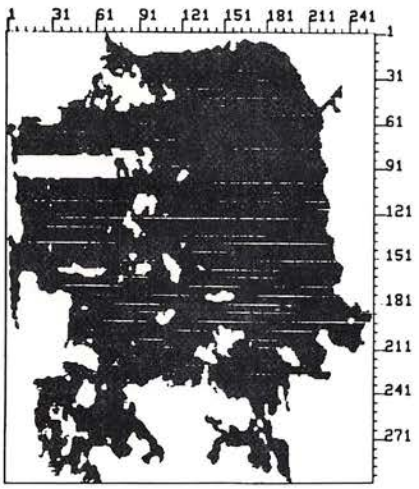
Units of the Model 2	Labels at different iterations					
	0	1	3	1	3	5
1	19(.15)	19(.26)	19(.36)	19(.43)	19(.65)	1(1.0)
2	10(.15)	19(.20)	19(.24)	19(.30)	19(.45)	19(1.0)
3	9(.24)	9(.56)	9(1.0)	9(1.0)	9(1.0)	9(1.0)
4	10(.67)	10(1.0)	10(1.0)	10(1.0)	10(1.0)	10(1.0)
5	11(.66)	11(1.0)	11(1.0)	11(1.0)	11(1.0)	11(1.0)
6	12(.65)	12(1.0)	12(1.0)	12(1.0)	12(1.0)	12(1.0)
7	13(.66)	13(1.0)	13(1.0)	13(1.0)	13(1.0)	13(1.0)
8	19(.15)	19(.22)	19(.27)	19(.32)	14(.42)	14(1.0)
9	1(.16)	1(.23)	1(.33)	1(.36)	19(.40)	19(1.0)
Value of Criteria	-	1.81	2.89	2.74	2.36	3.19
		$J^{(1)}$			$J^{(2)}$	

Total Computation Time = 42.9 seconds

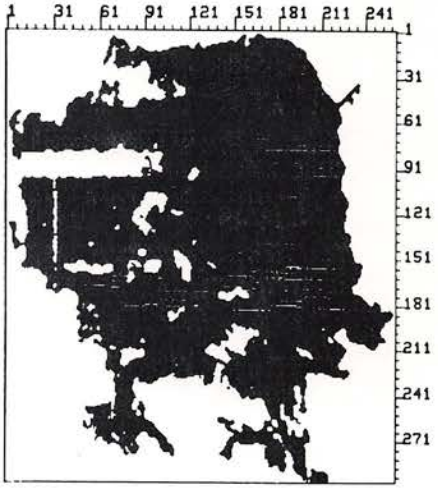
(a)

(b)

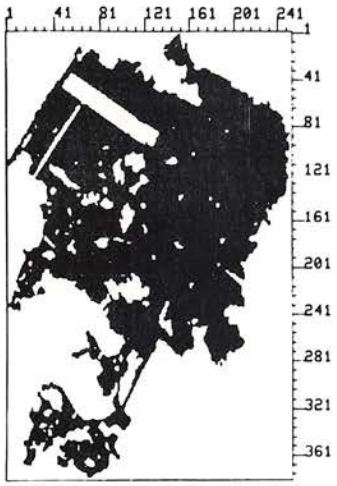
Fig. 2.7 Two aerial images of San Francisco taken at different times



(a) Size = 299 Rows by 258 Columns



(b) Size = 297 Rows by 261 Columns

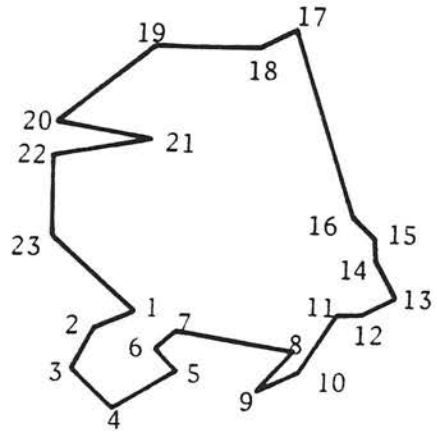


(c) Size = 381 Rows by 253 Columns

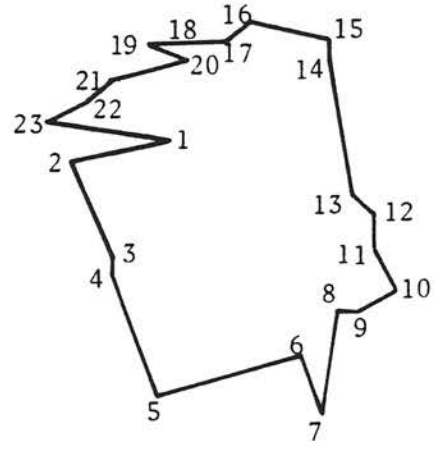
Fig. 2.8 Regions obtained using a recursive region splitting technique of segmentation. Regions in figs. 2.8(a) and 2.8(b) are obtained from fig. 2.7(a) with slightly different parameters in the segmentation scheme. Region in fig. 2.8(c) is obtained from fig. 2.7(b) using the same parameters used to obtain figs. 2.8(a) and 2.8(b). Regions shown are at different scales

segmentation. Regions in Fig. 2.8(a) and (b) are obtained from the image in Fig. 2.7(a) with slightly different parameters in the segmentation scheme. Fig. 2.8(c) shows the region obtained from the image in Fig. 2.7(b). This region remained the same when the parameters were slightly changed to obtain the segmentation of the image in Fig. 2.7(b). Regions shown in Fig. 2.8 are at different scales. The size of the regions in Fig. 2.8(a), (b) and (c) are 299 rows by 258 columns, 297 rows by 261 columns and 381 rows by 253 columns respectively. Since the region in Fig. 2.8(c) did not change with slightly different parameters, we consider it as the model and regions in Fig. 2.8(a) and (b) as objects.

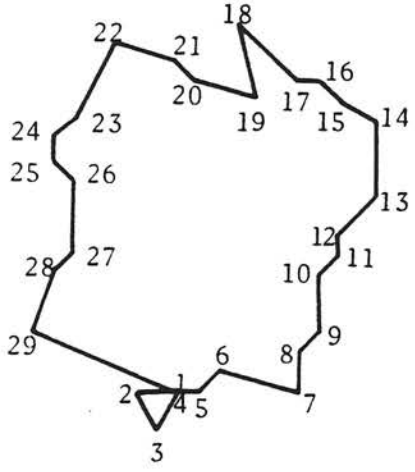
In order to reduce the computational complexity we reduce the regions shown in Fig. 2.8 by 14 times. This is done by looking in a window of size 14x14 at every pixel of the binary image (1 represents a region and 0 the background). If there are more than 1/2 of the number of pixels with a value of 1 in the window area, then a value of 1 is taken in the reduced picture, otherwise it is taken as zero. The polygonal approximation of the reduced regions with a smoothing factor of 4 are shown in Fig. 2.9. We want to match the shape of objects (Fig. 2.9(a) and (b)) against the model (Fig. 2.9(c)) so object segments are units in this example. These shapes



(a) Object 1, Number of segments = 23



(b) Object 2, Number of segments = 23



(c) Model, Number of segments = 29

Fig. 2.9 Polygonal approximation of the regions shown in fig. 2.8 after reducing them by 14 times. A smoothing factor of 4 is used to obtain the polygonal approximations. Figs. 2.9(a), (b) and (c) correspond to figs. 2.8(a), (b) and (c) respectively

appear to be very different from each other. The left side of the golden gate park in Fig. 2.8(a) and (b) did not close, whereas in Fig. 2.8(c) it is closed. This is typical of shape matching complexity when we deal with real images. There is a clue of similarity of shapes in Figs. 2.8 or 2.9. Segments 13 and 17 of object 1 match with segments 7 and 14 or 16 of the model respectively. Similarly segments 10 and 15 or 16 of object 2 match with the segments 7 and 14 or 16 of the model. Since the images are rotated by a small amount with respect to each other, we also use slope of a segment in the initial assignment of probabilities but give it a smaller weight relative to the weights of other features. Results of shape matching are shown in Table 2.9 and 2.10. Most of the assignments are very reasonable and correct although a few are incorrect. Label 30 is the nil class.

From the results of labeling, it is possible to compute the relative rotation between the model and the object. This is done by using the following formula,

$$\text{Average relative rotation} = \left(\sum_{\substack{i=1 \\ \text{label} \neq \text{nil}}}^{N} \left| \begin{array}{l} \text{slope of the } i^{th} \text{ unit} - \\ \text{slope of the assigned label} \\ \text{to the } i^{th} \text{ unit} \end{array} \right| \right) \Big/ \begin{array}{l} \text{no. of} \\ \text{units not} \\ \text{assigned to} \\ \text{nil class} \end{array} \qquad (2\text{-}28)$$

where N is the number of units. The idea behind this formula to compute the rotation is that although some

Table 2.9

Label of segments of the object 1. Example 2.3

Smoothing factor = 4, α = 0.99, p_i(nil) = .08, ITER1 = 5, ITER2 = 7, UPTHLD = 0.8, LFACT = 10, No. of neighboring labels = 1, Inverse weights = (4, 3, 5, 1)

Segments of the Object 1	Labels at different iterations					
	0	1	5	2	4	7
1	30(.08)	27(.20)	27(1.0)	27(1.0)	27(1.0)	27(1.0)
2	28(.16)	28(.22)	28(.78)	28(1.0)	28(1.0)	28(1.0)
3	25(.08)	29(.11)	29(.25)	28(.44)	28(1.0)	28(1.0)
4	5(.08)	30(.10)	3(.33)	3(.39)	3(1.0)	3(1.0)
5	30(.08)	5(.11)	5(.23)	5(.32)	5(.51)	5(1.0)
6	30(.08)	4(.14)	4(.39)	5(.56)	5(1.0)	5(1.0)
7	6(.10)	6(.11)	6(.77)	6(1.0)	6(1.0)	6(1.0)
8	19(.10)	19(.12)	1(.25)	7(.44)	7(.67)	7(1.0)
9	30(.08)	3(.13)	3(.30)	5(.31)	5(1.0)	5(1.0)
10	5(.09)	3(.20)	3(1.0)	3(1.0)	3(1.0)	3(1.0)
11	4(.21)	4(.35)	4(1.0)	4(1.0)	4(1.0)	4(1.0)
12	30(.08)	5(.15)	5(.47)	5(.79)	5(1.0)	5(1.0)
13	7(.11)	7(.20)	7(1.0)	7(1.0)	7(1.0)	7(1.0)
14	30(.08)	8(.13)	8(.58)	8(1.0)	8(1.0)	8(1.0)
15	30(.08)	30(.10)	9(.16)	9(.36)	9(.50)	8(1.0)
16	17(.08)	17(.14)	17(.63)	13(1.0)	13(1.0)	13(1.0)
17	30(.08)	30(.11)	30(.17)	18(.19)	16(.28)	16(1.0)
18	30(.08)	30(.12)	20(.35)	20(.71)	20(1.0)	20(1.0)
19	30(.08)	30(.12)	30(.17)	23(.26)	23(.44)	23(1.0)
20	18(.10)	2(.12)	18(.40)	30(.63)	30(1.0)	30(1.0)
21	1(.11)	21(.12)	21(.44)	19(1.0)	19(1.0)	19(1.0)
22	22(.14)	22(.17)	22(.53)	22(.77)	22(1.0)	22(1.0)
23	30(.08)	30(.10)	26(.24)	26(.33)	26(.53)	26(1.0)
Value of Criteria	-	.898	2.236	1.604	1.688	1.718
		$J^{(1)}$			$J^{(2)}$	

Total Computation Time = 243.75 seconds

Table 2.10

Label of segments of the object 2. Example 2.3. Parameters same as in Table 2.9 except ITER1 = 6, ITER2 = 6

Segments of the Object 2	Labels at different iterations					
	0	1	6	1	4	6
1	1(.15)	1(.20)	1(.64)	1(1.0)	1(1.0)	1(1.0)
2	2(.11)	2(.19)	2(.49)	2(.57)	2(1.0)	2(1.0)
3	30(.08)	27(.12)	27(.36)	27(.34)	3(.40)	3(1.0)
4	28(.09)	28(.15)	28(.68)	28(.79)	28(1.0)	28(1.0)
5	30(.08)	30(.12)	29(.31)	29(.35)	29(1.0)	29(1.0)
6	30(.08)	2(.14)	2(.66)	2(1.0)	2(1.0)	2(1.0)
7	3(.12)	3(.28)	3(1.0)	3(1.0)	3(1.0)	3(1.0)
8	4(.12)	4(.20)	4(1.0)	4(1.0)	4(1.0)	4(1.0)
9	30(.08)	5(.13)	5(.40)	5(.66)	5(1.0)	5(1.0)
10	7(.11)	7(.20)	7(1.0)	7(1.0)	7(1.0)	7(1.0)
11	30(.08)	8(.11)	8(.38)	8(.42)	8(1.0)	8(1.0)
12	11(.08)	11(.12)	9(.49)	9(.65)	9(1.0)	9(1.0)
13	30(.08)	30(.10)	12(.27)	12(.30)	11(.62)	11(1.0)
14	30(.08)	13(.15)	13(.55)	14(1.0)	14(1.0)	14(1.0)
15	14(.14)	14(.31)	14(1.0)	14(1.0)	14(1.0)	14(1.0)
16	30(.08)	16(.16)	16(.61)	16(1.0)	16(1.0)	16(1.0)
17	30(.08)	30(.13)	17(.33)	19(.38)	19(1.0)	19(1.0)
18	30(.08)	15(.10)	19(.20)	19(.26)	19(.45)	20(1.0)
19	18(.12)	18(.13)	18(1.0)	18(1.0)	18(1.0)	18(1.0)
20	1(.09)	19(.18)	19(.80)	19(1.0)	19(1.0)	19(1.0)
21	30(.08)	23(.13)	22(.41)	20(.63)	20(1.0)	20(1.0)
22	23(.08)	23(.11)	23(.44)	23(.49)	22(.61)	22(1.0)
23	18(.11)	29(.11)	29(.35)	29(.72)	29(1.0)	29(1.0)
Value of Criteria	-	0.917	1.985	1.606	1.658	1.672
		$J^{(1)}$			$J^{(2)}$	

Total Computation Time = 219.20 seconds

labels may be wrong, it is expected that the slope of the matching segments will not be widely different. In practice we consider two cases when using this formula.

Case 1 If it is given that the relative rotation is small ($< 90^{\circ}$), we subtract any term greater than 180° in the summation of (2-28), from 360°. So all the terms contributing to the sum in (2-28) will be less than 180°.

Case 2 If the relative rotation is greater than 90°, we do not subtract any term in the summation of (2-28) from 360° as in case 1, but we neglect the terms contributing less than 30° and by the same amount the number of units is reduced in the denominator of (2-28).

Tables 2-11 and 2-12 show calculations of the relative rotation for the object 1 and object 2. The relative rotation between the object 1 and the model is found to be 36.1°. Similarly the relative rotation between the object 2 and the model is 35.5°. The actual rotation is about 35°. We use this method of finding relative rotation in other examples presented in the followings.

Example 2.4

In Fig. 2.10(a) we show the top view of a piece of car shock absorber. In Fig. 2.10(b) we show the

Table 2.11

Computation of relative rotation between the object 1 and the Model. Example 2.3

Units of the object 1	Slope of the units of object 1 in degrees a	Label of the units of object 1	Slope of the segments of the labels in degrees b	|a-b| in degrees
1	207	27	225	18
2	243	28	252	9
3	315	28	252	63
4	33.7	3	63.4	29.7
5	135	5	45	90
6	45	5	45	0.0
7	351	6	346	5
8	225	7	90	180
9	26.6	5	45	18.4
10	56.3	3	63.4	7.1
11	0.0	4	0.0	0.0
12	26.6	5	45	18.4
13	117	7	90	27
14	90	8	45	45
15	135	8	45	90
16	107	13	90	17
17	207	16	180	27
18	180	20	135	45
19	219	23	225	6
20	349	30	-	-
21	191	19	162	29
22	270	22	243	27
23	315	26	270	45
				Sum = 796.6

Relative Rotation = $796.6 \div 22 = 36.1^{o}$

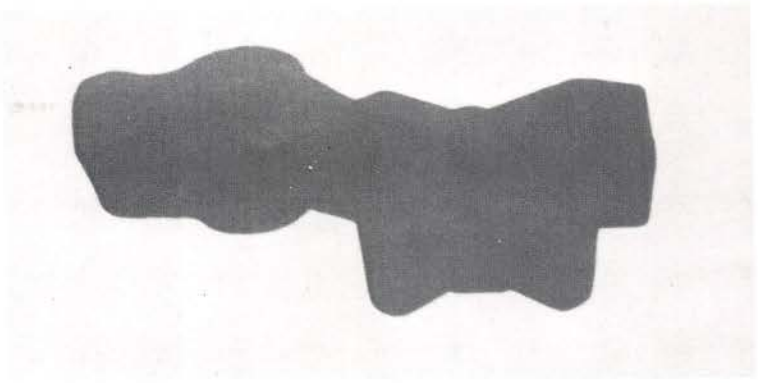
Table 2.12

Computation of the relative rotation between the object 2 and the Model. Example 2.3

Units of the object 2	Slope of the units of object 2 in degrees a	Label of the units of object 2	Slope of the segments of the labels in degrees b	|a-b| in degrees
1	191	1	180	11
2	292	2	297	5
3	270	3	63.4	206.6+
4	288	28	252	36
5	15.9	29	337	321.1+
6	288	2	297	9
7	78.7	3	63.4	15.3
8	0.0	4	0.0	0
9	26.6	5	45	18.4
10	117	7	90	27
11	90	8	45	45
12	135	9	90	45
13	98.1	11	90	8
14	90	14	153	63
15	166	14	153	13
16	225	16	180	45
17	180	19	162	18
18	180	20	135	45
19	333	18	284	49
20	194	19	162	32
21	225	20	135	90
22	207	22	243	36
23	351	29	337	14

+The contribution of these terms in the sum will be 360-value Sum = 8.17.1

Relative rotation = 817.1÷23 = 35.5°



(a)

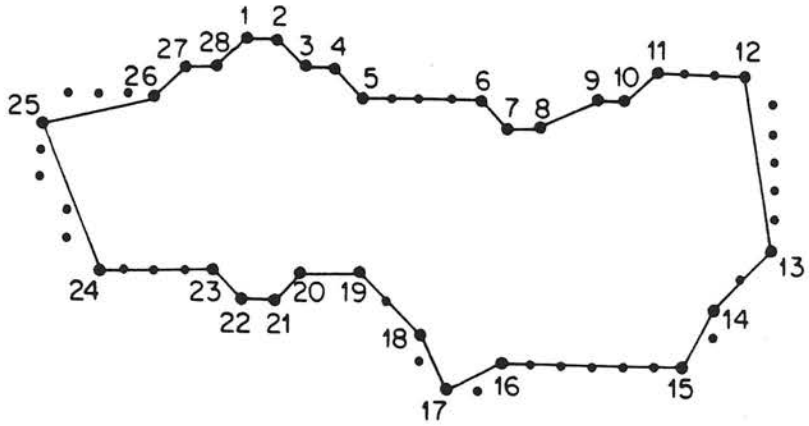
(b)

Fig. 2.10 (a) An automobile piece
(b) Superposition of two such pieces

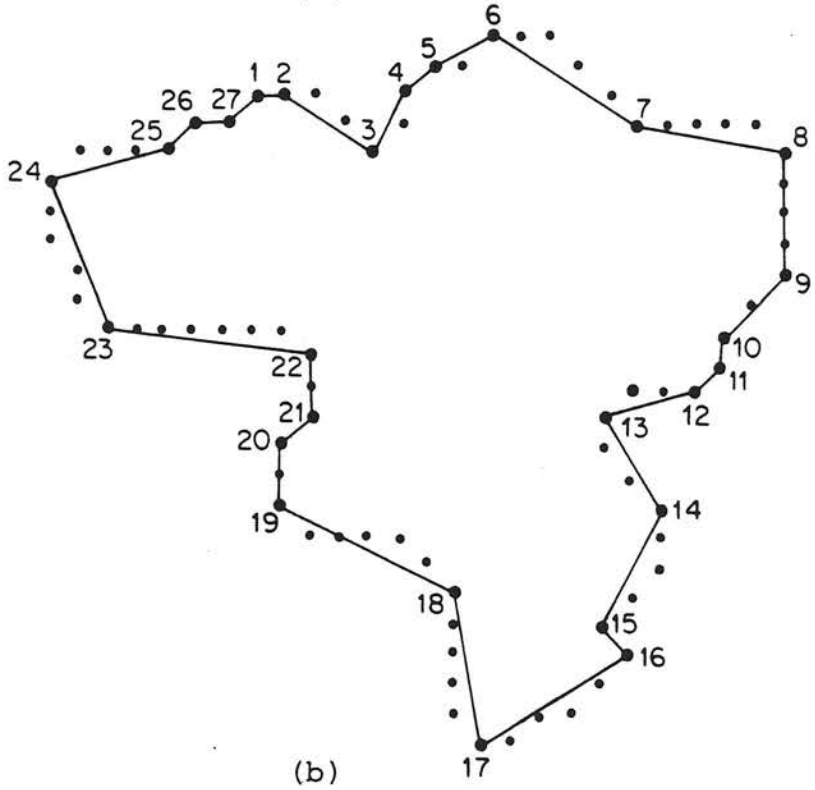
superposition of two such pieces, the one below being the one of Fig. 2.10(a). From a practical standpoint, it is important to identify in the shape of Fig. 2.10(b) (the observation) the visible part of the shape of Fig. 2.10(a) (the model). Fig. 2.11 shows the corresponding polygonal approximation with a smoothing factor of 4. In this example (N = L = 28). Table 2.13 shows the parameters used and the results of the hierarchical stochastic labeling algorithm at different iterations. In this example the units 8, 9, 10, 11 and 26, 27, 28, 1 are assigned to the same labels because their local structure is the same. The labeling could correctly identify the visible part of the model within the observation. In this example we used a weighted sum of features in the initial assignment of probabilities. The relative rotation is found to be 13^{o} using the procedure used in the last example. The actual rotation is about 5^{o}.

Example 2.5

This example is similar to the last example in that we want to identify the visible part of the model shown in Fig. 2.12(a) within the observation 2.12(b). The polygonal approximation with a smoothing factor of 10 is shown in Fig. 2.13. The polygonal approximation is crude since we limited the number of segments in the object to



(a)



(b)

Fig. 2.11 (a) Model, Number of segments = 28
(b) Object, Number of segments = 27

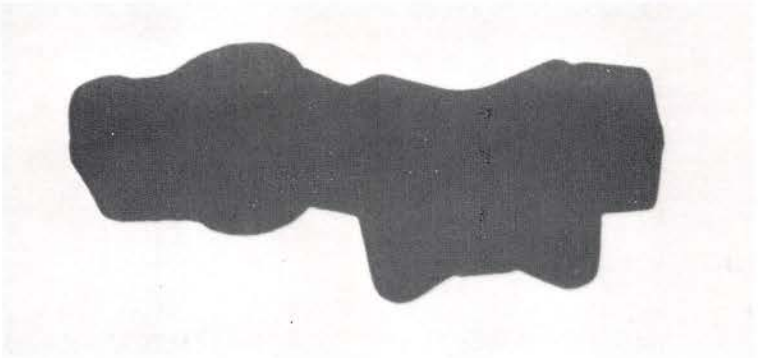
Table 2.13

Labels of units of the Model. Example 2.4

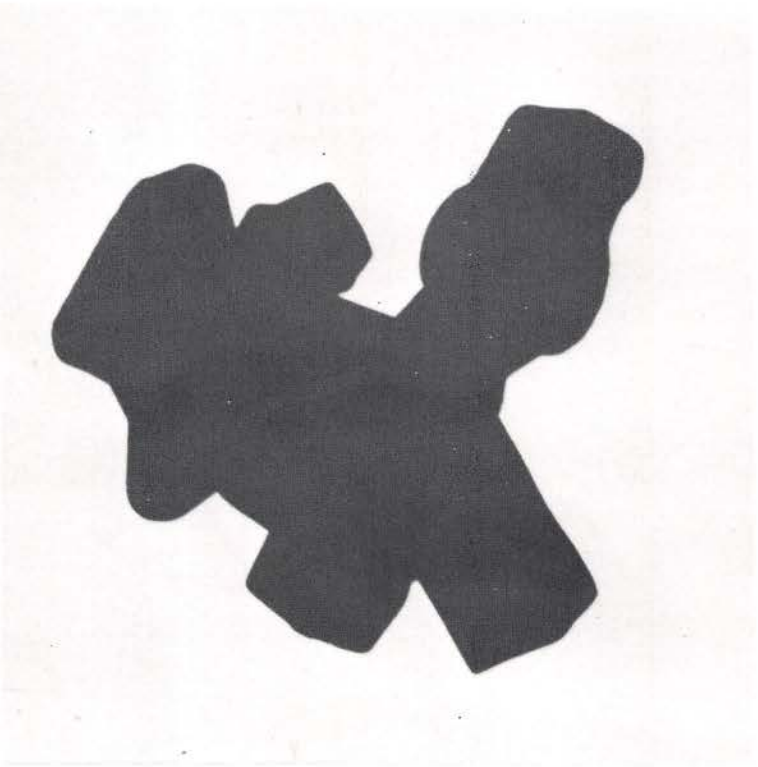
Smoothing factor = 4, $\alpha = 0.99$, $p_i(\text{nil}) = 0.25$, ITER1 = 3, ITER2 = 2, UPTHLD = 0.8, LFACT = 10, No. of neighboring labels = 1, Inverse weights = (4, 3, 5, 1)

Units of the Model	Labels at different iterations			
	0	1	3	2
1	28(.25)	1(.40)	1(1.0)	1(1.0)
2	28(.25)	28(.54)	28(.72)	28(1.0)
3	28(.25)	28(.49)	28(.68)	27(1.0)
4	28(.25)	28(.54)	28(.67)	26(1.0)
5	28(.25)	28(.49)	28(.73)	27(1.0)
6	28(.25)	28(.54)	28(.67)	26(1.0)
7	28(.25)	28(.57)	28(.10)	28(1.0)
8	28(.25)	28(.51)	28(.52)	25(1.0)
9	28(.25)	28(.37)	26(.60)	26(1.0)
10	27(.46)	27(.65)	27(1.0)	27(1.0)
11	28(.25)	28(.37)	28(.53)	1(1.0)
12	28(.25)	28(.51)	28(.59)	8(1.0)
13	28(.25)	28(.43)	28(.62)	28(1.0)
14	28(.25)	28(.65)	28(1.0)	28(1.0)
15	28(.25)	28(.51)	28(1.0)	28(1.0)
16	28(.25)	28(.66)	28(1.0)	28(1.0)
17	28(.25)	28(.55)	28(.76)	28(1.0)
18	28(.25)	28(.77)	28(1.0)	28(1.0)
19	28(.25)	28(.62)	28(1.0)	28(1.0)
20	28(.25)	28(.56)	28(1.0)	28(1.0)
21	28(.25)	28(.59)	28(1.0)	28(1.0)
22	28(.25)	28(.59)	28(1.0)	28(1.0)
23	28(.25)	28(.58)	28(.73)	22(1.0)
24	28(.25)	23(.52)	23(1.0)	23(1.0)
25	24(.48)	24(.67)	24(1.0)	24(1.0)
26	25(.44)	25(.62)	25(1.0)	25(1.0)
27	26(.46)	26(.67)	26(1.0)	26(1.0)
28	27(.46)	27(.62)	27(1.0)	27(1.0)
Value of Criterion	-	3.14	12.67	11.23
		$J^{(1)}$		$J^{(2)}$

Total Computation Time = 74.56 seconds

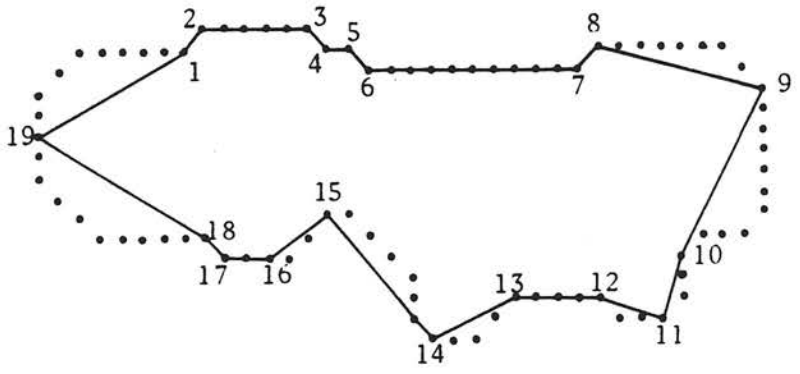


(a)

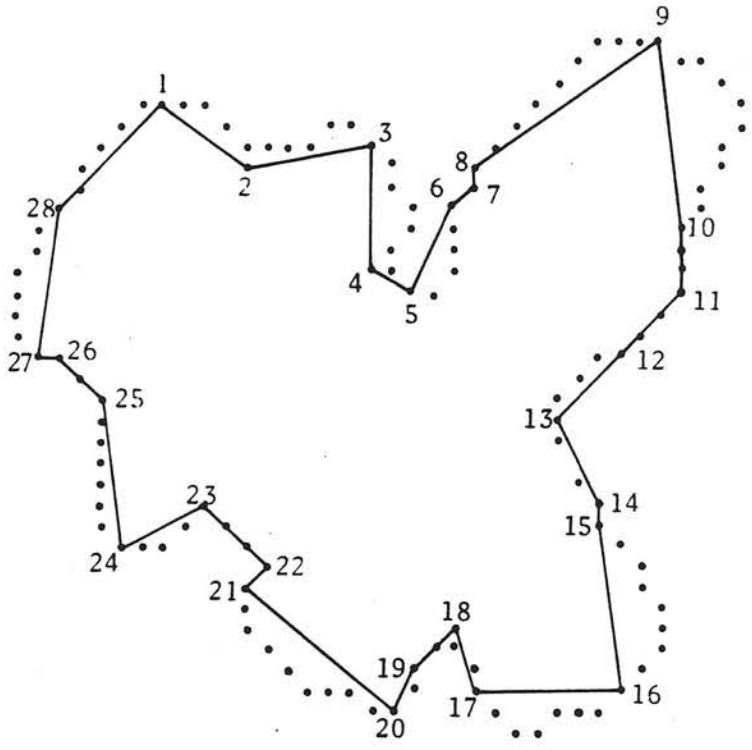


(b)

Fig. 2.12 (a) An automobile piece
(b) Superposition of two such pieces



(a)



(b)

Fig. 2.13 (a) Model, Number of segments = 19
(b) Object, Number of segments = 28

be less than 31. Table 2.14 shows the results of labeling. Label 29 is the nil class. Only the rotation invariant features are used in the initial probability assignment. Most of the labels are correct and a few, which are wrong, are reasonable because the local structure of the incorrect label matches better than the local structure of the correct label. The actual relative rotation between the model and object is 133.50°. Using the procedure outlined in Example 2.3 the relative rotation is found to be 134.35°.

Example 2.6

In this example we reverse the role of the model and object. We want to identify the composite object (model) of Fig. 2.14(b) within the object of Fig. 2.14(a). This was done because our program could not handle more than 31 units in the object. The images in Fig. 2.14 are 512x512, 8 bits. They were reduced by a factor of 12 and thresholded at 100 to get the binary image of the model and object. The polygonal approximation is obtained with a smoothing factor of 8 (Fig. 2.15). The results of the stochastic labeling are shown in Table 2.15. Only the rotation invariant features (i.e., interior angle, exangle and length of a segment) are used in the initial probability assignment. Label 30 is the nil class. Most

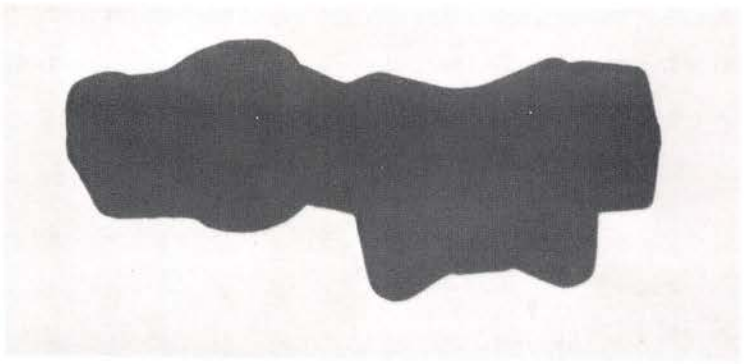
Table 2.14

Label of units of the Model. Example 2.5

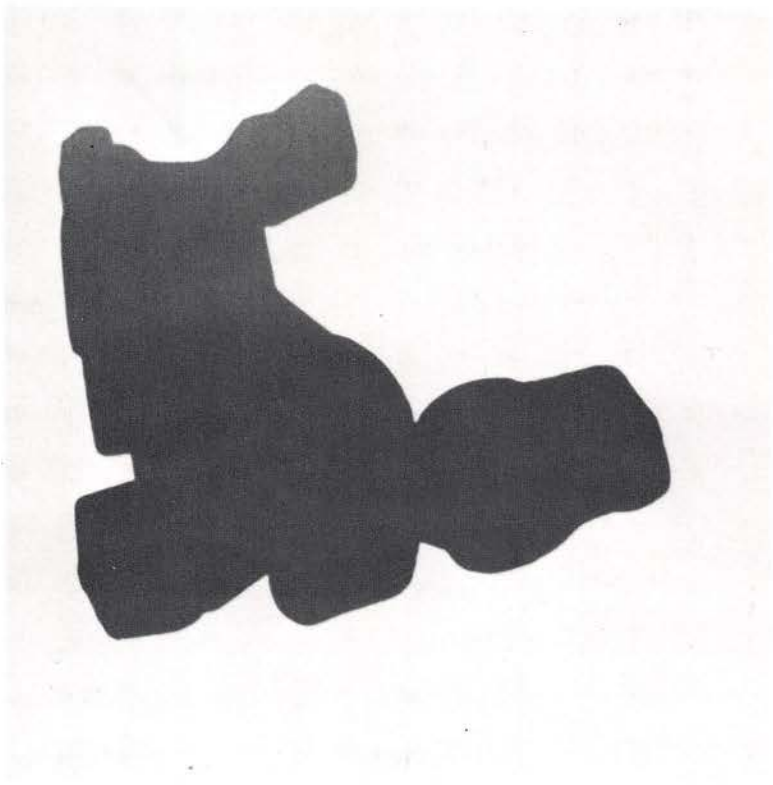
Smoothing factor = 10, α = 0.99, p_i(nil) = .15, ITER1 = 3, ITER2 = 9, UPTHLD = 0.8, LFACT = 20, No. of neighboring labels = 1, Inverse weights = (2, 12, -, 2)

Units of the Model	Labels at different iterations					
	0	1	3	1	7	9
1	29(.15)	29(.23)	29(.37)	29(.48)	29(1.0)	29(1.0)
2	11(.15)	11(.20)	11(.36)	11(.42)	29(1.0)	29(1.0)
3	29(.15)	29(.19)	29(.25)	29(.24)	11(.35)	11(1.0)
4	7(.28)	7(.38)	7(.59)	7(.69)	7(1.0)	7(1.0)
5	29(.15)	29(.18)	29(.25)	29(.26)	29(1.0)	29(1.0)
6	29(.15)	29(.19)	29(.25)	29(.36)	29(1.0)	29(1.0)
7	7(.32)	7(.42)	7(.58)	7(.71)	7(1.0)	7(1.0)
8	8(.16)	8(.19)	29(.29)	29(.34)	29(1.0)	29(1.0)
9	29(.15)	29(.19)	29(.25)	29(.29)	29(1.0)	29(1.0)
10	29(.15)	29(.20)	29(.25)	15(.27)	15(1.0)	15(1.0)
11	29(.15)	29(.19)	29(.24)	29(.28)	29(1.0)	29(1.0)
12	29(.15)	29(.21)	19(.28)	19(.34)	19(1.0)	19(1.0)
13	29(.15)	29(.20)	29(.27)	29(.32)	29(1.0)	29(1.0)
14	29(.15)	29(.19)	29(.22)	29(.25)	29(1.0)	29(1.0)
15	29(.15)	29(.29)	29(.23)	5(.24)	5(1.0)	5(1.0)
16	29(.15)	29(.22)	29(.28)	29(.29)	6(1.0)	6(1.0)
17	11(.20)	11(.26)	11(.43)	11(.54)	11(1.0)	11(1.0)
18	29(.15)	29(.22)	29(.35)	29(.44)	6(.53)	7(1.0)
19	9(.23)	9(.30)	9(.43)	9(.50)	9(1.0)	9(1.0)
Value of Criteria	-	1.33	2.65	3.11	3.97	3.99
		$J^{(1)}$			$J^{(2)}$	

Total Computation Time = 202.33 seconds

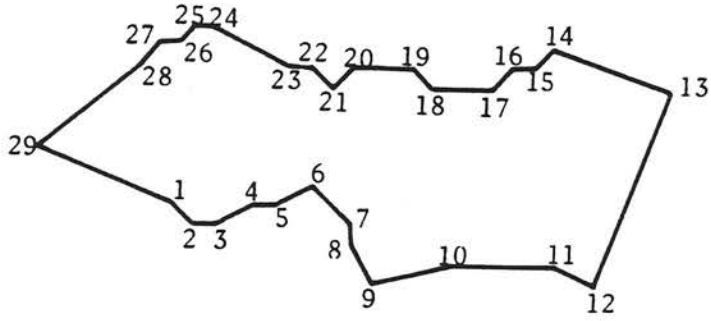


(a)

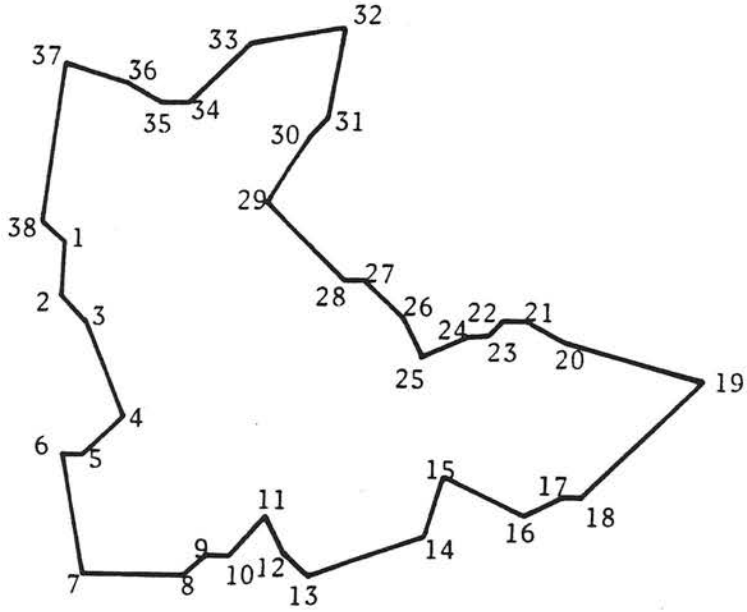


(b)

Fig. 2.14 (a) An automobile piece
(b) Superposition of two such pieces



(a)



(b)

Fig. 2.15 (a) Object, Number of segments = 29
(b) Model, Number of segments = 38

Table 2.15

Label of units of the Model. Example 2.6

Smoothing factor = 8, $\alpha = 0.99$, $p_i(\text{nil}) = 0.2$, ITER1 = 3, ITER2 = 7, UPTHLD = 0.8, LFACT = 25, No. of neighboring labels = 1, Inverse weights = (4, 3, -, 2)

Units of the Model	Labels at different iterations 0	3	1	7
1	30(.20)	30(.29)	30(.29)	18(1.0)
2	30(.20)	30(.35)	22(.48)	30(1.0)
3	30(.20)	30(.44)	30(.48)	30(1.0)
4	30(.20)	30(.39)	6(.46)	6(1.0)
5	30(.20)	30(.54)	30(.62)	30(1.0)
6	30(.20)	30(.41)	30(.44)	12(1.0)
7	30(.20)	30(.41)	30(.44)	30(1.0)
8	16(.24)	16(.38)	16(1.0)	16(1.0)
9	30(.20)	15(.28)	15(.29)	16(1.0)
10	30(.20)	30(.41)	30(.51)	14(1.0)
11	30(.20)	30(.51)	30(.53)	30(1.0)
12	30(.20)	30(.40)	30(.43)	8(1.0)
13	30(.20)	30(.43)	30(.46)	30(1.0)
14	30(.20)	30(.40)	30(.39)	20(1.0)
15	30(.20)	21(.45)	21(.48)	21(1.0)
16	30(.20)	30(.42)	30(.46)	30(1.0)
17	30(.20)	30(.36)	30(.40)	30(1.0)
18	30(.20)	30(.43)	30(.47)	22(1.0)
19	30(.20)	30(.44)	29(.48)	29(1.0)
20	30(.20)	30(.35)	30(.39)	30(1.0)
21	24(.31)	24(.66)	24(.76)	24(1.0)
22	16(.24)	25(.45)	25(.52)	25(1.0)
23	7(.33)	7(.47)	7(.46)	26(1.0)
24	30(.20)	30(.36)	30(.39)	30(1.0)
25	30(.20)	30(.43)	30(.45)	6(1.0)
26	30(.20)	30(.43)	30(.50)	30(1.0)
27	16(.23)	16(.33)	16(.36)	25(1.0)
28	30(.20)	30(.41)	30(.46)	30(1.0)
29	30(.20)	30(.45)	30(.47)	30(1.0)
30	30(.20)	30(.44)	30(.46)	30(1.0)
31	30(.20)	30(.47)	30(.50)	30(1.0)
32	30(.20)	30(.49)	30(.54)	13(1.0)
33	30(.20)	30(.47)	30(.51)	3(1.0)
34	30(.20)	30(.43)	30(.46)	30(1.0)
35	30(.20)	30(.36)	30(.43)	30(1.0)
36	30(.20)	30(.41)	30(.44)	24(1.0)
37	30(.20)	30(.41)	30(.43)	30(1.0)
38	30(.20)	30(.36)	30(.39)	30(1.0)
Value of Criteria	-	6.24	7.06	9.57
		$J^{(1)}$		$J^{(2)}$

Total Computation Time = 327.32 seconds

of the assignments are correct. Using the technique described in Example 2.3 to find out the relative rotation between the object and the model, the relative rotation is found to be 168.20°. The actual rotation is 165°.

Comments

The variety of examples presented in the above allow us to evaluate the capabilities of the hierarchical stochastic labeling technique. The success of the technique can be measured on the basis of two facts:

1) The final labeling should be as unambiguous as possible, and
2) It should be consistent with any a priori knowledge that we may have about the set of possible labelings.

Although the results of matching are not "perfect", and some wrong labels or multiple assignments do occur, the reasons for this are caused by several phenomena: the polygonal approximation is crude, i.e., the number of segments is small, local structure of the incorrect match is more similar than the correct one, segmentation of two images of the same scene is very different. The key assignments are correctly obtained. In the case of multiple assignments, an interpretation of the results may be required. The technique is quite effective in the shape matching applied to real images and it is able to

cope to a certain extent with the common problems of scene analysis such as noisy features, extra and missing segments and a large number of segments. The first stage does not correct all the mistakes of the initial assignment because it uses less world knowledge and some correct assignments are not among the early candidates. The second stage corrects mistakes of the labeling of the first stage, and since it uses more world knowledge, it increases our confidence in the results of matching. The results of matching can be used to compute the rotation.

The shape matching technique presented in this chapter is truly hierarchical in the following sense: At the first or second stage the criterion $J^{(1)}$ or $J^{(2)}$ increases of course. But at the first stage if we compute the criterion $J^{(2)}$ or at the second stage if we compute the criterion $J^{(1)}$, it also continually increases with the iteration. This has been verified experimentally for the examples presented in the above. Thus our computation of compatibilities is sound. At the first stage we have more world knowledge and more constraints (a subset of ternary relations) whereas at the first stage there is less world knowledge and constraints (binary relations), so the criterion of the first stage $J^{(1)}$ will be more than the criterion of the second stage $J^{(2)}$ either at the first or second stage of the labeling process.

It is possible to do the global maximization of the criteria $J^{(j)}$ (j = 1,2) by local computations only i.e., a processor for a unit communicates only with the neighboring processors to compute the gradient and the projection operator [2-34]. Thus a large amount of parallelism can be introduced. A processor associated with a unit performs simple operations mostly in parallel while a sequential process of going from one iteration to the next allows these processors to work towards the final goal in a coordinated fashion. The amount of computation per processor is of the order of L^2, where L is the number of classes. It is difficult to develop any useful absolute model for the complexity of the stochastic labeling process, because as the procedure iterates, many labels for a unit become zero and the complexity reduces. As a result the computation time per iteration for the later stages of the process is less than the early iterations. Normally, for 42 units and 31 classes, we never needed more than a total of 15 iterations of the first and second stages of stochastic labeling. In the worst case, only one label may be set to zero at an iteration, but it must be extremely rare as it never happened with the examples reported in this work. The computation time varied from 4 seconds to 5 minutes on a PDP-10 (KL-10 processor, performs about 1.8 million

operations per second). About 70% of this time is spent in computing the gradient and the computation of the gradient at the second stage is the most expensive. The algorithms presented in this work are implemented in SAIL [2-45]. Programs are written in a general fashion. There has been a small effort to implement relatively efficient code.

