Shape retrieval and recognition

Yu, Xiaozhou

2007

Yu, X. (2007). Shape retrieval and recognition. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/35743>

<https://doi.org/10.32657/10356/35743>

9529575

Shape Retrieval and Recognition

Yu Xiaozhou

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

2007

Acknowledgements

My deepest gratitude must go to my supervisor Dr. Maylor Leung, who is, by all means, a devoted teacher. His knowledge, vision, inspiration and encouragement have guided me through the wonder land of science and have taken me to the frontier of research. His refusal to accept any lowering of standards of any kind will affect all the rest of my life.

I am greatly indebted to my parents for their encouragement throughout all these years. Their teaching about the value of hard work and independence is always my source of impetus. Thank for their consistent support behind me for both my success and failure.

The Center for Advanced Media Technology (CAMT) provided space and equipment to work with, and the staff at CAMT untangled the complicated workings of the machines and office environment. My friends, Chen Xi, Zhang Xiaozheng, Liu Yongsheng, Ng Wankee, Li Li, Zhu Huabing, Liu Qi, Chi Yanling, Sun Yingying, helped lighten the atmosphere at work and beyond. To all, my heartfelt thanks.

Finally, I would like to thank Nanyang Technological University for the opportunity that they provided.

Contents

1	Introduction	1
1.1	Motivation	2
1.2	Objectives	4
1.3	The proposed approach	5
1.4	Input data	10
1.5	Thesis organization	10
2	Background	12
2.1	2D shape recognition	12
2.1.1	Template matching	13
2.1.2	Syntactic/structural approach	15
2.1.3	Line pattern matching	17
2.2	Feature point detection overview	20
2.3	Indexing technique	26
2.3.1	Dimension reduction	27
2.3.2	Multi-dimensional indexing structure	28

2.3.3	Filtering techniques	32
2.4	Distance transform	33
2.5	Hausdorff distance	36
2.6	Logo recognition overview	40
3	System Overview	47
3.1	Preprocessing	47
3.2	Shape representation	50
3.3	Indexing	53
3.3.1	Feature point selection	54
3.3.2	Local structure construction	55
3.3.3	Indexing structure generation	56
3.3.4	Indexing	56
3.4	Matching	57
3.4.1	Matcher selection	58
3.4.2	Curve segment Hausdorff distance	58
4	Shape Representation	59
4.1	Dynamic two-strip algorithm (Dyn2S)	60
4.2	Round corner point refinement	63
4.3	Results	70
4.4	Summary	71

5	Indexing and Alignment	75
5.1	Feature point selection	76
5.2	Local structure construction	84
5.3	Indexing structure generation	88
5.3.1	Integration of all neighbors	88
5.3.2	Integration of all models	90
5.4	Indexing	90
5.5	Summary	91
6	Curve Segment Hausdorff Distance	93
6.1	Line segment Hausdorff distance	94
6.2	Matcher selection	97
6.3	Curve segment Hausdorff distance (CsHD)	99
6.3.1	Attribute grammar description	99
6.3.2	CsHD computation	100
6.3.3	New formulation of $d_{\perp}(L_i, L_j)$	104
6.3.4	Complexity analysis	105
7	Experimental Results	106
7.1	Experimental results of indexing method	108
7.1.1	Parameter selection for indexing method	108
7.1.2	Indexing system performance	112
7.2	Experimental results of CsHD	116

7.2.1	Experiment 1: Effective shape recognition and system characteristics	117
7.2.2	Experiment 2: Comparison of MHD, LHD and CsHD	118
7.2.3	Experiment 3: Recognition of images seen from arbitrary viewing angle	120
7.2.4	Experiment 4: Recognition of occluded images	121
7.2.5	Experiment 5: Palmprint matching	123
7.2.6	Experiment 6: Real object recognition	125
7.2.7	Experiment 7: Mixed-object recognition	125
7.2.8	Performance discussion	129
7.3	Summary	138
8	Conclusion	139
8.1	Work summary	139
8.2	Discussion and future work	141
8.3	Contribution	144
8.4	Thesis conclusion	146
	Bibliography	147
A	Results of round corner refinement	171
B	The test logos used in our experiment	197
C	The test images from real objects used in our experiment	207

D The mixed images used in our experiment	212
---	-----

Abstract

Shape recognition has been an attractive research area with both theoretical and practical importance in the fields of computer vision, pattern analysis and object classification. However, to the best of our knowledge, there is as yet not one unique technique of general validity to measure shape similarity and robust to noise. In this research, we aim to propose and investigate one shape recognition system which can tackle different types of input data, such as logos, fingerprints, work tools and characters.

To achieve the desired accuracy and efficiency, the proposed system employs a three stage hierarchy, i.e. polygonal approximation, indexing and matching. In the first stage, the raw images are transformed into line representation from Edge Map (LEM). In the second stage, effective line pattern features are extracted to support indexing so as to prune the model space with respect to a test image. Meanwhile, the alignment problem can be solved as a byproduct. In the third stage, a new scheme – Curve segment Hausdorff Distance (CsHD), which measures the dissimilarity (distance) between two images, is proposed to screen further and generate the best matches. The proposed polygonal approximation approach is based on robust shape feature and tends to generate more consistent LEM, while the indexing and alignment are based on local distribution of the extracted features, which is relatively insensitive to noise. The proposed CsHD works on the line segments extracted from the edge information of the images and organizes them in a more proper way by regrouping the consecutive line segments into a curve segment. Then the distances between two

lines, between two curves and between two images are constructed systematically.

An in-depth study of the proposed technique has been carried out on logos, palmprints and stationery. The test images come from six sources, i.e. scaled and rotated, Gaussian noise, strip corrupted, partially occluded, skewed and mix noise corrupted. Encouraging results have been observed, which supports the concept experimentally. Compared with general shape recognition methods, the proposed techniques are simple in concept and can tackle more distortion and transformation types than the others. This suggests a strong potential for a general shape recognition system.

List of Figures

1.1	Examples of the input images, (a) logos, (b) palmprints, and (c) stationeries.	11
2.1	Masks describing the distance transformations, (a) Parallel mask for parallel computations, (b) Forward (left) and backward (right) masks for sequential computations.	34
2.2	The 3×3 neighborhood mask used in Chamfer distance transformation.	35
2.3	Similarity function of $\rho_T(x)$	40
3.1	The proposed system flowchart.	48
3.2	The preprocessing of the system.	49
3.3	An illustration of segmentation results, (a) original intensity image, (b) edge image, (c) thinned edges.	50
3.4	The shape representation flowchart.	51
3.5	An example of the line edge map. The feature points are circled. . . .	52
3.6	One example of the feature point post-processing, (a) point set from Dyn2S, (b) after round corner processing, (c) after filtering.	53

3.7	The proposed indexing scheme.	54
3.8	An example of a local structure with its center (the star) and the 6 nearest neighbors.	56
3.9	The proposed matching scheme.	57
4.1	Illustration of the dynamic two-strip algorithm.	62
4.2	Redundant feature points detected at the round corners and in the mid of a straight line.	63
4.3	An illustration of feature refinement; Δ_1, Δ_2 : the new fitting errors; M : the highest point of $\widehat{BD}$	64
4.4	The algorithm flowchart.	66
4.5	Illustration of <i>Criterion 1</i>	67
4.6	The step-wise linear relationships between P_N and $(\theta_1 - \theta_2)$	69
4.7	The sine function modeling of P_N in the middle of P_{i+1} and P_{i+2}	69
4.8	Comparison of the results before and after the corner refinement.	72
4.9	Comparison of the results before and after the corner refinement.	73
4.10	Examples of false refinement. The first column gives the original im- ages while the second column is the results after refinement. The bolded circles indicate the parts with unsatisfactory refinement.	74
5.1	Feature points P_i/P'_i on circular arc (a) and round corner (b).	77
5.2	An example of $Cir(P_{i-1}P_iP_{i+1})$	78
5.3	An example of S_1	79

5.4 Illustration of formula 5.1. 80

5.5 Illustration of S_1 and S_2 calculation. 81

5.6 Illustration of S'_1 calculation. 81

5.7 Illustration of the signs in S'_1 and S'_2 calculation. 82

5.8 A flowchart of the feature point selection algorithm. 83

5.9 Illustration of **Criterion 2**. 84

5.10 (a) LEM before feature point selection, (b) LEM after feature point
selection. 85

5.11 Illustration of $Ang(P)$ and $Dir(P)$ 86

5.12 An example of local structure. 87

5.13 Illustration of H 89

6.1 (a) line displacement measures. $d_{\parallel} = \min(l_{\parallel 1}, l_{\parallel 2})$, $d_{\perp} = l_{\perp}$; (b) illus-
tration of the angle distance. 95

6.2 Illustration of angle and perpendicular distances, (a) HD and MHD;
(b) $l_{\perp}$ and $l_{\theta} = d_{\theta}'(L_i, L_j)$ 96

6.3 Assuming $\|d_{\parallel}\| = 0$, (a) variation of d for fixed l_{θ} ; (b) variation of d
for fixed $l_{\perp}$ 97

6.4 (a) Chamfer distance map from C_{input} ; (b) overlay model curve C_{model}
on the distance map. 98

6.5 Illustration of angular distances between lines and curves. 103

6.6 Illustration of the new design: $d_{\perp}'$. To simplify the illustration, two lines are assumed to intersect and get split at the midpoint. 104

7.1 Model images and test images in our experiment. 107

7.2 Alignment results with respect to the numbers of required neighbors. 109

7.3 Alignment results with respect to different values of θ_{Sep} 110

7.4 Alignment results with respect to different values of tolerance angle. . 111

7.5 The result of alignment. 113

7.6 The filtering result on the test images. 114

7.7 The running time of indexing. 115

7.8 Recognition rates of CsHD according to different N_{layer} 116

7.9 Recognition rates of CsHD, (a) the lowest dissimilarity is accepted as the match; (b) the first 3 lowest dissimilarities are accepted as the matches. 118

7.10 The comparison between MHD, LHD and CsHD, (a) the lowest dissimilarity is accepted as the match; (b) the first 3 lowest dissimilarities are accepted as the matches. 119

7.11 An example of LEM with Gaussian noise: (a) original image; (b) line representation. 120

7.12 Some examples of test images: (a)-(e) are skewed horizontally with 5, 15, 25, 35 and 45 degrees respectively; (f)-(j) are skewed vertically with 5, 15, 25, 35 and 45 degrees respectively. 121

7.13 Normalization, (a) one example of a test image; (b)-(d) its normalized line segment drawings according to choices of different reference line segments. 122

7.14 Recognition rates of CsHD for skewed images, (a) using the original model database; (b) adding new models skewed by 20 and 40 degrees. 122

7.15 Some examples of test images, (a) images with 20% cut, (b) images with 40% cut, (c) images with 60% cut, (a) images with 80% cut. . . 123

7.16 Recognition rates of CsHD for occluded images. Top 1, 3 and 5 mean that the first, first 3 and first 5 lowest dissimilarities are accepted as the matches respectively. 124

7.17 Some examples of palmeprints. 125

7.18 Some examples of test images. 126

7.19 Comparison between LHD and CsHD on palmprint matching. The lowest dissimilarity is accepted as the match. 127

7.20 Recognition rates of CsHD for (a) occluded images and (b) skewed images. 127

7.21 Some examples of test images in real object recognition. The first row shows the model images while the test images are displayed in the second row. (a) occluded images; (b) images with added lines; (c) skewed images (20 degrees). 128

7.22 Recognition rates of CsHD for real object recognition. 128

7.23 Example results of cluttered object recognition. 129

7.24 An example of failure in mixed-object recognition, (a) input image, (b) expected model, (c) feature points distribution from the investigated part of input, (d) feature points distribution from the original scale corresponding to (c). 130

7.25 The recall rates versus different numbers of returned images. 130

7.26 The real computation time of LHD and CsHD on matching one test image to 500 models. 131

7.27 Some examples of failures due to incorrect alignment. The first column is the distorted input images and their LEMs while the second column is their corresponding models and LEMs. 132

7.28 An example of failed match from set 4. 133

7.29 An example of failed match from set 5. 134

7.30 An example of failed match from set 6. 135

7.31 An example of failed match from set 7. 136

7.32 Additional explanation for two returned models in Figure 7.28 and 7.31. 137

B.1 Test logos from set 1. 198

B.2 Test logos from set 2. 199

B.3 Test logos from set 3. 200

B.4 Test logos from set 4. 201

B.5 Test logos from set 5. 202

B.6 Test logos from set 6 (part 1). 203

B.7 Test logos from set 6 (part 2). 204

B.8 Test logos from set 7 (part 1). 205

B.9 Test logos from set 7 (part 2). 206

C.1 Test images of real objects from set 1. 208

C.2 Test images of real objects from set 2. 209

C.3 Test images of real objects from set 3. 210

C.4 Test images of real objects from set 4. 211

List of Tables

7.1	The running time of indexing system with respect to different values of θ_{Sep}	110
7.2	The running time of indexing system with respect to different values of tolerance angle.	111
7.3	The max, min and average percentages of models that get filtered out.	115

Chapter 1

Introduction

Shape recognition is one of the fundamental and important application areas in computer vision. Many shape recognition (matching) methods have been proposed in the past. However, most of them are just appropriate for particular types of shapes and applications. The presence of noise can greatly influence the choice and accuracy of the methods involved. In this investigation, a general and speedy shape recognition system is proposed. It is expected to be applicable to as many different applications as possible and relatively insensitive to noisy, occluded and cluttered environments.

1.1 Motivation

Object matching is an important problem which has widespread applications in image processing and computer vision, including database retrieval, robot vision and tracking. In all these applications, object shape plays an important role and object matching can be narrowed down to shape matching. Veltkamp [1] considered shape as something geometrical and used the term shape for a geometrical pattern, consisting of a set of points, curves, surfaces and solids. Many valuable shape matching algorithms have been developed for shape matching in the past several decades. Some of them are able to capture human perceptual similarity of shapes quite well [2, 3]. However, the input data fed to these methods were assumed to be complete shape outlines. This assumption limits the methods' applicability greatly, since images obtained from practical channels always comprise noisy and occluded parts rather than perfect outlines. It is difficult to apply this type of methods to different applications due to the limit. We propose here one approach that can perform in such situations provided that substantial part of the object can be seen.

The shape matching problem can be approached within five main frameworks [4]. Template matching is one of the simplest and earliest approaches. As other techniques (i.e. neural networks and statistical classification) have earned continuous updates and developments, template matching is still widely used because it has two key advantages:

1. Template matching is conceptually simple and easy to implement.

2. A template can capture almost all the inherent shape structures, while other descriptors which are equally representative are still being investigated.

Although effective in some application domains, template matching has a number of disadvantages. For instance, it cannot follow human perception well and would fail if the patterns are distorted due to the imaging process or viewpoint change.

The proposed new matching technique belongs to the template matching class but overcomes the disadvantages of template matching. We generalized the new approach from Hausdorff Distance method (with the non-explicit feature correspondence property) to match 2 sets of lines and two sets of curves.

Named after Felix Hausdorff (1868-1942), Hausdorff distance (HD) measures the difference between two sets of points which represent two images. Huttenlocher et al. [5] applied this concept in computer vision to compare two images represented by two sets of points where each set can be seen as a template. They claimed that this approach had three key advantages: (1) relative insensitivity to small perturbations of images, (2) simplicity and speedy computation, (3) natural allowance for portions of one shape to be compared with another. Potentially, these allow the HD approach to tolerate noisy, distorted and occluded objects in image processing and object matching applications. Unfortunately, low level matching can usually produce many false positives as illustrated by Paglieroni et al. in [6]. The situation becomes worse when there is a need to perform partial matching in a noisy and distorted environment. The generalized approach can tackle the false positive problem better, even in noisy, occluded and cluttered environments, and can be described and processed

using a syntactic technique.

In real applications, the input image will be matched to all the models in a database. A voting mechanism will suggest the most similar model as the match. When the number of model images in database is large, it will be slow to search linearly all stored models to find the match. Hence, indexing technique is necessary to provide efficient matching. Through indexing, non-interesting models are filtered out and the number of models that must be matched is reduced. To customize our algorithm into a general matching system, an efficient indexing method is also essential.

As a summary, we are motivated to propose a speedy shape recognition system which involves the characteristics mentioned above. It can identify different shapes even in distorted condition. Besides, the proposed system implies a great potential to be applied to different practical applications with superior performance.

1.2 Objectives

The major objective of this study is to design, explore and implement a novel, effective and efficient pattern recognition system based on LEM and the generalization of HD to CsHD. Particularly, we would like to investigate the following issues.

Effectiveness: It is of interest to find out whether the employed higher level primitives with inherent structural relationship from one primitive to the other can perform better than the low level point primitive, especially in noisy, distorted

and occluded environments. This can be measured by testing and comparing the recognition performances from HD to LHD and CsHD under different distortions.

Efficiency: Recognition requires the model database to be big enough for practical use. Meanwhile, the process to retrieve from this database should be as fast as possible. We will explore and compare timing of the proposed indexing scheme with respect to the database size in order to test our design and find out whether we have achieved what we have aimed for.

Extendibility: The proposed shape recognition system should be able to be extended to general object recognition and as many applications as possible. Since edges can be extracted in most images and are good representation of images, our system is expected to be able to be extended to handle different types of images and to be embedded as a component in an artificial intelligence system. We will test on 3 different types of inputs to observe and evaluate the performances.

1.3 The proposed approach

Detection of object boundaries/edges is an important part of the perception process. Edges are by far the most commonly used information for meaningful image analysis. There is much psychological evidence, such as our ability to understand cartoons, that boundaries/edges are sufficient for perception of a broad class of objects.

1.3 The proposed approach

6

Besides, edges are relatively insensitive to lighting changes and can be detected in most images for representation. Furthermore, if we extract line segments from this edge information of images, the shape can be represented in a more compact way. Cognitive psychological studies [7] indicated that human beings recognize line drawings as quickly and almost as accurately as gray-level pictures. These results might imply that line edges of objects could be used for object recognition and achieve similar accuracy as gray-level images. This is also the basic idea of what we tried to propose in this work. We first extract line segments from the edge information of the model and test image. The extracted lines are organized in a more proper way by regrouping the consecutive lines into a curve segment. An indexing step is then employed to prune the database to a great extent so that the number of candidate models for matching is limited to an acceptable range. Thus, the running time of the subsequent matching step can be reduced and the efficiency of the whole system can be improved. At the same time, the normalization problem can be solved without additional process and cost. Then a novel scheme – curve segment Hausdorff distance (CsHD), which measures the dissimilarity between two sets of curve segments, is applied to match two objects. The name curve segment Hausdorff distance derives from the well-known shape measurement method Hausdorff Distance (HD). It also inherits HD's advantage that no explicit correspondence is required between images so as to simplify the matching process.

In this research, the performance of the system will be tested on logo recognition, palmprint recognition and stationary recognition. A variety of different shapes have

been included in the experiment for processing and different sets of noisy images have been tested with encouraging experimental results.

System features and novelties

Here, we would like to summarize and highlight the important features of the proposed system in detail. Some of them have been discussed briefly in previous sections.

1. Line and curve are adopted as primitives due to their efficient representation, low storage demand and consistency in being detected in different scales. In addition, line and curve are higher level primitives than point. In this research, curve is defined as a sequence of consecutively connected line segments (or polyline) which simulates the basic shape of a smooth curve. A line or a curve can be seen as a structural entity integrating relevant points. Hence, they can tackle (or eliminate) the false positive problem better in distorted/cluttered/occluded environment [6] since it enforces curve-point association, i.e. points on one curve should match to points on another curve. The dissimilarity obtained for matching non-similar curves can thus be guaranteed to be high. Furthermore, the proposed system does not assume perfectly segmented object outline or boundary.
2. The proposed matching algorithm is a local structure approach in concept. Unlike most global features which employ the shape information of the whole image, our method employs local structures (lines and curves) only. Matching

based on these primitives works well with occlusion and cluttering because there are always some well-preserved local structures retained even in occluded or cluttered environment. We have developed a new scheme to extract and match on such well-preserved structures. In general, global approaches cannot tackle such problems.

3. The proposed matching method can be seen as a novel hybrid of template and syntactic/structural matching approaches. The curve segment collection of an image object is treated as a template which captures the inherent structural relations of curves. A distance transform is then used to pair structure entities from the input to the model. On the other hand, a curve is defined here as a consecutive run of connected line segments that can be constrained/matched using string attribute grammar. The relationships between curve segments can also be modeled using a graph. Hence, the proposed system has been designed and implemented from ideas based on the template and syntactic/structural matching approaches.
4. The matching process is based on the Chamfer/Hausdorff distance. Instead of matching 2 sets of points using HD, we propose the novel idea to generalize HD to match 2 sets of higher level primitives such as lines and curves. A novel distance computation between 2 lines is first derived. Based on this new distance, a novel scheme LHD (line segment Hausdorff distance) without explicit feature correspondence can be implemented to compute a distance measure between 2

sets of lines. Since a curve can be interpreted as a consecutive run of connected lines, curve segment Hausdorff distance (CsHD) can be implemented on top of LHD. The computation is simpler since non-explicit feature correspondence is employed to tolerate missing lines or gaps on curves.

5. The indexing process employs a novel fragmental neighborhood matching technique to prune the model search space effectively. The proposed technique first records the neighborhood information. Later, two neighborhoods are declared to match each other perfectly if fragments of one neighborhood can be found and matched in the other neighborhood. The indexing process then counts the number of such neighborhood matches from an input to each model. By controlling the size of the neighborhood and the minimal fragments that must match, we have found a new mechanism for fast indexing and retrieval that can effectively and efficiently identify the likely corresponding models of the input image. Simultaneously, as a byproduct, the alignment problem can be solved in the indexing process by aligning one input neighborhood to the corresponding model neighborhood.
6. The proposed system is based on the line structure pattern. Two types of features and relationships have been extracted from the line edge maps (LEM). The concept of feature points (the end points of lines) are utilized for the indexing process while the line structural relationships are integrated in the template matching process as LHD/CsHD. Our system presents an interesting

and effective approach that employs LEM for pattern recognition.

1.4 Input data

The concepts proposed in this work will be demonstrated on logos, palmprints and stationeries. Logos are 2D shapes of varying complexity, with interior and exterior contours, which are not necessarily connected. Logo recognition is of great interest in the document and shape matching domain. By recognizing the logo, semantic information about the document is obtained which may be useful to identify what category the processed document belongs to and to decide whether or not to analyze the textual parts. Palmprint which is biometric personal identification emerges as a powerful means for automatically recognizing a person's identity. It provides a reliable and automatic personal identification for effective security control. Several examples of logos, palmprints and stationeries are shown in Figure 1.1. A recognition system which works well on these 3 types of input images is also expected to be applicable on other types of images with lower complexity, such as block letter recognition.

1.5 Thesis organization

This thesis is divided into 8 chapters. A survey of previous work is carried out in Chapter 2. A system overview is described in Chapter 3. Chapter 4 describes one new method to refine the line segment maps from the original image to obtain a more consistent representation. Then, the work related to indexing and normal-

Figure 1.1: Examples of the input images, (a) logos, (b) palmprints, and (c) stationeries.

ization method is presented in Chapter 5. The detailed description of curve segment Hausdorff distance is given in Chapter 6. Chapter 7 shows the matching results using the proposed system. Finally, this thesis is concluded in Chapter 8.

