Shape retrieval and matching based on line segments.

Chi, Yan Ling.

2007

Chi, Y. L. (2007). Shape retrieval and matching based on line segments. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/2478>

<https://doi.org/10.32657/10356/2478>

Nanyang Technological University

Downloaded on 26 Apr 2023 10:16:35 SGT

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SHAPE RETRIEVAL AND MATCHING
BASED ON LINE SEGMENTS

**SHAPE RETRIEVAL AND MATCHING
BASED ON LINE SEGMENTS**

CHI YAN LING

**CHI YAN LING
SCHOOL OF COMPUTER ENGINEERING
2007**

2007

Shape Retrieval and Matching Based on Line Segments

Chi Yan Ling

School of Computer Engineering

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

2007

ACKNOWLEDGEMENTS

I am greatly indebted to my supervisor, Dr. Maylor K.H. Leung, for the invaluable guidance that he has provided to me over the last three years. His vision and enthusiasm were a constant source of inspiration, and provided the indispensable impetus behind all aspects of this work. His refusal to accept any lowering of standards will undoubtedly not be the least of the lessons that he has taught me.

The lifelong training that my parents instilled in me is the cornerstone that allowed me to accomplish this work. Their teachings about the value of hard work and independence is always my source of impetus. For this and so much more, which I cannot enumerate, I remain eternally thankful. I would also like to thank my husband, Jianhui, for his consistent support behind me.

The Database Technology Laboratory and the Center for Advanced Media Technology provided space and equipment to work with, and the technical staff, especially Teo, Huifang and Quick, untangled the complicated workings of the machines. G. Li, J. Li, H. Zou, H. Ou, T. Xiang, Do Tien Dung, Quan Thanh Tho, M. Rodionov, B. Zhou, H. Dong, C. Liu, B. Lu, X. Yuan, W. Wang, T. Jin, X. Yu, helped to lighten the atmosphere at work and beyond. To all, my heartfelt thanks.

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

TABLE OF CONTENTS

Abstract	I
List of Figures	III
List of Tables	VII
Chapter 1 Introduction	1
1.1 Problem Description	1
1.2 Motivations	1
1.3 Proposed Approach.....	2
1.4 Objectives.....	5
1.5 Input Data	6
1.6 Report Organization	7
Chapter 2 Background	8
2.1 2-D Shape Representation	8
2.1.1 Global methods	10
2.1.2 Local methods	18
2.1.3 Summary	23
2.2 Indexing Technique.....	23

2.2.1	Dimension reduction	24
2.2.2	Multi-dimensional indexing structure	25
2.2.3	Filtering techniques	33
2.2.4	Summary	34
2.3	Image Retrieval Systems.....	34
2.4	Conclusion.....	37
Chapter 3	System Overview	38
3.1	Image Preprocessing	38
3.2	Shape Primitive Extraction	40
3.3	Shape Representation	40
3.4	Indexing Structure Generation	41
3.5	Image Retrieval	42
3.6	Summary	42
Chapter 4	Image Pre-processing	43
4.1	Image Loading.....	44
4.2	Edge Detection and Thinning.....	44
4.3	Contour Tracing.....	45
4.4	Polygonal Approximation.....	46
4.5	Summary and discussion	48
Chapter 5	Shape Primitive Extraction	49
5.1	Overview	49

5.2	Basic Concepts	53
5.3	Partition Point Determination	55
5.4	Primitive Evaluation and Classification.....	60
5.5	Results	63
5.6	Primitives filtering	69
5.7	Summary and discussion	71
Chapter 6	Shape Representation.....	72
6.1	Overview of local shape description techniques	73
6.2	Local Structure Construction	74
6.2.1	Gestalt Laws for Grouping.....	75
6.2.2	Reference Feature Selection.....	77
6.2.3	Local Neighbor Selection.....	81
6.2.4	Results	93
6.3	Invariant Local Structure Representation.....	102
6.3.1	Line to line pair-wise geometric attributes.....	102
6.3.2	Arc to arc pair-wise geometric attributes	104
6.3.3	Arc-line pair-wise geometric attributes.....	105
6.4	Object Representation	106
6.5	Summary	109
Chapter 7	Indexing Structure Generation	110
7.1	Indexing Technique Overview	110

7.2	Indexing Structure Generation	112
7.3	Summary	113
Chapter 8 Retrieval by Indexing and Voting		114
8.1	Intersection	115
8.2	Filtering	116
8.3	Ranking	117
8.4	Retrieval Timing Analysis	127
8.5	Display	129
8.6	Summary	131
Chapter 9 Results and Discussion		132
9.1	Input Data	132
9.2	Test methodology	136
9.3	Experimental Results	140
9.3.1	Experiment 1: Letter retrieval	142
9.3.2	Experiment 2: Robustness to 3D Distortions	143
9.3.3	Experiment 3: Similarity based Retrieval	144
9.3.4	Experiment 4: Partial Query	144
9.3.5	Experiment 5: Mixed-object Query	146
9.3.6	Experiment 6: Real Object Image Query	147
9.3.7	Error analysis	152
9.3.8	Sensitivity analysis	157

9.4	Discussion and Comparison	159
9.4.1	Discussion	159
9.4.2	Comparison	161
9.5	Summary	166
Chapter 10	Conclusion and Future Work.....	167
10.1	Work Summary	168
10.2	Contributions.....	169
10.3	Future Work.....	170
10.4	Concluding remark	171
	References.....	172
	Appendix A 3-D stationery model images	183
	Appendix B Results of shape retrieval	187

Abstract

A novel local structure based image retrieval (ALSBIR) approach is proposed in this thesis to build a general framework for object / image retrieval. It performs speedy interpretation of input images and retrieval of structurally similar models from large database according to the input. It is first of its kind to experiment with mixed-object query in shape retrieval. A novel local shape representation inspired by Gestalt psychology and a novel hypercube indexing structure driven by dynamic programming are therefore proposed to cope with partially occluded object query and mixed-object query in the clutter environment besides traditional queries of recognition with respect to noisy environment, transform invariance, similarity, 3D distortion, and real objects with shadow. The approach can tackle partially occluded objects in a cluttered environment since it employs local information only and does not require segmentation (separation) of whole input objects from complex background. Its performance degrades gracefully with respect to the amount of structural information that is being occluded or lost. It is able to infer retrieval results from incomplete information of an input by first extracting consistent and structurally unique local neighborhood information from inputs or models, and then voting on the optimal matches. The local neighborhood information applicable to the system can be shape, color, texture feature, etc. The mechanism of voting is based on a novel hypercube based indexing structure,

and driven by dynamic programming. The proposed concepts have been applied on shape features and tested on database with thousands of images. Its efficiency and effectiveness were tested by comparisons with the 6 nearest-neighbors shape description. The proposed method was found performing as good as the 6 nearest-neighbors on single object retrieval. However it out-performed the 6 nearest-neighbors on the test of occluded and cluttered object queries. The sensitivity analysis and error analysis is also provided to fully assess the proposed concepts. The system is robust in the occluded and cluttered environments, while relatively sensitive to the corruption of major shape primitives.

List of Figures

Figure 1.1: Examples of model images in the Databases.....	7
Figure 1.2: Examples of test images.	7
Figure 2.1: Display of B ⁺ -tree structure [7].	26
Figure 2.2: Display of R-tree structure [7].....	28
Figure 2.3: Display of X-tree structure [113].....	29
Figure 2.4: Display of TV-tree structure [114].	30
Figure 3.1: Flowchart of the local structure based retrieval system.	39
Figure 4.1: Flowchart of the image pre-processing module.	43
Figure 4.2: Display of extracted edges.....	45
Figure 4.3: Display of extracted contour.....	46
Figure 4.4: Illustration of the dynamic two-strip algorithm.....	47
Figure 4.5: Display of Line Edge Map.	48
Figure 5.1: Flowchart of Primitive Extraction.	50
Figure 5.2: Partition Points and Primitives.	51
Figure 5.3: Flowchart of Contour Partition.....	52
Figure 5.4: Illustration of Angle Difference.....	53
Figure 5.5: Illustration of Circumferential Angle.	54
Figure 5.6: Illustration of contour.	55

Figure 5.7: Start / end partition points.	56
Figure 5.8: Corner partition point.	56
Figure 5.9: Inflexion partition point.....	57
Figure 5.10: Stem partition point.	58
Figure 5.11: Illustration of partition point refined.	59
Figure 5.12: Illustration of primitive classification.	62
Figure 5.13: Primitives extracted from simple objects.	64
Figure 5.14: Primitives extracted from similar objects.....	65
Figure 5.15: Primitives extracted from noisy objects.	66
Figure 5.16: Primitive extracted from occluded objects.	67
Figure 5.17: Primitives extracted from real 3-D object images.....	68
Figure 5.18: Display of primitive filtered.	70
Figure 5.19: Parameters of the parallel lines / arcs.	70
Figure 6.1: Illustration of the five Gestalt Laws.	76
Figure 6.2: Examples of various percentages of the top reference features.	79
Figure 6.3: Reference primitives extracted from images with frame.	80
Figure 6.4: Examples of dissimilarity measures.	83
Figure 6.5: Symmetry angles computation.	85
Figure 6.6: The probability distribution.	87
Figure 6.7: Accuracy test with respect to different number of neighbors.....	93
Figure 6.8: Typical local structures of simple objects	94
Figure 6.9: Typical local structures of similar / occluded objects	96

Figure 6.10: Typical local structures of noisy / distorted objects	98
Figure 6.11: Typical local structures of mixed objects	100
Figure 6.12: Typical local structures of real 3D object images	101
Figure 6.13: Pair-wise Geometric Attributes.	103
Figure 6.14: Relationships between two circles.....	105
Figure 6.15: Typical units in a 5-D hypercube structure H	108
Figure 6.16: 2-D spherical extension.	109
Figure 7.1: Display of Indexing Structure.	113
Figure 8.1: Flowchart of retrieval strategy.....	114
Figure 8.2: Intersection process.	115
Figure 8.3: Accuracies according to different thresholds	116
Figure 8.4: Similarity based query by L1-norm / L2-norm.	118
Figure 8.5: Partial query by L1-norm / L2-norm.	119
Figure 8.6: Neighborhood-matching.....	120
Figure 8.7: The desirable discriminating power of $nID(s')$	121
Figure 8.8: Global shape dominated query.	124
Figure 8.9: Non-global shape dominated query.	125
Figure 8.10: Partial query using two similarity measures.....	126
Figure 8.11: Relationship between retrieval time and input pattern size	128
Figure 8.12: Relationship between retrieval time and average pattern size.....	128
Figure 8.13: Relationship between retrieval time and database size	129
Figure 8.14: Query results in text mode.....	130

Figure 8.15: Visual query results.	130
Figure 9.1: Examples of test images.	133
Figure 9.2: Illustration of the precision-recall curves.	137
Figure 9.3: Procedures in local structure comparison.	138
Figure 9.4: Procedures in similarity measure comparison.	139
Figure 9.5: Precision-recall plot comparisons.	141
Figure 9.6: Example results of letter retrieval.	142
Figure 9.7: Example results of 3D-distortion queries.	143
Figure 9.8: Example comparison of similarity based query.	145
Figure 9.9: Example results of similarity based queries.	146
Figure 9.10: Example results of partial queries.	146
Figure 9.11: Example results of cluttered object queries.	150
Figure 9.12: Example results of 3D-object queries.	151
Figure 9.13: Error due to corrupting the major primitives.	155
Figure 9.14: Error due to misclassified primitives.	156
Figure 9.15: The acceptable misclassifications of primitives.	156
Figure 9.16: Error due to insufficient object analysis.	157
Figure 9.17: Sensitivity curves.	158
Figure 9.18: Neighbor selection with / without a frame.	161

List of Tables

Table 9.1: Description of Input Data.....	134
Table 9.2: Schemes and formula used in result comparison.	139
Table 9.3: Query types in relevant work.	162

Chapter 1 Introduction

1.1 Problem Description

With rapidly increasing usage of digital camera, finding a moderate number of matches relevant to a given visual query efficiently is the key to a wide range of applications, such as robot vision, trademark registration, etc. In some applications, such as robot vision and web searching, the visual query is unpredictable. It can have more than one object in cluttered environments, but it is not possible to segment (separate) each object from its background. The approaches that can tackle multiple object queries in cluttered environment are of great benefit to applications of shape retrieval.

1.2 Motivations

There are many good approaches [1-5] on image retrieval when there is only one complete / incomplete object in the visual query and the task is to find suitable models which match the input. However, there is a lack of effective approach when there are multiple unknown objects in the query with the complications of one being partially occluded by other or partially merged with other. To complicate the matter further, a model can also be seen as the composition of multiple objects if each part of the model

can be perceived separately as an independent object. Nevertheless, the task is to find models which best match each individual object from an input, or to determine whether a specific model is in the input. In addition, searching for perceptually similar objects related to a given query is a harder issue, since the mechanism of human perception is not yet clearly understood even in Psychology. The requirement for low computation complexity usually makes solving above problems even more difficult. Due to these difficulties, the research on shape retrieval in the cluttered environment is very challenging and there is a gap in the literature on mixed-object queries. In this thesis, we face these difficulties and aim to propose an effective retrieval approach that can tackle all the above challenges efficiently, and fill the literature gap.

1.3 Proposed Approach

Human can identify and recognize an occluded object from a cluttered environment, provided that enough or distinguishable portion of the object can be observed. This is an assumption for us to model unique local structure that can be consistently extracted. Hence, the proposed technique can extract unique local structure only when enough or distinguishable portion of the object can be observed.

To retrieve objects in cluttered and occluded environment, we propose a novel local structure based image retrieval (**ALSBIR**) approach to infer retrieval results from incomplete information of an input. The method employs a novel shape representation and a novel hypercube based indexing technique. The novel shape representation is able to extract consistent and structurally unique local neighborhood even in occluded /

cluttered environment. It uses both lines and arcs as primitives, and constructs the local structures guided by five Gestalt psychological laws. It has advantage over a good retrieval method, the six-nearest neighbor method [6], which uses only lines as primitive and midpoint distances to select neighbors. Such a simplified approach can run into problem when it is dealing with circular objects in complicate background. Besides, a statistical approach is employed to accommodate the inexact local structure construction, and it is of benefit to the structure construction in occluded/cluttered environment when the construction can be affected by distortion and segmentation error. In addition, the proposed shape representation do not required the detected line and arc being labeled as part of a certain object, i.e. our shape representation can be built on realistic input data instead of depending on pre-identified object outline or boundary. This characteristic makes our method distinct from other good shape retrieval methods [3-5]. Moreover, the proposed method represents each local structure using a set of pair-wise geometric relationships, which are simple, invariant to translation / scaling / rotation, relatively robust to end-point-erosion, and sufficient for discrimination. The novel hypercube based indexing technique works by first storing related local structural information in the hypercube indexing structure and followed by careful sorting procedure of the structural relation to produce accurate result. The sorting procedure is implemented using dynamic programming. Different from [6] where the identity of a local structure disappeared once the structural attributes are extracted and inserted into the histogram. Our approach retains such information in the indexing structure for finer matching later. In addition, The proposed indexing structure has the potential to

cope with “dimensional curse”[7], since there are limited numbers of local structures that one can extract from an input image to probe the pre-built indexing structure.

Besides object retrieval in cluttered and occluded environment, the techniques proposed have the potential to be easily extended to interpret a scene later. The rationale is as following. Because of the preservation of local structure’s identity, i.e. models are informatively stored in the indexing structure, it is easy for the proposed method to locate the matched local structures from both the input and models, thus, possible to locate a desirable object in a scene, That is, it has the potential for scene interpretation.

The concepts proposed in this thesis also have the potential to be generalized to other primitives easily, such as color and texture. For example, the perceptual organization rules¹, such as the contrast dye, the color similarity and texture similarity, can help to select color or texture patches to construct local color or texture information. The relative spatial information of these patches can be numerically represented by geometric attributes. They then can be stored into the hype-cube indexing structure for retrieval. Similarly, other shape primitives, e.g. quadratic curves, can also be employed for relative accurate approximation of shapes.

The proposed system has been extensively tested, and compared with other methods, e.g. method in [6]. The results are encouraging, especially in the occluded and cluttered environments, even though relatively sensitive to the corruption of major

¹ <http://www.macalester.edu/psychology/whathap/ubnnp/aesthetics/perception.html>

shape primitives. Moreover, we believe that the proposed method can go even further if combined with other types of primitives. Therefore, it is not impossible for the method to be used for robot vision navigation in the future.

Three papers [8-10] have stemmed from the work in this thesis. Our next paper is in preparation.

1.4 Objectives

To retrieval object in cluttered and occluded environment, several objectives are to be fulfilled in this thesis:

1. To set up an image retrieval system framework, which is expect to fast identify distorted / occluded / cluttered input objects and retrieve perceptually similar models from large database, provided that enough or distinguishable portion of the object can be observed;
2. To integrate a image pre-processing method, which is expected to extract line edge map from the gray scale image;
3. To propose a primitive extraction method, which is expected to extract line and arc primitives appropriately;
4. To propose a local structure construction method, which is expected to extract consistent and structurally unique local neighborhood from input and models under various adverse environments, that is, the construction is expected to be built on realistic input data instead of depending on pre-identified object outline or boundary;
5. To propose a local structure representation method, which is expected to represent the local structure simply and invariantly, that is, the representation is simple, invariant to translation / scaling / rotation,

- relatively robust to end-point-erosion, and sufficient for discrimination;
6. To propose a hypercube indexing structure, which is expected to store the image database efficiently and informatively for retrieval;
 7. To propose a retrieval strategy, which is expected to search in the image database efficiently;
 8. To propose a similarity measure, which is expected to match input and candidate models effectively;
 9. To design experiments, which is expected to measure the performance of the proposed system accurately and illustrate the proposed concept clearly.

1.5 Input Data

The system performance is evaluated in the retrieval of the logo database² and stationery images database³ downloaded from internet. The logo database consists of 2711 simple characters and 984 complex trademarks and graphs. Some examples are shown in Fig. 1.1. The character set includes characters with different rotations, scalings, 3D-transformations and added noise. Furthermore, it also includes combination of multiple characters, and occluded characters. The complex trademarks are used to assess the ability of our approach in similarity based retrieval, partial query, cluttered object query, and 3D distortion query. The stationery images are used to explore more general applications. Examples of some of the test images are displayed in Fig. 1.2.

² <http://www.eurecom.fr/~huet>

³ <http://www-2.cs.cmu.edu/~cil/v-images.html>.

letter A	logo	graph	cartoon	scissors	tape
(a) logo database				(b) stationery database	

Figure 1.1: Examples of model images in the Databases.

Corrupted image	Cylindrical transformed image	Similar images

Occluded objects	Mixed objects	Real object taken from random view angle

Figure 1.2: Examples of test images.

1.6 Report Organization

The rest of this thesis is organized as follows: the background of shape retrieval is reviewed in chapter 2. The system is overviewed in chapter 3. Chapter 4 introduces image pre-processing. The primitive extraction is described in chapter 5. The local structure based shape representation is presented in chapter 6. We illustrate the generation of indexing structure in chapter 7. The method of shape retrieval by indexing is described in chapter 8. Chapter 9 compares and discusses the experimental results for system evaluation. The conclusion and future work are given in chapter 10.

Chapter 2 Background

Content-based image retrieval (CBIR) is driven by the wide availability of digital cameras, Internet, and cheap storage devices of high capacity. It is explored in the directions of new users and usage styles, new applications, new interaction modes, large data sets, and new retrieval methods. A CBIR task consists of two components: storage and retrieval. In the storage process, images are processed to extract features relating to content. The extracted features are then indexed and stored in the database. In the retrieval process, the query content is analyzed first. A searching process is then performed and similarities are measured. Finally, the results are ranked, displayed, and feedbacked. The related techniques in the task are on content description and indexing. Usually, the image content is in terms of shape, color, and texture. Since our work only concerns shape features, here, we will review the related work in shape representation and shape indexing.

2.1 2-D Shape Representation

The shape representation is a language for describing shapes or some aspects of shapes. Shape features are very powerful when used in similarity retrieval since shape is usually strongly linked to object functionality and identity, and people can recognize characteristic objects only from their shapes. MPEG-7 gives general criteria [1] for

designing shape descriptors: 1) good retrieval accuracy, 2) small memory requirement, 3) general applications, 4) low computational complexity, 5) no drastic performance degradation with increasing database, and 6) hierarchical coarse to fine representation structure.

Shape descriptors can be classified as: 1) *global* methods [1, 2, 11] and *local* methods [3-6, 12-18] according to whether the shape is represented as a whole or represented by primitives or sub-parts, 2) *contour-based* methods [2-4, 19-21] and *region-based* methods [1, 22-26] according to whether shape features are extracted from the outline contour / boundary or the distribution of all pixels within outline contour / boundary, 3) *Transform domain* methods [16, 27-31] and *space domain* methods [15, 32, 33] according to whether the shape features are derived from the transformed domain or spatial domain.

Since a local structure based method is proposed in this work, the shape descriptors will be reviewed with respect to the global / local methods in this chapter.

Since there is no widely acceptable standard for the classification of global methods or local methods at this moment, we here give our own definitions according to people's common sense. That is, global methods are descriptors representing shapes as a whole no matter what kind of features are used. A typical example is *Moments* [34]. Even though the moments themselves support local information (local features), their applications on shape description usually depend on global properties of shapes, i.e. the moments of the whole shapes are calculated and matched, unless the shape has been pre-segmented and pre-classified into sub-parts. Therefore, they are classified as

global methods according to our definitions. We defined the local methods as descriptors representing shapes by primitives or sub-parts.

2.1.1 Global methods

Global methods compute feature vectors from the whole shapes. They do not divided shapes into sub-parts. The similarity of shapes is usually measured using a metric distance between the acquired feature vectors. Global methods require complete object information; hence, they are not suitable for retrieval of occluded objects, or mixed objects.

1. Contour-based methods

Contour-based methods only exploit shape boundary information. They extract shape features from the outline contour / boundary of the object. *Contour-based methods emulate well the shape similarity perception of the human visual system* [35]. They are suitable for similarity based image retrieval.

➤ Simple shape descriptors

The simple contour descriptors, such as perimeter, convexity, circular variance, and elliptic variance [36], have low discriminating power on different shapes. They are not suitable to be the standalone shape descriptor. Thus, they commonly work with other shape descriptors and as filters to screen false hits. For their definitions, one can refer to [36].

➤ Shape signature

Shape signature represents shape by a 1-D function derived from shape boundary points, e.g. polar signature, centroidal profile, complex coordinates, tangent angle, cumulative angle, curvature and chord-length [37-40]. In [37], the centroidal profile is used to represent a shape. It is translation and scale invariant but not rotation invariant, thus, shift matching is used to find the best match between two shapes. The shift matching is too time-consuming to retrieve online. In addition, it is sensitive to noise. Slight changes in the boundary can cause large errors in matching. In [40], the critical points on a 2D digital shape are first detected, then the shape is represented by polar signature. The shape representation is transformation, scale and orientation invariant. The computational complexity is approximately proportional to the number of the pixels on the contour. However, the authors assume the object boundary is a simple closed contour with unit width. The method's effectiveness and efficiency on complicated objects are unknown. Moreover, the very important perimeter, *critical level*, is set subjectively, which restricts the method's application on general objects.