2.5 Summary

In this chapter we developed the hierarchical stochastic labeling technique to solve the segment matching problem for the shape matching of nonoccluded two-dimensional objects. The technique is truly hierarchical. The effect of various parameters used in the labeling process is described. The assignment of the nil class works well. The results of labeling are used to compute the relative rotation between an object and the model. In this study we considered only two levels of hierarchy. At the first level binary relations and at the second level a subset of ternary relations are used in the computation of compatibilities. With these two levels of hierarchy we achieved good matching results. The first stage alone does not resolve all the ambiguities of labeling even if the number of iterations is very large. The second stage helps in resolving those ambiguous

labelings. It is possible to generalize the shape matching algorithm to include higher levels of hierarchy. With the increase in levels, we are using more world knowledge, so the complexity of the processing will increase, but it is expected that the reliability of the assignment of units will also increase.

Examples are presented using synthetic, aerial and industrial images. The technique is robust, flexible and effective in dealing with real images compared to the other approaches based on chain code cross-correlation, features etc., although it is computationally more expensive. The computation time for the examples presented in this chapter varied from 4 seconds to 5 minutes. However, the technique allows the parallel implementation in hardware and the criteria can be maximized using only the local computations in a iterative manner.

CHAPTER 3

SHAPE MATCHING OF TWO-DIMENSIONAL OCCLUDED OBJECTS

3.1 Introduction

In this chapter we shall extend the hierarchical stochastic labeling technique presented in the last chapter for the shape matching of occluded objects. The class of shapes that we consider are represented by simple closed curves and are two-dimensional in nature. These shapes are approximated by polygons. Their vertices being the points of high curvature [3-1]. Objects participating in the occlusion may move, rotate, undergo significant changes in the shape and their scale may also change. In this section first we shall present the past work that has been undertaken to solve the occlusion problem in two-dimensions and then we shall present our approach and in the subsequent sections describe the occlusion algorithm, study its computational properties and present several examples.

Matching of occluded objects is one of the prime capabilities of any shape analysis system. Particularly in the analysis of a sequence of images, the occlusion

problem becomes of major importance in the task of modelling a dynamic environment [3-2 to 3-4]. Aggarwal and co-workers [3-5 to 3-7] consider the problem of modelling the image of a partially occluded object and use the derived model to track the object through partial or even complete occlusion. These authors work on simulated binary images, processing by systematically relaxing the constraints on object contours. A limited global feature based description is used to compare sequential images of a moving scene and maintain a description of the scene. Features used are invariant to rotation and translation. They also assume that all the objects are known to the program before any analysis is done. Aggarwal and Duda [3-5] in their study on tracking clouds that partially occlude each other take the polygonal approximation of the shape of objects. These polygons are assumed to be rigid, however, holes in the polygons are allowed. The concept of a true and a false vertex is used to detect occluding parts. Their matching approach is sequential and suboptimal. Chow and Aggarwal [3-6] consider planar curvilinear objects having no holes. Their basic concept of matching is the same as of Aggarwal and Duda [3-5]. They mention that their matching technique can be extended by matching the boundary of the images. However, approaches such as chain code cross-correlation etc. are

not suitable especially, if we are dealing with real images [3-8]. Martin and Aggarwal [3-7] represent the shape of an object by a sequence of straight lines which have been derived from the graph of total subtended angle versus arc length function. Their line fitting-process is the technique of iterative end-point fit [p. 338, 3-9] and their segment matching approach is heuristic and cumbersome. They consider the tracking of individual boundary segments which decomposes the contour of two overlapping object images during the tracking process into boundary parts related to the component objects. Yachida et al. [3-3] apply the ideas similar to Chow and Aggarwal [3-5] to the natural images of fish swimming in a vat. For the occlusion problem, they consider boundary matching in a relatively simple fashion. Perkins [3-10, 3-11] considered touching and overlapping industrial parts. His matching technique is based on boundary matching and makes use of the problem specific knowledge. McKee and Aggarwal [3-12] recognized partial views of 3-D objects by matching the description based on the edges, and shape boundaries are analyzed at the level of boundary chains. Roach and Aggarwal [3-13] considered the occlusion problem in a 3-D blocks world by using a sequence of 2-D camera images of blocks in 3-D space.

We view the occlusion problem in two-dimensions basically a boundary matching problem similar to other studies in [3-4, 3-6, 3-7 and 3-10]. However, our approach is different from theirs. The occlusion problem is treated as a segment matching problem which involves matching the segments of two or more actual objects with the apparent object, which is formed by the occlusion of these objects. Some segments of the actual objects may not match with any of the segments of the apparent object. Also the matching algorithm should not assign the same segment of the apparent object to the segments of different actual objects which are occluding. The shape matching algorithm presented in the last chapter is based on stochastic labeling technique which explicitly maximizes the criterion function. This algorithm is hierarchical so that it reduces the computation time and uses results obtained at low levels to improve the accuracy of results obtained at higher levels [3-14]. This algorithm is extended so that several such hierarchical processes are executed in parallel for every object participating in the occlusion and are coordinated in such a way that the same segment of the apparent object is not matched to the segments of different actual objects (Fig. 3-1). This is done by combining the gradient projection method of Chapter 2 with the penalty function

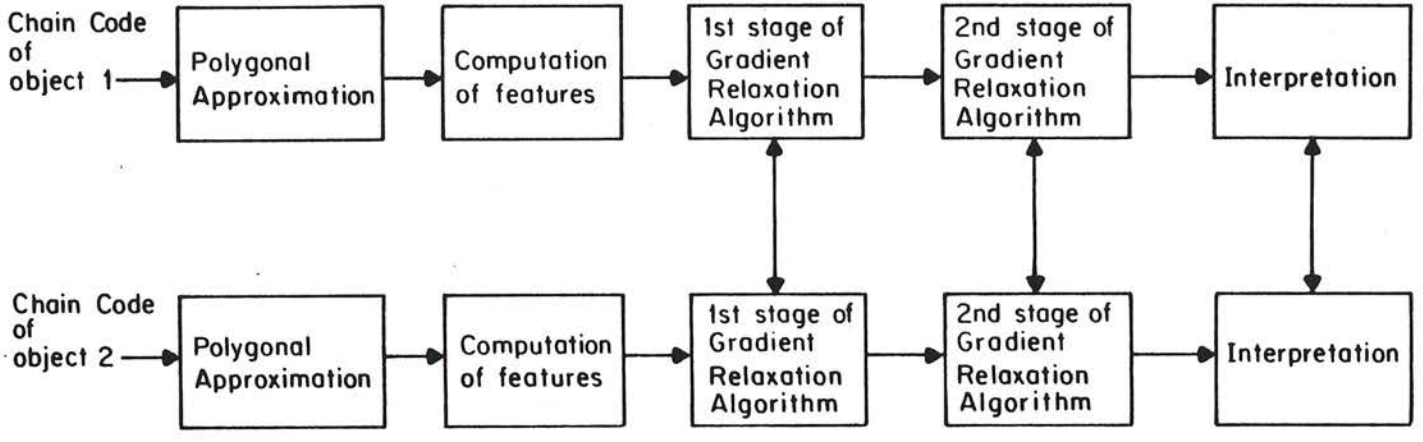


Fig. 3.1 Block diagram of the occlusion algorithm for the shape description of 2 occluding models using 2 stages of the coordinated hierarchical stochastic labeling technique based on the gradient projection method and penalty function approach

approach [3-15 to 3-17]. In the following section we shall formulate this problem as an optimization problem and present the occlusion algorithm in section 3.3. Section 3.4 includes synthetic, microscope and industrial examples in which 2 or 3 objects partially occlude. Finally, section 3.5 presents a summary of the chapter.

3.2 Problem Formulation

Consider a general case in which M ($\geq$ 2) actual objects, $(X_1,\ldots,X_M)$ occlude one another to form a single apparent object. In the following we shall designate the actual objects as models/templates and the apparent object as the object. Let a model X_m be represented by $X_m = (T_1,T_2,\ldots,T_{N_m})$, where N_m is the number of segments in the polygonal path representation of the model X_m. Similarly, let $O = (O_1,O_2,\ldots,O_{L-1})$ be the polygonal path representation of the object. The object has L-1 segments. We want to match the segments of the models against the segments of the object such that the following two conditions are satisfied.

1) None of the segments of the different models are assigned to the same segment of the object. We shall call this as the occlusion condition. This condition is necessary otherwise the labeling will be ambiguous.

2) One or more segments of the models which do not match to any of the segments of the object should be assigned to the nil class, i.e., no match class.

We are trying to identify parts of the models within the object. We shall designate the object segments as classes, and the model segments as units. Let the nil class be denoted by O_L. To each of the units T_i of a model X_m, we assign a probability denoted by $p_{im}(k)$ to belong to class O_k. This is conveniently represented as a vector $\vec{p}_{im} = [p_{im}(1),\ldots,p_{im}(L)]^T$. The set of all vectors $\vec{p}_{im}$ $(i = 1,\ldots,N)$ is called a probabilistic or stochastic labeling of the set of units.

As discussed in the last chapter the global criterion that measures the consistency and ambiguity of the labeling over the set of units of a model X_m is given by

$$J_m^{(n)} = \sum_{i=1}^{N_m} \vec{p}_{im} \cdot \vec{q}_{im}^{(n)} \quad , \quad n = 1,2 \qquad (3\text{-}1)$$

where

$$q_{im}^{(n)}(k) = \frac{Q_{im}^{(n)}(k)}{\sum_{\ell=1}^{L} Q_{im}^{(n)}(\ell)} \quad , \quad \begin{matrix} n = 1,2 \\ i = 1,\ldots,N_m \\ k = 1,\ldots,L \end{matrix} \qquad (3\text{-}2)$$

and,

$$Q_{ij,m}(k) = \sum_{\ell=1}^{L} C_1(i,k,j,\ell)\, p_{jm}(\ell), \quad \begin{matrix} j = i-1, i+1 \\ i = 1,\ldots,N_m \\ k = 1,\ldots,L \end{matrix} \qquad (3\text{-}3)$$

$$Q_{im}^{(1)}(k) = \frac{1}{2}(Q_{i\ i-1,m}(k) + Q_{i\ i+1,m}(k)) \qquad (3\text{-}4)$$

$$Q_{im}^{(2)}(k) = \sum_{\ell_1,\ell_2=1}^{L} C_2(i,k,i-1,\ell_1,i+1,\ell_2)\, p_{i-1\ m}(\ell_1)\, p_{i+1\ m}(\ell_2) \qquad (3\text{-}5)$$

n denotes the first or second stage of the hierarchy and C_1 and C_2 are the compatibility functions at the first and second stage respectively. Let $\vec{v}_m$ be the vector of $R^P = R^L x \ldots x R^L$ ($P = N_m L$) equal to $(\vec{p}_{1m}, \vec{p}_{2m}, \ldots, \vec{p}_{N_m})$. Then (3-1) can be written as

$$J_m^{(n)} = \sum_{i=1}^{N_m} J_{im}^{(n)}(\vec{v}_m) \qquad (3\text{-}6)$$

where

$$J_{im}^{(n)}(\vec{v}_m) = \vec{p}_{im} \cdot \vec{q}_{im}^{(n)}, \qquad n = 1,2$$

Now the total criterion of consistency and ambiguity for all the M models can be written as,

$$F(\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_M) = \sum_{m=1}^{M} \sum_{i=1}^{N_m} J_{im}^{(n)}(\vec{v}_m), \quad n = 1,2 \qquad (3\text{-}7)$$

where N_m is the number of segments and $\vec{v}_m$ is the P_m ($= N_m L$) dimensional probability vector associated with the

mth model X_m.

The occlusion condition that any segment of different models should not be assigned to the same segment of the object can be written as,

$$G(\vec{v}_1,\vec{v}_2,\ldots,\vec{v}_M) = \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} g(\vec{s}_i,\vec{s}_j) = 0 \qquad (3\text{-}8)$$

where $\vec{s}_\ell$ is obtained from $\vec{v}_\ell$ with the elements corresponding to the nil class set equal to zero for all the units of the model X_ℓ and $g(\vec{s}_i,\vec{s}_j)$ is the inner product of the vectors $\vec{s}_i$ and $\vec{s}_j$. What this condition essentially means is that if a unit i of a model X_m matches to the class O_ℓ (where $\ell \neq L$), then sum of the inner product of the probability vector of this unit $\vec{p}_{im}$ with the probability vectors of all the units of all the other models should be zero. The nil class components have been excluded in obtaining $\vec{s}_\ell$ from $\vec{v}_\ell$ because one or more segments of different models may match to the nil class. Let us consider an example, when there are two models, i.e., M = 2, and the number of segments in the model X_1 and X_2 be 3 and 4 respectively, then

$$G(\vec{v}_1,\vec{v}_2) = g(\vec{s}_1,\vec{s}_2) = 0 \qquad (3\text{-}9)$$

and

$$g(\vec{s}_1,\vec{s}_2) = (\sum_{i=1}^{3} \vec{s}_{i1})\cdot(\sum_{i=1}^{4} \vec{s}_{i2})$$

$$= (\vec{s}_{11}+\vec{s}_{21}+\vec{s}_{31})\cdot(\vec{s}_{12}+\vec{s}_{22}+\vec{s}_{32}+\vec{s}_{42})$$

$$= \vec{s}_{11}\cdot(\vec{s}_{12}+\vec{s}_{22}+\vec{s}_{32}+\vec{s}_{42})+\vec{s}_{21}\cdot(\vec{s}_{12}+\vec{s}_{22}+ \quad (3\text{-}10)$$

$$\vec{s}_{32}+\vec{s}_{42})+\vec{s}_{31}\cdot(\vec{s}_{12}+\vec{s}_{22}+\vec{s}_{32}+\vec{s}_{42})$$

where

$$\vec{s}_{im} = [p_{im}(1),p_{im}(2),\ldots,p_{im}(L-1),0]^T \quad (3\text{-}11)$$

Now the occlusion problem viewed as a segment matching problem can be stated as follows.

Problem Statement (A)

Given an initial labeling $\vec{v}_1^{(0)}, \vec{v}_2^{(0)},\ldots,\vec{v}_M^{(0)}$ for the set of M models $(X_1,X_2,\ldots,X_M)$ to belong to various segments of the object, find the labeling $\vec{u}_1, \vec{u}_2,\ldots,\vec{u}_M$ that corresponds to the local maximum of the criterion (3-7) which is closest to $\vec{v}_1^{(0)}, \vec{v}_2^{(0)},\ldots,\vec{v}_M^{(0)}$ subject to the following constraints.

(a) If $\vec{u}_m = (\vec{p}_{1m},\vec{p}_{2m},\ldots,\vec{p}_{N_m m})$ then $\vec{p}_{\ell m}$ is a probability vector for $\ell = 1,2,\ldots,N_m$ and $m = 1,2,\ldots,M$. For a particular unit y of the model X_m, this means that the sum of the components of the vector $\vec{p}_{ym}$ is equal to unity and that each component be greater than or equal to zero, i.e., if

$$\vec{p}_{ym} = [p_{ym}(1), p_{ym}(2), \ldots, p_{ym}(L)]^T \quad .$$

then

$$\sum_{\ell=1}^{L} p_{ym}(\ell) = 1$$

and

$$p_{ym}(\ell) \geq 0 \qquad \text{for } \ell = 1, \ldots, L$$

(b) $G(\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_M)$ as defined by (3-8) be equal to zero.

Note that the criterion (3-7) is nonlinear. Constraint (a) involves linear equality and nonnegativity restriction, and constraint (b) is nonlinear. In order to solve this optimization problem we use the penalty function concept [3-15 to 3-17] and extend the hierarchical shape matching technique of Chapter 2.

3.3 Occlusion Algorithm

Coordinated Hierarchical Stochastic Labeling Technique Based on Gradient Projection Method and Penalty Function Approach

In order to solve the problem (A) using the penalty function approach, we define the penalized objective function [3-15 to 3-17] as,

$$\psi_c(\vec{v}_1,\vec{v}_2,\ldots,\vec{v}_M) = F(\vec{v}_1,\vec{v}_2,\ldots,\vec{v}_M) + \sum_{i=1}^{M-1}\sum_{j=i+1}^{M} d_{ij}\phi_{ij}[g(\vec{s}_i,\vec{s}_j)] \qquad (3\text{-}12)$$

where ϕ_{ij} is a penalty function and $\{d_{ij}\}$ are penalty constants, also referred as the coordination factors. Since the constraint (b) given by (3-8) is an equality constraint, the penalty function is taken as the simple quadratic loss function given by

$$\phi_{ij}(a) \triangleq -a^2 \qquad (3\text{-}13)$$

Now the problem (A) described in the earlier section becomes equivalent to that of maximizing (3-12) subject to the constraints (a). It can be solved by using the gradient projection method applied to the linear constraints. We have used this method in the hierarchical shape matching algorithm presented in the last chapter. So here we extend this algorithm with respect to the penalized objective function. The maximization of (3-12) subject to the constraints (a) is equivalent to maximizing

$$\begin{cases} \max\limits_{\vec{v}_1} & F(\vec{v}_1) + S(\vec{v}_1,\ldots,\vec{v}_M) \\ \max\limits_{\vec{v}_1} & F(\vec{v}_2) + S(\vec{v}_1,\ldots,\vec{v}_M) \\ \vdots & \\ \max\limits_{\vec{v}_M} & F(\vec{v}_M) + S(\vec{v}_1,\ldots,\vec{v}_M) \end{cases} \qquad (3\text{-}14)$$

where $S(\vec{v}_1,\ldots,\vec{v}_M)$ corresponds to the second term of

(3-12). Thus in effect there is a hierarchical process for every model participating in the occlusion. These processes are executed in parallel and coordinated in such a way that the occlusion condition is satisfied. The algorithm has been implemented in a serial fashion on the computer, first we maximize with respect to $\vec{v}_1$, then with respect to $\vec{v}_2$ and so on. The main modification of the shape matching algorithm presented in the last chapter with respect to the occlusion problem is the computation of the gradient. In the last chapter we maximized $F(\vec{v})$ with respect to $\vec{v}$, but now the contribution to the gradient also comes from the second terms in (3-14), for example,

New gradient with respect to $\vec{v}_1$ = Old gradient of $F(\vec{v}_1)$ with respect to $\vec{v}_1 + \frac{\partial}{\partial \vec{v}_1}[S(\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_M)]$

As an example let M = 2, then the problem becomes to maximize

$$\psi_c(\vec{v}_1, \vec{v}_2) = F(\vec{v}_1, \vec{v}_2) - d[g(\vec{s}_1, \vec{s}_2)]^2$$

subject to the constraints (a). d_{12} is denoted as d in the above equation. In order to solve this we consider

$$\begin{cases} \max\limits_{\vec{v}_1} F(\vec{v}_1) - d[g(\vec{s}_1, \vec{s}_2)]^2 \\ \\ \max\limits_{\vec{v}_2} F(\vec{v}_2) - d[g(\vec{s}_1, \vec{s}_2)]^2 \end{cases} \qquad (3\text{-}15)$$

and use the gradient projection method as in the last chapter with the modification for the gradient term. Similarly, when M = 3, we solve the following problem,

$$\begin{cases} \max\limits_{\vec{v}_1} F(\vec{v}_1) - A \\ \max\limits_{\vec{v}_2} F(\vec{v}_2) - A \\ \max\limits_{\vec{v}_3} F(\vec{v}_3) - A \end{cases} \tag{3-16}$$

where,

$$A = d_{12}[g(\vec{s}_1,\vec{s}_2)]^2+d_{13}[g(\vec{s}_1,\vec{s}_3)]^2+d_{23}[g(\vec{s}_2,\vec{s}_3)]^2 \tag{3-17}$$

In general to solve (3-14) by maximizing with respect to $\vec{v}_i$ the algorithm can be stated as follows:

The Occlusion Algorithm

1. Pick an initial estimate of $(\vec{v}_1^{(0)}, \vec{v}_2^{(0)},\ldots,\vec{v}_M^{(0)})$. This is the initial assignment of probabilities to the units of the models.

2. Pick the penalty constant $\{d_{ij}\}$ so that it provides a suitable balance between the associated first and second terms of (3-14). This is done automatically and will be described in the next section. Penalty constants affect the convergence rate of the algorithm.

3. Determine the maximum $\vec{v}_m^{(n+1)}$ $(m = 1,2,\ldots,M)$ of the unconstrained penalized objective function (3-14) subject to the constraints (a) by using the value at the present iteration $\vec{v}_m^{(n)}$ and gradient projection method (described in detail in the last chapter).

4. Pick new penalty constants $\{d_{ij}\}$ in order to modify (or rebalance) the magnitude of the penalty terms. The magnitude of the penalties should be increased to force a closer approach to the boundary; replace n by n+1 and return to 3.

Under the assumption of the continuity of function F (3-7) and constraints (3-8) inherent in (3-12), the sequence of maxima $\{\vec{v}_m^{(n+1)}\}$ for $m = 1,\ldots,M$ generated by the above algorithm approaches a constrained maximum of the problem defined in (A). The iteration is terminated after the convergence is considered as adequate. Some of the details of numerical strategies have been described in the last chapter and others will be discussed in the following section. The rate of convergence and the possible ill-conditioning near the boundary are discussed in [3-15 to 3-17]. However, in our case since we are seeking only local maxima, ill-conditioning problems do not occur.

In the next section we present synthetic and real examples from the biological and industrial domains where two or three models occlude one another to form an object. We shall discuss the computational properties of the algorithm. Fig. 3.1 shows the block diagram of the algorithm when two models occlude each other.

3.4 Examples and Comments

Since the shape matching algorithm for occluded objects uses the hierarchical stochastic labeling technique presented in the last chapter, the discussion presented in section 2.4 is also applicable here. In addition to the parameters discussed in section 2.4, now we also have a new parameter, called the penalty constant or coordination factor. In the examples presented in this chapter, we have taken penalty constants associated with various terms in the occlusion condition to be the same e.g., in (3-17) $d_{12} = d_{13} = d_{23}$ is taken. We determine its value at every iteration such that the penalty term (second term) of (3-14) is a fixed percentage called PERCEN of the first term in (3-14). Since the criteria increase at every iteration, the penalty constant also increases and when the occlusion condition is satisfied, the penalty constant effectively becomes infinite. Normally we have chosen PERCEN to be between 10 to 90%

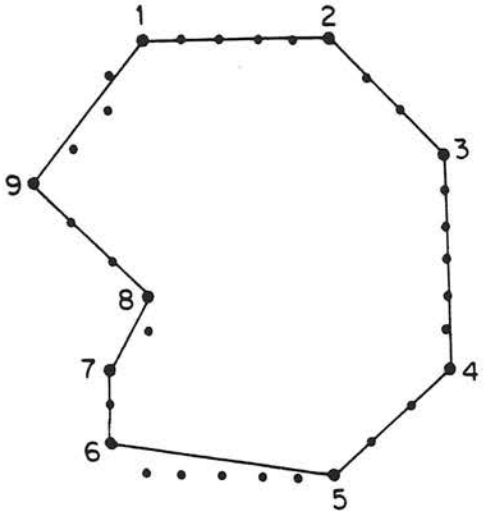
depending upon the complexity of the matching task. A higher value of PERCEN requires more computation time.

Examples

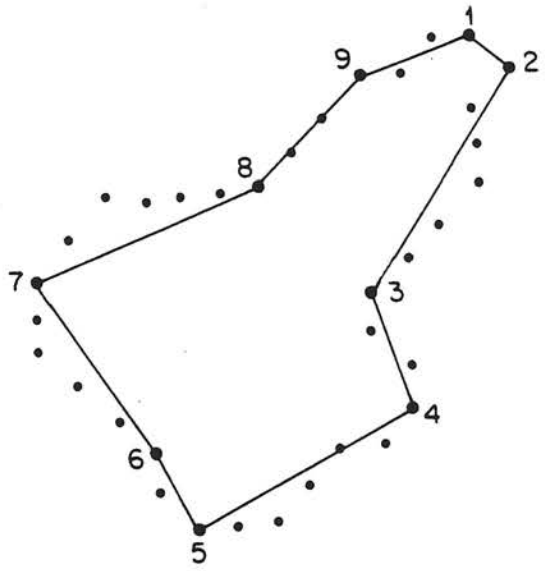
In the last chapter we presented industrial examples where only a partial view of a model was visible within the object. In the examples presented in the following we know a priori all the models which are occluding one another and we want to identify all of them based on their partial views. Now we shall present synthetic, industrial and biological examples.

Example 3.1

Fig. 3.2 (same as fig. 2.6) shows two models X_1 and X_2 which occlude each other to form an apparent object. Note that there is a change of scale for the model X_1 in the apparent object; it may be because of segmentation (different lighting conditions) or changes in the model itself even if the camera position is fixed. The dotted points show the boundary of the models and object. The polygonal approximation is obtained with a smoothing factor of 4. The expected assignments of the units of models X_1 and X_2 are shown in Table 3.1. The results of labeling of the units of the models using the occlusion algorithm are shown in Tables 3.2 to 3.5. In these tables



(a)

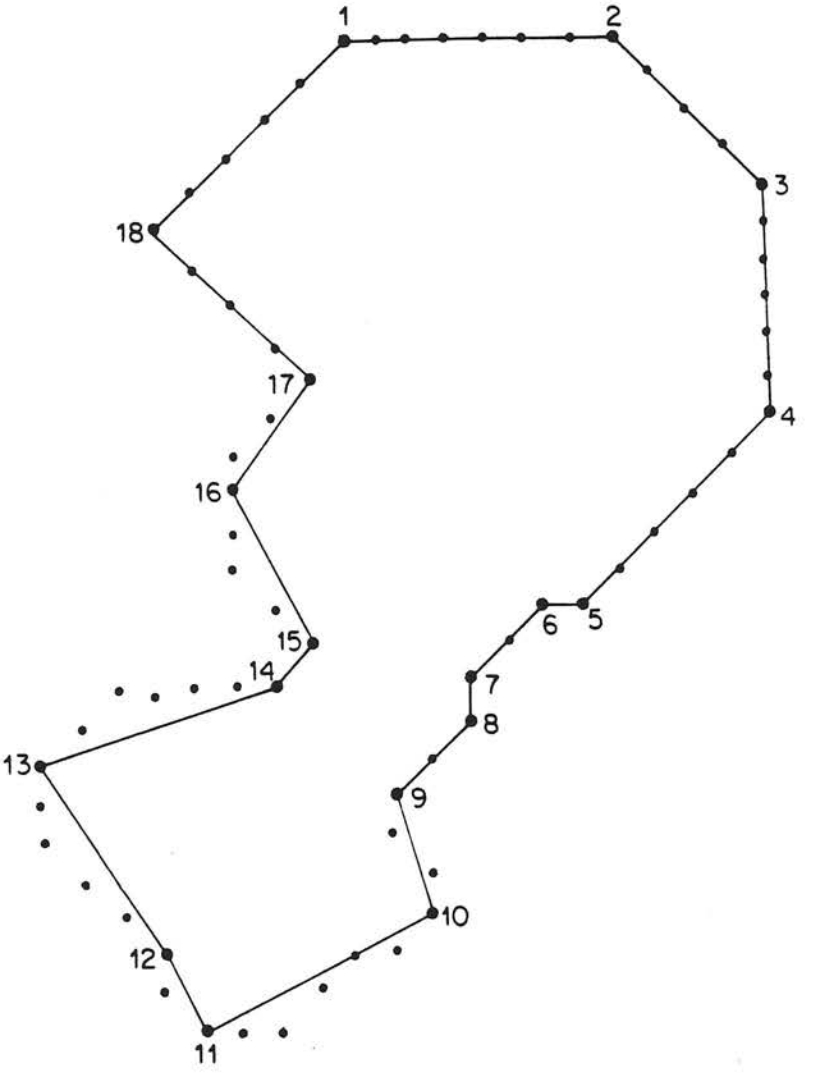


(b)

Fig. 3.2 (a) Model X_1, Perimeter = 34, Number of segments = 9

(b) Model X_2, Perimeter = 35, Number of segments = 9

(c) Apparent object, Perimeter = 67, Number of segments = 18



(c)

Fig. 3.2 (CONTINUED)

Table 3.1

Expected assignments of the units of the Models X_1 and X_2, Example 3.1

Model X_1		Model X_2	
Unit	Label	Unit	Label
1	1	1	19
2	2	2	8 or 19
3	3	3	9
4	4	4	10
5	5 or 19	5	11
6	15 or 19	6	12
7	16	7	13
8	17	8	14 or 19
9	18	9	19

Table 3.2

Assignment of the units of Model X_1, Example 3.1

Smoothing factor = 4, $\alpha = 0.99$, $p_i(\text{nil}) = 0.15$, ITER1 = 3, ITER2 = 10, UPTHLD = 0.8, LFACT = 20, No. of neighboring labels = 1, Inverse weights = (3, 2, 5, 2)
PERCEN = 10%

Units of Model X_1	Labels at different iterations						
	First stage			Second stage			
	0	1	3	1	4	7	10
1	1(.20)	1(.29)	1(.36)	1(.39)	19(.63)	19(1.0)	19(1.0)
2	2(.51)	2(.62)	2(1.0)	2(1.0)	2(1.0)	2(1.0)	2(1.0)
3	3(.63)	3(.74)	3(1.0)	3(1.0)	3(1.0)	3(1.0)	3(1.0)
4	4(.28)	4(.42)	4(.58)	4(.69)	4(1.0)	4(1.0)	4(1.0)
5	19(.15)	19(.24)	19(.27)	19(.32)	19(.52)	5(1.0)	5(1.0)
6	19(.15)	19(.23)	19(.31)	19(.34)	19(.54)	19(1.0)	19(1.0)
7	19(.15)	19(.21)	19(.24)	19(.26)	16(.36)	16(.72)	16(1.0)
8	17(.30)	17(.39)	17(1.0)	17(1.0)	17(1.0)	17(1.0)	17(1.0)
9	18(.29)	18(.43)	18(.56)	18(.67)	18(1.0)	18(1.0)	18(1.0)
First term of the objective function	-	1.037	1.536	1.382	1.558	1.569	1.610
Penalty term of the objective function	-	.1037	.1536	.1382	.1558	0	0
Criterion	-	.933	1.383	1.244	1.403	1.569	1.610
Penalty Constant	-	.0112	.1128	.2055	17.16	-	-

Table 3.3

Assignment of the units of Model X_2, Example 3.1.
Parameters same as in Table 3.2

Units of Model X_2	Labels at different iterations: First Stage 0	First Stage 1	First Stage 3	Second Stage 1	Second Stage 4	Second Stage 7	Second Stage 10
1	19(.15)	19(.26)	19(.31)	19(.35)	19(.43)	9(1.0)	9(1.0)
2	10(.15)	19(.18)	19(.21)	19(.23)	19(.26)	19(.40)	19(1.0)
3	9(.24)	9(.44)	9(.67)	9(1.0)	9(1.0)	9(1.0)	9(1.0)
4	10(.67)	10(1.0)	10(1.0)	10(1.0)	10(1.0)	10(1.0)	10(1.0)
5	11(.66)	11(1.0)	11(1.0)	11(1.0)	11(1.0)	11(1.0)	11(1.0)
6	12(.65)	12(1.0)	12(1.0)	12(1.0)	12(1.0)	12(1.0)	12(1.0)
7	13(.66)	13(1.0)	13(1.0)	13(1.0)	13(1.0)	13(1.0)	13(1.0)
8	19(.15)	19(.19)	19(.22)	19(.23)	14(.35)	14(1.0)	14(1.0)
9	1(.16)	19(.22)	19(.28)	19(.29)	19(.37)	19(.59)	19(1.0)
First term of the objective function	-	1.812	2.764	2.541	2.803	2.896	3.239
Penalty term of objective function	-	.1812	.2764	.2541	.2803	0	0
Criterion	-	1.631	2.487	2.287	2.523	2.896	3.239
Penalty Constant	-	.0196	.2030	.3779	30.86	-	-

Table 3.4

Assignment of the units of Model X_1, Example 3.1
Parameters same as in Table 3.2 except ITER2 = 11
and PERCEN = 90%

Units of Model X_1	Labels at different iterations: First Stage 0	First Stage 1	First Stage 3	Second Stage 1	Second Stage 4	Second Stage 7	Second Stage 10	Second Stage 11
1	1(.20)	1(.22)	1(.23)	1(.24)	1(.34)	1(.56)	1(1.0)	1(1.0)
2	2(.51)	2(.62)	2(.68)	2(1.0)	2(1.0)	2(1.0)	2(1.0)	2(1.0)
3	3(.63)	3(.72)	3(.75)	3(1.0)	3(1.0)	3(1.0)	3(1.0)	3(1.0)
4	4(.28)	4(.32)	4(.33)	4(.37)	4(.56)	4(1.0)	4(1.0)	4(1.0)
5	19(.15)	19(.19)	19(.20)	19(.21)	19(.23)	19(.33)	5(.43)	5(1.0)
6	19(.15)	19(.19)	19(.20)	19(.23)	19(.32)	19(.50)	15(.77)	15(1.0)
7	19(.15)	19(.18)	19(.19)	19(.20)	19(.27)	16(.47)	16(1.0)	16(1.0)
8	17(.30)	17(.34)	17(.38)	17(.46)	17(1.0)	17(1.0)	17(1.0)	17(1.0)
9	18(.29)	18(.31)	18(.33)	18(.37)	18(.58)	18(1.0)	18(1.0)	18(1.0)
First term of the objective function	-	1.037	1.252	1.095	1.243	1.352	1.516	1.323
Penalty terms of objective function	-	.9334	1.127	.9862	1.119	0	0	0
Criterion	-	.1037	.1252	.1095	.1243	1.352	1.516	1.323
Penalty Constant	-	.1009	1.071	2.758	16738.4	-	-	-

Table 3.5

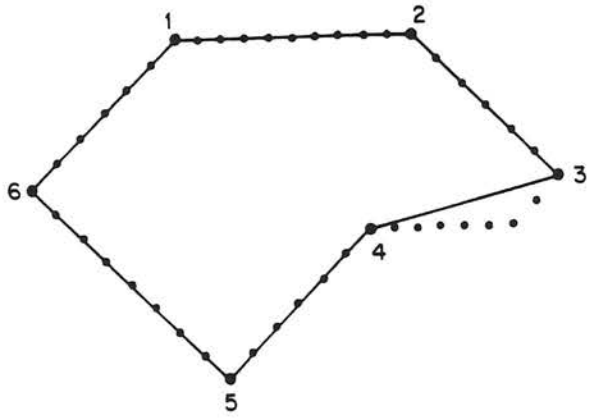
Assignment of the units of Model X_2, Example 3.1.
Parameters same as in Table 3.4

Units of Model X_2	Labels at different iterations: First Stage 0	First Stage 1	First Stage 3	Second Stage 1	Second Stage 4	Second Stage 7	Second Stage 10	Second Stage 11
1	19(.15)	19(.20)	19(.22)	19(.25)	19(.28)	19(.45)	19(.71)	19(1.0)
2	10(.15)	19(.17)	19(.20)	19(.21)	19(.25)	10(.39)	7(1.0)	7(1.0)
3	9(.24)	9(.29)	9(.31)	9(.34)	9(.36)	9(1.0)	9(1.0)	9(1.0)
4	10(.67)	10(1.0)	10(1.0)	10(1.0)	10(1.0)	10(1.0)	10(1.0)	10(1.0)
5	11(.66)	11(1.0)	11(1.0)	11(1.0)	11(1.0)	11(1.0)	11(1.0)	11(1.0)
6	12(.65)	12(1.0)	12(1.0)	12(1.0)	12(1.0)	12(1.0)	12(1.0)	12(1.0)
7	13(.66)	13(1.0)	13(1.0)	13(1.0)	13(1.0)	13(1.0)	13(1.0)	13(1.0)
8	19(.15)	19(.17)	19(.20)	19(.25)	19(.27)	14(.79)	14(1.0)	14(1.0)
9	1(.16)	19(.22)	19(.24)	19(.29)	19(.31)	19(.71)	19(1.0)	19(1.0)
First term of the objective function	-	1.812	2.581	2.312	2.388	1.974	2.202	2.212
Penalty term of objective function	-	1.631	2.323	2.081	2.149	0	0	0
Criterion	-	.1812	.2581	.2312	.2388	1.974	2.202	2.212
Penalty Constant	-	.1764	2.207	5.820	32158.9	-	-	-

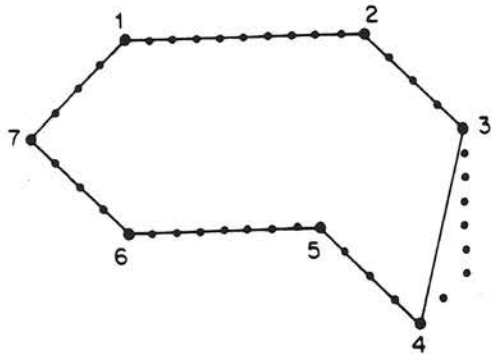
we have used the same parameters as in Table 2.7. The value of PERCEN in Tables 3.2 and 3.3 is 10% and in Tables 3.4 and 3.5 is 90%. The assignment of the units of both models are correct and the conflicting labeling of the two units which occurred in Tables 2.7 and 2.8 does not occur (unit 1 of both X_1 and X_2 are assigned to the label 1 in the previous tables). The total computation time for the results shown in Tables 3.2 and 3.3 is 200 seconds and that of Tables 3.4 and 3.5 is 235 seconds. Increasing the value of PERCEN i.e., increasing the value of the penalty term, requires more iterations and thus the computation time will be relatively more. In these tables we have shown values of the first and penalty function terms (first and second terms of (3.12)), criteria and penalty constants at various iterations. When the penalty term becomes 0, the nonlinear constraint (3-9) is satisfied and the penalty constant becomes effectively infinite. Note that with the iteration, the criteria and penalty constants increase.

Example 3.2

Figure 3.3 presents another synthetic example, where three models X_1, X_2 and X_3 occlude one another to form an apparent object. We want to identify each of the models within the apparent object. The problem is a kind of

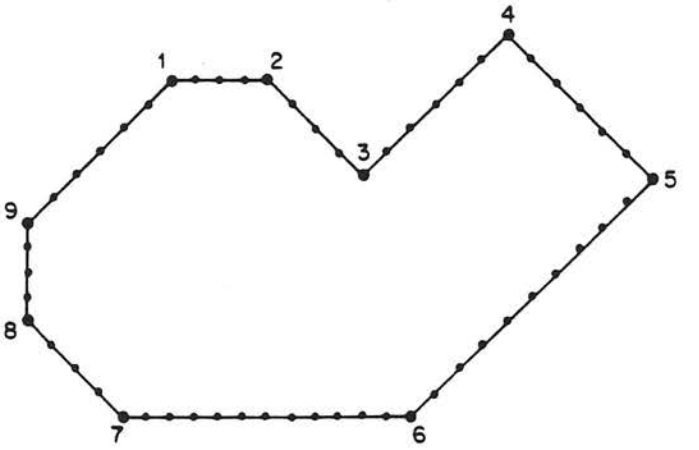


(a)

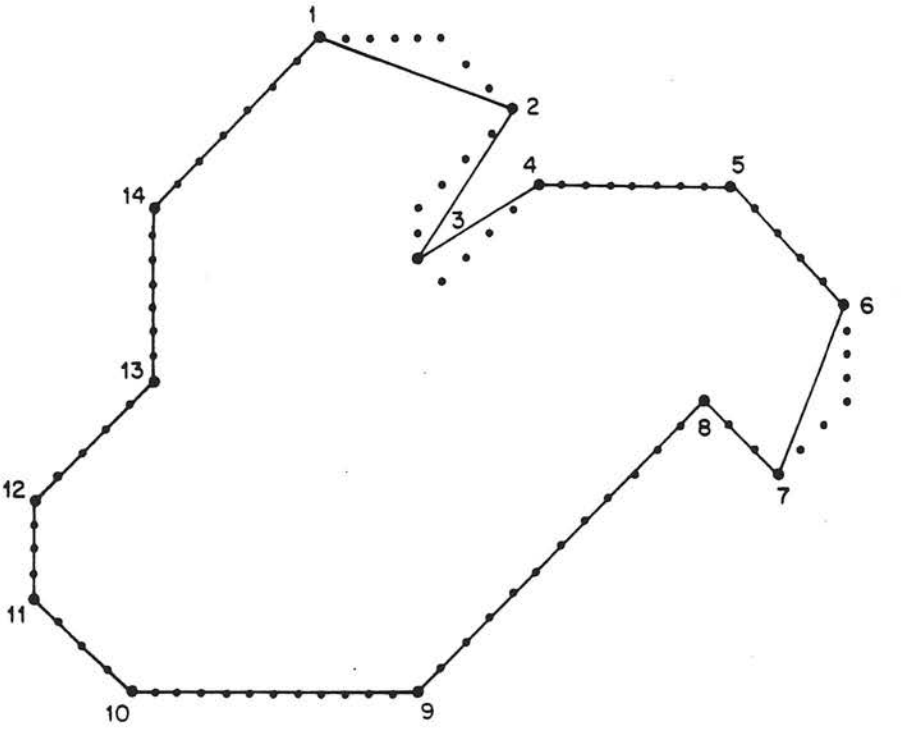


(b)

Fig. 3.3 (a) Model X_1, Number of segments = 6
(b) Model X_2, Number of segments = 7
(c) Model X_3, Number of segments = 9
(d) Apparent object, Number of segments = 14



(c)



(d)

Fig. 3.3 (CONTINUED)

"jig-saw puzzle". Expected assignments of the units of the models are shown in Table 3.6. Table 3.7 shows the results of labeling when individual models X_1, X_2 and X_3 are matched to the apparent object without using the occlusion algorithm i.e., just using the hierarchical stochastic labeling technique of chapter 2 . Although these results are good, an examination of this table shows that the segment 5 of the apparent object is assigned to the units of models X_2 and X_3, thus there is degree of confusion in the labeling. The labeling results using the occlusion algorithm are shown in Tables 3-8 to 3-10. All the labels of all the units of X_1, X_2 and X_3 are correct except the label of the unit 5 of the model X_1. This is because of the similarity of the local structure. The total computation time for this example (starting from polygonal approximation, computation of features and so on) is 152.7 seconds.

Example 3.3

Figure 3.4 shows gray scale images of industrial parts (fig. 3.4(a) and (b)) which occlude each other to form an occluded object shown in fig. 3.4(c). The images shown in fig. 3.4 are of size 512x512, 8 bits. The images in figs. 3.4(a) and (b) are reduced by 16 times and the image in fig. 3.4(c) by 18 times. To get the boundary of

Table 3.6

Expected assignments of the units of Models X_1, X_2 and X_3, Example 3.2

Units of a model	Labels of Models X_1	X_2	X_3
1	14 or 15	4	15
2	1	5	15
3	15	6	15
4	15	15	15
5	15	15	15
6	15	15	9
7	-	3	10
8	-	-	11
9	-	-	12

Table 3.7

Results of labeling of the Models X_1, X_2 and X_3 when they matched to the apparent object by themselves (i.e., no coordination). Example 3.2

Smoothing factor = 3, $\alpha = 0.99$, $p_i(nil) = 0.15$, ITER1 = 3, ITER2 = 6, UPTHLD = 0.8, LFACT = 20, No. of neighboring labels = 1, Inverse weights = (3, 2, 5, 2)

Units of a Model	Labels of X_1		Labels of X_2		Labels of X_3	
	Initial Labeling	Final Labeling	Initial Labeling	Final Labeling	Initial Labeling	Final Labeling
1	4(.17)	15(1.0)	4(.17)	4(1.0)	4(.16)	15(1.0)
2	1(.20)	1(1.0)	5(.31)	5(1.0)	5(.21)	5(1.0)
3	15(.15)	15(1.0)	6(.40)	6(1.0)	13(.17)	15(1.0)
4	15(.15)	15(1.0)	15(.15)	15(1.0)	1(.19)	15(1.0)
5	15(.15)	7(1.0)	13(.20)	13(1.0)	2(.22)	8(1.0)
6	15(.15)	15(1.0)	15(.15)	15(1.0)	9(.66)	9(1.0)
7	-	-	15(.15)	15(1.0)	10(.63)	10(1.0)
8	-	-	-	-	11(.63)	11(1.0)
9	-	-	-	-	14(.16)	12(1.0) .