Chapter 2

Background

The focus of this work is 2D shape recognition. Other related areas of interest are feature point detection, indexing techniques, distance transform, the Hausdorff distance and logo recognition. This chapter reviews the related work in these areas.

2.1 2D shape recognition

Shape recognition is an easy task for humans, but it is still quite difficult for computers. After an object is rotated, translated and distorted, shape recognition becomes more and more complicated. It is a challenge for researchers [8, 9, 10, 11, 12] to find a simple, less time consuming and highly accurate method for shape recognition.

The shape recognition problem can be approached within five main frameworks according to the matching algorithms they utilize: 1) template matching, 2) statistical classification, 3) syntactic/structural matching, 4) neural networks, 5) hybrid matching. As the proposed shape recognition algorithm can be seen as a hybrid of template and structural matching, we will review the methods on these two categories. Other methods can be found in other review papers [4]. In addition, line pattern matching methods are reviewed as a separate topic due to its relevancy to our work.

2.1.1 Template matching

One of the simplest and earliest approaches to shape recognition is based on template matching [13, 14, 15, 16, 17]. Matching is a generic operation which is used to determine the similarity between two entities of the same type. In template matching, a template or a prototype of the pattern to be recognized is available. The shape to be recognized is matched against the stored templates while taking into account all allowable pose and scale changes. Templates are generally represented as a set of pixels or points [18, 19]. Other appearances of template include curves [14, 20], surfaces [15] and regions of extracted points [21]. Template matching is considered computationally demanding, but some efficient variants of template matching have been investigated to overcome this disadvantage.

In [22], Olson used a probabilistic formulation for image matching in terms of maximum-likelihood estimation that can be used for both edge template matching

and gray-level image matching. His method is general with respect to the set of pose parameters allowed. They formulated the method using 2D edge and intensity templates with the pose space restricted to translations in the plane in order to simplify the presentation. Li and Hui [20] employed the template matching approach for feature recognition, which overcame the limitation of feature recognition methods that were limited to features of regular form and tackled the problem of free-form feature recognition. A feature selection method using template matching was proposed in [23]. It was based on the upper bound of the template matching error. A method for rotation-invariant template matching in gray scale images was proposed in [19]. It utilized gradient information in the form of orientation codes as the feature for approximating the rotation angles as well as for matching. They used a two-stage framework for realizing the rotation-invariant template matching; in the first stage, histograms of orientation codes were employed for approximating the rotation angle of the object and then in the second stage, matching was performed by rotating the object template by the estimated angle. Karungaru et al. made use of template matching for face recognition [24]. Template matching was performed with the help of a genetic algorithm to automatically test several positions around the target and also adjusted the size of the template as the matching process progressed. However, it lacked rotation factor in the genetic algorithm and improvement in the recognition result was also expected. In [25], a template matching method for spike sorting was proposed. It used the Levenberg-Marquardt algorithm to estimate the variances of each cluster based on distribution shape and a weighted Euclidean metric to refine

the classification results. Although template matching is effective in some application domains and provides high recognition rate [26], it has some disadvantages. Since the template is rigid, it cannot tolerate deviations from the model template.

2.1.2 Syntactic/structural approach

The basic idea of structural matching is to directly represent the models as well as the unknown test images by means of a suitable data structure (e.g. strings, tree and graphs) and to compare these structures in order to find the prototype which is most similar to the unknown input images. This approach requires a formal measure of similarity between two structural representations.

Many of such measures have been proposed in the past. String matching [27] is one of the most important and widely used methods. Suppose that two sequences of region boundaries, a and b are coded into strings denoted as $a_1a_2, \dots, a_n$ and $b_1b_2, \dots, b_m$, respectively. Let α represent the number of matches between the two strings, where a match occurs in the k^{th} position if $a_k = b_k$. The number of symbols that do not match is

$$\beta = \max(|a|, |b|) - \alpha \quad (2.1)$$

where $|\cdot|$ is the length (number of symbols) in the string representation of the argument. It can be shown that $\beta = 0$ if and only if a and b are identical. A simple measure of similarity between a and b is the ratio

$$R = \frac{\alpha}{\beta} = \frac{\alpha}{\max(|a|, |b|) - \alpha} \quad (2.2)$$

Hence R is infinite for a perfect match and 0 when none of the symbols in a and b matches. String matching is very efficient since similarity measures between strings can be computed quite fast. However, strings are limited in their representational power. Another important elemental structure for describing objects for shape recognition is tree [28]. Tree structures are intermediate between strings and graphs. A string or a graph can be transformed into a tree if some assumptions are taken into account. The string case is trivial as there is no need to consider any assumption, since a string is a tree with only one branch where depth is the length of the string.

Because string matching is conceptually simple and has a lower computational complexity, there is a broad spectrum of shape recognition based on string matching using line segments as symbols. In [29, 30], this technique was applied on Chinese characters recognition. Lee and Yu [31] used line segments to characterize scene images. Al-Khaiyat and Kamangar [32] employed arc segments extraction and matching to recognize fingerprints for personal identification. This technique was robust to digitization errors and noise effects, and can perform well when the number of concentric circles is relatively small.

As for syntactic methods [33], they provide a unified methodology for handling structural recognition problems. Basically, the idea behind the syntactic approach is the specification of a set of pattern primitives, a set of rules (in the form of grammar) that govern their interconnection, and a recognizer (called an automation), whose structure is determined by the set of rules in the grammar. In [34], a pyramidal approach to 2D object representation which relies on a linguistic description of the

object contour at different resolution levels was proposed. It implied the potential of higher dimensional grammars for defining production rules which describe the evolution of the contour between levels. This method is computationally efficient, rotationally invariant, as well as rather insensitive to noise.

2.1.3 Line pattern matching

In general, geometric features provide robust and useful information for extracting and describing the objects in a scene. The higher the level of descriptions at which matching is attempted, the more likely the descriptions are invariant to imaging changes. Among many geometrical features, from the viewpoint at the level of description, line features are the most appropriate ones, since most man-made objects can be effectively characterized by straight-line edges. Moreover, the extraction of these line features is much easier and robust than that of other features.

The structural description for a line pattern model can be established by exploiting the properties and the mutual relations between the line features in the model. The most prevailing properties are length, orientation and position (midpoint or endpoints) of a line segment. Some matching methods make use of these properties directly. The Hough transformation method is of this kind. For example, in [35], a shape matching technique based on the straight line Hough transform (SLHT) was presented. Each line in model provided a pair of parameters (θ, ρ) . In the $\theta - \rho$ space, the transform can be expressed as the sum of two new terms, namely, the translation term and the intrinsic term. A shape signature, called the scalable trans-

lation invariant rotation-to-shifting signature, was obtained from the $\theta - \rho$ space by computing the distances between pairs of points having the same θ value. This signature had the following properties: (1) it is invariant to translation; (2) rotation in the image space corresponds to circular shifting of the signature; (3) the signature can be easily normalized. Computing the similarity between two objects is equal to matching the signatures obtained from them. A similar work was presented in [36] to solve the correspondence problem in stereo vision. Yi and Camps [37] proposed a different line-feature-based approach for model based recognition. Midpoint position, length and orientation of line segments were three basic properties which were transformed to a four-dimensional space. The matching happened in the new space using Hausdorff distance. The implementation of the proposed algorithm can be naturally extended to higher dimensional space to efficiently find correspondences between n -dimensional patterns. Although distortions and occlusion can be tolerated to some extent for these methods, the searching for corresponding signatures was always a computational expensive process.

Besides, shape matching methods, employing the mutual relations between lines from a model, were developed in the last decade. The most important features describing line relations are length ratio and relative orientation [38, 39, 40, 41]. Moreover, perpendicular distance [42] and overlapping relationship [43] are utilized as shape descriptors. In these methods, the metrics for reference and candidate models are constructed to record relative distance and angle information between line features for each model. Then, the modes of the proximity matrices of the

two models are compared to yield the dissimilarity measure, which describes the quantitative degree of the difference between the two models. Since the features used here are mutual relations, occlusion may change the feature descriptors of model and deteriorate matching result.

Another set of shape matching methods based on line pattern is to simply match the structural properties of corresponding line segments from two shapes [44, 45, 46]. In [47], perpendicular distance from two lines was calculated. Omission and overlapping information was also taken into account to compute dissimilarity between two shapes. In similar way, orientation difference and distances between two lines endpoints were investigated in [48] to define two images' similarity. The methods derived from Hausdorff distance approach [49] belong to this category.

For some 3D pattern matching problems, 3D positions of the line segments and their orientation information are compared among different models to test their difference [50]. This technique can also provide a solution to 3D object motion estimation [51].

Recently, Park et al. [52] introduced a novel ARG matching based face recognition technique. Each face was represented by the Face-ARG model, in which all the geometric quantities and the relational information among features were encoded in a graph structure. This method was claimed to integrate the whole structure as well as local structural information of a face together. The matching process was implemented by constructing the correspondence graph between two faces, which made the approach capable to handle occlusion greatly. Although experimental results on real

face databases demonstrated that the proposed algorithm was very robust to various facial expressions, illumination conditions and occlusion, it was complex, time consuming and difficult to extend for object recognition in general.

2.2 Feature point detection overview

Feature point detection plays a crucial role in shape representation and object recognition. It can be classified into two major categories: outline-based approach and gray-level approach. The first approach is to initially segment the image into regions, represent the object boundary with a chain code, and search for points having maximum curvature or perform a polygonal approximation on the chains and then search for the line segment intersections. The major drawback of this approach is that the performance of corner detection depends on the preceding image segmentation result. The second approach which works directly on gray-level image [53, 54] has been employed recently. It mainly makes use of gradients and surface curvatures to define the sharpness measure of a corner. Points are then declared as corners if the value of sharpness measure meets some threshold conditions. Although this method does not require pre-segmented image contours, it is sensitive to the noise amplification effects of the second-derivative operators. The gray-level feature point detection algorithm can be divided into two groups. Template based methods [55] detect the similarity between a given template of a specific angle for each image sub-window. Because multiple orientation templates are used, this technique is computationally expensive.

Gradient based methods [56], on the other hand, rely on measuring the curvature of an edge that passes through a neighborhood. Gradient based methods are more likely to respond to noise than their contour-based counterparts.

In this study, our emphasis is on the LEM (line edge maps) based approach for feature point detection because

(1) Cognitive psychological studies indicated that human beings recognize line drawings as quickly and almost as accurately as gray-level pictures [7, 57];

(2) LEM provides rich structural information (such as line orientation, corner orientation, connectivity) that facilitates recognition.

A vision system, working with structural information, can capture the outline of an object, from which one can get its shape information. Object recognition through LEM can result in enormous data reduction, and a further data reduction can be obtained by extracting and connecting sequential feature points from the outline of the object. In general, many techniques for feature point detection can be classified into two types: curvature estimation and polygonal approximation.

Curvature estimation

The common stages for this kind of methods are: (1) curvature estimation, and (2) the local maximum curvature searching. It is important to choose a pre-specified length of support region to assess the significance of curvatures. However, if the length of the support region is large, some significant dominant points may be missed. On the other hand, a small length in the support region will result in some redundant

corners.

Ray and Pandyan [58] used the adaptive smoothing technique to detect dominant points. The roughness of the curve in the neighborhood of the point was computed firstly to determine the smoothing level. Marji and Siy [59] proposed an algorithm for detecting dominant points by using an optimal criterion for determining the region-of-support of each boundary point and a mechanism for selecting the dominant points. The algorithm does not require an input parameter and can handle shapes that contain features of multiple sizes efficiently. Choi et al. [60] suggested a dynamic programming (DP) approach to improve the vertex selection scheme. It is shown that DP can be successfully applied to vertex adjustment and enables global optimality without any iterative processes. Wu [61] proposed an adaptive method for the polygonal approximation of a digitized curve. Instead of setting a fixed length of support region in advance, the new method will compute the suitable length of the support region for each point to find the best approximated curvature. The dominant points are identified as the points with local maximum curvatures. Fu et al. [62] focused on the method's invariance under scaling and rotation and proposed a CBF (curve bend function) based method for searching critical points of a contour. Its effectiveness and reliability in generating the description of the contour were demonstrated. However, some parameters about the contour need to be assigned initially to support the process. In [63], a method of calculating a property – which can be regarded as discrete curvature – of planar digital boundaries, was presented. Chord-to-point distance accumulation is computed by accumulating the

distance from a point in the boundary to a chord specified by moving end-points. Experimental and analytical results show that the distance accumulation has similarities to curvature in describing curves. An improved, wavelet-based technique for feature point detection in 2D planar curves was proposed in [64]. The boundary based technique is simple to implement and computationally efficient and exploits the wavelet transform modulus maxima to detect corners. It is also claimed to be robust with respect to object geometry.

Polygonal approximation

The polygonal approximation attempts to find polygons that can best fit the curve, based on certain estimate of appropriateness. Most existing methods provide local optimal approximation results due to limited computational time resources. They can be divided into three groups [65]: (1) sequential approaches, (2) split-and-merge approaches, and (3) dominant point detection approaches.

For sequential approaches, Daescu and Mi [66] presented an approach for approximating polygonal chains in the plane. This method is simple, based on standard geometric operations, and thus suitable for efficient implementation. Kolesnikov and Franti [67] tried to bridge the gap between the slow but optimal, and the fast but sub-optimal algorithms of polygonal approaches by giving a new near-optimal approximation algorithm based on reduced-search dynamic programming. The algorithm can be iterated as many times as further improvement can be achieved in the optimization. Yin [68] presented a new approach based on the particle swarm

optimization (PSO) algorithm. Each particle represented as a binary vector corresponds to a candidate solution to the polygonal approximation problem. A swarm of particles are initiated and fly through the solution space for targeting the optimal solution. The experimental results manifest that PSO can significantly improve the approximation results in terms of the compression ratio, and the results obtained in different runs are more consistent. For split-and-merge approaches, Ku and Chiu [69] suggested a graduate iterative merging algorithm for polygon approximation. The algorithm always gives a minimum area difference between the curve and the polygon with the minimum number of breakpoints. They claimed that its performance is very close to that of a human viewer. Ray and Ray [70] proposed an orientation-invariant and scale-invariant method by introducing the use of ranks of points and the normalized distances. The approximation results of the split-and-merge approaches may be far from the optimal one if a poor initial segmentation is used. In [71], a method for segmentation images using stick growing and merging is described, which consists of four major steps: stick initialization, stick merging, sub-region merging, and boundary modification. It is started from an initial decomposition of the image into small sticks and non-stick. The small sticks are merged to obtain the initial sub-regions on the basis of homogeneity criteria. Then smaller sub-regions with only one stick are merged into larger sub-regions, and subsequently all sub-regions are merged into regions according to the criteria. Finally, non-stick and separate small sticks are merged and the degree of boundary roughness is reduced by boundary modification. For dominant point detection approaches, Sarfraz and Masood [72] presented a recur-

sive algorithm for piecewise polygonal approximation of a digital curves. It extracts break points first as the candidates to be selected as the end points of approximating straight lines. Then a two step polygonal approximation algorithm is applied on each segment between a pair of such candidate points. The dynamic programming technique is applied in [73] to improve the fitting quality of polygonal approximation. The dominant points of a digital curve are detected by applying a scale-based approach. Then, a polygonal approximation of the digital curve can be obtained by connecting its dominant points. The combination of the dominant point detection and the dynamic programming can produce a description that has high geometric importance and good fitting quality.

Among the existing methods, no one can satisfy all applications since each method has its own disadvantages. Therefore, methods that combine the curvature estimation and polygonal approximation were investigated. Wu and Wang [74] proposed a curvature-based polygonal approximation algorithm. They first estimated the curvature of each point on the curve and located the points which had local maximum curvatures as the corners. Then they performed polygonal approximation by partitioning the curves between two consecutive potential corners. In [75], a corner-diagnostic approach was suggested, in the form of a three-step algorithm which (a) identifies the number and positions of corners, (b) fits smooth curves between corners, and (c) splices together the smooth curves and the corners, to produce an over-all estimate of the convex set. The corner-finding step is parametric in character while the smooth-curve fitting step is entirely nonparametric. The splicing step marries

these two disparate approaches into a single, practical method.

2.3 Indexing technique

As more and more images are captured and stored in the digital form, many techniques and systems have been developed for indexing and retrieval. The retrieval is normally based on similarities between extracted feature vectors of the query and stored items. Feature vectors are usually multidimensional. When the number of stored objects and/or the number of dimensions of the feature vectors is large, it will be too slow to linearly search all stored feature vectors to find those that satisfy the query criteria. Techniques and data structures are thus required to organize feature vectors and manage the search process so that objects relevant to the query can be located quickly.

Some retrieval systems work without data structure [76]. Instead, the distance from each image to other images in the collection is pre-computed and stored. Thus, images can be retrieved quickly when the query is one of the images in the collection. Some systems make use of subject classification to improve retrieval speed [77]. Some adopt indexing techniques, e.g. k-d tree [78] is extended for image retrieval. Generally, three techniques are commonly used to cope with “dimensional curse” [79]: 1) Dimension reduction (DR), such as KLT [80]; 2) Multidimensional indexing structure, such as B^+ -tree [81], VAM-split R-tree [82], TV-tree [83], k-d tree [84], M-tree [85], quad-tree, and grid file [86] are commonly used; 3) Filtering techniques [87]. The

following provides a survey of the techniques and data structures for indexing.

2.3.1 Dimension reduction

Even though the dimension of the feature vectors in image retrieval is normally very high, the embedded dimension is much lower [84]. It is beneficial to perform dimension reduction before indexing techniques are used.

KLT: KLT has been used in dimension reduction [80]. Eigenimage is KLT's variation in face recognition and PCA is KLT's variation in information analysis. In [88], Ng and Sedighian followed the Eigenimage approach to carry out the dimension reduction. In [89], Faloutsos and Lin proposed a fast approximation to KLT to perform the dimension reduction. Experimental results from their research showed that most real data set can be considerably reduced in dimension without significant degradation in retrieval quality. Chandrasekaran et al. developed a low-rank singular value decomposition (SVD) update algorithm which was efficient and numerically stable in performing KLT [90]. One thing worth pointing out is that blind dimension reduction can be dangerous, since information can be lost if the reduction is below the embedded dimension. To avoid blind reduction, a post-verification stage is needed.

Clustering: clustering is a powerful tool in performing dimension reduction. The basic idea of using clustering [91] for efficient retrieval is as follows. Similar information items are grouped together to form clusters based on a certain similarity measurement. Each cluster is represented by the centroid of the feature vectors. During retrieval, the similarity between the query vector and each of the clusters (rep-

resented by their centroid) is calculated. Clusters that have similarities greater than a certain threshold are selected. Then the similarity between the query vector and each of the feature vectors in these clusters is calculated and the k nearest items are ranked and returned. Clustering not only makes retrieval efficient, but also facilitates browsing and navigation. One representative item, whose feature vector is closest to the centroid of its cluster, is displayed for each cluster. If the user is interested in a representative item, he or she can view other items in the cluster. Clustering techniques can work together with other data structures for higher search efficiency. Similar items are grouped into clusters. Centroid of clusters and/or items within each cluster is organized in a certain data structure for efficient retrieval.

2.3.2 Multi-dimensional indexing structure

Most works in index or data structures are carried out in database and geographic information system (GIS) community. They are extended for image retrieval.

- **Binary tree:** The principle of the binary tree is as follows. A node in the binary tree has three elements: a key value x , a left pointer pointing to the records having key values less than x , and a right pointer pointing to records with key values equal to or greater than x . Each key value in the tree is commonly associated with a data record. When building a binary tree, the first record inserted will be the root. The second record will go left or right depending on whether its key value is less than the root key value. This process applies for

each insertion. The binary tree is unbalanced and its structure depends on the order in which records are inserted into the tree.

- **B^+ -tree:** A B^+ -tree is a hierarchical structure with a number of nodes [81]. Each node in a B^+ -tree contains n pointers and $n - 1$ keys. n is the order of the tree. Pointer 1 is used to access all records which have key values less than key 1. Pointer 2 is used to access all records which have key values greater than or equal to key 1 but less than key 2, and so forth. The last pointer is used to access all records which have key values larger than or equal to key $n - 1$. To search in B^+ -tree with N records, the required number of key value comparison and memory access is in the order of $O(\log_n N)$.
- **MB^+ -tree:** The MB^+ -tree is an extension of the standard B^+ -tree from one dimension to multiple dimensions [92]. The structure and the insertion and deletion algorithms of MB^+ -tree are very similar to those of B^+ -tree. However, the search methods are different as MB^+ -tree can support range and nearest-neighbor queries.
- **The k -d tree:** The k -d tree is an extension of the binary tree [93]. Its each key is a k -d vector. If the k dimensions are named as dimension 1, dimension 2, and so on to dimension k , and name the root of a k -d tree as level 1, and its child as level 2, and so on, the k -d tree is constructed as follows. At level 1, the tree is branched based on the value of the first dimension of the k -d vector; at level 2 the tree is branched based on the value of the second dimension of the

k - d vector, and so on. This process continues until all dimensions have a turn and it then starts from the first dimension again.

- ***R*-tree family:** The *R*-tree is a multidimensional generalization of the B^+ -tree, hence the tree is height balanced [94]. In a non-leaf node, an entry contains a pointer pointing to a lower level node in the tree and a bounding rectangle covering all the rectangles in the lower nodes in the sub-tree. In a leaf node, an entry consists of an object-identifier (or a pointer pointing to a list of feature vectors) and a bounding rectangle which bounds the region object (or all feature vectors of the list). Each bounding rectangle is represented by its lower-left and upper-right coordinates. Each non-leaf node in the *R*-tree with the exception of the root must be at least half full.
- ***X*-tree:** The *X*-tree [95] is a high-dimensional structure based on the R^* -tree [96]. It is designed specifically to avoid the large amount of overlaps that are created in high-dimensional R^* -tree structures. The *X*-tree organizes the data using a hybrid model of an array-like and a hierarchical *R*-tree-like directory. The *X*-tree solves the high overlap problem by using an extended directory structure, called the super-node, thereby avoiding splits that would result in high overlapping of the child nodes.
- ***TV*-tree:** The *TV*-tree is a variant of the *R*-tree [97]. It organizes the data into a hierarchical structure. Objects (feature vectors) are clustered into the leaf nodes of the tree, and the description of their minimum bounding regions

is stored in their parent nodes. Parent nodes are recursively grouped as well, until the root is formed. The minimum bounding region can be any shape, e.g. cube, or sphere, depending on the application.