➤ Autoregressive models

Autoregressive models have proven to be useful in shape description [41-43]. It is based on the stochastic process modeling of the characteristic function. Autoregressive model is characterized by a set of unknown parameters and an independent noise sequence. In [41], the shape contour is modeled by a time-varying autoregressive model,

which can be used to detect curvature extreme points of the synthesized contours. The model uses least squares to estimate the model parameters, and uses the maximum likelihood approach to determine the model order. In [43], the object boundary coordinate sequence is processed by a nonlinear exponential autoregressive model. The invariant features are then extracted and used to recognize shapes. The parameters of the autoregressive model are calculated by using a neural network. Their disadvantage is that both methods only work on single, simple, closed contours. In case of complex boundaries, a small number of autoregressive model parameters are not sufficient for an adequate description.

➤ Scale-space methods

The scale space methods [44-51] have been used as shape features in many application. Witkin [52] first proposed a scale-space filtering approach which provides a useful representation for significant object features. The representation is created by tracking the position of inflection points in a shape boundary filtered by a low-pass Gaussian filter of varying widths. Based on this work, Mokhtarian and Mackworth [45, 46] proposed a multiscale curvature-based shape representation technique for plane curves. The method represents the shape by a curvature scale space (CSS) image. The matching between shapes is on the peaks of individual branches, which is proved to be very complex and expensive. Lindeberg [53] enriches the scale space theory by exploring its discrete aspects, and a scale space for 1-D discrete signals comprising a continuous scale parameter is defined. Recently, Lowe [54] proposed scale invariant

feature transform (SIFT) for shape representation. The image features are detected through a staged filtering approach that identifies stable points in scale space. A nearest-neighbor indexing method is used to identify candidate object matches. The results show that SIFT can recognize objects in cluttered partially-occluded images robustly. However, it cannot efficiently deal with identifying the multiple unknown objects in the scene from a large model image database, since it is impractical to use each model to search the scene. In addition, its ability on similarity-based retrieval is not clear.

➤ Convex hull

A region R is convex if and only if for any two points $x_1, x_2 \in R$, the whole line segment x_1x_2 is inside the region. The convex hull of a region is the smallest convex region H which satisfies the condition $H \subset R$. The convex hull can be used to describe region shape properties and can be used to build a tree structure of region concavity. A convex hull can be extracted using both boundary tracing methods [55] and morphological methods [37, 56]. In [57], a set of affine invariants derived from the convex hull of scattered feature points are used to represent objects. These invariants are used to establish the correspondences between the convex hull vertices of a test image with a reference image. A point matching approach for recognition is then applied. Since the convex hull is generated from scattered feature points of single object, the proposed method is not suitable for mixed objects retrieval in cluttered environments.

➤ Fourier descriptor (FD)

The standard Fourier transforms $F(m, n)$ of an image function $f(x, y)$ is

$$F(m, n) = \sum_{x=0}^{X-1} \sum_{y=0}^{Y-1} f(x, y) \exp(-j2\pi)(\frac{mx}{X} + \frac{ny}{Y}) \quad (2-2)$$

It is invariant to translation and rotation, and is relatively insensitive to noise and shape variations by analyzing shape in the spectral domain. The Fourier descriptors are suitable for the analysis and synthesis of the closed planar curves. In [28], the contour is first represented by a boundary function of tangent angle vs. arc length. The Fourier transform is then employed to extract coefficients for shape description. Ezer and Anarim [58] applied Fourier descriptors to the problem of classification of 2-D images of airplanes. Lin and Mithell et al. use FD to describe partial shapes [59, 60]. Arbter et al. introduced the affine-invariant FD to take into consideration of affine shape description [61]. Granlund employed the Fourier invariants which describe the rotational symmetry of shapes [62]. Rauber and Steiger proposed a *UNL* FD to describe disjointed or articulated contour shape [63], where the *UNL transform* was defined as a normalizing transform of the parametric curves from the Cartesian to the polar coordinate system with special properties. The advantage of FD descriptors is their easy implementation and well-developed theoretical foundation. The disadvantage is that the object's boundary has to be identified before the object can be represented. It makes FD not suitable for object recognition in cluttered environment, since identifying the object's boundary in cluttered environment is an even more difficult issue than shape representation.

➤ Wavelet descriptor (WD)

The major difference between Wavelet descriptor (WD) and Fourier descriptor (FD) is that WD is of multi-resolution in both spatial space, in the form of zero-crossing points, and spectral space, in the form of resolution levels. However, the increase of spatial resolution will certainly sacrifice frequency resolution. In [30], the wavelet transform zero-crossings are used to represent shape. The wavelet function is of the first derivative of a cubic spline. The matching scheme is the all-level shift matching of all the zero-crossing points of the wavelet approximation of shapes. Thus, the computation for similarity measurement is large and dependent on the complexity of the shape. Apart from the matching complexity, the dyadic wavelets used do not support local information description on the shape boundary, therefore, is not suitable for occluded object recognition.

➤ Other methods

Belongie etc [4] extracts a global feature, i.e. *shape context*, for each corresponding point on the object contours. The similarity of shapes is measured using a correspondence-based point-to-point matching. The shape context at a point p is a histogram map which stores the length and orientation of the vectors from point p to all the other boundary points in log-polar space. The histogram of each point is flattened and concatenated to form the context matrix. The matching of two shapes is to minimize the total cost of matching between one context matrix and all the permutations of another context matrix. To reduce the matching overhead, it samples

the boundary at a number of points and uses the shortest augmenting path algorithm for the matrix matching. Its major disadvantage is relying on the pre-identified outlines of shapes, which makes it impractical to mixed objects retrieval.

2. Region-based methods

Region-based methods take into account of all the pixels within a shape region to obtain the shape representation. Moments are commonly used to describe shapes.

➤ Simple shape descriptors

Area, circularity, rectangularity, elongation [64] are simple global, region-based descriptors. They usually work together with other shape descriptors. Their definitions are described as following.

Area = the number of pixels in the shape.

Circularity = $\text{perimeter}^2 / \text{area}$, it describes the compactness of a shape. The most compact shape is a circle (4π). All other shapes have compactness larger than 4π .

Rectangularity = $\text{area of object} / \text{area of bounding box}$. It is 1 for a rectangular and can approach 0 in the limit.

Elongation = the ratio of the height and width of a rotated minimal bounding box.

Eccentricity = the distance from the center of the conic section to the focus / half the distance across a conic curve along its long principal axes.

➤ Moments based methods

The standard 2-D moment, $m(u, v)$ of an image function $f(x, y)$ is defined as

$$m(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) x_u y_v dx dy \quad (2-1)$$

The use of moments for shape description was initiated by Hu [34]. He proved that moment-based shape description is invariant under image translation, scaling, and rotation. However, the description is not 100% information preserving since the higher order moments are not concerned. In [65], orthogonal moments, such as Zernike

moments and Legendre moments, are proposed. Zernike moments are computed as $A_{nm} = \frac{n+1}{\pi} \int_0^{2\pi} \int_0^{\infty} V_{nm}^*(\rho, \theta) \cdot f(\rho, \theta) \rho d\rho d\theta$, where $V_{nm} = R_n(\rho) \cdot e^{im\theta}$, and $R_n(\rho) =$

$$\sum_{s=0}^{(n-|m|)/2} (-1)^s \cdot \frac{(n-s)!}{s! \left(\frac{n+|m|}{2} - s\right)! \left(\frac{n-|m|}{2} - s\right)!} \rho^{n-2s}. \text{ Legendre moments are computed as}$$

$$\lambda_{mn} = \frac{(2m+1)(2n+1)}{4} \cdot \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} P_m(x) \cdot P_n(x) \cdot f(x, y) dx dy, \quad P_n(x) = \frac{1}{2^n n!} \frac{d^n (x^2 - 1)^n}{dx^n}.$$

The orthogonal moments allow object reconstruction since Legendre and Zernike polynomials are both complete sets of an orthogonal basis. Angular Radial Transformation (ART) [1] is the Zernike moments transform defined on a unit disk that consists of the complete orthonormal sinusoidal basis functions in polar coordinates.

For more methods of moments, one can refer to [22]. Generally, moment shape descriptors are usually concise, robust, easy to compute and match. The disadvantage of moment methods is that it is difficult to correlate high order moments with the shape's physical features. In addition, it requires pre-identifying a complete object region, and therefore, it is not suitable for occluded object or mixed object description.

➤ Shape matrix

Goshtashby proposed a shape matrix which was derived from a circular raster sampling technique [66]. A polar raster of concentric circles and radial lines were overlaid on a shape with respect to its centroid. The binary value of the shape was sampled at the intersections of the circles and radial lines. The shape matrix was formed by corresponding columns to the circles and rows to the radial lines. The resulting matrix representation was normalized by the maximum radius, therefore, was invariant to translation, rotation, and scaling. Taza and Suen [67] represented shape using a weighed shape matrix. The comparison of matrices was performed to classify shapes. Perui et al. [68, 69] proposed a shape description based on the relative areas of the shape contained in concentric rings located in the shape center of the mass. A shape vector scheme was used for shape matching. They were not suitable for occluded object recognition since shape matrix was extracted from global shapes. In addition, the shape matrix methods are sensitive to noise due to the sparse sampling.

2.1.2 Local methods

Local methods break the shape into its natural sub-parts, called *primitives*, using a particular criterion. The final representation is usually a string, a graph or a tree. The similarity measure is computed by string matching or graph matching.

1. Contour-based methods

In contour-based methods, the shape boundary is broken down into segments. The

following are commonly used methods of boundary decomposition.

➤ Chain coding

Chain coding method was introduced in 1961 by Freeman [70], which described an object by a sequence of unit-size line segments with a given orientation. In implementation, a planar curve was superimposed with a grid, the boundary points were approximated to the nearest grid point. From a selected starting point, the planar curve was traced on the boundary of a shape. At each boundary point, a choice of only four or eight directions can be considered to look for the next boundary point. Then the direction at each point on the curve can be coded by only two or three bits. Thus, an arbitrary curve can be represented by a sequence of unit vectors with a limited set of possible directions. Chain coding technique can provide significant data reduction. The disadvantage of this code is that it has high dimensions and is sensitive to noise. Chain coding is often used as an input to a higher level analysis, such as polygonal approximation, or boundary curvature detection.

➤ Polygonal Approximation

Polygonal approximation is the technique that a planar curve is fitted by a sequence of straight line segments. The line segments are used as primitives. The feature for each primitive can be expressed as internal angle, the distance from the next vertex, or its coordinates. Polygonal approximation can significantly reduce the amount of data for further processing. It depends on only local properties and can be used to approximate

the occluded contour. To make use of its advantages, several researchers used polygonal approximation to describe shapes. For example, Hu and Yan [71] developed a polygonal approximation method of digital curves based on the principles of perceptual organization. This method is independent of the starting point and searching directions, and is rotation and translation invariant. Hosur and Ma [72] approximated a shape contour by a polygon with a minimal number of vertices for a given allowable approximation error and initial vertex. Recently, polygonal approximation was used as the input to high level shape description [73, 74]. Huet and Hancock [73] extracted higher level local structures from the result of polygonal approximation to represent the object. The polygonal approximation method has disadvantages of missing some features such as sharp corners and spikes, and sometimes being sensitive to the setting of the starting points for partitioning curves. However, it has an advantage of low computational complexity and simple implementation. A promising direction is to use it for high-level image analysis.

➤ Shape decomposition

In shape decomposition techniques, a shape is represented as a combination of components. The idea is to represent complex shapes in terms of simple components. Pavlidis [75] stated the global shape decomposition as: *“Among the boundary points find sets of points which are closely related. Such sets maybe used to assign labels to corresponding parts of the object.”* In [76], the curvature zero-crossing points from a Gaussian smoothed boundary were used to obtain primitives, called *tokens*. The feature

for each token was its maximum curvature and its orientation. The similarity between two tokens was measured by the weighted Euclidean distance. The features were not rotation invariant due to the orientation. The method depends on the continuous and identified boundary of the shape. Therefore, it is not suitable for mixed object recognition from cluttered environment.

➤ Shape invariants

Shape invariants are primitives that represent properties of the shape, which remain unchanged under appropriate transformations. Common invariants include (1) geometric invariants such as *cross-ratio*, *length ratio*, *distance ratio*, *angle*, *invariants from coplanar points* [6, 73, 77, 78]; (2) algebraic invariants such as *determinant*, *eigenvalues*, and *trace* of matrices related to plane conics [55, 79]; (3) differential invariants such as *curvature*, *torsion* and *Gaussian curvature* [3, 76]. Geometric invariants and algebraic invariants are suitable in situations when boundaries can be represented by straight lines or algebraic curves. Typical applications of them are for man-made object recognition. However, if object boundaries cannot be represented by lines or algebraic curves, invariants are difficult to derive. This problem restricts invariants' application in real world. Even though differential invariants local in nature can be formed in this case, they are large in number and expensive in computation. In addition, when invariants are used to represent shapes, its stability in noisy environment, its distinguishing power among complicate objects or scenes, and its matching load are challenging issues.

2. Region-based methods

Region-based methods decompose the shape region into parts which are then used for shape representation and description.

➤ Morphological methods

Mathematical morphology has evolved as a useful tool for various image processing. It is suitable for shape-related processing since morphological operations are directly related to the object shape. There are a number of shape description approaches using mathematical morphology, such as the shape description method based on the area of morphologically processed images [80] or morphological covariance [81]. Loui [82] used the geometrical correlation function, which was based on morphological covariance, to represent 2D shape. Medial axis transform (MAT) is one well-known morphology method. It is represented by Blum's medial axis transform [83]. It converts a full figure boundary into a skeleton or stick figure drawing, which is defined as a connected set of medial lines along the limbs of figure, i.e. the skeleton. The skeleton can then be decomposed into segments and represented as a graph according to certain criteria. The matching between shapes becomes a graph matching problem. The computation of the medial axis is a rather challenging problem. Medial axis is highly sensitive to boundary noise and variations. Preprocessing the contour of the shape and finding its polygonal approximation has been suggested as a way of overcoming these problems. But, as has been pointed out by Pavlidis [75], obtaining such polygonal approximation have been quite sufficient in itself for shape description.

2.1.3 Summary

In this section, shape representation techniques have been reviewed. Global methods are usually robust against noise and shape distortions. Local methods are more suitable than global methods for occluded or partial visible object retrieval. Contour-based approaches are relatively efficient since only a part of shape information is used. However, they are generally sensitive to noise and variation. Region-based methods are relatively robust since all the shape information is available. However, they might be not efficient on similarity based retrieval [35]. In addition, it should be pointed out that the classification is only based on the typical applications of the techniques since some methods can be classified into different groups depending on applications. For example, the convex hull is traditionally a contour-based method, however, researchers recently compute the convex hull of a set of scattered points that even have no boundary. In this meaning, it can also be classified as region based method.

2.2 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. Some retrieval systems work without using a data structure [84, 85]. 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 the retrieval speed [86]. Some adopt

indexing techniques, e.g. k -d tree [87] is extended for image retrieval. Generally, three techniques are commonly used to cope with the “*dimensional curse*” [7]. 1) *Dimension reduction* (DR), such as Karhunen-Loeve Transform (KLT) [88-91]. 2) *Multidimensional indexing structure*, such as B^+ -tree[92], VAM-split R-tree[93], TV-tree[94], k -d tree [95, 96], M-tree [3], quad-tree, and grid file [97] are commonly used. 3) *Filtering techniques*[98, 99]. The following provides a survey of the techniques and data structures for fast retrieval.

2.2.1 Dimension reduction

Even though the dimension of feature vectors in image retrieval is normally very high, the embedded dimension is much lower [95]. It is beneficial to perform dimension reduction before indexing techniques are used.

KLT: KLT has been used in dimension reduction [88-91]. Eigenimage is KLT’s variation in face recognition and PCA is KLT’s variation in information analysis. In [100], Ng and Sedighian followed the eigenimage approach to reduce dimension. In [101], Faloutsos and Lin proposed a fast approximation to KLT to perform the dimension reduction. The 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 [102]. One should be cautious in reducing dimension blindly, since information can be lost if the reduction is below the embedded dimension. To avoid

blind reduction, a post-verification stage is necessary.

Clustering: clustering [103, 104] is a powerful tool in performing dimension reduction. It groups similar items into a cluster and represents the cluster by the centroid. In retrieval, only the similarities between the query vector and the cluster representatives are calculated to locate relevant candidates. Clustering technique makes retrieval efficient, and facilitates browsing and navigation. It can also work together with other data structures for higher search efficiency, e.g. centroid of clusters and / or items within each cluster is organized in a certain data structure for efficient retrieval.

2.2.2 Multi-dimensional indexing structure

Most work in index or data structures is carried out in the database and geographic information system (GIS) community and extended for image retrieval.

1. Binary tree

A binary tree is made up of a finite set of elements called nodes. 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 .

2. B⁺-tree

A B⁺-tree is a hierarchical structure with a number of nodes [105]. Each node in a B⁺-tree contains n pointers and $n-1$ keys, as shown in Fig. 2.1. 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 n is used to access all records which have key values larger than or equal to key $n-1$. To search in B^+ -trees with N records, the required number of key value comparison and memory access is in the order of $O(\log_n N)$.

Figure 2.1: Display of B^+ -tree structure [7].

3. MB^+ -tree

The MB^+ -tree is an extension of the standard B^+ -tree from one dimension to multiple dimensions [106]. The structure of MB^+ -trees and B^+ -trees are similar, but the search methods are different since MB^+ -trees can support range and nearest-neighbor queries. For ease of explanation, the MB^+ -tree is described in a two-dimensional (2-D) space. In 2-D space, each 2-D feature vector can be considered as a point. The entire feature space can be considered as a large rectangle. Therefore, if it is divided into rectangular regions so that each region has a similar number of feature vectors and these regions are somehow ordered, feature vectors can be accessed based on their corresponding region numbers. MB^+ -trees are built based on this idea. Therefore, the MB^+ -tree is obtained by extending the B^+ -tree in the following ways: 1) replace each key value with a rectangular region; 2) the pointers of leaf nodes point to lists of feature vectors within corresponding rectangular regions.

4. The k -d tree

The k -d tree is an extension of the binary tree [107]. Its node 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. The search process in a k -d tree is similar to the insertion process. Range queries can be implemented relatively easily in a k -d tree since its key values decreases on the left branch and increases on the right branch. The main problem with k -d trees is that the tree is not balanced and depends on the order of object insertion. The search complexity is the same as linear search in the worst case.

5. R-tree family

The R-tree is a multidimensional generalization of the B^+ -tree, hence the tree is height balanced [108]. 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. Fig. 2.2 shows a 2-D feature space with bounding rectangles and the corresponding R-tree. The query object is represented by its minimum bounding rectangle (MBR). A common query is starting from the root and traversing down the sub-trees. The search efficiency of an R-tree is largely determined by *coverage* and *overlap* [109]. Efficient R-tree search demands that both coverage and overlap be minimized. However, it is impossible to minimize both at the same time. A balanced criterion must be used to achieve the best result.

(a) 2-D feature space with bounding rectangles.

(b) Structure of an R-tree.

Figure 2.2: Display of R-tree structure [7].

There are some variations of the R-tree (R^+ , R^* , VAMSplit, SS, SS^+), which differ mainly in the criteria used for node splitting and branching. The R^* -tree is a R-tree improved by minimizing coverage [110]. The R^+ -tree is proposed to overcome the

overlap problem of internal nodes of the R-tree [109]. VAMSplit R-tree [95] is constructed by splitting the feature space into rectangles by recursively choosing splits of the data set using the dimension of the maximum Variance and choosing a split that is Approximately the Median. SS-tree [111] and SS^+ -tree [112] are two variants of the R-tree. The minimum bounding region used in an SS-tree is a sphere. The SS^+ -tree uses a tighter bounding sphere for each node which is an approximation to the smallest enclosing sphere, and a different split algorithm from SS-tree.

6. X-tree

Figure 2.3: Display of X-tree structure [113].

The X-tree [113] is a high-dimensional structure based on the R^* -tree as in Fig. 2.3. 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.

7. TV-tree

The TV-tree is a variant of the R-tree [114]. It organizes the data into a hierarchical structure as in Fig. 2.4. 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. The basic assumption made is that a set of multidimensional feature vectors tend to have similar values on some dimensions. Thus, only *active dimensions* (dimensions having different values) are used to build a tree, which leads to a shallower tree requiring fewer disk accesses. The number of dimensions used to organize (index) feature vectors is then effectively reduced. The main problem with the TV-tree is that it can only be applied to some very specific applications where its assumption holds.

Figure 2.4: Display of TV-tree structure [114].

8. M-tree

M-tree, proposed in [115], 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. The partitioning algorithm is critical to the effectiveness of the tree. A major difference of M-tree from other metric trees is that it gives an 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.

Each node in an M-tree is either a leaf node that stores a database object, or an internal node that stores a so-called routing object. Each routing object has an associated pointer referring to as the root of a covering tree. All objects in the covering tree are within the covering radius. An entry for a database object in a leaf is quite similar to that of a routing object, but no covering radius is needed, and the pointer field stores the actual object identifier, which is used to provide access to the whole object possibly residents on a separate data file. The M-tree is an effective dynamic index for similarity queries used for high-dimensional spaces. However, the storing of computed distances at each node implies higher storage overhead. The distance is assumed to be a metric distance that can only cover a “spherical volume”. This reduces the flexibility of applying the index to some applications.

9. The Pyramid-Technique

The basic idea of the Pyramid-Technique [116] 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. For dynamic insertion of a point, the pyramid value of the point is determined and works as a key. Thus the point can be inserted into a B^+ -tree and stored in the corresponding data page of the B^+ -tree together with its key. For the update and delete operations, they can be done similarly. The assumption made is that the data is uniformly distributed. Therefore, how to adapt the Pyramid-Technique to real data is challenging.

10. Grid Files

The grid file is an extension of the fixed-grid (FG) access structure [117]. In the FG structure, an n -dimensional space is divided into equal-sized hypercubes. 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. The FG structure is simple and very effective when feature vectors are evenly distributed in the feature space. However, when feature vectors are unevenly distributed in the feature space, some grids will be empty while others will be too full, leading to lower search efficiency. The grid file is introduced to overcome the problem of the FG structure. The grid file method divides the feature space based on distribution of feature vectors instead of using fixed-size grids. That is, smaller grid sizes will be used for densely populated areas and larger grid sizes for sparsely

populated areas. Other concepts are the same as those of the FG structure.

11. Hashing

Hashing is defined as the process of accessing a record by mapping a key value to a position in the table. The function that maps key values to positions is called *hashing function*. The array that holds the records is called the *hash table*. Hashing is generally appropriate for sets. Its complexity is $O(1)$ theoretically if each record has a unique key. It is extremely efficient when implemented correctly. However, it is not suitable for range searches. This drawback restricts its application in image retrieval where range query is necessary.

2.2.3 Filtering techniques

Usually, the search space can be reduced by using filtering processes based on certain criteria. In [118], shape eccentricity is used as the filtering criterion, i.e. only shapes within the specified eccentricity range need to be searched. In [119], triangle inequality is used to reduce the number of direct feature comparison in a database search. In [120], 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 technique is then used to eliminate some unnecessary comparisons.

2.2.4 Summary

Some multi-dimensional data structures, filtering techniques and dimensional reduction methods for efficient retrieval are presented in this section. Even though efficient indexing techniques can speed up the retrieval process, most of the techniques are designed for traditional database queries but not for the similarity queries used in image retrieval, there was a need to explore the new characteristics and requirements for indexing structures in image retrieval. Such techniques were explored in [8, 121]. In [121], Tagare developed a tree adaptation approach which refined the tree structure by eliminating inefficient tree nodes for similarity queries.

2.3 Image Retrieval Systems

Several multimedia systems have been described in the literature [122-124].