Table 3.8

Results of labeling for the Model X_1 when Models X_1, X_2 and X_3 are matched to the apparent object using the Occlusion Algorithm. Example 3.2. Parameters same as in Table 3.7, PERCEN = 30%

Units of Model X_1	Labels at different iterations					
	First Stage			Second Stage		
	0	1	3	1	4	6
1	4(.17)	15(.23)	15(.34)	15(.37)	15(.68)	15(1.0)
2	1(.20)	1(.24)	1(.30)	1(.34)	1(.71)	1(1.0)
3	15(.15)	15(.33)	15(.47)	15(.54)	15(1.0)	15(1.0)
4	15(.15)	15(.30)	15(.36)	15(.41)	15(1.0)	15(1.0)
5	15(.15)	15(.23)	15(.27)	15(.33)	7(.60)	7(1.0)
6	15(.15)	15(.29)	15(.35)	15(.40)	15(1.0)	15(1.0)
First term of the objective function	-	.5183	.9785	1.074	1.6071	2.437
Penalty term of the objective function	-	.1554	.2935	.3223	.4821	0
Criterion	-	.3628	.6849	.7522	1.125	2.437
Penalty Constant	-	.00624	.0427	.0758	1.131	-

Table 3.9

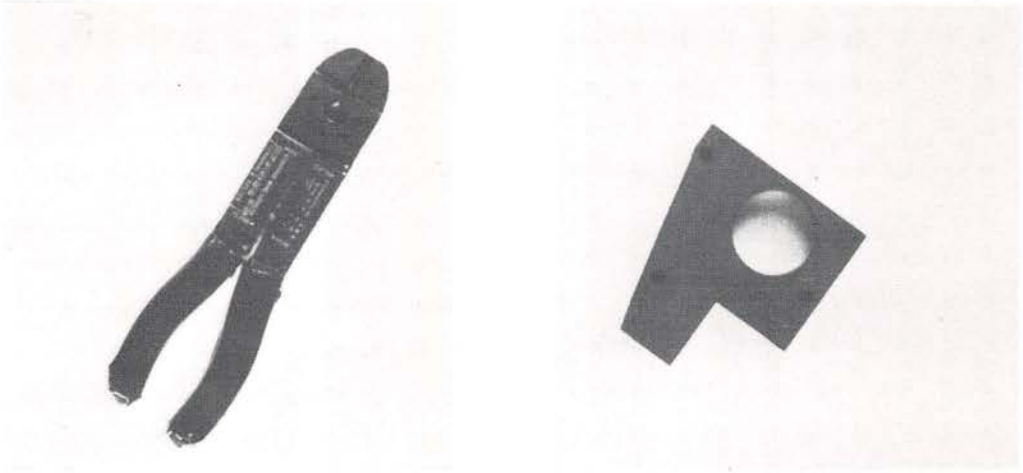
Results of labeling for the Model X_2 when Models X_1, X_2 and X_3 are matched to the apparent object using the Occlusion Algorithm. Example 3.2. Parameters same as in Table 3.7. PERCEN = 30%

Units of Model X_2	Labels at different iterations					
	First Stage			Second Stage		
	0	1	3	1	4	6
1	4(.17)	4(.33)	4(.42)	4(.48)	4(1.0)	4(1.0)
2	5(.31)	5(.53)	5(.73)	5(1.0)	5(1.0)	5(1.0)
3	6(.40)	6(.62)	6(.77)	6(1.0)	6(1.0)	6(1.0)
4	15(.15)	7(.40)	7(.55)	7(.56)	15(1.0)	15(1.0)
5	13(.20)	13(.30)	13(.38)	13(.39)	15(.68)	15(1.0)
6	15(.15)	15(.50)	15(.71)	15(1.0)	15(1.0)	15(1.0)
7	15(.15)	15(.20)	15(.33)	15(.44)	15(.61)	3(1.0)
First term of the objective function	-	.8197	1.5709	1.3875	2.7729	3.451
Penalty term of the objective function	-	.2459	.4712	.4162	.8318	0
Criterion	-	.5738	1.0996	.9712	1.9410	3.451
Penalty Constant	-	.009875	.0685	.0979	1.953	-

Table 3.10

Results of labeling for the Model X_3 when Models X_1, X_2 and X_3 are matched to the apparent object using the Occlusion Algorithm. Example 3.2. Parameters same as in Table 3.7. PERCEN = 30%

Units of Model X_3	Labels at different iterations					
	First Stage			Second Stage		
	0	1	3	1	4	6
1	4(.16)	15(.26)	15(.32)	15(.39)	15(.52)	15(1.0)
2	5(.21)	15(.33)	15(.46)	15(.52)	15(1.0)	15(1.0)
3	13(.17)	15(.33)	15(.49)	15(.53)	15(.59)	13(1.0)
4	1(.19)	15(.35)	15(.44)	15(.47)	15(.56)	15(1.0)
5	2(.22)	15(.28)	15(.37)	15(.41)	15(.61)	15(1.0)
6	9(.66)	9(.77)	9(1.0)	9(1.0)	9(1.0)	9(1.0)
7	10(.63)	10(.73)	10(1.0)	10(1.0)	10(1.0)	10(1.0)
8	11(.63)	11(.76)	11(1.0)	11(1.0)	11(1.0)	11(1.0)
9	14(.16)	15(.20)	15(.25)	15(.29)	12(.40)	12(1.0)
First term of the objective function	-	1.721	3.031	3.079	4.100	3.692
Penalty term of the objective function	-	.5165	.9095	.9237	1.230	0
Criterion	-	1.205	2.122	2.155	2.870	3.692
Penalty Constant	-	.0207	.1323	.2172	2.887	-



(a) (b)

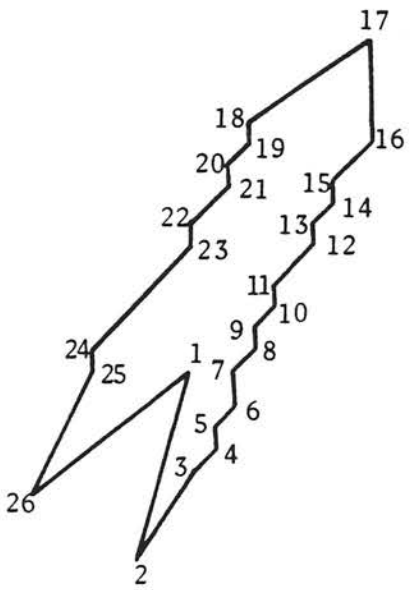
(c)

Fig. 3.4 (a) An industrial piece
(b) An industrial piece
(c) Partial occlusion of industrial pieces in (a) and (b)

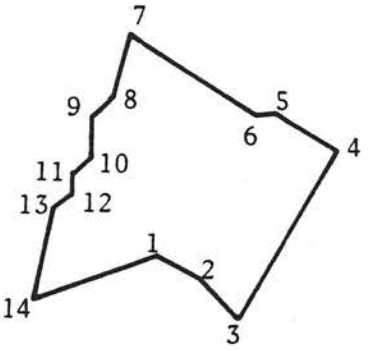
these objects, the thresholds are interactively selected. They are 180, 170 and 140 for the images of fig. 3.14(a), (b) and (c) respectively. The polygonal approximation of the reduced images is shown in fig. 3.5. A smoothing factor of 6 is used for all the three reduced images. Only the rotation and scale invariant features (interior angle and exangle) are used in the initial probability assignment. We prefer that a label be assigned to the nil class rather than to an incorrect class. The results of the labeling are shown in Table 3.11 without the use of the occlusion algorithm. Label 25 is the nil class. The results of the occlusion algorithm for the labeling of models X_1 and X_2 are shown in Tables 3.12 and 3.13. Note that all the key assignments of the units are correct. The total computation time for this example is 530 seconds.

Example 3.4

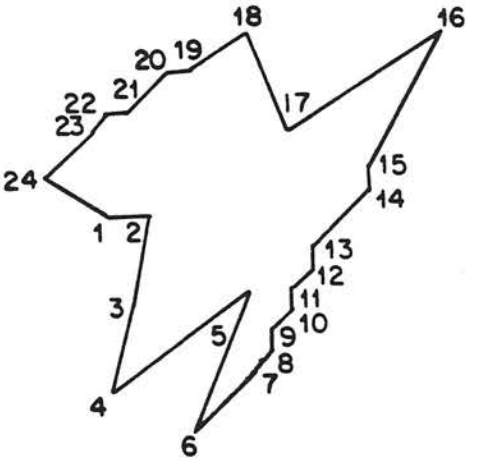
In this example we use the industrial pieces of figs. 3.4(a) and 3.4(b) used in the last example and also the small object shown in fig. 3.6(a). These three objects occlude as shown in fig. 3.6(b). These images are 512x512, 8 bits. The thresholds and the reducing factor for the images of figs. 3.4(a) and (b) are the same as in the last example. Since the object of fig. 3.6(a) is very



(a)



(b)



(c)

Fig. 3.5 (a) Model X_1, Number of segments = 26
(b) Model X_2, Number of segments = 14
(c) Apparent object, Number of segments = 24

Table 3.11

Results of labeling of the Models X_1 and X_2 when they are matched to the apparent object by themselves (i.e., no coordination). Example 3.3

Smoothing factor = 6, $\alpha = 0.99$, $p_i(nil) = 0.20$, ITER1 = 3, ITER2 = 7, UPTHLD = 0.8, LFACT = 20, No. of neighboring labels = 1, Inverse weights = (4, 3, -, -)

	Labels of X_1		Labels of X_2	
Units of a Model	Initial Labeling	Final Labeling	Initial Labeling	Final Labeling
1	5(.25)	5(1.0)	25(.20)	25(1.0)
2	6(.26)	6(1.0)	25(.20)	23(1.0)
3	25(.20)	7(1.0)	25(.20)	24(1.0)
4	25(.20)	8(1.0)	25(.20)	18(1.0)
5	25(.20)	25(1.0)	25(.20)	25(1.0)
6	25(.20)	25(1.0)	25(.20)	25(1.0)
7	25(.20)	25(1.0)	25(.20)	25(1.0)
8	25(.20)	25(1.0)	25(.20)	13(1.0)
9	25(.20)	25(1.0)	25(.20)	25(1.0)
10	25(.20)	25(1.0)	25(.20)	25(1.0)
11	25(.20)	25(1.0)	25(.20)	25(1.0)
12	25(.20)	25(1.0)	25(.20)	19(1.0)
13	25(.20)	25(1.0)	25(.20)	14(1.0)
14	25(.20)	25(1.0)	25(.20)	18(1.0)
15	25(.20)	25(1.0)		
16	25(.20)	14(1.0)		
17	25(.20)	25(1.0)		
18	25(.20)	12(1.0)		
19	25(.20)	25(1.0)		
20	25(.20)	25(1.0)		
21	25(.20)	25(1.0)		
22	25(.20)	25(1.0)		
23	25(.20)	11(1.0)		
24	14(.20)	25(1.0)		
25	25(.20)	25(1.0)		
26	25(.20)	4(1.0)		

Table 3.12

Results of labeling for the Model X_1 when models X_1 and X_2 are matched to the apparent object using the Occlusion Algorithm. Example 3.3. Parameters same as in Table 3.11, except ITER2 = 5, PERCEN = 10%

Units of Model X_1	Labels at different iterations					
	First Stage			Second Stage		
	0	1	3	1	3	5
1	5(.25)	5(.37)	5(.62)	5(1.0)	5(1.0)	5(1.0)
2	6(.26)	6(.42)	6(.60)	6(.64)	6(1.0)	6(1.0)
3	25(.20)	25(.25)	7(.34)	25(.39)	7(.55)	7(1.0)
4	25(.20)	25(.23)	8(.59)	8(.61)	8(1.0)	8(1.0)
5	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
6	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
7	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
8	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
9	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
10	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
11	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
12	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
13	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
14	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
15	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
16	25(.20)	25(.28)	25(.32)	25(.51)	14(1.0)	14(1.0)
17	25(.20)	25(.33)	25(.51)	25(.74)	25(1.0)	25(1.0)
18	25(.20)	25(.29)	25(.41)	25(.73)	25(1.0)	25(1.0)
19	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
20	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
21	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
22	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
23	25(.20)	25(.23)	25(.80)	25(1.0)	25(1.0)	25(1.0)
24	14(.22)	14(.31)	14(.51)	14(.56)	14(1.0)	14(1.0)
25	25(.20)	25(.28)	25(.51)	25(.60)	25(1.0)	25(1.0)
26	25(.20)	25(.28)	4(.39)	25(.40)	25(1.0)	25(1.0)
First term of the objective function	-	3.350	4.715	16.21	17.04	17.20
Penalty term of the objective function	-	0.3350	.4715	1.621	0	0
Criterion	-	3.015	4.244	14.59	17.04	17.20
Penalty Constant	-	.0025	.0085	1.696	-	-

Table 3.13

Results of labeling for the Model X_2 when models X_1 and X_2 are matched to the apparent object using the Occlusion Algorithm. Example 3.3. Parameters same as in Table 3.11 except ITER2 = 5, PERCEN = 10%

	Labels at different iterations					
Units of Model X_1	First Stage			Second Stage		
	0	1	3	1	3	5
1	25(.20)	25(.31)	25(.50)	25(.51)	25(1.0)	25(1.0)
2	25(.20)	25(.34)	25(.44)	25(.47)	23(.54)	23(1.0)
3	25(.20)	25(.32)	25(.52)	25(.55)	25(.62)	25(1.0)
4	25(.20)	25(.28)	25(.38)	25(.39)	25(.44)	25(1.0)
5	25(.20)	25(.28)	25(.39)	25(.49)	25(1.0)	25(1.0)
6	25(.20)	25(.30)	25(.39)	25(.42)	25(.44)	15(1.0)
7	25(.20)	25(.29)	18(.44)	18(.49)	18(1.0)	18(1.0)
8	25(.20)	25(.33)	25(.49)	25(.57)	25(1.0)	25(1.0)
9	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
10	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
11	25(.20)	25(.23)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
12	25(.20)	25(.28)	25(.36)	25(.51)	25(.54)	19(1.0)
13	25(.20)	25(.33)	25(.64)	25(.75)	22(1.0)	22(1.0)
14	25(.20)	25(.30)	25(.43)	25(.44)	25(1.0)	25(1.0)
First term of the objective function	-	1.377	2.821	5.684	5.450	7.136
Penalty Term of the objective function	-	.1377	.2821	.5684	0	0
Criterion	-	1.239	2.539	5.116	5.450	7.136
Penalty Constant	-	.00103	.00511	.5947	-	-

(a)

(b)

Fig. 3.6 (a) An industrial piece

(b) Partial occlusion of industrial pieces in figs. 3.4(a), 3.4(b) and 3.6(a)

small, it has been reduced by a factor of 8 and the occluded object of fig. 3.6(b) by a factor of 14. The thresholds used for figs. 3.6(a) and (b) are 170 and 140 respectively. In this example we have used not only the different scaling for the objects participating in the occlusion and the occluded object but also different smoothing factors in the polygonal approximation. The polygonal approximation with a smoothing factor of 6 for the three models is shown in fig. 3.7(a), (b) and (c). A smoothing factor of 8 is used for the apparent object (fig. 3.7(d)). Only the rotation and scale invariant features (interior angle and exangle) are used for the initial probability assignment. The results of the labeling of the models without the use of occlusion algorithm are shown in Table 3.14. Label 29 is the nil class. Note that some assignments are not correct because the visible structure of the individual models is not matched intact. The results of labeling for the models when the occlusion algorithm is used are shown in Tables 3.15 to 3.17. Note that all the key assignments are correct. Model X_3 (Table 3.17) is not identified correctly because only a very small portion of it is visible in the apparent object. In fact only the segments 11 and 12 of the apparent object belong to model X_3. Ideally the correct assignment of unit 10 of the model X_3

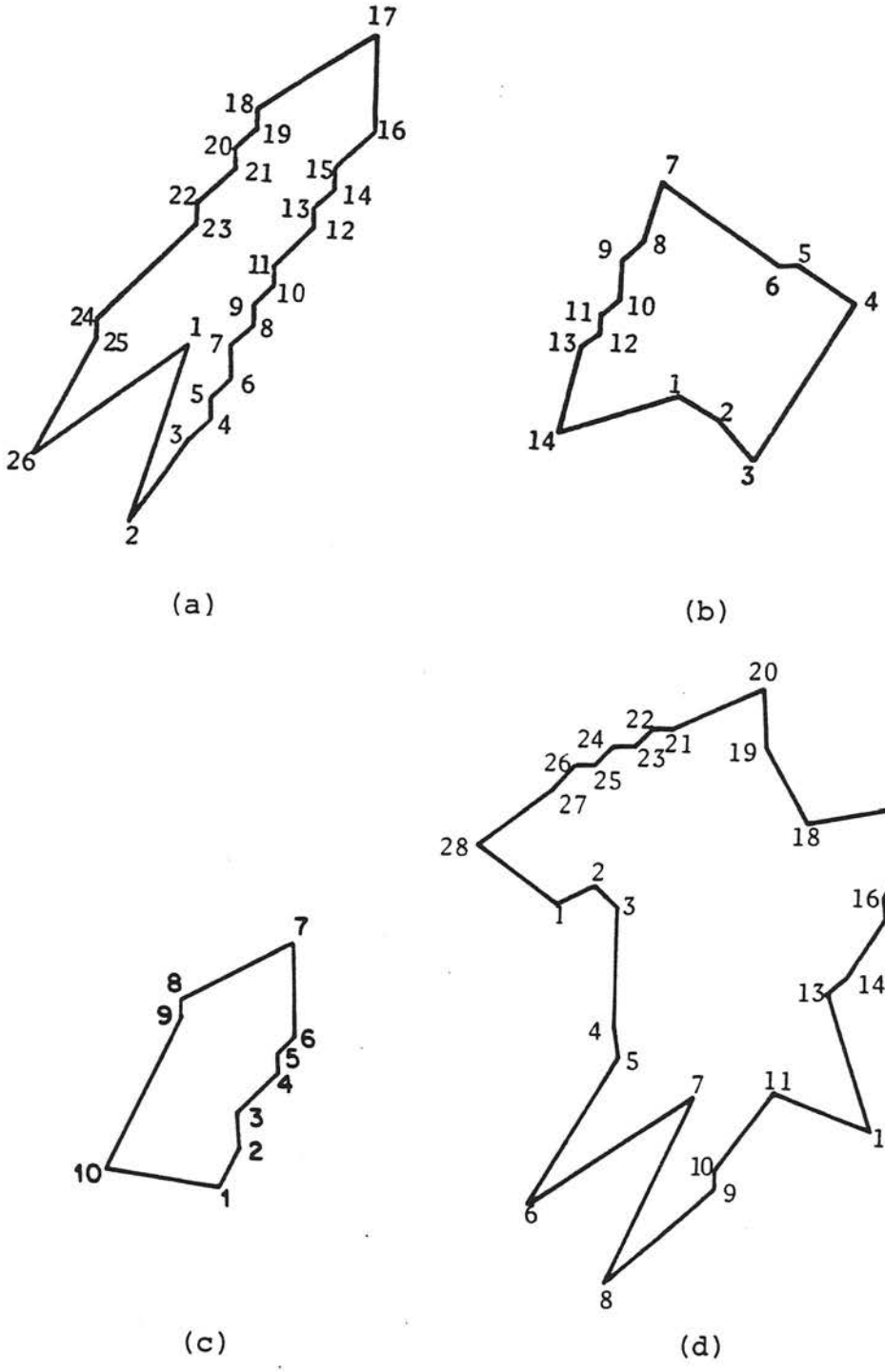


Fig. 3.7 (a) Model X_1, Number of segments = 26
(b) Model X_2, Number of segments = 14
(c) Model X_3, Number of segments = 10
(d) Apparent object, Number of segments = 28

Table 3.14

Results of the labeling of the models X_1, X_2 and X_3 when they are matched to the apparent object by themselves (i.e., no coordination). Example 3.4

Smoothing factor for X_1, X_2 and X_3 = 6. Smoothing factor for the apparent object = 8, α = 0.99, p_i(nil) = 0.25, ITER1 = 3, ITER2 = 4, UPTHLD = 0.8, LFACT = 25, No. of neighboring labels = 1, Inverse weights = (2, 3, -, -)

	Labels of X_1		Labels of X_2		Labels X_3	
Units of a Model	Initial Labeling	Final Labeling	Initial Labeling	Final Labeling	Initial Labeling	Final Labeling
1	29(.25)	9(1.0)	29(.25)	29(1.0)	29(.25)	29(1.0)
2	8(.32)	8(1.0)	29(.25)	29(1.0)	29(.25)	29(1.0)
3	29(.25)	29(1.0)	28(.30)	28(1.0)	23(.28)	23(1.0)
4	22(.27)	24(1.0)	29(.25)	29(1.0)	22(.27)	29(1.0)
5	23(.28)	29(1.0)	15(.27)	15(1.0)	23(.28)	23(1.0)
6	22(.27)	29(1.0)	29(.25)	29(1.0)	29(.25)	24(1.0)
7	23(.28)	23(1.0)	29(.25)	20(1.0)	29(.25)	28(1.0)
8	22(.27)	29(1.0)	29(.25)	29(1.0)	29(.25)	29(1.0)
9	23(.28)	29(1.0)	22(.27)	22(1.0)	29(.25)	19(1.0)
10	22(.27)	22(1.0)	23(.28)	23(1.0)	29(.25)	28(1.0)
11	23(.28)	29(1.0)	22(.27)	24(1.0)		
12	22(.27)	24(1.0)	29(.25)	25(1.0)		
13	23(.28)	29(1.0)	29(.25)	29(1.0)		
14	22(.27)	22(1.0)	29(.25)	12(1.0)		
15	23(.28)	23(1.0)				
16	29(.25)	29(1.0)				
17	29(.25)	29(1.0)				
18	29(.25)	9(1.0)				
19	23(.28)	23(1.0)				
20	22(.27)	29(1.0)				
21	23(.28)	29(1.0)				
22	22(.27)	24(1.0)				
23	23(.28)	25(1.0)				
24	29(.25)	29(1.0)				
25	29(.25)	29(1.0)				
26	6(.31)	6(1.0)				

Table 3.15

Results of labeling for the model X_1 when models X_1, X_2 and X_3 are matched to the apparent object using the Occlusion Algorithm. Example 3.4. Parameters same as in Table 3.14 except ITER2 = 8, PERCEN = 40%

	Labels at different iterations				
Units of Model X_1	First Stage			Second Stage	
	0	1	3	1	8
1	29(.25)	29(.32)	29(.41)	29(.41)	7(1.0)
2	8(.32)	8(.42)	8(.51)	8(.62)	8(1.0)
3	29(.25)	29(.31)	29(.40)	29(.35)	8(1.0)
4	22(.27)	24(.30)	29(1.0)	29(1.0)	29(1.0)
5	23(.28)	23(.33)	29(.38)	29(.65)	29(1.0)
6	22(.27)	22(.29)	29(.40)	29(.43)	22(1.0)
7	23(.28)	23(.33)	29(.39)	29(.72)	29(1.0)
8	22(.27)	22(.29)	29(.38)	29(.42)	22(1.0)
9	23(.28)	23(.33)	29(.39)	29(.73)	29(1.0)
10	22(.27)	22(.29)	29(.40)	29(1.0)	29(1.0)
11	23(.28)	23(.33)	29(.40)	29(.76)	29(1.0)
12	22(.27)	22(.29)	29(.40)	29(1.0)	29(1.0)
13	23(.28)	23(.33)	29(.38)	29(.73)	29(1.0)
14	22(.27)	22(.30)	29(.44)	29(1.0)	29(1.0)
15	23(.28)	23(.33)	29(1.0)	29(1.0)	29(1.0)
16	29(.25)	29(.35)	29(.53)	29(.64)	29(1.0)
17	29(.25)	29(.35)	29(.52)	29(.61)	6(1.0)
18	29(.25)	29(.36)	29(.52)	29(.62)	9(1.0)
19	23(.28)	23(.33)	29(1.0)	29(1.0)	29(1.0)
20	22(.27)	22(.29)	29(.42)	29(.50)	22(1.0)
21	23(.28)	23(.33)	29(1.0)	29(1.0)	29(1.0)
22	22(.27)	29(.30)	29(.55)	29(1.0)	29(1.0)
23	23(.28)	25(.34)	29(.79)	29(1.0)	29(1.0)
24	29(.25)	29(.36)	29(.58)	29(.72)	29(1.0)
25	29(.25)	29(.32)	29(.57)	29(.61)	5(1.0)
26	6(.31)	6(.41)	6(.64)	6(1.0)	6(1.0)
First term of the objective function	-	2.460	6.555	10.85	12.31
Penalty term of the objective function	-	.9842	2.622	4.343	0
Criterion	-	1.476	3.933	6.515	12.31
Penalty Constant	-	.00501	.0335	.322	-

Table 3.16

Results of labeling for the model X_2 when models X_1, X_2 and X_3 are matched to the apparent object using the Occlusion Algorithm. Example 3.4. Parameters same as in Table 3.14 except ITER2 = 8, PERCEN = 40%

Units of Model X_2	Labels at different iterations: First Stage 0	First Stage 1	First Stage 3	Second Stage 1	Second Stage 8
1	29(.25)	29(.29)	29(.39)	29(.51)	29(1.0)
2	29(.25)	29(.31)	29(.47)	29(.49)	16(1.0)
3	28(.30)	28(.37)	28(.50)	28(.56)	29(1.0)
4	29(.25)	29(.28)	29(.35)	29(.36)	29(1.0)
5	15(.27)	15(.35)	15(.54)	15(1.0)	15(1.0)
6	29(.25)	29(.30)	29(.48)	29(.53)	29(1.0)
7	29(.25)	29(.31)	29(.50)	29(.55)	20(1.0)
8	29(.25)	29(.32)	29(.54)	29(.60)	21(1.0)
9	22(.27)	22(.30)	29(.42)	29(1.0)	29(1.0)
10	23(.28)	23(.33)	29(1.0)	29(1.0)	29(1.0)
11	22(.27)	24(.30)	29(.42)	29(1.0)	29(1.0)
12	29(.25)	29(.31)	29(.57)	29(.62)	29(1.0)
13	25(.25)	29(.32)	29(.54)	29(.59)	15(1.0)
14	29(.25)	29(.32)	29(.41)	29(.42)	29(1.0)
First term of the objective function	-	1.430	3.280	4.537	5.774
Penalty term of the objective function	-	.5723	1.312	1.815	0
Criterion	-	.8584	1.968	2.722	5.774
Penalty Constant	-	.0029	.1678	.1346	-

Table 3.17

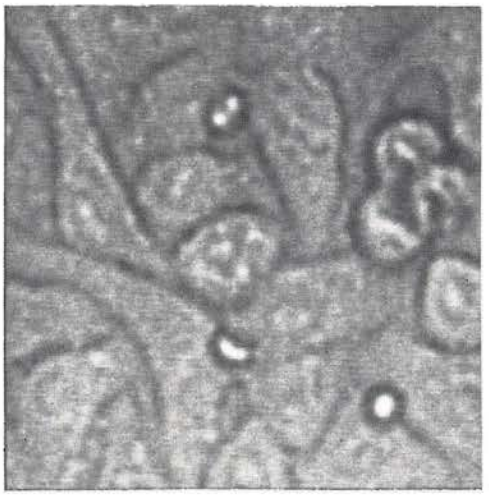
Results of labeling for the model X_3 when models X_1, X_2 and X_3 are matched to the apparent object using the Occlusion Algorithm. Example 3.4. Parameters same as in Table 3.14 except ITER2 = 8, PERCEN = 40%

Units of Model X_3	Labels at different iterations				
	First Stage			Second Stage	
	0	1	3	1	8
1	29(.25)	29(.31)	29(.53)	29(.61)	29(1.0)
2	29(.25)	29(.39)	29(.52)	29(.56)	29(1.0)
3	29(.28)	23(.33)	29(1.0)	29(1.0)	29(1.0)
4	22(.27)	22(.29)	29(.67)	29(1.0)	29(1.0)
5	23(.28)	23(.33)	29(.35)	29(.67)	29(1.0)
6	29(.25)	29(.34)	29(.55)	29(.66)	29(1.0)
7	29(.25)	29(.34)	29(.47)	29(.55)	28(1.0)
8	29(.25)	29(.33)	29(.57)	29(.63)	29(1.0)
9	29(.25)	29(.29)	29(.41)	29(.41)	29(1.0)
10	29(.25)	29(.33)	29(.47)	29(.53)	29(1.0)
First term of the objective function	-	1.003	2.056	3.680	8.145
Penalty term of the objective function	-	.4012	.8226	1.472	0
Criterion	-	.6018	1.233	2.208	8.145
Penalty Constant	-	.00204	.0105	.1091	-

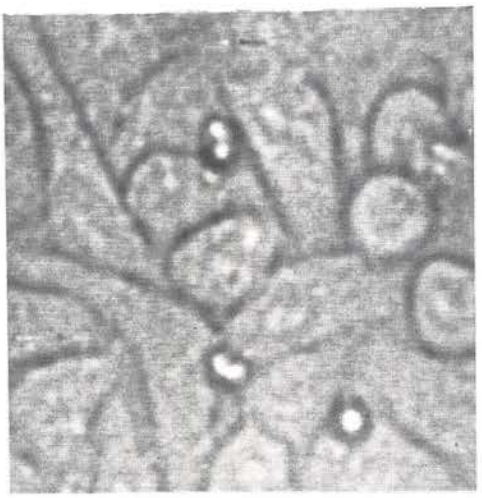
would be 22. The solution to such problems is to increase the number of segments and thus obtain a better polygonal approximation. Also we should use large size images. The total computation time for this example is 1500 seconds. As compared to the example 3.3 it can be seen that the computation time increases with the increase in the number of objects participating in the occlusion and the complexity of the matching task. If matching is relatively simple, computation time will be less.

Example 3.5

Fig. 3.8 shows two 128x128, 8 bit images of cells taken from a sequence of images. These images are 15 minutes apart in the sequence. The system used to obtain these cell images is described in Appendix B. The background in these images consists of human skin cancer cells and the small circular shaped objects are human lymphocytes and red blood cells. One cancer cell in the image of fig. 3.8(a) is undergoing mitosis phenomenon. In the image of fig. 3.8(b), the cell of fig. 3.8(a) undergoing mitosis (parent cell) has been divided into two cells (daughter cells). We provide this example in order to critically evaluate the powers of our occlusion algorithm. We want to match the daughter cells of fig. 3.8(b) with the parent cell of fig. 3.8(a). Note



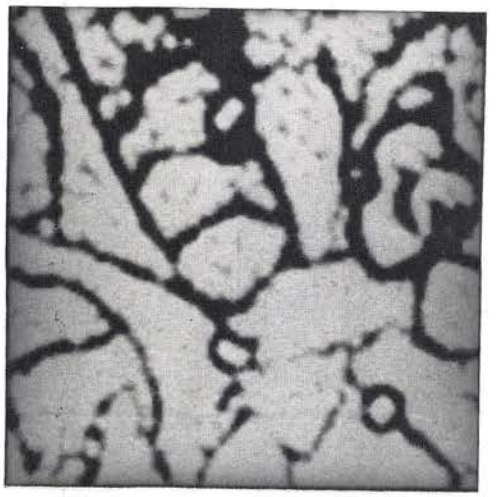
(a) The cell undergoing mitosis



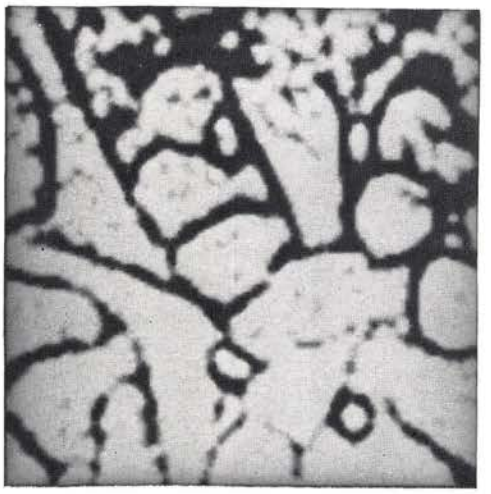
(b) The cell in fig. 3.8(a) is divided into 2 cells after 15 minutes

Fig. 3.8 Mitosis of cancer cells

that significant changes in shape have taken place. We use the segmentation technique described in Chapter 6 to get the cell boundaries. In this technique we have used FACT = 0.9, $\alpha_1 = 0.5$, $\alpha_2 = 0.1$ (see Chapter 6). The resulting images obtained at the third iteration are shown in fig. 3.9(a) and 3.9(b) corresponding to the images of fig. 3.8(a) and 3.8(b) respectively. Figs. 3.10 and 3.11 show the polygonal approximation of the boundary of the cells obtained from the images of fig. 3.9. A smoothing factor of 8 is used in the polygonal approximation. Note that the upper daughter cell in fig. 3.9 is segmented quite differently than the lower daughter cell. We match the models X_1 and X_2 (fig. 3.10) with the apparent object in fig. 3.11. The results of labeling without the use of Occlusion algorithm are shown in fig. 3.18. Note that the same segments of the apparent object are assigned to the units of the different models. These errors are eliminated by using the occlusion algorithm (see Table 3.19 and 3.20). The assignment of unit 20 of model X_1 is wrong, but the assignments of units 18 and 19 are correct. The unit 13 should have been labeled as 23, but a careful examination of the figures in 3.10 and 3.11 shows that the local structure of unit 20 matches to label 23 better than to 13. All the assignments of the units of model X_2 are correct except the assignment of unit 9 which is matched

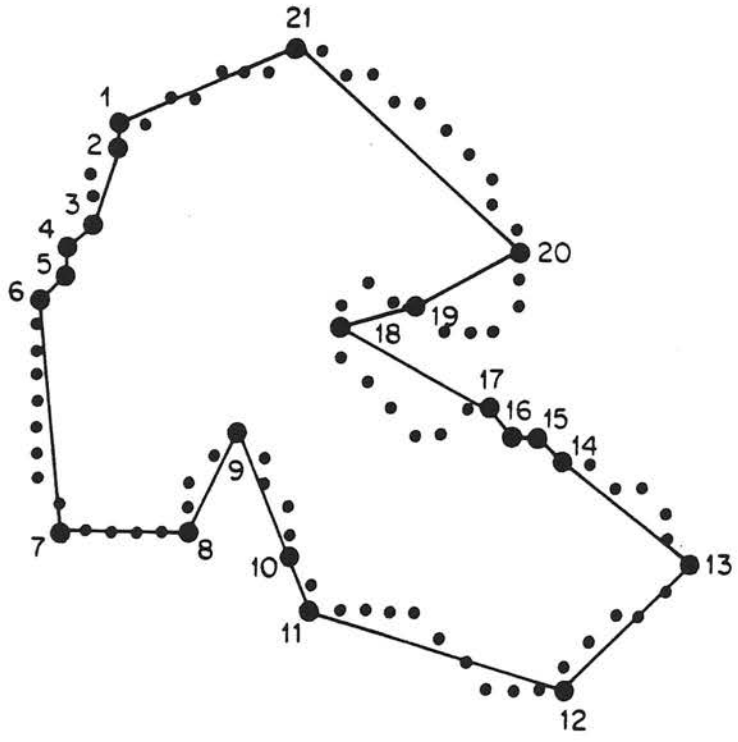


(a) Segmentation of the image in fig. 3.8(a)

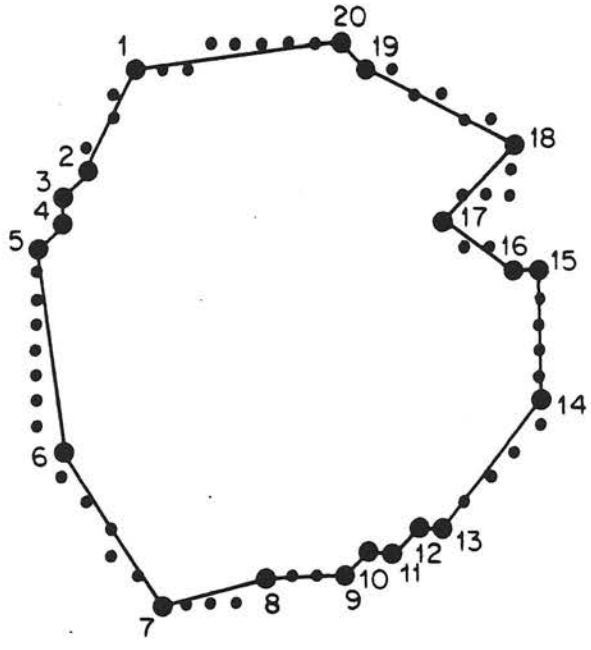


(b) Segmentation of the image in fig. 3.8(b)

Fig. 3.9 Images obtained at the third iteration when the segmentation technique described in Chapter 6 is applied to the images in Fig. 3.8. Parameters used are FACT = .9, α_1 = .5, α_2 = .1



(a) Model X_1, Number of segments = 21



(b) Model X_2, Number of segments = 20

Fig. 3.10 Polygonal approximation of the two cells in fig. 3.8(b), which resulted from the mitosis of the cell in fig. 3.8(a)

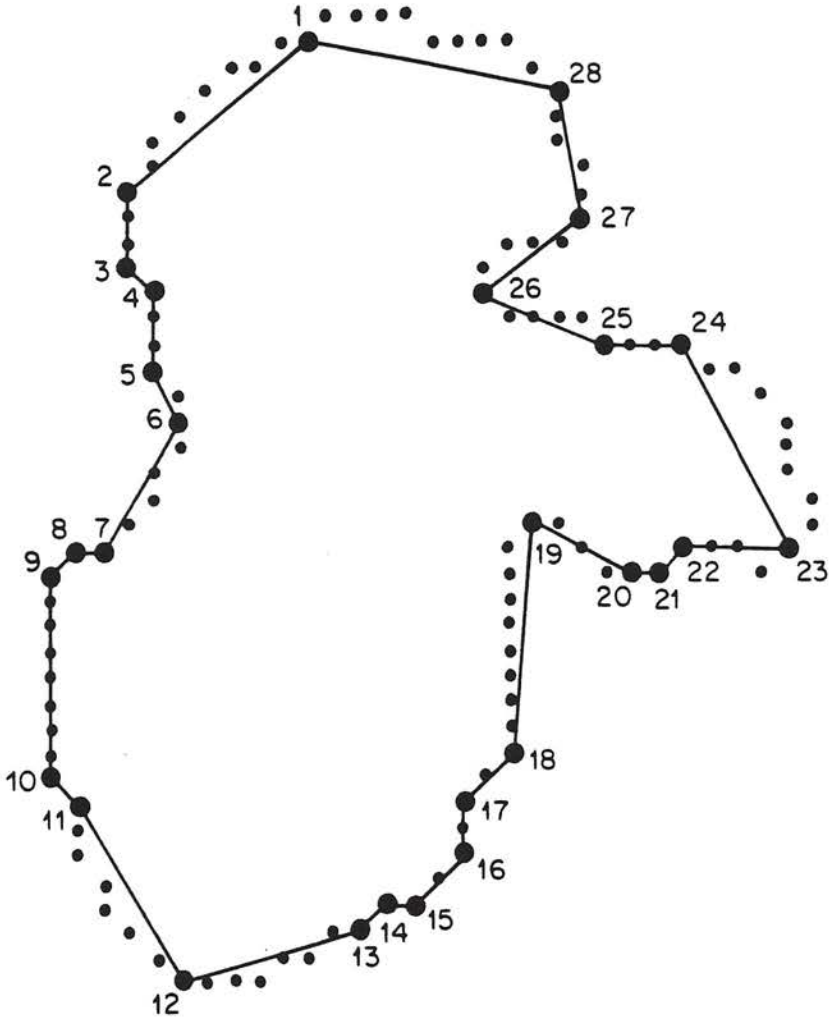


Fig. 3.11 Polygonal approximation of the cell undergoing mitosis (fig. 3.8(a))
Apparent object, Number of segments = 28

Table 3.18

Results of labeling of the models X_1 and X_2 when they are matched to the apparent object by themselves (i.e., no coordination). Example 3.5

Smoothing factor = 8, α =0.99, p_i(nil) = 0.25, ITER1 = 3, ITER2 = 4, UPTHLD = 0.8, LFACT = 20, No. of neighboring labels = 1, Inverse weights = (4, 3, 5, 1)

Units of a Model	Labels of X_1		Labels of X_2	
	Initial Labeling	Final Labeling	Initial Labeling	Final Labeling
1	29(.25)	29(1.0)	29(.25)	29(1.0)
2	29(.25)	29(1.0)	29(.25)	29(1.0)
3	29(.25)	29(1.0)	29(.25)	3(1.0)
4	29(.25)	3(1.0)	29(.25)	7(1.0)
5	29(.25)	7(1.0)	29(.25)	9(1.0)
6	29(.25)	29(1.0)	29(.25)	29(1.0)
7	29(.25)	29(1.0)	29(.25)	12(1.0)
8	29(.25)	29(1.0)	29(.25)	29(1.0)
9	29(.25)	29(1.0)	21(.43)	13(1.0)
10	29(.25)	29(1.0)	14(.47)	14(1.0)
11	29(.25)	29(1.0)	21(.43)	15(1.0)
12	29(.25)	29(1.0)	14(.26)	29(1.0)
13	29(.25)	29(1.0)	29(.25)	15(1.0)
14	29(.25)	25(1.0)	29(.25)	29(1.0)
15	29(.25)	24(1.0)	29(.25)	29(1.0)
16	29(.25)	29(1.0)	29(.25)	25(1.0)
17	29(.25)	29(1.0)	29(.25)	29(1.0)
18	29(.25)	26(1.0)	29(.25)	29(1.0)
19	29(.25)	29(1.0)	29(.25)	25(1.0)
20	29(.25)	23(1.0)	29(.25)	29(1.0)
21	29(.25)	29(1.0)		

Table 3.19

Results of labeling for the model X_1 when models X_1 and X_2 are matched to the apparent object using the Occlusion Algorithm. Example 3.5. Parameters same as in Table 3.18 except ITER2 = 8, PERCEN = 80%

Units of Model X_1	Labels at different iterations				
	First Stage			Second Stage	
	0	1	3	1	7
1	29(.25)	29(.32)	29(.47)	29(.50)	29(1.0)
2	29(.25)	29(.29)	29(.44)	29(.47)	29(1.0)
3	29(.25)	29(.31)	29(.47)	29(.48)	29(1.0)
4	29(.25)	29(.36)	29(.46)	29(.47)	29(1.0)
5	29(.25)	29(.32)	29(.50)	29(.54)	29(1.0)
6	29(.25)	29(.31)	29(.48)	29(.49)	29(1.0)
7	29(.25)	29(.33)	29(.45)	29(.49)	29(1.0)
8	29(.25)	29(.29)	29(.41)	29(.42)	29(1.0)
9	29(.25)	29(.30)	29(.37)	29(.40)	29(1.0)
10	29(.25)	29(.32)	29(.42)	29(.43)	29(1.0)
11	29(.25)	29(.31)	29(.45)	29(.47)	29(1.0)
12	29(.25)	29(.31)	29(.40)	29(.43)	29(1.0)
13	29(.25)	29(.29)	29(.38)	29(.40)	29(1.0)
14	29(.25)	29(.31)	29(.51)	29(.54)	29(1.0)
15	29(.25)	29(.32)	29(.41)	29(.44)	29(1.0)
16	29(.25)	29(.31)	29(.37)	29(.38)	29(1.0)
17	29(.25)	29(.28)	29(.41)	29(.43)	29(1.0)
18	29(.25)	29(.30)	29(.36)	29(.38)	26(1.0)
19	29(.25)	29(.31)	29(.45)	29(.49)	26(1.0)
20	29(.25)	29(.30)	29(.48)	29(.58)	23(1.0)
21	29(.25)	29(.29)	29(.44)	29(.45)	29(1.0)
First term of the objective function	-	1.862	3.137	3.755	15.99
Penalty term of the objective function	-	1.490	2.509	3.004	0
Criterion	-	.3725	.6274	.7511	15.99
Penalty Constant	-	.0206	.5420	15.44	-

Table 3.20

Results of labeling for the model X_2 when models X_1 and $\dot{X}_2$ are matched to the apparent object using the Occlusion Algorithm. Example 3.5. Parameters same as in Table 3.18 except ITER2 = 7, PERCEN = 80%

Units of Model X_2	Labels at different iterations				
	First Stage			Second Stage	
	0	1	3	1	7
1	29(.25)	29(.56)	29(.74)	29(.77)	29(1.0)
2	29(.25)	29(.55)	29(1.0)	29(1.0)	29(1.0)
3	29(.25)	29(.39)	29(.52)	29(.54)	29(1.0)
4	29(.25)	29(.50)	29(.73)	29(.75)	29(1.0)
5	29(.25)	29(.37)	29(.55)	29(.64)	9(1.0)
6	29(.25)	29(.40)	29(.68)	29(.74)	29(1.0)
7	29(.25)	29(.41)	29(.60)	29(.63)	29(1.0)
8	29(.25)	29(.41)	29(.53)	29(.56)	14(1.0)
9	21(.43)	21(.51)	21(.56)	21(.61)	21(1.0)
10	14(.47)	14(.54)	14(.56)	29(.60)	29(1.0)
11	21(.43)	21(.49)	21(.54)	29(.79)	29(1.0)
12	14(.26)	29(.36)	29(.42)	29(.43)	14(1.0)
13	29(.25)	29(.37)	29(.52)	29(.56)	15(1.0)
14	29(.25)	29(.57)	29(1.0)	29(1.0)	29(1.0)
15	29(.25)	29(.54)	29(1.0)	29(1.0)	29(1.0)
16	29(.25)	29(.55)	29(.72)	29(1.0)	29(1.0)
17	29(.25)	29(.46)	29(.60)	29(.65)	29(1.0)
18	29(.25)	29(.50)	29(1.0)	29(1.0)	29(1.0)
19	29(.25)	29(.60)	29(.72)	29(.80)	29(1.0)
20	29(.25)	29(.46)	29(.62)	29(.67)	29(1.0)
First term of the objective function	-	1.676	5.210	8.169	11.00
Penalty term of the objective function	-	1.341	4.168	6.535	0
Criterion	-	.3352	1.042	1.633	11.00
Penalty Constant	-	.0185	.9002	33.59	-

to the label 21. This is again because of the close resemblance of the local structure. The total computation time for this example is 1100 seconds.

Comments

From the examples presented in this chapter, it can be seen that the occlusion algorithm works very well. Although, "perfect" matching results are not obtained (some labels are incorrect), the technique is quite powerful. The reasons for incorrect matches are: The local structure of the incorrect match is more similar than the correct match, the polygonal approximation is crude, segmentation results are very different and changes in the shape are too drastic. The variety of examples with different scales, rotation, poor polygonal approximation and significant changes in the shape illustrate the applications of the technique and critically evaluate its potential especially when we deal with real images. The technique allows for the selection of penalty constants automatically and provides stable results.