- ***M*-tree:** *M*-tree, proposed in [98], is an effective dynamic index for similarity queries. The concept of *M*-tree relies on metric tree which partitions a given search space by relative distances of objects. Partitioning algorithm is critical to the effectiveness of the tree. A major difference of *M*-tree from other metric trees is that it gives efficient secondary storage organization: (a) Paged: tree is paged (consisting of fixed-size or variable-size nodes), (b) Balanced: paths from the root to leaves all have the same length, (c) Dynamic: able to deal with insertions and/or deletions without degrading search performance and storage utilization, and avoiding global tree re-organization.
- **The Pyramid-Technique:** The basic idea of the Pyramid-Technique [99] is to transform the d -dimensional data points into 1-dimensional values and then store and access the values using an efficient index structure such as the B^+ -tree. The transformation first partitions the data space into pyramids. Then rules are used to number the pyramids. Next, each point in the d -dimensional space is allocated a pyramid value. Then, an index is built based on the pyramid value.
- **Grid files:** The grid file is an extension of the fixed-grid (FG) access structure [100]. In the FG structure, an n -dimensional space is divided into equal-

sized hyper-cubes. Each hypercube contains zero or more feature vectors. They are accessed by a pointer associated with the hypercube. Each hypercube is indexed based on its location within the n -dimensional space.

2.3.3 Filtering techniques

Usually, the search space can be reduced by using filtering processes based on certain criteria. That is, the filtering processes are conducted first to select items satisfying certain criteria, such as those based on dates and author names, and the search based on complicated features is then carried out on the selected items only. As the number of selected items is significantly smaller than the total number of items in the database, the overall retrieval process can be completed quickly. In [101], shape eccentricity is used as the filtering criterion, i.e. only shapes within the specified eccentricity range need to be searched. In [102], triangle inequality is used to reduce the number of direct feature comparison in a database search. In [103], Striker and Dimai use histograms with very few bins to select potential retrieval candidates, and then use the full histograms to calculate accurate distances between the query and the potential candidates. The drawback of filtering is that additional information should be applied to enhance the filtering rate (keep the search space low) when database is getting larger and larger. In practice, the combination of indexing structures and filtering techniques can be used in different stages to speed up the retrieval process. Models are organized into a certain indexing structure. Filtering techniques are then used to eliminate some unnecessary comparisons.

2.4 Distance transform

The distance transformation (DT) is a powerful image processing transformation that assigns to object pixels in a binary image its distance to the background pixels. The operation converting a binary image to an approximated distance image is called a distance transformation. It can be used for a variety of applications such as thinning [104, 105, 106], skeletonization [107], pattern matching [108], clustering [109], etc. The fundamental idea of DT was first introduced by Rosenfeld and Pfaltz [110]. In [111], they discussed different digital distance functions including city blocks and chess board distances and described how they can be mixed to create octagonal DTs that were better approximations of the Euclidean distance, since Euclidean DT (EDT) generates complete error-free Euclidean distance maps. However, computing the Euclidean distance is essentially a global operation that is very expensive. Although many algorithms have been developed to improve the efficiency to compute EDT [112, 113, 114, 115], it is necessary to consider the local operation and give a reasonable approximation to the Euclidean distance [116]. Borgefors [117] investigated the performance of six distance transformations on how accurately they can approximate EDT and suggested two of them which outperformed the others: 3-4 DT and 5-7-11 DT.

In order to approximate the global Euclidean distances in the image by propagating local distances between neighboring pixels, all DTs can be described in graphical form as a $n \times n$ neighborhood “masks” (a 5×5 mask is shown in Figure 2.1(a)).

In parallel computation of DT [111, 118, 119, 120], a mask is applied by placing the center of the mask over each pixel in the image. The local distance in each mask-pixel C_n is added to the value of the image pixel “below” it. The new value of the image pixel is the minimum of all the sums. Another way to compute DT is a sequential algorithm [110, 121]. The symmetrical parallel mask is split into two masks, shown in Figure 2.1(b). The masks are passed over the image once each: the forward one from left to right, and from top to bottom, and the backward one from right to left

C ₅	C ₄	C ₃	C ₄	C ₅
C ₄	C ₂	C ₁	C ₂	C ₄
C ₃	C ₁	<u>0</u>	C ₁	C ₃
C ₄	C ₂	C ₁	C ₂	C ₄
C ₅	C ₄	C ₃	C ₄	C ₅

(a)

C ₅	C ₄	C ₃	C ₄	C ₅
C ₄	C ₂	C ₁	C ₂	C ₄
C ₃	C ₁	<u>0</u>		

		<u>0</u>	C ₁	C ₃
C ₄	C ₂	C ₁	C ₂	C ₄
C ₅	C ₄	C ₃	C ₄	C ₅

(b)

Figure 2.1: Masks describing the distance transformations, (a) Parallel mask for parallel computations, (b) Forward (left) and backward (right) masks for sequential computations.

4	3	4
3	0	3
4	3	4

Figure 2.2: The 3×3 neighborhood mask used in Chamfer distance transformation.

and from bottom to top. The new value of the “central” image pixel is the minimum of the sums of the image pixel values and the local distances in mask-pixel C_n . After these two passes, the distance transform is computed.

3-4 DT applies a 3×3 mask as shown in Figure 2.2. It is also called a “Chamfer” DT because the distance values are “chamfered out” in two passes over the image when the computation is sequential. Chamfer distance is mainly applied on Chamfer matching [122, 123, 124, 125]. We also make use of Chamfer distance in our work due to its efficiency and simplicity.

Besides conventional methods, DT has been developed in many other ways. In [126, 127], graph-theoretic approaches were presented. Saha [128] described the theory and algorithms of distance transform for fuzzy subsets, called fuzzy distance transform (FDT). Rodriguez-Ponce and Rosas-Romero [129] approximated distance transform function by a multilayered neural network with satisfactory results. In [130, 131], they proposed morphology approach to DT. An interesting “knight’s distance” was proposed in [132] which made use of the knight’s moves in the game of chess. A

knight's transform (KT) was also introduced, which can be seen as a variant of DT.

2.5 Hausdorff distance

Hausdorff distance (HD) is one of the commonly used measures for shape matching and has been widely investigated [37, 133, 134, 135, 136, 137, 138]. It is a distance defined between two sets of points. Unlike most shape comparison methods that build a one-to-one correspondence between a model and a test image, HD can be calculated without explicit point correspondence.

Given two sets of points $A = a_1, \dots, a_m$ (representing a model in the database) and $B = b_1, \dots, b_n$ (representing a test image), HD was defined as

$$H(A, B) = \max(h(A, B), h(B, A)) \quad (2.3)$$

where

$$h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\| \quad (2.4)$$

where $\|\cdot\|$ is any norm, e.g. Euclidean distance. The function $h(A, B)$ is called the directed Hausdorff distance from A to B. It identifies the point $a \in A$ that is farthest from any point of B, and computes the distance from a to its nearest neighbor in B. Thus $H(A, B)$ measures the degree of mismatch between two sets, as it reflects the distance of the point of A that is farthest from any point of B and vice versa. Intuitively, if $H(A, B)$ is d , then every point in A must be within a distance d of some point of B and vice versa. It is well known that the Hausdorff distance is a

metric over the set of all closed, bounded sets — it obeys the properties of identity, symmetry and triangle inequality.

Huttenlocher et al. proposed the partial HD measure in comparing partial portions of images containing severe occlusions or degradation [5]. The directed distance of the partial HD was defined as

$$h_K(A, B) = K_{a \in A}^{th} \min_{b \in B} \|a - b\| \quad (2.5)$$

$K_{a \in A}^{th}$ denotes the K^{th} ranked value of $\min_{b \in B} \|a - b\|$. The partial HD is naturally defined as

$$H_{LK}(A, B) = \max(h_L(A, B), h_K(B, A)) \quad (2.6)$$

where h_L is defined similarly. This HD measure needs to set one parameter, $f = K/N_A$, where N_A the number of element number of set A . The range of f is from 0.0 to 1.0. Depending on the fractional value of f , its performance widely varies. Experimentally, when f is about 0.6, good matching results are obtained. Olson and Huttenlocher enhanced their methods later in [139] by incorporating edge point orientations in the Hausdorff distance definition, which reduced the rate of false-alarm and enhances performance. However, their method required costly preprocessing which should be done off-line.

Dubuisson and Jain proposed the modified HD (MHD) based on the average distance value in comparing the synthetic images contaminated by four types of noise [140]. The directed distance of MHD was defined as

$$h(A, B) = \frac{1}{N_A} \sum_{a \in A} \min_{b \in B} \|a - b\| \quad (2.7)$$

The MHD measure does not require a parameter, however its matching performance is not as good as the partial HD since it is a sum over all the distances including the outliers.

Following the idea of ranked order statistics, Azencott et al. proposed the Censored HD (C-HD) measure in comparing binary images [141]. The directed distance of the C-HD is defined as

$$h_{K,L}(A, B) = P_{a \in A}^{th} Q_{b \in B}^{th} \|a - b\| \quad (2.8)$$

where $P_{a \in A}^{th}$ denotes the P^{th} ranked value of $Q_{b \in B}^{th} \|a - b\|$, and $Q_{b \in B}^{th}$ represents the Q^{th} ranked value of the norm distance set. The C-HD measure requires two parameters: $p = P/N_A$ and $q = Q/N_E$, where N_E denotes the size of the norm distance set. The range of both parameters is from 0.0 to 1.0. Experimentally, where p is from 0.8 to 0.9 and q is from 0.01 to 0.05, good matching results are obtained.

In [142], Sim et al. proposed the robust HD measures based on the M-estimator and least trimmed squares (LTS). These algorithms (M-HD and LTS-HD) can cope with a large percentage of noise because they are derived from the concept of robust statistics. The directed distance $h_M(A, B)$ of the proposed M-HD is realized by the cost function that can eliminate outliers [143]. M-estimation minimizes the summation of a cost function ρ . $h_M(A, B)$ is defined as

$$h_M(A, B) = \frac{1}{N_A} \sum_{a \in A} \rho(d_B(a)) \quad (2.9)$$

where $d_B(a)$ represents the minimum distance value at point a to the point set B ,

2.5 Hausdorff distance

39

and the cost function ρ is convex and symmetric:

$$\rho(x) = \begin{cases} |x|, & |x| \leq \tau \\ \tau, & |x| > \tau \end{cases} \quad (2.10)$$

where τ is a given threshold to eliminate outliers. In LTS-HD, the directed distance $h_{LTS}(A, B)$ is defined by a linear combination of order statistics

$$h_{LTS}(A, B) = \frac{1}{H} \sum_{i=1}^H d_B(a)_{(i)} \quad (2.11)$$

where H denotes $h \times N_A$, as in the partial HD case, and $d_B(a)_{(i)}$ represents the i^{th} distance value in the sorted sequence $(d_B(a)_{(1)} \leq d_B(a)_{(2)} \leq \dots \leq d_B(a)_{(N_A)})$. M-HD and LTS-HD are insensitive to outliers and occlusions and they yield efficiency larger than that of the conventional ones. However, parameter τ and h are necessary to be given beforehand.

Besides M-HD and LTS-HD, Sim et al. [144] proposed another variant of HD – oriented Hausdorff similarity (OHS) measure for object alignment. By adopting the robust similarity function, it is robust against outliers, where outliers are defined by anomalous data far from the assumed error distribution. OHS is defined as

$$H_{OHS} = \min(h_{OHS}(A_G, A_E, d_B), h_{OHS}(B_G, B_E, d_A)) \quad (2.12)$$

where $h_{OHS}(A_G, A_E, d_B)$ denotes the directed OHS, and A_E and B_E represents the edge images extracted from images A and B , respectively. The directed OHS is defined as

$$\begin{aligned} h_{OHS}(A_G, A_E, d_B) &= \sum_{a \in A_E} O_{A_G(a)} \cdot O_{B_G(a)} \rho_T(d_B(a)) \\ &= \sum_{a \in A} s(a) \rho_T(d_B(a)) \end{aligned} \quad (2.13)$$

Figure 2.3: Similarity function of $\rho_T(x)$.

where an orientation vector $O_{AG(a)}$ represents a unit gradient vector of A at position a , $s(a)$ denotes the dot product of two gradient vectors obtained from two images. $\rho_T(x)$ is a symmetric threshold function, as shown in Figure 2.3. Apparently, T is a given threshold.

2.6 Logo recognition overview

Since our proposed method is tested on logos, some logo recognition methods are briefly reviewed.

Logos are 2D shape of varying complexity, with interior and exterior contours, that are not necessarily connected. The problem of logo recognition is of great interest in document processing, especially for document database. By recognizing the logo, semantic information about the document is obtained which may be useful to decide

whether or not to analyze the textual parts. But the recognition process is difficult because of its complexity. There are not many published works in this area and therefore this is an interesting and challenging research topic.

Statistical approach

A trademark indexing and matching method based on low-level feature analysis was presented in [145]. The features used to index an image were the invariant moments, the histogram of the edge directions, and the mean and variance of the absolute values of the coefficients obtained from the sub-images of the first three levels of the multi-resolution wavelet transform of the image. A global metric used to evaluate the dissimilarity between two images i and j of the database was defined by linearly combining the distance between the individual features

$$Dissimilarity(X^i, X^j) = \sum_{h=1}^p w_h D_h(X_h^i, X_h^j) \quad (2.14)$$

in which the w_h are weights. Furthermore, a relevance feedback mechanism was applied in the method to update the weights of the different features in the overall evaluation of the image similarity and reformulate the user's query accordingly. The experiments were carried out using 20 test trademarks and a database of 1100 trademarks. For a test trademark, the system gave the correctly matched trademark within the top 24 matches. This method was efficient but not suitable for occluded and distorted trademark matching.

Another content-based logo retrieval method was proposed by Kim [146]. In this method, a novel visually salient feature was defined by utilizing the Zernike moments

of global shape of the trademark. The 2D Zernike moment of order (n, m) of an image, $I(\rho, \theta)$, in polar coordinate was defined as

$$A_{nm} = \frac{n+1}{\pi} \sum_{\rho} \sum_{\theta} [V_{nm}(\rho, \theta)] * I(\rho, \theta), s.t. \rho \leq 1. \quad (2.15)$$

Here, $V_{nm}(\rho, \theta)$ is a Zernike basis polynomial defined as

$$V_{nm}(\rho, \theta) = R_{nm}(\rho) \exp(-jm\theta) \quad (2.16)$$

and $R_{nm}(\rho)$ is radial polynomial

$$R_{nm}(\rho) = \sum_{s=0}^{n-|m|} (-1)^s \frac{(n-s)!}{s! \left(\frac{n+|m|}{2}\right) - s! \left(\frac{n-|m|}{2} - s\right)!} \rho^{n-2s} \quad (2.17)$$

$n = 0, 1, 2, \dots, \infty, |m| \leq n$, and $n - |m|$ is even. The degree of the similarity between the test trademark and the model was determined by Euclidean distance. This method has been examined with 10 test trademarks and a database of 3000 trademarks. Among the top 30 retrieved trademarks, the recognition rate was 92.5%. It was an efficient approach because the computation complexity for moment was low, and it can deal with certain noise that did not change the global structure of the shape. However, such method based on the moment was still not robust to occlusion.

Lin et al. [147] proposed a new region based shape description method, the distance angle pair-wise histogram, as the trademark shape feature vector to retrieve similar trademarks. Experiments have been conducted on about 1000 trademark images with better performance than other moment-based methods. Folkers and Samet [148] made use of Fourier descriptors to extract the contour of the shape, which was invariant against translation, scale and rotation. The descriptors, which were

claimed to have a good flexibility in choosing an arbitrary shape of the components of the query image, were used in a system to locate logos in an image database.

Syntactic/structural approach

Peng and Chen [149] proposed a new trademark recognition method based on closed contours. Trademarks were first decomposed into complete sets of elementary closed contours, each of which was coded as an angle-code string according to chain-code information. The string matching algorithm was then employed to compute similarities between closed contours. Finally, the maximum and average terms of the similarities between contours were integrated into the whole trademark similarity measure. After that, the similarity measure can be the criterion for retrieving similar trademarks. More specifically, since the proposed method to encode closed contour was insensitive to translation, rotation and size variations, it had the ability to detect infringement cases in which trademarks were changed in translation, rotation and size. The proposed method was implemented on a model database of 250 trademarks and a test set of 50 trademarks. For a test trademark, the method can give the most similar trademark within the top 8 matches. The authors declared that in general, most model trademarks similar to the test one can be retrieved. Such boundary based method was not applicable to the situation when there was a small opening in the boundary or there were boundaries wrongly connected to neighboring boundaries due to segmentation error. Besides, Cortelazzo et al. [150] presented a hierarchical contour representation and a string matching procedure for trademark recognition. However,

it was only useful for checking similarity between logos that had the same hierarchical representation. Hollander and Hanjalic [151] extracted strings from logos in a video to match with template strings extracted from the original logo pictures. Experiments performed on a sport video demonstrated the performance of the proposed string matching template. In [152], they proposed a method to retrieve trademarks using query by sketches. The similarity score between the strokes was calculated after finding the corresponding parts between the query strokes and the model strokes. Two strokes were considered similar if they were of different types.

Neural network

In [153], a new approach for logo recognition based on auto-associate artificial neural networks (AANN) was proposed. An AANN was a Multi-layered Neural Network (MLN) trained to force an identity mapping. It had one hidden layer with fewer neurons than the input and output layer. For each class of logos, an auto-associator based network was trained. The Euclidean distance between the input vector and output vector was the criterion that was used to determine the class of an unknown logo. This approach was significantly robust with respect to spot-noise. Recently, a generalized regression neural network [154] was presented to deal with the rotated and scaled logo. The network used in this method was capable of interpolating between data sets upon which it was trained to predict the probability that a target logo was presented in a test page, even when that logo was distorted in scale or rotation. By dividing the target logo into tiles for training, and searching the test page in tile-sized

regions, the likelihood of detecting the logo was increased. Other used techniques, such as eliminating all solid color tiles from the training and testing processes, and adjusting the acuity, or specialization, of the network, also increased the accuracy of target logo detection. This approach achieved an average recognition rate about 90% and 95% for test logos with rotation angles from -10 to +10 degree and scales from 90% to 110%. However, for neural network methods, a training stage was needed. More training examples would lead to better recognition result. Unfortunately, training was a very expensive process. Zyga [155] suggested to modeling the structure and functionality of the human retina for use as an automatic identification system for logo images in document processing. Scale, rotation, noise, and distortion invariant identification was achieved using a suitable classification algorithm and a reference database.

Hybrid approach

A method for representing and matching logos based on positive and negative shape features was presented by Soffer and Samet [156, 157]. They proposed a new representation of such symbols based on their interior with the shapes considered as holes, termed a negative symbol. Negative features were based on having some geometric shape (e.g. square or circle) that enclosed several smaller shapes. While, many logos by their design already had this property, others did not, and then they artificially added a border around these logos and thus created a negative symbol. They employed four global descriptors (invariant moment, circularity, eccentricity

and rectangularity) and three local shape descriptors (horizontal gaps per total area, vertical gaps per total area and ratio of hole area to total area) to represent logos. The similarity between the logos was measured by the weighted Euclidean distance.

In [158], a geo-statistical second-order moment known as the variogram was presented as a new approach for feature extraction for the neural recognition of trademark images. The initial experimental results have shown potential application of this method.

Mehetre et al. [159] indexed and matched a trademark database using combined measures (invariant moment, chain coded string, Fourier descriptor, etc.). The system performance was tested on a database of 500 trademarks and a test set of 15 trademarks. They achieved 93.8% correct rate within the top 5 matches. This approach was simple to implement but not robust to occluded and distorted trademarks. A content based database retrieval system [2] used a fast pruning method (i.e. indexing) making use of edge direction and invariant moment and a deformable template matching process to achieve correct matches of trademarks. This system has shown good results on whole shape image, however, it can hardly be robust to occluded image.

Chapter 3

System Overview

The shape recognition system comprises four modules, i.e. the preprocessing, shape representation, indexing and matching. The preprocessing module operates on raw inputs of the images. Outputs of the matching are dissimilarity distances between models and an input image. The model with the least dissimilarity with the input is proposed as the corresponding match. Figure 3.1 shows the flowchart of the whole system, while the following subsections will describe each process briefly.

3.1 Preprocessing

Preprocessing is the first module of the system which contains 3 steps (Figure 3.2). After scanning, the edge detection and thinning processes will be performed

Figure 3.1: The proposed system flowchart.

to segment the shape. The contour extraction step will then be employed to generate a list of consecutive pixel locations on each contour. The output of this module will be sent to the shape representation process.

Edges of intensity images are usually used as important features in image processing. Cognitive psychological studies [7, 57] indicated that human beings recognize

Figure 3.2: The preprocessing of the system.

line drawings as quickly and almost as accurately as gray-level pictures. These results imply that edge images of objects could be used for object recognition and achieve similar accuracy as gray-level images. The edge detection process is to detect pixel discontinuities of image intensity. This has been studied extensively. Heath et al. [160] had investigated the performance of different edge detectors. They compared the edge detectors based on experimental psychology and statistics, in which humans rated the output of low level vision algorithms. One of their clear results is “No one single edge detector was best overall; for any given image it is difficult to predict which edge detector will be best”. In this thesis, the proposed shape recognition method does not rely on any specific edge detector. Any edge detector can be used, and the implementation from Nevatia and Babu [161] is adopted in this study. After

Figure 3.3: An illustration of segmentation results, (a) original intensity image, (b) edge image, (c) thinned edges.

edge detection, a thinning process is used to reduce thick edges to chains of single pixels that can be easily traversed. Examples of the edge image and thinned edge are shown in Figure 3.3.

Contour is a compact way to represent the shape of an image. It stores sequential lists of consecutive pixel locations which can be represented by integer coordinates. Given a sequence of consecutive coordinate points $p_1(x_1, y_1), p_2(x_2, y_2), \dots, p_n(x_n, y_n)$, where p_i is connected to p_{i+1} ($1 \leq i \leq n - 1$), we have $|x_i - x_{i+1}|$ and $|y_i - y_{i+1}|$ both less than or equal to one but not both equal to zero.

3.2 Shape representation

This module includes two steps: polygonal approximation and feature refinement (Figure 3.4).

Figure 3.4: The shape representation flowchart.

Polygonal approximation

The polygonal approximation is employed here to represent shape. It extracts feature points from the generated contours of a shape. After feature point detection, every line segment that connects two consecutive points on the contour is used to represent the shape. Consequently, the shape of an object can be represented in a more compact way. Computation time can thus be saved.

Line segments are used as matching primitives here since they possess good properties, i.e. (1) line segment are easier to obtain from digital images compared to nonlinear features such as curves, (2) they represent higher level structure as compared to contour points, (3) line segments occupy less memory than contour point representation, (4) it is difficult to represent a curve if there are gaps on a contour (line representation however has no such problem). The representation is

Figure 3.5: An example of the line edge map. The feature points are circled.

also called the line edge map (LEM) in [162]. We make use of dynamic two-strip algorithm (Dyn2S) in this step [163]. One example of the representation is shown in Figure 3.5.

Feature refinement

A set of feature points is extracted from previous process to represent a shape. However, instead of generating one suitable feature point for a round corner, the process usually generates a few in the vicinity as shown in Figure 3.6(a). This cannot be avoided since a circle or circular arc should not have any outstanding feature point but the algorithm has to return some to represent such shape. It is desirable to have a refinement process to compute one appropriate point to represent each round corner as shown in Figure 3.6(b). This can be achieved by modeling and detecting a round corner appropriately.

Figure 3.6: One example of the feature point post-processing, (a) point set from Dyn2S, (b) after round corner processing, (c) after filtering.