MARS: MARS [125, 126] allows image queries using color, texture, shape and layout features. An adaptive resolution approach is used to do similar shape retrieval. It first normalized the shape by its major axis and bounding rectangle to guarantee rotation and scale invariance. Then, the shape is represented by grid cells of different sizes overlaying on different portions of the shape, i.e. the shape is represented at different resolutions. This is computed by applying quad-tree decomposition on the bitmap representation of the shape. The distance between shapes is measured by the

area difference. Similar shapes are queried by executing the k -NN algorithm. The method might be unsuitable for partial query, since insufficient prior shape information might cause the incorrect shape alignment.

PicToSeek¹: PicToSeek [127] uses the color invariance combined with shape invariance as the indexing features to provide content based access to images. The shape invariants are computed from the color invariant edges, and include similarity invariant and projective invariant. The similarity invariant is the relative angle between edges, which is independent of 2-D translation, rotation, and scaling. The projective invariant is the cross-ratio, which is independent of the projection viewpoint. In their experiments, The shape invariant can produce good results only when they are combined with the color invariant.

C-BIRD²: C-BIRD [128, 129] presents an illumination invariant and Recognition Kernel approach to conduct content-based image retrieval. It is a multi-level object representation and retrieval technique. The color features are extracted in the lowest resolution, the texture features are extracted in the intermediate level, and shape features are extracted in the highest resolution. The shape feature used in C-BIRD is the edge orientations. The shape is represented using the generalized Hough transform. The shape feature plays a minor role in the recognition kernel. The search is a coarse-to-fine strategy. It conducts from the coarsest level (color) to the highest resolution level.

¹ PicToSeek: <http://www.wins.uva.nl/research/isis/zomax/>

² C-BIRD: <http://jupiter.cs.sfu.ca/cbird/>

QBIC³: QBIC [23, 84] allows image databases to be queried using visual features such as color, shape and texture that are matched against a meaningfully clustered database of pre-computed features. The shape features are based on a combination of heuristic shape feature area, circularity, eccentricity, major axis orientation and a set of algebraic moment invariants. All shape are assumed to be unoccluded planar shapes allowing for each shape to be represented as a binary image. *R**-tree is used as indexing structure.

VideoQ⁴: VideoQ [130] is a web enabled content based video search system. It allows users to search compressed video based on a rich set of visual features and spatio-temporal relationships. Visual features include color, texture, shape, and motion. The shape features include the principal components generated using eigenvalue analysis, the first two order moments of region, the normalized area, and the percentage area. The composite distance function is used to measure object dissimilarity.

Blobworld⁵: Blobworld [131] is a system for content based image retrieval by finding coherent regions that may correspond to objects. It performs search on parts of an image which is treated as a group of blobs roughly representing homogeneous color or texture regions. No shape information is used in the system.

Other systems: More retrieval systems, such as CORE, Virage⁶, RetrievalWare⁷,

³ QBIC: <http://www.qbic.almaden.ibm.com/>

⁴ VideoQ: <http://www.ctr.columbia.edu/VideoQ/>

⁵ BlobWorld: <http://elib.cs.berkeley.edu/photos/blobworld/>

⁶ Virage: <http://www.virage.com/cgi-bin/query-e/>

Netra⁸, Informedia digital video library⁹, Chabot¹⁰, WebSeek¹¹ can be found in [132-138].

2.4 Conclusion

This chapter presents an overview of shape description, indexing technique and several current retrieval systems. For the shape description techniques, robustness and effectiveness in various adverse environments are two major concerns in the current research work. Even though good solutions of these concerns can expand the applications of CBIR, the efficiency is critical to whether or not these techniques are practical. That is why only simple shape descriptors are used in the current CBIR system. This situation indicates that the efficiency of most shape descriptors is far from meeting the requirement of a real-time task. To improve the efficiency of shape descriptors themselves deserve attentions, since indexing techniques can only speed up searching into the database. As for the indexing techniques, it is observed that there are rarely data structure is designed for the similarity based retrieval. This finding suggests a new research direction for CBIR.

⁷ RetrievalWare: <http://vrw.excalib.com/cgi-bin/sdk/cst/cst2.bat/>

⁸ Netra: <http://vivaldi.ece.ucsb.edu/Netra/>

⁹ Informedia: <http://www.informedia.cs.cmu.edu>

¹⁰ Chabot: <http://www.cs.berkeley.edu/ginger/chabot.html/>

¹¹ WebSeek: <http://www.ctr.columbia.edu/webseek/>

Chapter 3 System Overview

To build up a prototype shape retrieval system, we adopt several preprocessing models from previous work by other. Our main contributions are the proposed shape representation and shape indexing / retrieval techniques. The proposed system flowchart is displayed in Fig. 3.1 with five modules, i.e. image preprocessing, primitive extraction, shape representation, indexing structure generation, and image retrieval. Our work focuses on the last four modules.

3.1 Image Preprocessing

The function of the *image preprocessing* module is to extract the Line Edge Map (LEM) [139] from input images. The proposed system concerns with gray scale images for the simplicity of edge detection.

The preprocessing module includes image loading, edge detection, thinning, contour tracing, and polygonal approximation. Image loading is to read image information into the system. The gray scale image is loaded as TIFF file. Then edge detection will be performed to detect discontinuities in the image intensity. In this thesis, the edge detector of Nevatia and Babu [140] is adopted, which is inherited from previous work. Next, a thinning process is carried out to reduce thick edges to chains of single pixel. Based on the thinned edges, contours are traced. The contour stores

sequential lists of pixel locations which can be represented by integer coordinates. Finally, polygonal approximation computes lines that can best fit the curves. The dynamic two-strip algorithm [141] is employed in this stage. The fitted lines become the LEM representation. After image preprocessing, the LEM of the input image is used for shape primitive extraction.

→→ Test&model image flow → Test image flow → model image flow

Figure 3.1: Flowchart of the local structure based retrieval system.

3.2 Shape Primitive Extraction

The *primitive extraction* module extracts primitives from the LEM. Primitives are the simplest / elementary sub patterns decomposed from a complex pattern, and used to represent a complex pattern in terms of their interrelationships. In this work, both line and arc are adopted as primitives. A rule based method is proposed to extract lines and arcs. It firstly partitions the contours into primitive candidates by rules. Next these primitive candidates are classified as line features or arc features. The line / arc primitives are then evaluated. In this module, a line is represented by its two endpoints. An arc is represented by a sequential list of pixel location coordinates.

The line primitives and arc primitives extracted in this stage are used to construct the local structures.

3.3 Shape Representation

The *shape representation* module extracts local structure information by means of line / arc primitives and incorporates the local information into a hypercube based structure, which is a compact representation of the gray scale image. The process can be further divided into three steps as shown in Fig. 3.1. They are the *local structure construction*, *local structure representation*, and *incorporation*.

Local Structure Construction: The *local structure construction* is to build the local structure by means of image primitives. The local structure is a relatively high level structure, which can highlight the salient structure in one object. It is constructed

by one reference primitive and several neighbor primitives selected around the reference. In this work, all the primitives will be selected as a reference primitive. Two or more neighbor primitives are selected in one local structure.

Local Structure Representation: The *local structure representation* is to represent the extracted local structures numerically. The local structure is represented by the relationship between its reference primitive and the neighbor primitives, i.e. the primitive pairs. The pair-wise primitives are described by a set of four geometric attributes, which are invariant to translation / scaling / rotation, relatively robust to end-point-erosion, and sufficient for discrimination. Robust representation is important in the unpredicted adverse environments.

Incorporation: The *incorporation* is to incorporate all the pair-wise attributes into a hypercube based structure to generate a compact representation of the input image.

The *shape representation* module outputs a hypercube based structure, which is used in image indexing / retrieval. There are two choices after shape representation as shown in Fig. 3.1. If the input is a model, the structure is used to generate the indexing structure. If the input is a test image, the structure is used to retrieve similar model candidates from database.

3.4 Indexing Structure Generation

The *indexing structure generation* is one of the two branches after *shape representation* module. Indexing structure is a data structure of the same size as the

hypercube based structure in section 3.3. It is generated by organizing and storing all the models in the database dynamically, i.e. can add a new model representation efficiently at will. With the structure, efficient search strategy can be employed, thus, the large computational costs associated with similarity measures in model-by-model image retrieval can be reduced greatly.

3.5 Image Retrieval

The *image retrieval* module is the other branch after the *shape representation* module. It is used to fast identify similar models from the image database by searching the pre-built indexing structure. Image retrieval is performed by first nominating suitable candidates from the intersection of the input local structures to the indexing structure. This is then followed by a voting step that aligns and ranks the suitable candidates. Ranking is carried out by computing the similarity between the query and the candidates. The top ranking ones will be selected. Finally, the retrieved images are presented to the user in descending order according to their similarity to the query.

3.6 Summary

In this chapter, we give a general introduction to the proposed system. The system is divided into five modules according to the functionalities. Except for the *Image Preprocessing* module, we design and implement all the other modules. Description of techniques in each module will be described in detail in the following chapters.

Chapter 4 Image Pre-processing

The image preprocessing module operates on the input images and extracts useful information that is needed by other processing modules. This is the first stage of the prototype system. The work described in this chapter is adopted from the system implemented by Leung and Yang [141] and the flowchart is illustrated in Fig. 4.1.

Figure 4.1: Flowchart of the image pre-processing module.

4.1 Image Loading

Before an image can be processed, it must be read into the memory. There are many picture formats available for use. Some are public and others proprietary. A picture format specifies the layout of the image information in the file. It is possible to store color or gray scale images. Since images used in this thesis are all gray scale images, storing color is not required. The TIFF was chosen and a publicly available library called “libtiff”¹ was used. This library includes source code and examples, and provides common primitives for file loading and saving.

4.2 Edge Detection and Thinning

Edge detection is to detect discontinuities in the image intensity. It converts a 256 gray scale image to black and white. One example is illustrated in Fig. 4.2. White represents edges, and black represents the background. The edge detection technique has been studied most extensively, and many reliable algorithms have been proposed and implemented. Heath et al. [142] concluded that no single edge detector is best overall and it is difficult to predict which edge detector is best for any given image. In this thesis, the implementation from Nevatia and Babu [140] is inherited. However, the proposed method does not rely on any specific edge detector. Any edge detector can be used in this study.

¹ Libtiff TIFF library version 3.0. Copyright Sam Leffler and Silicon Graphics Inc. Can be found at <ftp://ftp.sgi.com/graphics/tiff>.

After edge detection, a thinning process is first used to reduce thick edges to chains of single pixel that can be easily traversed. Then the gray scale image can be reduced to its edge representation.

(a) Gray scale image

(b) Edges detected

(c) Edges thinned

Figure 4.2: Display of extracted edges.

4.3 Contour Tracing

A contour is traced by traversing the edge map generated by edge detection and thinning. It stores sequential lists of pixel locations which are represented by integer coordinates. Given a sequence of integer coordinate points $p_1(x_1, y_1)$, $p_2(x_2, y_2)$, ..., $p_i(x_i, y_i)$, $p_{i+1}(x_{i+1}, y_{i+1})$, ..., $p_n(x_n, y_n)$, where p_i is connected to p_{i+1} , $1 \leq i \leq n$, we have $|x_i - x_{i+1}| \leq 1$ and $|y_i - y_{i+1}| \leq 1$ but $|x_i - x_{i+1}|$ and $|y_i - y_{i+1}|$ are not both equal to zero. The quality of the resulting contours may affect the polygonal approximation; therefore, accurate contour extraction is important. One example of the output is shown in Fig. 4.3.

Figure 4.3: Display of extracted contour.

4.4 Polygonal Approximation

Polygonal approximation is accomplished by the dynamic two-strip algorithm (Dyn 2S) [141], which fits the curve with two parallel lines. Dyn 2S is a curve fitting method using feature points. The whole procedure has two stages. In the first stage, important properties of the data points are derived. Two strips are fitted by the left and right sides of each point on the curve respectively and the points inside each strip are approximated as the left or right straight line. The best fitted strips are derived by adjusting the width of the strip and rotate it up or down dynamically using the starting point as the pivot. Longer and narrower strips are preferred. The ratio, which is called the elongated 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 $f = E^{left} \cdot S^{\theta} \cdot E^{right}$, where E^{left} and E^{right} are the elongated values of the left and right strips, and $S^{\theta} = |180^{\circ} - \theta|$ is the acuteness of the angle θ between the two strips. Larger f is preferred in the fitting. In the second stage, a subset of the most representational points is selected based on the local maxima of point merits.

Figure 4.4: Illustration of the dynamic two-strip algorithm.

One example is illustrated in Fig. 4.4 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 , the pivot point, to encompass as many points as possible to its left side (i.e. the dashed-line strip). 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 strip). The measure of fit (i.e. elongate value) is recorded for each strip as a function of strip width but only the strip with the largest elongate value is marked and used. The same procedure is applied on the other side of the feature point. There are advantages of using strips to extract local features. First, digitization noise can be accommodated because the strip has width and it can enclose points that can be approximated as a straight line. Then the support region of one feature point can be reflected by the strips.

After fitting all the contours by polygons, the gray scale image is reduced to its Line Edge Map (LEM) [139], one example is shown in Fig. 4.5 where the dots represent the detected feature points.

Figure 4.5: Display of Line Edge Map.

4.5 Summary and discussion

The techniques used in image pre-processing are described in this chapter. Their visual effects are illustrated by several examples. The effectiveness of the image preprocessing is crucial to the other processes which build on top of its results. The proposed system may fail if most edges cannot be detected. Under such situation, one can seek different edge detectors to meet special requirement since the preprocessing step is application independent, or seek other primitives, such as color, and texture, since the concepts proposed is generic.

Chapter 5 Shape Primitive Extraction

In many recognition problems involving complex patterns, it is more appropriate to adopt a hierarchical perspective to view a pattern as being composed of simple sub-patterns which are themselves built from yet simpler sub-patterns. The simplest / elementary sub patterns to be recognized are called *primitives* and the given complex pattern is represented in terms of the interrelationships between these primitives [143].

In this work, the proposed shape primitives are line and arc. They are adopted as primitives due to their good properties: 1) efficient representation; 2) low storage demand; 3) ease of being detected. Cognitive psychological studies [144] indicated that human recognizes line drawings as quickly and almost as accurately as gray-level pictures. Line patterns are highly effective for shape representation [6]. Due to these good properties, line is one promising primitive feature for shape matching involving large databases. Arc is the complement of line when dealing with circular objects, since it can approximate circular objects better than line.

5.1 Overview

A simple approach is proposed for primitive (line / arc) extraction. It works on the LEM [139], which is a compact representation for gray scale images. The computation on primitive (line / arc) detection is greatly reduced if primitives are extracted from the

LEM instead of from the edge map of a gray scale image. The proposed approach can detect line and arc with just a few rules. Its flowchart is illustrated in Fig. 5.1.

Figure 5.1: Flowchart of Primitive Extraction.

First, the LEM is generated and contours are stored. One can refer to chapter 4 for the details. Second, the contours are partitioned and four types of partition points are generated. One example is shown in Fig. 5.2. They are the (1) *start* and *end points* of open contour, (2) *corner point* where the turning angle is sharp, (3) *inflexion point*, and (4) *stem point* which are defined as the endpoints of a line connected to an arc (where the line length is longer than the diameter of the arc). Third, the segment between 2 partition points will form a primitive candidate. Fourth, a classification rule is used to select arcs. After selection of arcs, the remaining segments are treated as lines. Finally,

the relevant parameters are calculated to evaluate the primitives.

Figure 5.2: Partition Points and Primitives.

(1: start and end point, 2: corner point, 3: inflexion point, 4: stem point)

The partition process is illustrated in Fig. 5.3. It starts from tracing a contour point by point (using index i). When a contour point is determined as the start / end point, corner point, or inflexion point, the program will examine if one primitive candidate can be segmented out of the contour by this point, i.e. the current partition point (PP_0). If the detected partition point (PP_0) is the first partition point on the contour ($j=0$), the program will detect its next partition point (PP_1) so that the primitive candidate can be extracted between the current partition point (PP_0) and the next partition point (PP_1). When the input is a closed contour, there might be only 0 or 1 partition point ($PP_1 = \Phi$). Under both situations, the complete contour will be extracted as a primitive candidate. If more than one partition point is detected prior to the current partition point, the primitive candidate is extracted between the current partition point (PP_1) and its immediate previous partition point (PP_0). After extraction of the primitive candidate, a similar partition process is carried out on the extracted primitive. The primitive

candidate may be re-partitioned if a stem point is detected.

Figure 5.3: Flowchart of Contour Partition.

By now, the two problems left to solve are: (1) how to determine the partition points? and (2) how to evaluate and classify the candidate primitive?

5.2 Basic Concepts

In this section, we introduce some terms associated with primitive extraction for later technical description.

Let point $P_s, \dots, P_{i-1}, P_i, P_{i+1}, \dots, P_e \dots$ be feature points detected on an edge map (contour) by the polygonal line fitting process [141]. $l_{i-1}, l_i, l_{i+1} \dots$ are line segments connecting feature point P_{i-2} and P_{i-1} , P_{i-1} and P_i , P_i and $P_{i+1}, \dots$. We define angle difference, circumferential angle, and curvature (i) as follows:

Angle Difference: the angle difference, $\Delta\theta_i$, shown in Fig. 5.4, is the turn angle from the vector $\overrightarrow{P_i P_{i+1}}$ to the vector $\overrightarrow{P_{i-1} P_i}$ which is calculated in formula 5-1. If the turn angle from vector $\overrightarrow{P_i P_{i+1}}$ to $\overrightarrow{P_{i-1} P_i}$ is counterclockwise, a positive sign is assigned to the angle difference. If the turn angle from vector $\overrightarrow{P_i P_{i+1}}$ to $\overrightarrow{P_{i-1} P_i}$ is clockwise, a negative sign is assigned to the angle difference. The range of the angle difference $\Delta\theta_i$ is $-\pi \sim \pi$.

Figure 5.4: Illustration of Angle Difference.

$$\Delta\theta_i = \pm \arccos \left[\frac{\overrightarrow{P_{i-1}P_i} \bullet \overrightarrow{P_iP_{i+1}}}{|\overrightarrow{P_{i-1}P_i}| |\overrightarrow{P_iP_{i+1}}|} \right] \quad (5-1)$$

Circumferential Angle: The circumferential angle, $\angle P_e P_i P_s$, is the smallest intersecting angle between line segment $P_s P_i$ and line segment $P_e P_i$, which is illustrated in Fig. 5.5.

Figure 5.5: Illustration of Circumferential Angle.

The $\angle P_s P_i P_e$ is calculated in formula (5-2), and its range is $0 \sim \pi$.

$$\angle P_s P_i P_e = \arccos \left[\frac{\overrightarrow{P_i P_s} \bullet \overrightarrow{P_i P_e}}{|\overrightarrow{P_i P_s}| |\overrightarrow{P_i P_e}|} \right] \quad (5-2)$$

Curvature (i): Curvature (i) refers to the curvature at point P_i , which is defined as the derivative of the tangent angle to the contour length. Let us assume that the contour C is represented by equation: $C(t) = [x(t), y(t)]$, where x , y are the coordinates of the pixels belonging to the contour and t is the arc length parameter, the curvature is calculated as:

$$\kappa(t) = \left| \frac{\dot{x}(t)\ddot{y}(t) - \dot{y}(t)\ddot{x}(t)}{(\dot{x}(t)^2 + \dot{y}(t)^2)^{3/2}} \right| \quad (5-3)$$

For circle, the curvature is constant, i.e. the reciprocal of radius, R , as in (5-4). For straight line, the curvature is zero as in (5-5).

$$\kappa = \frac{1}{R} \quad (5-4)$$

$$\kappa = 0 \quad (5-5)$$

5.3 Partition Point Determination

In this section, the determination of partition points is presented. The partition points are used to divide a planar, continuous contour into segments. In Fig. 5.6, a segment of the contour and a primitive candidate extracted from the contour are illustrated. Let $P_0, \dots, P_{n+1}$ be feature points detected by the polygonal approximation in [141], $l_1, \dots, l_{n+1}$ be the line segments connecting the feature point, and $\Delta\theta_1, \dots, \Delta\theta_n$ be the angle differences. Based on Fig. 5.6, the rules to determine the four types of partition points are described in the following.

(a) A piece of contour C.

(b) The primitive candidate C'.

Figure 5.6: Illustration of contour.

- (1) *Start and end points*: they are defined as the endpoints of an open contour, as shown in Fig. 5.7. No start point and end point will be detected on enclosed contour. The definition also serves as a rule to determine the start / end points, e.g. P_0 / P_{n+1} will be the start / end points respectively on the open contour shown in Fig. 5.6 (a).

Figure 5.7: Start / end partition points.

- (2) *Corner point*: it is defined as the feature point where the turning angle is large, as shown in Fig. 5.8. The sharp corner point is determined by formula (5-6). If formula (5-6) is true, P_i , the point with $\Delta\theta_i$, is seen as a sharp corner point. τ_1 is the tolerance. In this work, τ_1 is set as 2 degrees to make up for possible computational error.

Figure 5.8: Corner partition point.

$$|\Delta\theta_i| > \frac{\pi}{2} - \tau_1 \quad \{i \in C\} \quad (5-6)$$

- (3) *Inflexion point*: it is defined as the point on a contour at which the curvature changes sign. The contour changes from being concave upwards (positive

curvature) to concave downwards (negative curvature), or vice versa, as shown in Fig. 5.9 where the mid-point of l_i is declared as the inflexion point. An inflexion point is detected by formula (5-7). If formula (5-7) is true, the mid-point of l_i is seen as an inflexion point.

$$\Delta\theta_{i-1} \bullet \Delta\theta_i < 0 \quad \{i-1, i \in C\} \quad (5-7)$$

Figure 5.9: Inflexion partition point.

- (4) *Stem point*: it is defined as the endpoint of a line connected to an arc where the line length is longer than the diameter of the arc, as shown in Fig. 5.10. Stem points are extracted only after the previous 3 types of partition points have been extracted to partition a curve into candidate primitives as shown in Fig. 5.6 (b) where there is only one candidate. Among the lines that are expected to approximate an arc candidate, none should be longer than the diameter of the arc. Each line will be examined by condition (5-8). If condition (5-8) is true, both point P_{i-1} and point P_i will be declared as stem points. Here, m , $Curvature(j)$, and l_i follow terminologies used in Fig. 5.6 (b).

$$Length(l_i) > \frac{2m}{\sum_{j=1}^m Curvature(j)} \quad \{i-1, i \in C'\} \quad (5-8)$$

Condition (5-8) is derived from

$$Length(l_i) > Diameter = 2R = \frac{2}{\kappa} = \frac{2}{\frac{1}{m} \sum_{j=1}^m Curvature(j)} = \frac{2m}{\sum_{j=1}^m Curvature(j)}.$$

Figure 5.10: Stem partition point.

For the evaluation of diameter, one can refer to formula (5-4). As for the evaluation of κ , we use the average curvature to estimate the curvature κ of an arc.