3.5 Summary

In this chapter we presented an extension of the hierarchical stochastic labeling technique to do shape

matching of partially occluded two-dimensional objects by combining the gradient projection method and penalty function approach. The objects participating in the occlusion are known a priori. There is a hierarchical process for every object participating in the occlusion and these processes are executed in such a fashion that none of the segments of the different models are assigned to the same segment of the apparent object. Penalty constants are chosen in an automatic manner. The technique is quite powerful. Its capabilities and powers are illustrated by presenting synthetic, industrial and biological examples. The computation time varies linearly with the number of objects occluding one another. If the objects are rigid as has been mostly assumed in the past work, matching will be relatively simple. After matching actual objects with the apparent object, it will be easier to track them and carry out the motion analysis. As compared to the previous studies, the framework presented in this work provides a mathematical basis for the solution of the occlusion problem which is of basic importance in the shape matching of two-dimensional objects.

CHAPTER 4

REPRESENTATION, MODELLING AND THREE-DIMENSIONAL SCENE ANALYSIS

4.1 Introduction

In the last two chapters we presented a 2-D shape matching algorithm based on hierarchical stochastic labeling technique and its extension to include occluded objects using a gradient projection method and penalty function approach. In this chapter, the representation and modelling aspects of 3-D scene analysis will be considered. A method based on a laser triangulation to acquire 3-D data will be described. The problems related with 3-D data acquisition and geometric processing will be addressed. A technique for representing a three-dimensional object by a set of planar convex faces will be presented. These faces are determined by sequentially choosing three very close noncolinear points and investigating the set of points lying in the plane of these points. Two simple tests, one for convexity and the other for narrowness ensure that the set of points is an object face. This set of points is approximated by

polygons. The method is used to generate a 3-D model of an object by combining the object points from a sequence of range data images corresponding to various views of the object and applying the necessary transformations. The method can also be used on the individual range data image. An example which includes a complicated casting of an automobile piece used in the suspension system will be presented. The control structure for the shape matching of 3-D objects will be described in the next chapter.

4.2 Three-Dimensional Data Acquisition

Most of the work in scene analysis has been the interpretation of a 2-D image as a 3-D scene [4-1, 4-2]. It has been concerned with some of the difficult problems caused by projection, occlusion, illumination effects etc. Although many monocular depth clues can be used to produce a three-dimensional interpretation of a 2-D image, the direct measurement of range (the distance of the observed points in a scene from the viewer) simplifies many of these problems considerably. The availability of range is particularly useful in 3-D shape analysis and segmentation.

Fig. 4.1 shows the schematic diagram of the 3-D scene analysis system implemented in this work. First we acquire 3-D data using a sensory device (described in this

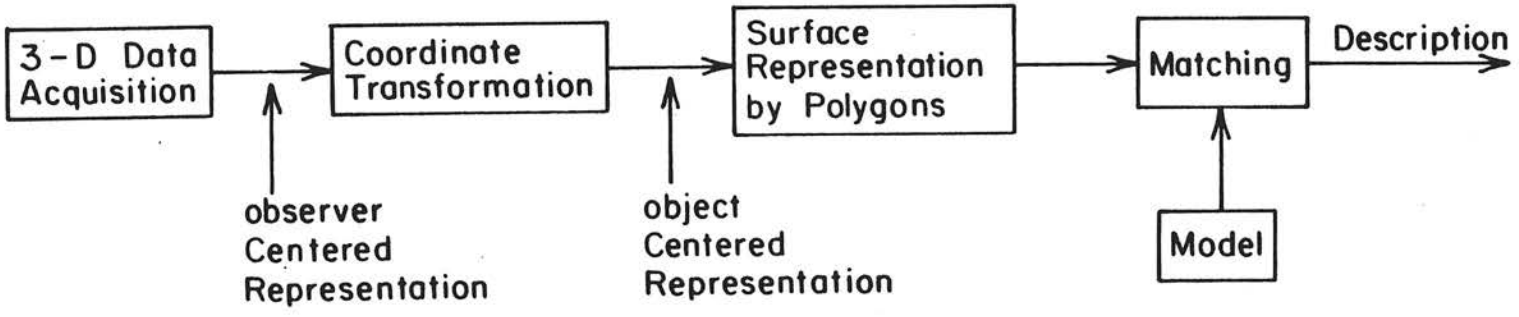


Fig. 4.1 The schematic diagram of 3-D scene analysis system

section). This data is in the observer centered coordinate system (the one in which the observer or camera receives the image). It needs to be suitably transformed to the object centered representation (a system centered about the object which allows all points on the surface of the object to be referred with respect to this system). We then obtain a higher level representation of surface or volume (surface is used here) and finally the unknown scene is matched against the model to obtain the description of the scene. In this section we shall describe 3-D data acquisition. Details about the coordinate transformation are presented in section 4.5.

Many different methods for acquiring three-dimensional data have been developed [4-3 to 4-11]. They include direct manual measurement, mechanical moving devices, holographic methods, Moire methods, multiple 2-D images, controlled illumination on objects etc.

Humans are able to perceive the 3-D world using either the monocular cues (size perspective, shading and shadow, accomodation of lens, texture gradient etc.) or binocular cues (stereo, motion parallax, converging of the eyes). All other cues except the accomodation of lens (focussing differently for different objects) and the converging of the eyes give the relative depth. Machine

implementation of monocular depth cues such as texture gradients [4-12], shadows, highlights etc. do not provide the detailed shape information so the binocular depth cues such as stereo and motion parallax are increasingly used. The techniques used in the 3-D scene analysis work for range measurement can be classified into two categories: Triangulation and time of flight. Triangulation is subdivided into passive (rely solely on normal scene illumination) and active (use controlled illumination). The passive methods take the stereo pair of pictures either from two cameras or from one camera in two positions. Range to a point in one image can be obtained by finding the corresponding point in the other image. Most of work related to stereopsis in scene analysis has focused on solving this corresponding problem [4-13, 4-14]. The active triangulation method eliminates the correspondence problem by using a spatially controlled illumination and obtaining the image by means of a TV camera. A number of active methods have been developed [4-3, 4-5].

The accuracy of triangulation range finding methods depends upon the distance between two views. The inherent drawback of these methods is the inaccurate range values for distant targets, and missing data for close targets because many points in the scene may not be seen by both

views. The time of flight techniques remove the problems of triangulation methods. Lewis and Johnston [4-7] used a pulsed laser and measured the elapsed time of reflected light. Nitzan et al. [4-3, 4-4] used a continuous wave modulated beam and measured the phase shift. Their system can also get the intensity of reflected light at the same time, thus it provides the registered intensity and range data.

The object surface point data used in this study was obtained with a laser ranging system whose principle is shown in Fig. 4.2. A laser emits a beam of ruby red light which is reflected by a mirror which rotates and sweeps the beam along the x-axis to produce one scan line. The beam is reflected from the object, and the z-distance is calculated from the location of the maximum response in each bank of detectors (and the orientation of the mirror, if necessary). The platform on which the object rests can be raised or lowered (this is the y-axis) and can also be rotated (around the y-axis). The sampling distances used here are 3.0 mm in the x-axis, 2.0 mm in the y-axis, and an accuracy of 0.01 mm in the z-distance is achieved.

This data is in the observer centered coordinate system. While creating a 3-D model of the object, object centered representation is required. This is computed by

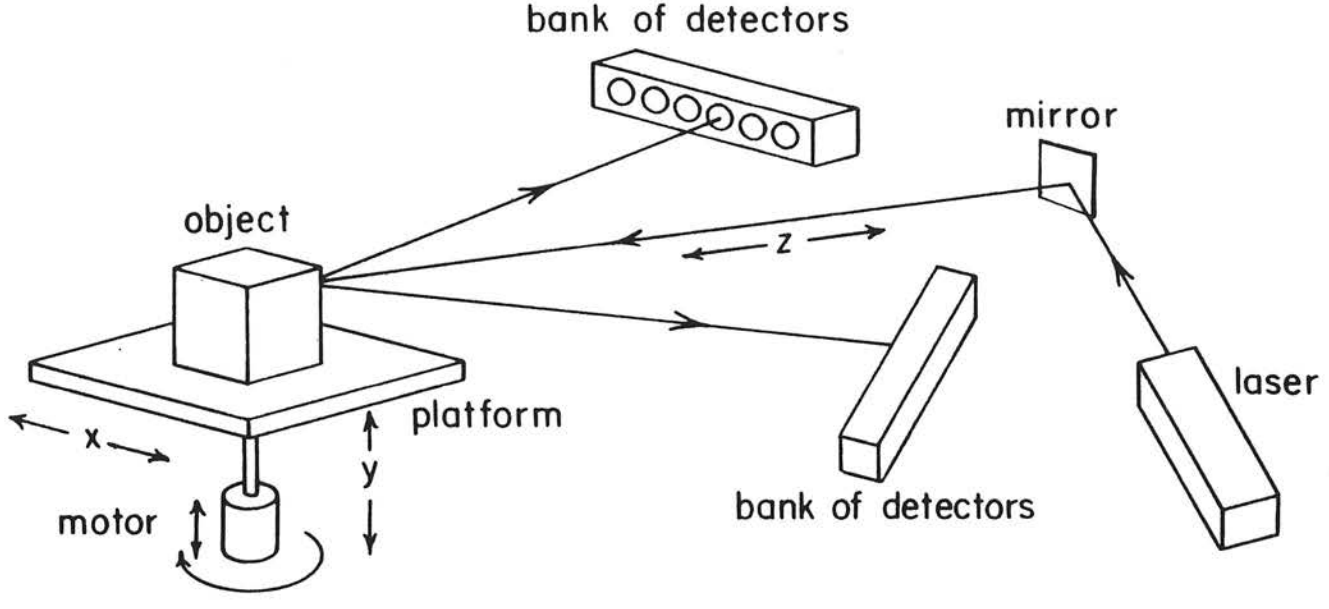


Fig. 4.2 Laser ranging system

marking the zero position for x- and y-axis and obtaining a reference value for z-axis on the platform on which the object rests (fig. 4.2). Thus all the points can be transformed to the same coordinate system. While acquiring the data related to an unknown view of the object, the actual position or orientation of the object on the platform does not matter. The objective of the shape matching algorithm (presented in the next chapter) is to do the partial shape recognition and use the results of matching to determine the orientation of the object in space. In order to create a 3-D model of the object, a range data image was produced for every 30° rotation of the object around the y-axis. Finally, top and bottom views of the object were taken. These two views were put in correspondence with the other views by using several control points on the object to compute the transformation. More details about the issues related to data acquisition and coordinate transformation are discussed in detail in section 4.5.

4.3 Representation and Modelling of 3-D Objects

Representation

A direct model of a 3-D object as a 3-D array can easily exhaust the memory capacity of a system (for example a 3-D array of size 128 will require $128^3=2097152$

bytes of memory). Moreover, this array is sparse. Therefore, we are interested in a suitable representation, not for storage purposes only, but for recognition and description as well. Representation of a three-dimensional object by means of oct-trees may make space array (triply subscripted binary array) operations more economical in terms of memory space [4-15].

"Representation" is defined as the act of making a description of an object. The description should capture an intuitive notion of shape and it should be compact and simple. The interest in object description is not limited to the computer vision and 3-D scene analysis. For example in computer graphics, the generation of object description for use with the hidden surface algorithm is a major task. The object must first be approximated by a set of planar faces, which has been considered not unlike an art of representing objects with paints on canvas [4-16]. Similarly, for the representation of terrain (data obtained by synthetic aperture radar or laser ranging devices) an efficient alternative to dense grids is to approximate the surface by planar faces of irregular size and shape.

Our interest is in the matching based on shape description. The simplest approach to analyzing 3-D

objects is to model them as polyhedra [4-17] and store the 3-D data as in computer graphics [4-16]. Once the structure of the model has been determined, unknown objects can be matched to this structure. However, this requires a description of the object in terms of vertices, edges and faces. Modelling 3-D objects in this manner results in substantial compression of the data. In order to handle curved and more complex objects, other representations and models have been investigated [4-11,4-18 to 4-22].

Binford [4-18] proposed the concept of a generalized cylinder (or cone) to represent curved 3-D objects. These are defined by a 3-D space curve, known as the axis, and cross-section of arbitrary shapes and sizes along the axis. There are an infinite number of possible generalized cones representing a single object. More constraints are needed to get a unique description. Agin and Binford [4-11], Nevatia [4-19] and Bajcsy and Soroka [4-20] all have used this concept to segment complex objects. Agin's description techniques were unstructured and have some major deficiencies such as some descriptions are merged with nearby but distinct parts. Nevatia did not use depth information except for boundary points. Therefore the representation is very sensitive to boundary shape. Connectivity relations of the subparts at various

types of joints represent the structure of the objects. Finally, symbolic description was compared with stored models represented in the same way. Bajcsy and Soroka used a generalized cone with elliptical cross-sections for representation of serial tomograms. Marr [4-21] proposes a hierarchy of models using generalized cones for representation. Since organs have a typical shape, a longitudinal axis and are smooth (no sharp changes and irregularities in boundaries), Shani [4-22] uses generalised cylinder representation in the study of recognition of abdominal anatomy from computer tomograms (CT) scans.

Although generalized cones or volume representations imply some surface description, they fail to describe the junctions or surface peculiarities [4-14]. Also one detects surfaces first from partial views, and only after several different views of the object we have enough data to obtain volume properties. Hence the need to find a suitable surface representation. It is possible to represent arbitrary shapes with generalised cones by making them arbitrarily complex, but their computation is difficult. The generalized cone primitives used in [4-19] are not sufficient to represent the complicated casting, as has been used in this work. Badler and Bajcsy [4-23] present a good discussion of the relative merits of

surface representations (a set of 3-D points, a polygonal network, curved surface patches of various kinds, quadric patches) and volume representations (cellular space, convex polyhedra, geometric forms, ellipsoids, cylinders, spheres) on the basis of storage cost, complexity, operation on representation, conversion of one representation into another etc.

Our approach to the analysis of a 3-D range data image is to first extract the relevant 3-D object as sets of 3-D points and then work directly on these sets without regard to the original image. This approach frees one from a particular image and is in the vein of much work in computational geometry [4-24]. The goal is to produce a complete description of the surface of a 3-D object in terms of faces. Such a representation should be complete, that is, it should sample the entire surface of the object, and allow for matching of individual views of the object taken from any vantage point. An object is thus defined as a finite number of selected points in three-space. However, only the geometrical position of each point is known; no topological information is available. This lack of an ordering on the points is quite different from the case of one-dimensional curves and thus leads to a radically different segmentation strategy.

Methods for segmentation of range data can be classified as "region" or "edge" based just as in the segmentation of intensity images. Many researchers adopted a method which is most suitable for the input device. For example, Duda et al. [4-4] describe a sequential procedure for determining planar surfaces in a scene from registered range and intensity data. The vertical and horizontal surfaces are obtained directly from the range image by a histogram analysis. Slanted surfaces are assumed to have constant intensity and are obtained from the reflectance image. Finally an iterative procedure refines the initial set of surfaces found. Milgram and Bjorklund [4-25] find planar surfaces in a range image (consisting mostly planar regions) to determine the actual sensor position with respect to a given reference model. Planar surfaces are found by fitting a least squares plane in the small neighborhood of each pixel. The orientation of the plane, its altitude from the sensor and goodness of fit are recorded as local features of each pixel. Finally, pixels are examined for their association with the neighbors to form connected components. When the range image consists of complex objects having non-planar surfaces also, use of this approach does not seem suitable. Ishii and Nagata [4-6] obtained the contour of an object by controlling a laser

spot. Agin [4-11] fitted quadratic curves to the images of sheets of a laser beam. Shirai [4-5] and co-workers have used region and edge based methods to represent polyhedra and simple curved objects. Popplestone et al. [4-26] dealt with polyhedra and cylinders. Sugihara and Shirai [4-27] used a dictionary of Huffman/Clowes type labels to find missing edges and to segment range data. Inokuchi and Nevatia [4-28] presented a technique for obtaining surface edges, effectively a 3-D edge detector, while Zucker and Hummel [4-29] described an optimal 3-D surface edge operator which produces a smooth surface separating adjacent volumes. The drawback of these techniques is that the edge responses must be grouped, thinned and linked in order to produce a reasonable object description in terms of coherent region. On the other hand, once the line segments are found, the theory of 3-D line semantics can be directly applied. It is possible to extract planar surfaces from single view range data images by extending the iterative end point fit method [4-1] from two dimensions to three. This may work well since the range z can be considered as a function of two spatial coordinates x and y. All the above past techniques use one range image only. It should be noted that the kinds of representation used and the amount of computation required to extract such a representation depends upon the

intended use and the ultimate purpose of scene analysis.

Modelling

Representation and models are very intimately connected. Since most of the work in scene analysis has been the interpretation of a 2-D intensity image as a 3-D scene, two-dimensional models have been commonly used in the analysis with constraints on the configuration of objects whenever possible. Chien and Chang [4-30] take as input a list of vertices in the 2-D line drawing of a 3-D scene of curved objects. The curved edges of a body are represented piecewise linearly and the curved surfaces are represented as a list of vertices with various restrictions. Recognition is accomplished by a model in the form of a tree. McKee and Aggarwal [4-31] recognize partial views of 3-D curved objects like cup or hammer by matching the edge description with the stored model. The method requires good input of the surface boundaries. The method can describe and recognize a partial view of an isolated object in the presence of rotation, distortion and scale changes. Fischler and Elschlager [4-32] decomposed a human face into subparts and constructed the model with intensity arrays of subparts, and their configuration. Recognition is performed by matching an input picture to the intensity arrays placed at the best

position. Models of subparts can be easily generated, but the matching is sensitive to the scale and shading. Perkins [4-33] uses two-dimensional models to recognize industrial parts. These models can not handle occlusion and oblique viewing angle problems effectively.

A tradeoff is involved between representation and modelling [4-5]. 2-D models make the representation easier at the expense of a complex modelling task. 3-D models are more general, but the representation must take care of mapping it to the view-domain. They are very powerful for 3-D shape analysis. Horn [4-34] uses powerful 3-D surface models of terrains for registration of aerial images. Hierarchical models which involve both 2-D and 3-D models have been used [4-35]. In this work we generate a 3-D model of the object in terms of the planar faces on its surface approximated by polygons. Since our objective is to do partial 3-D shape recognition and use these results to find the orientation of the object in space, the surface approach has been preferred. The control structure used in this work is hierarchical and described in the next chapter.

4.4 Algorithm for Surface Approximation by Polygons

In this section we present an algorithm for surface approximation by polygons.

Representing a 3-D object as a set of planar faces approximated by polygons is a two step process (fig. 4.3). In the first step we find the set of points that belong to various faces of the object using a three point seed algorithm and in the second step, approximate the face points obtained in step 1 by polygons.

The three point seed method for the extraction of planar faces from range data is a sequential region growing algorithm. It is not applied directly to range images, but rather to a set of points. It is not restricted to single view range data image, but applicable to a composite object and does not require the ordering of points. It finds the convex faces of the object, but the information exists to merge convex parts of nonconvex faces . Although the 3-point seed algorithm is applied to a set of 3-D points, it is not directly related to how these points are obtained. The method is ultimately tied to the sampling distance between points on the object.

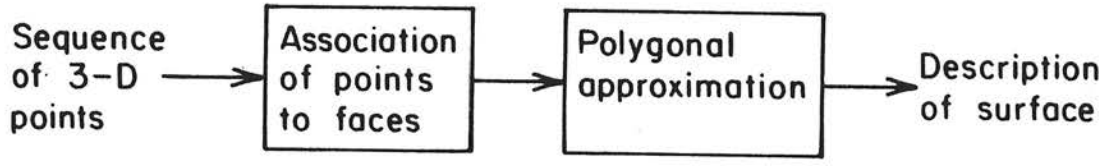


Fig. 4.3 Two step process to approximate surface by polygons

The 3-point seed method

In a well-sampled 3-D object, any three points lying within the sampling distance of each other (called a 3-point seed) form a plane (called the seed plane) which:

(a) coincides with that of the object face containing the points, or

(b) cuts any object face containing any of the 3 points.

A seed plane satisfying (a) results in a plane from which a face should be extracted, while a seed plane satisfying (b) should be rejected. Two simple conditions that suffice to determine if a plane falls into category (b) are: convexity and narrowness.

Let us consider the intent of the convexity and narrowness tests. Fig. 4.4 shows an object which has two faces (labeled 1 and 2) lying in the same plane. Thus, the set of points lying on this plane will be reduced by convexity to just those points in face 1 if the 3-point seed is in face 1, otherwise, face 2. Note that face 3 (a non-convex face) will be segmented differently depending on the order in which the points in that face are examined. The second case in which convexity plays a role is shown in Fig. 4.5. In this example (fig. 4.5(a)) the

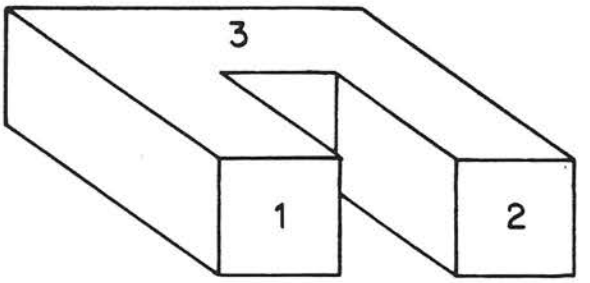
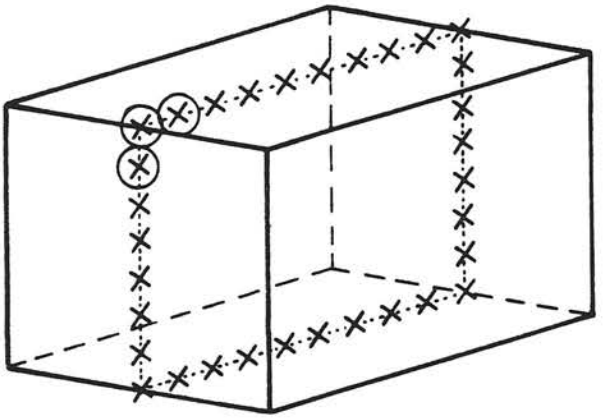
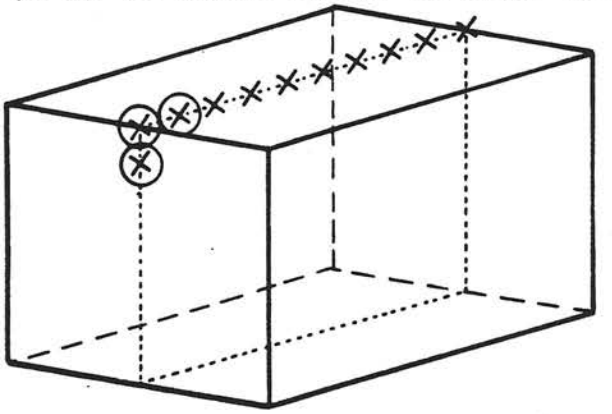


Fig. 4.4 Face 1 and 2 lie in the same plane, but are distinct; face 3 is non-convex.



(a) A three point seed (the circled x's) which straddles an edge produces a plane which cross-sections the object. The convexity condition reduces the set of points (x's) to those shown in fig. 4.5(b)



(b) The narrowness condition excludes faces like this whose points all lie very near the same line

Fig. 4.5 Convexity and Narrowness conditions

3-point seed (the three circled *'s) straddles an edge, and consequently the points in the seed plane (shown by *) lie on a section through the object and not on any actual object face. The convexity condition reduces this set of points to the points shown in Fig. 4.5(b). The peculiar property of such a reduced set is that its points all lie essentially on the line passing through the two most distant points in the set. This fact gives rise to the narrowness test, and any set of points which is "narrow" in this sense is rejected as a possible face.

The algorithm involves the following steps.

1. From the list of surface points select 3 points which are noncolinear and near relative to sampling distances.
2. Obtain the equation of the plane passing through the three points chosen in step 1.
3. Find the set of points P which are very close to this plane.
4. Apply the convexity condition to the set P to obtain a reduced convex set P'. This separates faces lying in the same plane.
5. Check the set P' obtained in step 4 for narrowness.
6. If the face is obtained correctly (i.e.,

convexity and narrowness conditions are satisfied), remove the set of points belonging to this face from the list of points and proceed to step 1 with the reduced number of points in the list.

The complete 3-point seed algorithm is described below as Algorithm 1.

```
procedure Three-point-seed;
begin
  POINTS := {X̄_i}, i = 1, n;
  for i := 1 to n-2 do
    if not marked (X̄_i)
      then for j := i+1 to n-1 do
        if not marked (X̄_j)
          then for k := j+1 to n do
            if not marked (X̄_k) and close-enough (X̄_i,X̄_j,X̄_k)
              then begin
                Set-plane-coefficients (X̄_i,X̄_j,X̄_k,a,b,c,d);
                PLANE := Points-in-plane (POINTS,a,b,c,d);
                FACE := Convex (PLANE,X̄_i,X̄_j,X̄_k);
                if not Narrow(FACE)
                  then begin
                    STORE (FACE);
                    Mark (FACE, POINTS);
                      end;
                  end;
end;
```

Algorithm 1. The 3-Point Seed Algorithm

POINTS is the set of n object points. Each point is described by its (x,y,z) coordinates. The predicate marked ($\bar{X}_i$) returns true if $\bar{X}_i$ has already been associated with a face and false otherwise. In this way, the number of combinations of 3 points to be tried as a seed is reduced since, once a point is associated with a face, it cannot be part of a new 3-point seed. The predicate close-enough returns true if the three unmarked points are each within $(\Delta x^2 + \Delta y^2)^{1/2}$ of the others (i.e. 8 neighbors in the sampling grid), where Δx and Δy are the distances between samples in the x-direction and y-direction, respectively. These thresholds will be discussed more fully in the next section.

Once the 3-point seed has been found, the seed plane, PL, passing through these points can be analyzed for the existence of an acceptable face containing the seed ponts. The coefficients (a,b,c,d) characterizing the seed plane, PL, are first computed. The set PLANE is then determined which consists of all unmarked points in POINTS "lying" on the plane PL. How close a point must be to the plane to be considered "lying on the plane" gives rise to a threshold which must be chosen. The set PLANE is then reduced to the maximal convex set in PLANE containing the seed points; this convex set is called FACE (see [4-36] for digital convexity). If the points in FACE constitute

a face which is not too narrow, then the information concerning the face is stored, and the points in POINTS belonging to FACE are marked. The predicate NARROW returns true if all the points in the face lie very close to the line (in 3-D space) which passes through the two most distant points in FACE. Such a set of points occurs when the seed points straddle an edge or vertex of the object. The perpendicular distance from a point m_0 to the line A_1A_2 in 3-D is obtained as follows (refer fig. 4.6).

Let $a = x_0-x_1$, $b = y_0-y_1$, $c = z_0-z_1$

$x = x_2-x_1$, $y = y_2-y_1$, $z = z_2-z_1$

Then, $\overrightarrow{m_0A_1} = (a,b,c)^T$

$$||\overrightarrow{m_0m}\times\overrightarrow{A_1A_2}||_2^2 = ||\overrightarrow{m_0A_1}\times\overrightarrow{A_1A_2}||_2^2 = ||\overrightarrow{m_0m}||_2^2 \, ||\overrightarrow{A_1A_2}||_2^2$$

but $\overrightarrow{m_0A_1} \times \overrightarrow{A_1A_2} = (bz-cy,\ cx-az,\ ay-bx)^T$ and $A_1A_2 = (x,y,z)^T$

$$\therefore ||m_0m||_2^2 = \frac{(bz-cy)^2 + (cx-az)^2 + (ay-bx)^2}{x^2+y^2+z^2}$$

$$\text{or,} \quad m_0m = \left(\frac{(bz-cy)^2 + (cx-az)^2 + (ay-bx)^2}{x^2+y^2+z^2}\right)^{1/2}$$

After the surface points belonging to a face have been obtained using the 3-point seed algorithm, two checks are made.

1. All the points which have been previously associated with various faces are checked for the

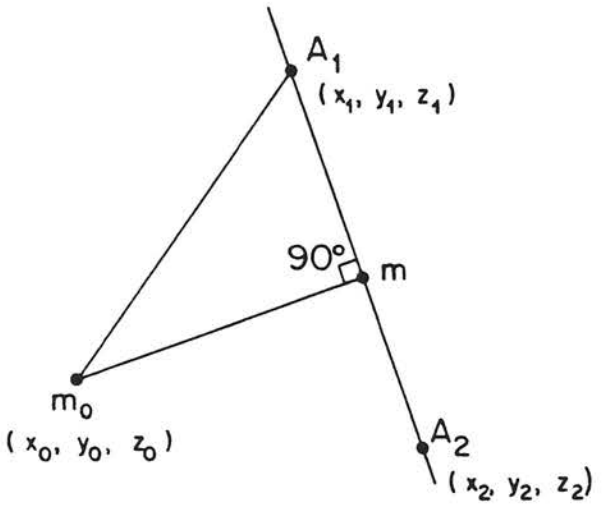


Fig. 4.6 Perpendicular distance from a point m_0 to the line A_1A_2 in 3-D

possible inclusion in the present face.

2. The set of points of the present face is checked for the possible inclusion in the previous faces.

The application of the above two tests provides the points which belong to more than one face. This information in turn provides the knowledge about the neighbors of a face and relations among them.

Now the surface points have been associated with various planar faces. Although some edge points and vertices will be known, an independent step is required to obtain polygonal faces. The polygonal approximation of a face involves the following steps.

1. Get the points belonging to a face.
2. Obtain the binary image of the face points.
3. Trace the boundary of the image obtained in step 2 using a boundary follower. Obtain (x,y,z) coordinates of the boundary points of the face.
4. Perform a polygonal approximation of the boundary of the face. Since the points associated with a face are in the same plane, we obtain projection of the points in the x-y plane by setting the corresponding z components of the points to be zero. The polygonal approximation is found by detecting the points of high curvature as in 2-D

case (see Chapter 2). Normally smoothing factor k is taken to be between 1/5 and 1/20 of the number of boundary points.

The complexity of the 3-point seed algorithm is $O(n^4)$. There are O(n) 3-point seeds to consider since each object point can be grouped in twelve ways with its neighbors to produce a seed. For each 3-point seed considered, the largest cost is in the convexity test which is $O(n^3)$ since every time a point is added to the FACE list, all rejected points in that plane have to be reconsidered.

Several important consequences follow from using convexity and narrowness tests to reject category (b) faces. Only convex faces are found for the object. This is not a serious problem since the information exists to merge the convex parts of non-convex faces. Namely, if the intersection of two faces contains sufficient number of points and the faces lie in the same plane, then those faces can be merged. Another problem is that the seed points are chosen sequentially, and this influences the selection of faces. However, if the interior of every face is represented by enough sample points, the major faces will always be found. Finally, the narrowness condition rejects small or narrow faces, i.e., a face one

or two points wide may be rejected. In the majority of cases, rejecting these kinds of faces is all right, and can always be overcome by increasing the sampling rate.

The 3-point seed algorithm can be viewed as a generalization of a method for segmenting one-dimensional curves (in any dimension). Choose two close points, X_0 and Y_0, on the curve. Let S = {X:X lies on $\overrightarrow{X_0Y_0}$ and X $\notin$ any segment already found}. Reduce S to the maximal convex set containing X_0 and Y_0. Mark all points in the resulting segment in S, and start all over. This algorithm does not take advantage of the ordering of points on the curve; such an ordering contains information of the adjacency of the points on the curve, and most piecewise linear approximation schemes do take advantage of the ordering [4-37, 4-38]. Thus, this method easily generalizes to two-dimensional surfaces, and is implemented here as the 3-point seed algorithm.

An alternate approach to finding planar faces could be a "clustering" type approach which may involve the following steps.

1. Find the reasonable planes.
2. Select the individual faces using connectivity.
3. Consider left over and boundary points etc.

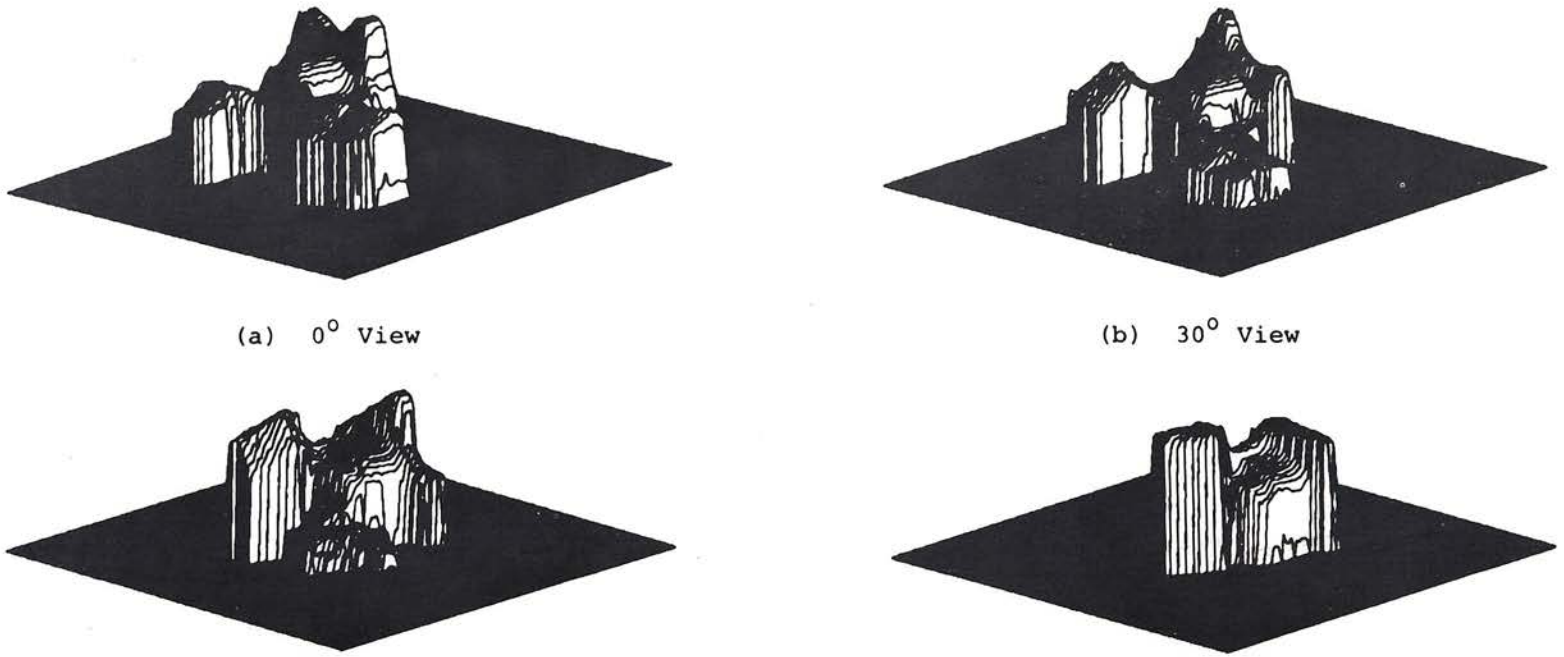
This approach has the advantage in that all the faces (convex or non-convex) are found at the same time. However, the connectivity used in step 2 will require an ordering of points and if the object does not contain major horizontal or vertical surfaces, step 1 based on Hough transform or obtaining the histogram of z distance or some other local features may be quite expensive.

4.5 Experiments and Discussion

Experiments

Both synthesized objects and an industrial piece used in an automible suspension system have been analyzed with the proposed method for surface approximation by polygons. Results obtained on synthetic cubes were the faces of the cubes exactly, and faces found on synthetic spheres were reasonable. A more difficult case is that of the complicated casting of an automobile piece shown in fig. 4.7. This object does not contain any major horizontal or vertical surface. The 14 views obtained with the range data acquisition system described in section 4.3 are shown in fig. 4.8 and fig. 4.9 shows these range views as gray scale images. In this figure lighter points are farther away from the observer and the darker ones are closer. After thresholding the background points, each individual view shown in figs. 4.8 and 4.9

Fig. 4.7 Automobile piece analyzed



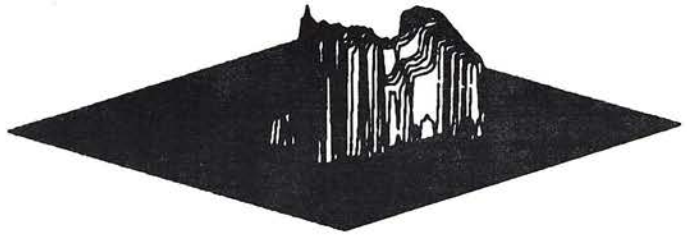
(a) 0^{o} View

(b) 30^{o} View

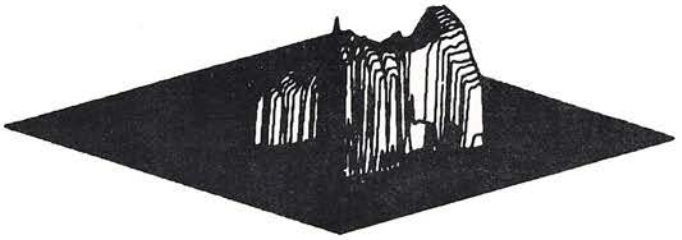
(c) 60^{o} View

(d) 90^{o} View

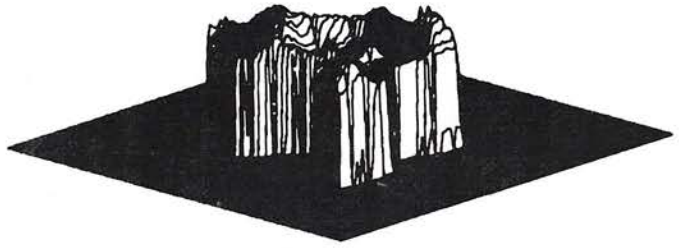
Fig. 4.8 The 14 range data views of the object; (a) - (l) correspond to orientations 0^{o} to 330^{o}; (m) is the top view and (n) is the bottom view



(e) 120° View

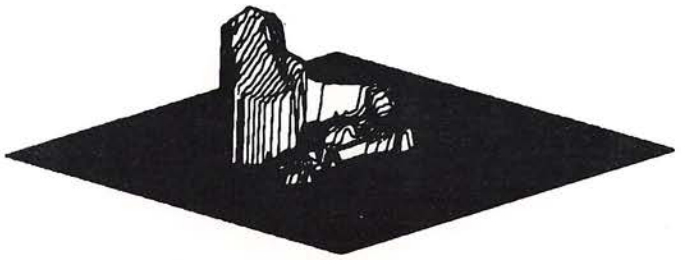


(f) 150° View

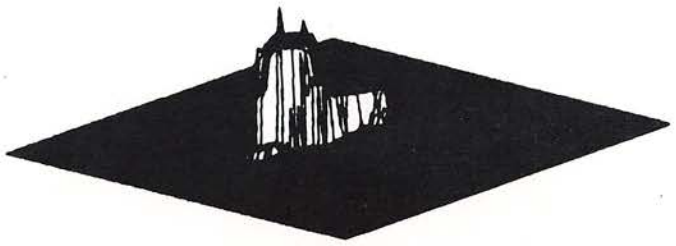


(g) 180° View

(h) 210° View

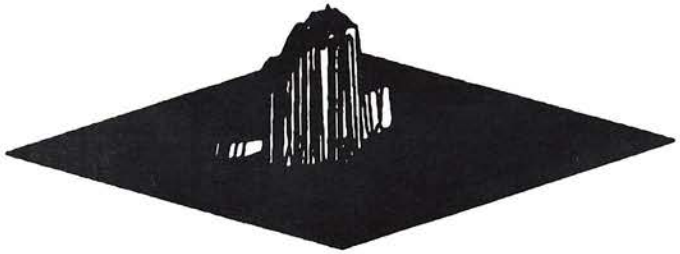


(i) 240° View

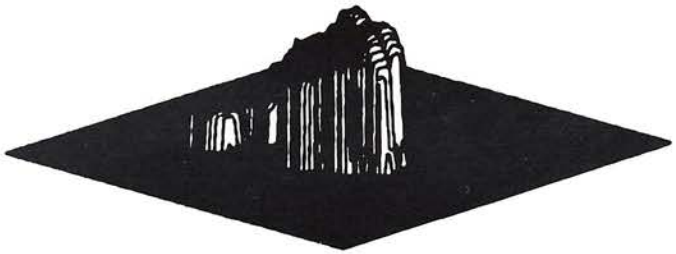


(j) 270° View

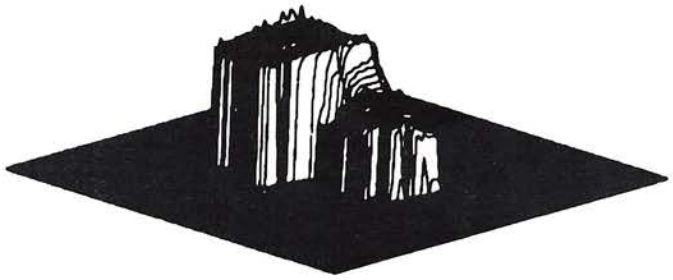
Fig. 4.8 (CONTINUED)



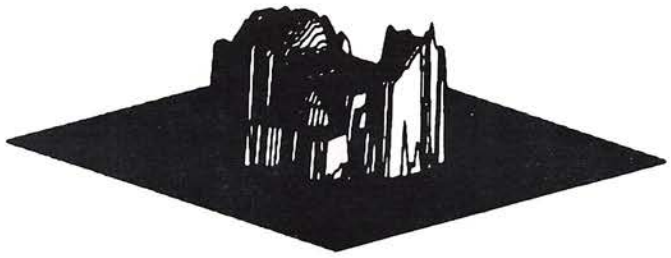
(k) 300° View



(l) 330° View



(m) Bottom View



(n) Top View

Fig. 4.8 (CONTINUED)

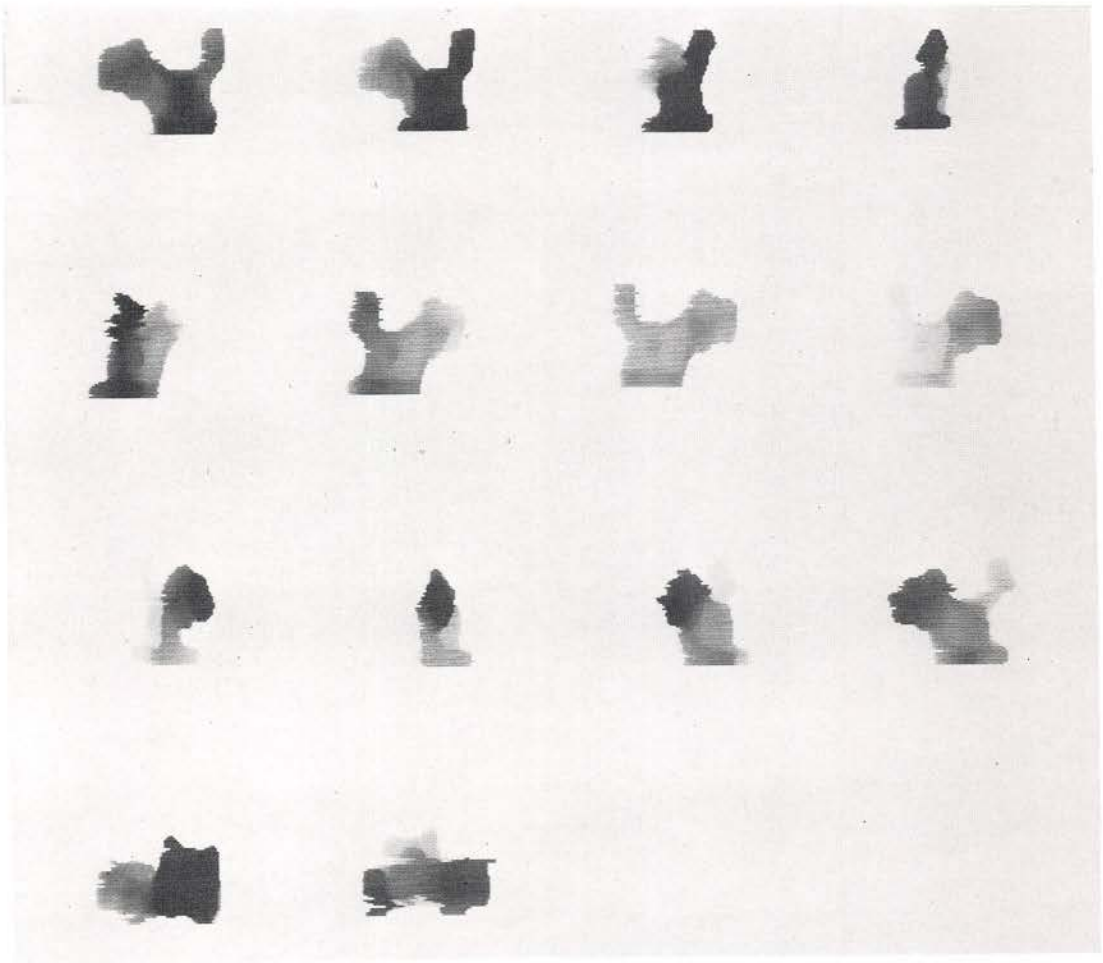


Fig. 4.9 The 14 range data views of the object shown as gray scale images. The lighter points are away from the observer and the darker ones are closer

had approximately 2000 points. Surface points for the composite object were obtained using the following steps.

1) 12 views (fig. 4.8(a) to 4.8(l)) are obtained by rotation in the x-z plane. Thus, knowing the rotation, we can transform these views to the original coordinate system (object centered).

$$x = x'\cos\theta - z'\sin\theta$$

$$y = y'$$

$$z = x'\sin\theta + z'\cos\theta$$

where x', y' and z' are the coordinates of a point after rotation of angle θ and x, y and z are the coordinates with respect to the original coordinate system.