3.3 Indexing

Instead of linearly searching all models in database, an indexing module is proposed to select a moderate number of candidate models for matching. Several relatively simple feature structures are extracted from each image. All the shape information of models is stored in an indexing structure. Then the input features are fed into the structure to search corresponding feature with similar geometric property. The process, which searches a suitable corresponding feature pair for alignment, becomes the indexing and filtering steps to eliminate non-suitable matches prior to the matching process. The proposed system also needs to align (normalize) an input with respect to each model since the input could have been captured in different scale and orientation. Fortunately, the corresponding feature pairs from input to model obtained from indexing can be utilized and the alignment can be done by simply computing a transformation matrix between the features. The proposed indexing

Figure 3.7: The proposed indexing scheme.

scheme is outlined in Figure 3.7.

3.3.1 Feature point selection

When the number of models in database is large, it is important to control the size of the indexing structure to speed up the indexing process. As a preliminary step of indexing, we need to decrease the number of features used to generate the indexing structure to satisfy this requirement. Meanwhile, these features (points) should be extracted consistently. Given a model and the input that can be matched to such model, it is necessary that the same feature point set can be extracted. Based on

this point, the feature point on a round or sharp corner is a good feature point while those on a circle or a circular arc are not. We model the circular arc and ignore all feature points on such arc for the next process. One example of the output generated from this process is shown in Figure 3.6(c).

3.3.2 Local structure construction

Each selected feature point P will form an indexing local structure with its 6-nearest neighboring feature points as

$$\mathbf{S}(\mathbf{P}) = \begin{bmatrix} (\theta_1^1, \theta_2^1, \theta_3^1), \\ (\theta_1^2, \theta_2^2, \theta_3^2), \\ \dots \\ (\theta_1^6, \theta_2^6, \theta_3^6) \end{bmatrix}$$

where $(\theta_1^i, \theta_2^i, \theta_3^i)$ describes the orientation relationships from P to its i^{th} neighboring point. Six nearest neighbors are chosen here because simple structure (with one or two neighboring features) cannot form a distinguishing local structure. The details will be given in a later chapter. For practical consideration, we can later tackle the issue of missing neighbors in noisy or occluded environment by allowing matches of two structures based on the best n nearest neighbors with $0 < n < 6$. One example of the local structure extent is shown in Figure 3.8.

Figure 3.8: An example of a local structure with its center (the star) and the 6 nearest neighbors.

3.3.3 Indexing structure generation

When the numbers of inputs and model images are large, the price for searching all possible matches from inputs to models is high. To cut down CPU time, an indexing structure can be constructed to store all the feature information of models to facilitate prefiltering. A 3-dimensional hyper-cube $\tilde{I}$ is employed to conglomerate all the local structure of the model images and work as the framework for indexing. The three dimensions of the hyper-cube correspond to the 3-D attribute vector from previous process $((\theta_1, \theta_2, \theta_3))$. A similar cube is also constructed for the input image and it is intersected with $\tilde{I}$. Each pair (model, input) votes for the corresponding features that have the most intersected entries.

3.3.4 Indexing

The intersected cubes from model and input will identify the non-empty entries first. The weak entries, which identify the feature pairs from the two images that are

not similar enough, are filtered out. If input and model have no overlaid strong entry, the model is excluded from further processing. The strongest entries which have the most similar neighbors in the overlaid cubes represent the top ranking match from one local structure of a model to another local structure of the input, and contribute to the extraction of corresponding features from model to input. A transformation matrix can then be derived to align the corresponding features and subsequently to align the input to the model.

3.4 Matching

We first modified the existing LHD (line segment Hausdorff distance) technique for better performance. Then a new curve segment Hausdorff distance method is proposed to finish the matching task (Figure 3.9).

Figure 3.9: The proposed matching scheme.

3.4.1 Matcher selection

Before the matching can be carried out, a distance transform process is needed to compute the nearest model curves to each input curve. Distance transform (DT) assigns to a background pixel the approximated distance to the nearest object pixel. It is an efficient technique to find the nearest object. We make use of the Chamfer metric DT algorithm [117] and apply a 3×3 neighborhood mask (Figure 2.2) on an input LEM.

3.4.2 Curve segment Hausdorff distance

An input image is matched to all the likely models in a curve-to-curve matching manner in this process. Each curve is in fact a sequence of consecutive line segments. The matching is implemented as a dissimilarity computation process, which is based on the geometric distance between two sets of curve segments. After the curve segments of an object have been compared with all the curve segments from candidate models, the model with the lowest dissimilarity is proposed as a match.

Chapter 4

Shape Representation

The initial and very important step in shape recognition procedure is to define a set of meaningful shape descriptors. The chosen descriptor should be efficient in its use of computation, as well as faithful in providing a usable representation of the original shape because this choice directly influences the efficiency of the shape recognition system and whether it is invariant with respect to geometric transformation, such as translation, rotation, and scale changes. Shape descriptors may be classified as being either external or internal descriptors depending on whether they code the boundary of a shape (external descriptor) or the area within the boundary (internal descriptors). Descriptors may also be classified as being either scalar transform or space domain. Scalar transform methods generate shape descriptors which are

mathematical summations of the shape whereas space domain techniques comprise descriptors which express the structural and relational properties of the shape.

Boundary representation of shape and its approximation economizes memory storage and processing time for subsequent procedures. Attneave [164] proposed that information along a contour is concentrated in the regions of high magnitude of curvature. He joined these high curvature points with straight lines, which are in fact external descriptors, suggesting that not much loss of information had occurred. Following Attneave's observation, there are many approaches developed for detecting these dominant points and to seek the best linear polygonal approximation of the digital curves. The first category is to extract dominant points by curvature evaluation, which depends upon the accuracy of curvature evaluation and correct determination of the region of support at each contour point. The second category, polygonal approximation, is to fit the longest straight line segments along the contour. Its goal is achieved by capturing the essence of the boundary shape with the fewest possible line segments and vertices, which is more logical and accurate.

4.1 Dynamic two-strip algorithm (Dyn2S)

Dyn2S is a curve representation method using feature points which this work is based on due to their flexibility in computing the support for a feature point. The whole procedure has two stages. In the first stage, important properties of the data points are derived. Two strips (i.e. the supports) are fitted to the left and right

sides of each point on the curve respectively and the points inside each strip can be approximated by the left or right straight line running from one end of the strip to the other end. The best fitted strips are derived by adjusting the width of the strip and rotating it up or down dynamically with the starting point as the pivot. Longer and narrower strips are preferred. The ratio, which is called the elongate value of a strip, is defined as $E = L/W$ of its length L over its width W to represent the merit of fit. The merit f for a point can be computed as

$$f = E^{left} \cdot \theta \cdot E^{right} \quad (4.1)$$

where E^{left} and E^{right} are the elongate values of the left and right strips, and $\theta = |180^\circ - \alpha|$ is the acuteness of the smaller angle, α , between the two stripes. In the second stage, a subset of the most representative points is selected based on the local maxima of point merits.

One example is illustrated in Figure 4.1 where the minimum and maximum strip widths can be easily pre-selected as 1 and $0.1 \times D$ pixels, where 1 is considered small enough while D is the length of the image diagonal. Let P_0 be the current point to be processed, $L_i (i = 1, 2, \dots, 6)$ be points on P_0 's left side and $R_j (j = 1, 2, 3)$ be points on P_0 's right side. An initial strip of minimal width would first extend from P_0 to encompass as many points as possible to its left side (i.e. the dashed-line box). Its width is then increased and its orientation is adjusted to include as many points as possible, until the maximum strip width is reached (i.e. the left solid-line box). The measure of fit (i.e. elongate value) is recorded for each strip as a function of

Figure 4.1: Illustration of the dynamic two-strip algorithm.

strip width but only the strip with the highest value is marked and used. The same procedure is applied on the other side of the point.

There are two advantages in using strips to extract local features. First, digitization noise can be tolerated because the strip has width and it can enclose points that can be approximated as a straight line. Second, the support region of one point can be reflected honestly by the strips.

However, it does not work well on circular curves which have no obvious feature points. This gives rise to inconsistent point sets being generated on different instances of the same model. This is not a problem if we are looking for a representative scheme for the purpose of data reduction. In the proposed work, we however need to find feature points that can represent round corners which are circular curves at a smaller scale. In addition, Dyn2S can sometimes generate feature points at the mid of a seemingly straight line as shown in Figure 4.2. The redundant feature points for round corners are labelled with dashed-line boxes, while those at the mid of a line are labelled with dashed-line circles. By analyzing Dyn2S algorithm, we found that the

Figure 4.2: Redundant feature points detected at the round corners and in the mid of a straight line.

selection of feature points is based on the merit (f , formula 4.1) of points along the contour. On seemingly straight lines, each feature point has large E_{left}/E_{right} with non-zero turning angle. The product of them gives large f (merit value) which causes one of these points to be chosen. These points can be easily eliminated by excluding feature points with turning angle less than a small threshold (e.g. 10 degrees). As for the points on a round corner, it is desirable to get one appropriate point to represent a round corner instead of several points. This process will be described in detail in next section.

4.2 Round corner point refinement

The motivation of the refinement is illustrated in Figure 4.3 where feature points B and D are preferred to be combined as one feature point C . The refinement process

Figure 4.3: An illustration of feature refinement; Δ_1, Δ_2 : the new fitting errors; M : the highest point of $\widehat{BD}$.

can be seen as “When” and “How” to combine the feature points.

Given a digital curve with n extracted feature points $\{P_1, P_2, \dots, P_n\}$, we have P_1 being the succeeding point of P_n if the curve forms a closed loop. We further define arc $\widehat{P_i P_{i+1}}$ as the sequence of consecutive points from the circular contour between P_i and P_{i+1} , chord $\overline{P_i P_{i+1}}$ as the line segment connecting points P_i and P_{i+1} , and $|P_i P_{i+1}|$ as the length of $\overline{P_i P_{i+1}}$.

Necessary condition for refinement

We model a round corner as shown in Figure 4.3 where there must be a bow shape curve from point B to D . The height h of this segment is measured as

$$h = \max_{M \in \widehat{BD}} \text{Perp}(M, \overline{BD}) \quad (4.2)$$

where $\text{Perp}(M, \overline{BD})$ denotes the perpendicular distance from point M (the highest point) to line $\overline{BD}$. In addition, we need points A and E to check against the new errors if point C is going to replace points B and D . Let A, B, D, E be P_i, P_{i+1}, P_{i+2} and P_{i+3} . We can summarize the steps in finding a suitable round corner as shown in Figure 4.4 where **Criterion 1** is the necessary condition to discover and accept

4.2 Round corner point refinement

65

one suitable round corner.

Criterion 1

Let P_m be the mid-point of $\widehat{P_{i+1}P_{i+2}}$ in Figure 4.5(a). Let Δ_1 and Δ_2 be the distances from P_{i+1} to $\overline{P_iP_m}$ and P_{i+2} to $\overline{P_mP_{i+3}}$ respectively. Two elongate values can be defined as

$$\text{elongate}_1 = \frac{|P_iP_m|}{\Delta_1} \quad (4.3)$$

$$\text{elongate}_2 = \frac{|P_mP_{i+3}|}{\Delta_2} \quad (4.4)$$

Besides, we define the minimum arc height, h' , of $\widehat{P_{i+1}P_{i+2}}$ as

$$h' = \frac{1}{5} \cdot H \quad (4.5)$$

where H is the height of the triangle $P_iP_{i+1}P_{i+2}$ with $\overline{P_{i+1}P_{i+2}}$ as the base line (Figure 4.5(b)). Let the two base angles of the triangle be denoted as θ_1 and θ_2 , H can be derived as

$$H = |P_{i+1}P_{i+2}| \cdot \frac{\sin \theta_1 \cdot \sin \theta_2}{\sin(\theta_1 + \theta_2)} \quad (4.6)$$

Formula 4.5 is designed to work on a round corner. We do not want to apply the feature point refinement process on a flat corner if the corner is made from the lines $\overline{P_iP_{i+1}}$, $\overline{P_{i+1}P_{i+2}}$ and $\overline{P_{i+2}P_{i+3}}$. Hence, we need a measure of height threshold, h' . Any corner with height less than h' is considered flat. In this implementation, we set h' to have a low value, i.e. 20% of H . We consider a ratio of 20% (or 5) is low enough such that we will not miss any round corner.

Figure 4.4: The algorithm flowchart.

Figure 4.5: Illustration of **Criterion 1**.

“Criterion 1” can then be defined as the combination of the following two conditions

- (1). $elongate_1 > 10$ and $elongate_2 > 10$
- (2). $h + 1 > h'$.

Condition (1) controls the errors of the new combination. It should be applied on round corner with long arms on the left and right. This consideration will save us from doing round corner fit on a circle like object. The support of long arm can be computed as the “elongate” value. For a circle, the usual elongate value is from 5.0 to

8.3. Hence, we set the threshold as 10 to retain the feature points on circle like shape from combination. Condition (2) enforces the necessary bow shape of curve between P_{i+1} and P_{i+2} . h is the real height of arc $\widehat{P_{i+1}P_{i+2}}$. The “1” in condition (2) caters for the case when both h and h' are zero due to integer arithmetic computation.

Corner refinement

The new feature point P_N obtained by combining P_{i+1} and P_{i+2} is generated as

$$P_N = \begin{cases} P_{i+1} & \text{if } \theta_1 - \theta_2 > \pi/3 \\ P_{i+2} & \text{if } \theta_2 - \theta_1 > \pi/3 \\ P' & \text{otherwise} \end{cases} \quad (4.7)$$

The first term specifies that P_N should be P_{i+1} if P_{i+1} is much sharper than P_{i+2} where θ_1 and θ_2 are the turning angles from $\overline{P_iP_{i+1}}$ to $\overline{P_{i+1}P_{i+2}}$ and then to $\overline{P_{i+2}P_{i+3}}$. Here, we use one threshold to determine whether we should keep one of the feature points by measuring the difference of the turning angles. If the difference is large enough, one of the feature points must be much sharper than the other. Hence, we do not need to compute P' (i.e. save computation). The difference of $\pi/3$ is considered rare and large. The performance and quality should not be much different if we pick other reasonable large thresholds since we are just picking one point in between points P_{i+1} and P_{i+2} . The second term can also be explained in a similar manner. The third term, P' , is one of the points in between P_{i+1} and P_{i+2} . Its position on the round corner is linearly proportional to the turning angle difference $(\theta_1 - \theta_2)$ as shown in Figure 4.6, i.e. P' should be closer to the sharper point as

Figure 4.6: The step-wise linear relationships between P_N and $(\theta_1 - \theta_2)$. Figure 4.7: The sine function modeling of P_N in the middle of P_{i+1} and P_{i+2} .

$$P' = \frac{3}{2\pi}(P_{i+2} - P_{i+1}) \times (\theta_1 - \theta_2) + \frac{1}{2}(P_{i+1} + P_{i+2}) \quad (4.8)$$

where P_{i+2} and P_{i+1} refer to the vector values of point P_{i+2} and P_{i+1} . All the operators in the formula can follow the rules of vector operation. Hence, $P_{i+2} - P_{i+1}$ is the vector difference between them and $P_{i+2} + P_{i+1}$ refers to the vector value of the mid-point between them. Alternatively, P' can be modeled by the sine function as in Figure 4.7 with

$$P' = \frac{(P_{i+2} - P_{i+1})}{2} \times \sin\left(\frac{3}{2} \times (\theta_1 - \theta_2)\right) + \frac{1}{2}(P_{i+1} + P_{i+2}) \quad (4.9)$$

We use formula 4.9 in this work. Consequently, formula 4.7 can be rewritten as

$$P_N = (P_{i+2} - P_{i+1}) \times \left(\frac{1 - \text{sgn}(|x| - \pi/3)}{2} \times \sin(x) \right) + \frac{\text{sgn}(|x| - \pi/3) + 1}{2} \times \text{sgn}(x) + \frac{1}{2}(P_{i+1} + P_{i+2}) \quad (4.10)$$

where

$$\text{sgn}(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x < 0 \end{cases} \quad (4.11)$$

4.3 Results

The proposed method has been implemented using the C++ language running on a DELL workstation. Two examples after polygonal approximation and round corner point refinement are shown below (Figure 4.8 and 4.9). More examples can be found in appendix A.

From the experimental results, we found that our combination criterion is not good at processing detailed local shape information. Some feature points which are expected to be retained on small-sized angles are filtered out. Because from a small scale view (e.g. pixel-level), it is difficult to distinguish a small-sized angle, a small round corner and even a very short line. The missed feature points are considered locating on a segment of flat contour and combined with the nearest feature point on sharp corner. Several examples are given in Figure 4.10. The disadvantage of this method also limits its usage. For example, when the images with details of low

resolution (palmprints and stationeries shown in Figure 1.1) are processed, corner refinement is difficult to be applied.

4.4 Summary

In this section, a novel refinement algorithm is proposed to refine the position of the feature points. It provides a consistent and compact shape representation conform to human perception. In next chapter, the refinement algorithm will also contribute to the indexing and normalization process.

(a) Result obtained from Dyn2S

(b) Result obtained after round corner point refinement

Figure 4.8: Comparison of the results before and after the corner refinement.

(a) Result obtained from Dyn2S

(b) Result obtained from round corner point refinement

Figure 4.9: Comparison of the results before and after the corner refinement.

Figure 4.10: Examples of false refinement. The first column gives the original images while the second column is the results after refinement. The bolded circles indicate the parts with unsatisfactory refinement.

Chapter 5

Indexing and Alignment

When the numbers of input and model images are large, the price for brute force matching between the input and model images is high. To cut down CPU time, one can do image indexing in order to narrow down the scope for a match. In this study, simple and easily computed shape features are designed to index all the models and generate a moderate number of likely models with respect to an input. When the group of models for matching is generated, the normalization can be done as a byproduct without extra process.

Some researchers have attempted to develop indexing systems based on multiple features that describe the image content. The QBIC (query by image content) [165] system allows users to search through large online image database using queries based

on shape, color, texture and position. Photobook [166] is a set of interactive tools for browsing and searching an image database. Users can choose image features based on the appearance, shape and texture to index large database. Jain and Vailaya [167], Lam et al. [168] and Kim et al. [169] used a combination of color and shape features for indexing. Brunelli and Mich [170] analyzed the image indexing system based on color and luminance. The above schemes are based on multiple image features and proven more effective than the indexing systems based on single feature. However, there is no one simple way to decide which features (shape, color and texture) to be used for a particular application. Furthermore, the systems were designed for interactive usage, not for automatic indexing and retrieval. In this study, our indexing process is based on shape attributes that can support automation design. In order to develop an effective indexing method that is robust to employ under noisy condition, an indexing method incorporating local structural and global spatial information is proposed. As explained in section 3.3, this module comprises four steps. The details of the processes will be described in the following sections.

5.1 Feature point selection

In this study, the images have been preprocessed to obtain their polygonal representation and refined feature points. Therefore, the indexing using feature points is preferred. We propose a feature structure which is expected to be consistent in noisy and occluded environment to facilitate indexing, prefiltering and alignment. Since the

proposed structure derives from the feature points extracted from last process, only the most reliable feature points are selected to guarantee (1) an indexing structure of moderate size, (2) consistent feature extraction.

Given a model and the input that can be matched to such model, it is necessary that the same feature point set can be extracted. Based on this requirement, the feature point on a round or sharp corner is a good feature point while those on a circle or a circular arc are not. We model the circular arc and ignore all feature points on such arc for next process.

By analyzing the geometric characteristics of the points on round or sharp corners which should be retained and those on circular arc which are expected to be removed, we find that it is difficult to distinguish them by merely utilizing the positions of the feature points only. We should also make use of the edge points on the contour between feature points. For example, in Figure 5.1, the three feature points have same coordinates in (a) and (b). However, P'_i in (b) is preferred to be retained rather than P_i in (a) since feature points like P_i have equal probability to appear

Figure 5.1: Feature points P_i/P'_i on circular arc (a) and round corner (b).

at any position along a circular arc. Hence, P_i cannot be consistently detected at the same location. We notice that the areas between the contour parts and their corresponding chords (the dashed line segments) in the two cases are different (the shadowed areas in Figure 5.1). This observation is employed to design the feature point selection algorithm. For feature point P'_i , the two areas beside it are compared with two similar areas (as in Figure 5.1(a)) formed between a circle (determined by the points P'_{i-1} , P'_i and P'_{i+1}) and the two chords $\overline{P'_{i-1}P'_i}$, $\overline{P'_iP'_{i+1}}$. If the differences of these two pairs of areas are within an acceptable tolerance, P'_i is considered to be from a circle or a circular arc and P'_i will not be retained.

Let the feature points obtained from last module be $P_1, P_2, \dots, P_n$. The new terminologies and concepts used in our algorithm are introduced first for clear illustration.

- $Cir(P_{i-1}P_iP_{i+1})$: Three points ($P_{i-1}P_iP_{i+1}$) (not collinear) determine a unique circle, denoted as $Cir(P_{i-1}P_iP_{i+1})$, whose center is the intersecting point of $\overline{P_{i-1}P_i}$ and $\overline{P_iP_{i+1}}$'s perpendicular bisectors (Figure 5.2).

Figure 5.2: An example of $Cir(P_{i-1}P_iP_{i+1})$.

5.1 Feature point selection

79

- C : The center of $Cir(P_{i-1}P_iP_{i+1})$.
- R : The radius of $Cir(P_{i-1}P_iP_{i+1})$.
- S_1 : The area between $\overline{P_{i-1}P_i}$ and curve $\widehat{P_{i-1}P_i}$. An example is given in Figure 5.3 (the shadowed area). If there are m points located along $\widehat{P_{i-1}P_i}$, denoted as $D_1(= P_{i-1}), D_2, \dots, D_m(= P_i)$,

$$\begin{aligned} S_1 &= \sum_{j=2}^m S(j-1, j) \\ &= \sum_{j=2}^m (Perp(\overline{D_{j-1}D_j}, \overline{P_{i-1}P_i}) \cdot (\overline{D_{j-1}D_j})') \end{aligned} \quad (5.1)$$

where $Perp(\overline{D_{j-1}D_j}, \overline{P_{i-1}P_i})$ denotes the perpendicular distance from $\overline{D_{j-1}D_j}$'s midpoint to $\overline{P_{i-1}P_i}$. $(\overline{D_{j-1}D_j})'$ denotes the length of the projection of $\overline{D_{j-1}D_j}$ on $\overline{P_{i-1}P_i}$, as shown in Figure 5.4. Their product $S(j-1, j)$ approximates the area of the small trapezia enclosed by $\overline{D_{j-1}D_j}$ and $(\overline{D_{j-1}D_j})'$. S_1 is obtained by accumulating all these products to approximate the area between $\overline{P_{i-1}P_i}$ and $\widehat{P_{i-1}P_i}$. $S(j-1, j)$ will be assigned positive or negative depending on which side

Figure 5.3: An example of S_1 .

Figure 5.4: Illustration of formula 5.1.

D_j resides with respect to vector $\overrightarrow{P_{i-1}P_i}$, i.e. positive for right side, negative for left side. Different signs might appear alternatively for the same curve segment. Two representative cases are given in Figure 5.5. In (a), all the $S(j-1, j)$ s are negative whereas in (b) both negative and positive signs coexist.