Corner point refinement

Due to the adverse conditions (noisy, broken line etc.), the sharp corners sometimes become round corners, and thus corner points detected by formula (5-6) may deviate from the positions where people expect. To cope with this deviation, the end points of the candidate arc will be refined by formula (5-9). P_0 , P_{i-1} , and P_{m+1} have the same meanings as those in Fig. 5.6 (b). The physical meaning behind formula (5-9) is that the circumferential angle, $\angle P_0 P_i P_{m+1}$, of the minor arc (the arc whose central angle is less than 180 degrees) should be an obtuse angle, and the circumferential angle of the major arc (the arc whose central angle is larger than 180 degrees) should be an acute angle as shown in Fig. 5.11 (a). Therefore, the point on the minor arc, whose circumferential angle is acute, and the point on the major arc, whose circumferential angle is obtuse, are

considered as outliers. They will be re-detected as corner points using formula (5-9). They are demonstrated in Fig. 5.11 (b). In Fig. 5.11 (c), the point P_i is a refined partition point if formula (5-9) is true, i.e. the circumferential angle changes from obtuse angle to acute angle, or from acute angle to obtuse angle when tracing the candidate arc.

$$\left(\angle P_0 P_{i-1} P_{m+1} - \frac{\pi}{2} \right) \bullet \left(\angle P_0 P_i P_{m+1} - \frac{\pi}{2} \right) < 0 \quad \{i-1, i \in C'\} \quad (5-9)$$

Figure 5.11: Illustration of partition point refined.

In summary, formula (5-6) is first applied to detect the corner point. Once a corner point is detected, the refinement step, formula (5-9), is applied to the contour between the corner point and its last partition point. The corner point will be relocated to the point that satisfies formula (5-9).

After the continuous contours are partitioned into segments, these segments are classified as arc features and line features. Their properties are evaluated in next section.

5.4 Primitive Evaluation and Classification

Evaluation of arc primitives: Full circles, ellipses and arcs of large span are preferred arc features, since they are relatively definite and stable shape once detected, and usually carry relatively enriched shape information. An arc measure, *Arcoid*, is proposed in this thesis and used to pick up preferred arc features. The *Arcoid* is defined

as in formula (5-12), which is in direct proportion to the arc length. It increases or decreases in the same direction as the arc angle, which is described by curve 2 in the right figure.

There are two reasons that curve 2 is employed instead of curve 1, which represents a linear relationship of direct proportion. First, we want full circles and ellipses always be selected. Second, we don't want to overlook the long arc with very small arc angle.

The exponent α is the weight, which can balance the impact of arc length and arc angle. In this thesis, α is set as 1. That is, arc length and arc angle have same effect on *Arcoid*. The larger the *Arcoid* is, the more desirable the arc feature.

$$SPAN_{ArcAngle} = \sum_{i=1}^m \Delta\theta_i \quad \{i \in C'\} \quad (5-10)$$

$$SPAN_{ArcLength} = \sum_{i=1}^{m+1} Length(l_i) \quad \{i \in C'\} \quad (5-11)$$

$$Arcoid(\alpha) = \frac{SPAN_{ArcLength}}{\left(1 - \frac{SPAN_{ArcAngle}}{2\pi}\right)^\alpha} \quad \alpha \in (0, +\infty) \quad (5-12)$$

Classification of primitives: Given a piece of curve, C' , bounded by two feature points, it is classified as a line or an arc according to the following 2 steps.

- (1) The rule as specified by formula (5-14) is used to determine which segments are desirable arc features. Let us assume that C' in Fig. 5.6 (b) is one candidate primitive extracted from the LEM. C' is classified as an arc feature when the *Arcoid* of the primitive is larger than its *Diagonal* (the diagonal length of the smallest surrounding rectangular of the primitive). The examples are shown in Fig. 5.12, where all the primitives are of the same length. When C' corresponds to a full circle or ellipse, e.g. primitive 1 in Fig. 5.12, $SPAN_{ArcAngle}$ equals to 2π , and *Arcoid* is infinite. Since *Diagonal* is finite, Formula (5-14) is always true in this situation. That means a full circle or ellipse can always be selected as arc features. On the contrary, when C' corresponds to the straight line, e.g. primitive 4 in Fig.

5.12, $SPAN_{ArcAngle}$ equal to 0, and $Arcoid$ is the line length. Thus, the rate of $Arcoid$ to $Diagonal$ will be 1 and Formula (5-14) is false, i.e. the line can never work as arc feature. In addition, two more primitives are also displayed in Fig. 5.12. Primitive 2 is an arc with the central angle of 180 degrees. Primitive 3 is an arc with the central angle of 90 degrees. It can be calculated that their rates of $Arcoid$ to $Diagonal$ are 2.81 and 1.45 respectively. These two values are larger than 1 and therefore make formula (5-14) true. It can be inferred that any arc can make formula (5-14) true since its arc length is always longer than the diagonal. Usually, α in formula (5-14) is set as 1.

$$Diagonal = \sup_{C \subseteq LEM} \left\{ \sqrt{\left(\max_{x_i \in C} (x_i) - \min_{x_i \in C} (x_i) \right)^2 + \left(\max_{y_i \in C} (y_i) - \min_{y_i \in C} (y_i) \right)^2} \right\} \quad (5-13)$$

$$\frac{Arcoid(\alpha)}{Diagonal} > 1 \quad (5-14)$$

Figure 5.12: Illustration of primitive classification.

(2) After the selection of arcs, lines remained to be identified. The feature points in the LEM are adopted as the endpoints of the lines.

5.5 Results

In this section, the method is evaluated. The experimental results on primitive extraction are illustrated in Fig. 5.13 to Fig. 5.17. For simple objects, we mark the partition points. For complex objects, the partition points are not shown to give a cleaner picture of segmented primitives. In the following, the results are commented in two aspects. The first is to observe if suitable primitives are extracted to represent contour segments. The second is to observe if the primitive detection is consistent, especially for those primitives dominating the object's major features. Here, consistency is defined as same or similar contour segments always being detected and classified as the same or similar primitives under all conditions, such as noisy environment, and occlusion. Since the whole system is compared with other methods in chapter 9, the proposed primitive extraction, as a part of the system, are not chosen to compared with others independently in this section.

Fig. 5.13 displays primitives extracted from some simple objects. The first logo is the cinquefoil, its five petals have been detected as arcs and its pistils are detected as lines. The second logo is an occluded bull-head. Its two horns are detected as arcs. The third logo is segmented as two circles and a pentacle composed of lines. The results in this group are encouraging and suitable primitives have been extracted to represent the major features of the objects. In Fig. 5.14, primitives are extracted from some objects similar in their global shapes. All the objects have been mainly partitioned as a half circle and a block.

Figure 5.13: Primitives extracted from simple objects.

(solid: lines, dotted: arcs).

Figure 5.14: Primitives extracted from similar objects.

(solid: lines, dotted: arcs).

In Fig. 5.15, primitives are extracted from some noisy objects. It is observed that the extra lines in the three images have not interfered with the primitive extraction process though the topology property has been affected. In Fig. 5.16, primitives are extracted from some varieties of the Mickey logo. It is found that line and arc have been extracted desirably. The major features, such as ears, mouth, and feet are partitioned as suitable primitives. In Fig. 5.17, primitives are extracted from 3-D stationery object. The results are not as good as those in previous figures because the edge detection degrades slightly

when processing 3-D objects with shadow and reflection. However, the results are still acceptable and sufficient for later use.

Figure 5.15: Primitives extracted from noisy objects.

(solid: lines, dotted: arcs).

Figure 5.16: Primitive extracted from occluded objects.

(solid: lines, dotted: arcs).

Figure 5.17: Primitives extracted from real 3-D object images.

(solid: lines, dotted: arcs).

5.6 Primitives filtering

In this work, undesirable primitives, i.e. the short lines and the redundant primitives, are filtered out prior to shape representation.

Since the relationships between short lines, e.g. the relative angle, are relatively sensitive to noises, short lines are not suitable to be used in the shape modeling and representation process. For example, given two lines of length of 2-pixels, their relative angle will change at least 30 degrees when one endpoint deviates by a pixel. This is undesirable, therefore, short lines with length less than seven pixels are filtered out. The threshold of seven pixels is empirically determined. On the other hand, drawing with thick line style (Fig. 5.18 (a)) usually generates parallel lines (arcs) close to each other (Fig. 5.18 (b)) after the image pre-processing. Such parallel lines (arcs) provide redundant representation of the shape but no distinctive feature for shape modeling. Hence, parallel lines (arcs) close to each other should be merged into one representative primitive. In this work, if the lines (arcs) are parallel, and the space between the lines (arcs) are much less than the line (arc) length, they will be merged. For lines with $\frac{d_1 + d_2 + d_3}{3} < 0.1 \bullet \max(l_1, l_2)$ as shown in Fig. 5.19 (a), we keep the longest one as the representative and the others are eliminated. For parallel arcs, the condition is $d < 0.2r$ as shown in Fig. 5.19 (b). Fig 5.18 (c) displays the result when redundant primitives are filtered out. Compared with Fig 5.18 (b), Fig 5.18 (c) retains a much concise and unique structure.

(a) drawing with thick line style

(b) primitives extracted

(c) primitives filtered

Figure 5.18: Display of primitive filtered.

Figure 5.19: Parameters of the parallel lines / arcs.

5.7 Summary and discussion

A primitive extraction method has been proposed and described in this chapter. Instead of traversing the whole image to detect line / arc, it works on the LEM and extracted lines / arcs from contours. Four types of partition points are defined and two types of primitives, i.e. lines and arcs, are detected. The process can highlight major shape components for classification. The results show that it is difficult to handle the gray area between the two types of primitives, i.e. curve segments that can be classified as either a line or an arc. For example, in Fig. 5.16, the foreheads of the three Mickey figures are not always detected as the same/similar primitives. This problem is tackled by a selection process in later higher level consideration to select the best rather than all primitives.

Even though we have presented a practical primitive detection method in this chapter, there is no special requirement for the primitive extraction used in this thesis. Any other methods for arc / line detection can be employed here.

Chapter 6 Shape Representation

Shape representation is important for image retrieval. The shape of objects is usually strongly linked to object functionality and identity, and people can recognize objects only from their shape. It makes shape different from other visual feature, such as color and texture. Without powerful shape descriptor, image cannot be retrieved effectively and efficiently. Many good approaches have been proposed [1-5] for shape-based retrieval. However, there is a lack of effective approach to tackle partially occluded and cluttered objects found in an input image. A local structure based method is proposed to cope with the difficulties since local method is more suitable for such problems than global method. Psychology studies [145, 146] indicate that a shape can be viewed as a hierarchical structure, which is formed around one or several centers. Salient structures can capture the essential information of an object [147]. Therefore, local structures of the proposed method are constructed by grouping several primitives around one reference primitive from an object. The local structure is salient and can be extracted consistently. The inherent structural relationship in one structure will be extracted to build a unique, robust, and invariant representation. The proposed approach employs local structure information only and does not require segmentation (separation) of input objects from complex background. It is also able to tackle retrieval of structurally similar objects from a large image database. In this chapter, the shape

descriptors are overviewed first. Then, the proposed shape representation will be presented in three steps: local structure construction, local structure representation and incorporation.

6.1 Overview of local shape description techniques

Shape description techniques can be classified according to different criteria: *global* methods [1, 2, 11] and *local* methods [3-6, 12-18] according to whether the shape is represented as a whole or represented by local shape features (*primitives*). Global features are usually easy to compute and robust against noise and shape distortions. Local methods are complicated requiring sophisticated implementations; however, they are more suitable than global methods for occluded or partial visible object retrieval. Local shape based representations have been used for model-based recognition for a long time and shape-based retrieval can be viewed as a model-based recognition problem except that the number of models is very large and no verification is needed. However, not all local object recognition approaches can be extended to apply to image retrieval, especially those that do not support shape indexing and cannot handle the problem of a large number of models. For example, the traditional chain coding [70] is inefficient since it has a long list of code; the medial axis transform [75] is sensitive to boundary noise and variations; the graph representation [77, 79] is too complex to be indexed. Some other local methods are found more suitable for image retrieval. In [3], the curvature zero-crossing points from a Gaussian smoothed boundary are used to obtain primitives, i.e. tokens. The feature representation is scale invariant

but not rotation invariant. All the tokens are indexed into the feature database by an M-tree. Its major drawback is the assumption that the occlusion boundaries have been identified, and segments belonging to the contours of single objects are available as open shapes. In [5], constrained morphological decomposition is performed on a given object, and then an iterative merging stage provides a compact graph based representation. In [4], shape contexts are used for finding correspondence between shapes. In [148], a shape axis tree is used to represent the object. In [149], a sub-matrix matching is proposed to achieve good partial shape recognition. It is a template based method and works under controlled environment. Unfortunately, all these methods rely on good segmentation results to provide the needed shape information prior to retrieval. Huet and Hancock [6] extracted line segments from an image as primitives. An N -nearest neighbor graph was used as the shape representation. The Bhattacharyya distance worked as the similarity measure between two histograms. This is a local structure approach and impressive results were obtained. However, their approach has a few shortcomings. First, the unique property of a local structure loses its identity once it is inserted into the histogram for all local structures. In addition, their approach cannot avoid the inaccuracy of polygonal approximation on round objects, and it has no mechanism to handle the problem of *dimensional curse*. The proposed technique here tackles all of these concerns.

6.2 Local Structure Construction

In this work, the local structure consists of one reference primitive, and several

neighbors around the reference primitives. Based on human's perception of significant shape features, the Gestalt psychology laws [150] are employed to guide the searching for salient local structures. The geometrical organizations grouped by the Gestalt laws have been proved to possess the best, simplest and most stable shape in adverse environments than those accidental arrangements [150]. This satisfies our need to identify the same local neighborhood (i.e. get consistent neighbors of a feature) from the inputs and the corresponding models.

6.2.1 Gestalt Laws for Grouping

The Gestalt theory is a family of psychological theories, and was generally accepted in perceptual grouping [151]. Gestalt theories are usually expressed as laws. Five laws from the Gestalt theory [152, 153] are namely the law of focal point, proximity, continuity, similarity and symmetry. They are adopted here to determine a local structure. Fig. 6.1 shows the five laws.

- (1) **Law of Focal Point.** Every visual presentation needs a focal point, called the centre of interest or point of emphasis. This focal point catches the viewer's attention and persuades the viewer to follow the visual message further [154]. Fig. 6.1 (a) demonstrates that a different shape (circle) appears to protrude out from other shapes (line) and draws attention. In our work, this law is adapted to focus on long line or arc with high *Arcoïd*. It is because long line and stable arc tend to capture one's attention easily. Around each focus, one local structure will be formed.

Figure 6.1: Illustration of the five Gestalt Laws.

- (2) **Law of Proximity.** It states that objects near each other tend to be seen as a unit [152]. Viewers will mentally organize closer elements into a coherent object, because they assume that closely spaced elements are related and those further apart are unrelated. In Fig. 6.1 (b), one tends to mentally arrange the lines into three pairs and one single. In our work, the primitives placed near each other are preferred to appear in one local structure. Therefore, the primitives near the focused reference tend to be selected as its neighbors.
- (3) **Law of Continuity.** It states that objects arranged in either a straight line or a smooth curve tends to be seen as a unit. Continuity is the eye's instinctive action

to follow a direction derived from the visual field. For example, in Fig. 6.1 (c), our eyes follow the straight line from left to right. In our work, the continuous primitives tend to be grouped in one local structure. That is, the primitives connected with the focused reference are its preferred neighbors.

(4) **Law of Similarity**. It states that items that are similar in some way tend to be grouped together. According to Fisher and Smith-Gratto [152], similar objects will be seen as from the same group and this technique can be used to draw a viewer's attention. In Fig. 6.1 (d), the viewer can recognize two pairs of long lines among three pairs of short lines. In our work, similar primitives are likely to be grouped in the same local structure, i.e. primitives similar to the focused reference are preferred to be its neighbors.

(5) **Law of Symmetry**. It states that items are organized into figures according to symmetry. A visual object will appear as incomplete if the visual object is not balanced or symmetrical [152]. In Fig. 6.1 (e), one tends to extract the symmetry relationships of the two diagonal lines, and two pairs of vertical lines. In our work, the symmetric primitives tend to be grouped in one local structure, i.e. the reference would like to choose its symmetric primitives as the neighbors.

In the following section, we will present how these five laws are used to construct a local structure, i.e. the selection of the reference primitives and the neighbors.

6.2.2 Reference Feature Selection

According to the law of focal point, the top $\xi\%$ of longest lines, and top $\xi\%$ of arcs

with the highest *Arcoïd* are selected as reference features. This is due to two reasons. The first is that long lines and arcs with high *Arcoïd* tend to catch view's attention more easily than those short ones. The second one is that the information they carry are consistent and abundant. It is further explained by means of Fig. 6.2.

For the image in Fig. 6.2 (a), different percentages of primitive features are selected and shown in Fig. 6.2 (b)-(f). It can be observed from the results that 20% of primitives can describe the image well and highlight the important shape information. This indicates that the long lines and arcs with high *Arcoïd* carry more shape information than short lines and arcs with low *Arcoïd*. Long lines and arcs with high *Arcoïd* should be the primitives of emphasis; therefore, one can refer to them when extracting local information. These primitives are focal centers at which the salient and consistent local structures can be investigated further. Finally, the object can be described by local structures.

There are exceptions as shown in Fig. 6.3. The object of interest in Fig. 6.3 (a) is surrounded by a rectangle which will be allocated unnecessary emphasis by the previous rules. However, it does not reduce the effectiveness of the proposed method since we choose the reference primitives by percentage. Enough information from the object is definitely retained if a suitable percentage is set. Otherwise, if the object is even simpler than the frame as shown in Fig. 6.3 (f), it is reasonable to have one's attention drawn to the frame from the object. In Fig. 6.3 (a) ~ (j), we illustrated the top 20%, 40%, 60% and 80% primitives respectively. It is found that 20% is not yet enough for object description, while the top 60% primitives provide sufficient shape

information of the object. To retain enough shape information for all kind of images, all the primitives are selected as primitives in this work to explore the optimal performance.

Figure 6.2: Examples of various percentages of the top reference features.

(Solid: lines, dotted: arcs).

Figure 6.3: Reference primitives extracted from images with frame.

6.2.3 Local Neighbor Selection

Around one reference feature, a certain numbers of neighbors are selected to construct a local structure. There are two factors in determining local neighbors. The first one is the dissimilarity measure, which evaluates the suitability of each candidate neighbor. The second one is the number of neighbors in a local structure. This determines the robustness of the local structure. In the following paragraphs, we first present the proposed dissimilarity measure in details, and then a statistical experiment is carried out to establish the empirical size of a local structure.

6.2.3.1 Dissimilarity Measure

Dissimilarity is measured by a combination of the laws of proximity (D_{pr}), continuity (D_{co}), similarity (D_{si}), and symmetry (D_{sy}). It is applied to determine N suitable neighbors with the lowest dissimilarities. Each dissimilarity is captured using formula (6-1), where D_{pr} , D_{co} , D_{si} , D_{sy} are the differences of proximity, continuity, similarity and symmetry, respectively.

$$Dissimilarity = D_{pr} * D_{co} * (D_{si} + D_{sy}) \quad (6-1)$$

The approximate ranges of these differences are $(1 \leq D_{pr} \leq M)$, $(D_{co} = 0 \text{ or } 1)$, and $(0 \leq D_{si}, D_{sy} \leq M)$ where M is the image size as $\max(\text{imageHeight}, \text{imageWidth})$. Their definitions are described in detail below. D_{co} takes the value of 0 or 1 for neighbor that is directly connected to the reference or not. We prefer connected neighbors. Hence, a natural combination of the differences is $D_{co} * (D_{pr} + D_{si} + D_{sy})$.

Unfortunately, each term inside the brackets can dominate the other if it has a large value. Multiplication, on the other hand, would allow equal influence from each term. In addition, multiplication will give a larger range and consequently with less number of neighbor candidates occupying the lower end (good for picking candidates from this end). We do not want to multiply D_{si} with D_{sy} because D_{sy} is zero in most cases.

The definition and detail of each of the components, D_{pr} , D_{co} , D_{si} and D_{sy} are described in the following paragraphs while the rationale in combining the components in (6-1) will be explained in next section.

Definitions of D_{pr} , D_{co} , D_{si} , and D_{sy}

Three different types of primitive relationships are considered here. They are

1. *Line-to-line*: from one reference line to its line neighbor.
2. *Arc-to-arc*: from one reference arc to its arc neighbor.
3. *Arc-to-line*: from one reference arc to its line neighbor.

Fig. 6.4 illustrates the relationships. The corresponding terminology is

- p_i^m : the midpoint of line i .
- c_j : the circle center when a circle is fitted to the arc j .
- r_j : the radius of the fitted circle.
- Φ_j : the central angle of arc j .
- h_j : the height of arc j .
- $\bar{L}_i$: the length of line i .
- $\hat{L}_j$: the length of arc j .
- d : the shortest end point gap.

(a) Line-to-line

(b) Arc-to-arc

(c) Arc-to-line

Figure 6.4: Examples of dissimilarity measures.

(The primitives, distances and auxiliary drawings are drawn in thick, normal and dotted line styles respectively.)

We define D_{pr} , D_{co} , D_{si} , and D_{sy} as:

- D_{pr} computes the proximity difference from p_i^m to p_j^m , from c_i to c_j , and from c_i to p_j^m when the relationships of line-to-line, arc-to-arc, and arc-to-line are considered respectively. The circle center is selected to calculate D_{pr} because of its robustness to broken arc and distorted arc.

- D_{co} is the continuity difference computed as:

$$D_{co} = \begin{cases} 0 & \text{if } d \leq 2 \text{ pixels} \\ 1 & \text{if } d > 2 \text{ pixels} \end{cases} \quad (6-2)$$

where “ d ” is the shortest distance between any 2 end points of primitive i and j . If one of the primitives is a full circle or a close loop, e.g. primitive i , “ d ” is defined as the shortest distance between any point on primitive i and any end point of primitive j . If both are full circles, “ d ” is defined as the shortest distance between any point on primitive i and any point on primitive j . The interpretation of D_{co} is that it prefers a connecting neighbor and a value 0 will be assigned to D_{co} to grab such neighbor.

- D_{si} is the similarity difference computed as:

$$D_{si} = \begin{cases} |\bar{L}_i - \bar{L}_j| & \text{Line-to-line} \\ 2|r_i - r_j| + \min(r_i, r_j)|\Phi_i - \Phi_j| & \text{Arc-to-arc} \\ |\bar{L}_i - \bar{L}_j| + r_i \left(1 - \cos \frac{\Phi_i}{2}\right) & \text{Arc-to-line} \end{cases} \quad (6-3)$$

It is the difference of lengths for the case of *line-to-line*. For the case of *arc-to-arc*, it has 2 parts. The first part is the difference of circle diameters, while the second part is the difference of center angles. In order to sum them together, the second part, $|\Phi_i - \Phi_j|$, is multiplied by the radius to get an arc length. D_{si} also has two parts for the case of *arc-to-line*. The first part is the difference of lengths while the second part is the height of arc.

- D_{sy} computes the symmetry difference of two lines only. Let these 2 lines be

$\overline{AB}$ and $\overline{CD}$ as shown in Fig. 6.5 with the midpoints O_1 and O_2 . When $\overline{L}_{AB} = \overline{L}_{CD}$ and $\theta_1 + \theta_2 = \pi$, then $\overline{AB}$ and $\overline{CD}$ (the solid lines) are axial symmetric; if $\overline{L}_{AB} = \overline{L}_{CD}$ and $\theta_1 = \theta_2$, then $\overline{AB}$ and $\overline{CD}$ (the dot lines) are central symmetric. D_{sy} is defined according to these two symmetries as:

$$D_{sy} = \begin{cases} \left(\frac{\overline{L}_{AB} + \overline{L}_{CD}}{2} \right) \min(|\theta_1 - \theta_2|, |\theta_1 + \theta_2 - \pi|) & \text{Line-to-line} \\ 0 & \text{Arc-to-arc} \\ 0 & \text{Arc-to-line} \end{cases} \quad (6-4)$$

where $0.5(\overline{L}_{AB} + \overline{L}_{CD})$ is the average length of line AB and CD. The unit of θ_1 , and θ_2 is radian. The D_{sy} between two arcs is not considered, since the result improvement cannot make up for the computation load.