For the bottom and top views (figs. 4.8(m) and 4.8(n)) we had 3 control points in each of these views which were also visible in the 0° view (fig. 4.8(a)) (requiring 6 control points in the 0° view). We can compute the transformation matrix T, where

$$T\underline{x} = \underline{b}$$

or,

$$\begin{bmatrix} \ell_1 & \ell_2 & \ell_3 \\ m_1 & m_2 & m_3 \\ n_1 & n_2 & n_3 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

and (l_1, m_1, n_1), (l_2, m_2, n_2) and (l_3, m_3, n_3) are the direction cosines of the x', y', z' axis relative to (x,y,z) coordinates respectively. Transformation T is obtained by solving the following set of equations.

$$\begin{bmatrix} x'_1 & y'_1 & z'_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ x'_2 & y'_2 & z'_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ x'_3 & y'_3 & z'_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & x'_1 & y'_1 & z'_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x'_2 & y'_2 & z'_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & x'_3 & y'_3 & z'_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x'_1 & y'_1 & z'_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & x'_2 & y'_2 & z'_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & x'_3 & y'_3 & y'_3 \end{bmatrix} \begin{bmatrix} \ell_1 \\ \ell_2 \\ \ell_3 \\ m_1 \\ m_2 \\ m_3 \\ n_1 \\ n_2 \\ n_3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ y_1 \\ y_2 \\ y_3 \\ z_1 \\ z_2 \\ z_3 \end{bmatrix}$$

(x_1, y_1, z_1), (x_2, y_2, z_2) and (x_3, y_3, z_3) are the coordinates of the reference points in the 0° view and (x'_1, y'_1, z'_1), (x'_2, y'_2, z'_2) and (x'_3, y'_3, z'_3) are the coordinates of the same reference points in the top or bottom view. Since these points are different and noncoplanar, a solution exists which is unique for the above set of equations.

Thus using the transformations as described above, we

can transform surface points in any of the views with respect to the original coordinate system.

2) Now starting from the 30° view to the top view (fig. 4.8(b) to 4.8(n)), we apply the required transformation as discussed in 1 and compute the distance between the transformed coordinates and the points which are already in the list (in the beginning just the 0° view points). If the difference is less than a certain threshold related to the sampling distance, we discard this point, otherwise this point is added to the list. Sampling distances were 3 mm and 2 mm in the x and y direction for the automobile piece and using a distance threshold of $\sqrt{15}$, the composite object has 8314 points which are stored as a list.

The 3-point seed method was applied to the 14 individual views shown in fig. 4.8 and to the composite object. Fig. 4.10 shows the faces found for 0° view, (fig. 4.8(a)) of the object. In this figure various faces are shown in different colors. The rejected points and the points common to two or more faces (edge points) are shown in blue and khaki color respectively. The points that could not make up a face having at least 20 points were rejected. The area of rejected points fall either on jump points resulting from large z-distance change with

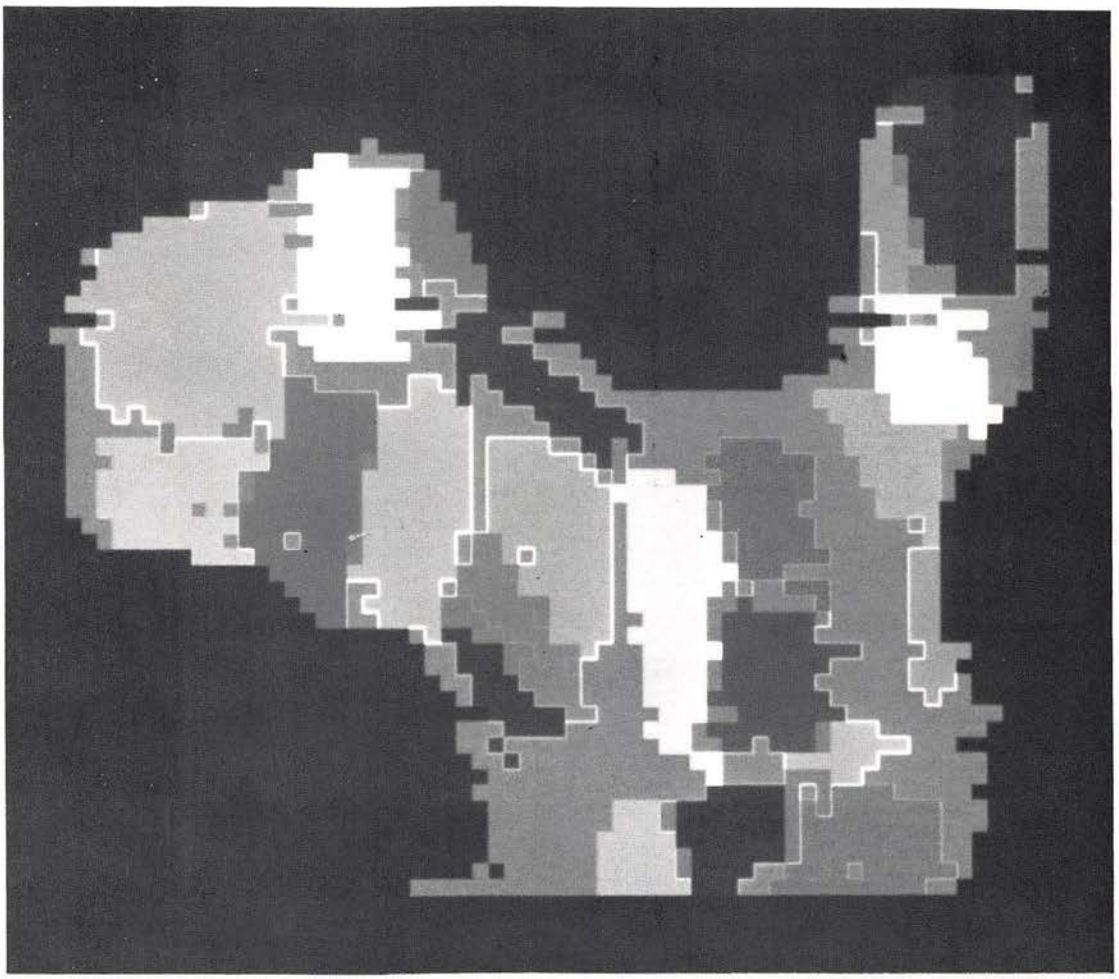


Fig. 4.10 Faces found in the 0^{o} view (fig. 4.8(a)). There are 22 faces in this view. The rejected points and the points common to two or more faces are shown in blue and khaki color respectively

correspondingly little x or y change, or they occur in extremely uneven parts of the surface of the object. A rejected point lies inside some of the faces because it has been missed in the process of data acquisition. Also some of the rows have been shifted because of the continuous nature of the data. As can be seen, most of the faces found are reasonable. The object has major curved surfaces that were split symmetrically into different faces. Note that the edge points are not very well detected. This result is directly related to the choice of the point-plane distance allowed and the sampling distances. It is clear that actual edge points will probably be missed during the sampling of the surface. This means that if the point-plane threshold is chosen low, then two adjacent faces will have no points in common as only interior points were sampled on the object's surface.

Table 4.1 gives the properties of faces in the 0^{o} view. I1, I2 and I3 are the indices in the list of points for this view which make up a face. Fig. 4.11 shows the faces obtained in the 90^{o} view (fig. 4.8(d)) and table 4.2 list the properties of these faces. Results similar to the 0^{o} view are obtained. Table 4.3 shows the neighbors of a face in the 0^{o} and 90^{o} views. These neighbors are arranged in the descending order of the number of points

TABLE 4.1 List of faces in the 0^{o} view (fig. 4.8(a)). I1, I2 and I3 are the indices in the list (of point for this view) which make up a face

EPS1	$=\sqrt{19}$	EPS2	= 3	EPS3	$=\sqrt{9.5}$
Δx	= 3	Δy	= 2	Δz	= 3
MAXPTS	= 80	DECREMENT	= 20	MINPTS	= 20 N = 1936

FACE	I1	I2	I3	TOTAL NO. OF POINTS
1	2	3	11	103
2	69	89	90	84
3	140	141	170	176
4	565	625	626	93
5	573	574	634	94
6	797	798	856	84
7	904	960	961	105
8	782	839	897	64
9	816	817	875	70
10	1310	1350	1391	83
11	1718	1719	1751	67
12	328	329	367	52
13	338	376	377	60
14	597	657	658	43
15	1139	1193	1241	50
16	1725	1758	1759	49
17	38	49	50	35
18	83	84	105	30
19	1165	1166	1213	23
20	1409	1443	1444	36
21	1589	1590	1621	24
22	1766	1767	1799	31

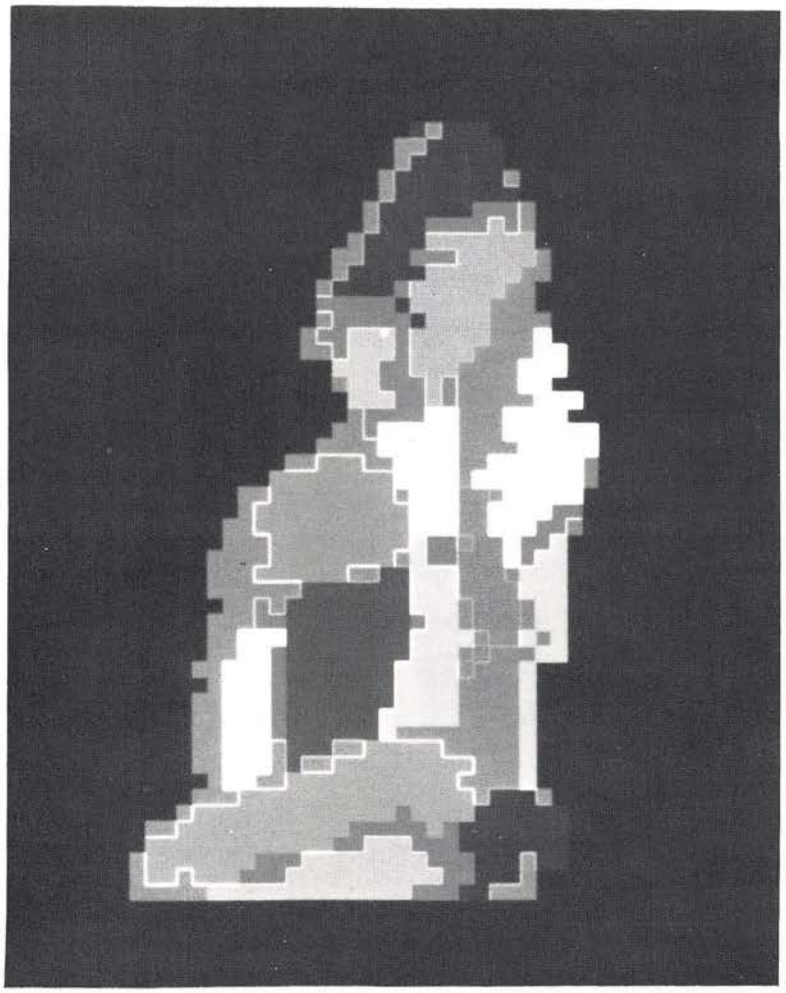


Fig. 4.11 Faces found in the 90^{o} view (fig. 4.8(d)). There are 14 faces in this view. The rejected points and the points common to two or more faces are shown in blue and khaki color respectively

TABLE 4.2 List of faces in the 90^{o} view (fig. 4.9(d)). I1, I2 and I3 are the indices in the list (of points for this view) which make up a face

EPS1 $=\sqrt{19}$ EPS2 $= 3$ EPS3 $=\sqrt{9.5}$
$\Delta x = 3$ $\Delta y = 2$ $\Delta z = 3$
MAXPTS = 80 DECREMENT = 20 MINPTS = 20 N = 914

FACE	I1	I2	I3	TOTAL NO.
1	1	6	14	82
2	432	455	456	88
3	674	695	718	111
4	143	159	177	60
5	284	303	324	62
6	51	61	75	72
7	385	386	408	43
8	734	757	758	43
9	152	153	170	38
10	227	228	244	42
11	316	317	338	28
12	377	399	400	32
13	531	555	576	37
14	818	819	845	58

TABLE 4.3 Neighbors of a face in 0^{o} and 90^{o} views. These neighbors are arranged in the descending order of their size
(a) 0^{o} view (fig. 4.8(a))
(b) 90^{o} view (fig. 4.8(d))

FACE NUMBER	NEIGHBORS		
1	12	17	0
2	3	13	18
3	2	9	0
4	5	0	0
5	4	9	0
6	15	0	0
7	10	8	13
8	7	10	0
9	3	5	0
10	7	8	21
11	16	21	0
12	1	17	0
13	7	2	18
14	19	0	0
15	6	20	0
16	11	22	21
17	1	12	0
18	2	13	0
19	14	0	0
20	15	0	0
21	10	11	16
22	16	0	0

(a)

FACE NUMBER	NEIGHBORS		
1	6	9	0
2	7	13	0
3	14	8	0
4	12	0	0
5	0	0	0
6	1	10	9
7	2	10	0
8	3	14	0
9	1	6	10
10	6	7	9
11	0	0	0
12	4	0	0
13	2	0	0
14	3	8	0

(b)

that they possess. Note that a face may have no neighbors, because a face that could not possess more than a certain minimum number of points was rejected. Different faces have different numbers of neighbors. For example face 1 in 0^{o} view has face 12 and face 17 as neighbors, and face 11 in 90^{o} view has no neighbors. The method was applied to the composite object to get the complete 3-D model. In the model 85 faces were found. The number of faces found, and their distribution fits well with the results from the individual views.

Discussion

The 3-point seed algorithm alludes obliquely to several thresholds which must be chosen carefully. We are interested in obtaining the theoretical bounds for these thresholds. In general, these thresholds are directly related to the sampling distances. First, three points are close enough together to be chosen as seed points if each pair of the three are within $(\Delta x^2+\Delta y^2)^{1/2}$, where Δx and Δy are the distances between samples in the x and y direction, respectively. Call this distance EPS1. The second threshold, call it EPS2, to be chosen is the distance from the plane a point can have and still be considered in the plane. This threshold should be low enough so that the number of points is low for the

convexity check. However, when EPS2 is too low, neighboring faces will most likely have no common points, making the relations between faces harder to find. The third threshold, EPS3, is that used to determine face convexity. The check for convexity involves finding the midpoint of the candidate point and each point already in the face (initially just the seed points). If for some point of the face, the midpoint is not close enough to any point already in the face, then the candidate point is rejected. Of course, every time a new face point is found, all rejected points must be reconsidered. The threshold EPS3 cannot be chosen smaller than $1/2\ (\Delta x^2+\Delta y^2)^{1/2}$ or else no set of points is convex.

Finally, narrowness is decided by finding the line passing through the two most distant points in the set and checking if some point in the set lies far enough away from this line. The threshold EPS4 for this distance determines how narrow a face can be. For the automobile piece, $\Delta x=3$, $\Delta y=2$, $\Delta z=3$, and the thresholds actually used were EPS1 = $\sqrt{19}$, EPS2 = 3 and EPS3 = $\sqrt{9.5}$. Δz is chosen as 3 since this is the maximum sampling rate that the z-axis undergoes (this occurs at 90° from the original 0° view). The threshold EPS4, however, is dependent on the orientation of the line in space between the two most distant points; in particular, EPS4 = $\Delta x\cos\alpha + \Delta y\cos\beta$

+ $\Delta z \cos\gamma$, where $\cos\alpha$, $\cos\beta$ and $\cos\gamma$ are the direction cosines of the line.

The method is applied in stages; the largest faces (in terms of the number of points in the face) are found first, then smaller faces on down to some minimum size. In Tables 4.1 and 4.2, in the beginning we find faces which have at least MAXPTS number of points and iteratively decrease it by an amount of DECREMENT until we reach MINPTS. The faces having points less than MINPTS are rejected. N is the number of point in the file of a view after the background points have been removed. The application of the method in stages is necessary in order to limit the fragmentation of large faces near their extremes. In any case, the unevenness of the surface of the object can still lead to poor face selection; however, the peculiarities of the object to be modeled can be accounted for by the proper choice of thresholds and the tradeoff involved between the number of faces and the quality of representation can be balanced. Although the method is rather slow, techniques exist for achieving greater speed. For example, a "plane detector" i.e. a filter with high response where the data fits a plane well, can be passed over a range data image to produce the most likely locations for seed points. Also once the neighbors of a face are known, the neighboring faces could

be tested for merger and the more refined face boundaries can be obtained by finding the intersection of planes that make up the neighboring faces.

4.6 Summary

In this chapter we presented some of the issues related to three-dimensional scene analysis. In particular modelling, representation and data acquisition methods were described. A new technique for the extraction of planar faces from a set of 3-D points is proposed. This method is of time complexity $O(n^4)$ and sequential in nature. The underlying physical motivation for the method is given. The application of the method to synthetic and real world 3-D objects is described. The major advantage of the technique is that it is applicable to a composite object, and not restricted to single view range data images. Information on neighboring faces is also suitably obtained. Once the planar faces are found, they are approximated by polygons.

In the next chapter we shall use the hierarchical stochastic labeling technique used in 2-D to do shape matching of real world 3-D objects based on matching the faces of the unknown view against the faces of the model.

CHAPTER 5

SHAPE MATCHING OF THREE-DIMENSIONAL OBJECTS

5.1 Introduction

In the last chapter we presented a technique for approximating the surface of a three-dimensional object by a set of planar polygonal faces. This technique is used to generate the 3-D model and obtain the description of an unknown view of a three-dimensional object. The objective is to match individual views of the object (taken from any vantage point) against the model. The results of matching can be used to determine the orientation of the object in three-space. In Chapter 2 we described the basic hierarchical stochastic labeling method to do shape matching of two-dimensional objects. The method is based on "segment matching" i.e., a piece of a 2-D shape is recognized as an approximate match to a part of a larger 2-D shape. In this chapter we present an extension of this method to do shape matching of 3-D objects. The matching is called "face matching" i.e., a partial 3-D shape is recognized as an approximate match to a part of a larger 3-D shape. It is based on matching the faces of

the unknown view against the faces of the model. In this chapter first we present the relevant work in 3-D scene matching in this section, and discuss the shape matching algorithm with methods for computing initial likelihoods and compatibilities in section 5.2. Shape matching results on the complex part of an automobile used in the last chapter are presented in section 5.3 and finally section 5.4 summarizes the chapter.

Most of the work in scene analysis has been the interpretation of a two-dimensional image in terms of a configuration of three-dimensional objects by making use of the a priori information about the objects. It has some inherent problems in that the image of a three-dimensional object changes with the perspective (viewing angle), it is sensitive to shadows, time of day, weather conditions and specularity; when several objects occlude each other, only parts of some objects are visible in the image and the occluding objects need to be separated from each other. Furthermore, a non-convex object can partially occlude itself. Now with the availability of three-dimensional sensing devices, the area of recognition of complex objects using 3-D models can be explored and thus overcoming some of the limitations of 2-D scene analysis. 3-D scene analysis is very important for computer vision systems to be used in

the industrial environment [5-1]. In 3-D scene analysis we have a model for 3-D objects and a method for matching unknown objects with the model. Milgram and Bjorklund [5-2] mention preliminary efforts of three-dimensional matching by pairing the list of planes of the reference model and the sensed image using a guided search procedure. The number of flat surfaces (roof, walls, etc.) in their study is usually small (less than 50).

The use of Fourier descriptors and moments has not been limited to the two-dimensional shape analysis. They have also been used for the recognition of three dimensional shapes [5-3 to 5-6]. Wallace et al. [5-4] use normalized Fourier descriptors for the recognition of three-dimensional aircraft. Dudani et al. [5-5] use moments for the purpose of aircraft identification. Wong and Hall [5-3] use moments for scene matching of two-dimensional objects. However, moments or Fourier descriptors are global features and cannot solve the important class of problems which require the partial recognition of the shape. This is because the descriptors of the entire shape do not bear any simple relationship with the descriptors of a part of a shape. Wallace et al. [5-4] also consider shape analysis of three-dimensional objects (aircraft synthesised using a graphics program) using local shape descriptors by combining statistical and

structural information. Their matching approach is not a parallel iterative scheme, but involves several steps and the alignment of the two local feature vectors can be very expensive. In order to handle the partial shape recognition problem, their techniques need major modifications. Furthermore like Dudani et al. [5-5] these authors are not dealing with the three-dimensional data, but rather with projections of a 3-D object. Since the image of a three-dimensional object changes with the viewing angle, they have a large library of 3 dimensional projections corresponding to a single object. For example Dudani et al. [5-5] in the identification of six different aircraft use a training sample set of over 3000 projected images. Sadjadi and Hall [5-6] present an extension of two-dimensional moment invariants [5-7] to three-dimensions. However, these moments are again global features and cannot solve the partial shape recognition problem.

In section 4.3 we presented a discussion and our approach to representation and modelling of three-dimensional objects. Representation and modelling are closely related and the control structures normally depend on the choice of representation. Control structures are defined as the strategy of utilizing the available knowledge to efficiently obtain the goal

descriptions. A literature survey of 3-D scene analysis work reveals that bottom up, top down and a mixture of these two have been used [5-8 to 5-10]. The hierarchical control structure is a popular choice since it eliminates unnecessary search during the recognition process. Agin [5-11], Binford et al. [5-12] and Nishihara [5-13] use a hierarchical structure for 3-D analysis. Reviews of the recent research works can be found in [5-8 to 5-10]. Recently, Shani [5-14] uses hierarchical top down approach directed by a model based on generalised cylinders for the recognition of abdominal anatomy from computer tomograms.

Our approach for 3-D shape matching uses planar faces as primitives and matches an unknown view with the structural 3-D model. The control structure of the 3-D shape matching algorithm is hierarchical in the sense that at higher levels of hierarchy more contextual information and world knowledge is used to accomplish the partial shape matching task. In the next section we present an extension of the hierarchical shape matching algorithm described in Chapter 2 to perform shape matching of three dimensional objects. Results of using this technique on a complex automobile part will be presented in section 5.3.

5.2 Shape Matching Algorithm

Fig. 5.1 shows a block diagram of the two stage hierarchical stochastic labeling technique for the shape matching of three-dimensional objects. Shape matching is performed by matching the face description of an unknown view with the stored model using the available contextual information. The same set of descriptors is used for the description of both the faces of the model and an unknown view. Input to the stochastic labeling process (also called the gradient relaxation algorithm) includes,

1) Features of the faces.
2) Information about the neighbors of a face.
3) Equation of the plane describing a face.

Let $T = (T_1, T_2, \ldots, T_N)$ and $O = (O_1, O_2, \ldots, O_{L-1})$ be the face representation of an unknown view and the model respectively, where T_i and O_j are planar faces, $i = 1, \ldots, N$ and $j = 1, \ldots, L-1$. The elements of the unknown view will be referred to as units and elements of the model as classes. We want to accomplish partial shape recognition i.e., identify an unknown view within the model. We are trying to label each of the faces of an unknown view T_i $(i = 1, \ldots, N)$ either as a face O_j $(j = 1, \ldots, L-1)$ or as not belonging to the model O (label O_L = Nil). Each face T_i of the unknown view therefore has

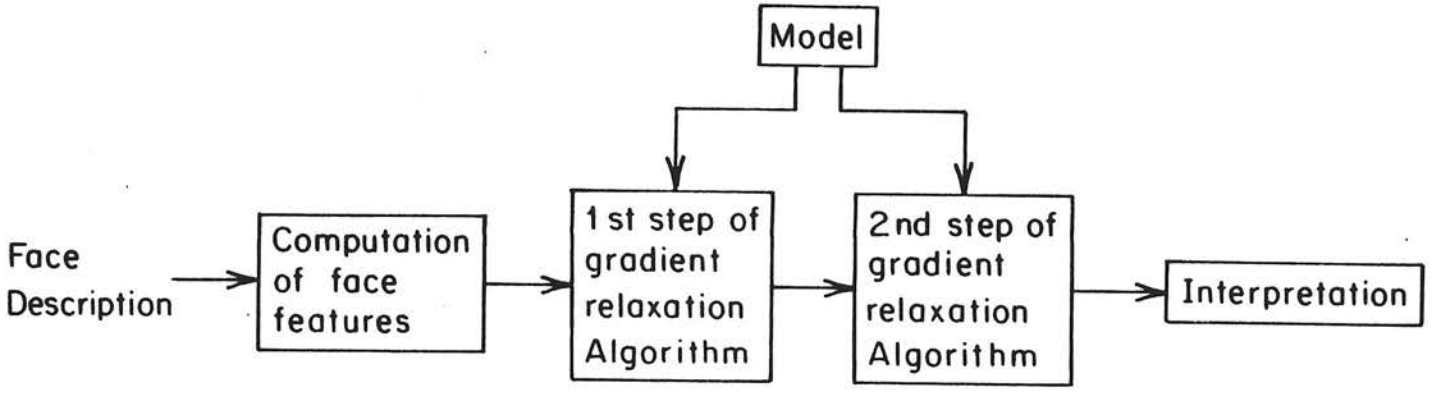


Fig. 5.1 Block Diagram of the 3-D Shape Matching Algorithm

L possible labels.

To each of the units T_i, we assign a probability $p_i(\ell)$, $\ell = 1,\ldots,L$ (using a technique described subsequently in this section) that the unit belongs to class O_k. This is conveniently represented as a probability vector $\vec{p}_i = [p_i(1),\ldots,p_i(L)]^T$. The set of all vectors $\vec{p}_i$ $(i = 1,\ldots,N)$ is called a stochastic labeling of the set of units. Units are related to one another through their neighbors. The set of units related to T_i is denoted by V_i. In order to compare the local structure of T and O, the world model is specified by the compatibility functions C_1 and C_2, which are defined over a subset $S_1 \subseteq (N\times L)^2$ and $S_2 \subseteq (N\times L)^3$ for the first and second stage of the hierarchy respectively. At the first stage $C_1(T_i, O_k, T_j, O_\ell)$ measures the adequacy of calling unit T_i as O_k and unit T_j as O_ℓ where $T_j \in V_i$. Similarly at the second stage $C_2(T_i, O_k, T_{i_1}, O_{\ell_1}, T_{i_2}, O_{\ell_2})$ measures the adequacy of calling unit T_i as O_k, unit T_{i_1} as O_{ℓ_1} and T_{i_2} as O_{ℓ_2} where T_{i_1} and $T_{i_2} \in V_i$. We also define a compatibility vector $\vec{q}_i = [q_i(1),\ldots,q_i(L)]^T$ for all the units at each of the labels of hierarchy. Intuitively this tells us what $\vec{p}_i$ should be given $\vec{p}_j$ at the related units and the compatibility function. For simplicity we shall denote $C_1(T_i, O_k, T_j, O_\ell)$ and $C_2(T_i, O_k, T_{i_1}, O_{\ell_1}, T_{i_2}, O_{\ell_2})$ as $C_1(i,k,j,\ell)$ and

$C_2(i,k,i_1,\ell_1,i_2,\ell_2)$ respectively. Mathematically,

$$q_i^{(j)}(k) = \frac{Q_i^{(j)}(k)}{\sum_{\ell=1}^{L} Q_i^{(j)}(\ell)} \quad , \quad \begin{matrix} j = 1,2 \\ k = 1,\ldots,L \end{matrix} \tag{5-1}$$

where, at the first stage,

$$Q_i^{(1)}(k) = \sum_{j \in V_i} \sum_{\ell=1}^{L} C_1(i,k,j,\ell) p_j(\ell) \quad \begin{matrix} i = 1,\ldots,N \\ k = 1,\ldots,L \end{matrix} \tag{5-2}$$

and at the second stage,

$$Q_i^{(2)}(k) = \sum_{\ell_1,\ell_2=1}^{L} C_2(i,k,i_1,\ell_1,i_2,\ell_2) p_{i_1}(\ell_1) p_{i_2}(\ell_2) \tag{5-3}$$

$$i = 1,\ldots,N$$

$$k = 1,\ldots,L \text{ and } i_1,i_2 \in V_i$$

As discussed in Chapter 2, the global criteria that measure the consistency and ambiguity of the labeling over the set of units are given by,

$$J^{(j)} = \sum_{i=1}^{N} \vec{p}_i \cdot \vec{q}_i^{(j)}, \quad j = 1,2 \tag{5-4}$$

The maximization of the criteria is done using the gradient projection method as described in Chapter 2. Using the initial probability assignment $\vec{p}_i(0)$

$(i = 1,\ldots,N)$ first we maximize $J^{(1)}$ and then use these results to maximize $J^{(2)}$. Equations (2-13) to (2-25) needed in the maximization scheme are also applicable here. Now in Eq. (2-13), $j \in V_i$ and $i_1, i_2 \in V_i$, $i_3 \in V_{i_1}$ and $i_4 \in V_{i_2}$ in eqs. (2-15) to (2-21).

Initial Assignment of Probabilities

The descriptors of a face are,

1) Area, the number of points belonging to a face,

2) Perimeter, the number of points on the boundary of a face,

3) Length of the maximum, minimum and average radius vectors from the centroid of a face,

4) Number of vertices in the polygonal approximation of the boundary of a face,

5) Angle between the maximum and minimum radius vectors,

6) Ratio of Area/Perimeter^2 of a face,

7) (x,y,z) centroid of a face,

8) Equation of the plane characterizing a face,

9) Neighbors of a face.

The initial probabilities are computed using the features 1) to 6) in the above list. Let P be the number of features used. We measure the quality of correspondence between the faces T_i and O_k as

$$M(T_i,O_k) = \sum_{p=1}^{P} |f_{tp} - f_{op}|W_p \qquad (5\text{-}5)$$

where,

f_{tp} = pth feature value for the face of an unknown view

f_{op} = pth feature value for the face of the model

W_p = weight factor for the pth feature

We need the weights of the features to account for their importance and range of values. Also any a priori knowledge that we may have regarding the type of change or deformation can be incorporated into these weights. The initial probabilities are chosen proportional to $1/(1+M(T_i,O_k))$ and normalized so that they sum to 1. Thus the initial assignment of probabilities involve unary relations.

Computation of Compatibilities

The compatibility function determines the degree by which two or three neighboring units are compatible with each other. At the first stage the computation of $C(i,k,j,\ell)$ involves binary relations and at the second stage $C(i,k,i_1,\ell_1,i_2,\ell_2)$ involves a subset of ternary relations. The compatibility of a face of an unknown view with a face in the model is obtained by finding transformations, applying them and computing the error in feature values. The method used here is similar to the third method used to compute compatibilities in 2-D case described in Chapter 2.

At the first stage, we find two transformations TR1 and TR2 such that

$$\text{TR1: } T_i \rightarrow O_k$$

and $$\text{TR2: } T_j \rightarrow O_\ell$$

Now TR1 is applied to T_j giving matching error $M(TR1(T_j),O_\ell)$ and TR2 is applied to T_i giving matching error $M(TR2(T_i),O_k)$, where matching error is given by,

$$M(TR(T_m),O_n) = \sum_{p=1}^{P} |f_{t'p} - f_{op}| W_p \qquad (5\text{-}6)$$

where $f_{t'p}$ = pth feature value for the transformed unit,

and other quantities are similar to those defined in (5-5). Features used in computing (5-6) are (x,y,z) centroid, area, orientation and rotation.

The average of these two errors is obtained and

$$C_1(i,k,j,\ell) = \frac{1}{1 + \text{average error}}$$

At the second stage instead of finding two transformations, we find three transformations and the average error will be the average of six error terms and the compatibility

$$C_2(i,k,i_1,\ell_1,i_2,\ell_2) = \frac{1}{1 + \text{average error}}$$

The transformations used in computing C_1 and C_2 are based on,

1) Scale, the ratio of area of two faces.

2) Translation, difference in the centroidal coordinates of the two faces.

3) Orientation, difference in the orientation of two faces so that they are in the same plane.

4) Rotation, to obtain maximum area of intercept, once the two faces are in the same plane.

The orientation information is obtained by finding the angle between the planes describing the two faces. This angle is equal to the angle between their normals. It could be θ or $180-\theta$. In order to determine it uniquely, we always take constant term D in the equation of the plane $Ax + By + Cz + D = 0$, to be negative.

The operation of finding the rotation (in the x-y plane) of a model face with respect to the face of an unknown view involves the following steps:

(a) Get the image boundary of the model and unknown view faces.

(b) Fill the region within the boundary of the faces. Now they are called as the image of the model face and the unknown view face.

(c) Obtain the (x-y) centroid of the images obtained in (b).

(d) Shift the centroid of the model image obtained in (c) so that it coincides with the centroid of the unknown view image.

(e) Rotate the model image and count the points that overlap the unknown image.

(f) Repeat (e) for rotations of 0°, 90°, 180° and 270°.

(g) The maximum count corresponding to a particular rotation is the desired angle of rotation.

The problem of defining p_i(nil) or C_1 and C_2 when some of the faces in the unknown view are matched to the nil class is solved in the second chapter.

Data Structure

As described in Chapter 4, while obtaining the polygonal approximation of the boundary of a face, we get the image of the points belonging to a face. It is at this time we compute all the features of a face and store them in a two-dimensional array. Also the boundary of the image of all the faces is stored in two files, one for the model and the other for an unknown view. This is needed in the computation of compatibilities. Information about the neighbors of a face and the plane coefficients (A, B, C, D) describing a face are stored in 2-D arrays. The basic descriptors of a typical face are as follows-

Face number 27

Area = 61, Perimeter = 31, No. of vertices = 23, Area/Perimeter = 0.634, Radmax = 16.06, Radmin = 3.21, Radavg = 10.46, Angle between Radmax & Radmin = 95.05 degrees, Xcent = 23.4, Ycent = 157.4, Zcent = 3.5.

5.3 Examples and Comments

In this section we present examples of the partial shape recognition of a three-dimensional industrial object using the hierarchical stochastic labeling described in the last section. The results of labeling are used to compute the relative orientation of an unknown view with respect to the model. We are seeking only the approximate matches as the exact matches may not exist when we deal with the real data.

In section 2.4 we provided a discussion of various parameters used in the stochastic labeling process. The discussion on the parameters used to control the stochastic labeling process, number of neighboring labels, value of α and the nil class value is also applicable here. Some differences are in the requirements of weights needed in the computation of compatibilities and initial probabilities and the number of neighbors used in the computation of compatibility vector $q_i^{(j)}$ $(j = 1,2)$.

Weights of Features

We require the weights of features in the initial assignment of probabilities and computation of compatibilities to account for their importance and range of values (in chapter 2 and 3 we needed weights only in

the initial assignment of probabilities). Angle between the maximum and minimum radius vectors and the angle of orientation vary between 0° and 180°. The angle of rotation can take values of 0°, 90°, 180° and 270°. The initial assignment of probabilities requires the weights for area, perimeter, number of vertices in the polygonal approximation, length of radius vectors, area/perimeter^2 and the angle between maximum and minimum radius vectors. The computation of compatibilities requires the weights of length, size, orientation and rotation angle. The weights for length and size are taken equal to the weights of perimeter and area used in the initial assignment of probabilities.

Number of Neighbors in the Computation of Compatibility Vector

The evaluation of the compatibility vector $q_i^{(j)}$ $(j = 1,2)$ requires the knowledge about the neighbors of a face (see eq. 5-1). In Chapter 4 we observe that a face may have no neighbors, it may have several neighbors and in general different faces have different number of neighbors. However, most of the units have two neighbors. We arrange the neighbors of a unit in the descending order of their size. The larger neighbors are given preference over the smaller neighbors, when a unit has several

neighbors and we consider only a subset of them in the computation of compatibility vector. Normally we have considered the number of neighbors to be 1 or 2 in the computation of $q_i^{(1)}$ and 2 neighbors in the computation of $q_i^{(2)}$ for all the units. It is possible to select two neighbors in the computation of $q_i^{(2)}$ in a number of different ways. For example, if a unit has 4 neighbors, then we may select two neighbors in 6 different ways. In this work we have selected the largest two neighbors of a unit to obtain $q_i^{(2)}$. If a unit has only one neighbor while computing $q_i^{(2)}$, then compatibility C_1 is used instead of C_2 in its computation. Considering more than one neighbor in the computation of $q_i^{(1)}$ requires more computation time, but the results are improved. In the computation of $q_i^{(2)}$ considering all the possible combinations of neighbors will be very expensive. Therefore, we have used only one pair of neighbors. If a unit has no neighbors, then this unit is firmly assigned to the best matched class at the time of computation of initial probabilities. The number of neighbors used at the first and second stage are shown as a two-tuple in the tables corresponding to various examples.

Examples

We present examples using the industrial object shown in fig. 4.7. As described in chapter 4, the number of faces in an unknown view of this automobile piece varied from 10 to 25 and the number of faces in the model is 85. In matching only the best 29 faces of the model are considered in order to reduce the complexity of the matching task. The shape matching algorithm uses various strategies described in chapter 2, which lead to faster computation. In testing the shape matching algorithm we consider 3 unknown views shown in figs. 4.8(a), 4.8(b) and 4.8(1) corresponding to 0°, 30°, and 330° respectively. Although the model is obtained using these views (see section 4.5), the model does not contain all the faces corresponding to any unknown view. This is because of the nature of how the surface points corresponding to the composite object were obtained. Therefore, the use of these unknown views is justified for the evaluation of the shape matching technique. The inverse of weights of the area, perimeter, number of vertices in the polygonal approximation, length of radius vectors, ratio area/perimeter^2, angle between the maximum and minimum radius vectors, orientation and rotation angles will be denoted as a 8-tuple in the tables associated with various examples.

Example 5.1

Fig. 5.2 shows the faces found in the view shown in fig. 4.8(a). There are 22 faces in this view and they are labeled in the order they are found using the algorithm described in chapter 4. The rejected points and the points common to two or faces are shown in brown and white respectively. Table 5.1 shows the neighbors of the faces of fig. 5.2. The neighbors are arranged by size in descending order.

We want to identify the shape of the unknown view within the model. Table 5.2 shows the results of labeling at different iterations. The various parameters used in the hierarchical stochastic labeling technique are also shown in this table. Note that the criteria increase with the iterations. The total computation time is about 1254 seconds. One way of checking the results of labeling is to compute the relative orientation of the object in 3-space using the final assignment of units. To compute the orientation, we need to compute the transformation matrix T in

$$T\underline{x} = \underline{b} \qquad (5\text{-}7)$$

where

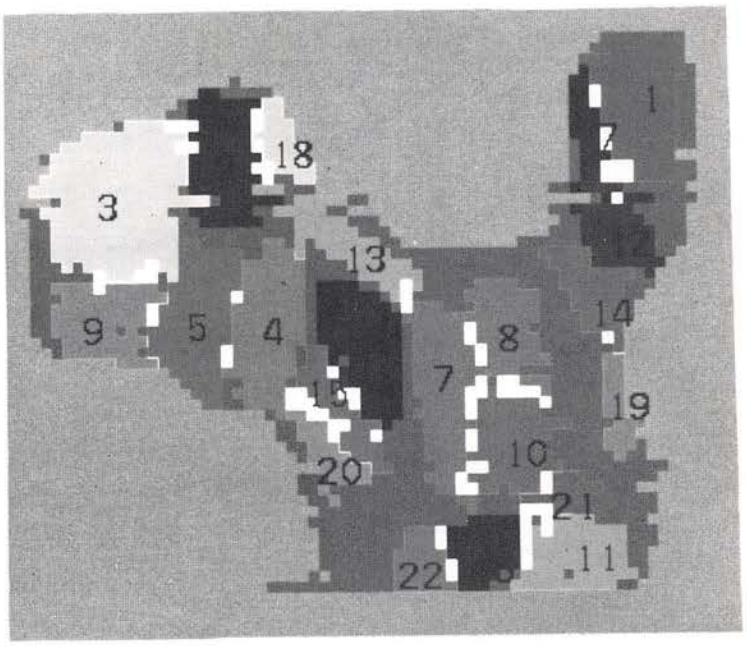


Fig. 5.2 Faces found in the view shown in fig. 4.8(a). There are 22 faces in this view and they are labeled in the order they are found using the algorithm described in chapter 4. The rejected points and the points common to two or more faces are shown in brown and white color respectively

Table 5.1 Neighbors of the faces shown in fig. 5.2. Neighbors are arranged in the descending order of their size

Face Number	Neighbors		
1	12	17	0
2	3	13	18
3	2	9	0
4	5	0	0
5	4	9	0
6	15	0	0
7	10	8	13
8	7	10	0
9	3	5	0
10	7	8	21
11	16	21	0
12	1	17	0
13	7	2	18
14	19	0	0
15	6	20	0
16	11	22	21
17	1	12	0
18	2	13	0
19	14	0	0
20	15	0	0
21	10	11	16
22	16	0	0

Table 5.2 Labels of the faces shown in fig. 5.2. Example 5.1. $\alpha = 0.99$, $p_i(nil) = 0.08$, ITER1 = 3, ITER2 = 6, UPTHLD = 0.8, LFACT = 10, No. of neighbors = (2, 2), No. of neighboring labels = 1, Inverse weights = (9, 7, 7, 4, 6, 30, 30, 40)

Face Number	Labels at different iterations					
	0	1	3	1	3	6
1	1(.08)	1(.17)	1(.40)	1(.58)	1(1.0)	1(1.0)
2	2(.08)	2(.13)	2(.25)	2(.46)	2(1.0)	2(1.0)
3	3(.38)	3(1.0)	3(1.0)	3(1.0)	3(1.0)	3(1.0)
4	86(.08)	4(.13)	4(.28)	4(.32)	4(.77)	4(1.0)
5	5(.08)	5(.16)	5(.31)	5(.43)	5(1.0)	5(1.0)
6	6(.09)	6(.20)	6(.37)	6(.55)	6(1.0)	6(1.0)
7	7(.11)	7(.65)	7(1.0)	7(1.0)	7(1.0)	7(1.0)
8	21(.20)	21(.76)	21(1.0)	21(1.0)	21(1.0)	21(1.0)
9	86(.08)	22(.36)	22(1.0)	22(1.0)	22(1.0)	22(1.0)
10	24(.20)	24(.76)	24(1.0)	24(1.0)	24(1.0)	24(1.0)
11	25(.15)	25(.40)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
12	86(.08)	86(.14)	86(.19)	45(.20)	45(.29)	45(1.0)
13	33(.11)	33(.62)	33(1.0)	33(1.0)	33(1.0)	33(1.0)
14	86(.08)	86(.24)	34(.31)	34(.42)	34(1.0)	34(1.0)
15	86(.08)	86(.13)	86(.16)	86(.16)	86(.22)	86(1.0)
16	86(.08)	53(.15)	53(.36)	53(.53)	53(1.0)	53(1.0)
17	86(.08)	45(.17)	45(.52)	45(.67)	45(1.0)	45(1.0)
18	46(.11)	46(.70)	46(1.0)	46(1.0)	46(1.0)	46(1.0)
19	86(.08)	86(.17)	86(.22)	86(.22)	50(.65)	50(1.0)
20	86(.08)	86(.24)	86(.25)	34(.28)	34(.61)	34(1.0)
21	86(.08)	50(.14)	50(.25)	50(.49)	50(1.0)	50(1.0)
22	86(.08)	86(.21)	86(.24)	86(.24)	79(.33)	79(1.0)
Value of Criterion	--	1.030	8.337	10.222	14.698	20.063
		$J^{(1)}$			$J^{(2)}$	

Total Computation Time = 1254.4 Seconds

$$T = \begin{bmatrix} \ell_1 & \ell_2 & \ell_3 \\ m_1 & m_2 & m_3 \\ n_1 & n_2 & n_3 \end{bmatrix} , \underline{x} = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} , \quad \underline{b} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} .$$

and (l_1, m_1, n_1), (l_2, m_2, n_2) and (l_3, m_3, n_3) are the direction cosines of the x', y', z' axis (unknown view) relative to x, y, z coordinates (model) respectively.

In order to compute T we need at least 3 matched pairs of units. From the results of matching (when the units are firmly assigned) the transformation matrix T is obtained by selecting any 3 units, (called a triple of units) which are not assigned to the nil class, and solving a set of 9 linear equations (see section 4.5) to evaluate the 9 coefficients of the matrix T. The (x, y, z) and (x', y', z') of any triple are taken as the centroids of the matched model face and an unknown view face respectively. It is possible that the matching results of any 3 units may not give the correct direction cosines as indicated by the values of the coefficients of the matrix T. All the coefficients of the matrix T should be within ± 1. So if the labeling of any of the 3 selected units happens to be wrong, the direction cosines will be errorneous. Moreover since we are interested only in the approximate matches not the exact matches (as they may not

exist and measurement errors are possible), it is quite likely that some triples do not lead to the valid direction cosines. A solution to this is to take the average of several valid solutions or obtain a least squares solution. The arccos of the coefficients of T will give the angle in degrees.

For the matching results shown in table 5.2, several triple of units such as (1, 2, 3), (1,3,4), (1, 13, 16), (16, 17, 18), (5, 6, 7), (2, 3 ,4), (2, 17, 18) etc. produce the coefficients of the matrix T very accurately. For example, the triple (5, 6, 7) solves the matrix T as

$$\begin{bmatrix} 1.0000 & -0.00305 & -0.011019 \\ -0.00534 & 1.00000 & 0.00586 \\ -0.009175 & 0.00000 & 1.00000 \end{bmatrix}$$

The arccos of each coefficients of the matrix T leads to

$$\begin{bmatrix} 0 & 90.14 & 90.63 \\ 90.30 & 0 & 89.66 \\ 90.52 & 90.00 & 0 \end{bmatrix}$$

Thus the relative orientation of the unknown view with respect to the model is about 0 degrees. Ideally the matrix T should be identity for this example.

In table 5.3 we show the matching results when we use the same parameters as in table 5.2 except only one neighbor is considered for the units of the unknown view when computing the compatibility and gradient at the first stage of hierarchy. Now the computation time is 1033 seconds which is little less than needed for the results shown in table 5.2 even though one more iteration is required at the second stage to obtain the firm assignment of all the units. However, now the label of units 6 and 15 are wrong. Thus using more neighbors improves the matching results as expected.