- S_2 : The area between $\overline{P_iP_{i+1}}$ and $\widehat{P_iP_{i+1}}$. It is defined in a similar manner as S_1 .
- S'_1 : The area between $\overline{P_{i-1}P_i}$ and the circular arc on the circle $Cir(P_{i-1}P_iP_{i+1})$ from point P_{i-1} to P_i , which can be computed using the following formula.

$$S'_1 = \begin{cases} R \cdot \angle P_{i-1}CP_i - S(\triangle P_{i-1}CP_i) & \text{if } \angle P_iP_{i+1}P_{i-1} < 90^\circ \\ R \cdot (2\pi - \angle P_{i-1}CP_i) + S(\triangle P_{i-1}CP_i) & \text{if } \angle P_iP_{i+1}P_{i-1} \geq 90^\circ. \end{cases} \quad (5.2)$$

where $S(\triangle)$ denotes the area of a triangle. $\angle ABC$ denotes the smaller angle formed by $\overline{AB}$ and $\overline{BC}$. The two situations in formula 5.2 cater to the cases shown in Figure 5.6.

Figure 5.5: Illustration of S_1 and S_2 calculation.

- S'_2 : The area between $\overline{P_i P_{i+1}}$ and the circular arc on the circle $Cir(P_{i-1} P_i P_{i+1})$ from point P_i to P_{i+1} . It is defined in a similar manner as S'_1 . We distinguish positive or negative value for S'_1 and S'_2 depending on the circular curves' loca-

Figure 5.6: Illustration of S'_1 calculation.

tion with respect to the vectors $\overrightarrow{P_{i-1}P_{i+1}}$. They will be assigned positive if P_i appears on the right side of $\overrightarrow{P_{i-1}P_{i+1}}$ or negative for left side. This mechanism is designed in order to distinguish the two cases as shown in Figure 5.7. Although the absolute values of S'_1 and S'_2 from both cases are almost the same, the distribution of the three feature points in (a) and (b) are different. Besides, the sign assignment is also consistent to the computation of S_1 and S_2 .

Figure 5.7: Illustration of the signs in S'_1 and S'_2 calculation.

The details of the selection process is summarized below (Figure 5.8).

Step 1: For each feature point P_i , the point set $(P_{i-1}P_iP_{i+1})$ and the contour part between them are analyzed to compute R , C , S_1 , S_2 , S'_1 , S'_2 .

Step 2: For each feature point P_i from the image, if it satisfies Criterion 2, an elimination flag is assigned to it.

Step 3: After all the feature points have been processed, remove those which are flagged for elimination.

Criterion 2

P_i satisfies Criterion 2 if both of the following two conditions occur.

Figure 5.8: A flowchart of the feature point selection algorithm.

- a. $|S_1| \in [\frac{1}{2} \cdot |S'_1|, 2 \cdot |S'_1|]$ and $S_1 \cdot S'_1 > 0$.
- b. $|S_2| \in [\frac{1}{2} \cdot |S'_2|, 2 \cdot |S'_2|]$ and $S_2 \cdot S'_2 > 0$.

According to Criterion 2, the difference between the areas of S_1 and S'_1 (or S_2 and S'_2)

Figure 5.9: Illustration of **Criterion 2**.

should be within an acceptable range. A reasonable threshold is to set the minimum value of S_1 (or S_2) to be at least half of S_1' (or S_2') and maximum value to be at most double of S_1' (or S_2'). Besides, S_1 and S_1' (or S_2 and S_2') are required to be both positive or negative. P_i should be retained when one is positive while the other is negative as shown in Figure 5.9. In (a), P_i is preferred to be retained although $|S_1|$ is similar to $|S_1'|$ and $|S_2|$ is similar $|S_2'|$. In summary, Criterion 2 judges whether the shape structure between P_{i-1} , P_i and P_{i+1} coincides with the shape characteristics of an arc segment.

An example is given in Figure 5.10 to illustrate the outcome of feature point selection. The feature points retained after the feature point selection process are input to the local structure construction.

5.2 Local structure construction

To employ the feature points from the LEM of the image, further description of features are required to be extracted. In this study, we introduce a shape descriptor to

Figure 5.10: (a) LEM before feature point selection, (b) LEM after feature point selection.

describe the coarse distribution of the local structure with respect to a given feature point of the shape. The indexing process is then equivalent to retaining the models containing the feature points that has the most similar local structure to the feature point of input shape.

The richer information the local structure contains, the easier the correspondence searching is, e.g. the gray-scale window is used instead of the brightness at a single pixel. Rich structure descriptor reduces the ambiguity in matching. Belongie [171] designed a shape descriptor – shape context which could play a role in solving the correspondence problem. Each point was assigned a shape context which derived from all the rest sample points of the shape. A matching approach based on shape context was proposed and claimed to be invariant under scaling and translation. However,

it was not robust enough to geometric distortions and occlusions since global shape structure was applied. We shrink the reference range of a feature point from the whole shape to its 6-nearest neighboring feature points [172] and generate an appropriate local feature structure. In our work, n is set to 6 according to our experimental results, which will be explained in section 7.1.

The process of the local structure extraction can be summarized as the following 3 steps.

Step 1: For each feature point P_i , an angle value $Ang(P_i)$ and an angle direction $Dir(P_i)$ are computed. $Ang(P_i)$ is the smaller angle formed by its two strips (θ in formula 4.1) and $Dir(P_i)$ indicates the direction of the angle bisector of $Ang(P_i)$ (shown in Figure 5.11).

Figure 5.11: Illustration of $Ang(P)$ and $Dir(P)$.

Step 2: For each feature point P_i , rank all the other feature points according to their Euclidean distances to P_i as $P_{i1}, P_{i2}, \dots, P_{i(n-1)}$ (assuming n points altogether).

Step 3: An indexing local structure of P_i is computed as

$$S(P_i) = \begin{bmatrix} (\theta_1^1, \theta_2^1, \theta_3^1), \\ (\theta_1^2, \theta_2^2, \theta_3^2), \\ \dots \\ (\theta_1^6, \theta_2^6, \theta_3^6) \end{bmatrix}$$

where $(\theta_1^q, \theta_2^q, \theta_3^q)$ ($q = 1, \dots, 6$) is denoted as $GA_q(P_i)$, which describes the orientational relationship from P_i to its q^{th} neighboring feature point P_{iq} . θ_1^q is the angle from $Dir(P_i)$ to $\overrightarrow{P_i P_{iq}}$ in anti-clockwise direction ($0^\circ - 360^\circ$). θ_2^q is the angle from $\overrightarrow{P_i P_{iq}}$ to $Dir(P_{iq})$ in anti-clockwise direction ($0^\circ - 360^\circ$). θ_3^q is $Ang(P_{iq})$. Their relationship can be illustrated in Figure 5.12.

Figure 5.12: An example of local structure.

The proposed feature structure $S(P_i)$ can be used to reconstruct the local structure around P_i without much information loss. Six nearest neighbors are chosen as refer-

ence points because simple structure (with one or two neighboring features) cannot form a distinguishing local structure.

5.3 Indexing structure generation

5.3.1 Integration of all neighbors

A 3-dimensional hyper-cube (H) (Figure 5.13) is employed to conglomerate all the local structure of one image and work as the framework for indexing. The three dimensions of H corresponds to the 3-D attribute $GA_q = [\theta_1^q, \theta_2^q, \theta_3^q]$. The input to this integration process is $\{GA_q, ID_r, NG_q\}$ where

- ID_r is the current reference feature (local structure) identity,
- NG_q is the q^{th} neighbor of ID_r ,
- GA_q is the geometric relationships from ID_r to NG_q .

H is built step-by-step as $H(i, j, k) = H(i, j, k) \cup (ID_r, NG_q)$ over all reference features and their neighbors. In the union process, ID_r and NG_q are the information to be stored with $i = [\theta_1^q/\theta_{sep}]$, $j = [\theta_2^q/\theta_{sep}]$, $k = [\theta_3^q/\theta_{sep}]$. The range of the three attributes are $[0^\circ, 360^\circ]$, $[0^\circ, 360^\circ]$ and $[0^\circ, 180^\circ]$. The value of θ_{sep} determines the number of bins into which the corresponding ranges are divided. Small θ_{sep} will make the corresponding feature selection very strict and unable to tolerate noise and distortions. On the other hand, a large θ_{sep} will reduce the number of bins (entries), which depresses the distinguishing ability of the mechanism and causes false

Figure 5.13: Illustration of H .

corresponding feature because many angle values will crowd into one entry. In this work, θ_{sep} is set to 30° [171]. Therefore, H has $12 \times 12 \times 6$ entries (elements). It is possible that i , j or k falls near bin margin. This can be tackled by growing the separating range from 30° to 40° (with a tolerance angle of 5° at each side). A 10 degree region overlapped by the neighboring bins is set as an “intergradation” scope. Angle value within this scope is stored into the two neighboring entries simultaneously, e.g. $[350^\circ, 310^\circ, 30^\circ]$ is stored at position $(12, 11, 1)$ and $(12, 11, 2)$. Moreover, this mechanism is only applied to H of the model image. The computation of H of an input image does not go through this process. It is because the correction needs to be done only once. As for how to choose the value of θ_{sep} and tolerance angle, it will be explained in section 7.1.

5.3.2 Integration of all models

Once we have constructed H for each model image, we can take union of H s to build one indexing structure ($\tilde{I}$) with the same dimensions as H . ($\tilde{I}$) is built step-by-step similarly as

$$\tilde{I}(i, j, k) = \tilde{I}(i, j, k) \cup (ID_m, ID_r, NG_q) \quad (5.3)$$

where ID_m is the corresponding model identity. This structure captures all the model geometric relationships of local structures, ready for intersection with an input H to compute and isolate the most similar feature with respect to the features from input image.

5.4 Indexing

From last step, all the required model information has been stored in $\tilde{I}$. Then, the input image's feature points are extracted and a local structure is computed for each point. Our purpose is to search all the models in the indexing structure with similar structures that can be found from the input image. Thus the models without explicit correspondence to input can be filtered out.

With $\tilde{I}$, one can perform the corresponding search with respect to an input hyper cube, H_{input} , in 3 steps: (1) intersect, (2) filter, and (3) rank. First, H_{input} is intersected with $\tilde{I}$ to identify non-empty entries from both structures. Each entry could contain numerous (ID_m, ID_r, NG_q) . Weak ID_m is then filtered out to compose

a candidate list X . The weak ID_m here refers to the model without any strong neighborhood match with H_{input} . These models will be filtered out from the next matching process. Since each neighborhood is composed of 1 feature and 6 neighbors, we define a weak response as a match with less than half of the neighbors being matched successfully, i.e. the intersection needs to produce 3 (or more) entries of (ID_m, ID_r, NG_q) for strong neighborhood match when m and r are fixed.

Since the input could have been captured in different scales and orientations, the proposed system needs to align (or normalize) an input with respect to each model prior matching. To solve this problem, each pair of model and input votes the corresponding features which have the most intersected entries. A transformation matrix can be obtained from the corresponding features $S(P_i^I)$ from input and $S(P_j^M)$ from model. We need to extract two pairs of points to derive the transformation matrix to align the input to the model. Point pair (P_i^I, P_j^M) is a certain choice. Another pair is selected from their matched neighboring points, which has the minimal angle difference $(\sum_{k=1}^3 |\theta_k(P_i^I) - \theta_k(P_j^M)|)$. When two LEMs have been aligned, they will be matched in detail employing the proposed CsHD (curve segment Hausdorff distance) technique to be described in the next chapter.

5.5 Summary

This chapter describes two important processes (i.e. indexing and normalization) in the proposed shape recognition system. The indexing method is based on local

5.5 Summary

92

structures extracted from feature point and the spatial distribution in its neighboring areas. The normalization problem can be solved simultaneously with our indexing method. In next chapter, the final matching will be illustrated in detail.

Chapter 6

Curve Segment Hausdorff Distance

After the indexing and alignment process, the input image is matched to all the likely models in a curve-to-curve matching manner. The matching is implemented as a dissimilarity computation process. In this chapter, the curve segment Hausdorff distance (CsHD) measure is proposed to match images. The proposed approach has better distinctive capability (especially for distorted images) than other HD methods. Moreover, this approach has the advantage to incorporate structural and spatial information to compute the dissimilarity between two sets of curve segments rather than two sets of points. The added information can conceptually provide more and better distinctive capability for recognition.

6.1 Line segment Hausdorff distance

Most Hausdorff distance (HD) methods (reviewed in section 2.5) work on sets of points. Gao and Leung [162] generalized the non-explicit feature correspondence concept of HD to higher level line feature. They proposed a line segment Hausdorff distance (LHD) to match two sets of lines with the following additional characteristics:

1. Points on one line must match to points on another line.
2. Line orientation difference is taken into consideration to compute the dissimilarity between 2 lines.

Similar to MHD, LHD is defined as

$$h(M, N) = \frac{1}{\|M\|} \sum_{L_i \in M} \|L_i\| \cdot \min_{L_j \in N} d(L_i, L_j) \quad (6.1)$$

M and N denote the sets of line segments from the model and test images. $\|M\|$ represents the sum of line segment lengths in M . L_i denotes the i^{th} line segment and $\|L_i\|$ represents the length of L_i . $d(L_i, L_j)$ is the vector that represents the transformation distance between two lines L_i and L_j . The ratio $\|L_i\|/\|M\|$ allows each line to contribute according to its length. The vector $d(L_i, L_j)$ is defined as

$$d(L_i, L_j) = [d_\theta(L_i, L_j) \quad d_\parallel(L_i, L_j) \quad d_\perp(L_i, L_j)]^T \quad (6.2)$$

where $d_\theta(L_i, L_j)$, $d_\parallel(L_i, L_j)$ and $d_\perp(L_i, L_j)$ are angle, parallel and perpendicular distances (see Figure 6.1(a)). These distances are defined as

$$d_\theta(L_i, L_j) = f(\theta(L_i, L_j)) \quad (6.3)$$

Figure 6.1: (a) line displacement measures. $d_{\parallel} = \min(l_{\parallel 1}, l_{\parallel 2})$, $d_{\perp} = l_{\perp}$;

(b) illustration of the angle distance.

$$d_{\parallel}(L_i, L_j) = \min(l_{\parallel 1}, l_{\parallel 2}) \quad (6.4)$$

$$d_{\perp}(L_i, L_j) = l_{\perp} \quad (6.5)$$

$\theta(L_i, L_j)$ is the smallest intersecting angle between L_i and L_j . $f()$ is a non-linear monotonic penalty function to map an angle to a penalty value. A weight W which was determined by a training process was needed for $f()$ to normalize with respect to $d_{\parallel}$ and $d_{\perp}$. The design of the parallel and perpendicular displacements was illustrated with a simplified example of two parallel lines, L_i and L_j , as shown in Figure 6.1(a). $d_{\parallel}(L_i, L_j)$ was defined as the minimum displacement to align either the left end points or the right end points of the lines. $d_{\perp}(L_i, L_j)$ is $l_{\perp}$ which is the perpendicular distance between the two parallel lines. In general, L_i and L_j would not be parallel but one can rotate the shorter line with its midpoint as rotation center to make L_i parallel to L_j . d_{θ} was later modified in [173] as

$$d_{\theta}'(L_i, L_j) = \min(\|L_i\|, \|L_j\|) \cdot \sin(\theta(L_i, L_j)) \quad (6.6)$$

6.1 Line segment Hausdorff distance

96

to remove the weight W . The term “ $\min(\|L_i\|, \|L_j\|) \cdot \sin(\theta(L_i, L_j))$ ” transforms the angular difference into a Euclidean distance as shown in Figure 6.1(b).

HD, MHD and LHD can be related using the simplified example in Figure 6.2 with 2 line segments. In Figure 6.2(a), the lines are represented as 2 sets of points. HD measures the longest distance while MHD measures the average distance. In Figure 6.2(b), the measures of $l_\perp$ and l_θ from [173] are shown. We use l_θ instead of $d_\theta'(L_i, L_j)$ for simplicity. Then, we have $MHD = l_\perp$, $HD = l_\perp + \frac{1}{2}l_\theta$.

Figure 6.2: Illustration of angle and perpendicular distances, (a) HD and MHD; (b) $l_\perp$ and $l_\theta = d_\theta'(L_i, L_j)$.

Figure 6.3 shows the rough relationship between HD, MHD and LHD when calculating the distance between two lines. From Figure 6.3(a), it is obvious that the three distances have similar variation trends with respect to $l_\perp$. In Figure 6.3(b), the curve of LHD gives the sharpest slope among the three, which implies that its angular dissimilarity has a larger weight when calculating the distance between two line segments than HD and MHD. This also indicates the fact that additional orientation dissimilarity is included in LHD.

Figure 6.3: Assuming $\|d_{\parallel}\| = 0$, (a) variation of d for fixed l_{θ} ; (b) variation of d for fixed $l_{\perp}$.

6.2 Matcher selection

After the indexing and normalization from chapter 5, a model LEM would have been aligned with an input LEM. The proposed CsHD is then applied to match the input with the model. Prior to the match can be carried out, a distance transform process is needed to compute the nearest model curves to each input curve. Distance transform (DT) assigns to a background pixel the approximated distance to the nearest object pixel. It is an efficient technique to find the nearest object. We make use of the Chamfer metric DT algorithm [117] and apply a 3×3 neighborhood mask (Figure 2.2) on an input LEM. One example of the application result, i.e. a distance map, on an input curve is shown in Figure 6.4(a) where the curve positions are labeled with the distance values of zero. The distance from any location to the curve can be approximated by dividing the value at each location by 3. In general, there will be more than one curve from an input LEM, and hence the dis-

Figure 6.4: (a) Chamfer distance map from C_{input} ; (b) overlay model curve C_{model} on the distance map.

tance transform of a curve overlaps with the other. This can be handled by recording all the distance transform information at each location with a linked list as shown in Figure 6.4(b) where the linked list of one particular position is illustrated. Each node of the list records the identities of different nearest curve, nearest line and the distance from such line. Unfortunately, the linked list will take up a lot of memory when the input LEM possesses many curves. We tackle this problem by limiting the distance (or layer number, N_{layer}) that a curve can influence/propagate. A layer can roughly be defined as the points whose loci are equidistant from the given curve. The layer number, N_{layer} , will be determined experimentally. With the distance map of an input LEM, a model curve can be conceptually overlaid onto the map to search for nearest curves. One example of a model curve, C_{model} , is shown in Figure 6.4(b). The distance from C_{model} to C_{input} can be computed as $\frac{1}{3} \sqrt{\frac{1}{n} \sum_{k=1}^n v_k^2}$, where v_k s are the distance values to C_{input} at the overlaid positions.

6.3 Curve segment Hausdorff distance (CsHD)

The proposed CsHD can be seen as a hybrid of template and syntactic matching approaches. The curve segment collection (LEM) of an image object is treated as a template which captures the inherent structural relationships of curves. At the same time, a curve is a consecutive run of connected line segments, and can be described/matched using string attribute grammar [33]. The relationships between curve segments can be modeled using graph. In the following, we simplify the modeling using attribute grammar.

6.3.1 Attribute grammar description

From the viewpoint of syntactic matching, two LEMs can be seen as strings of symbols [174]. Let I be an image, C be a curve and L be a line with attributes (l, θ, x, y) , where l , θ , x and y are the length, orientation and midpoint location of the line. The grammar description of I will be

$$\begin{aligned}
 I &\rightarrow C I \\
 I &\rightarrow C \\
 C &\rightarrow L C \\
 C &\rightarrow L \\
 C &\rightarrow \Lambda
 \end{aligned} \tag{6.7}$$

6.3 Curve segment Hausdorff distance (CsHD)

100

where C , L and Λ represent the curve, line and empty string respectively. The basic edit operations defined between two strings include

$$\begin{aligned} \text{change} : & \langle L_i \rangle \rightarrow \langle L_j \rangle \\ \text{deletion} : & \langle L_i \rangle \rightarrow \Lambda \\ \text{insertion} : & \Lambda \rightarrow \langle L_j \rangle \end{aligned} \quad (6.8)$$

This description forms the syntactic foundation for CsHD. With the emphasis on structural description of patterns, the syntactic approach attempts to draw an analogy between the structure of patterns and the syntax of a language. The analogy is attractive primarily because of the availability of mathematical linguistics as its theoretical foundation. In our work, we use $d()$ to represent the cost function of an edit operation (edit distance), e.g. $d(X, Y)$ for changing X to Y . In the following discussion, we use M and N to represent the collection of curves from model and input images respectively. Let C_i^M and C_j^N denote the i^{th} and j^{th} curves from M and N . Each C is the collection of one consecutive run of lines. We use $L_{i,p}^M$ and $L_{j,q}^N$ to denote the p^{th} and q^{th} lines from C_i^M and C_j^N . Furthermore, let $\|C_i^M\|$, $\|L_{i,p}^M\|$, and $\|M\|$ represent the lengths of C_i^M , $L_{i,p}^M$, and all the curves in M .

6.3.2 CsHD computation

Similar to MHD and LHD, CsHD is defined as $\max(cshd(M, N), cshd(N, M))$ with $cshd(M, N)$ being

$$cshd(M, N) = \frac{1}{R_M} \cdot (d(\Lambda, L^N) + d(L^M, \Lambda) + d(L^M, L^N)) \quad (6.9)$$

6.3 Curve segment Hausdorff distance (CsHD)

101

where $d(\Lambda, L^N)$, $d(L^M, \Lambda)$ and $d(L^M, L^N)$ represent sums of all the necessary edit distances of insertions, deletions and changes. R_M is used for scale normalization. It denotes half of the diagonal of the smallest rectangle that encloses M .

Computing $d(\Lambda, L^N)$

$d(\Lambda, L^N)$ is defined as $\sum_p \|L_p^N\|$. Each $\|L_p^N\|$ denotes the partial length of a line from test image that cannot find its counterpart from model image for matching in its neighborhood. Each line in N needs to keep track of the length portion that has been matched.

Computing $d(L^M, \Lambda)$

$d(L^M, \Lambda)$ is defined similarly as $\sum_p \|L_p^M\|$.