Figure 6.5: Symmetry angles computation.

O_1 and O_2 are the midpoints of lines AB and CD .

6.2.3.2 The choices of dissimilarity measures

In this section, the *Dissimilarities* computed using solely D_{pr} , using weighted sum of D_{pr} , D_{co} , D_{si} , and D_{sy} , and using multiplication of D_{pr} , D_{co} , D_{si} , and D_{sy} are analyzed theoretically by means of probability distribution. In the analysis, we generalize the factors mentioned above as A or B for simplifying analysis. The

multiplication is proved having the best distinctive power.

Initially, the *Dissimilarity* is computed using formula (6-5) (the same as the 6-nearest-neighbor method [73], for which one can refer to Chapter 9).

$$Dissimilarity = D_{pr} \quad (6-5)$$

Conceptually, better distinctive measure can be obtained if more features of a neighbor can be used. One example is illustrated below with two features A and B with independent uniform probability distributions as shown in Fig. 6.6(a) and (b). We assume $m_1 < m_2$. When A and B are combined as “Dissimilarity = A + B”, the new probability distribution is displayed in Fig. 6.6(c) which is computed as

$$f_{A+B}(x) = \int_{-\infty}^{\infty} f_A(z) f_B(x-z) dz = \int_0^x f_A(z) f_B(x-z) dz \quad m_2 > m_1, x \geq z$$

$$= \begin{cases} \int_0^x \frac{1}{m_1} * \frac{1}{m_2} dz = \frac{1}{m_1 m_2} x, & 0 \leq x < m_1 \\ \int_0^{m_1} \frac{1}{m_1} * \frac{1}{m_2} dz = \frac{1}{m_2}, & m_1 \leq x < m_2 \\ \int_{x-m_2}^{m_1} \frac{1}{m_1} * \frac{1}{m_2} dz = \frac{m_1 + m_2 - x}{m_1 m_2}, & m_2 \leq x \leq m_1 + m_2 \end{cases}$$

Here, the integral intervals are determined in the following.

$$\begin{cases} 0 \leq z \leq m_1 \\ 0 \leq x - z \leq m_2 \end{cases} \Rightarrow \begin{cases} 0 \leq z \leq m_1 \\ x - m_2 \leq z \leq x \end{cases}$$

$$\Rightarrow \max(0, x - m_2) \leq z \leq \min(m_1, x)$$

$$\Rightarrow \begin{cases} 0 \leq z \leq x & \text{when } 0 \leq x < m_1 \\ 0 \leq z \leq m_1 & \text{when } m_1 \leq x < m_2 \\ x - m_2 \leq z \leq m_1 & \text{when } m_2 \leq x \leq m_1 + m_2 \end{cases}$$

Figure 6.6: The probability distribution.

$$F_{A+B}(x) = P(y+z < x) = \iint_{y+z < x} f_A(y)f_B(z)dydz = \int_{-\infty}^x f_{A+B}(u)du$$

$$= \begin{cases} \int_0^x f_{A+B}(u)du = \frac{x^2}{2m_1m_2}, & 0 \leq x < m_1 \\ \int_0^{m_1} f_{A+B}(u)du + \int_{m_1}^x f_{A+B}(u)du = \frac{x}{m_2} - \frac{m_1}{2m_2}, & m_1 \leq x < m_2 \\ \int_0^{m_1} f_{A+B}(u)du + \int_{m_1}^{m_2} f_{A+B}(u)du + \int_{m_2}^x f_{A+B}(u)du \\ \quad = \frac{2m_1m_2 - (x - m_1 - m_2)^2}{2m_1m_2}, & m_2 \leq x \leq m_1 + m_2 \end{cases}$$

This new distribution has wider spread and lower probability at each dissimilarity value. Hence, it has better distinctive measure since less number of neighbor candidates will occupy the lower end. In this work, we have extracted three more new features as D_{co} , D_{si} , and D_{sy} with D_{co} taking the discrete value of 0 or 1 for neighbor that is directly connected to the reference or not. A better dissimilarity combination of the features is

$$Dissimilarity = D_{co} * (D_{pr} + D_{si} + D_{sy}) \quad (6-6)$$

However, we prefer multiplications since the measure will have an even larger range and consequently better distinctive power than formula (6-6). Another example is illustrated here with features A and B again. When A and B are combined as “Dissimilarity = A × B”, the new probability distribution is displayed in Fig. 6.6 (d) which is computed as

$$F_{A \times B}(x) = P(yz < x) = \iint_{yz < x} f_A(y)f_B(z)dydz$$

$$= \int_0^{m_1} \left[\int_0^{z(y)} f_A(y)f_B(z)dz \right] dy$$

$$= \int_0^{m_1} \int_0^{z(y)} f_A(y)f_B(z)dzdy$$

$$= \int_0^{m_1} f_A(y)dy \int_0^{z(y)} f_B(z)dz$$

$$= \int_0^{x/m_2} f_A(y)dy \int_0^{m_2} f_B(z)dz + \int_{x/m_2}^{m_1} f_A(y)dy \int_0^{x/y} f_B(z)dz$$

$$= \frac{1}{m_1 m_2} \left(x + x \ln \left(\frac{m_1 m_2}{x} \right) \right), \quad 0 < x \leq m_1 m_2.$$

$$f_{A \times B}(x) = \frac{dF_{A \times B}(x)}{dx} = \frac{\ln(m_1 m_2) - \ln x}{m_1 m_2}, \quad 0 < x \leq m_1 m_2.$$

The probability distribution in Fig. 6.6 (d) is a decreasing sequence. The distribution is plotted starting from $x = 1$, since our system works on integer value of x only, and the value at $x = 0$ occur rarely in reality. The distribution is not superior to (i.e. lower than) Fig. 6.6 (c) near the origin. However, the values at $x = 1$ and $x = 2$ are quite low as $\frac{\ln(m_1 m_2)}{m_1 m_2}$, and $\frac{\ln(m_1 m_2) - \ln 2}{m_1 m_2}$. We concern the sum of the probabilities, i.e. $F(x)$, with dissimilarity value less than a threshold. The smaller $F(x)$ value means the less candidate neighbors occupying the lower end. The $F_B(x)$, $F_{A+B}(x)$, and $F_{A \times B}(x)$ are therefore plotted in Fig. 6.6 (e). $F_B(x)$ is larger than $F_{A+B}(x)$ and $F_{A \times B}(x)$ at all its dissimilarity values. (τ, ρ) is the intersection point of $F_{A+B}(x)$ and $F_{A \times B}(x)$. At the lowest end, $F_{A+B}(x)$ is smaller (but not much smaller) than $F_{A \times B}(x)$. However, $F_{A+B}(x)$ grows faster than $F_{A \times B}(x)$ when the dissimilarity threshold value start to increase. When the dissimilarity value exceeding τ , $F_{A+B}(x)$

becomes larger than $F_{A \times B}(x)$. Usually, ρ is of a small value, e.g. ρ is 0.026 when m_1 is set as 30 and m_2 is set as 100. Furthermore, multiplication can provide a more stable measure. The changes due to noise or segmentation errors are lower in general. One example is given below. Let the dissimilarities be $D_1 = A + B$ and $D_2 = A \bullet B$ respectively. The changes due to noise, distortion and segmentation errors can be

$$dD_1 = dA + dB,$$

$$dD_2 = AdB + BdA.$$

The changes with respect to the ranges are

$$\frac{dD_1}{m_1 + m_2} = \frac{dA + dB}{m_1 + m_2},$$

$$\frac{dD_2}{m_1 m_2} = \frac{AdB + BdA}{m_1 m_2}.$$

The comparison of changes is

$$\begin{aligned} & \frac{dD_1}{m_1 + m_2} - \frac{dD_2}{m_1 m_2} \\ &= \frac{m_1 m_2 (dA + dB) - (m_1 + m_2) (AdB + BdA)}{m_1 m_2 (m_1 + m_2)} \\ &= \frac{(m_1 m_2 - (m_1 + m_2) B) dA + (m_1 m_2 - (m_1 + m_2) A) dB}{m_1 m_2 (m_1 + m_2)} \\ &= \frac{\left(m_1 \left(\frac{m_2}{2} - B \right) + m_2 \left(\frac{m_1}{2} - B \right) \right) dA + \left(m_1 \left(\frac{m_2}{2} - A \right) + m_2 \left(\frac{m_1}{2} - A \right) \right) dB}{m_1 m_2 (m_1 + m_2)} \end{aligned}$$

Usually, $m_1 \gg A$, $m_1 \gg B$, $m_2 \gg A$, $m_2 \gg B$, since the system chooses the features with the lowest $A + B$ or $A \bullet B$.

$$\text{Hence, } \frac{dD_1}{m_1 + m_2} - \frac{dD_2}{m_1 m_2} > 0.$$

In conclusion, when small changes occur, multiplication can result in the smallest deviation of dissimilarity due to wider spread of the probability distribution. Thus, the same neighbor has higher probability to be selected consistently. Therefore, we use formula (6-1) rather than formula (6-5) or (6-6) as dissimilarity measure in this work.

6.2.3.3 Size of the local structure

We compute N neighbors instead of just one. With N neighbors, the probability that one common neighbor can be extracted from both the input object and its corresponding model is relatively high, given that the same reference feature is considered and the *Dissimilarity* measure tends to choose similar types of neighbors from similar local structures. On the other hand, less (or no) common neighbors are likely to be found if two different reference features are considered. With this concept, salient and unique local neighborhood structures can be constructed from a reference feature and a fraction of its best neighbors with the lowest dissimilarities. For example, if we compute 4 best neighbors of a reference feature, there will be 6 unique structures that can be constructed from the reference feature and any 2 of these 4 neighbors (C_4^2 combinations). However, it is difficult for an imposter to find 2 out of its 4 best neighbors to compose even one similar structure to claim kinship. Nevertheless, it is easy for the genuine neighborhood to find one suitable combination even in situations when objects are partially occluded or merged together, provided that enough or distinguishable portion of the object can be observed as discussed in the introduction

chapter.

The local structure, constructed by one reference primitive and several neighbors, is capable of grouping perceptually unique structures of the object. However, it is difficult to tell how many neighbors will be suitable for a structure conceptually. Generally, too few neighbors are not desirable, it is not only because the provided information will be insufficient to build unique structures, but also because too few neighbors are difficult to build robust structures. However, it is also not true that more neighbors can build a structure that is more robust. When the number of neighbors greatly exceeds the adaptive size of a structure, there will be more random arrangements included; thus, the selected structure is no longer stable and unique. The robustness is definitely impaired. Therefore, to select a uniform size for all structures, we have to decide on the number of neighbors by evaluating the average system performance under varying numbers of neighbors. To evaluate the system performance, we select all primitives as reference features, and the histogram of retrieval accuracy versus the number of neighbors is plotted in Fig. 6.7. The accuracy was measured in similarity based queries by “Bull’s Eye Percentage Score” (BEP) [35]. BEP is defined for each query as the percentage of correct matches (i.e. items within the same class as the query) among the $2C$ top retrieved results, in which C denotes the size of the considered query class. From the results illustrated in Fig. 6.7, four neighbors got the highest accuracy; therefore, four neighbors are selected in a local structure in this work. That is, four sets of pair-wise geometric attributes relative to each reference feature are computed to represent one local structure. Such a local structure representation is for

the balance between efficiency and effectiveness. As for the test set for similarity query, one can refer to Chapter 9 for more details.

Figure 6.7: Accuracy test with respect to different number of neighbors in a local structure.

6.2.4 Results

In this section, the experiment results on local structure extraction are illustrated in Fig. 6.8 ~ Fig. 6.12. Since the proposed method extracted unique local structures by means of line / arc primitives, local structures constructed here are required to be relatively consistent and salient under all conditions, especially for those unique structures of the object. Therefore, the results were evaluated by analyzing whether they have met such requirement.

Figure 6.8: Typical local structures of simple objects
(R- Reference primitive).

In Fig. 6.8, local structures are extracted from some simple objects. The results are used to illustrate that the extracted local structure is able to reflect important image

structural feature. Fig. 6.8 (a) is the cinquefoil. Its five petals are grouped as one structure, which is desirable due to extraction of major feature of the figure. Fig. 6.8 (b) is a graph consist of several bars. The global shape of the graph is one rectangular. The extracted local structure is favorable for highlighting the global shape. Fig. 6.8 (c) was an occluded bull-head. The extracted local structure consists of the two horns, which are commonly regards as the sign of the bullhead. These results manifest that the important and compatible structural features in simple objects are extracted relatively correctly.

In Fig. 6.9, local structures are extracted from some similar objects and occluded objects. The results are used to demonstrate that similar local structures can be extracted from similar objects, if the reference primitives are similar. The objects in Fig. 6.9 are different in trivial details, but similar in global shape. Fig. 6.9 (a) is different from the others in the caption and legend. Fig. 6.9 (c) is similar to the others only in global shape. Fig. 6.9 (d) is an occluded image, which is the common part of the first and second images. From the results, it is found that the general shape of all the objects, a half circle and a block, is extracted to construct a local structure. The results manifest the proposed method having capability to extract similar structure information from similar objects.

Figure 6.9: Typical local structures of similar / occluded objects
(R- Reference primitive).

In Fig. 6.10, local structures are extracted from the object, which are corrupted by extra lines, combined with the other object and distorted by pasting on a sphere randomly. The results are used to illustrate that the important structure of the object can be extracted consistently even under adverse conditions. Most of the extracted local structures include the shape of “Λ”, which is important structure information of the object. The only exception is Fig. 6.10 (c), in which the letter A is combined with the letter B on the left. The left edge of A is therefore corrupted. Hence, we choose the corresponding right edge as the reference primitive to construct the local structure. It is observed that similar neighborhood relationship is extracted. The result indicates that the symmetry of an object can compensate for the object deficiency of one symmetrical edge's corruption to some extent. From the results, it is observed that similar local structure from one object is extracted in all conditions. It substantiates that the proposed local structure construction because the geometrical organizations grouped by Gestalt laws possess the best, simplest and most stable shape under adverse environments than those accidental arrangements.

Figure 6.10: Typical local structures of noisy / distorted objects

(R- Reference primitive).

In Fig. 6.11, local structures are extracted from objects in cluttered environment.

The combined image (Fig. 6.11(b)) has one fish and one letter overlaying on top of a bee figure. The results are used to demonstrate that the proposed method selects stable neighbors with or without the background. Fig. 6.11 (a) is a bee figure. A local structure consisting of the wings and head is extracted. Around the same reference primitive as that from the first example, the same local structure has been extracted from the combined image (Fig. 6.11 (b)). Fig. 6.11 (c) is a fish, and the local structure around its eye is displayed. Similarly, we extract a local structure around the eye from the combined image (Fig. 6.11 (d)). It is found that the extracted local structure has three common neighbors with Fig. 6.11 (c). The two pairs of results indicate the proposed method has the potentiality to extract consistent local structures under cluttered environment.

In figure 6.12, local structures were extracted from 3-D stationery objects. The results are used to illustrate that the proposed method can extract meaningful parts of the object to compose one local structure consistently. Fig. 6.12 displays the tapes from different viewing angles. The tapes are fastened on the brackets. In Fig. 6.12 (a), the extracted local structure reflects the relationship between the tape and its bracket. In Fig. 6.12 (b) and (c), the extracted local structures have two common neighbors with Fig. 6.12 (a). From the results, it is observed that the general shape description of the objects is relatively acceptable: the tape is described as an arc and the brace is described as a pair of parallel lines even though the arcs or lines are not exactly same in all the objects. Similar local structures were extracted consistently from the images of an object captured from different view angles.

Figure 6.11: Typical local structures of mixed objects
(R- Reference primitive).

Figure 6.12: Typical local structures of real 3D object images
(R- Reference primitive).

6.3 Invariant Local Structure Representation

After local structure construction, the inherent structural relationships of the local structures can be extracted for a compact and numerical representation. The extracted numbers will be used for indexing and retrieval later. Pair-wise geometric relationships are employed to represent the local structure, i.e. one local structure is represented as a set of pair-wise geometric attributes between the reference and neighbors. A set of four geometric relationships are proposed to represent a pair of reference and neighbor primitives. The geometric relationships are simple, invariant to translation / scaling / rotation, relatively robust to fragmentation / end-point-erosion, and sufficient for discrimination. Three types of attributes are considered in this work for line-to-line, arc-to-arc, and arc-to-line relationships.

6.3.1 Line to line pair-wise geometric attributes

A vector of four Euclidean invariant attributes (refer to Fig. 6.13 (a)),

$$GA_{LL} = \left[\frac{\pi + \theta_1}{2\pi}, \frac{\pi + \theta_2}{2\pi}, \frac{\pi + \theta_3}{2\pi}, \frac{d}{|AB|} \right]^T$$

is computed to represent the relationship between a pair of line segments. These four attributes can be used to reconstruct the pair of line segments at a normalized scale and orientation.

(a) Relationships of 2 lines.

(b) Relationships of 2 arcs.

(c) Relationships of a line and arc.

Figure 6.13: Pair-wise Geometric Attributes.

Let the line segments $\overline{AB}$ and $\overline{CD}$ be represented by the vector $\overrightarrow{AB}$ and $\overrightarrow{CD}$. M is the midpoint of $\overline{AB}$. The vectors are pointing away from their intersect point. The relative orientation attribute θ_1 (the smallest intersecting angle) is given by formula (6-11).

$$\theta_1 = \pm \arccos \left[\frac{|\overrightarrow{AB} \bullet \overrightarrow{CD}|}{\|\overrightarrow{AB}\| \|\overrightarrow{CD}\|} \right] \quad (6-11)$$

A positive sign is given to the relative angle θ_1 if the direction of the angle from the reference line $\overrightarrow{AB}$ to its neighbor $\overrightarrow{CD}$ is clockwise, and a negative sign is given if the direction is counterclockwise. The range of angle θ_1 is $[-\pi, \pi]$. Similarly, θ_2 and θ_3 can be defined as

$$\theta_2 = \pm \arccos \left[\frac{\overrightarrow{AB} \bullet \overrightarrow{MC}}{|\overrightarrow{AB}| |\overrightarrow{MC}|} \right] \quad (6-12)$$

$$\theta_3 = \pm \arccos \left[\frac{\overrightarrow{AB} \bullet \overrightarrow{MD}}{|\overrightarrow{AB}| |\overrightarrow{MD}|} \right] \quad (6-13)$$

The distance, d , is

$$d = \frac{|\overrightarrow{MD}|}{|\overrightarrow{AB}|} \sqrt{1 - \left(\frac{\overrightarrow{CD} \bullet \overrightarrow{MD}}{|\overrightarrow{CD}| |\overrightarrow{MD}|} \right)^2} \quad (6-14)$$

The attributes θ_2 , θ_3 , and d are the measures of relative positions between reference line AB and its neighbor CD . Vectors $\overrightarrow{MC}$ and $\overrightarrow{MD}$ denote the line segments MC and MD that direct away from the middle point M of the reference line segment to the two endpoints C and D of its neighbor. θ_2 is the angle between $\overrightarrow{AB}$ and $\overrightarrow{MC}$, and θ_3 is the angle between $\overrightarrow{AB}$ and $\overrightarrow{MD}$. The range of angle θ_2 , θ_3 is $[-\pi, \pi]$. The attribute d is the normalized distance between the middle point M of the reference line segment and its neighbor vector $\overrightarrow{CD}$.

6.3.2 Arc to arc pair-wise geometric attributes

Similarly, a vector of four Euclidean invariant attributes

$$GA_{cc} = \left[\frac{\Phi_1}{2\pi}, \frac{\Phi_2}{2\pi}, \frac{|r_1|}{|r_1| + |r_2|}, \frac{1}{1 + |O_1O_2|/(|r_1| + |r_2|)} \right]^T \text{ can be computed to describe a pair}$$

of arcs (see Fig. 6.13 (b)), arc-AB and arc-CD where Φ_1 and Φ_2 are the central angles.

For the 4th attribute, the term $|O_1O_2| / (|r_1| + |r_2|)$ determines whether two circles fitted to the arcs intersect. It is normalized here to the range (0, 1]. The circles will have two common points (when they intersect), one common point (when they touch), or no common point (when they are separate) according to whether the computed value is larger than 0.5, equal to 0.5 or less than 0.5, respectively, as illustrated in Fig. 6.14.

(a) No intersection point (b) One intersection point (c) Two intersection point

Figure 6.14: Relationships between two circles.

6.3.3 Arc-line pair-wise geometric attributes

According to Fig. 6.13 (c), the arc-line pair-wise geometric attributes are computed

$$\text{as } GA_{CL} = \left[\frac{\Phi_1}{2\pi}, \frac{\pi + \theta_1}{2\pi}, \frac{\pi + \theta_2}{2\pi}, \frac{1}{1 + d/|r_1|} \right]^T. \text{ If arc AB is a closed curve, } \vec{r_1} \text{ is set to}$$

have the same direction as CD. The 4th attribute is used to determine whether arc AB intersects line CD. They intersect, touch, or not intersect when the value is larger than 0.5, equal to 0.5, or less than 0.5, respectively.

Let the line segments $\overline{r_1}$ and $\overline{CD}$ be represented by the vector $\vec{r_1}$ and $\vec{CD}$. The vectors are pointing away from their intersection point. The relative orientation attribute θ_1 (the smallest intersecting angle) is given by formula (6-15).

$$\theta_1 = \pm \arccos \left[\frac{\vec{r}_1 \bullet \overrightarrow{CD}}{|\vec{r}_1| |\overrightarrow{CD}|} \right] \quad (6-15)$$

A positive sign is given to the relative angle θ_1 if the direction of the angle from the reference line $\vec{r}_1$ to its neighbor $\overrightarrow{CD}$ is clockwise, and a negative sign is given if the direction is counterclockwise. The range of angle θ_1 is $[-\pi, \pi]$.

Similarly, θ_2 can be defined as

$$\theta_2 = \pm \arccos \left[\frac{\vec{r}_1 \bullet \overrightarrow{O_1M}}{|\vec{r}_1| |\overrightarrow{O_1M}|} \right] \quad (6-16)$$

The distance, d , is

$$d = \frac{|\overrightarrow{O_1M}|}{|\vec{r}_1|} \sqrt{1 - \left(\frac{\overrightarrow{MD} \bullet \overrightarrow{O_1M}}{|\overrightarrow{MD}| |\overrightarrow{O_1M}|} \right)^2} \quad (6-17)$$

The attributes θ_2 , and d are the measures of relative positions between reference arc AB and its line neighbor CD. Vector $\overrightarrow{O_1M}$ denotes the line segment which directs away from the centre point O_1 of the reference arc to the midpoint M of its line neighbor. θ_2 is the angle between $\vec{r}_1$ and $\overrightarrow{O_1M}$. The range of angle θ_2 is $[-\pi, \pi]$. The attribute d is the normalized distance between the centre point O_1 of reference arc and its neighbor line $\overrightarrow{CD}$.

6.4 Object Representation

All the pair-wise geometric attributes of an object are aggregated together to form one 5-dimensional (5D) hypercube representation, H . The first dimensions of H marks the geometric attribute type GA_0 (i.e. line-line, arc-arc, arc-line) while the other

correspond to the 4-D attribute vectors GA_{LL} , GA_{CC} , or GA_{CL} . Each 4-D attribute vectors is generalized as $[GA_1, GA_2, GA_3, GA_4]$. Let $GA = [GA_0, GA_1, GA_2, GA_3, GA_4]$ be a 5D vector. The input to this process is $\{GA, ID_s, P_s, NG_n\}$ of different r and q , where

- ID_s is the structure identity.
- NG_n is the n^{th} neighbor of the reference primitive of ID_s .
- GA represents the geometric relationships from the reference primitive of ID_s to NG_n .
- P_s is the confidence, i.e. length of the reference feature of ID_s normalized by total length of all features.