Table 5.4 shows the matching results when we use the same parameter values as used in table 5.2 except the value of $p_i(nil)$, the weight of the features and the number of neighbors at the first stage. Most of the matches are correct as in Table 5.2, however the computation time is reduced by a factor of 2. This is because we have taken a little higher value of $p_i(nil)$ and only one neighbor is considered at the first stage. Now the weights are also slightly different. Thus, although the matching results depend upon the weights (the initial assignment of probabilities), the exact values of weights is not critical to the success of the stochastic labeling technique.

Table 5.3 Labels of the faces shown in fig. 5.2. Example 5.1. Parameters same as in Table 5.2 except ITER2 = 8 and No. of neighbors = (1, 2)

Face Number	Labels at different iterations					
	0	1	3	1	3	8
1	1(.08)	1(.17)	1(.30)	1(.56)	1(1.0)	1(1.0)
2	2(.08)	2(.13)	2(.26)	2(.46)	2(1.0)	2(1.0)
3	3(.38)	3(1.0)	3(1.0)	3(1.0)	3(1.0)	3(1.0)
4	86(.08)	4(.13)	4(.28)	4(.32)	4(.77)	4(1.0)
5	5(.08)	5(.16)	5(.31)	5(.43)	5(1.0)	5(1.0)
6	6(.09)	6(.20)	6(.43)	6(.52)	27(1.0)	27(1.0)
7	7(.11)	7(.65)	7(1.0)	7(1.0)	7(1.0)	7(1.0)
8	21(.20)	21(.68)	21(1.0)	21(1.0)	21(1.0)	21(1.0)
9	86(.08)	22(.38)	22(1.0)	22(1.0)	22(1.0)	22(1.0)
10	24(.20)	24(.68)	24(1.0)	24(1.0)	24(1.0)	24(1.0)
11	25(.15)	25(.40)	25(1.0)	25(1.0)	25(1.0)	25(1.0)
12	86(.08)	86(.11)	45(.19)	45(.22)	45(.35)	45(1.0)
13	33(.11)	33(.71)	33(1.0)	33(1.0)	33(1.0)	33(1.0)
14	86(.08)	86(.24)	34(.31)	34(.42)	34(1.0)	34(1.0)
15	86(.08)	86(.09)	43(.14)	43(.17)	36(.28)	43(1.0)
16	86(.08)	53(.17)	53(.40)	53(.55)	53(1.0)	53(1.0)
17	86(.08)	45(.19)	45(.56)	45(.75)	45(1.0)	45(1.0)
18	46(.11)	46(.23)	46(.48)	46(1.0)	46(1.0)	46(1.0)
19	86(.08)	86(.17)	86(.22)	86(.22)	50(.65)	50(1.0)
20	86(.08)	86(.24)	86(.33)	86(.38)	79(.56)	79(1.0)
21	86(.08)	50(.14)	50(.25)	50(.50)	50(1.0)	50(1.0)
22	86(.08)	86(.21)	86(.24)	86(.24)	79(.33)	79(1.0)
Value of Criterion	--	.963	7.382	11.176	14.257	21.338
		$J^{(1)}$			$J^{(2)}$	

Total Computation Time = 1033 Seconds

Table 5.4 Labels of the faces shown in fig. 5.2. Example 5.1. Parameters same as in Table 5.2 except p_i(nil) = 0.1, No. of neighbors = (1, 2) and Inverse weights = (10, 8, 5, 5, 10, 25, 25, 25)

Face Number	Labels at different iterations					
	0	1	3	1	3	6
1	86(.10)	86(.22)	1(.37)	1(.41)	1(1.0)	1(1.0)
2	86(.10)	86(.13)	2(.19)	2(.28)	2(.48)	2(1.0)
3	3(.35)	3(.80)	3(1.0)	3(1.0)	3(1.0)	3(1.0)
4	86(.10)	86(.22)	86(.29)	86(.33)	86(.36)	4(1.0)
5	86(.10)	86(.21)	5(.33)	5(.42)	5(.61)	5(1.0)
6	86(.10)	86(.24)	6(.36)	6(.46)	6(1.0)	6(1.0)
7	7(.11)	7(.62)	7(1.0)	7(1.0)	7(1.0)	7(1.0)
8	21(.19)	21(.60)	21(1.0)	21(1.0)	21(1.0)	21(1.0)
9	86(.10)	22(.38)	22(1.0)	22(1.0)	22(1.0)	22(1.0)
10	24(.18)	24(.60)	24(1.0)	24(1.0)	24(1.0)	24(1.0)
11	25(.14)	25(.37)	25(.53)	25(.69)	25(1.0)	25(1.0)
12	86(.10)	86(.26)	86(.33)	86(.41)	86(.78)	86(1.0)
13	86(.10)	33(.61)	33(1.0)	33(1.0)	33(1.0)	33(1.0)
14	86(.10)	86(.35)	86(.35)	86(.34)	34(.58)	86(1.0)
15	86(.10)	86(.33)	86(.40)	86(.52)	86(1.0)	86(1.0)
16	86(.10)	53(.18)	53(.36)	53(.52)	53(1.0)	53(1.0)
17	86(.10)	86(.34)	86(.44)	45(.53)	45(1.0)	45(1.0)
18	46(.10)	46(.22)	46(.53)	46(.67)	46(1.0)	46(1.0)
19	86(.10)	86(.27)	86(.30)	86(.27)	67(.39)	50(1.0)
20	86(.10)	86(.32)	86(.35)	86(.34)	34(.59)	34(1.0)
21	86(.10)	86(.16)	56(.34)	50(.53)	50(1.0)	50(1.0)
22	86(.10)	86(.26)	86(.34)	86(.34)	79(.39)	79(1.0)
Value of Criterion	--	1.091	8.094	9.419	13.463	20.857
		$J^{(1)}$			$J^{(2)}$	

Total Computation Time = 566.4 Seconds

Example 5.2

Fig. 5.3 shows the faces found in the view shown in Fig. 4.8(b). There are 24 faces in this view and they are labeled in the order thay are found using the algorithm described in chapter 4. Table 5.5 shows the neighbors of the faces shown in Fig. 5.3. Comparing figs. 5.2 and 5.3, it can be seen how some of the faces of fig. 5.3 should be labeled. For example faces 11 and 7 in fig. 5.3 correspond to faces 8 and 10 in fig. 5.2 respectively. Similarly the correspondence for some other faces can be obtained and the matching results can be verified by using tables 5.2 and 5.6. For the results shown in table 5.6 we have used the same parameters as used in obtaining the results shown in table 5.2. Most of the labels are correct. The total computation time is about 7 minutes.

As in the example 5.1, various triple of units allow us to compute the transformation matrix T. For example using the triple (4,7,8), matrix T is obtained as

$$\begin{bmatrix} 0.88383 & 0.09058 & 0.46854 \\ -0.20947 & 1.00000 & -0.19653 \\ -0.45183 & 0.01863 & 0.86441 \end{bmatrix}$$

For a relative orientation of 30 degrees in the x-z plane the matrix T is,

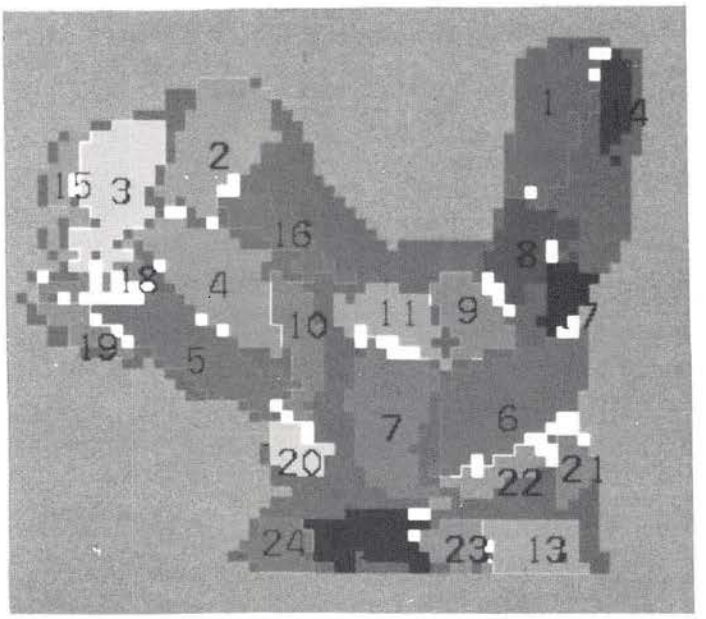


Fig. 5.3 Faces found in the view shown in fig. 4.8(b). There are 24 faces in this view and they are labeled in the order they are found using the algorithm described in chapter 4. The rejected points and the points common to two or more faces are shown in brown and white color respectively

Table 5.5 Neighbors of the faces shown in fig. 5.3.. Neighbors are arranged in the descending order of their size

Face Number	Neighbors			
1	8	14	0	0
2	4	16	0	0
3	18	15	0	0
4	5	2	18	0
5	4	18	19	20
6	22	17	21	0
7	11	0	0	0
8	1	9	17	0
9	8	0	0	0
10	20	0	0	0
11	7	0	0	0
12	22	23	0	0
13	23	0	0	0
14	1	0	0	0
15	3	18	0	0
16	2	0	0	0
17	6	8	0	0
18	5	4	3	15
19	5	0	0	0
20	5	10	0	0
21	6	22	0	0
22	6	12	21	0
23	12	13	0	0
24	0	0	0	0

Table 5.6 Labels of the faces shown in fig. 5.3. Example 5.2. $\alpha = 0.99$, $p_i(nil)=0.08$, ITER1 = 3, ITER2 = 7, UPTHLD = 0.8, LFACT = 10, No. of neighbors = (2, 2), No. of neighboring labels = 1, Inverse weights = (9, 7, 7, 4, 6, 30, 30, 40)

Face Number	Labels at different iterations					
	0	1	3	1	4	7
1	86(.08)	86(.08)	86(.19)	86(.21)	1(.56)	1(1.0)
2	86(.08)	86(.17)	86(.21)	2(.25)	2(.74)	2(1.0)
3	86(.08)	86(.18)	2(.27)	86(.26)	2(1.0)	2(1.0)
4	86(.08)	86(.16)	5(.21)	5(.32)	5(1.0)	5(1.0)
5	86(.08)	86(.17)	86(.20)	86(.31)	5(.49)	5(1.0)
6	86(.08)	86(.18)	86(.21)	86(.30)	86(1.0)	86(1.0)
7	86(.08)	86(.17)	86(.20)	86(.25)	24(.57)	24(1.0)
8	86(.08)	86(.16)	20(.20)	20(.27)	20(.80)	20(1.0)
9	86(.08)	86(.24)	34(.34)	34(.39)	34(1.0)	34(1.0)
10	86(.08)	86(.23)	35(.30)	35(.35)	45(1.0)	45(1.0)
11	86(.08)	86(.22)	34(.30)	34(.40)	34(.61)	53(1.0)
12	86(.08)	86(.19)	26(.23)	26(.31)	26(.49)	53(1.0)
13	86(.08)	86(.30)	86(.33)	40(.33)	53(1.0)	53(1.0)
14	86(.08)	86(.24)	86(.31)	64(.41)	86(1.0)	86(1.0)
15	86(.08)	86(.20)	86(.25)	86(.31)	86(.54)	86(1.0)
16	86(.08)	86(.23)	86(.27)	41(.29)	86(1.0)	86(1.0)
17	86(.08)	86(.20)	86(.25)	86(.28)	62(.39)	62(1.0)
18	86(.08)	86(.28)	77(.38)	86(.54)	86(1.0)	86(1.0)
19	86(.08)	86(.16)	86(.20)	65(.25)	65(1.0)	65(1.0)
20	86(.08)	86(.16)	86(.22)	86(.22)	50(.54)	52(1.0)
21	86(.08)	86(.17)	70(.22)	86(.26)	86(.36)	50(1.0)
22	86(.08)	86(.23)	77(.39)	77(.50)	77(1.0)	77(1.0)
23	86(.08)	86(.19)	86(.23)	52(.27)	52(.39)	52(1.0)
24	86(1.0)	86(1.0)	86(1.0)	86(1.0)	86(1.0)	86(1.0)
Value of Criterion	--	1.874 $J^{(1)}$	3.647	4.327	12.606 $J^{(2)}$	23.017

Total Computation Time = 425.6 Seconds

$$\begin{bmatrix} 0.866 & 0 & 0.500 \\ 0 & 1 & 0 \\ -0.500 & 0 & 0.866 \end{bmatrix}$$

Thus the relative rotation in the x-z plane of about 30 degrees is obtained for the unknown view shown in fig. 5.3.

Example 5.3

Fig. 5.4 shows the faces obtained in the view shown in fig. 4.8(1). There are 24 faces in this view and as before they are labeled in the order they are found using the algorithm described in chapter 4. Neighbors of the faces in fig. 5.4 are shown in table 5.7. These neighbors are arranged in the descending order of their size. Comparing figs. 5.2, 5.3 and 5.4 one can observe how the face description has changed. Also it can be inferred how the faces of fig. 5.4 should be labeled with respect to the labeling of faces in figs. 5.2 and 5.3. Table 5.8 shows the results of the stochastic labeling. Most of the labels are correct, but a few of them are wrong because of the higher degree of similarity of the local structure of the incorrect match. Total computation time for this example is about 1022 seconds.

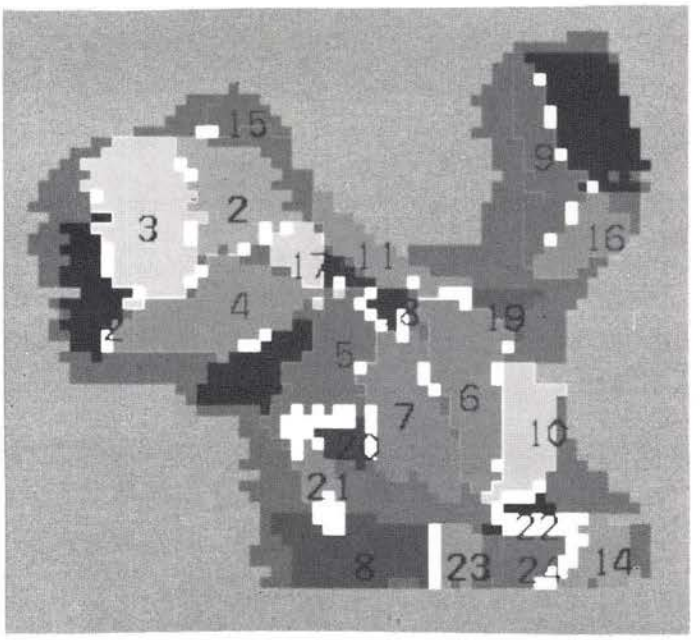


Fig. 5.4 Faces found in the view shown in fig. 4.8(1). There are 24 faces in this view and they are labeled in the order they are found using the algorithm described in chapter 4. The rejected points and the points common to two or more faces are shown in brown and white color respectively

Table 5.7 Neighbors of the faces shown in fig. 5.4.· Neighbors are arranged in the descending order of their size

Face Number	Neighbors			
1	9	16	0	0
2	3	4	17	15
3	4	2	12	0
4	3	2	12	13
5	7	20	18	0
6	7	10	11	19
7	6	5	20	18
8	21	23	0	0
9	1	16	0	0
10	6	22	0	0
11	6	0	0	0
12	3	4	0	0
13	4	0	0	0
14	24	22	0	0
15	2	0	0	0
16	1	9	0	0
17	2	0	0	0
18	7	5	0	0
19	6	0	0	0
20	7	5	21	0
21	8	20	0	0
22	10	24	14	0
23	8	0	0	0
24	4	22	0	0

Table 5.8 Labels of the faces shown in fig. 5.4. Example 5.3. $\alpha = 0.99$, $p_i(nil)=0.08$, ITER1 = 3, ITER2 = 7, UPTHLD = 0.8, LFACT = 10, No. of neighbors = (2, 2), No. of neighboring labels = 1, Inverse weights = (9, 5, 6, 4, 6, 30, 30, 40)

Face Number	Labels at different iterations					
	0	1	3	1	4	8
1	86(.08)	86(.15)	5(.21)	86(.30)	86(.62)	86(1.0)
2	2(.08)	2(.18)	2(.34)	2(.46)	2(1.0)	2(1.0)
3	86(.08)	86(.15)	86(.22)	86(.24)	86(.28)	3(1.0)
4	86(.08)	86(.13)	5(.20)	5(.25)	5(.50)	5(1.0)
5	86(.08)	86(.17)	6(.24)	6(.35)	6(1.0)	6(1.0)
6	86(.08)	86(.17)	7(.32)	7(.41)	7(1.0)	7(1.0)
7	86(.08)	86(.16)	86(.22)	7(.32)	7(1.0)	7(1.0)
8	86(.08)	86(.21)	86(.25)	5(.38)	5(.71)	5(1.0)
9	86(.08)	86(.18)	86(.22)	86(.50)	86(1.0)	86(1.0)
10	86(.08)	86(.15)	86(.20)	24(.24)	24(.41)	21(1.0)
11	86(.08)	86(.21)	86(.25)	86(.26)	33(1.0)	33(1.0)
12	86(.08)	86(.19)	86(.25)	86(.36)	86(.61)	86(1.0)
13	86(.08)	86(.21)	86(.22)	86(.29)	27(.53)	27(1.0)
14	86(.08)	86(.24)	86(.27)	86(.39)	86(.59)	34(1.0)
15	86(.08)	86(.13)	86(.20)	86(.23)	46(.58)	46(1.0)
16	86(.08)	86(.23)	34(.30)	86(.36)	34(1.0)	34(1.0)
17	86(.08)	46(.13)	46(.27)	46(.33)	46(.73)	46(1.0)
18	86(.08)	86(.18)	86(.24)	79(.35)	79(.60)	79(1.0)
19	86(.08)	86(.21)	86(.27)	86(.31)	70(.53)	70(1.0)
20	86(.08)	86(.24)	86(.28)	86(.38)	86(1.0)	86(1.0)
21	86(.08)	86(.23)	86(.30)	86(.29)	47(.36)	72(1.0)
22	86(.08)	86(.27)	77(.68)	77(1.0)	77(1.0)	77(1.0)
23	86(.08)	86(.16)	86(.19)	67(.21)	86(.71)	86(1.0)
24	86(.08)	86(.31)	86(.36)	86(.41)	36(1.0)	36(1.0)
Value of Criterion	--	.9126	2.759	3.867	9.358	23.50
		$J^{(1)}$			$J^{(2)}$	

Total Computation Time = 1022 Seconds

As in the previous examples, triples of units can be used to compute the transformation matrix T. For example using the triple (2,5,6), matrix T is obtained as

$$\begin{bmatrix} 0.696 & -0.026 & -0.587 \\ -0.141 & 1.000 & -0.010 \\ 0.491 & -0.025 & 0.707 \end{bmatrix}$$

For a relative orientation of 330 degrees in the x-z plane, the matrix T is,

$$\begin{bmatrix} 0.866 & 0 & -0.500 \\ 0 & 1 & 0 \\ 0.500 & 0 & 0.866 \end{bmatrix}$$

Thus the matching results provide reasonable orientation information.

Comments

The examples presented in the above show that the partial shape recognition can be accomplished using the hierarchical stochastic labeling technique based on "face matching." The results of labeling are good and very reasonable. A few incorrect assignments result because the structure and description of a unit with its neighbors (called the local structure) matches better with the

incorrect match than the correct match. Also if the object has some symmetry, it is likely that there will be multiple matches. The results of labeling depend upon the planar surface approximation and neighborhood information. An approximation of the surface of an object which includes planar and curved (for example quadratic) faces and which are contiguous (there are no rejected points) and provides complete neighborhood information will be the ideal case since the contextual information will be more effective in the stochastic labeling process. Also the number of views to obtain a model depends upon the complexity of the object. For the examples presented the computation time varied from about 7 minutes to 20 minutes. Over 95% of this time is spent in the computation of rotation needed in the compatibility computation. This is because we store only the boundary of the image of a face. Also we do not store the compatibility values, and recompute them when the gradient is required. Thus if we store the images of the faces and the compatibility values, computation time will be very small.

The results of labeling allow us to obtain the orientation of the object in three-space just like in 2-D the results of matching were used to compute the rotation. Normally, we used 3 iterations at the first stage and 4 to

8 iterations at the second stage. At the first stage of hierarchy we used only a limited number of neighbors (1 or 2) at a time (binary relations) and at the second stage two largest neighboring faces together (a subset of ternary relations) to compute the compatibilities. We found that these two labels of hierarchy are sufficient for matching purposes, although the method generalizes to include higher levels at the expense of increased computation. The first stage does not resolve all the ambiguous labelings. The second stage helps in correcting these labelings.

5.4 Summary

In this chapter we presented an extension of the hierarchical stochastic labeling technique developed in Chapter 2 to the shape matching of real world 3-D objects. The matching algorithm is based on "face matching" i.e., matching the large faces in an unknown view against the faces of the 3-D model of the object. Several examples of a complex industrial object are presented, which illustrate how the hierarchical stochastic labeling technique could be successfully applied to the problem of partial shape recognition of objects in three-dimensions. The results of matching are used to obtain the orientation of the object in three-space in a relatively simple

fashion. These results could be fed to a robot manipulator on an assembly line or inspection stages of the production.

CHAPTER 6

SEGMENTATION OF IMAGES

6.1 Introduction

In the previous chapters we have used the hierarchical stochastic labeling technique for the shape matching of objects in two and three dimensions. In this chapter we investigate the use of a stochastic labeling method in another important area of image analysis: segmentation. In particular we shall consider the segmentation of images having unimodal intensity (or color) distributions. Unimodal distributions are typically obtained when the image consists mostly of a large background area with other small but significant regions. For example, in the biomedical area the extraction of the boundaries of various types of cells is complicated by the fact that the cells are very close together, their boundaries are poorly defined and the gray level histogram is unimodal. Similarly in scenes with many different objects as the case with aerial photographs, the histogram may have only one peak because the range of intensities for each object will probably

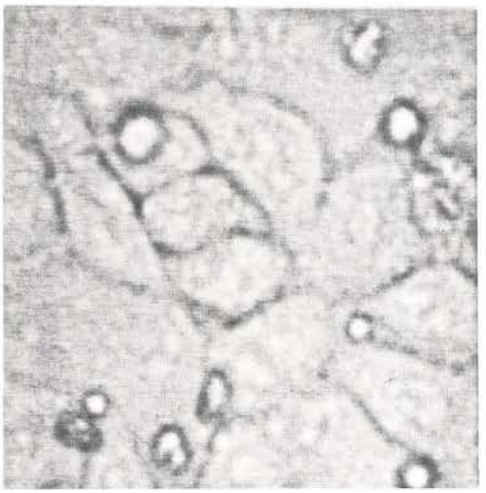
overlap with the ranges of other objects.

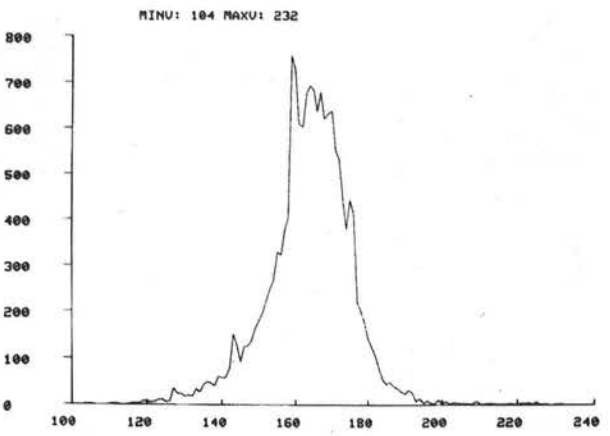
Various approaches based on thresholding have been used by many researchers for the segmentation of both monochrome and color pictures [6-1, 6-3]. Normally in the application of these techniques the histogram shows two or more peaks in at least one of the spectral features corresponding to various homogeneous regions of an image. Very often preprocessing is done to improve the histograms and local properties are used to compute the global, local or dynamic thresholds. However, if the intensity (or color) histogram of the image is unimodal, then the application of such methods gives a poor segmentation and there are no criteria for automatic threshold selection. Jain et al. [6-4] consider the segmentation of muscle cell pictures using 10 low level operators which are very time consuming and require the selection of 5 thresholds. Rosenfeld and Davis [6-5] and Peleg [6-6] use iterative methods to modify the histogram. Their methods do not take into account the possibility that small regions in an image may have significance even though they may not show significant peaks on the image's histogram. Rosenfeld [6-7] considers thresholding by relaxation. After the application of basic segmentation methods described in section 6.2, we shall study in detail the gradient relaxation method (section 6.3) based on maximizing a

criterion function (as used in shape matching) and compare it with the nonlinear probabilistic relaxation method [6-7, 6-8] for the purpose of segmenting images having unimodal gray level distributions. Although both methods provide comparable segmentation results, the gradient method has the additional advantage of providing control over the relaxation process by choosing three parameters which can be tuned to obtain the desired segmentation results. Examples are given on two different types of scenes.

6.2 Segmentation Schemes

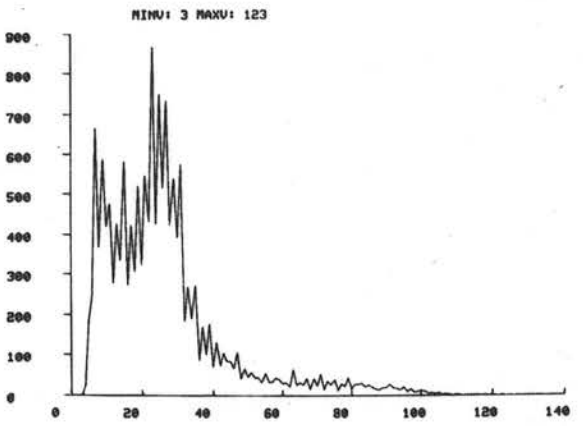
Fig. 6.1 shows two 128×128 pixels 8-bit images and their gray level histograms. The image in Fig. 6.1(a) has been obtained using a real time video acquisition and digital display system [6-9] described in Appendix B. The background in this image consists of a confluent monolayer of human skin cancer cells and the small circular shaped objects are human lymphocytes and red blood cells. The objective is to get the boundaries of all the cells. The image in Fig. 6.1(b) is part of an aerial photograph. Here the objective is to detect significant features such as roads, etc. Note that the histograms of the images (Fig. 6.1(c) and 6.1(d)) are almost unimodal and as a consequence there is no reliable way of automatically





(a) Cell image

(c) Gray level histogram of the cell image. Mean = 163.76



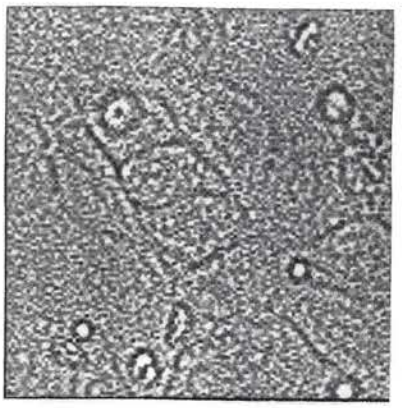
(b) An aerial image

(d) Gray level histogram of the aerial image. Mean = 26.47

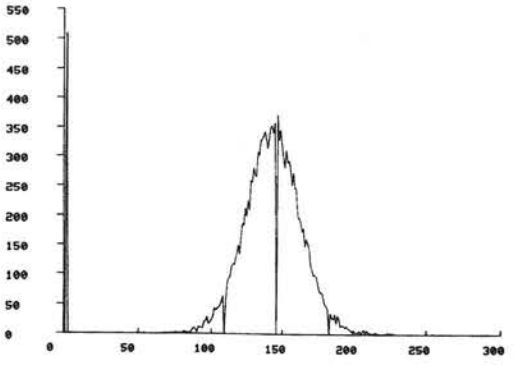
Fig. 6.1 Two typical 128x128, 8 bit images and their gray level histograms

choosing a threshold for segmenting these pictures.

Commonly used difference operators such as gradient, Laplacian and Sobel were applied to the images shown in Fig. 6.1. For example Fig. 6.2(a) shows the Laplacian of the cell image obtained by convolution of the cell image by a 3×3 mask. The gray level histogram of the image in Fig. 6.2(a) is shown in Fig. 6.2(b). We also considered the methods based on thresholding the histogram of the picture where the gradient, Laplacian and edge values are high [6-1]. However, the pictures so obtained have unimodal histogram and lack the criterion for segmenting them at the valley of two peaks. Thresholding at the gray level corresponding to the mode or mean of the filtered histogram gave very poor results. Edge detection has also been done by convolving the images in Fig. 6.1 with 5×5 masks corresponding to the ideal step edges in six directions [6-10]. Thresholding of the magnitude image does not show good segmentation. A number of bar masks of various sizes and orientations have also been used, but the results were poor (for the image shown in Fig. 6.1a the width of an edge is about 5-6 pixels).



(a) Laplacian of the cell image



(b) Histogram of the image in fig. 6.2(a)

Fig. 6.2 Laplacian of the cell image and the corresponding gray level histogram

6.3 Segmentation Using Stochastic Labeling

Having considered the basic differencing, edge, bar operators and their variations, we focussed our attention on the stochastic labeling methods for segmentation. As mentioned before "stochastic labeling" is an iterative approach to classifying a set of interrelated units. A set of estimates of class assignment probabilities is initially associated with each unit. At subsequent iterations, the probabilities are adjusted in accordance with the support they receive from the class probabilities of related units. This process, when repeated, leads to a reduction in the ambiguity and inconsistency of classification.

a. Gradient Relaxation Algorithm

In [6-11] Faugeras and Berthod proposed a relaxation algorithm which is based upon the explicit use of consistency and ambiguity to define a global criterion upon the set of units. This criterion has the inherent problem in that the consistency and ambiguity tend to go in opposite directions. Therefore, as in the study of shape matching, we consider a simpler criterion [6-12] based on the inner product of probability vector $\vec{p}_i$ and compatibility (or consistency) vector $\vec{q}_i$.

$$C = \sum_{i=1}^{N} \vec{p}_i \cdot \vec{q}_i \qquad (6.1)$$

and carry out its maximization using the gradient projection approach. This criterion is also easier to manipulate computationally [6-14].

Suppose we have a set of N units (pixels) $a_1,\ldots,a_N$, which fall into two classes (L=2), λ_1 and λ_2 corresponding to the white (gray value =255) and black (gray value = 0). The relaxation process is specified by choosing a model of interaction between pixels. We attach to every pixel a_i the set V_i of its 8 nearest neighbors. Assuming that objects of interest in the picture are continuous, we will make like reinforce like and define a compatibility function c such that:

$$\begin{aligned} &c(a_i,\lambda_k,a_j,\lambda_\ell) = 0,\ k \neq \ell,\ a_j \in V_i \quad \text{for all } i \\ &c(a_i,\lambda_k,a_j,\lambda_k) = 1,\ k = 1,2,\ a_j \in V_i \quad \text{for all } i \end{aligned} \qquad (6.2)$$

The consistency vector $\vec{q}_i$ is then defined as

$$q_i(\lambda_k) = \frac{1}{8} \sum_{a_j \in V_i} \sum_{\ell=1}^{2} c(a_i,\lambda_k,a_j,\lambda_\ell) p_j(\lambda_\ell) \qquad (6.3)$$

$$k = 1,2$$
$$i = 1,\ldots,N$$

The choice of compatibility function in (6.2) will give the desired result in the interior of the region, but on the border of a region the pixel label may be ambiguous

because of two different classes of neighbors and may cause a distortion of the boundary.

The initial assignment of probabilities to every unit is very important. It affects the convergence rate and the results of relaxation schemes. The simplest way to compute the initial probabilities [6-7, 6-13] is to define

$$p_i(\lambda_1) = \frac{I(a_i)}{G-1} \tag{6.4}$$

where $I(a_i)$ is the intensity at pixel a_i and G the number of possible gray levels ($0 \leq I(a_i) \leq G-1$). This has the problem of completely ignoring any a priori information that we may have about the contents of the image. It is possible to include a rough knowledge of the relative number of white pixels versus black pixels. If $\bar{I}$ denotes the mean of the image, we may have an approximate idea of the value of the number of white versus black pixels (the ratio $r_0 = n_{white}/n_{black}$). If we estimate this ratio for the image, we obtain,

$$r = \frac{n_{white}}{n_{black}} = \frac{E(\lambda_1)}{E(\lambda_2)} = \frac{\frac{1}{N^2}\sum p_i(\lambda_1)}{\frac{1}{N^2}\sum p_i(\lambda_2)} = \frac{\bar{I}}{G-1-\bar{I}} \tag{6.5}$$

If we know a priori that there are more white than black pixels in the image we may want to modify the distribution

of gray levels so as to make the ratio r closer to the known ratio r_0. One very simple way to do so is to define

$$I' = FACT * (I-\bar{I}) + \bar{I}_0 \qquad (6.6)$$

where $\bar{I}_0$ is a desired mean and FACT a function of the intensity which is taken to be equal to 1 if $I > \bar{I}$ and less than 1 if $I < \bar{I}$. In our experiments the initial assignment of probabilities has been obtained by,

$$p_i(\lambda_1) = FACT * \left(\frac{I(a_i) - \bar{I}}{255}\right) + 0.5 \qquad (6.7)$$

when $I(a_i) < \bar{I}$, FACT has usually been taken between 0.7 and 1. Of course, if the first term of (6.7) happens to be less than -0.5, then a probability of zero is assigned to that pixel.

The gradient of the criterion C in (6.1) is obtained as,

$$\frac{\partial C}{\partial p_i(\lambda_1)} = 2q_i(\lambda_1) \qquad (6.8)$$

$$\frac{\partial C}{\partial p_i(\lambda_2)} = 2q_i(\lambda_2) \qquad (6.9)$$

and the iteration of p_i's is given by,

$$p_i^{(n+1)}(\lambda_1) = p_i^{(n)}(\lambda_1) + \rho_i^{(n)} P_i^{(n)} \left[\frac{\partial C}{\partial p_i(\lambda_1)}\right] \qquad (6.10)$$

$$p_i^{(n+1)}(\lambda_2) = p_i^{(n)}(\lambda_2) + \rho_i^{(n)} P_i^{(n)} \left[\frac{\partial C}{\partial p_i(\lambda_2)}\right] \qquad (6.11)$$

In order to have $p_i^{(n+1)}(\lambda_1)+p_i^{(n+1)}(\lambda_2) = 1$, the projection of the gradient should be such that

$$P_i^{(n)} \left[\frac{\partial C}{\partial p_i(\lambda_1)}\right] = 2q_i(\lambda_1) - \frac{1}{2}\left[\frac{\partial C}{\partial p_i(\lambda_1)} + \frac{\partial C}{\partial p_i(\lambda_2)}\right] \qquad (6.12)$$

and

$$P_i^{(n)} \left[\frac{\partial C}{\partial p_i(\lambda_2)}\right] = 2q_i(\lambda_2) - \frac{1}{2}\left[\frac{\partial C}{\partial p_i(\lambda_1)} + \frac{\partial C}{\partial p_i(\lambda_2)}\right] \qquad (6.13)$$

but

$$\frac{\partial C}{\partial p_i(\lambda_1)} + \frac{\partial C}{\partial p_i(\lambda_2)} = 2$$

So the iteration in (6.10) and (6.11) reduces to

$$p_i^{(n+1)}(\lambda_1) = p_i^{(n)}(\lambda_1) + \rho_i^{(n)}[2q_i(\lambda_1)-1] \qquad (6.14)$$

$$p_i^{(n+1)}(\lambda_2) = p_i^{(n)}(\lambda_2) + \rho_i^{(n)}[2q_i(\lambda_2)-1] \qquad (6.15)$$

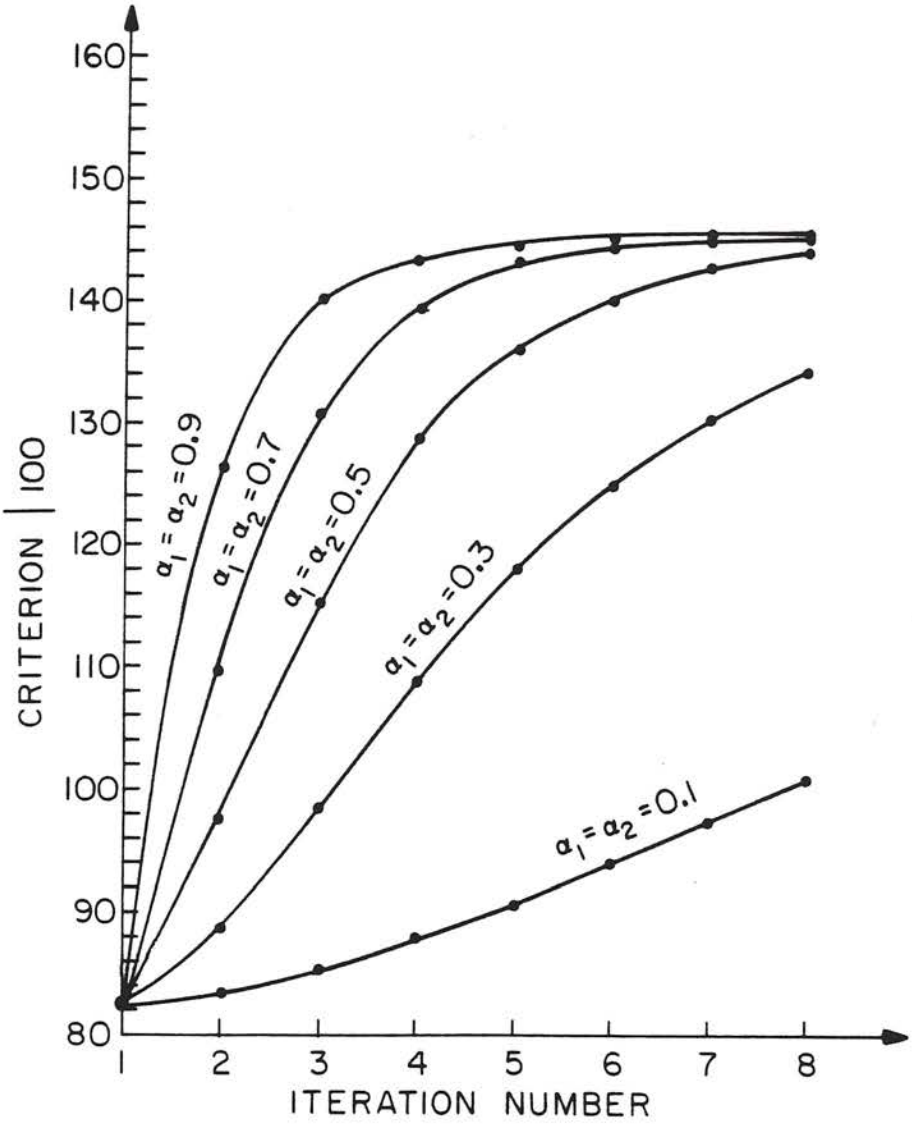
Normally, $\rho_i^{(n)}$ is kept constant for all pixels during each iteration and is assigned the largest possible value such

that p_i's at the n+1st iteration still lie in the bounded convex region of 2N dimensional Euclidean space defined by $p_i(\lambda_1)+p_i(\lambda_2) = 1$ and $p_i(\lambda_k) \geq 0$, k = 1,2 and i = 1,...,N. However, in the 2 class case considered, it is easier to compute $\rho_i^{(n)}$ for each pixel. This leads to a faster convergence rate. It is obtained from (6.14) and (6.15) as

$$\rho_i^{(n)} = \alpha_1 \left(\frac{1-p_i^{(n)}(\lambda_k)}{2q_i(\lambda_k)-1}\right), \qquad \text{if } 2q_i(\lambda_k)-1 > 0 \tag{6.16}$$

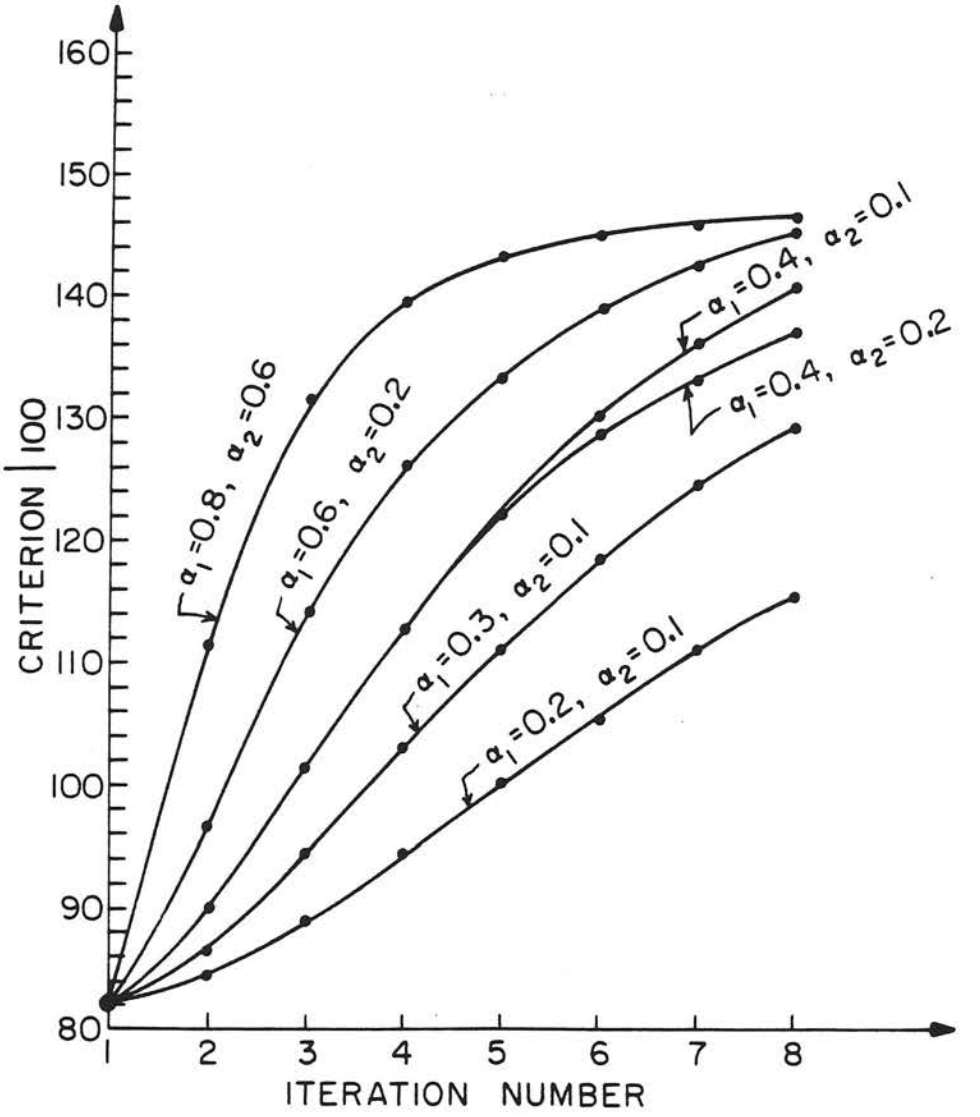
$$= \alpha_2 \left(\frac{p_i^{(n)}(\lambda_k)}{1-2q_i(\lambda_k)}\right), \qquad \text{if } 2q_i(\lambda_k)-1 < 0 \tag{6.17}$$

where, k = 1,2 and α_1 and α_2 are constants less than unity. A side effect of computing $\rho_i^{(n)}$ for every pixel is that we may not be following the gradient exactly. However, it can be expected that we are approximately in the direction of the gradient and the criterion (6.1) is still maximized. It is our conjecture that for two class case, the criterion will always increase. Figure 6.3a shows the behavior of criterion as the iteration number increases for the cell image when $\alpha_1 = \alpha_2$. The same behavior is obtained when α_1 and α_2 are not equal (Fig. 6.3b and c). The values of α_1 and α_2 can be used to bias a class and control the speed of convergence, hence



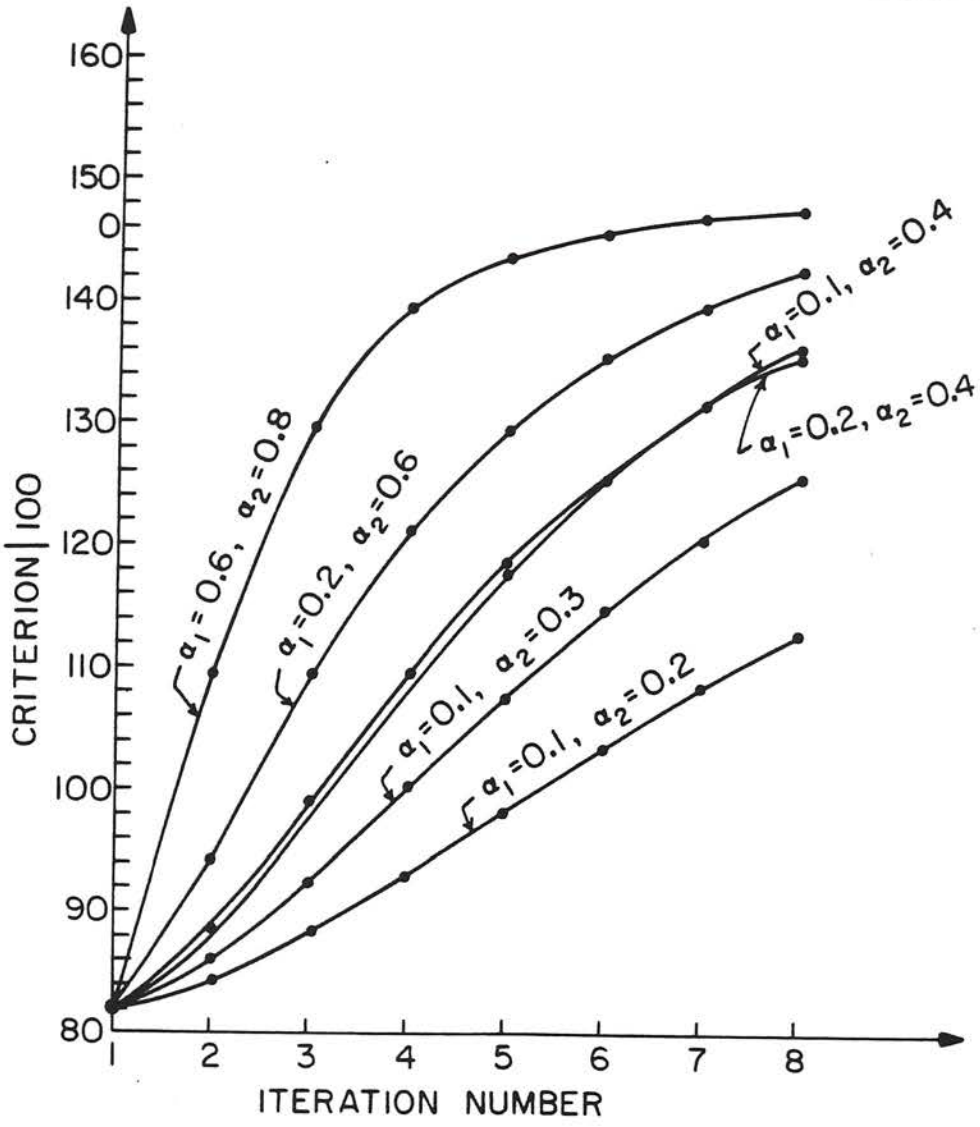
(a) α_1/α_2 = FACT = 1 in all cases

Fig. 6.3 Variation of the criterion (6.1) with the iteration number for various values of α_1 and α_2 for the cell image



(b) α_1/α_2 > 1 and FACT = 1 in all cases

Fig. 6.3 (CONTINUED)



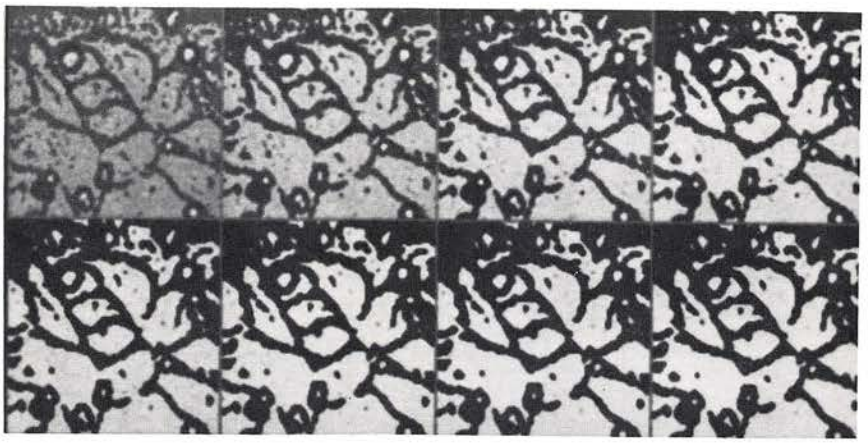
(c) $\alpha_1/\alpha_2 < 1$ and FACT = 1 in all cases

Fig. 6.3 (CONTINUED)

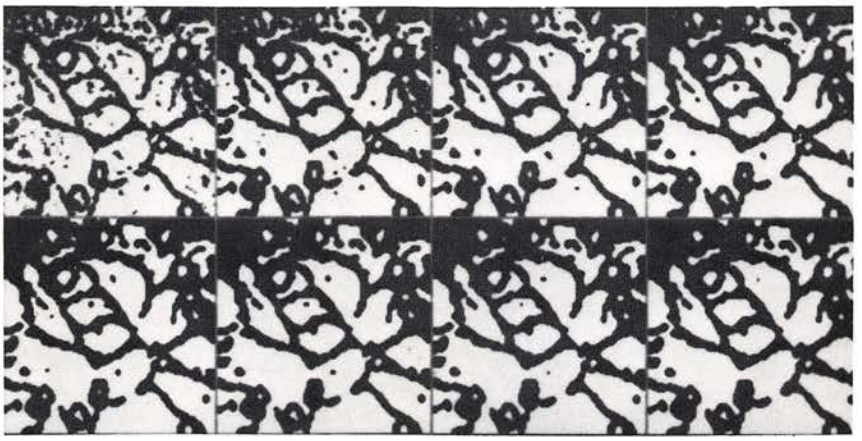
the control over the relaxation process. Figs. 6.4-6.7 illustrate these effects for different vales of α_1 and α_2 on the cell image.