Computing $d(L^M, L^N)$

$d(L^M, L^N)$ is defined as $d_{\perp}(M, N) + d_{\theta}(M, N)$. Similar to LHD, they denote the perpendicular and angular distances of the matched curves from M to N . $d_{\perp}(M, N)$ is computed as the weighted sum of the perpendicular distance of each curve from M to its counterparts in the test image as

$$d_{\perp}(M, N) = \frac{1}{\|M\|} \cdot \sum_i \|C_i^M\| \cdot d_{\perp}(C_i^M, N) \quad (6.10)$$

where $\|C_i^M\|/\|M\|$ is the weight and $d_{\perp}(C_i^M, N)$ is the perpendicular distance from C_i^M to its counterparts in N . Since the counter input curve in N could be broken

6.3 Curve segment Hausdorff distance (CsHD)

102

into a few pieces, $d_{\perp}(C_i^M, N)$ is defined as the weighted sum of lines in C_i^M as

$$d_{\perp}(C_i^M, N) = \frac{1}{\|C_i^M\|} \cdot \sum_q \|L_{i,q}^M\| \cdot \left\{ \sum_j d_{\perp}(L_{i,q}^M, C_j^N) \right\} \quad (6.11)$$

In this formulation, each C_j^N has been ranked from the nearest to the farthest with respect to C_i^M for computing $d_{\perp}(L_{i,q}^M, C_j^N)$. Each $L_{i,q}^M$ will first find its match from the nearest C_j^N . If the match cannot be found or the match is shorter than $\|L_{i,q}^M\|$, the next nearest C_j^N will be searched until the total length ($\|L_{i,q}^M\|$) of the matched counter parts equals $\|L_{i,q}^M\|$. In case $\|L_{i,q}^M\|$ is shorter than $\|L_{i,q}^M\|$, the difference becomes $\|L_p^M\|$ in computing $d(L^M, \Lambda)$. Basically, the formulation is a LHD based line matching from $L_{i,q}^M$ to N . The curve-to-curve matching scheme is loosely enforced by ranking C_j^N from the nearest to the farthest. This enforcement follows the principle of non-explicit feature correspondence to simplify and speed up the computation. Each $d_{\perp}(L_{i,q}^M, C_j^N)$ is defined as

$$d_{\perp}(L_{i,q}^M, C_j^N) = \frac{1}{\|L_{i,q}^M\|} \cdot \sum_s \|L_{j,s}^N\| \cdot d_{\perp}(L_{i,q}^M, L_{j,s}^N) \quad (6.12)$$

In this formulation, $d_{\perp}(L_{i,q}^M, L_{j,s}^N)$ is the basic LHD computation for perpendicular distance as described in section 6.1. $L_{j,s}^N$ represents the collection of lines in C_j^N and have been ranked from the nearest to the farthest from $L_{i,q}^M$. $\|L_{j,s}^N\|$ represents portion of length from $L_{j,s}^N$ that has been matched to $L_{i,q}^M$.

Computing $d_{\theta}(M, N)$

Similarly, $d_{\theta}(M, N)$ is defined as

$$d_{\theta}(M, N) = \sum_i d_{\theta}(C_i^M, N) \quad (6.13)$$

$$d_{\theta}(C_i^M, N) = \sum_q \sum_j d_{\theta}(L_{i,q}^M, C_j^N) \quad (6.14)$$

$$d_{\theta}(L_{i,q}^M, C_j^N) = \sum_s d_{\theta}(L_{i,q}^M, L_{j,s}^N) \quad (6.15)$$

Note that the weighting factors have been removed because the basic distance element $d_{\theta}(L_{i,q}^M, L_{j,s}^N)$ in formula 6.6 has included the information of the line's length. In addition, we distinguish positive or negative value for each $d_{\theta}(L_{i,q}^M, L_{j,s}^N)$ depending on their orientation difference with respect to $L_{i,q}^M$. For example, in Figure 6.5(a), all the angle distances $d_{\theta}(L_{i,q}^M, L_{j,s}^N)$ from lines on C_j^N with respect to $L_{i,q}^M$ are positive. In Figure 6.5(b), the $d_{\theta}(L_{i,q}^M, L_{j,s}^N)$ from the four lines on C_j^N are positive, negative, positive and negative. Finally, the absolute value of their summation is acquired as angle distance. In this way, the $d_{\theta}(L_{i,q}^M, C_j^N)$ value computed from Figure 6.5(b) is zero, and this is consistent with human perception.

Figure 6.5: Illustration of angular distances between lines and curves.

6.3.3 New formulation of $d_{\perp}(L_i, L_j)$

The measure $d_{\perp}(L_i, L_j)$ from [162] works well as long as the lines do not intersect. Two intersecting lines can give rise to inconsistent $d_{\perp}(L_i, L_j)$ depending on whether one or both lines get split due to segmentation inconsistency. One example is illustrated in Figure 6.6 where the input line has a gap in (b). Similar perpendicular distances are expected from both cases. However, based on the original definition of $d_{\perp}(L_i, L_j)$ (formula 6.5), we get $d_{\perp}(L_i, L_j) = 0$ for (a) and $d_{\perp} = \frac{1}{2} l_{\perp 1}$ ($d_{\perp} = \frac{1}{2} l_{\perp 2}$) for the lower (upper) input line segment in (b), assuming that the distances from the other end points to the model line are zeros. This inconsistency can be resolved if we detect each intersecting line and cut it into two at the intersecting point. Instead of finding the intersecting point, $d_{\perp}$ is modified in this study as

$$d_{\perp}' = \frac{1}{2} \cdot (w_1 \cdot l_{\perp 1} + w_2 \cdot l_{\perp 2}) \quad (6.16)$$

where w_1 and w_2 are defined as

$$w_i = \begin{cases} \frac{l_{\perp i}}{l_{\perp 1} + l_{\perp 2}} & \text{if two lines intersect} \\ 1 & \text{otherwise} \end{cases} \quad (i = 1, 2)$$

Figure 6.6: Illustration of the new design: $d_{\perp}'$. To simplify the illustration, two lines are assumed to intersect and get split at the midpoint.

6.3 Curve segment Hausdorff distance (CsHD)

105

Formula 6.16 reduces to formula 6.5 for non-intersecting lines, and $d_{\perp}'$ remains unchanged for both Figure 6.6(a) and (b), which is also consistent with the design of MHD. Hence, the implementation of CsHD will employ this new formulation.

6.3.4 Complexity analysis

The computational complexities of MHD, LHD and CsHD are of the order of $O(m_p + t_p)$, $O(m_l + t_l)$ and $O(m_c t_c)$ respectively. m_p and t_p are the numbers of pixels in model and test images, while m_l and t_l are the numbers of lines in model and test LEMs. MHD and LHD need to compute first the distance transform and then the overlaying operation of a model onto the distance map. For CsHD, m_c is the number of curves in model image and t_c is the number of m_c 's corresponding curves from Chamfer distance map. It is more time consuming since it needs to do book keeping of which line has been matched and how much has been matched. On the other hand, we have $m_p \gg m_c$ and $t_p \gg t_c$. Furthermore, the book keeping operation is equivalent to the MHD to work on the lowest entity, i.e. points. The real timings of MHD, LHD and CsHD are expected to be similar.

Chapter 7

Experimental Results

A series of experiments have been conducted using the logo database from the work of Jain and Vailaya [2]. We employed 910 distinct logo images in our experiment and generated 373 test logos from 80 of the 910 models. The 80 logos were selected randomly and the 373 test logos were grouped into 7 sets. Some examples of the inputs are given in Figure 7.1. All the test images can be found in appendix B. The first set of test logos was produced by rotating and scaling the original images randomly. The second and third sets were generated by adding Gaussian noise with standard deviations of 25 and 45 respectively. The noise has deteriorated the segmentation results of the test images to a great extent. The fourth set of test logos was added with 2 white stripes in random directions that changed the topology property. The

Test types	Number of test logos	Model logos	Test examples
Set 1 Rotated and scaled logos	20
Set 2 Gaussian noise corrupted logos (standard deviation = 25)	55
Set 3 Gaussian noise corrupted logos (standard deviation = 45)	55
Set 4 Strip corrupted logos	51
Set 5 Occluded logos	55
Set 6 Skewed logos	72
Set 7 Logos with mixed noise	65

Figure 7.1: Model images and test images in our experiment.

strip width was set to 7 pixels. The fifth set of test images was obtained by cutting the original images. This simulates occlusion that occurs frequently on items in our surroundings. Ninety percent of the cuttings cut out more than 1/4 of the image size while 49% of the cuttings cut out more than half of the image size. In the sixth set, all the test logos were skewed in horizontal direction with 10 and 20 degrees. This simulates observation of a 2D pattern from different angles in 3D space. The seventh set of test logos has mixed distortions (combination of the above distortions).

7.1 Experimental results of indexing method

7.1.1 Parameter selection for indexing method

In Chapter 5, there are several parameters whose values have not been set, or need further illustration, including the number of reference features, θ_{sep} and tolerance angle. To choose them properly, we design the following experiment. In this experiment, we use 100 test images which are made of 4 groups. They are randomly chosen from Set 2, Set 4, Set 5 and Set 7 as specified in Figure 7.1 (25 images from each group). The corresponding indexing results are shown with respect to different parameter values.

The number of reference features

As mentioned in section 5.2, rich structure descriptor reduces the ambiguity in matching. However, the richer information the structure contains, the easier it is affected by distortions, such as occlusion and noise. Our indexing algorithm is based

Figure 7.2: Alignment results with respect to the numbers of required neighbors.

on the n -nearest reference features of one feature point. We set n to 3, 6, 9 and 12. The corresponding indexing results are shown in Figure 7.2 respectively. It is observed that more images could be aligned correctly when n was set to 6. Hence, in the following experiment, we will adopt $n = 6$. If the number of extracted features is less than 6, all the features will be made use of in the indexing process.

The value of θ_{Sep}

In section 5.3, the granularity of the 3-D cube is decided by the value of θ_{Sep} . To determine this parameter in our system, we have done this experiment to test the choices of θ_{Sep} . We set θ_{Sep} to 15 degrees, 30 degrees and 45 degrees respectively. The corresponding indexing results are shown in Figure 7.3. This result is easy to explain. A relatively small θ_{Sep} (15 degrees) can increase the distinguishing ability of the shape descriptor. As the number of corresponding feature pairs decreases, the indexing becomes faster. However, it cannot tolerant distortions well. An even smaller with less than 15 degrees will likely result in a different shape descriptor from the original

Figure 7.3: Alignment results with respect to different values of θ_{Sep} .

one and the indexing may fail. On the other hand, a large θ_{Sep} (45 degrees) makes a loose criterion for corresponding feature searching, which will greatly increase the number of corresponding feature pairs and candidate model number for each input image. Consequently, the indexing process is slow since the indexing stage cannot filter out enough candidate models and indexing becomes meaningless (see Table 7.1). Therefore, in our work, θ_{Sep} is set as 30 degrees, which separates the neighboring area of a feature point into 12 bins radically.

θ_{Sep}	Running time of indexing (s)
15 degrees	51.53
30 degrees	177.95
45 degrees	486.75

Table 7.1: The running time of indexing system with respect to different values of θ_{Sep} .

Figure 7.4: Alignment results with respect to different values of tolerance angle.

The value of tolerance angle

We did this experiment by changing tolerance angle from 5 degrees to 10 until 15 degrees with totally overlapped separation regions. The indexing results are given in Figure 7.4. The running time of indexing using these three tolerance angles are also recorded in Table 7.2.

A small tolerance angle equips the separation region with a flexible margin. As

tolerance angle	Running time of indexing (s)
5 degrees	177.95
10 degrees	331.54
15 degrees	552.29

Table 7.2: The running time of indexing system with respect to different values of tolerance angle.

it grows larger, the overlapped area increases and the same descriptor exists in two separation regions more frequently, which will cause redundant data stored in the indexing structure. From the above result, the tolerant angle of 5 degrees is the best.

7.1.2 Indexing system performance

Since the indexing is the first part of the system, its performance is tested from two aspects: first, whether the alignment problem can be solved as a byproduct of the indexing process; second, how many models can be filtered out from the database to save running time of the matching process.

The first point has been tested in the experiments designed in last section to some extent. In this section, we will test it more systematically. All the 7 sets of test images were fed into the indexing module. The indexing process is expected to align the test images with respect to model images correctly. Figure 7.5 records the findings from our experiment. As shown in this figure, the process is able to find and suggest all the corresponding feature pairs as output for set 1. Inevitably, the output can include some incorrect pairings. Such incorrect pairs will be tackled by the next processing CsHD. As for the other sets, the process cannot find and suggest all the corresponding feature pairs as output. Hence, there are some inputs that cannot be aligned correctly with their corresponding models, and this will lead to definite failure of matching. We found that set 3 performed the worst since the added Gaussian noise (standard deviation=45) brought many outlying points that had become the unwanted feature points. They interfered with the local structure construction process (relying on

Figure 7.5: The result of alignment.

the neighboring information of a feature point) greatly. Hence, the correspondence problem between local structures cannot be solved properly. This caused the failure of alignment. For test set 6 (with skewing), we have around 15% failures of alignment since the local structure construction for indexing depends much on angle information, which has been affected by skewing. The failures point to the need in future to design a more robust alignment method.

Based on the selected images from previous step, we tested the indexing method’s effectiveness in pruning the model space with respect to a test image. After the indexing process, a list recording the selected candidate models is generated for each test image. It includes all the models that the test image needs to match. We found that the candidate lists are much shorter than the full list as in Figure 7.6 and Table 7.3.

We also explored the timing of the indexing scheme. Figure 7.7 shows the running time in retrieving the candidate list for a test image with respect to different

7.1 Experimental results of indexing method

Figure 7.6: The filtering result on the test images.

	Min	Max	Average
Set 1	97.03%	61.75%	77.40%
Set 2	99.89%	49.89%	81.24%
Set 3	100.00%	69.67%	96.70%
Set 4	99.01%	64.39%	81.10%
Set 5	96.20%	50.00%	77.58%
Set 6	97.47%	50.11%	82.51%
Set 7	99.45%	53.62%	79.17%

Table 7.3: The max, min and average percentages of models that get filtered out.

input complexities and sizes of the database. Four test images whose feature point numbers equal to 10, 31, 50 and 75 respectively are used. It is observed that, as a preliminary process, our indexing method is relatively efficient at working with middle-sized database (with 100 to 500 models). Future work can focus on improving its efficiency with large database.

Figure 7.7: The running time of indexing.

7.2 Experimental results of CsHD

We first tested the performance of CsHD with different numbers (N_{layer}) of layers in generating the Chamfer distance map. Small N_{layer} cannot capture enough information for neighborhood search while large N_{layer} will increase the running time. Furthermore, a large N_{layer} will make matching more rigid for CsHD that enforces curve-to-curve match and cannot easily turn down a bad match of outlier. On the other hand, LHD is not affected by choosing a large N_{layer} . We tested the performance of CsHD with different N_{layer} values of 5, 8, 11 and 14. Empirically, N_{layer} was found to perform the best on the input test sets when it was set to 8 (Figure 7.8). Hence, we adopt $N_{layer} = 8$ in the rest of the experiment.

Figure 7.8: Recognition rates of CsHD according to different N_{layer} .

7.2.1 Experiment 1: Effective shape recognition and system characteristics

We computed the CsHD (formula (6.9)) on each test set based on the lowest (best) 20%, 40%, 60%, 80% or 100% curve-to-curve distances. The recognition rates of CsHD on the 7 input sets are recorded in Figure 7.9.

A general trend of better results when more curves are included is observed since the most distinctive differences usually reside in the last 20% curves that distinguish similar but actually different logos. However, this trend is not observed on input set 3. For slightly corrupted input, it is not difficult to find near-perfect/good-enough matching curves from the corresponding model. Hence, the matching process converges to the correct model easily when more curves are used or when more shaping conditions are satisfied. For input set 3, the outlying points from Gaussian noise produce many short curves (e.g. even more than 80% of all the curves) evenly distributed all over the input image. These short curves contribute little to image-image distance due to their small lengths. Therefore, the lowest (best) 20% curve-curve distances which always come from the dominant curves play the major role for final distance computation and a flat top is observed for set 3. From Figure 7.9, there is another interesting observation: the recognition rates using the top 20% curves are quite high, not too much different from those using the top 40%, 60%, or 80% curves. This highlights the great potential to perform occluded-object recognition using CsHD.

Figure 7.9: Recognition rates of CsHD, (a) the lowest dissimilarity is accepted as the match; (b) the first 3 lowest dissimilarities are accepted as the matches.

7.2.2 Experiment 2: Comparison of MHD, LHD and CsHD

We have implemented the algorithms of MHD and LHD (accommodating the new $d(L_i, L_j)$) to compare with CsHD in Figure 7.10. In set 3, the added Gaussian

7.2 Experimental results of CsHD

Figure 7.10: The comparison between MHD, LHD and CsHD, (a) the lowest dissimilarity is accepted as the match; (b) the first 3 lowest dissimilarities are accepted as the matches.

noise has created a lot of outlying points and badly corrupted LEM. One typical example is shown in Figure 7.11. They lower the distinctiveness of each input greatly and MHD gives a zero recognition rate. In all the situations (especially for occluded images), CsHD performs better from our observation.

This experimental result from set 3 also indicates that the extraction of consistent features is important in matching process. CsHD is built on top of LHD while LHD is based on MHD. Hence, the performance of CsHD will be similar to MHD

Figure 7.11: An example of LEM with Gaussian noise: (a) original image; (b) line representation.

conceptually when it needs to process point clouds. For point clouds, we cannot have line-point or curve-point association. Therefore, the process is less certain of the correctness of the matching result. From another point of view, to identify a cloud of points is also a challenging work for human beings.

7.2.3 Experiment 3: Recognition of images seen from arbitrary viewing angle

Many man-made objects can be modeled as 2D patterns to be viewed in 3D from different viewing (skewing) angles. Hence, we investigate the effect of image skewing in a more systematic manner here. A new set of 144 test images were generated from 18 logos with different degrees of skewing. Sample inputs are given in Figure 7.12. The number of model is 500. Because we focus on testing the performance of the matching theme here, we employ another registration algorithm (applying reference line segment) [173] (Figure 7.13) to give better alignment in this and the next experi-

Figure 7.12: Some examples of test images: (a)-(e) are skewed horizontally with 5, 15, 25, 35 and 45 degrees respectively; (f)-(j) are skewed vertically with 5, 15, 25, 35 and 45 degrees respectively.

ment. This registration, however, has no indexing capability. The recognition results are shown in Figure 7.14(a). All of the 5 degrees skewed test images can be recognized successfully. Once the skewing increases further, the recognition rate drops dramatically. Only about 20% of the 45 degrees skewed images can be matched. To enhance the performance of CsHD, we add more models (i.e. the original skewed by 20 and 40 degrees) to the database. The new results are shown in Figure 7.14(b) with dramatic improvement.

7.2.4 Experiment 4: Recognition of occluded images

It is very common that captured input images are occluded, which are also expected to be recognized. We experiment on test images with cutting (to simulate

Figure 7.13: Normalization, (a) one example of a test image; (b)-(d) its normalized line segment drawings according to choices of different reference line segments.

Figure 7.14: Recognition rates of CsHD for skewed images, (a) using the original model database; (b) adding new models skewed by 20 and 40 degrees.

occlusion) to investigate our method’s performance on this problem. Two hundred and sixteen test images are generated from 54 models with 20% to 80% cutting. Some

Figure 7.15: Some examples of test images, (a) images with 20% cut, (b) images with 40% cut, (c) images with 60% cut, (a) images with 80% cut.

examples are given in Figure 7.15.

Figure 7.16 gives the recognition results of CsHD with respect to different percentages of cutting. In Figure 7.16, for the images with 20% and 40% cutting, the recognition rate of CsHD can reach as high as 96%. When the cutting increases to 60% the recognition rate drops to 80%. Finally, the recognition rate is less than 20% when 80% content is cut. It is even hard for human beings to recognize this set of test images since most useful shape information has been lost due to cutting.

7.2.5 Experiment 5: Palmprint matching

Similar experiments have been performed on palmprint recognition. The palmprint images were obtained from the Biometric Research Centre, Hong Kong Polytech-

Figure 7.16: Recognition rates of CsHD for occluded images. Top 1, 3 and 5 mean that the first, first 3 and first 5 lowest dissimilarities are accepted as the matches respectively.

nic University [175]¹. Some examples are given in Figure 7.17. A special palmprint capture device was used to collect data from 100 individuals. 500 palmprint test images were generated from the 100 models and grouped into 5 sets (100 images per set). 10 examples for each set are shown in Figure 7.18. Set 1 and 2 were generated by adding Gaussian noise with standard deviations of 25 and 45 respectively. The third set was added with 2 white strips. The forth set was obtained by cutting the original images. In the fifth set, all the test images were in horizontal direction with 20 degrees. Similar recognition results (as compared to the logos) have been obtained in Figure 7.19. An important observation is LHD performs much better than CsHD with set 2 input images. Because CsHD enforces curve-to-curve matching, it cannot adapt as well as LHD to the situation of badly corrupted LEM. A systematic experiment to test the system performance under occlusion and skewing has been concluded

¹<http://www4.comp.polyu.edu.hk/biometrics/>

Figure 7.17: Some examples of palmeprints.

in Figure 7.20. The identification rate of CsHD is lower than that of the logo images on set 2 due to the acquisition noise present in the original palmprint images.

7.2.6 Experiment 6: Real object recognition

Besides logos and palmprints, we also tested CsHD's performance in real object recognition [176]. The test images have 4 sets (30 images per set): occluded images, images with added lines and skewed images with 10 or 20 degrees. Some examples are given in Figure 7.21. All the test images can be found in appendix C. The database used in this experiment has 280 models of 7 objects. The matching result is shown in Figure 7.22. 80% of the test images can be recognized correctly from all the test images.

7.2.7 Experiment 7: Mixed-object recognition

To test the potential of recognition in cluttered environment, 15 images composed from multiple objects were used as input to find the most similar models from the database. It is found that the proposed system can handle this kind of inputs from such mixed/cluttered environment. Two examples are displayed in Figure 7.23.

Figure 7.18: Some examples of test images.

Figure 7.19: Comparison between LHD and CsHD on palmprint matching. The lowest dissimilarity is accepted as the match.

Figure 7.20: Recognition rates of CsHD for (a) occluded images and (b) skewed images.

All the other test images and their matching results can be found in appendix D. The number of model images is 910. A failure case is given in Figure 7.24 for analysis, where part (a) is the input image while part (b) is the expected returned image. Unfortunately, the expected model has been filtered out in the indexing step because its feature distribution has changed a lot due to the problem of scale change as shown in Figure 7.24(c) and (d). As discussed in chapter 4.4, the feature extraction and refinement method is not good at processing small pixel-level shape structures, which

Figure 7.21: Some examples of test images in real object recognition. The first row shows the model images while the test images are displayed in the second row. (a) occluded images; (b) images with added lines; (c) skewed images (20 degrees).

Figure 7.22: Recognition rates of CsHD for real object recognition.

needs improvement in future. The recall rate versus the number of retrieved models is plotted in Figure 7.25. Figure 7.25 shows that at least one correct model can be found within the top 5 returned images. However, each input composes of a few models and not all the corresponding models can be retrieved. About 40% of inputs

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Figure 7.23: Example results of cluttered object recognition.

failed to retrieve all the models even 40 images are retrieved and returned.

7.2.8 Performance discussion

We record the computation time of LHD and CsHD on matching one test image to 500 model images in Figure 7.26. Their performances are similar. In our experiment, all the images are 500×500 pixels in size which makes MHD much slower than LHD and CsHD because of its extremely large number of points to be processed. The complexity analysis of the three methods can be found in chapter 6.3.4. The

Figure 7.24: An example of failure in mixed-object recognition, (a) input image, (b) expected model, (c) feature points distribution from the investigated part of input, (d) feature points distribution from the original scale corresponding to (c).

Figure 7.25: The recall rates versus different numbers of returned images.

experiments were conducted on a DELL PC with 1.6GHz CPU and 512MB RAM.

In Figure 7.27, some examples which cannot be matched correctly due to index-

Figure 7.26: The real computation time of LHD and CsHD on matching one test image to 500 models.

ing and alignment failure are given. It can be seen that the feature points distribution of the images are distorted and no consistent feature structures can be extracted from both input images and their corresponding models, which causes the failure of alignment and matching.