As discussed in section 6.2, one local structure contains only one reference primitive, which is unique to this particular local structure. Hence, ID_s represents both the local structure and the reference primitive. H is built step-by-step (t) over all local structures (the reference features and their neighbors) as

$$H_{t+1}(h, i, j, k, l) = H_t(h, i, j, k, l) \cup \{ID_s, P_s, NG_n\} \quad (6-18)$$

where $H_t(h, i, j, k, l)$ is an entry of H at step t . In the union process, ID_s , P_s , and NG_n are the information to be stored, while GA provides the 5-D index into H . One illustration of H is given in Fig. 6.15 with each location having an access to a list of

stored identity information, where $1 \leq h \leq 3$, $i = \frac{n_1}{N_1 + 1} * GA_1$, $j = \frac{n_2}{N_2 + 1} * GA_2$,

$k = \frac{n_3}{N_3 + 1} * GA_3$, $l = \frac{n_4}{N_4 + 1} * GA_4$. N_1, N_2, N_3 , and N_4 are the normalized ranges of

the 4 attributes as $[0, N_1]$, $[0, N_2]$, $[0, N_3]$ and $[0, N_4]$. Corresponding ranges are

divided into n_1, n_2, n_3, n_4 bins, i.e. H has $3 \times n_1 \times n_2 \times n_3 \times n_4$ entries (elements). We used

$3 \times 10 \times 10 \times 10 \times 10$ in this work empirically.

Figure 6.15: Typical units in a 5-D hypercube structure H .

To compute i, j, k and l , we use floating point number arithmetic. It is possible that i, j, k or l falls near bin margin. This can be tackled by growing a 4-D sphere with (i, j, k, l) as center with a radius of 0.5. Instead of updating one bin at (i, j, k, l) , we update all the locations within the sphere. Moreover, this process is only applied to H of the model image. A 2-D example is illustrated in Fig.6.16, where (i, j) is the location in floating point number which is computed from the input. Then the lists corresponding to the hypercube location $H((\text{int})i, (\text{int})j)$, $H((\text{int})i, (\text{int})j + 1)$, $H((\text{int})i + 1, (\text{int})j)$, and $H((\text{int})i + 1, (\text{int})j + 1)$ are all updated, where $(\text{int})i$ means floating point number i is converted to an integer.

The computation of H of a test image does not go through this process, i.e. i, j, k and l are computed using integer arithmetic without sphere expansion. It is because the correction needs to be done only once. For more details on indexing, one can refer

to chapter 7 and chapter 8.

Figure 6.16: 2-D spherical extension.

6.5 Summary

The work related to shape description has been reviewed in this chapter. A local structure based shape representation method is proposed. It first employs a combination of Gestalt laws to guide the extraction of local structures. Some typical local structures have been displayed. Then, a set of invariant pair-wise geometric attributes are proposed to capture the inherent local structural information accurately. Finally, these local structure relations are incorporated into a hypercube structures for compact represent of the object. Compared with other good methods, e.g. selecting neighbors by midpoint distance, the proposed method is capable of finding stable and salient structures of the object.

Chapter 7 Indexing Structure Generation

A hypercube based indexing structure is proposed to store all the model images. It organizes the model images by indices so that the searching strategy can be used for image retrieval. Searching strategy is necessary for an efficient retrieval, since the price for brute force matching between the test and model images is high when the number of images in database is large. In this work, all the models are organized and stored in the proposed indexing structure in terms of the local structure based shape representation (chapter 6). The models' indices to the database are their local structure information. In retrieval, the query local structures locate the relevant candidates by probing into the pre-built indexing structure, thus the searching scope is narrowed down.

7.1 Indexing Technique Overview

Three techniques are commonly used to cope with the “*dimensional curse*” [7].

- 1) *Dimension reduction methods*. KLT [88, 89] and PCA [90, 91] are commonly used dimension reduction methods. The computational complexity of dimension reduction approach is high. In addition, it is dangerous to reduce dimensions blindly, since information can be lost if the reduction is below the embedded dimension.
- 2) *Multidimensional indexing structure*. The large computational costs associated with similarity measures necessitate the organization of image indices such that

efficient search strategies can be used. Flexible data structures should be used in order to facilitate storage / retrieval in visual information systems. Structures such as B^+ -tree[92], VAM-split R-tree[93], TV-tree[94], k - d tree [95, 96], M-tree [3], quad-tree, and grid file [97] are commonly used. It was point out by Lu [7] that the search computation complexity of almost all these indexing structures increases exponentially with the number of feature vector dimensions. The search might be time consuming during k -nearest neighbors retrieval (The k objects which have distances from the query object that are the smallest are retrieved). In this paper, we propose a hypercube based indexing structure. The search processes can be simplified as an intersection of input and indexing structure. Its computation complexity is linear to the number of input structures.

- 3) *Filtering techniques* [98, 99]. Filtering processes are used to reduce the search space such that the retrieval process can speed up. In [98], Charikar proposed an incremental clustering technique for dynamic information retrieval. In [99], the triangular inequality was used to reduce the number of direct feature comparison in a database search. The drawback is that additional information is required to enhance the filtering rate when the database increases in size. In practice, the combination of indexing structures and filtering techniques can be used in different stages to speed up the retrieval process. Models can be organized into an indexing structure. Filtering techniques can then be used to eliminate the unnecessary comparisons.

7.2 Indexing Structure Generation

To generate the indexing structure ($\tilde{I}$), we first construct the hypercube structure, H , for each model and then take the union of H s to generate $\tilde{I}$. H and $\tilde{I}$ are of the same size and dimensions. $\tilde{I}$ can also be built step-by-step (t) similarly as in formula (6-18) as

$$\tilde{I}_{t+1}(h, i, j, k, l) = \tilde{I}_t(h, i, j, k, l) \cup \{ID_m, ID_s, P_s, NG_n\} \quad (7-1)$$

where $\tilde{I}_t(h, i, j, k, l)$ is an entry in $\tilde{I}$ at t^{th} step and ID_m is the additional information representing 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 model with respect to the input image. The indexing structure is illustrated in Fig. 7.1. At each location (hypercube) in $\tilde{I}$, a list of models with similar local structure are stored. This is equivalent to classify the models by the local structures. Thus, with $\tilde{I}$, it is easy to locate the relevant models that have common local structures with the input query in retrieval process. That is, no complicate searching strategy is necessary. Moreover, since the indexing structure is generated off-line and sufficient information of the models is stored, it benefits for efficient on-line retrieval.

Insertion / Deletion operations on $\tilde{I}$

The operation of inserting a new model into $\tilde{I}$ is similar to the construction of indexing structure, i.e. take union: $\tilde{I}(h, i, j, k, l) = \tilde{I}(h, i, j, k, l) \cup$

$\{ID_{insertion}, ID_s, P_s, NG_n\}$. Deletion operation takes a subtraction: $\tilde{I}(h, i, j, k, l) = \tilde{I}(h, i, j, k, l) - \{ID_{deletion}, ID_s, P_s, NG_n\}$.

Indexing Structure $\tilde{I}$

index	list
...	...
$\tilde{I}(h, i, j, k, l)$	$\left\{ \begin{array}{l} \{ID_{m'}, ID_{s'}, P_{s'}, NG_{n'}\} \\ \{ID_{m''}, ID_{s''}, P_{s''}, NG_{n''}\} \\ \dots \\ \{ID_{\hat{m}}, ID_{\hat{s}}, P_{\hat{s}}, NG_{\hat{n}}\} \end{array} \right.$
...	...

Figure 7.1: Display of Indexing Structure.

7.3 Summary

In this chapter, the work related to shape indexing has been reviewed. For fast identifying reasonable results, a hypercube based indexing structure is proposed. It uses 5-D hypercubes as indices. The list at each location records models that share similar local structure. Such information will be manipulated and compared in later processes to identify the most likely matches. The indexing structure is effective since sufficient object information is captured. It is efficient since no training is required. Models can be inserted / deleted conveniently. With the indexing structure, searching strategy can be performed, which will be described in next chapter.

Chapter 8 Retrieval by Indexing and Voting

In the retrieval process, the system analyzes a query and then a search process is performed and finally results of relevant matches are displayed. Efficient retrieval strategy is important for speeding up the search process. In this work, images are retrieved by indexing and voting (similarity measure). The computational complexity of on-line retrieval is linear with respect to the input pattern size, and the database size. With the pre-built indexing structure, image retrieval can be performed in 4 steps as shown in Fig. 8.1: (1) intersect, (2) filter, (3) rank, and (4) display.

Figure 8.1: Flowchart of retrieval strategy.

First, H_{input} is intersected with the pre-built indexing structure to screen out irrelevant models. Then, weak relevant models are filtered out to compose a candidate

list. Next, one can rank the candidates by voting for optimal models. Finally, the retrieved images are presented to the user in the descending order of the similarity to the query.

8.1 Intersection

To cut down computational expense on retrieval, irrelevant models are first screened out to reduce search space. Since the indexing structure is of the same size and dimensions as the hypercube structure of the input, the indexing process can be simplified as an intersection operation. H_{input} (hypercube structure of input) is intersected with $\tilde{I}$ (the indexing structure) to identify non-empty hypercubes in both structures. The list in each non-empty hypercube stores the information of the models that are relevant to the query. For easy demonstration, the 2-D sections of the structures are used to illustrate the process as in Fig. 8.2. Each list could contain numerous $\{ID_m, ID_s, P_s, NG_n\}$.

Figure 8.2: Intersection process.

8.2 Filtering

Figure 8.3: Accuracies according to different thresholds for weak neighborhood filtering (Four neighbors are grouped in one local structure).

Even though irrelevant models have been filtered out by intersection, we still need to filter out weak relevant models extracted such that the computation of similarity measure can be further simplified. Define a weak relevant model (ID_m) as one without any strong neighborhood match with the input (H_{input}). Since each neighborhood has one reference feature and n neighbors, we define a weak response as a match with less than half (less than $\frac{n}{2}$) of the neighbors being matched successfully. Hence, the intersection needs to produce $\frac{n}{2}$ (or more) entries of $\{ID_m, ID_s, P_s, NG_i\}$ for a strong neighborhood match with $1 \leq i \leq n$. In this work, n is set to 4 according to the finding in Fig. 6.7. The threshold for strong neighborhood match is set as half of the neighbors (i.e. $\frac{n}{2}$) according to the experimental results listed in Fig. 8.3, which records the accuracies with respect to different thresholds. The accuracy was measured in

similarity based queries by Bull's Eyes Percentage (BEP). Using two neighbors (half of the neighbors) has the highest accuracy; therefore, $\frac{n}{2}$ is used as the threshold in this work.

8.3 Ranking

One can rank the candidates with different means. L1-norm and L2-norm are two common choices to calculate the distance (dissimilarity) between an input cube and a model cube as shown in formulae (8-1) and (8-2).

$$D_{L1} = \frac{1}{(N_{input} + N_m)} \sum_{(h,i,j,k,l)} |cnt(H_{input}(h,i,j,k,l)) - cnt(H_{model}(h,i,j,k,l))| \quad (8-1)$$

$$D_{L2} = \frac{1}{(N_{input} + N_m)} \sqrt{\sum_{(h,i,j,k,l)} (cnt(H_{input}(h,i,j,k,l)) - cnt(H_{model}(h,i,j,k,l)))^2} \quad (8-2)$$

Here, D_{L1} and D_{L2} represent the distances calculated using L1-norm and L2-norm respectively, $cnt(H(h,i,j,k,l))$ represents the count of local structures falling into the location $H(h,i,j,k,l)$, N_{input} is the feature size of input query, and N_m is the feature size of model m . The matches with the shortest distances are considered as the similar shape of each other. However, the result is not desirable since formula (8-1) and (8-2) do not make full use of the local structure information recorded in H . The overall system performance of L1-norm and L2-norm will be analyzed in chapter 9. Here, two example queries are given to illustrate the less desirable results in Fig. 8.4 ~ 8.5. The first example is a logo as shown in Fig. 8.4 (a). It has six different variations in the database. From Fig. 8.4 (b) ~ (c), it is found that only five and two variations are

retrieved by the L1-norm and the L2-norm respectively. The second example makes use of one component of the same logo as shown in Fig. 8.5 (a). The results are expected to be better to return more if not all the six variations. However, the results are similar and the rankings of the relevant logos have dropped further as shown in Fig. 8.5 (b) and (c). To go for better clustering ability, various similarity measures will be proposed and discussed in the following. These similarity measures will make full use of the local structure information recorded in H .

(a) Query.

(b) Results by formula (8-1),

(c) Results by formula (8-2).

Figure 8.4: Similarity based query by L1-norm / L2-norm.

(a) Query.

(b) Results by formula (8-1),

(c) Results by formula (8-2).

Figure 8.5: Partial query by L1-norm / L2-norm.

After intersection and filtering, the obtained neighborhood matching from an input to a model are many to many mapping, i.e. one input neighborhood can be a possible match to several neighborhoods (ID_s) of a model (ID_m) and vice versa. This is not desirable since there are so many combinations to find the maximum match that the searching efficiency decreases. In view that the image changes usually scale evenly in each sub-part, we impose a sequential order on the local structure to narrow down the mapping. We first order each local structure according to the length (i.e. the P_s defined in Section 7.2) of its reference feature. We then allow pairing of local structures

from an input to a model in a diagonal fashion as shown in Fig. 8.6 with s and s' representing the structures from model and test input, respectively. This matching restriction produces a 1 – to – 1 matching schema and guarantees scaling consistency, i.e. when a long input reference feature is matched to a relatively long model reference feature, then a short input reference feature is matched to a relatively short model reference feature. The Δ in Fig. 8.6 is the dynamic tolerance to control pairing. It is set as 10% of current expected diagonal value. For example, a new match point (e.g. p_5 in Fig. 8.6) can be added if it is within the width tolerance with respect to the tip of estimated diagonal passing through all the previously accepted match points (e.g. p_1 , p_2 , p_3 and p_4 in Fig. 8.6). Δ is not an absolute value since the relative difference between two primitives are better than the absolute value in specifying their relationship. This matching scheme is implemented using Dynamic Programming to find the maximum similarity measures. The matches along the best diagonal path can be found to vote for a candidate model. Finally, each candidate model is ranked by the vote and the top most likely matches can be found.

Figure 8.6: Neighborhood-matching.

The basic similarity (voting) measure is defined in formula (8-3), while formulae (8-4) and (8-5) are adaptation of (8-3) with refinements for different needs. In formula (8-3), s (from model) is matched to s' of the input. N_{input} is the pattern size (number of local structures) of input query ID_{input} . N_m is the pattern size (number of local structures) of candidate model ID_m . $nID(s')$ is the number of models that have a similar structure to s' . $nID(s')$ is used to measure the discriminating power of s' as $f(nID(s')) = \frac{(DBSIZE - nID(s'))}{DBSIZE}$. A low $nID(s')$ implies a more unique local structure $ID_{s'}$ as specified in Fig. 8.7. $DBSIZE$ is the number of models in the database.

$$Score(m) = \sum_{ID_m \in \chi} \left(\left(\frac{1}{N_{input}} + \frac{1}{N_m} \right) \frac{(DBSIZE - nID(s'))}{DBSIZE} \right) \quad (8-3)$$

$$Score(m) = \sum_{ID_m \in \chi} \left((P_{s'}^{input} + P_s^m) \frac{(DBSIZE - nID(s'))}{DBSIZE} \right) \quad (8-4)$$

$$Score(m) = \sum_{ID_m \in \chi} \left(\left(P_{s'}^{input} + P_s^m + \frac{1}{N_{input}} + \frac{1}{N_m} \right) \frac{(DBSIZE - nID(s'))}{DBSIZE} \right) \quad (8-5)$$

Figure 8.7: The desirable discriminating power of $nID(s')$.

Since N_{input} and N_m are constant for ID_{input} and ID_m , the similarity measure in

formula (8-3) applies the same confidence / weight, i.e. $\frac{1}{N_{input}} + \frac{1}{N_m}$, on each structure in ID_m . It is desirable for images without dominant features where each structure votes equally for the best match, i.e. contributes equally to the similarity score. However, it is unfavorable for long feature dominant images since the dominant primitive (the long lines / arc with high *ArcoId*) vote equally as the details (the short lines / arc with low *ArcoId*). Therefore, we define another similarity measure using P_s^m and $P_{s'}^{input}$ in formula (8-4) to discriminate dominant structures and minor structures, where s (from model) has been matched to s' of the input, P_s^m is the confidence of local structure ID_s voting on candidate model ID_m , and $P_{s'}^{input}$ is the confidence of local structure $ID_{s'}$ voting on input query ID_{input} . This similarity measure favors long features, since P_s^m and $P_{s'}^{input}$ are proportional to length. Empirically, we found that formula (8-3) and (8-4) can be combined to give formula (8-5) to provide better performance since the longer features can still dominate the match, while the fine details are given the necessary influence on the vote too.

In Fig. 8.8 ~ 8.9, we make a direct comparison among the top 20 query results measured by formula (8-3), (8-4) and (8-5). The results are ranked according to the similarity from left to right and from top to down. It is observed that the query results in Fig. 8.8 (c) (d) are much better than Fig. 8.8 (b), since only four desirable logos are ranked in top 20 positions in Fig. 8.8 (b), while all the six desirable logos are ranked among top 20 in Fig. 8.8 (c) and (d). The results illustrate the limitations of the similarity measure with uniform confidence / weight on all features. Even though Fig. 8.8 (c) (d) show that both similarity measures have similar results as a whole on the

given query (Fig. 8.8 (a)), some of the results are different. Formula (8-5) ranks the models with letters (details) in higher positions, e.g. the 8th, and 11th logos in Fig. 8.8 (d). On the contrary, formula (8-4) ranks models with more global circular shape in higher positions, e.g. the 8th, and 11th logos in Fig. 8.8(c). It substantiates that formula (8-5) gives more consideration on object details than formula (8-4). To see the rationale in choosing formula (8-5) over formula (8-4), another query example (Fig. 8.9(a)) is sent to the system and the results are illustrated in Fig. 8.9 (b) (c) (d).

In Fig 8.9, the input query is a pentacle with two circles. Compared with the circles, the line segments of the pentacle are not the dominant features since each line is much shorter than the circles. However, the pentacle captures much of the shape information (i.e. important feature points). Therefore, similar models with pentacle should be given certain priority (rank in high position), and this is fulfilled by formula (8-3) and (8-5). It can be observed that five models containing pentacle, i.e. the 1st, 5th, 11th, 12th, and 20th, have been retrieved by formula (8-4) as shown in Fig. 8.9 (c). However, seven models containing pentacle, i.e. the 1st, 5th, 6th, 7th, 14th, 15th, and 20th, can be retrieved by formula (8-5) as shown in Fig. 8.9 (d). The best results are generated by formula (8-3), nine models containing pentacle, i.e. 1st, 2nd, 3rd, 6th, 8th, 12th, 13th, 16th, and 20th, are returned as in Fig. 8.9 (b). Eclectically, formula (8-5) is employed in this work in relevant queries.

Figure 8.8: Global shape dominated query.

Figure 8.9: Non-global shape dominated query.

For partial query, since people care more on "how much of the query content has been matched", the similarity measure is adapted to highlight the input as in formula (8-6). A simple comparison of the top 20 retrieval results measured by formula (8-5) and (8-6) is shown in Fig. 8.10. The query input is a trademark with some details

removed. There are six versions of similar logos in the database. Results in Fig. 8.10 (c) are better than those in Fig. 8.10 (b), since they rank more similar models in high positions. Formula (8-6) ranks six logos, i.e. 1st, 4th, 5th, 6th, 7th, and 8th in high positions, while formula (8-5) can only rank five out of six logos, i.e. 1st, 3rd, 4th, 7th, and 9th in high positions. Furthermore, formula (8-6) retrieves a very similar logo at 18th place.

$$Score(m) = \sum_{ID_m \in \chi} \left(\left(P_{s'}^{input} + \frac{1}{N_{input}} \right) \frac{(DBSIZE - nID(s'))}{DBSIZE} \right) \quad (8-6)$$

(a) Query.

(b) Results by Formula (8-5)

(c) Results by Formula (8-6)

Figure 8.10: Partial query using two similarity measures.

In the experiments in chapter 9, formula (8-5) is employed for the retrieval queries with respect to noisy environment (test set 1), 3D distortion (test set 2), similarity (test set 3), cluttered environment (test set 5) and real object images (test set 6) while Formula (8-6) is used for the partial query (test set 4).

8.4 Retrieval Timing Analysis

The on-line complexity of image indexing can be expressed in three parts; one is the number of local structures of the input image that need to be computed, i.e. query pattern size. The others are the intersection and formula (8-4) / formula (8-5) that needs to be computed. Let C_X be the number of model candidates in set X after intersection, the operation complexity will be $O(N_{input}) + O(Intersection) + O(C_X) \approx O(N_{input} + C_X) = O(N_{input} + c N_{input}) = O((c+1) N_{input})$. Hence the complexity (retrieval time) is linear with respect to the input complexity as illustrated in Fig. 8.11. The relationship curve in Fig. 8.11 is plotted by recording the retrieval times when querying images with various pattern sizes are fed to the system with a database of 1000 models. On average, there are 58 local structures per model image. It can be observed that the retrieval time is approximately linear with respect to the query pattern size.

The $O(C_X)$ is also related to the size of the image database and the average pattern size (per image) of the database as illustrated in Fig. 8.12~8.13. Since the indexing structure and image retrieval are based on the local structure, the average complexity of the images in the database (average pattern size per image) as well as the size of the database will affect the computational expense of a query. In Fig. 8.12, we

display the relationship between retrieval time and average pattern size. A query image with 52 local structures probes the pre-built indexing structure (consisting of 1000 images), and the retrieval time is measured and plotted. It is found that the query time is directly proportional (linear) to the average pattern size. The relationship of the retrieval time and database size is shown in Fig. 8.13. It can be observed that the query time is approximately linear to the database size.

Figure 8.11: Relationship between retrieval time and input pattern size
(Size of image database: 1000. Average pattern size per image: 58 local structures.)

Figure 8.12: Relationship between retrieval time and average pattern size per image in the database (Size of database: 1000. Query size: 52 local structures).

Figure 8.13: Relationship between retrieval time and database size
(Average pattern size per image: 58 local structures, Query size: 52 local structures).

8.5 Display

The query results are presented to the user in the form of text and images. The results in text are stored in a text file. It gives query information about the images in the candidate list. The information includes rank, image identification, similarity score, number of local structure matched as shown in Fig. 8.14.

The visual results are presented to the user as illustrated in Fig. 8.15. The images are displayed in dialog, and in similarity descend order, from left to right and from up to down. The title of the dialog is image identification. By setting a parameter in program, one can choose the number of images to display, e.g. 5, 10, or 20 etc.