Figs. 6.4 to 6.6 show how changing the ratio of α_1 and α_2 allows the control of where we converge by biasing one class. From Fig. 6.3 it can be seen that for a fixed ratio of α_1 and α_2, increasing both of them by a constant factor increases the speed of convergence and Fig. 6.7 verifies this fact that indeed we converge towards the same result. Therefore, changing the values of α_1 and α_2 not only allows where one wants to converge but also how fast one converges to the same limit. Thus control over the relaxation process is achieved.

Fig. 6.8 and 6.9 show the results of gradient relaxation method at various iterations and corresponding histograms for the cell and aerial image respectively. Observe that at the first iteration itself we get two peaks separated by a valley which can be used to automatically select the threshold to obtain segmentation. As the number of iteration increases the two peaks get farther apart, average brightness increases, and the convergence of probabilities takes place as expected. When the peaks are far apart thresholding can be done at the mean value. Also it has been found that the actual

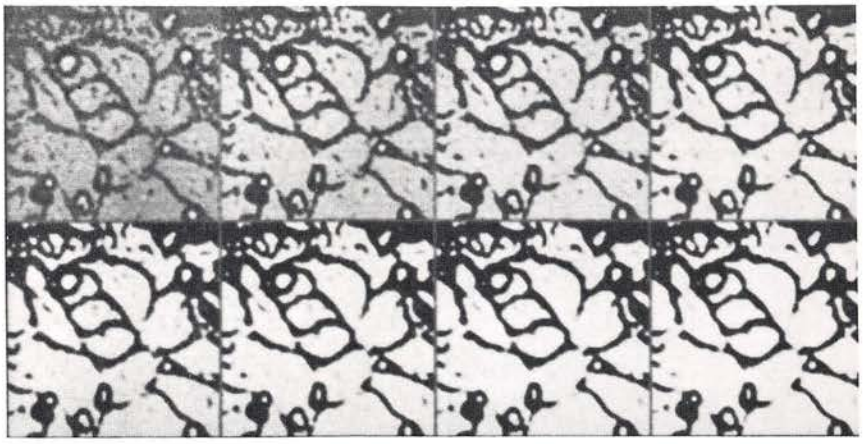


(a) $\alpha_1 = \alpha_2 = 0.2$, FACT = 1.0

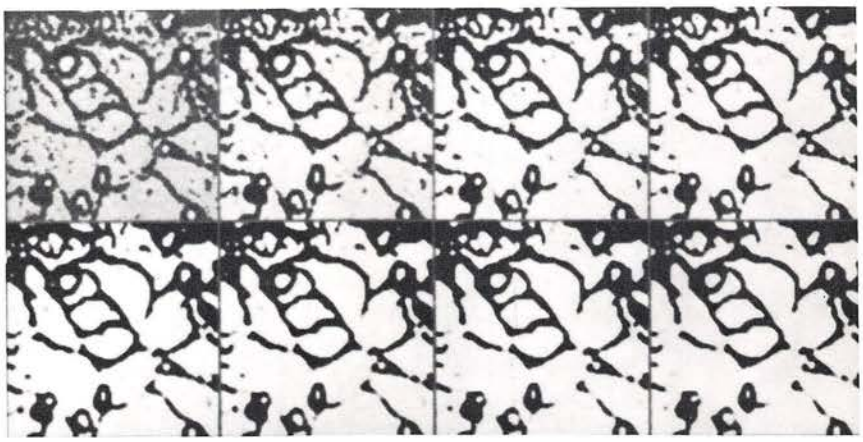


(b) $\alpha_1 = \alpha_2 = 0.8$, FACT = 1.0

Fig. 6.4 Effect of different values of α_1 and α_2 on the cell image such that $\alpha_1/\alpha_2 = 1$. In each figure the first row contains images for the first 4 iterations and the second row for the next 4 iterations

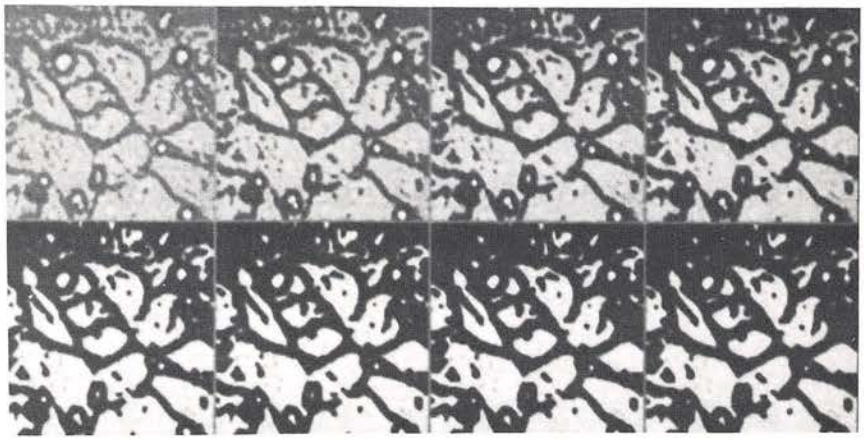


(a) $\alpha_1 = 0.2$, $\alpha_2 = 0.1$, FACT = 1.0

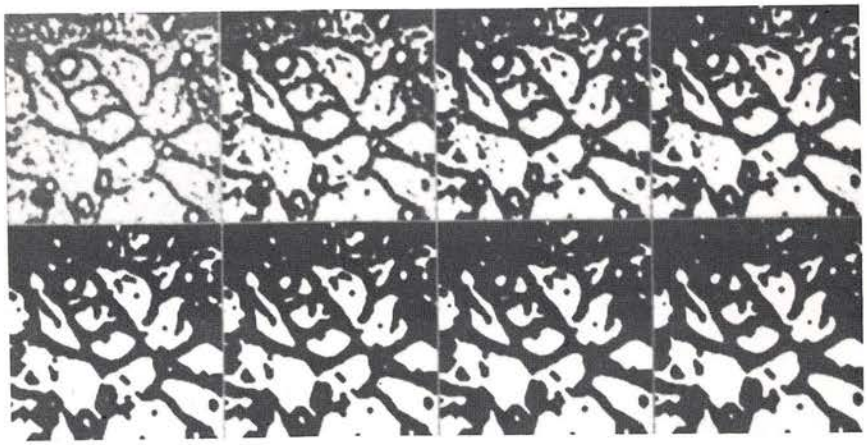


(b) $\alpha_1 = 0.4$, $\alpha_2 = 0.2$, FACT = 1.0

Fig. 6.5 Effect of biasing of class λ_1 on the cell image such that $\alpha_1/\alpha_2 = 2$



(a) $\alpha_1 = 0.1$, $\alpha_2 = 0.2$, FACT = 1.0



(b) $\alpha_1 = 0.2$, $\alpha_2 = 0.4$, FACT = 1.0

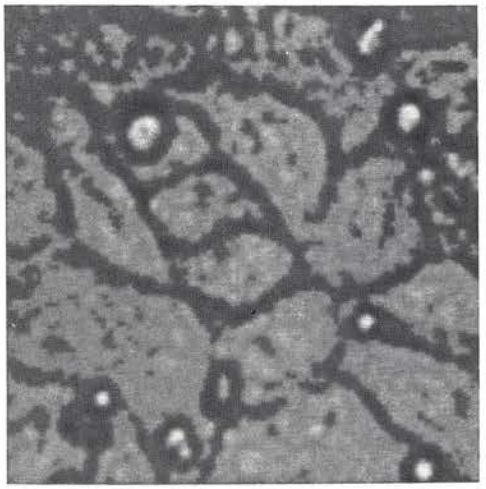
Fig. 6.6 Effect of biasing class λ_2 on the cell image such that $\alpha_1/\alpha_2 = 1/2$

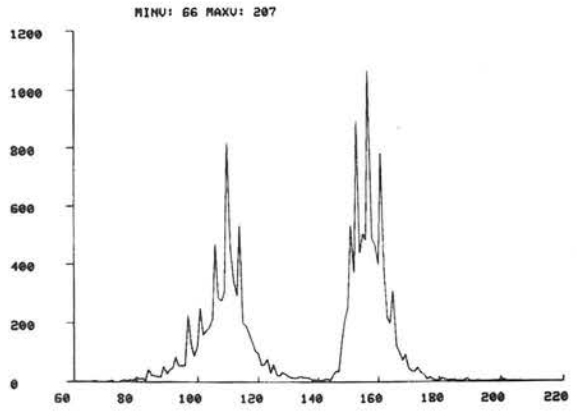
(a) $\alpha_1 = \alpha_2 = 0.2$, iteration 8

(b) $\alpha_1 = \alpha_2 = 0.5$, iteration 4

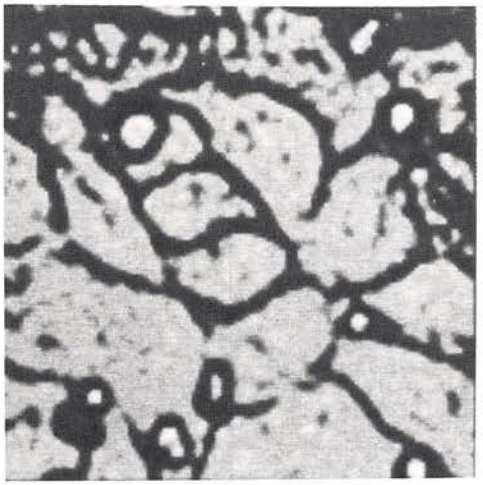
(c) $\alpha_1 = \alpha_2 = 0.9$, iteration 3

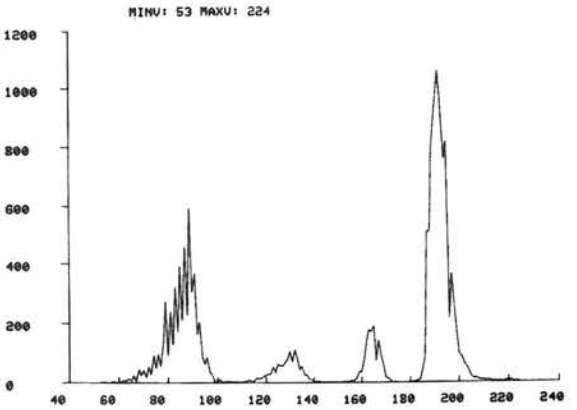
Fig. 6.7 Results showing that for a fixed ratio of α_1 and α_2 increasing both of them by a constant factor increases the speed of convergence. In each figure FACT = 1 is taken





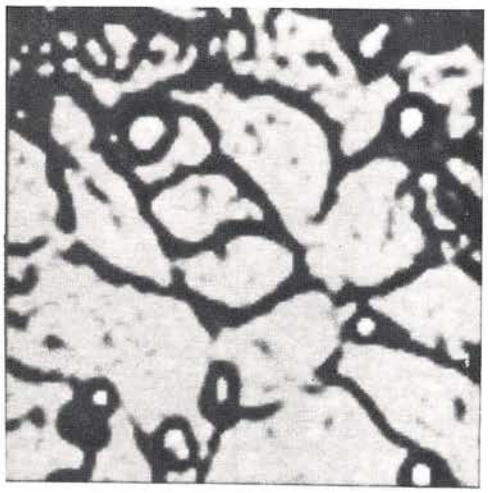
(a) Iteration 1 (b) Histogram of fig. 6.8(a)

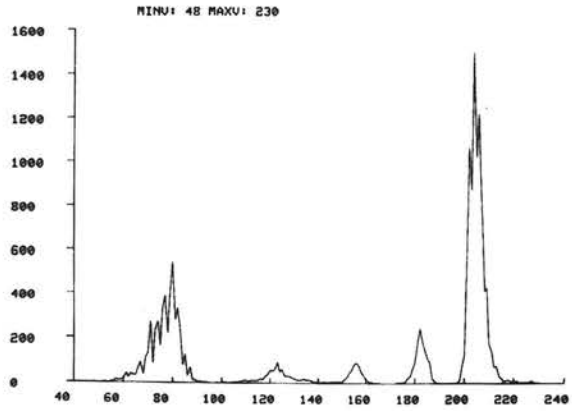




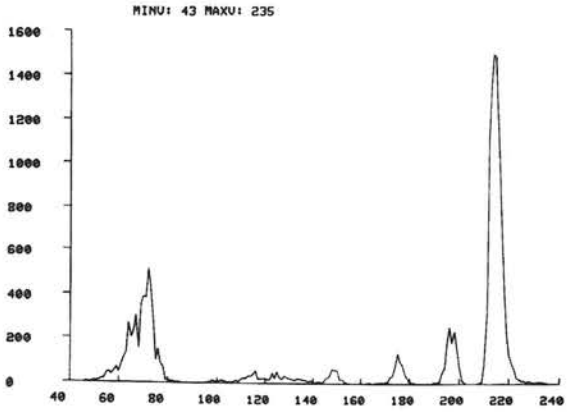
(c) Iteration 3 (d) Histogram of fig. 6.8(c)

Fig. 6.8 Results of Gradient Relaxation method at various iterations and corresponding histograms for the cell image. FACT = 0.9, α_1 = 0.2, α_2 = 0.1



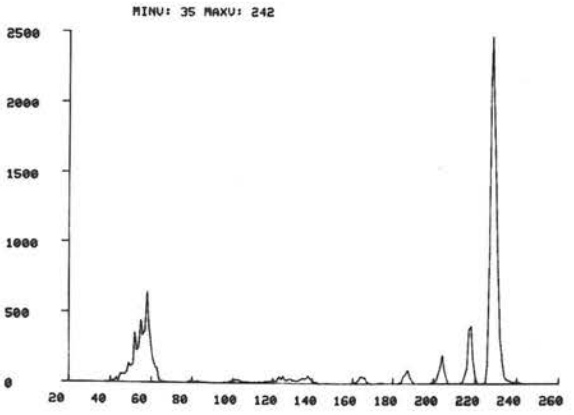


(e) Iteration 4 (f) Histogram of fig. 6.8(e)

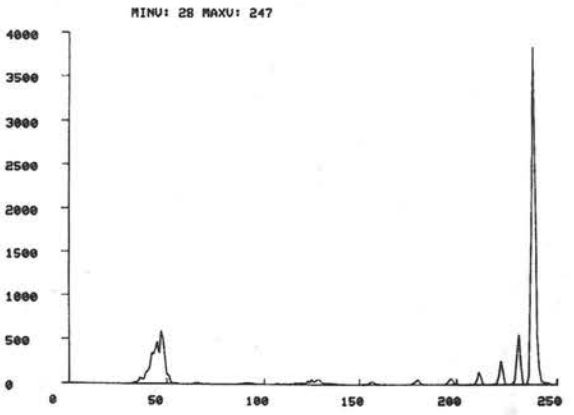


(g) Iteration 5 (h) Histogram of fig. 6.8(g)

Fig. 6.8 (CONTINUED)

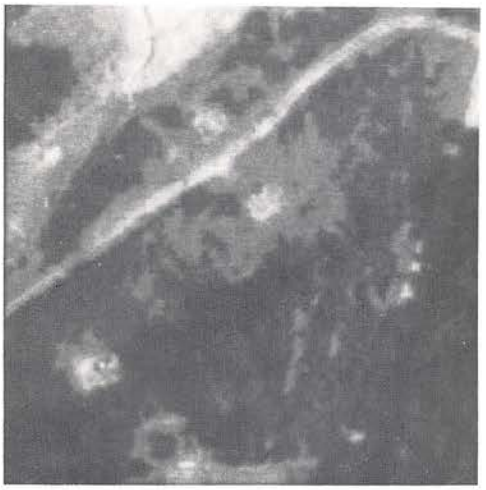


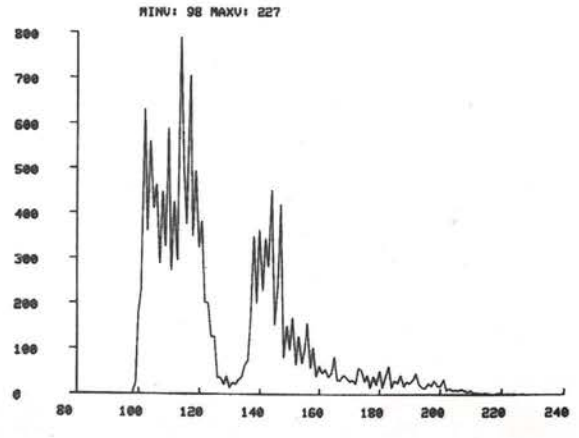
(i) Iteration 7 (j) Histogram of fig. 6.8(i)



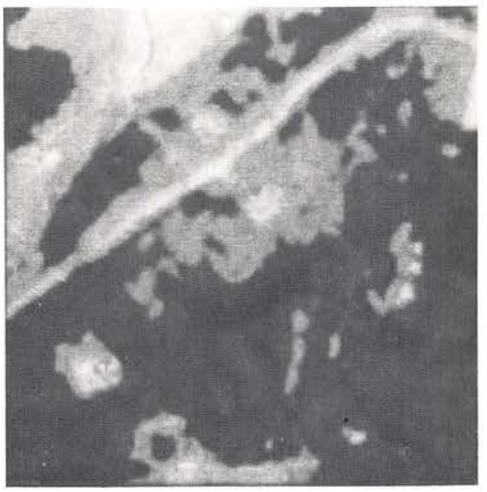
(k) Iteration 9 (l) Histogram of fig. 6.8(k)

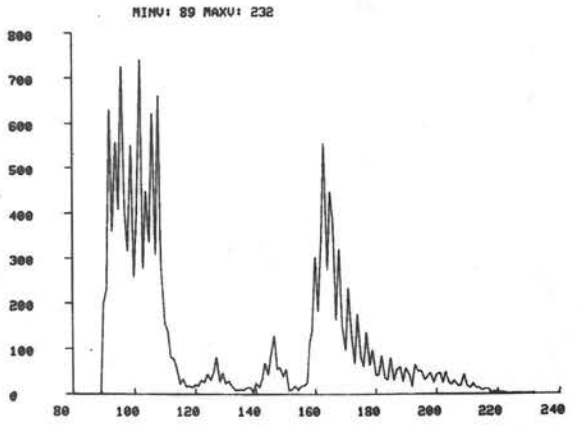
Fig. 6.8 (CONTINUED)





(a) Iteration 1 (b) Histogram of fig. 6.9(a)

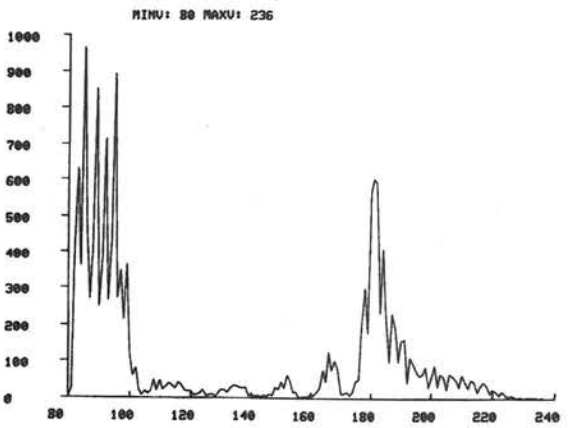




(c) Iteration 3 (d) Histogram of fig. 6.9(c)

Fig. 6.9 Results of Gradient Relaxation method at various iterations and corresponding histograms for the aerial image. FACT = 1, α_1 = 0.1, α_2 = 0.5

(e) Iteration 5



(f) Histogram of fig. 6.9(e)

Fig. 6.9 (CONTINUED)

value of FACT does not affect the value of the criterion, hence the speed of convergence (see Fig. 6.10) very much, but it affects the segmentation results as expected. Fig. 6.11 shows the gradient relaxation results at various iterations for different values of FACT when $\alpha_1 = \alpha_2 = 0.5$ for the cell image

b. Nonlinear Probabilistic Relaxation Algorithm

Rosenfeld [6-7] has used the nonlinear probabilistic relaxation algorithm for thresholding. In this algorithm the probability of labeling unit a_i with label λ_k at the n+1st iteration is given by

$$p_i^{(n+1)}(\lambda_k) = \frac{p_i^{(n)}(\lambda_k) q_i^{(n)}(\lambda_k)}{\sum_{\ell=1}^{2} p_i^{n}(\lambda_\ell) q_i^{(n)}(\lambda_\ell)}, \qquad \begin{matrix} k = 1,2 \\ i = 1,\ldots,N \end{matrix} \tag{6.18}$$

Assuming the same compatibility function and the initial assignment of probabilities as in the discussion of gradient technique, the results on the images shown in Fig. 6.1 are illustrated in Fig. 6.12 and 6.13. It can be seen from the histogram plots that in the first few iterations histogram equalization takes place resulting in the enhanced image and then black and white classes get separated. However, in this method unlike the gradient

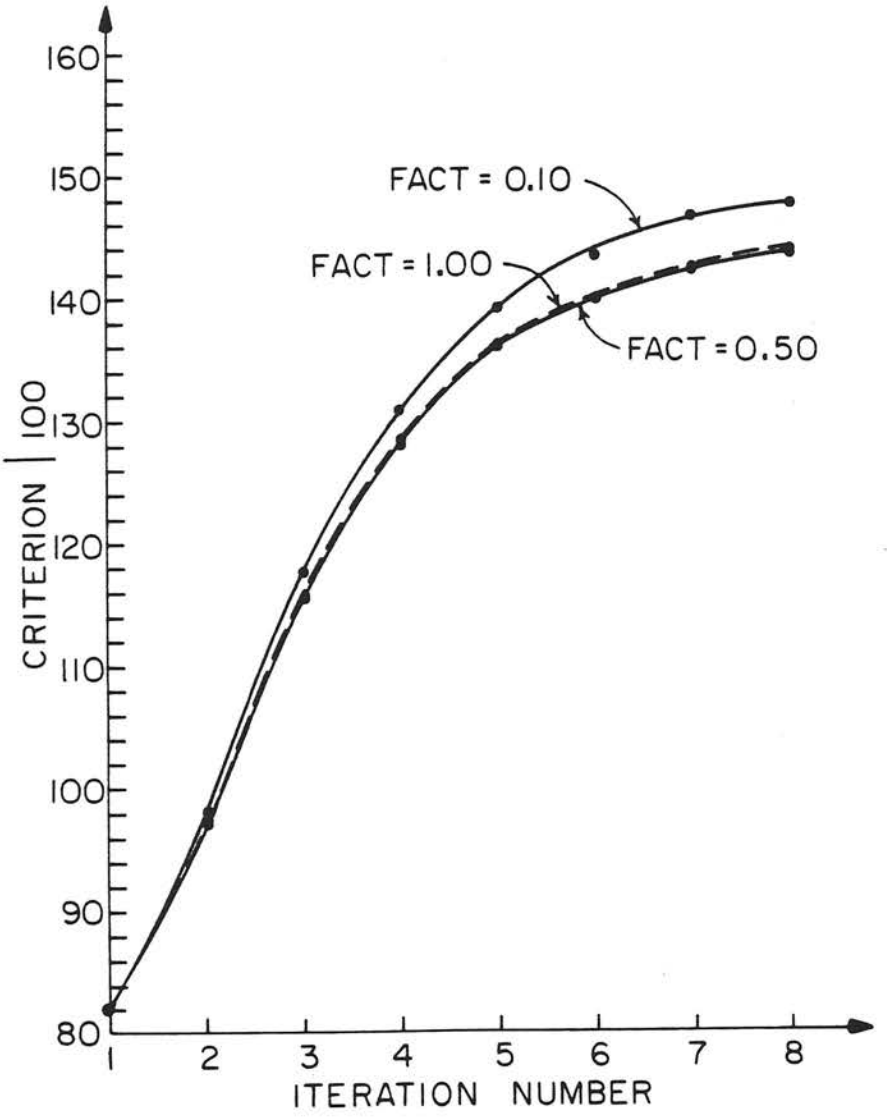
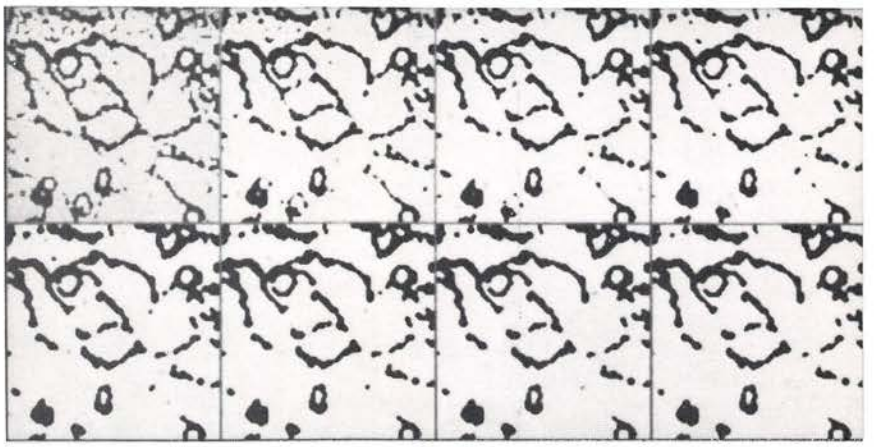
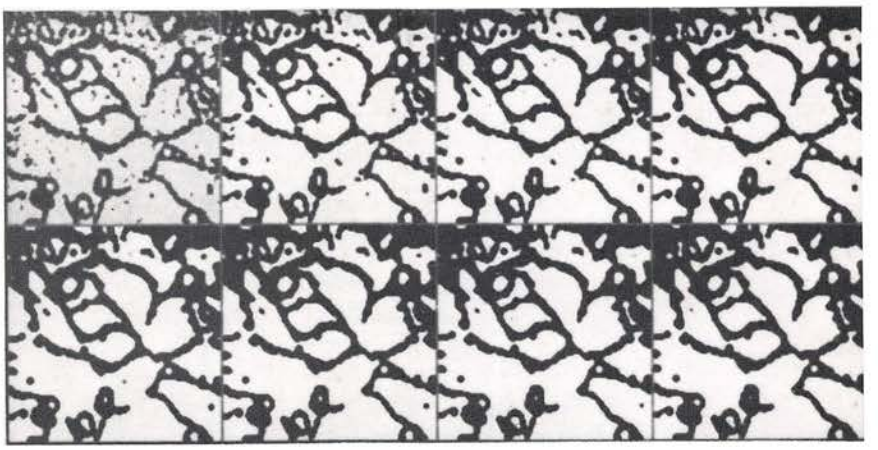


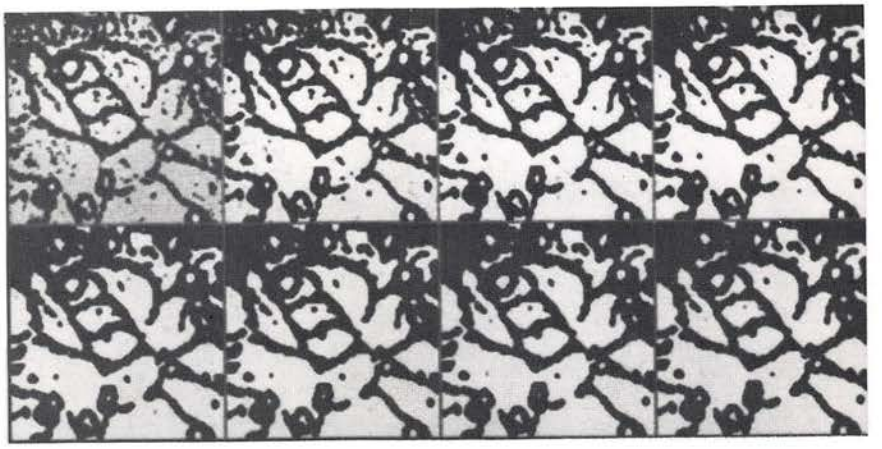
Fig. 6.10 Variation of the criterion (6.1) with the iteration number for various values of FACT for the cell image. $\alpha_1 = \alpha_2 = 0.50$ in all cases



(a) FACT = 0.1

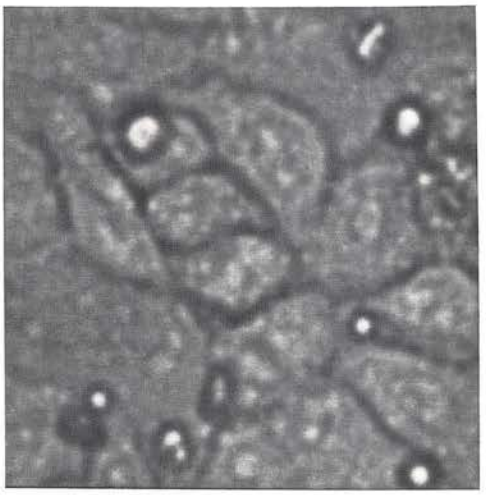


(b) FACT = 0.5

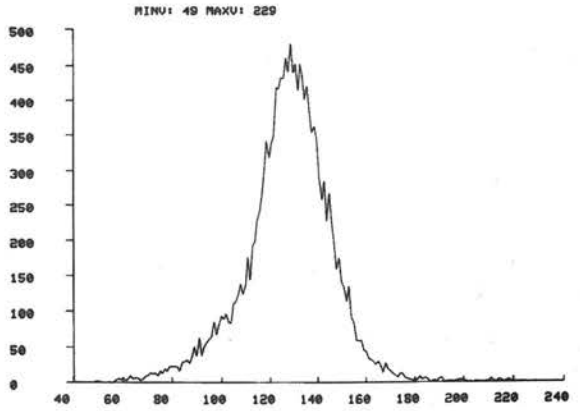


(c) FACT = 1.0

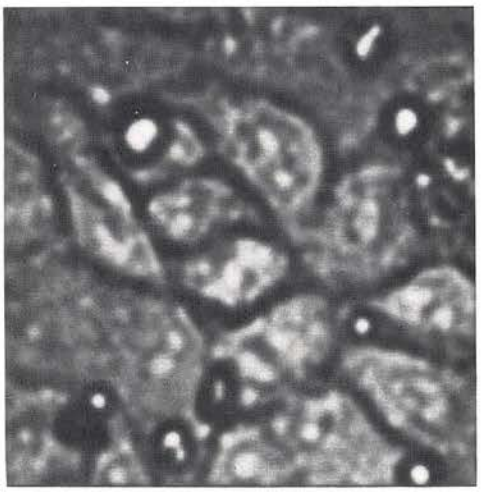
Fig. 6.11 Gradient relaxation results at various iterations for different values of FACT for the cell image. $\alpha_1 = \alpha_2 = 0.5$ in all the 3 cases



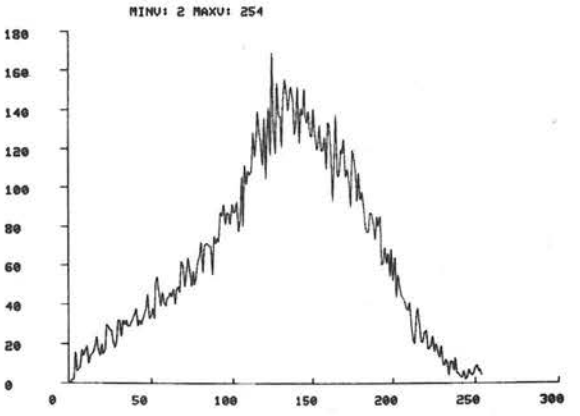
(a) Iteration 1



(b) Histogram of fig. 6.12(a)

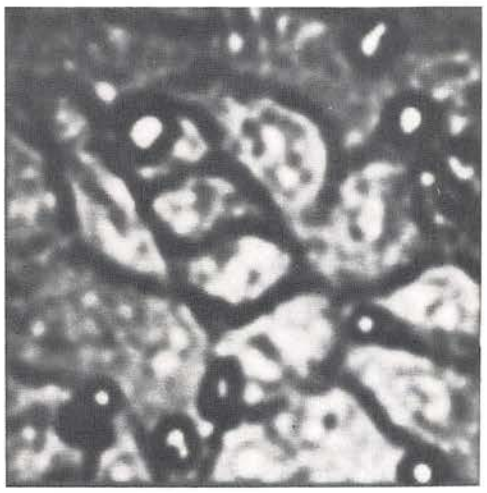


(c) Iteration 3

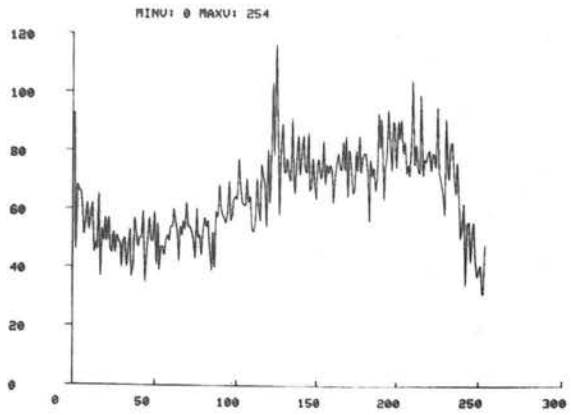


(d) Histogram of fig. 6.12(c)

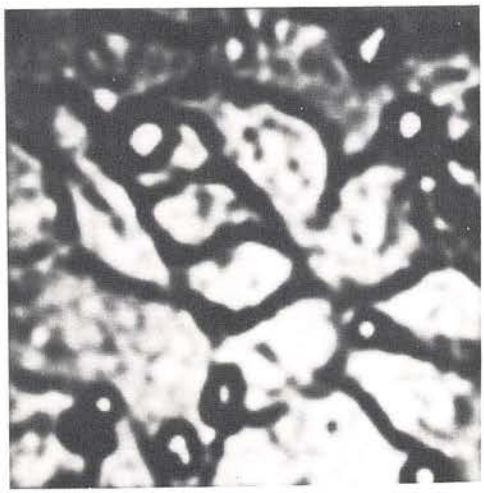
Fig. 6.12 Results of Nonlinear relaxation method at various iterations and corresponding histograms for the cell image. FACT = 0.9



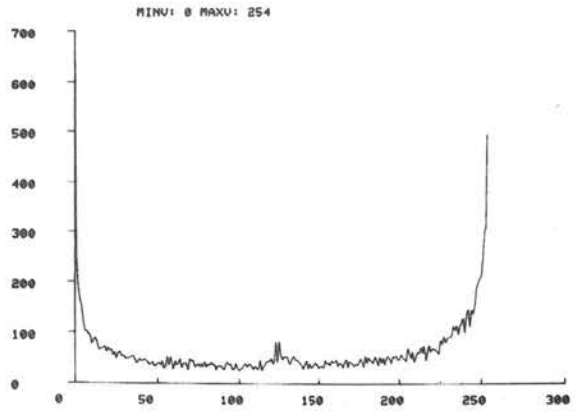
(e) Iteration 4



(f) Histogram of fig. 6.12(e)

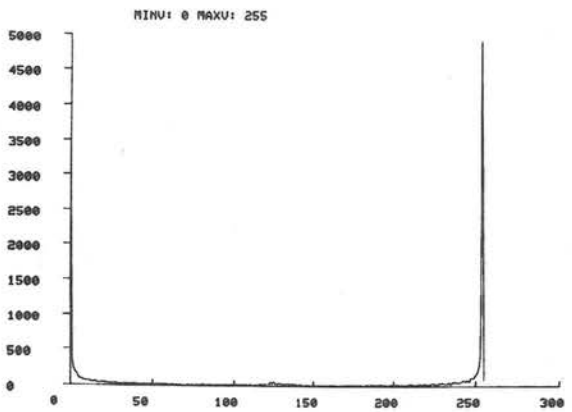


(g) Iteration 5

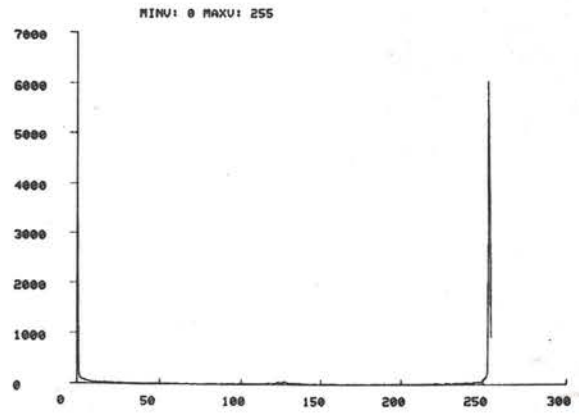


(h) Histogram of fig. 6.12(g)

Fig. 6.12 (CONTINUED)



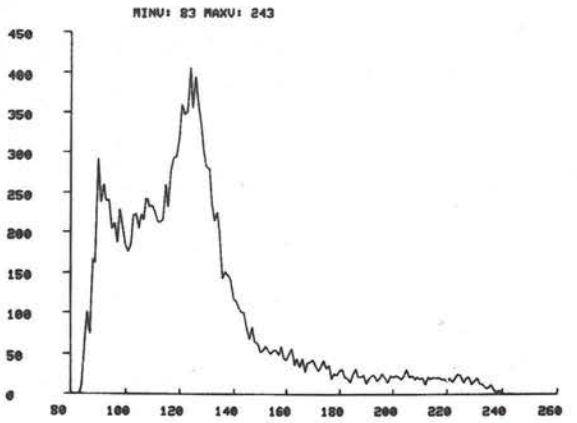
(i) Iteration 7 (j) Histogram of fig. 6.12(i)



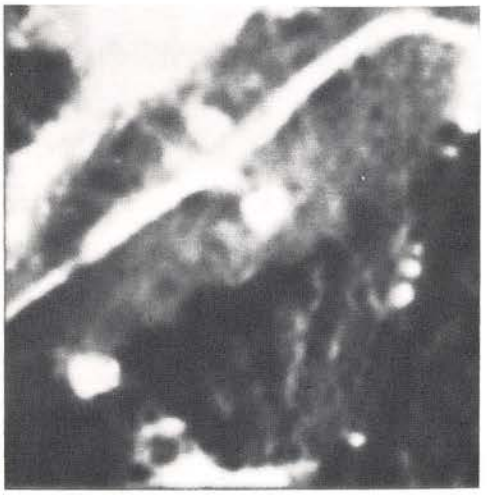
(k) Iteration 9 (l) Histogram of fig. 6.12(k)

Fig. 6.12 (CONTINUED)

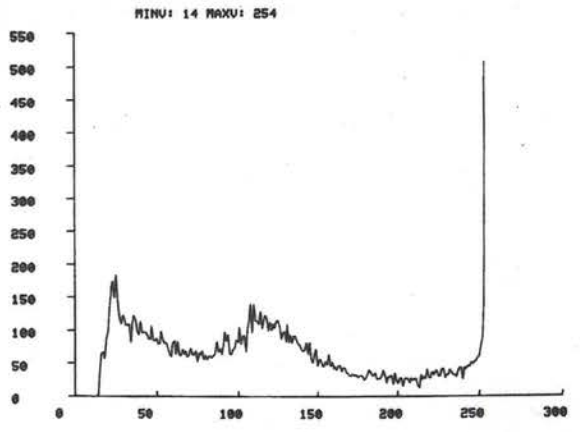
(a) Iteration 1



(b) Histogram of fig. 6.13(a)



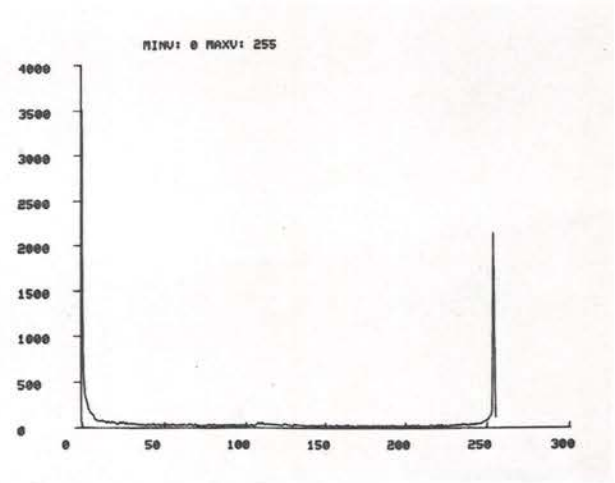
(c) Iteration 3



(d) Histogram of fig. 6.13(c)

Fig. 6.13 Results of Nonlinear relaxation method at various iterations and corresponding histograms for the aerial image. FACT = 1

(e) Iteration 5



(f) Histogram of fig. 6.13(e)

Fig. 6.13 (CONTINUED)

method we do not have any control on where we converge and how fast we converge.

6.4 Conclusion

From the images and histograms shown in Figs. 6.1, 6.7, 6.12 and 6.13 two observations can be made. First, the gradient and nonlinear relaxation methods provide comparable segmentation results. Secondly, it is not possible to compare these two methods directly because they do not converge to exactly the same limit. However, as illustrated in figs. 6.4, 6.7 and 6.10 it is only the gradient method that provides the control over the relaxation process by choosing the α_1, α_2 and FACT parameters which can be tuned to obtain the desired segmentation results at a faster rate.