The figures from Figure 7.28 to Figure 7.31 show some examples of logos that cannot be recognized successfully in the matching process even though the alignment has been done correctly. The first row of each of these figures shows the input images and the expected matches. The second row gives the top 5 matches returned from system with their dissimilarity distances while the third row displays the overlapping between input (solid lines) and the top match (dotted lines).

Figure 7.28 is obtained from set 4 (with added line segments). The topology property of input has been changed by the two added line segments, which break the

Figure 7.27: Some examples of failures due to incorrect alignment. The first column is the distorted input images and their LEMs while the second column is their corresponding models and LEMs.

Figure 7.28: An example of failed match from set 4.

original curves into several pieces and generate some additional curves. The returned top 1 match has similar shape structure with the input (as shown in Figure 7.28(c)) and the mismatch between them is not much. The correct match can still be ranked high position at the second place without much difference of the dissimilarity distance. The images in Figure 7.29 are from set 5 (with occlusion) with part of its distinctive structural information being cut. Although the correct match (the top 2 match) seems more similar to the input than the top match from human perception, it only appears as the second rank since the occluded part creates a large value of $d(L^M, \Lambda)$ (formula

Input	LEM from input	Expected match

(a) Input image and expected model

Top 1	Top 2	Top 3	Top 4	Top 5

distance = 4.88	distance = 4.94	distance = 6.84	distance = 6.99	distance = 7.04

(b) Returned matches and their dissimilarity merit with respect to input

(c) Overlapping input with the top match

Figure 7.29: An example of failed match from set 5.

6.9) and causes a large total distance. If more information about the input is available beforehand, i.e. the input is with occlusion, we can decide whether $d(L^M, \Lambda)$ should be included in the matching process. As for this example, the expected match will be returned as the first top match if $d(L^M, \Lambda)$ is omitted. The failed match in Figure 7.30 is found in set 6 (with skewing). Because our alignment algorithm is not completely invariant to affine transformation, the skewed input image and its corresponding model do not guarantee to overlap perfectly even after correct alignment. Therefore,

Input	LEM from input	Expected match

(a) Input image and expected model

Top 1	Top 2	Top 3	Top 4	Top 5

distance = 7.01	distance = 7.51	distance = 7.85	distance = 8.00	distance = 8.26

(b) Returned matches and their dissimilarity merit with respect to input

(c) Overlapping input with the top match

Figure 7.30: An example of failed match from set 6.

the failure of matching happens and the expected match appears at the fifth rank. The top match possesses four small rectangles which correspond to the small areas residing in the input with similar appearance due to skewing. The images from Figure 7.31 are found from set 7 (with mixed noise). The added Gaussian noise has deteriorated the segmentation result and part of the shape structure has been cut. The input matches to a more similar model (the top match) and the expected match ranks as number four.

Additional explanation is given here for the appearance of the fifth top match

Input	LEM from input	Expected match

(a) Input image and expected model

Top 1	Top 2	Top 3	Top 4	Top 5

distance = 4.03	distance = 4.19	distance = 5.16	distance = 5.32	distance = 6.45

(b) Returned matches and their dissimilarity merit with respect to input

(c) Overlapping input with the top match

Figure 7.31: An example of failed match from set 7.

in Figure 7.28 and 7.31, which seems quite different from the input but ranks high. Figure 7.32 gives their overlapping with the inputs. As shown in Figure 7.32(a), the size of the model is much larger than the input relatively and the input matches to only a small portion of the model. The large scale of model creates large value of R_M (formula 6.9). Although the sum of $d(\Lambda, L^N)$, $d(L^M, \Lambda)$ and $d(L^M, L^N)$ is big enough to distinguish the model from the input, the total dissimilarity distance after normalization using R_M is low, which causes the high rank of incorrect model. Figure 7.32(b) can be explained similarly. Fortunately, this situation is not frequent

Figure 7.32: Additional explanation for two returned models in Figure 7.28 and 7.31.

in the experiment. In future, it is expected to be avoided by normalizing the three distances in formula 6.9 separately to generate the total distance.

We notice that some expected models from failure matchings can still be ranked high among the returned top 20 results. It implies that although the system cannot suggest the correct model as the top 1 match, it is still useful for interactive system when human can examine the top 20 returned images and choose the relevant results. Although curves perform better than points and lines for most types of input as matching primitives, it is still a relatively low level matcher. This limits the system's applicability and performance in simulating human perception, especially for the failure cases (although it is very effective in recognizing occluded and distorted images for most cases). To achieve better performance, higher level primitives with more contextual information can be attempted in future.

7.3 Summary

A series of experiments have been conducted to test the system performance using logos, palmprints and stationeries. Encouraging experimental results have been obtained especially on distorted test images. We also compared our method with another two HD approaches and found that CsHD has superior performance in most cases. On the other hand, the results also show the direction of our future work to overcome the system's disadvantage and improve its performance.

Chapter 8

Conclusion

8.1 Work summary

The central theme of this thesis is to propose and implement a novel shape recognition system, particularly in employing high level geometric features (lines and curves) extracted from shape contour, to investigate their functions and performance in different modules of a shape recognition system.

Chapter 4 proposes a refinement method based on an existing polygonal approximation algorithm – Dynamic two-strip algorithm (Dyn2S). We investigate the feature points extracted from Dyn2S and identified its weakness in locating feature point on round corner. This leads to our refinement algorithm which is inspired by modeling

and detecting a round corner first, and then applying a combination method to get consistent and non-redundant representation of a round corner. The refined feature points are fed to the following modules of the system as input.

In Chapter 5, we provide an indexing approach to prune the model space and generate a moderate number of likely models with respect to a test image. It employs a relatively simple but effective indexing structure to filter out unwanted models, narrow down the scope of match and create a highly efficient system. Moreover, the alignment problem can be solved simultaneously.

Chapter 6 addresses the matching process which suggests the most similar model in a database as a match. The input image is matched to the candidate models obtained from indexing using a novel matching algorithm, the curve segment Hausdorff distance (CsHD), in a curve-to-curve matching manner. It has better distinctive capability (especially for distorted images) than other HD methods. Moreover, this approach has the advantage to incorporate structural and spatial information to compute the dissimilarity between two sets of curve segments rather than two sets of points. This has been proven in our experimental results.

An in-depth study of the proposed technique has been carried out on logos, palmprints and stationeries. The test images come from six sources, i.e. scaled and rotated, Gaussian noise, strip corrupted, partially occluded, skewed and mix noise corrupted. Encouraging results have been observed.

8.2 Discussion and future work

From our experimental results, this study has fulfilled all the objectives below.

1. The proposed shape matching method generalized the HD approach with higher level primitives to improve performance and alleviate the false positive. The experiment in section 7.2 demonstrates consistently that the proposed CsHD (or LHD) had much better performance than the MHD approach. This is expected since CsHD (LHD) enforces curve-point (line-point) association that makes matching to a non-similar logo produce high dissimilarity. MHD is a popular approach for matching due to its good performance. Hence, CsHD (LHD) is even better and useful according to the comparison in section 7.2.2. CsHD performed worse than LHD only on data set 3 when the noise and distortion were too much to handle. This illustrates that higher level primitives can give better performance but the extent of noise and distortion can play a part in choosing the type of primitive.
2. Our local structure used to facilitate indexing can be extracted consistently from model and input images, even in noisy, distorted and occluded environments. The indexing process employs a novel fragmental neighboring matching technique, which supports the effectiveness of indexing structure in filtering unwanted models from database. This has been demonstrated in section 7.1. Simultaneously, the alignment problem can be solved properly as shown in Figure 7.5.

3. The proposed system performance has been tested in our experiments. Its shape recognition capability and partial matching characteristics have been demonstrated. Humans can identify and recognize an occluded object from a cluttered environment, provided that enough or distinguishable portion of the object can be observed. Our system has demonstrated the potential of such ability according to the experiment example shown in section 7.2.4 and 7.2.7.
4. CsHD is built on the line and arc primitives from a shape instead of the complete connected outline of a shape. The success of CsHD demonstrates that shape analysis can be built on realistic input data instead of making the requirement that the object has to be segmented perfectly. Furthermore, line and arc are deemed as suitable for shape recognition in different adverse environments.
5. CsHD has been applied in three different types of shape recognition (logos, palmprints and stationeries) with good performance. These demonstrate its potential to be used for object recognition in general.
6. The complexity analysis of CsHD in Figure 7.26 demonstrates that CsHD is an efficient approach with potential for real time application.

An in-depth, systematic study of the proposed system has been carried out to understand the characteristics of the system. The tolerance and ability to overcome noisy conditions have been investigated and tested. At the same time, the findings provide valuable directions and hints for system development in future work.

Optimization of feature refinement

The feature refinement algorithm in Chapter 4 improves the feature extraction process readily and provides a more consistent and consistent shape representation. However, it is still difficult to model different types of corners, i.e. sharp corners and round corners, regardless of scaling problem since different people may have different understandings. Applying various templates of angles of different size and an interactive mechanism might solve this problem.

Robust local structure construction

The extraction of consistent local features suffers a lot from the skewing and high deviation Gaussian noise. Besides angle direction, more properties (e.g. length ratio) can be included in the local structure which may bring the robustness under skewing. As for Gaussian noise, an image enhancement step might be needed according to image acquisition parameters.

Texture shape analysis

Some models from the original database were not used in our experiment because they include textures. Our system utilizes the line/curve segments of the Line Edge Map (LEM) from an image. LEM is based on the result of contour extraction (edge detection). “Contour analysis (edge detection) may be adequate for untextured images, but in a textured region it results in a meaningless tangled web of contours” [177]. A framework which can process regions with or without texture is expected.

General object recognition

In real applications, more general objects with high complexity will be captured as input to the system. To process this type of images, two improvements are suggested: (1) a preliminary shape representation method which can exclude unwanted information and retain useful information, (2) an accurate matching algorithm which can tackle multiple types of noises and distortions.

8.3 Contribution

Based on the proposed concept, new avenues for research such as general object matching, optimization of feature point selection have been suggested.

This thesis has made the following contributions:

- **Feature point detection.** A novel feature point refinement is proposed. It is based on the line segment representation of shapes from polygonal approximation. It aims to generate a more consistent feature point set at obtuse sections of a curve.
- **Indexing.** An adapted indexing structure is generated to store all the feature information of models to facilitate pre-filtering. The indexing process also prunes the search space of models to cut down CPU time.
- **Registration.** Since the input could have been captured in different scale and orientation, the proposed system aligns (or normalizes) an input with respect

to each model prior matching. The alignment is possible only when the same corresponding feature can be found from the input to the model. A feature incorporating the local structure is designed to facilitate the correspondence problem.

- **CsHD Matching.** A curve segment Hausdorff distance matching algorithm is proposed here. The proposed approach employs high level primitives (curve segment) which incorporate structural and spatial information to compute the dissimilarity between two sets of curves. The added information can conceptually provide more and better distinctive capability for recognition. It has a good performance under corruption such as broken lines, added noise and occlusion. The proposed technique produces better matching results compared to other Hausdorff distance matching methods of MHD and LHD.
- **System understanding.** An in-depth, systematic study of the proposed system has been carried out to understand the characteristics of the system. The tolerance and ability to overcome noisy conditions have been investigated and tested. The findings provide valuable directions and hints for system development in future work.

In summary, besides the general shape recognition system we developed, the other contributions can be concluded as (1) generalization of a line to line matching scheme to a curve to curve matching concept (Chapter 6) by treating a curve as a consecutive run of connected line, (2) adaptation (Chapter 5) of a general image

retrieval approach to identify likely candidates efficiently and support alignment of input shape with a model in even occluded and cluttered environment, and (3) derivation of an attributed grammar description for the proposed curve to curve matching. Some of the ideas or generalization might seem simple but the derivation process is novel and meaningful. Conceptually, the last (i.e. 3) concept has transformed a template approach into the syntactic domain. This new concept can help to generate even more flexible approach for the matching problem in situations when the object is distorted (i.e. skewed) or being viewed from a slightly different viewpoint in 3D space.

8.4 Thesis conclusion

This thesis has made a substantial contribution in the design of a general shape recognition system. Various problems have been investigated and solutions are proposed. A thorough demonstration has been made on the technique. Suggestions for future research are made based on the current work. The effect of the technique can be far-reaching with unforeseen potential.

Bibliography

- [1] R.C. Veltkamp, "Shape matching: similarity measures and algorithms," *Technical report*, Utrecht University, 2001.
- [2] A.K. Jain and A. Vailaya, "Shape based retrieval: a case study with trademark image database," *Pattern Recognition*, vol. 31, no. 9, pp. 1369-1391, 1998.
- [3] L.J. Latecki and R. Lakamper, "Shape similarity measure based on correspondence of visual parts," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1185-1190, 2000.
- [4] A.K. Jain, R.P. Duin and J. Mao, "Statistical pattern recognition: a review," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4-38, 2000.
- [5] D.P. Huttenlocher, G.A. Klandeman and W.J. Rucklidge, "Comparing images using the Hausdorff distance," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 9, pp. 850-863, 1993.
- [6] D.W. Paglieroni, G.E. Ford and E.M. Tsujimoto, "The position-orientation

- masking approach to parametric search for template matching," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, no. 7, pp. 740-747, 1994.
- [7] V. Bruce, E. Hanna, N. Dench, P. Healy and A.M. Burton, "The importance of mass in line drawings of faces," *Applied Cognitive Psychology*, vol. 6, pp. 619-628, 1992.
- [8] R. Singh and N.P. Papanikolopoulos, "Planar shape recognition by shape morphing," *Pattern Recognition*, vol. 33, no. 10, pp. 1683-1699, 2000.
- [9] R. Singh and N.P. Papanikolopoulos, "Geometric active deformable models in shape modeling," *IEEE Trans. Image Processing*, vol. 9, no. 2, pp. 302-308, 2000.
- [10] C.C. Lu and J.G. Dunham, "Shape matching using polygon approximation and dynamic alignment," *Pattern Recognition Letter*, vol. 14, no. 12, pp. 945-949, 1993.
- [11] L. Huang and M.J. Wang, "Efficient shape matching through model based shape recognition," *Pattern Recognition*, vol. 29, no. 2, pp. 201-215, 1995.
- [12] A. Loui, A. Venetsanopoulos and K. Smith, "Morphological autocorrelation transform: a new representation and classification scheme for two dimensional images," *IEEE Trans. Image Processing*, vol. 1, no. 3, pp. 337-354, 1992.
- [13] H. Liu and M.D. Srinath, "Partial shape classification using contour matching in distance transformation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 11, pp. 1072-1079, 1990.

- [14] C. A. Higgins and D.M. Ford, "On-line recognition of connected handwriting by segmentation and template matching," *International Conf. Pattern Recognition*, pp. 200-203, 1992.
- [15] R.M. Dufour, E.L. Miller and N.P. Galatsanos, "Template matching based object recognition with unknown geometric parameters," *IEEE Trans. Image Processing*, vol. 11, no. 12, pp. 1385-1396, 2002.
- [16] B.R. Meijer, "Rules and algorithms for the design of templates for template matching," *Proc. Eleventh IAPR International Conf. Computer Vision and Applications*, pp. 760-763, 1992.
- [17] L. Liu and S. Sclaroff, "Index trees for accelerating deformable template matching," *Pattern Recognition Letters*, vol. 23, no. 12, pp. 1483-1493, 2002.
- [18] F. Jurie and M. Dhome, "Hyperplane approximation for template matching," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 996-1000, 2002.
- [19] F. Ullah and S. Kaneko, "Using orientation codes for rotation-invariant template matching," *Pattern Recognition*, vol. 37, no. 2, pp. 201-209, 2004.
- [20] C.L. Li and K.C. Hui, "Feature recognition by template matching," *Computers and Graphics*, vol. 24, pp. 569-582, 2000.
- [21] M.S. Choi and W.Y. Kim, "A novel two stage template matching method for

- rotation and illumination invariance,” *Pattern Recognition*, vol. 35, no. 1, pp. 119-129, 2002.
- [22] C.F. Olson, “Maximum-likelihood image matching,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 853-857, 2000.
- [23] T. Kaneko and O. Hori, “Feature selection for reliable tracking using template matching,” *Conf. Computer Vision and Pattern Recognition*, pp. 796-802, 2003.
- [24] S. Karungaru, M. Fukumi and N. Akamatsu, “Face recognition using genetic algorithm based template matching,” *International Symposium on Communications and Information Technologies*, pp. 26-29, 2004.
- [25] H. Cho, D. Corina, J.F. Brinkley, G.A. Ojemann and L.G. Shapiro, “A new template matching method using variance estimation for spike sorting,” *Proc. Second International IEEE EMBS Conf. Neural Engineering Arlington*, pp. 16-19, 2005.
- [26] R. Bruneli and T. Poggio, “Face recognition: features versus templates,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 10, pp. 1042-1052, 1993.
- [27] H. Bunke and A. Sanfeliu, “String matching for structural pattern recognition,” *Syntactic and Structural Pattern Recognition: Theory and Applications*, World Scientific, 1987.

- [28] H. Bunke and A. Sanfeliu, "Matching tree structures," *Syntactic and Structural Pattern Recognition: Theory and Applications*, World Scientific, 1987.
- [29] F.H. Cheng, W.H. Hsu and M.C. Kuo, "Recognition of handprinted Chinese characters via stroke relaxation," *Pattern Recognition*, vol. 26, no. 4, pp. 579-593, 1993.
- [30] H.J. Lee and B. Chen, "Recognition of handwritten Chinese characters via short line segments," *Pattern Recognition*, vol. 25, no. 5, pp. 543-552, 1992.
- [31] H.J. Lee and D.J. Yu, "Line based structural matching via segment splitting," *Pattern Recognition Letters*, vol. 11, no. 3, pp. 181-189, 1990.
- [32] M. Al-Khaiyat and F. Kamangar, "Planar curve representation and matching," *Conf. British Machine Vision*, pp. 174-184, 1998.
- [33] R.C. Gonzalez and R.E. Woods, "Digital image processing," *Prentice Hall*, Upper Saddle River, New Jersey 07458, 2001.
- [34] S. Levialdi and L.G. Cinque, "Shape description by a syntactic pyramidal approach," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 10, pp. 573-585, 1996.
- [35] D.C.W. Pao, H.F. Li and R. Jayakumar, "Shapes recognition using the straight line Hough transform: theory and generalization," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, no. 11, pp. 1076-1089, 1992.

- [36] Z.N. Li, "Stereo correspondence based on line matching in Hough space using dynamic programming," *IEEE Trans. Systems, Man and Cybernetics*, vol. 24, no. 1, pp. 144-152, 1994.
- [37] X. Yi and O.I. Camps, "Line-based recognition using a multidimensional Hausdorff distance," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 9, pp. 901-916, 1999.
- [38] S.H. Park, K.M. Lee and S.U. Lee, "A line feature matching technique based on an eigenvector approach," *Computer Vision and Image Understanding*, vol. 77, no. 3, pp. 263-283, 2000.
- [39] M. Dastani and R. Scha, "Languages for Gestalts of line patterns," *Journal of Mathematical Psychology*, vol. 44, no. 4, pp. 429-449, 2003.
- [40] B. Huet and E.R. Hancock, "Line pattern retrieval using relational histograms," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 12, pp. 1363-1370, 1999.
- [41] W.H. Wang and Y.C. Chen, "Point pattern matching by line segments and labels," *Electronics Letters*, vol. 33, no. 6, pp. 478-479, 1997.
- [42] A.C. Evans, N.A. Thacker and J.E.W. Mayhew, "Pairwise representations of shape," *International Conf. Pattern Recognition*, vol. 1, pp. 133-136, 1992.
- [43] R.K.K. Yip, "A multi-level dynamic programming method for line segment

- matching in axial motion stereo," *Pattern Recognition*, vol. 31, no. 11, pp. 1653-1668, 1998.
- [44] O. De Vel and S. Aeberhard, "Line-based face recognition under varying pose," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, pp. 1081-1088, 1999.
- [45] D. Zhang and W. Shu, "Two novel characteristics in palmprint verification: datum point invariance and line feature matching," *Pattern Recognition*, vol. 32, no. 4, pp. 691-702, 1999.
- [46] A. Chalechale and A. Mertins, "Line segment distribution of sketches for Persian signature recognition," *Conf. Convergent Technologies for Asia-Pacific*, vol. 1, pp. 11-15, 2003.
- [47] J.R. Beveridge and E.M. Riseman, "How easy is matching 2D line models using local search?" *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 6, pp. 564-579, 1997.
- [48] Y.L. Chang and J.K. Aggarwal, "Line correspondences from cooperating spatial and temporal grouping processes for a sequence of images," *Computer Vision and Image Understanding*, vol. 67, no. 2, pp. 186-201, 1997.
- [49] Y.S. Gao, M.K. Leung, S.C. Hui and M.W. Tananda, "Facial expression recognition from line-based caricatures," *IEEE Trans. Systems, Man and Cybernetics*, vol. 33, no. 3, pp. 407-412, 2003.

- [50] B. Kamgar-Parsi, "Algorithms for matching 3D line sets," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 582-593, 2004.
- [51] H.H. Chen and T.S. Huang, "Matching 3-D line segments with applications to multiple-object motion estimation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 1002-1008, 1990.
- [52] B.G. Park, S.U. Lee and K.M. Lee, "A novel face recognition technique using face-ARG matching," *Asian Conf. Computer Vision*, pp. 252-257, 2004.
- [53] L. Parida and D. Geiger, "Junction: detection, classification, and reconstruction," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 7, pp. 687-998, 1998.
- [54] M. Trajkovic and M. Hedley, "Fast corner detection," *Image and Vision Computing*, vol.16, pp. 75-87, 1998.
- [55] S.C. Bae, I.S. Kweon and C.D. Yoo, "COP: a new corner detector," *Pattern Recognition Letter*, vol. 23, no. 11, pp. 1349-1360, 2001.
- [56] S. Ando, "Image field categorization and edge/corner detection from gradient covariance," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 2, pp. 179-190, 2000.
- [57] I. Biederman and J. Gu, "Surface versus edge-based determinants of visual recognition," *Cognitive Psychology*, vol. 20, pp. 38-64, 1988.

-
- [58] B.K. Ray and R. Pandyan, "ACORD—an adaptive corner detector for planar curves," *Pattern Recognition*, vol. 36, no. 3, pp. 703-708, 2003.
- [59] M. Marji and P. Siy, "A new algorithm for dominant points detection and polygonization of digital curves," *Pattern Recognition*, vol. 36, no. 10, pp. 2239-2251, 2003.
- [60] J.G. Choi, S.W. Lee and H.S. Kang, "Dynamic programming approach to optimal vertex selection for polygon-based shape approximation," *IEE Proc. Vision, Image and Signal Processing*, vol. 150, no. 5, pp. 287-291, 2003.
- [61] W.Y. Wu, "An adaptive method for detecting dominant points," *Pattern Recognition*, vol. 36, no. 10, pp. 2231-2237, 2003.
- [62] A.M.N. Fu, H. Yan and K. Huang, "A curve bend function based method to characterize contour shapes," *Pattern Recognition*, vol. 30, no. 10, pp. 1661-1671, 1996.
- [63] J.H. Han and T. Poston, "Chord-to-point distance accumulation and planar curvature: a new approach to discrete curvature," *Pattern Recognition Letter*, vol. 22, no. 10, pp. 1133-1144, 2001.
- [64] A. Quddus and M.M. Fahmy, "Fast wavelet-based corner detection technique," *Electronics Letters*, vol. 35, no. 4, pp. 287-288, 1999.
- [65] P.Y. Yin, "Ant colony search algorithms for optimal polygonal approximation of plane curves," *Pattern Recognition*, vol. 36, no. 8, pp. 1783-1797, 2002.