Rank	Image ID	Similarity Score	Local Structure Matched
1	44-1024.tif	0.99874	92
2	44-1022.tif	0.624458	47
3	44-1023.tif	0.599065	43
4	44-1025.tif	0.542991	46
5	56-1705.tif	0.41999	41
6	56-1408.tif	0.404089	28
7	44-1169.tif	0.378204	51
8	54-1202.tif	0.360372	30
9	44-209.tif	0.336118	31
10	54-1702.tif	0.315577	43
11	54-1171.tif	0.307378	29
12	54-1025.tif	0.295837	29
13	54-1123.tif	0.29356	33
14	54-1708.tif	0.291503	35
15	54-113.tif	0.285529	36
16	46-109.tif	0.285529	36
17	54-1190.tif	0.281164	29
18	56-1120.tif	0.27562	11
19	44-1040.tif	0.268283	40
20	56-508.tif	0.267364	42
21	54-202.tif	0.263211	38
22	54-1101.tif	0.251127	28
23	54-1122.tif	0.247131	26
24	54-1143.tif	0.243689	39
25	54-1811.tif	0.243291	26

Figure 8.14: Query results in text mode.

Figure 8.15: Visual query results.

8.6 Summary

In this chapter, the work relating to shape retrieval by indexing and voting has been presented. It is done by indexing candidate models from the pre-built indexing structure firstly, and then voting on the optimal matches. The retrieval time of the proposed method is linear to the database size.

Chapter 9 Results and Discussion

In this chapter, system performance evaluation is carried out. The proposed system has been employed in the retrieval of the logo database and stationery images database. Six types of query, i.e. letter retrieval, 3D-distortion query, similarity based query, partial query, mixed object query, and real object image query (set 6) were conducted in various adverse environments respectively. The proposed method was compared with existing local structure construction method (6NN) and similarity measure methods (L1-norm, L2-norm) on these 6 types of query. Impressive results have been obtained.

9.1 Input Data

The proposed system has been tested using the logo database from the work of Huet and Hancock [73]¹. In addition, a stationery image database was also tested². The logo database consists of 2711 simple characters and 984 complex trademarks and cartoons. The stationery image database has 53 images of 8 types of real objects taken at different viewing angles. One can refer to Appendix A for the whole database. Some examples of the images are displayed in Fig. 9.1. The details of the input and the corresponding model sets can be found in table 9.1. In total, six sets of test images were

¹ <http://www.eurecom.fr/~huet>

² <http://www-2.cs.cmu.edu/~cil/v-images.html>

used to assess the effectiveness of the proposed method.

Figure 9.1: Examples of test images.

Test Set	Model Database Size	Test Image Set Specification			
		Size	group	description	Objective
Set 1	260 characters	260	26 groups of the English alphabets; each group consists of 10 images.	General variations of shapes, such as deformation, erosion, rotation, scaling, insertion of additional line and arc features, edge fragmentation, etc.	Assessment of robustness to variations.
Set 2	200 transformed trademarks	200	4 groups in total, each group consists of 50 images.	Random 3D cylindrical and spherical distortions.	Assessment of robustness to 3D Distortions.
Set 3	984 trademarks	50	22 groups: one 6-image group, two 3-image groups, and nineteen 2-image groups.	Similar images grouped by human's perception.	Assessment of similarity-based retrieval.
Set 4	984 trademarks	109	-	Cuttings of the original models in database.	Assessment of partial query.
Set 5	984 trademarks	20	-	Merging of the complete/incomplete original models in database.	Assessment of mixed object query.
Set 6	53 Stationery objects	53	8 groups in total, each group captures 4 ~13 views of a particular stationery object.	2D images of 3D real objects captured from 4 ~13 different viewing angles.	Assessment of real 3D object query.

Table 9.1: Description of Input Data.

- Test set 1 (Fig. 9.1) has clear cut 26 groups of distinct letters, with sufficient variations of shape, like combinations of multiple characters, characters with distortions of added noise, occlusion, and different types of transformations like rotation, scale change and 3D effect. Each combined-characters input (e.g. the first image of Fig. 9.1, set 1) was classified into multi-groups

according to the image components.

- Set 2 investigates the adverse effect of random 3D cylindrical and spherical distortions. Since many applications deal with 2D projection of 3D objects, a set of 3D cylindrical and spherical transformed characters were added into the database to explore the algorithm's reaction to these aspects. Four models were selected randomly from the trademark database, and four groups were formed accordingly. Each group captures 50 random 3D distortions of one distinct model.
- Set 3 has 50 test images from 22 groups. The test images in each group are perceptually similar to each other. The test groups were generated from the judgments of eight persons on whether the images were similar and could be grouped into one group. Each group got no less than six person's supports.
- Set 4 tests on partial matching. There are 109 test images obtained by cutting 14 model logos in random. The more complicate model images had been cut more number of times to produce more test images. Almost 60 percents of the cuttings were larger than half of the logo content as shown in Fig. 9.1. This simulates occlusions that occur frequently on items in our surroundings. The objective is to retrieve the 14 models by feeding the 109 test images to the system.
- Set 5 consists of 20 mixed images. They were created by merging 2 or 3 different objects into a new image. Set 5 was designed to test system's performance on retrieval under cluttered environment.

- Set 6 consists of 53 real images of 8 types of stationeries captured from 4 ~ 13 different viewing angles. This provides a small test-bed to test the proposed approach in tackling images of real 3D objects, and its potential for real image vision task such as robot vision.

9.2 Test methodology

The experiments were carried out as follows. An example query image was selected by the user, and the retrieval results were displayed on the screen from left to right, from top to bottom in decreasing similarity. As for the system evaluation, the ranking lists of matches were described by *precision-recall plot*, which is defined as follows. For each query model in Class A and any number K of top matches, r is the number of models in class A returned within the top K matches and T is the number of models in Class A. *Recall* (the horizontal axis) represents the ratio of r to T , while *precision* (the vertical axis) indicates the ratio of r to K . A perfect retrieval result produces a horizontal line across the top of the plot (at *precision* = 1.0) as shown in Fig. 9.2, indicating that all the models within the query object's class are returned as the top ranked matches. Practically, the result is seldom perfect, which produces a curve dropping at the tail as shown in Fig. 9.2. It indicates that the models within the query object's class are not all returned as the top ranked matches, e.g. the last points is calculated when $K = 2r = 2T$. Good retrieval results produce curves close to the ideal curve.

Figure 9.2: Illustration of the precision-recall curves.

The system performance was compared with that of the six-nearest-neighbor local structure method [73] in terms of precision-recall plot. The purpose of this comparison is to demonstrate that the proposed concept is superior to six-nearest-neighbor (6NN) in perceptual grouping primitives, especially in the case of partial query and cluttered object query. The 6 nearest-neighbor method uses line-segments as primitives and constructs local structures by selecting six neighbors with the shortest distances to the midpoint of a reference line. The performance was also compared to that using L1-norm and L2-norm, which had been described in section 8.3. The purpose of these comparisons is to illustrate that the better performance of the proposed image indexing / retrieval approach.

In order to perform the above mentioned comparisons, retrieval experiments were performed under different conditions and the results obtained were compared. Under all conditions used, the local structure information was extracted first and similarity was then measured. To compare local structure construction methods, we used the same indexing and similarity measure method as shown in Fig. 9.3, where the main

difference between the implementations of two methods is presented in the dotted box.

Figure 9.3: Procedures in local structure comparison.

Similarly, to evaluate the indexing and similarity measure against L1 / L2 norm, we used the same local structure information as shown in Fig. 9.4. The different implementations of the two methods are also surrounded by a dotted box.

Corresponding to the flowchart in Fig. 9.3 ~ 9.4, the schemes and formulae used for each method are list in table 9.2. The local structure construction employed in L1-norm, L2-norm, and ALSBIR were described in section 6.2, while that in 6NN was proposed in [73]. For these distance / similarity measures, one can refer to section 8.3 for more details.

Figure 9.4: Procedures in similarity measure comparison.

Method	Local structure construction	Distance/similarity measure
L1-norm	Section 6.2	Formula (8-1)
L2-norm	Section 6.2	Formula (8-2)
6NN	6NN from [73]	Formula (8-5) or (8-6)
ALSBIR	Section 6.2	Formula (8-5) or (8-6)

Table 9.2: Schemes and formula used in result comparison.

9.3 Experimental Results

The precision-versus-recall curves of the proposed system and methods for comparison are plotted in Fig. 9.5. Note that the plotting of curves in Fig. 9.5 (b) and (f) stops prior to the value of recall reaches 1 since the retrievals stopped when 100 images or half of the database images (depending on which came first) have been retrieved. The L1-norm, L2-norm, 6NN and ALSBIR schemes are tested here. Their definitions are summarized in Table 9.2. The L1-norm and L2-norm are the commonly used measures. Their performances were found inferior to the 6NN and ALSBIR, because the L1-norm and L2-norm did not make use of the structural information recorded in the hypercube. The 6NN was proposed by Huet and Hancock to compare with other graph structures, such as the Delaunay graph [155]. 6NN was found to perform the best. Hence, we compare here the proposed approach with 6NN. The results show that the proposed method performs as good as the 6NN on single object retrieval. However, it out-performed 6NN on the tests of occluded and cluttered object queries. In the following, each of the tests is discussed in detail with retrieved examples displaying the top 10 most likely matches suggested by the system from Fig. 9.6 to Fig. 9.12. The query results are arranged in descending order according to similarity from left to right and from top to bottom.

(a) Letter retrieval (set 1)

(b) 3D distortion query (set 2)

(c) Similarity based query (set 3)

(d) Partial query (set 4)

(e) Mixed-object query (set 5)

(f) 3D object query (set 6)

Figure 9.5: Precision-recall plot comparisons.

9.3.1 Experiment 1: Letter retrieval

The letter test set has 26 clear-cut types of letters. The experiment tested the proposed system's robustness to variations such as scaling by a factor of 0.5 or 2, rotation by 135, 270 degrees, corruption with extra lines or combination with other letters. The curves of precision versus recall on the first set are shown in Fig. 9.5 (a). It is observed that ALSBIR performed slightly better than the 6NN and much better than the L1-norm and L2-norm measures. Three example results are displayed in Fig. 9.6. It is observed that most of the letters in the A's group were returned and ranked in relatively high positions at rows 1 and 2, except for the input combined with a small letter A (e.g. the first image of the third row in Fig. 9.6). It is reasonable since letter B dominates the image content in such shape. Besides images of the letter A, the system also retrieved images of the letter V, which is reasonable because V is part of A from the viewpoint of shape analysis. On the third row, the B shape or shapes that seem to be part of B were favored. For more results, one can refer to Appendix B.

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10

Figure 9.6: Example results of letter retrieval.

9.3.2 Experiment 2: Robustness to 3D Distortions

The second experiment tested the proposed method's performance on various cylindrical and spherical distortions systematically. Four models were selected to generate 200 test images, produced by random 3D cylindrical and spherical transformation. The transformed images from the same model were grouped together. The precision-recall curves are displayed in Fig. 9.5 (b). It is observed that all four methods produced similar results but ALSBIR had performed better. Four example results are illustrated in Fig. 9.7. It is observed that most of the same trademarks can be grouped and retrieved by the proposed system, regardless of where they were pasted on, a ball or a cylinder. For more results, one can refer to Appendix B.

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10

Figure 9.7: Example results of 3D-distortion queries.

9.3.3 Experiment 3: Similarity based Retrieval

The third experiment tested the proposed approach's ability in similarity clustering. The precision-versus-recall curves are displayed in Fig. 9.5 (c). The results demonstrate that the proposed method slightly outperforms the 6NN but greatly outperforms the L1-norm / L2-norm similarity measure. Example results are presented in Fig. 9.8 and Fig. 9.9. The query at Fig. 9.8 (a) is a complex logo. Six versions of this logo can be found in the database. Four have the same overall shape but different caption and two have legends at the bottom. They form one group in test set 3 (group 1 in Fig. 9.9). The example results in Fig. 9.8 (c) demonstrate that the proposed method not only ranked the six relevant logos in the highest positions, but also ranked a very similar logo in high position too, i.e. "56-1705". In [73], only four logos were ranked high in Fig. 9.8 (b). Some more results are illustrated in Fig. 9.9. It is observed that objects of one group were clustered in high positions when the input was any member in the group. For more results, one can refer to Appendix B.

9.3.4 Experiment 4: Partial Query

The fourth experiment is partial query by feeding 109 cut logos to the system to retrieve the original 14 models. The results are displayed in Fig. 9.5 (d). It was found that ALSBIR greatly outperformed 6NN and L1-norm / L2-norm. Almost 100% of the test images could be retrieved successfully among the top 25 candidates by the proposed method, while only 60% could be retrieved successfully by the 6NN.

Example results are illustrated in Fig. 9.10. It was found that the system retrieved similar logos (i.e. top2 at row 1, top6 at row 2) in addition to the intended models (i.e. top1 at row 1 and top3 at row 2) from which the test images were generated. For more results, one can refer to Appendix B.

(a) Query example

(b) Retrieval results reproduced from [73]³

(c) Retrieval results from ALSBIR.

Figure 9.8: Example comparison of similarity based query.

³ Reprinted from Pattern Recognition, vol 35, B. Huet, E.R. Hancock, "Relational object recognition from large structural libraries", pp.1895-1915, Copyright (2002), with permission from Elsevier.

	Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10
Group 1
Group 4

Figure 9.9: Example results of similarity based queries.

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10

Figure 9.10: Example results of partial queries.

9.3.5 Experiment 5: Mixed-object Query

To test the potential of image retrieval from cluttered environment, images composed from multiple objects were used as input queries to find the most similar

models in the database. The system performance and comparisons with the 6NN and L1-norm / L2-norm are displayed in Fig. 9.5 (e). It is found that the proposed method can handle object retrieval from such mixed / cluttered environment. In addition, ALSBIR was found to perform much best. Examples of the retrieval results are displayed in Fig. 9.11(a)-(c).

9.3.6 Experiment 6: Real Object Image Query

To test the proposed approach in tackling images of real 3D object images captured from random viewing angles, and its potential for real image vision task such as robot vision, 8 groups of stationery object images were used. The system performance and comparisons are displayed in Fig. 9.5 (f). The results demonstrated that the proposed method performed similarly to the 6NN but outperformed the L1-norm / L2-norm. Examples of query results are shown in Fig. 9.12. The retrieval precisions are quite different from object to object; because some alter a lot in shape (and shadow) when the viewing angle is changed, e.g. the tape (2nd row). The result of the triangular shape is the best since it is flat without much influence from shadowing effect. For more results, one can refer to Appendix B. Our future work will spend more effort on 3D real object recognition, since the representation of 3D real object is more challenging.

Query	Retrieval Results

(a)

Query	Retrieval Results

(b)

Query	Retrieval Results

(c)

Figure 9.11: Example results of cluttered object queries
(solid: lines, dotted: arcs).

Query	Retrieval Results

Figure 9.12: Example results of 3D-object queries.

9.3.7 Error analysis

During the experiments, it was found that three major factors have led to the inaccuracy of the retrieval. First, the corruption of major primitives, e.g. the broken long lines and arcs with high *ArcoId*, can lower the rank of the model relatively. An example is analyzed in Fig. 9.13. To demonstrate the extracted local structures clearly, we use the letter A as an input. The model, which should have been returned in the top 10 (TN), is the letter A combined with the letter B on the left side (the 2nd row). The model should not have appeared in top 10 (FP) is the letter V (the 3rd row). Fig. 9.13 (a) displays all of the local structures in the images. For the TN, we expect that the local structure around each primitive of the letter A is similar to the local structure in the input with regard to the same primitive, i.e. we expect to see a similar dash dotted line arrangement about the letter A. However, it is found that the local structures (the dash dotted line arrangement) around the joint of the letter A and B have changed in TN from those in the input. For the FP, it is ideal if the local structures around primitives of the letter V are different from the local structures in the input. However, it is found that quite a lot of local structures of the letter V are similar to those in the input. In fact, the dash dotted line arrangements of the input and FP are quite similar to each other. For clarity of illustration, two of the local structures around the reference R1 and R2 are display in Fig. 9.13 (b) for the error analysis. The R1 and R2 are the similar primitives in the three images and R1 is shortened in TN. From the structures in Fig. 9.13 (b), it is observed that two common neighbors of R1 are selected, i.e. 1-1 and 1-2, from the input

and in TN respectively. It indicates that the local structures R1 in the input and in TN should be matched during the retrieval process. However, it is not the case since the local structures are matched based on their numeric representations. R1 in TN is much shorter than the R1 in the input. The numeric representations of the local structures R1 in the input and in TN are actually much different from each other, even though, both structures have two neighbors in common. Therefore, they are not matched. However, the local structures R1 in FP and in the input can be matched. It is because they have two common neighbors, i.e. 1-1 and 1-3, and R1 is not broken in FP, thus, the reference R1 computes similar numerical representations with the similar neighbors in both structures. Similar analysis can be applied to the reference R2. It is observed that the local structures R2 in the three images can be well matched. The previous analysis shows that the FP gets chance to be the candidate by at least two local structures R1 and R2, and TN loses chance to be the candidate by at least one local structure R1. Since R1 and R2 are longer than the other lines, their influences on similarity score (refer to formula (8-5)) are relative high. Thus, mismatching of the local structure R1 will reduce the similarity score obviously. For the TN, if the decrease is too large to be compensated for by the scores from the other relative small consistent structures, it will fall behind the FP.

Second, we have no effective mechanism to correct the misclassification error of primitives. The arc features and line features have been clearly sorted in an image representation structure, and the match between a reference arc and a reference line is not supported in this work. Therefore, when major primitives are misclassified, the

retrieval result is incorrect. An example is illustrated in Fig. 9.14. The input is a letter C, and two incomplete circles are extracted to represent the shape. The TN is rejected since the larger circle are classified as the line segments due to noises, while the FP is accepted since two circles similar to the input are extracted. Even though severe misclassifications as in Fig. 9.14 degrade the system performance sharply, fortunately, they seldom occur. The misclassifications that do not destroy the salient structure of the object can still produce acceptable results. For example, there is an inconsistency of arc classification among the three examples in Fig. 9.15, but they are still clustered together, i.e. the two correct models (TP) are ranked at the top. It is because the unique structure of the three objects, triangular, is not destroyed by the misclassifications.

Third, object analysis in 3D space is not explored in this work. Thus, the LEM might not provide enough information to discriminate real 3D objects. For example, in Fig. 9.16, it is found difficult to cluster the two tapes (input and TN) by using only the 2D information from the LEM. On the other hand, the input tape retrieved the stapler by the 2D LEM information, even though they are very different objects. Therefore, the lack of 3D object analysis has degraded the system performance on real object queries. In the future, we will explore this topic.

Figure 9.13: Error due to corrupting the major primitives.

(Solid: lines, Dotted: arcs, Dash dotted: connection from a reference to its neighbors.

TN: the correct model not retrieved, FP: the wrong model retrieved.)

Figure 9.14: Error due to misclassified primitives.

(Solid: lines, Dotted: arcs, Dot: partition points. TN: the correct model not retrieved, FP: the wrong model retrieved.)

Figure 9.15: The acceptable misclassifications of primitives.

(Solid: lines, Dotted: arcs, Dot: partition points. TP: the correct model retrieved)

Figure 9.16: Error due to insufficient object analysis.

(Solid: lines, Dotted: arcs, Dot: partition points. TN: the correct model not retrieved, FP: the wrong model retrieved.)

9.3.8 Sensitivity analysis

In this section, the sensitivities of the proposed system to different adverse factors, such as primitive corruption (broken or misclassified), 3D distortion, occlusion and cluttering, are analyzed. The sensitivity curves with respect to these factors are plotted in Fig. 9.17. Fig. 9.17(a) illustrates the accuracy, measured by BEP, with respect to the amount of broken and misclassified primitives. The curve is generated on test set 1 which consists of 260 letters. It is observed that the curve drops fast with the percentage of broken and misclassified primitives increasing. It has been analyzed how broken and misclassified primitives can affect the system performance in section 9.3.7. Moreover, the curves are produced based on the letter database where different

letters can be in similar shape. It is difficult to discriminate these from the shape's point of view, especially when one letter loses part of features (primitives), thus, the probability of these different letters in similar shape being ranked in top position is relatively high. These account for fast dropping of the curve after 20% percent primitives corrupts.

(a) Primitive Corruption sensitivity plot.

(b) 3D distortion sensitivity plot.

(c) Occlusion sensitivity plot.

(d) Cluttering sensitivity plot.

Figure 9.17: Sensitivity curves.

Fig. 9.17(b) illustrates the accuracy, measured by BEP, with respect to 3D distortions which is computed as the angle the object deviating from its front view. The angle is normalized by 180 degree. It is observed that the system is robust when the distortion is less than 30% (54 degree). Fig. 9.17(c) illustrates the accuracy,

measured by BEP, with respect to the amount of occlusions. It is observed that the system is robust when the cut is less than 60%. Fig. 9.17 (d) illustrates the accuracy, measured by BEP, with respect to the amount of the cluttering which is computed as $(1 - \alpha_i \beta_i)$, where α_i is the area ratio of the i^{th} input object with respect to its corresponding model image, and β_i is the area ratio of the i^{th} input object with respect to the input image. α_i will be less than 1 if the i^{th} input object is occluded. It is found that the system is robust when the cluttering is less than 60%.

9.4 Discussion and Comparison

9.4.1 Discussion

From the above experimental results, the proposed system, ALSBIR, has fulfilled all the objectives.

1. Human can identify and recognize an occluded object from a cluttered environment, provided that enough or distinguishable portion of the object can be observed. ALSBIR has demonstrated such ability according to experiments described in 9.3.4 and 9.3.5.
2. ALSBIR is built using the line and arc primitives from a shape instead of from the complete outline of a shape. The success of the system demonstrates that shape analysis can be tested on realistic input data instead of having the requirement that contours of the target object has to be pre-segmented. Furthermore, lines and arcs are deemed as suitable for shape

recognition in different adverse environments.

3. ALSBIR has shown a trend to retrieve similar models with respect to the input as its top 10 choices. This is supported by the example results shown in Fig. 9.5, 9.7, 9.8, 9.9, and 9.10. Hence, the proposed approach is suitable for similarity based retrieval.
4. The proposed local structure construction is guided by Gestalt Laws to construct salient structures that can be extracted consistently. From the above results in Section 9.3, ALSBIR consistently outperformed the 6NN which employs only the condition of nearest neighbors for construction. Hence, the proposed construction has been demonstrated to be better since Gestalt laws have been combined to look for stable shape. One example is shown in Fig. 9.18 where ALSBIR proposes the same neighbors with or without the surrounding frame. Thus, the same unique local neighborhood information around the pentacle has been extracted from both images. The 6NN on the other hand, relies only on distance and chooses different neighbors from one situation to the other. It fails to extract unique local neighborhood information around the pentacle when a frame exists. Hence, the generated local structures from 6NN are not useful in similarity based object retrieval.
5. The results from the simple L1-norm or L2-norm are consistently inferior to ALSBIR. This strengthens the proposed approach to store more related

structural information in the hypercube and followed by careful sorting procedure of the structural relation to produce more accurate result.

6. The complexity of the proposed system is linear to the size of the image database (refer to Fig. 8.13), therefore it is potential to cope with the *dimensional curse*.

Figure 9.18: Neighbor selection with / without a frame.

(The dot lines represent the connection from a reference feature to its neighbors, and two best neighbors are displayed for clear demonstration)

9.4.2 Comparison

It is difficult to compare the proposed system with other method comprehensively, since different approaches have different strengths and weaknesses,

and hence each test data set has been designed to highlight capability of each particular system. Furthermore, a fair comparison has to be under the same experiment conditions, which is usually not easy to guarantee. However, to the best of our knowledge, we have not found any system that can tackle as many query types as shown in our experiments and can function as good as the proposed technique. Based on the query types tested in each method, one can compare approximately the advantages of each method as shown in table 9.3. From the table, it can be observed that ALSBIR is designed for and can tackle more distortion types and environments.