The gradient relaxation method has been applied to a number of aerial and biomedical pictures having unimodal distributions and gave good segmentation results. In the cancer cell mitosis example presented in Chapter 3 we also used this method. In the above presentation we have considered the 8-neighborhood for both classes λ_1 and λ_2 for both methods. If we consider 8-neighborhood for class λ_1 and 4-neighborhood for class λ_2, the gradients are obtained as,

$$\frac{\partial C}{\partial p_i(\lambda_1)} = q_i(\lambda_1) + \sum_{j \varepsilon V_i(\lambda_1)} \frac{1}{8D_j}[p_j(\lambda_1) - \vec{p}_j \cdot \vec{q}_j] \qquad (6.19)$$

and

$$\frac{\partial C}{\partial p_i(\lambda_2)} = q_i(\lambda_2) + \sum_{j \varepsilon V_i(\lambda_2)} \frac{1}{8D_j}[p_j(\lambda_2) - \vec{p}_j \cdot \vec{q}_j] \qquad (6.20)$$

where

$$q_i(\lambda_1) = \frac{\frac{1}{8} \sum_{j \varepsilon V_i(\lambda_1)} p_j(\lambda_1)}{D_j}$$

$$q_i(\lambda_2) = \frac{\frac{1}{4} \sum_{j \varepsilon V_i(\lambda_2)} p_j(\lambda_2)}{D_j}$$

$$Dj = \frac{1}{8} \sum_{j \varepsilon V_i(\lambda_1)} p_j(\lambda_1) + \frac{1}{4} \sum_{j \varepsilon V_i(\lambda_2)} p_j(\lambda_2)$$

and $V_i(\lambda_1)$ and $V_i(\lambda_2)$ are the 8 and 4 nearest neighbors of a_i associated with class λ_1 and λ_2 respectively.

After finding the projection of the gradient as before, iterative equations similar to (6.14) and (6.18) can be obtained. Use of these equations gave results similar to those repeated in this chapter.

Also in this study we have considered only two classes. In scenes with many different objects as in the case of aerial photographs, there are more classes. In

such situation two generalizations are possible. In the first we simply extend the method to more classes, which are known a priori. In the second case, we apply the procedure outlined above to each segmented region iteratively in a tree structure until the area of a region has reduced to a specified limit. If a region is homogeneous in intensity, then we will not expect to find two peaks and will stop iterating after a predefined number of iterations.

In the next chapter we shall present the summary of the research work and suggestions for future research.

CHAPTER 7

CONCLUSIONS

7.1 Summary

In this dissertation we presented new results in the areas of shape matching of nonoccluded and occluded two dimensional objects, 3-D data acquisition and geometric processing, surface approximation by polygons, shape matching of three-dimensional objects and the segmentation of images having unimodal distributions. The basic technique used is a stochastic labeling technique with various extensions. To illustrate the capabilities of the techniques we presented examples taken from synthetic, biological, aerial and industrial images. In the following we shall present a summary of the research work and in the next section suggestions for future research.

Shape matching is viewed as a segment matching problem, where a piece of a shape is recognized as an approximate match to a part of a larger shape. The gradient relaxation algorithm developed by Faugeras and Berthod [7-1] based on the maximization of a criterion function has been extended to do the shape matching of two

dimensional objects in a hierarchical manner. The hierarchical nature of the algorithm reduces the computation time and uses results at low levels to speed up and improve the accuracy of results obtained at higher levels. The two-dimensional shapes are represented by their polygonal approximation. As compared to the previous studies in 2-D shape matching, the technique developed here has been applied not only to the synthetic images, but also to the real images taken from the aerial, industrial parts and biological fields, where the matching is done after using the actual segmentation methods. The method allows for changes in scale, rotation, translation and significant changes in shape. The results of shape matching are used to compute the rotation. Although the technique is computationally more costly than the other feature based or correlation techniques, it is more robust and flexible. It allows the parallel implementation in hardware.

The hierarchical stochastic labeling technique described above has been extended to do the shape matching of two-dimensional occluded objects. This is done by combining the gradient projection method and penalty function approach. For each of the objects participating in the occlusion, there is a hierarchical process. These processes are executed in parallel and are coordinated in

such a way that the same segment of the apparent object, formed as a result of occlusion of two or more actual objects, is not matched to the segments of different actual objects. Results are presented on synthetic objects and the sequence of microscope and industrial objects.

Generation of 3-D object descriptions in terms of polygons which approximate the surface of an object has been studied. A new technique for the extraction of planar faces from a set of 3-D points is proposed. The method is a sequential region growing algorithm. It is not applied directly to range images, but rather to a set of points. It is not restricted to a single range data image, but applicable to a composite object and does not require the ordering of points. It finds the convex faces of the object, but the information exists to merge convex parts of nonconvex faces. It is not directly related to how the 3-D surface points were obtained, but is tied to the sampling distances between points on the object. Such a representation should be of use not only to 3-D scene analysis, but also in computer graphics, the representation of terrain data obtained by synthetic aperture radar, etc. The method is used to obtain a 3-D model of an object by combining the object points from a sequence of range data images corresponding to various

views of the object and applying the necessary transformations. Results are presented on a fairly complex industrial object.

The problem of shape matching of real world 3-D objects has been attacked by using the hierarchical stochastic labeling technique used in 2-D. In 2-D the matching was "segment matching," but now we have "face matching." A system for 3-D scene analysis is presented. This is useful for shape matching of real world 3-D objects. Techniques used for acquiring 3-D data and geometric processing are presented. Issues related to representation and modelling of 3-D objects are discussed. Using several examples of an automobile casting, it is shown that hierarchical stochastic labeling technique can be successfully used to accomplish partial shape recognition in 3-D. The results of shape matching are used to compute the orientation of the object in space.

Unimodal histograms are typically obtained when the image consists of a large background area with other small but significant regions. Particularly, in the biomedical area the extraction of the boundaries of various types of cells is complicated by the fact that cells are very close together, their boundaries are poorly defined and the gray level histogram is unimodal. Similarly unimodal

histograms are obtained for the aerial pictures, because the range of intensities of different objects overlap. The technique used for segmentation is again the stochastic labeling technique used for shape matching. It provides the control over the relaxation process by choosing three parameters which can be used to obtain the desired segmentation results at a faster rate. This is the major advantage of the technique compared to the classical nonlinear relaxation method. Aerial and biological examples are presented.

7.2 Suggestions for Future Research

This section will present several directions for future research in shape matching of two and three dimensional objects, segmentation of images and use of these concepts in the analysis of a real sequence of images.

Shape Matching in 2-Dimensions

1. We presented four methods for the computation of compatibilities. The fourth method computes an exact transformation, but allows changes only in scale, rotation and translation. A more general transformation may be computed and its effect on labeling can be investigated.

2. Providing an indexing capability to the shape matching algorithm. This means that selecting a suitable subclass of models to match with the observed object so that the matching with each known model is not necessary. This is necessary if the number of models is large.

Shape Matching in 3-Dimensions

1. It is possible to extend the techniques like iterative end pont fit [7-2] from 2-D to 3-D with suitable modifications for finding the planar faces in a single range view. Then the problem is how to combine various views of the object so as to get a 3-D model of the object.

2. Our matching technique based on the stochastic labeling method uses features of the faces. Other techniques for shape matching based on matching the polygons would be interesting.

3. Obtain more refined face boundaries in the technique for surface approximation by polygons once the neighboring faces are known. This can be done by finding the line of intersection of the planes of these faces. Techniques for the possible merger of convex faces into nonconvex faces and the

effectiveness of the technique presented here for surface approximation by polygons on terrain data taken by synthetic aperture radar should be undertaken.

Segmentation

We presented segmentation technique only for two classes. In scenes with many different objects as in the case of aerial images there are more classes although the histogram may have only one peak because the range of intensities for each object will probably overlap with the range of other objects. Two generalizations are possible. In the first we simply extend the method to more classes, which are known a priori. In the second case, we apply the two class technique to each segmented region in a tree structure until the area of a region is reduced to a specified limit. If the region is homogeneous in intensity, we shall not expect to find two peaks and stop after a predefined number of iterations. An experiment to investigate the sensitivity of the two class technique to homogeneous regions with varying amount of noise would be interesting.

Motion Analysis

Fig. 7-1 shows a system for analyzing a sequence of

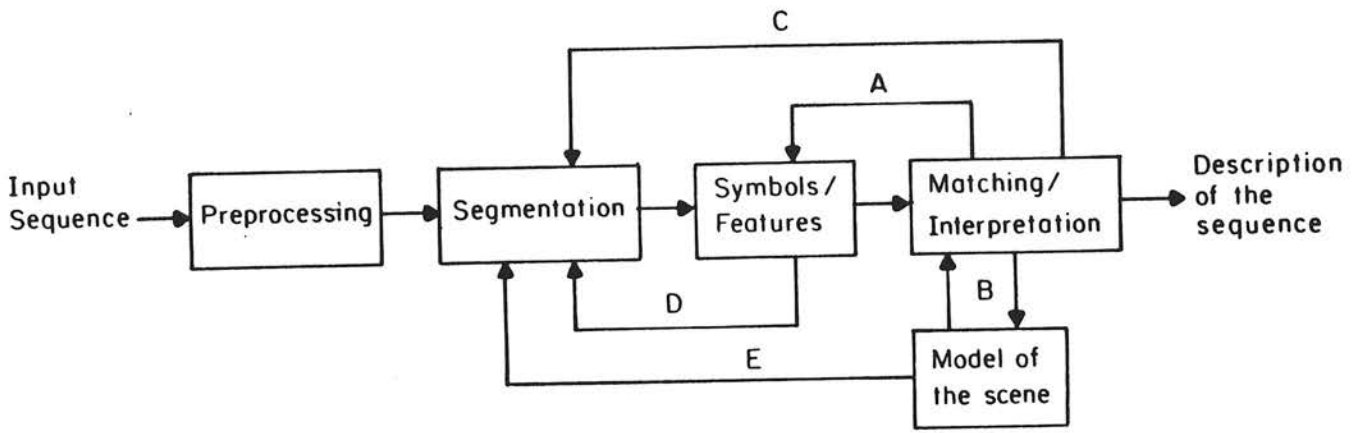


Fig. 7.1 A system for the analysis of image sequences

images [7-3, 7-4] where a number of objects are moving at random and changing their shape. Segmentation and shape matching/modelling blocks are the key components of this system. Distributed feedback (paths labeled as A to E) is used to obtain the control on symbols/features and segmentation. On the basis of the fact that the predicted model of the scene at a particular time matches with the scene at that time, we do simple or complicated processing for motion and shape analysis in a hierarchical manner. Also note that the segmentation varies with the dynamics of the scene. Shape matching and segmentation techniques presented in this work can be tried in analyzing a sequence of images using this system. Particularly the shape matching algorithm should be tested in the system of fig. 7-1 and be modified to account for the prediction of shape changes depending upon the requirement of the task domain.

APPENDIX A

PROJECTION OPERATOR

In this appendix we describe in brief the projection operator $P_i^{(n)}$ needed in the gradient projection method presented in Chapter 2. Details of the method can be found in [A-1 to A-3].

We want to obtain the projection of the gradient at point $\vec{p}_i^{(n)}$ of the closed convex region R_i defined by:

$$\sum_{\ell=1}^{L} p_i(k) = 1, \quad p_i(k) \geq 0, \; k = 1,\ldots,L \qquad \text{(A-1)}$$

We drop the indices i and n which are irrelevant to the definition of the projection operator. There are mainly two situations.

1. None of the components of $\vec{p}$ are zero. The projection operator P is simply the projection on the hyperplane of equation (A-1). That is,

$$P\vec{g} = \vec{g} - \left(\frac{1}{L}\sum_{k=1}^{L} g_k\right)\vec{v} \qquad \text{(A-2)}$$

where, $\vec{g} = [g_1,\ldots,g_L]^T$ and $\vec{v} = [1,1,\ldots,1]^T$.

2. Some of the components of $\vec{p}$ are equal to zero. Let K be the set of indices for which $p(k) = 0$. We compute the projection $\vec{w}$ of the gradient g on the intersection of hyperplanes,

$$\sum_{k=1}^{L} p(k) = 1,\ p(k) \geq 0,\ k = 1,\ldots,L \text{ and } k \notin K,\ p(k) = 0, \text{ for } k \in K \qquad \text{(A-3)}$$

Let P_K be the corresponding projection operator. Two cases arise depending upon whether $\vec{w}$ is zero or not.

Case 1. If $\vec{w}$ is not zero, projection operator $P = P_K$ is taken. Thus, we do not desaturate (change) the constraints $p(k) = 0$, $k \in K$.

Case 2. If $\vec{w}$ is zero, there is no direction of ascent in the intersection of hyperplanes defined by (A-3). We find the index $\ell \in K$ for which the projection $\vec{w}$ of $\vec{g}$ on the intersection of hyperplanes,

$$\sum_{k=1}^{L} p(k) = 1,\ p(k) \geq 0,\ k = 1,\ldots,L \text{ and } k \notin K-\{\ell\},\ p(k) = 0, \text{ for } k \in K-\{\ell\} = K' \qquad \text{(A-4)}$$

has the largest negative value for w_ℓ. Let $P_{K'}$ be the corresponding projection. If there is no index $\ell \in K$ such that w is negative, there is no direction of ascent at the point p, and we have reached a constrained maxima. If

there is such an index ℓ we take $P = P_{K'}$. Thus at the next iteration constraint $p(\ell) = 0$ is desaturated (changed).

APPENDIX B

SYSTEM USED TO OBTAIN SEQUENCE OF CELL IMAGES

In this appendix we describe the system used to obtain the sequence of cell images and the contents of these images.

Fig. B.1 shows the real time video acquisition and digital display system. The system was recently built at the USC Image Processing Institute [B-1]. The system under the control of a PDP 11/34 processor will acquire, digitize and store a number of consecutive subframes of video images from the monochrome TV camera. During and after the acquisition phase, the acquired image can be displayed on the monochrome TV monitor. After the acquisition phase, the acquired image subframes can be transferred from the image memory (512x512x8 bits) to the PDP 11/34. An image or image subframes can also be transferred from the PDP 11/34 to the image memory. The maximum size of the image subframes (window area) is 140x400 pixels. The minimum sampling interval between two frames is of the order of 0.75 seconds. It is limited by the size of the subframe and the amount of time required

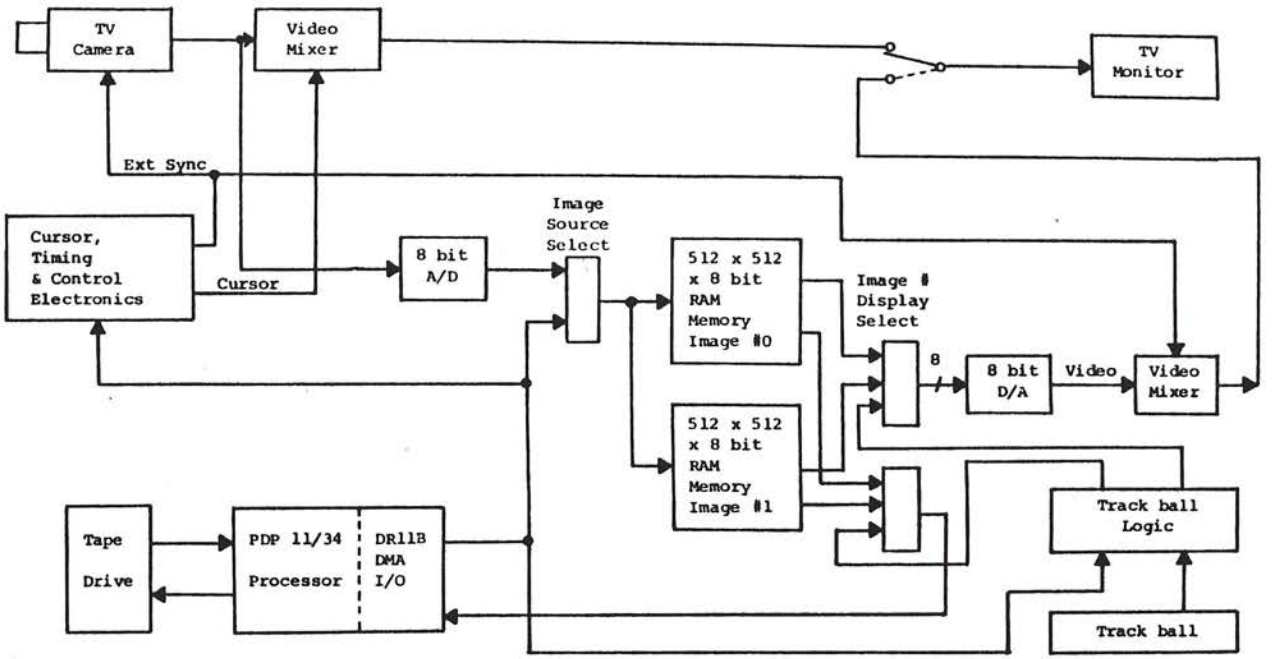


Fig. B.1 Real time video acquisition and digital display system

to transfer and store the image into a file on the disk or tape.

Programs were written on PDP 11/34 for the system shown in Fig. B.1, which allow the user to store the subframes of specific sizes at a specified rate. Odd and even lines in the subframe can be stored as separate files on the disk for interlacing to be carried out at a later time or they can be interlaced before storing on the disk. Images so obtained can be put back on the display for observing a part of the sequence. Images obtained are transferred to the PDP-10 after a format conversion.

In obtaining the cell images using this system, the long term objective was to analyze the behavior of lymphocytes in the presence of cancer cells in vitro. The TV camera in Fig. B.1 is connected to a NIKON phase contrast microscope. A confluent monolayer of human skin cancer cells was grown on microslides. Using this slide a Sykes-Moore Chamber was formed and filled with a RPMI 1640 media with 10% serum. Lymphocytes were isolated from fresh human blood by a gelative technique and their concentration was about 90%. A small number of these lymphocytes were injected into the chamber. The whole system was incubated at $37^{o}C$. The movement of lymphocytes was recorded at a 30 second to 300 second interval in the

images (128x128 pixels) over a period of 48 hours. 128x128 pixels image corresponds to an area of 60 μm X 60μm of the slide at 200 magnification. The representative pictures are shown in Chapters 3 and 6.

REFERENCES

[1-1] O.D. Faugeras, "Optimization Techniques in Image Analysis," Proc. of the 4th Int. Conf. on Analysis and Optimization of Systems, Lecture notes in Control and Information Sciences, Springer-Verlag, 1980, pp. 790-823.

[1-2] L. Davis and A. Rosenfeld, "Cooperative Processes for Low-Level Vision: A Survey," Dept. Comp. Sci., Univ. of Texas, Austin, Tech. Rep. 123, Jan. 1980.

[1-3] D.L. Waltz, "Generating Semantic Descriptions from Drawings of Scenes and Shadows," MIT technical report AI271, Nov. 1972. For briefer versions see D. Waltz, "Understanding the Drawings of Scenes and Shadows, "in P.H. Winston, Ed., The Psychology of Computer Vision, McGraw Hill, New York, 1975, pp. 19-91.

[1-4] A. Rosenfeld, R. Hummel and S.W. Zucker, "Scene Labeling by Relaxation Operations," IEEE Trans. Systems, Man and Cybernetics, vol. SMC-6, pp. 420-433, 1976.

[1-5] H.G. Barrow and J.M. Tenenbaum, "MSYS: A System for Reasoning About Scenes," Tech. Note 121, Artificial Intelligence Center, SRI Intl., Menlo Park, CA, 1976.

[1-6] D. Marr, T. Poggio and G. Palm, "Analysis of a Cooperative Stereo Algorithm," Biol. Cybernetics, Vol. 28, pp. 223-239, 1977.

[1-7] S.W. Zucker, and J.L. Mohammed, "Analysis of Probabilistic Relaxation Labeling Processes," Proc. IEEE Comp. Soc. Conf. on Pattern Recognition and Image Processing, Chicago, 1978, pp. 307-312.

[1-8] S.W. Zucker, E.V. Krishnamurthy and R.L. Haar, "Relaxation Processes for Scene Labeling: Convergence, Speed and Stability," IEEE Trans. on System, Man, and Cybernetics, Vol. SMC-8,

pp. 41-48, Jan. 1978.

[1-9] O.D. Faugeras and M. Berthod, "Improving Consistency and Reducing Ambiguity in Stochastic Labeling: An Optimiztion Approach," to appear in IEEE Trans. Pattern Analysis and Machine Intelligence.

[1-10] O.D. Faugeras and M. Berthod, "Using Context in the Global Recognition of a Set of Objects: An Optimization Approach," Proceedings of the 8th world computer congress (IFIP 80), Tokyo, pp. 695-698.

[1-11] O.D. Faugeras and M. Berthod, "Scene Labeling: An Optimization Approach," Proc. IEEE Comp. Sci. Conf. on Pattern Recognition and Image Processing, Aug. 6-8, 1979, pp. 318-326.

[1-12] S. Ullman, "Relaxation and Constrained Optimization by Local Processes," Computer Graphics and Image Processing , Vol. 10, pp. 115-125, 1979

[1-13] A. Rosenfeld, "Image Understanding Project Status Report," Proc. DARPA Image Understanding Workshop, April 1979, pp. 14-24.

[2-1] L.S. Davis, "Shape Matching Using Relaxation Techniques," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI-1, pp. 60-72, Jan. 1979.

[2-2] H. Freeman, "On the Encoding of Arbitrary Geometric Configurations," IRE Trans. Electron. Computers, Vol. EC-10, pp. 260-268, 1961.

[2-3] H. Freeman, "Computer Processing of Line-drawing Images," Computing Surveys, Vol. 6, pp. 57-97, March 1974.

[2-4] G.H. Granlund, "Fourier Preprocessing for Hand Print Character Recognition," IEEE Trans. on Computers, Vol. C-21, pp. 195-201, Feb. 1972.

[2-5] E. Persoon and K.S. Fu, "Shape Discrimination Using Fourier Descriptors," IEEE Trans. Systems, Man and Cybernetics, Vol. SMC-7, pp. 170-179, March 1977.

[2-6] C.W. Richard, Jr. and H. Hemami, "Identification

of Three-dimensional Objects Using Fourier Descriptors of the Boundary Curve," IEEE Trans. Systems, Man and Cybernetics, Vol. SMC-4, pp. 371-380, July 1974.

[2-7] C.T. Zahn and R.Z. Roskies, "Fourier Descriptors for Plane Closed Curves," IEEE Trans. on Computers, Vol. C-21, pp. 269-281, March 1972.

[2-8] S.A. Dudani et al., "Aircraft Identification by Moment Invariants," IEEE Trans. on Computers, Vol. C-26, pp. 39-46, Jan. 1977.

[2-9] M.K. Hu, "Visual Pattern Recognition by Moment Invariants," IEEE Trans. on Information Theory, Vol. IT-8, pp. 179-187, Feb. 1962.

[2-10] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, John Wiley & Sons Inc., 1973.

[2-11] K. Price and R. Reddy, "Matching Segments of Images," IEEE Trans. on Pattern Analysis and Machine Intelligence Vol. PAMI-1, pp. 110-116, Jan. 1979.

[2-12] O.D. Faugeras and K. Price, "Semantic Description of Aerial Images using Stochastic Labeling," To appear in IEEE Trans. on Pattern Analysis and Machine Intelligence.

[2-13] L.S. Davis and A. Rosenfeld, "Application of Relaxation Labeling, 2: Spring-loaded Template Matching," Proc. 3rd Int. Joint Conf. on Pattern Recognition, 1976, pp. 591-597.

[2-14] L.S. Davis and A. Rosenfeld, "Hierarchical Relaxation for Waveform Parsing," in Computer Vision Systems, A.R. Hanson and E.M. Riseman (Eds.), Academic Press, 1978, pp. 101-109.

[2-15] T. Henderson, "Hierarchical Constraint Processes for Shape Analysis," Ph.D. Thesis, Dept. of Computer Sci., Univ. of Texas, Austin, 1979.

[2-16] W.H. Tsai and K.S. Fu, "A Syntactic-Statistical Approach to Recognition of Industrial Objects," Proc. 5th International Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 251-259.

[2-17] L.S. Davis, "Hierarchical Relaxation for Shape

Analysis," Proc. IEEE Comp. Soc. Conf. on Pattern Recognition and Image Processing, 1978, pp. 275-279.

[2-18] T. Pavlidis, "The Use of Syntactic Shape Analyzer for Contour Matching," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-1, pp. 307-310, July 1979.

[2-19] L. Kitchen and A. Rosenfeld, "Discrete Relaxation for Matching Relational Structures," IEEE Trans. Systems, Man and Cybernetics, Vol. SMC-9, pp. 869-874, Dec. 1979.

[2-20] L. Kitchen, "Relaxation Applied to Matching Quantitative Relational Structures," IEEE Trans. Systems, Man and Cybernetics, Vol. SMC-10, pp. 96-101, Feb. 1980.

[2-21] L.G. Shapiro, "A Structural Model of Shape," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, pp. 111-126, March 1980.

[2-22] T. Pavlidis and F. Ali, "A Hierarchical Syntactic Shape Analyzer," IEEE Trans. pattern Analysis and Machine Intelligence, Vol. PAMI-1, pp. 2-9, Jan. 1979.

[2-23] J.W. McKee and J.K. Aggarwal, "Computer Recognition of Partial Views of Curved Objects," IEEE Trans. on Computers, Vol. C-22, pp. 790-800, Sept. 1977.

[2-24] W.A. Perkins, "A Model Based Vision System for Industrial Parts," IEEE Trans. on Computers, Vol. C-27, pp. 126-143, February 1978.

[2-25] Y. Shirai, "Edge Finding, Segmentation of Edges and Recognition of Complex Objects," Proc. 4th Int. Conf. on Artificial Intelligence, 1975, pp. 674-681.

[2-26] M. Yachida and S. Tsuji, "A Versatile Machine Vision System for Complex Industrial Parts," IEEE Trans. on Computers, Vol. C-26, pp. 882-894, Sept. 1977.

[2-27] T. Pavlidis, "A Review of Algorithms for Shape Analysis," Computer Graphics and Image Processing, Vol. 7, pp. 243-258, 1978.

[2-28] O.D. Faugeras and M. Berthod, "Using Context in the Global Recognition of a Set of Objects: An Optimization Approach," Proc. 8th World Computer Congress (IFIP 80), Tokyo, pp. 695-698.

[2-29] O.D. Faugeras and M. Berthod, "Scene Labeling: An Optimization Approach," Proc. IEEE Conf. on Pattern Recognition and Image Processing, 1979, pp. 318-326.

[2-30] B. Bhanu and O.D. Faugeras, "Shape Matching Using Hierarchical Gradient Relaxation," USCIPI Report 990, Image Processing Inst. ,Univ. of Southern Calif. , Los Angeles, Oct. 1980.

[2-31] B. Bhanu and O.D. Faugeras, "Shape Recognition of 2-D Objects," Proc. 2nd Scandinavian Conf. on Image Analysis, Helnsiki, Finland, June 15-17, 1981.

[2-32] O.D. Faugeras and B. Bhanu, "Reconnaissance de formes planes par une methode hierarchique d'Etiquetage Probabiliste," To be presented at AFCET, 3 eme Congres, Reconnaissance Des Formes et Intelligence Artificielle, Sept. 16-18, 1981, Nancy, France.

[2-33] A. Rosenfeld, R. Hummel and S.W. Zucker, "Scene Labeling by Relaxation Operations," IEEE Trans. Systems, Man and Cybernetics, Vol. SMC-6, pp. 420-433, 1976.

[2-34] O.D. Faugeras and M. Berthod, "Improving Consistency and Reducing Ambiguity in Stochastic Labeling: An Optimization Approach," to appear in IEEE Trans. Pattern Analysis and Machine Intelligence.

[2-35] B. Bhanu and O.D. Faugeras, "Segmentation of Images Having Unimodal Distributions," submitted to the IEEE Trans. Pattern Analysis and Machine Intelligence, July 1980.

[2-36] K.E. Price, "Relaxation Matching Applied to Aerial Images," Proc. Image Understanding Workshop, Washington D.C., April 1981, pp. 22-25.

[2-37] J.B. Rosen, "The Gradient Projection Method for Nonlinear Programming, Part I. Linear Constraints," J. Soc. Indust. Appl. Math.,

Vol. 8, pp. 181-217, March 1960.

[2-38] A. Rosenfeld and E. Johnston, "Angle Detection on Digital Curves," IEEE Trans. Computers, Vol. C-22, pp. 875-878, Sept. 1973.

[2-39] L.S. Davis, "Understanding Shape: Angles and Sides," IEEE Trans. Computers, Vol. C-26, pp. 236-242, March 1977.

[2-40] T. Pavlidis and S. Horowitz, "Segmentation of Plane Curves," IEEE Trans. on Computers, Vol. C-23, pp. 860-870, 1974.

[2-41] L. Davis and A. Rosenfeld, "Cooperative Processes for Low-Level Vision: A Survey," Dept. Comp. Sci., Univ. of Texas, Austin, Tech. Rep. 123, Jan. 1980.

[2-42] S. Peleg, "Monitoring Relaxation Algorithms Using Labeling Evaluations," Proc. 5th International Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 54-57.

[2-43] R. Ohlander, K. Price and D.R. Reddy, "Picture Segmentation Using a Recursive Region Splitting Method," Computer Graphics and Image Processing, Vol. 8, pp. 313-333, 1978.

[2-44] K.E. Price, "Change Detection and Analysis in Multi-Spectral Images," Ph.D. Thesis, 1976, Dept. of Comp. Sci., Carnegie-Mellon Univ. PA.

[2-45] J.F. Reiser, Ed., "SAIL", Memo AIM-289, Stanford Artificial Intelligence Laboratory, Computer Sci. Dept. Report no. STAN-CS-76-574, August 1976.

[3-1] A. Rosenfeld and E. Johnston, "Angle Detection on Digital Curves," IEEE Trans. on Computers, Vol. C-22, pp. 875-878, 1973.

[3-2] W.N. Martin and J.K. Aggarwal, "Dynamic Scene Analysis," Computer Graphics and Image Processing, Vol. 7, pp. 356-374, 1978.

[3-3] H.H. Nagel, "Analysis Techniques for Image Sequences," Proc. 4th Int. Joint Conf. on Pattern Recognition, Kyoto, Japan, Nov. 1978, pp. 186-211.

[3-4] M. Yachida, M. Asada and S. Tsuji, "Automatic

Motion Analysis System of Moving Objects from the Records of Natural Processes," Proc. 4th Int. Joint Conf. on Pattern Recognition, Kyoto, Japan, Nov. 1978, pp. 726-730.

[3-5] J.K. Aggarwal and R.O. Duda, "Computer Analysis of Moving Polygonal Images," IEEE Trans. Computers, Vol. C-24, pp. 966-976, Oct. 1975.

[3-6] W.K. Chow and J.K. Aggarwal, "Computer Analysis of Planar Curvilinear Moving Images," IEEE Trans. Computers, Vol. C-26, pp. 179-185, Feb. 1977.

[3-7] W.N. Martin and J.K. Aggarwal, "Computer Analysis of Dynamic Scenes Containing Curvilinear Figures," Pattern Recognition, Vol. 11, pp. 169-178, 1979.

[3-8] H. Freeman, "Computer Processing of Line Drawing Images," Computing Surveys, Vol. 6, pp. 57-97, March 1979.

[3-9] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, John Wiley and Sons, 1973.

[3-10] W.A. Perkins, "Simplified Model-Based Part Locator," Proc. 5th International Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 260-263.

[3-11] W.A. Perkins, "A Model Based Vision System for Industrial Parts," IEEE Trans. on Computers, Vol. C-27, pp. 126-143, Feb. 1978.

[3-12] J.W. McKee and J.K. Aggarwal, "Finding The Edges of The Surfaces of Three-dimensional Curved Objects by Computer," Pattern Recognition, Vol. 7, pp. 25-52, 1975.

[3-13] J.W. Roach and J.K. Aggarwal, "Computer Tracking of Objects Moving in Space," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI-1, pp. 127-135, April 1979.

[3-14] B. Bhanu and O.D. Faugeras, "Shape Matching Using Hierarchical Gradient Relaxation Technique," USCIPI Report 990, Image Processing Inst. ,Univ. of Southern Calif. , Los Angeles, Oct. 1980.

[3-15] C.N. Dorny, A Vector Space Approach to Models and Optimization" John Wiley and Sons, 1975.

[3-16] D.G. Luenberger, Introduction to Linear and Nonlinear Programming, Addison-Wesley Publishing Co., 1973.

[3-17] F.A. Lootsma, (ed.), Numerical Methods for Nonlinear Optimization, Chapter 23, Academic Press, New York, 1972.

[4-1] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, New York, Wiley-Interscience, 1973.

[4-2] P.H. Winston, Ed., The Psychology of Computer Vision, McGraw Hill, New York, 1975.

[4-3] D. Nitzan, A.E. Brain and R.O. Duda, "The Measurement and Use of Registered Reflectance and Range Data in Scene Analysis," Proc. IEEE, Vol. 65, pp. 206-220, Feb. 1977.

[4-4] R.O. Duda, D. Nitzan and P. Barrett, "Use of Range and Reflectance Data to Find Planar Surface Regions," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-1, pp. 259-271, July 1979.

[4-5] Y. Shirai, "Recent Advances in 3-D Scene Analysis," Proc. 4th Int. Joint Conf. on Pattern Recognition, Kyoto, Japan, Nov. 1978, pp. 86-94.

[4-6] M. Ishii and T. Nagata, "Feature Extraction of Three-Dimensional Objects and Visual Processng in a Hand-Eye System Using a Laser Tracker," Pattern Recognition, Vol. 8, pp. 229-237, 1976.

[4-7] R.A. Lewis and A.R. Johnston, "A Scanning Laser Range Finder for a Robotic Vehicle," Proc. 5th Int. Joint Conf. on Artificial Intelligence, Cambridge, MA, Aug. 1977, pp. 762-768.

[4-8] M. Idesawa and T. Yatagai, "3-D Shape Input and Processing by Moire Technique," Proc. 5th Int. Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 1085-1090.

[4-9] T. Vamos and M. Bathor, "3-D Complex Object Recognition Using Programmed Illumination," Proc. 5th Int. Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 1091-1093.

[4-10] H. Fuchs, "The Automatic Sensing and Analysis of 3-D Surface Points from Visual Scenes," Ph.D. Thesis, Dept. of Computer Science, Univ. of Utah, Salt Lake City, Utah, Aug. 1975.

[4-11] G.J. Agin and T.O. Binford, "Computer Description of Curved Objects," Proc. 3rd Int. Joint Conf. on Artificial Intelligence, pp. 629-640.

[4-12] R. Bajcsy and L. Lieberman, "Texture Gradient as a depth cue," Computer Graphics and Image Processing Vol. 5, pp. 52-67, 1976.

[4-13] M.D. Levine, D.A. O'Handley and G.M. Yagi, "Computer Determination of Depth Maps," Computer Graphics and Image Processing, Vol. 2, pp. 131-151, 1973.

[4-14] R. Bajcsy, "Three Dimensional Scene Analysis," Proc. 5th Int. Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 1064-1074.

[4-15] C.L. Jackins and S.L. Tanimoto, "Oct-Trees and Their Use in Representing Three-Dimensional Objects," Computer Graphics and Image Processing, Vol. 14 pp. 249-270, 1980.

[4-16] I.E. Sutherland, R.F. Sproull and A. Schumacker, "A Characterization of Ten Hidden-Surface Algorithms," ACM Computing Surveys, Vol. 6, pp. 1-55, March 1974.

[4-17] L.G. Roberts, "Machine Reception of Three-Dimensional Solids," in Optical and Electro-Optical Information Processing, J.T. Tippett et. al., (Eds.), 1965.

[4-18] T.O. Binford, "Visual Perception by Computer," IEEE Conf. on Systems and Control, Miami, Dec. 1971.

[4-19] R. Nevatia and T.O. Binford, "Description and Recognition of Curved Objects," Artificial Intelligence, Vol. 8, pp. 77-98, 1977. Also Computer Analysis of Scenes of 3-Dimensional Curved Objects, Birkhauser Verlag, Basel und Stuttgart, 1976.

[4-20] R.K. Bajcsy and B.I. Soroka, "Steps Towards the Representation of Complex Three-Dimensional

Objects," Proc. 5th Int. Joint Conf. on Artificial Intelligence, p. 596.

[4-21] D. Marr, "Representing Visual Information - A Computational Approach," in Computer Vision Systems, Allen R. Hanson and Edward M. Riseman (Eds.), pp. 61-80, Academic Press, 1978.

[4-22] U. Shani, "A 3-D Model-Driven System for the Recognition of Abdominal Anatomy from CT Scans," Proc. 5th Int. Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 585-591.

[4-23] N. Badler and R. Bajcsy, "Three-Dimensional Representations for Computer Graphics and Computer Vision," ACM Computer Graphics, Vol. 12, pp. 153-160, Aug. 1978.

[4-24] G.T. Toussaint, "Pattern Recognition and Geometrical Complexity," Proc. 5th Int. Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 1324-1347.

[4-25] D.L. Milgram and C.M. Bjorklund, "Range Image Processing : Planar Surface Extraction," Proc. 5th Int. Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 912-919.

[4-26] R.J. Popplestone, et. al., "Forming Models of Plane-and-Cylinder Faced Bodies," Proc. 4th Int. Joint Conf. on Artificial Intelligence, pp. 664-668.

[4-27] K. Sugihara and Y. Shirai, "Range Data Understanding Guided by a Junction Dictionary," Proc. 5th Int. Joint Conf. on Artificial Intelligence, p. 706.

[4-28] S. Inokuchi and R. Nevatia, "Boundary Detection in Range Pictures," Proc. 5th Int. Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 1301-1303.

[4-29] S.W. Zucker and R.A. Hummel, "An Optimal Three-Dimensional Edge Operator," Proc. IEEE Conf. on Pattern Recognition and Image Processing, 1980, pp. 162-168.

[4-30] R.T. Chien and Y.H. Chang, "Recognition of Curved Objects and Object Assemblies," Proc. 2nd Int. Joint Conf. on Pattern Recognition, Aug. 13-15,

1974, Denmark, pp. 496-510.

[4-31] J.W. McKee and J.K. Aggarwal, "Finding the Edges of The Surfaces of Three-Dimensional Curved Objects by Computer," Pattern Recognition, Vol. 7, pp. 25-52, 1975.

[4-32] M.A. Fischler and R.A. Elschlager, "The Representation and Matching of Pictorial Structures," IEEE Trans. on Computers, Vol. C-22, pp. 67-92, Jan. 1973.

[4-33] W.A. Perkins, "A Model Based Vision System for Industrial Parts," IEEE Trans. on Computers, Vol. C-27, pp. 126-143, Feb. 1978.

[4-34] B.K.P. Horn and B.L. Bachman, "Using Synthetic Images to Register Real Images with Surface Models," Proc. Image Understanding Workshop, October 1977.

[4-35] T. Kanade, "Model Representations and Control Structures in Image Understanding," Proc. 5th Int. Joint Conf. on Artificial Intelligence, pp. 1074-1082.

[4-36] A. Rosenfeld and A.C. Kak, Digital Picture Processing, New York, Academic Press, 1976.

[4-37] T. Pavlidis and S. Horowitz, "Segmentation of Plane Curves," IEEE Trans. on Computers, Vol. C-23, pp. 860-870, 1974.

[4-38] U. Ramer, "An Iterative Procedure for the Polygonal Approximation of Plane Curves," Computer Graphics and Image Processing, Vol. 1, pp. 244-256, 1972.

[5-1] G.J. Agin, "Computer Vision Systems for Industrial Inspection and Assembly," IEEE Computer, pp. 11-20, May 1980.

[5-2] D.L. Milgram and C.M. Bjorklund, "Range Image Processing: Planar Surface Extraction," Proc. 5th Int. Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 912-919.

[5-3] R.Y. Wong and E.L Hall, "Scene Matching with Invariant Moments," Computer Graphics and Image Processing, Vol. 8, pp. 16-24, 1978.

[5-4] T.P. Wallace, O.R. Mitchell and K. Fukunaga, "Three-Dimensional Shape Analysis Using Local Shape Descriptors," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-3, pp. 310-323, May 1981.

[5-5] S.A. Dudani, K.J. Breeding and R.B. McGhee, "Aircraft Identification by Moment Invariants," IEEE Trans. on Computers, Vol. C-26, pp. 39-45, Jan. 1977.

[5-6] F.A. Sadjadi and E.L. Hall, "Three-Dimensional Moment Invariants," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, pp. 127-136, March 1980.

[5-7] M.K. Hu, "Visual Pattern Recognition by Moment Invariants," IRE Trans. on Information Theory, Vol. IT-8, pp. 179-187, Feb. 1962.

[5-8] T. Kanade, "Model Representations and Control Structures in Image Understanding," Proc. 5th Int. Joint Conf. on Artificial Intelligence, 1977, pp. 1074-1082.

[5-9] R. Bajcsy, "Three Dimensional Scene Analysis," Proc. 5th Int. Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 1064-1074.

[5-10] Y. Shirai, "Recent Advances in 3-D Scene Analysis," Proc. 4th Int. Joint Conf. on Pattern Recognition, Kyoto, Japan, Nov. 1978, pp. 86-94.

[5-11] G.J. Agin, "Hierarchical Representation of 3-D Objects Using Semantic Models," Workshop on the Representation of 3-D Objects, Philadelphia, May 1-2, 1979.

[5-12] T. Binford, R. Brooks and R. Greener, "Progress Report on a Model-Based Vision System," Workshop on the Representation of 3-D Objects, Philadelphia, May 1-2, 1979.

[5-13] P.H. Winston and the staff, "MIT Representation Techniques," Proc. DARPA Image Understanding Workshop, Nov. 1979, pp. 128-135.

[5-14] U. Shani, "A 3-D Model-Driven System for the Recognition of Abdominal Anatomy from CT Scans,"

Proc. 5th Int. Conf. on Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 585-591.

[6-1] J.S. Weszka, "A Survey of Threshold Selection Techniques," Computer Graphics and Image Processing, Vol. 7, pp. 259-265, 1978.

[6-2] R. Ohlander, K. Price and O. Reddy, "Picture Segmentation Using A Recursive Region Splitting Method," Computer Graphics and Image Processing, Vol. 8, pp. 313-333, 1978.

[6-3] A. Rosenfeld and A.C. Kak, Digital Picture Processing, Academic Press, New York, 1976.

[6-4] A.K. Jain, S.P. Smith and E. Backer, "Segmentation of Muscle Cell Pictures: A Preliminary Study," IEEE Trans. on Pattern Analysis and Machine Intelligence, PAMI-2, pp. 232-242, May 1980.

[6-5] A. Rosenfeld and L.S. Davis, "Iterative Histogram Modification," IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-8, pp. 300-302, 1978.

[6-6] S. Peleg, "Iterative Histogram Modification, 2," IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-8, pp. 555-556, 1978.

[6-7] A. Rosenfeld, "Image Understanding Project Status Report," Proc. DARPA Image Understanding Workshop, April 1979, pp. 14-24.

[6-8] A. Rosenfeld, R. Hummel and S.W. Zucker, "Scene Labeling by Relaxation Operations," IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-6, pp. 420-433, 1976.

[6-9] B. Bhanu and O.D. Faugeras, "Computer Analysis of Moving Images," USCIPI Report 960, Image Processing Inst., Univ. of Southern Calif., Los Angeles, March 1980.

[6-10] R. Nevatia and K.R. Babu, "Linear Feature Extraction and Descripton," Computer Graphics and Image Processing, Vol. 13, pp. 257-269, June 1980.

[6-11] O.D. Faugeras and M. Berthod, "Improving Consistency and Reducing Ambiguity and Stochastic Labeling: An Optimization Approach," to appear in IEEE Trans. on Pattern Analysis and Machine Intelligence.

[6-12] O.D. Faugeras and M. Berthod, "Using Context in the Global Recognition of a Set of Projects: An Optimization Approach," Proceedings of the 8th World Computer Congress (IFIP 80), Tokyo, pp. 695-698.

[6-13] G. Fekete, "Relaxation: Evaluation and Applications," Tech. Report-796, Dept. of Comp. Sci., Univ. of Maryland, Aug. 1979.

[6-14] B. Bhanu and O.D. Faugeras, "Segmentation of Images Having Unimodal Distributions," Submitted to IEEE Trans. on Pattern Analysis and Machine Intelligence, July 1980.

[7-1] O.D. Faugeras and M. Berthod, "Using Context in the Global Recognition of a Set of Objects: An Optimization Approach," Proceedings of the 8th World Computer Congress (IFIP 80), Tokyo, pp. 695-698.

[7-2] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, John Wiley & Sons Inc., 1973.

[7-3] B. Bhanu, "Computation of Features in the Analysis of Images of Moving Objects," USCIPI Report 990, Image Processing Inst., Univ. of Southern Calif., Los Angeles, Oct. 1980.

[7-4] B. Bhanu and O.D. Faugeras, "Computer Analysis of Moving Images," USCIPI Report 960, Image Processing Inst., Univ. of Southern Calif., Los Angeles, March 1980.

[A-1] O.D. Faugeras and M. Berthod, "Scene Labeling: An Optimization Approach," Proc. IEEE Comp. Soc. Conf. on Pattern Recognition and Image Processing, Aug. 6-8, 1979, Chicago, pp. 318-326.

[A-2] J.B. Rosen, "The Gradient Projection Method for Nonlinear Programming, Part I. Linear Constraints," J. Soc. Indust. Appl. Math., Vol. 8, pp. 181-217, March 1980.

[A-3] C.N. Dorny, A Vector Space Approach to Models and Optimization, John Wiley, 1975.

[B-1] T. Mayeda, "Real Time Video Acquisition and Digital Display System," Image Processing Inst., Univ. of Southern Calif., Los Angeles, 1979