- [66] O. Daescu and N. Mi, "Polygonal chain approximation: a query based approach," *Computational Geometry*, vol. 30, no. 1, pp. 41-58, 2005.
- [67] A. Kolesnikov and P. Franti, "Reduced-search dynamic programming for approximation of polygonal curves," *Pattern Recognition Letters*, vol. 24, no. 14, pp. 2243-2254, 2003.
- [68] P.Y. Yin, "A discrete particle swarm algorithm for optimal polygonal approximation of digital curves," *Journal of Visual Communication and Image Representation*, vol. 15, no. 2, pp. 241-260, 2004.
- [69] K.M. Ku and P.K. Chiu, "Polygonal approximation of digital curve by graduate iterative merging," *Electronics Letters*, vol. 31, no. 6, pp. 444-446, 1995.
- [70] B.K. Ray and K.S. Ray, "A new split-and-merge technique for polygonal approximation of chain coded curves," *Pattern Recognition Letters*, vol. 16, no. 2, pp. 161-169, 1995.
- [71] D.W. Sun and C.J. Du, "Segmentation of complex food images by stick growing and merging algorithm," *Journal of Food Engineering*, vol. 61, no. 1, pp. 17-26, 2004.
- [72] M. Sarfraz, M.R. Asim and A. Masood, "Piecewise polygonal approximation of digital curves," *Proc. the Eighth International Conf. Information Visualisation*, pp. 991-996, 2004.

- [73] J.H. Horng, "Improving fitting quality of polygonal approximation by using the dynamic programming technique," *Pattern Recognition Letters*, vol. 23, no. 14, pp. 1657-1673, 2002.
- [74] W.Y. Wu and M.J. Wang, "Detecting the dominant points by the curvature-based polygonal approximation," *CVGIP: Graphical Model Image Process*, vol. 55, pp. 79-88, 1993.
- [75] P. Hall and B.A. Turlach, "On the estimation of a convex set with corners," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 3, pp. 225-234, 1999.
- [76] W. Niblack, "Updates to the QBIC system," *International Conf. Storage and Retrieval for Image and Video Databases VI, SPIE*, vol. 3312, pp. 150-161, 1998.
- [77] J.R. Smith and S.F. Chang, "Visually searching the web for content," *IEEE Multimedia*, pp. 12-19, 1997.
- [78] M.S. Lew, "Next-generation web searches for visual content," *IEEE Computer*, vol. 33, pp. 46-53, 2000.
- [79] G. Lu, "Techniques and data structures for efficient multimedia retrieval based on similarity," *IEEE Trans. Multimedia*, vol. 4, pp. 372-384, 2002.
- [80] A. Franco, A. Lumini and D. Maio, "Eigenspace merging for model updating," *International Conf. Pattern Recognition*, vol. 2, pp. 156-159, 2002.

- [81] R. Elmasri and A.B. Navathe, "Fundamentals of database systems," *Redwood City, CA: Benjamin Cummings*, 1994.
- [82] D.A. White and R. Jain, "Algorithms and strategies for similarity retrieval," *Storage and Retrieval in Image, and Video Database*, vol. 2060, pp. 634-648, 1997.
- [83] K.I. Lin, H.V. Jagadish and C. Faloutsos, "The TV-tree: an index structure for high-dimensional data," *VLDB J.*, vol. 3, pp. 517-549, 1994.
- [84] D.A. White and R. Jain, "Similarity indexing: Algorithms and performance," *Proc. SPIE Storage and Retrieval for Image and Video Databases*, vol. 2670, pp. 62-75, 1996.
- [85] S. Berretti, A. Del Bimbo and P. Pala, "Retrieval by shape similarity with perceptual distance and effective indexing," *IEEE Transactions on Multimedia*, vol. 2, pp. 225-239, 2000.
- [86] J. Nievergelt, H. Hinterberger and K.C. Sevcik, "The grid file: an adaptive, symmetric multikey file structure," *ACM Trans. Database Syst.*, vol. 9, pp. 38-71, 1984.
- [87] M. Charikar, C. Chekur, T. Feder and R. Motwani, "Incremental clustering and dynamic information retrieval," *ACM Symposium on Theory of Computing*, pp. 626-635, 1997.

- [88] R. Ng and A. Sedighian, "Evaluating multi-dimensional indexing structures for images transformed by principal component analysis," *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1996.
- [89] C. Faloutsos and K.I. Lin, "Fastmap: a fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets," *Proc. SIGMOD*, pp. 163-174, 1995.
- [90] S. Chandrasekaran, B.S. Manjunath, Y.F. Wang, J. Winkeler and H. Zhang, "An eigenspace update algorithm for image analysis," *CVGIP: Graphical Models and Image Processing Journal*, 1997.
- [91] S. Prabhakar, D. Agrawal and A.E. Abbadi, "Data clustering for efficient range and similarity searching," *SPIE Conf. Multimedia Storage and Archiving System III*, vol. 3527, pp. 419-430, 1998.
- [92] S. Dao, Q. Yang and A. Vellaikal, " MB^+ -tree: an index structure for content-based retrieval," *Multimedia Database Systems: Design and Implementation Strategies*, K. C. Nwosu, B. Thuraisingham, and P. B. Berra, Eds. Norwell, MA: Kluwer, 1996.
- [93] C.A. Shaffer, *A practical introduction to data structures and algorithm analysis*. NJ: Prentice-Hall, 1997.
- [94] O. Guttman, "R-tree: A dynamic index structure for spatial searching," *ACM SIGMOD International Conf. Management of Data*, pp. 47-57, 1984.

- [95] S. Berchtold, D.A. Keim and H.P. Kriegel, "The X-tree: an index structure for high-dimensional data," *Proc. 22nd International Conf. Very Large Databases*, pp. 28-39, 1996.
- [96] N. Beckmann, H.P. Kriegel, R. Schneider and B. Seeger, "The R^* -tree: an efficient and robust access method for points and rectangles," *ACM SIGMOD International Conf. Management of Data*, pp. 322-331, 1990.
- [97] K.I. Lin, H.V. Jagadish and C. Faloutsos, "The TV-tree: an index structure for high-dimensional data," *VLDB J.*, vol. 3, pp. 517-542, 1995.
- [98] P. Ciaccia, M. Patella and P. Zezula, "M-tree: an efficient access method for similarity search in metric spaces," *Proc. 23rd International Conf. Very Large Data Bases (VLDB'97)*, pp. 426-435, 1997.
- [99] S. Berchtold, C. Bohm and H.P. Kriegel, "The pyramid-technique: towards breaking the curse of dimensionality," *ACM SIGMOD International Conf. Management of Data*, pp. 142-153, 1998.
- [100] J. Nievergelt, H. Hinterberger and K.C. Sevcik, "The grid file: an adaptive symmetric multi-key file structure," *ACM Trans. Database System*, vol. 9, 1984.
- [101] A. Sajjanhar and G. Lu, "Indexing 2-D non-occluded shape for similarity retrieval," *SPIE Conf. Applications of Digital Image Processing XX*, vol. 3164, pp. 188-197, 1997.

- [102] A. Berman and L.G. Shapiro, "Efficient retrieval with multiple distance measures," *Conf. Storage and Retrieval for Image and Video Databases V, San Jose, CA*, 1997.
- [103] M. Stricker and A. Dimai, "Color indexing with weak spatial constraints," *Conf. Storage and Retrieval for Image and Video Databases IV*, vol. 2670, pp. 29-40, 1996.
- [104] C. Arcelli and G.S.D. Baja, "A width-independent fast thinning algorithm," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 7, no. 4, pp. 463-474, 1985.
- [105] U. Montanari, "A method of obtaining skeletons using a quasi-Euclidean distance," *Journal of the Association for Computing Machinery*, vol. 15, no. 4, pp. 600-624, 1968.
- [106] S. Suzuki and K. Ade, "Sequential thinning of object pictures using distance transformation," *International Conf. Pattern Recognition*, pp. 56-74, 1986.
- [107] C. Wayne and B. Philip, "Generation skeletons and centerlines from the distance transform," *Computer Vision, Graphics, and Image Processing: Graphic Model and Image Processing*, vol. 54, no. 5, pp. 420-437, 1992.
- [108] R.L. Brown, "The fringe distance measure: an easily calculated image distance measure with recognition results comparable to Gaussian blurring," *IEEE Trans. Systems, Man and Cybernetics*, vol. 24, no. 1, pp. 111-115, 1994.

-
- [109] V. Starovoitov, "A clustering technique based on the distance transform," *Pattern Recognition Letters*, vol. 17, no. 3, pp. 231-239, 1996.
 - [110] A. Rosenfeld and J.L. Pfaltz, "Sequential operations in digital picture processing," *Journal of the ACM*, vol. 13, no. 4, pp. 471-494, 1966.
 - [111] A. Rosenfeld and J.L. Pfaltz, "Distance functions on digital pictures," *Pattern Recognition*, vol. 1, pp. 33-61, 1968.
 - [112] M.N. Kolountzakis, "Fast computation of the Euclidean distance maps for binary images," *Information Processing Letters*, vol. 43, pp. 181-184, 1992.
 - [113] L. Chen and H.Y.H. Chuang, "A fast algorithm for Euclidean distance maps of a 2-D binary image," *Information Processing Letters*, vol. 51, pp. 25-29, 1994.
 - [114] T. Hirata, "A unified linear-time algorithm for computing distance maps," *Information Processing Letters*, vol. 58, pp. 129-133, 1996.
 - [115] I. Ragnemalm, "The Euclidean distance transform in arbitrary dimensions," *Pattern Recognition Letters*, vol. 14, no. 11, pp. 883-888, 1993.
 - [116] I. Sintorn and G. Borgefors, "Weighted distance transforms in rectangular grids," *Proc. Eleventh International Conf. Image Analysis and Processing*, pp. 322-326, 2001.
 - [117] G. Borgefors, "Distance transformations in digital images," *Computer Vision, Graphics, and Image Processing*, vol. 34, pp. 344-371, 1986.

- [118] A. Datta and S. Soundaralakshmi, "Fast parallel algorithm for distance transform," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 44, no. 5, pp. 429-434, 2003.
- [119] Y.H. Lee, S.J. Horng, T.W. Kao and F.S. Jaung, "Parallel computation of exact Euclidean distance transform," *Parallel Computing*, vol. 22, pp. 311-325, 1996.
- [120] H. Yamada, "Complete Euclidean distance transformation by parallel operation," *International Conf. Pattern Recognition*, 69-71, 1984.
- [121] X.L. Wang, "Some sequential algorithms for a generalized distance transformation based on Minkowski operations," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, no. 11, pp. 1114-1121, 1992.
- [122] M.A. Butt and P. Maragos, "Optimum design of Chamfer distance transforms," *IEEE Trans. Image Processing*, vol. 7, no. 10, pp. 1477-1484, 1998.
- [123] G. Borgefors, "Hierarchical Chamfer matching: a parametric edge matching algorithm," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 849-865, 1988.
- [124] A. Thayananthan, B. Stenger, P.H.S. Torr and R. Cipolla, "Shape context and Chamfer matching in cluttered scenes," *Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 127-133, 2003.
- [125] C. Fouard and G. Malandain, "3-D Chamfer distances and norms in anisotropic grids," *Image and Vision Computing*, vol. 23, no. 2, pp. 143-158, 2005.

- [126] Y.M. Sharaiha and N. Christofides, "A graph-theoretic approach to distance transformations," *Pattern Recognition Letters*, vol. 15, pp. 1035-1041, 1994.
- [127] R.A. Lotufo, A.X. Falcao and F.A. Zampiroli, "Fast Euclidean distance transform using a graph-search algorithm," *Proc. Eighth Brazilian Sym. Computer Graphics and Image Processing*, pp. 269-275, 2000.
- [128] P.K. Saha, "Fuzzy distance transform: theory, algorithms, and applications," *Computer Vision and Image Understanding*, vol. 86, pp. 171-190, 2002.
- [129] H. Rodriguez-Ponce and R. Rosas-Romero, "Distance transform approximation with a multilayer perceptron for 2D image recognition," *Proc. Fifteenth International Conf. Electronics, Communications and Computers*, pp. 213-216, 2005.
- [130] C.T. Huang and O.R. Mitchell, "A Euclidean transform using grayscale morphology decomposition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, no. 4, pp. 443-448, 1994.
- [131] F.Y. Shih and O.R. Mitchell, "A mathematical morphology approach to Euclidean distance transformation," *IEEE Trans. Image Processing*, vol. 1, no. 2, pp. 197-204, 1992.
- [132] P.P. Das and B.N. Chatterji, "Knight's distance in digital geometry," *Pattern Recognition Letters*, vol. 7, no. 4, pp. 215-226, 1988.
- [133] H.S. Alhichri and M. Kamel, "Multi-resolution image registration using multi-

- class Hausdorff fraction,” *Pattern Recognition Letters*, vol. 23, no. 1-3, pp. 279-286, 2002.
- [134] D.P. Huttenlocher and W.J. Rucklidge, “A multi-resolution technique for comparing images using the Hausdorff distance,” *Conf. Computer Vision and Pattern Recognition*, pp. 705-706, 1993.
- [135] W.J. Rucklidge, “Comparing images using the Hausdorff distance under translation,” *Conf. Computer Vision and Pattern Recognition*, pp. 654-656, 1992.
- [136] A. Pujol and J.J. Villanueva, “A supervised modification of the Hausdorff distance for visual shape classification,” *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 16, no. 3, pp. 349-359, 2002.
- [137] Y. Wang, “Robust object matching using a modified version of the Hausdorff measure,” *International Journal of Image and Graphics*, vol. 2, no. 3, pp. 361-373, 2002.
- [138] D.P. Huttenlocher, R.H. Lilien and C.F. Olson, “View-based recognition using an eigenspace approximation to the Hausdorff measure,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 9, pp. 951-955, 1999.
- [139] C.F. Olson and D.P. Huttenlocher, “Automatic target recognition by matching oriented edge pixels,” *IEEE Trans. Image Processing*, vol. 6, no. 1, pp. 103-113, 1997.

- [140] M.P. Dubuisson and A.K. Jain, "A modified Hausdorff distance for object matching," *Proc. Twelfth International Conf. Pattern Recognition and Image Processing*, pp. 566-568, 1994.
- [141] R. Azencott, F. Durbin and J. Paumard, "Multiscale identification of buildings in compressed large aerial scenes," *International Conf. Pattern Recognition*, vol. 3, pp. 974-978, 1996.
- [142] D.G. Sim, O.K. Kwon and R.H. Park, "Object matching algorithms using robust Hausdorff distance measures," *IEEE Trans. Image Processing*, vol. 8, no. 3, pp. 425-429, 1999.
- [143] C.V. Stewart, "A new robust operator for computer vision: theoretical analysis," *Conf. Computer Vision and Pattern Recognition*, pp. 1-8, 1994.
- [144] D.G. Sim and R.H. Park, "Two-dimensional object alignment based on the robust oriented Hausdorff similarity measure," *IEEE Trans. Image Processing*, vol. 10, no. 3, pp. 475-483, 2001.
- [145] G. Ciocca and R. Schettini, "Content-based similarity retrieval of trademarks using relevance feedback," *Pattern Recognition*, vol. 34, no. 8, pp. 103-119, 2001.
- [146] Y.S. Kim and W.Y. Kim, "Content-based trademark retrieval system using a visually salient feature," *Image and Vision Computing*, vol. 16, no. 12, pp. 931-939, 1998.

- [147] S.D. Lin, B.Y. Hsu and X.L. Yang, "Content-based trademark retrieval system using a new region based shape description method: the distance-angle pair-wise histogram," *Proc. Twenty Second International Conf. Distributed Computing Systems Workshops*, pp. 1-5, 2002.
- [148] A. Folkers and H. Samet, "Content-based image retrieval using Fourier descriptors on a logo database," *International Conf. Pattern Recognition*, vol. 3, pp. 521-524, 2002.
- [149] H. Peng and S. Chen, "Trademark shape recognition using closed contours," *Pattern Recognition Letters*, vol. 18, no. 8, pp. 791-803, 1997.
- [150] G. Cortelazzo, G.A. Mian, G. Vezzi and P. Zamperoni, "Trademark shapes description by string matching techniques," *Pattern Recognition*, vol. 27, no. 8, pp. 1005-1018, 1994.
- [151] R.J.M. den Hollander and A. Hanjalic, "Logo recognition in video stills by string matching," *International Conf. Image Processing*, vol. 2, pp. 517-20, 2003.
- [152] W.H. Leung and T. Chen, "Trademark retrieval using contour-skeleton stroke classification," *Conf. Multimedia and Expo*, vol. 2, pp. 517-520, 2002.
- [153] F. Cesarini, E. Francesconi, M. Gori, S. Marinaid, J.Q. Sheng and G. Soda, "A neural based architecture for spot noisy logo recognition," *Proc. Fourth International Conf. Document Analysis and Recognition*, pp. 175-179, 1997.

- [154] K. Zyga, R. Price and B. Williams, "A generalized regression neural network for logo recognition," *International Conf. Knowledge Based Intelligent Engineering System and Allied Technologies*, pp. 475-478, 2000.
- [155] K.J. Zyga, J. Schroeder and R. Price, "Logo recognition using retinal coding," *Conf. Signals, Systems and Computers*, vol. 2, pp. 1549-1553, 2004.
- [156] A. Soffer and H. Samet, "Using negative shape features for logo similarity matching," *International Conf. Pattern Recognition*, vol. 1, pp. 571-573, 1998.
- [157] J. Neumann, H. Samet and A. Soffer, "Integration of local and global shape analysis for logo classification," *Pattern Recognition*, vol. 23, no. 12, pp. 1449-1457, 2002.
- [158] T.D. Pham, "Recognition of trademarks with spatial statistics and neural learning," *Conf. Artificial Intelligence Systems*, pp. 205-210, 2002.
- [159] B.M. Mehtre, M.S. Kankanhalli and W.F. Lee, "Shape measures for content based image retrieval: a comparison," *Information Processing and Management*, vol. 33, no. 3, pp. 319-337, 1997.
- [160] M.D. Heath, S. Sarkar, T. Sanoeki and K.W. Bowyer, "Comparison of edge detectors: a methodology and initial study," *Computer Vision and Image Understanding*, vol. 69, pp. 38-54, 1998.
- [161] R. Nevatia and K.R. Babu, "Linear feature extraction and description," *Computer Graphics and Image Processing*, vol. 13, pp. 257-269, 1980.

- [162] Y.S. Gao and M.K. Leung, "Face recognition using line edge map," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 764-779, 2002.
- [163] M.K. Leung and Y.H. Yang, "Dynamic two-strip algorithm in curve fitting," *Pattern Recognition*, vol. 23, no. 1/2, pp. 69-79, 1990.
- [164] F. Attneave, "Some information aspects of visual perception," *Psychol. Rev.*, vol. 61, pp. 183-193, 1954.
- [165] M. Flickner and H. Sawhney, "Query by image and video content: the qbic system," *IEEE Computer*, vol. 23, pp. 23-39, 1995.
- [166] A. Pentland, R.W. Picard and S. Sclaroff, "Photobook: content-based manipulation of image database," *Proc. SPIE Vol 2185 Storage and Retrieval for Image and Video Database 11*, 1994.
- [167] A.K. Jain and A. Vailaya, "Image retrieval using color and shape," *Pattern Recognition*, vol. 29, pp. 1233-1244, 1996.
- [168] C.P. Lam, J.K. Wu and B. Mehtre, "Star – a system for trademark archival and retrieval," *Proc. Asian Conf. Computer Vision*, pp. 214-217, 1995.
- [169] I.J. Kim, J.H. Lee, Y.M. Kwon and S.H. Park, "Content based image retrieval method using color and shape feature," *Proc. IEEE International Conf. Information, Communications and Signal Processing*, pp. 948-954, 1997.

-
- [170] R. Brunelli and O. Mich, "Histograms analysis for image retrieval," *Pattern Recognition*, vol. 34, pp. 1625-1637, 2001.
 - [171] S. Belongie, J. Malik and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 24, pp. 509-522, 2002.
 - [172] B. Huet and E.R. Hancock, "Relational object recognition from large structural libraries," *Pattern Recognition*, vol. 35, no. 9, pp. 1895-1915, 2002.
 - [173] J.Y. Chen, M.K. Leung, Y.S. Gao, "Noisy logo recognition using line segment Hausdorff distance," *Pattern Recognition*, vol. 36, no. 4, pp. 943-955, 2003.
 - [174] Y.S. Gao, M.K. Leung, "Human face profile recognition using attributed string," *Pattern Recognition*, vol. 35, no. 2, pp. 353-360, 2002.
 - [175] D. Zhang, W.K. Kong, J. You, M. Wong, "Online palmprint identification," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 9, pp. 1041-1050, 2003.
 - [176] D. Hall, J.L. Crowley, "V.C. Verdire, View invariant object recognition using colored receptive fields," *Machine Graphics and Vision*, vol. 9, no. 2, pp. 341-352, 2000.
 - [177] J. Malik, S. Belongie, T. Leung, J. Shi. "Contour and texture analysis for image segmentation," *Computer Vision*, vol. 43, no. 1, pp. 7-27, 2001.

Appendix A

Results of round corner refinement

In the following pages, some results of feature point refinement on input images are given. They are selected according to their contour points (from 689 to 8842 in ascending order). The first column is gray-level images. The second column is feature points extracted using Dyn2S. The third column show the results obtained from round corner point refinement.

Appendix B

The test logos used in our experiment

In the following pages, all the test logos are given.

Figure B.1: Test logos from set 1.

Figure B.2: Test logos from set 2.

Figure B.3: Test logos from set 3.

Figure B.4: Test logos from set 4.

Figure B.5: Test logos from set 5.

Figure B.6: Test logos from set 6 (part 1).

Figure B.7: Test logos from set 6 (part 2).

Figure B.8: Test logos from set 7 (part 1).

Figure B.9: Test logos from set 7 (part 2).

Appendix C

The test images from real objects used in
our experiment

In the following pages, all the test images of real objects are given.

Figure C.1: Test images of real objects from set 1.

Figure C.2: Test images of real objects from set 2.

Figure C.3: Test images of real objects from set 3.

Figure C.4: Test images of real objects from set 4.

Appendix D

The mixed images used in our experiment

In the following pages, all the results from mixed-object recognition are given.

App. D The mixed images used in our experiment

213

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

App. D The mixed images used in our experiment

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10
	SMK
Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

App. D The mixed images used in our experiment

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10

Input	Rank1	Rank2	Rank3	Rank4	Rank5

	Rank6	Rank7	Rank8	Rank9	Rank10