Ref.	Query type							
	Exact query						In-exact query	
	noisy	transform invariance	non-rigid deform	3-D distorted	similarity based query	3-D object	Partial query	Mixed multi objects
Jain [11]	yes	no	no	no	yes	no	no	no
Kim [156]	yes	no	no	no	yes	no	no	no
Berretti [3]	no	no	no	no	yes	yes	yes	no
Huet [73]	yes	yes	no	no	yes	no	yes	no
Zhang [27]	yes	yes	yes	no	yes	no	no	no
Bimbo [21]	no	yes	yes	no	yes	yes	no	no
Belongie [4]	yes	yes	yes	no	yes	yes	yes	no
ALSBIR	yes	yes	no	yes	yes	yes	yes	yes

Table 9.3: Query types in relevant work.

We had compared our retrieval results in Fig. 9.8 (c) with that reported in [6, 73] by Huet and Hancock in Fig. 9.8 (b). The proposed approach was able to perform better on such example to retrieval 6 instead of 4 related models. In addition, it suggested another similar shape at the top 5 position. In [11], two sets of shape features were used to describe the global shape information: a 72 bin histogram of the shape edge direction and seven invariant moments. Shape similarity was calculated as a weighted sum of the Euclidean distance between edge histograms and the Euclidean distance between invariant moments. The experiment was conducted on a database of 1100 trademarks. The test trademarks included hand drawn and filled trademarks. The trademark can be correctly retrieved within the top 20 matches with 99% correct rate under rotated, scaled and added uniform noise conditions. No indexing scheme was proposed. Moreover, the input data that were fed to their method were assumed to be complete shape outlines. This perfect segmentation assumption limits their method's applicability to a great extent, since images obtained from practical channels always comprise noisy and occluded parts rather than perfect outlines. In [156], Kims employed the Zernike moments to extract visually salient feature that dominantly affected the global shape of trademarks by ignoring minor details. Their method depended on radial complexity and m-fold circular symmetry of the shape. The authors examined the proposed system with ten test trademarks and a database of 3000 trademarks. Among the top 30 retrieved trademarks, the correct rate was 92.5%. This method was efficient 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 moments cannot tackle occlusion which results in shift of its center of gravity. In [3], the curvature zero-crossing points from a Gaussian smoothed boundary were used to obtain primitives, i.e. tokens. The feature representation was scale invariant but not rotation invariant. An M-tree was exploited to index the tokens into the feature database. Its major drawback was the assumption that the occlusion boundaries have been identified, and segments belonging to the contours of single objects were available as open shapes. Their experiments were conducted on a database of over 400 shapes taken from 20th century paintings. From their performance curve, the precision was 75% and the recall was 70% when 30% shape was occluded. In [27], Zhang and Lu proposed a generic Fourier descriptor. It was derived by applying two-dimensional Fourier transform on a polar-raster sampled shape image. The similarity between two shapes was measured by the city block distance between their generic shape descriptors. Four sets of images used for system evaluation. 420 images (70 groups) were used to test scale invariance. 420 images (70 groups) were used to test rotation invariance. 1400 images (70 groups) were used to test similarity-based retrieval. 200 out of 1300 images were used to test non-rigid deformations. The authors got almost 100% recognition rate on non-rigid deformations and 60% rate on similar object retrieval. However, their method could not cope with occlusion and clutter problem since global shape features were used. In [21], the technique on boundary deformation minimized a cost function that depended on a weighted sum of the mismatches of the two shapes and on the deformation energy. The complexity depends on the number of points on the contour. Hence, the optimization

was computationally expensive and this, in spite of the better precision, has limited its usefulness in image database. Twenty-two images were used for similarity retrieval and 100 images were used to conduct sketch-based retrieval. An example was used for the query with multiple templates. They achieved 89%, 88%, 86% query accuracy in similarity retrieval, sketch-based retrieval, and the query with multiple templates, respectively. However, the technique is computation expensive and there is no mechanism to handle occlusion or object cluttering. In [4], Belongie *et al.* extract a global feature, called shape context, for each corresponding point. The matching between corresponding points is equivalent to the matching between the corresponding shape context features. The matching of two shapes is done by matching two context maps of the shapes, which is a matrix-based matching. It minimizes the total cost of matching between one context matrix and all the permutations of another context matrix. In order to reduce the matching overhead, it samples the boundary at a number of points and uses the shortest augmenting path algorithm for the matrix matching. The authors tested their system on abundant experiments: digit recognition with 0.63% error rate (60,000 training and 10,000 test digits), 3D object Recognition with 2.4% error rate (22 common household objects, each object was photographed of 72 views), MPEG-7 shape silhouette database with 76.51% retrieval rate (1400 images of 70 groups), trademark retrieval with visual query results (8 test images out of 300 trademarks). Unfortunately, it relies on good segmentation to provide the needed shape information prior they can perform as claimed.

On the other hand, the proposed approach, ALSBIR, has taken a new and fresh

idea to explore a generalized framework for object / image interpretation. The experimental results are very encouraging and comparable to other work that placed more restrictions to simplify the problem.

9.5 Summary

This chapter has displayed the retrieval results by the proposed method on six sets of images and made comparisons with existing methods, i.e. 6NN, L1-norm and L2-norm in terms of precision-recall curves. The proposed method was observed to perform much better than these methods especially on occluded objects and mixed multi-object query. Moreover, to the best of our knowledge, the proposed mixed-object query is the first of its kind to experiment with in shape retrieval. Based on the encouraging results, the future work can focus on investigating distinct 3D object shape information further and on exploring various natural objects, e.g. domestic/farm animals, fish, or bird to generalize the proposed concepts.

Chapter 10 Conclusion and Future Work

Shape retrieval in cluttered environment is of great interest. Particularly, an ideal technique that satisfies the requirements of perceptual similarity measure on partial query, and can overcome dimensionality curse as well as function in adverse environment, is a challenging task. A novel and unique local structure based image retrieval approach has been proposed in this work for identification of input multiple/occluded objects and retrieval of structurally similar models from large database according to the input. The investigation included two major parts:

1. Effective shape representation: a technique has been found to compose local structure that can be consistently extracted when enough or distinguishable portion of the object can be observed. Sufficient and invariant features of each local structure have been derived to represent each structure.
2. Efficient shape retrieval: a indexing and voting scheme have been designed to identify objects from input, and retrieve similar models with respect to the input. It has the potential to cope with the “*dimensional curse*”. The search time is linearly proportional to the input complexity and the size of the image database.

10.1 Work Summary

We have built one prototype system to demonstrate and investigate the proposed scheme. The implementation employed line and arc as primitives. The local structure construction is guided by Gestalt Laws to group five (or more) primitives from an object to construct one salient structure. This is followed by a statistical approach to accept a local structure that have half of the pre-determined primitives, i.e. a reference primitive plus half of its supporting primitives. Each local structure is represented by a set of pair-wise geometric relationships, which are simple, invariant to translation / scaling / rotation, relatively robust to end-point-erosion, and sufficient for discrimination. A fast indexing mechanism has been built to store and retrieve local structures and no expensive training is needed. The prototype system was found to be able to tackle cluttered environment and partially occluded objects. The performance degrades gracefully with respect to the amount of information that is being occluded or mixed. The system has been tested on 2D logo database with thousands of images. Excellent retrieval results have been obtained. The success of the system supports the idea that machine can interpret partially occluded objects in cluttered environment and retrieve similar models to match to the input.

10.2 Contributions

The main contribution of the work is its capability to tackle the need of shape retrieval in cluttered environment. Furthermore, the proposed approach has taken a new and fresh idea to explore a generalized framework for object / image interpretation. The experimental results are very encouraging and comparable to other work that placed more restrictions to simplify the problem. The following are contributions made in this thesis:

1. A novel prototype system has been built up, first of its kind, to tackle the need of “shape retrieval in cluttered environment”.
2. Line and arc are adopted as primitives due to their efficient representation, low storage demand and consistency in being detected. In addition, we do not assume well identified object outline or boundary.
3. Human can identify and recognize an occluded object from a cluttered environment, provided that enough or distinguishable portion of the object can be observed. This is an important observation (assumption) for us to model unique local structure that can be consistently extracted. Here, we search for technique that can extract unique local structure when enough or distinguishable portion of the object can be observed.
4. The local structure construction is guided by the Gestalt Laws to group five (or less) primitives from an object to construct salient structure that can be

extracted consistently. In general, this is difficult to satisfy in occluded / cluttered environment when the construction is affected by the background object, distortion and segmentation error. We employ a statistical approach and accept unique structure that can be more consistently extracted when a reference primitive needs to gather only half of its supports (neighbors) from the same object to form a local structure.

5. Each local structure is represented by a set of pair-wise geometric relationships, which are simple, invariant to translation / scaling / rotation, relatively robust to end-point-erosion, and sufficient for discrimination.
6. The local structures of models are inserted into a hypercube indexing structure. No training is required and new models can be inserted efficiently.
7. An in-depth systematic study of the proposed system has been carried out to understand the characteristics of the shape retrieval. The tolerance and ability to overcome various adverse environments have been investigated and tested. The findings provide valuable directions and hints for the system development in future work.

10.3 Future Work

Future work will explore potential of the proposed scheme on scene interpretation; especially on complicate 3D objects identification to generalize the proposed concepts. We have obtained some fair results in experiment 9.3.6 which

exposed the problems of shadow and unpredictable 3D viewing angle. In future, the color or texture features can be incorporated into the proposed system to enrich object representations. This will be useful for adapting the proposed system to wider applications, e.g. searching in nature scenes. The retrieval mechanism to cope with information from user feedback can also be investigated to bridge semantic gap in the future.

10.4 Concluding remark

This thesis has made a significant contribution in design of a novel prototype system, first of its kind, to tackle the need of “shape retrieval in cluttered environment”. Accordingly, a novel shape representation and a novel indexing mechanisms are proposed. An in-depth study of the system is carried out and a thorough demonstration has been made on the techniques. Future researches are suggested based on the current work. The effectiveness and efficiency can be far-reaching with unforeseen potential.

References

- [1] H.-K. Kim and J.-D. Kim, "Region-based shape descriptor invariant to rotation, scale and translation," *Signal Processing: Image Communication*, vol. 16, pp. 87-93, 2000.
- [2] F. Mokhtarian and A. K. Mackworth, *The curvature scale space representation: theory, applications and MPEG-7 standardization*. Kluwer Academic Publishers, 2002.
- [3] S. Berretti, A. Del Bimbo, and P. Pala, "Retrieval by shape similarity with perceptual distance and effective indexing," *IEEE Trans. Multimedia*, vol. 2, pp. 225-239, 2000.
- [4] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape context," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, pp. 509-522, 2002.
- [5] D. H. Kim, I. D. Yun, and S. U. Lee, "A new shape decomposition scheme for graph-based representation," *Pattern Recognition*, vol. 38, pp. 673-689, 2005.
- [6] B. Huet and E. R. Hancock, "Line pattern retrieval using relational histograms," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, pp. 1363-1370, 1999.
- [7] G. Lu, "Techniques and data structures for efficient multimedia retrieval based on similarity," *IEEE Trans. Multimedia*, vol. 4, pp. 372-384, 2002.
- [8] Y. Chi and M. K. H. Leung, "A Local Structure Matching Approach for Large Image Database Retrieval," *International Conference image Analysis and Recognition (ICIAR) 2004*, pp. 761-768, 2004.
- [9] Y. Chi and M. K. H. Leung, "ALSBIR: a local structure based image retrieval," *Pattern Recognition*, vol. 40, pp. 244-261, 2007.
- [10] Y. Chi and M. K. H. Leung, "Part based object retrieval in cluttered environment," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, pp. 890-895, 2007.
- [11] A. Jain and A. Vailaya, "Shape-based retrieval: a case study with trademark image databases," *Pattern Recognition*, vol. 31, pp. 1369-1390, 1998.
- [12] R. Mehrotra and J. E. Gary, "Similar-shape retrieval in shape data management," *Computer*, vol. 28, pp. 57-62, 1995.

- [13] J. Iivarinen and A. Visa, "Shape recognition of irregular object," *Intelligent Robots and Computer Vision XV: Algorithms, Techniques, Active Vision, and Materials Handling*, pp. 25-32, 1996.
- [14] I. S. Okatani and A. Sugimoto, "Range image registration preserving local structures of object surfaces," *IEEE 17th International Conference on Pattern Recognition (ICPR)*, vol. 2, pp. 224-228, 2004.
- [15] P. J. Giblin and B. B. Kimia, "On the intrinsic reconstruction of shape from its symmetries," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, pp. 895-910, 2003.
- [16] H. Ki, J. Shin, and J. Paik, "Wavelet transform-based hierarchical active shape model for object tracking," *Proceedings of 2004 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2004)*, pp. 256-261, 2004.
- [17] B. Rieger and L. J. v. Vliet, "Curvature of n-dimensional space curves in grey-value images," *IEEE Trans. Image Processing*, vol. 11, pp. 738-745, 2002.
- [18] T. B. Sebastian and B. B. Kimia, "Metric-based shape retrieval in large databases," *Proceedings of 16th International Conference on Pattern Recognition*, vol. 3, pp. 291-296, 2002.
- [19] Y. Gdalyahu and D. Weinshall, "Flexible syntactic matching of curves and its application to automatic hierarchical classification of silhouettes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, pp. 1312-1328, 1999.
- [20] S. Gold, A. Rangarajan, C.-P. Lu, S. Pappu, and E. Mjolsness, "New algorithms for 2D and 3D point matching: pose estimation and correspondence," *Pattern Recognition*, vol. 31, pp. 1019-1031, 1998.
- [21] A. D. Bimbo and P. Pala, "Visual image retrieval by elastic matching of user sketches," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, pp. 121-132, 1997.
- [22] C.-H. Teh and R. T. Chin, "On image analysis by the methods of moments," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 10, pp. 496-513, 1988.
- [23] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin, "The QBIC project: querying image by content using color, texture and shape," *SPIE Storage and Retrieval for Image and Video Databases*, vol. 1908, pp. 173-187, 1993.
- [24] T. Cootes, D. Cooper, C. Taylor, and J. Graham, "Active shape models-their training and application," *Computer Vision and Image Understanding*, vol. 61, pp. 38-59, 1995.
- [25] M. A. Nascimento, V. Sridhar, and X. Li, "Effective and efficient region-based image retrieval," *Journal of Visual Languages and Computing*, vol. 14, pp.

- 151-179, 2003.
- [26] M. Novotni and R. Klein, "Shape retrieval using 3D Zernike descriptors," *Computer-Aided Design*, vol. 36, pp. 1047-1062, 2004.
 - [27] D. Zhang and G. Lu, "Shape-based image retrieval using generic Fourier descriptor," *Signal Processing: Image Communication*, vol. 17, pp. 825-848, 2002.
 - [28] C. Zahn and R. Roskies, "Fourier descriptors for plane closed curves," *IEEE Trans. Computers*, vol. 21, pp. 269-281, 1972.
 - [29] E. Persoon and K. Fu, "Shape discrimination using Fourier descriptors," *IEEE Trans. Systems, Man and Cybernetics*, vol. 7, pp. 170-179, 1977.
 - [30] Q. M. Tieng and W. W. Boles, "Recognition of 2-D object contours using the wavelet transform zero-crossing representation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, pp. 910-916, 1997.
 - [31] H. S. Yang, S. U. Lee, and K. M. Lee, "Recognition of 2D object contours using starting-point-independent wavelet coefficient matching," *Journal of Visual Communication and Image Representation*, vol. 9, pp. 171-181, 1998.
 - [32] J. MacLean, R. Herpers, C. Pantofaru, L. Wood, K. Derpanis, D. Topalovic, and J. Tsotsos, "Fast hand gesture recognition for real-time teleconferencing applications," *Proceedings of IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, pp. 133-140, 2001.
 - [33] X. Zabulis, J. Sporring, and S. C. Orphanoudakis, "Perceptually relevant and piecewise linear matching of silhouettes," *Pattern Recognition*, vol. 38, pp. 75-93, 2005.
 - [34] M. K. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Inf. Theory IT-8*, pp. 179-187, 1962.
 - [35] B. S. Manjunath, *Introduction to MPEG-7 : multimedia content description interface*: Wiley, 2002.
 - [36] M. Peura and J. Iivarinen, "Efficiency of simple shape descriptors," *Proceedings of the Third International Workshop on Visual Form*, vol. 5, pp. 443-451, 1997.
 - [37] E. R. Davies, *Machine Vision: Theory, Algorithms, Practicalities*. New York: Academic Press, 1997.
 - [38] P. J. v. Otterloo, *A contour-oriented approach to shape analysis*. Englewood Cliffs, NJ: Prentice-Hall International (UK) Ltd, 1991.
 - [39] W. N. Lie and Y. C. Chen, "Shape representation and matching using polar signature," *Proceedings of International Computer Symposium*, pp. 710-718, 1986.
 - [40] P. F. Zhu and P. M. Chirlian, "On critical point detection of digital shapes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, pp. 737-748,

- 1995.
- [41] K. B. Eom, "Contour analysis using time-varying autoregressive model," *Proceedings of international Conference on Image Processing*, pp. 891-894, 2000.
 - [42] R. H. Glendinning, "Singal plus noise models in shape classification," *Pattern Recognition*, vol. 27, pp. 777-784, 1994.
 - [43] S. S. Xiong, Z. Y. Zhou, L. M. Zhong, and T. H. Cui, "Complex nonlinear exponential autoregressive model for shape recognition using neural network," *Proceedings of IEEE International Conference on Instrumentation and Measurement Technology*, vol. 1, pp. 289-292, 1998.
 - [44] H. Asada and M. Brandy, "the curvature primal sketch," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, pp. 2-14, 1986.
 - [45] F. Mokhtarian and A. Mackworth, "Scale-based description and recognition of planar curves and two-dimensional shapes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, pp. 34-43, 1986.
 - [46] F. Mokhtarian and A. K. Mackworth, "A theory of multiscale, curvature-based shape representation for planar curves," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, pp. 789-805, 1992.
 - [47] S. Abbasi, F. Mokhtarian, and J. Kitter, "Curvature scale space image in shape similarity retrieval," *Multimedia Systems*, vol. 7, pp. 467-476, 1999.
 - [48] S. Abbasi, F. Mokhtarian, and J. Kittler, "Enhancing CSS-based shape retrieval for objects with shallow concavities," *Image and Vision Computing*, vol. 18, pp. 199-211, 2000.
 - [49] S. J. (Ed.), "MPEG-7 Visual part of experimentation model version 5.0," in *ISO/IEC/JTC1/SC29/WG11/N3321*, 2000.
 - [50] M. Daoudi and S. Matusiak, "Visual image retrieval by multiscale description of user sketches," *Journal of Visual Languages and Computing*, vol. 11, pp. 287-301, 2000.
 - [51] D. H. Eberly, "Geometric methods for analysis of ridges in N-dimensional images," Ph.D. Thesis, University of North Carolina at Chapel Hill, 1994.
 - [52] A. P. Witkin, "Scale-space filtering," *Proc. International Conference on Artificial Intelligence*, pp. 1019-1022, 1983.
 - [53] T. Lindeberg, "Scale-space for discrete signals," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, pp. 234-254, 1990.
 - [54] D. Lowe, "Distinctive image features from scale-invariant keypoints," *international Journal of Computer Vision*, vol. 60, pp. 91-110, 2004.
 - [55] M. Sonka, V. Hlavac, and R. Boyle, *Image processing, analysis and machine vision*. London, UK, NJ: Chapman & Hall, 1993.
 - [56] G. Ostrouchov and N. F. Samatova, "On FastMap and the convex hull of

- multivariate data: toward fast and robust dimension reduction," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, pp. 1340-1343, 2005.
- [57] Z. Yang and F. S. Cohen, "Image registration and object recognition using affine invariants and convex hulls," *IEEE Trans. Image Processing*, vol. 8, pp. 934-946, 1999.
- [58] N. Ezer and E. Anarim, "A comparative study of moment invariants and Fourier descriptors in planar shape recognition," *Proceedings of the 7th Electrotechnical Conference on Mediterranean*, vol. 1, pp. 242-245, 1994.
- [59] C. C. Lin and R. Chellappa, "Classification of partial 2-D shapes using Fourier descriptors," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 9, pp. 686-690, 1987.
- [60] O. R. Michell and T. A. Grogan, "Global and partial shape discrimination for computer vision," *Opt. Eng.*, vol. 23, pp. 484-491, 1984.
- [61] K. Arbter, W. E. Snyder, H. Burkhardt, and G. Hirzinger, "Application of affine-invariant Fourier descriptors to recognition of 3-D objects," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, pp. 640-647, 1990.
- [62] G. Granlund, "Fourier preprocessing for hand print character recognition," *IEEE Trans. Computers*, vol. 21, 1972.
- [63] T. W. Rauber, "Two-dimensional shape description," University Nova de Lisboa, Portugal, Technical Report: GR UNINOVA-RT-10-94, 1994.
- [64] P. L. Rosin, "Measuring shape: ellipticity, rectangularity, and triangularity," *Machine Vision and Applications*, vol. 14, pp. 172-184, 2003.
- [65] M. R. Teague, "Image analysis via the general theory of moments," *J. Opt. Soc. Amer.*, vol. 70, pp. 920-930, 1980.
- [66] A. Goshtasby, "Description and discrimination of planar shapes using shape matrices," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 7, pp. 738-743, 1985.
- [67] A. Taza and C. Suen, "Discrimination of planar shapes using shape matrices," *IEEE Trans. Systems, Man and Cybernetics*, vol. 19, pp. 1281-1289, 1989.
- [68] S. Loncaric, "A survey of shape analysis techniques," *Pattern Recognition*, vol. 31, pp. 983-1001, 1998.
- [69] S. Parui, E. Sarma, and D. Majumder, "How to discriminate shapes using the shape vector," *Pattern Recognition Letters*, vol. 4, pp. 201-204, 1986.
- [70] H. Freeman, "On the encoding of arbitrary geometric configurations," *IRE Trans. Electron. Comput.*, vol. 10, pp. 260-268, 1961.
- [71] J. Hu and H. Yan, "Polygonal approximation of digital curves based on the principles of perceptual organization," *Pattern Recognition*, vol. 30, pp. 701-718, 1997.
- [72] P. I. Hosur and K. K. Ma, "A novel scheme for progressive polygon

- approximation of shape contours," *Proceedings of IEEE 3rd workshop on Multimedia Signal Processing*, pp. 309-314, 1999.
- [73] B. Huet and E. R. Hancock, "Relational object recognition from large structural libraries," *Pattern Recognition*, vol. 35, pp. 1895-1915, 2002.
- [74] Y. Chi and M. K. H. Leung, "ALSBIR:a local structure based image retrieval," *Pattern Recognition (accepted)*, 2005.
- [75] T. Pavlidis, "Algorithm for shape analysis of contours and waveforms," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 2, pp. 301-312, 1980.
- [76] S. Berretti, A. D. Bimbo, and P. Pala, "Efficient shape retrieval by parts," *Computer Analysis of Images and Patterns*, pp. 57-64, 1999.
- [77] S. Z. Li, "Shape matching based on invariants," in *Shape Analysis, Progress in Neural Networks*, O. Omidvar, Ed. Ablex, Norwood, NJ, 1999, pp. 203-228.
- [78] C.-L. Huang and D.-H. Huang, "A content-based image retrieval system," *Image and Vision Computing*, vol. 16, pp. 149-163, 1998.
- [79] D. M. Squire and T. M. Caelli, "Invariance signature: characterizing contours by their departures from invariance," *Computer Vision and Image Understanding*, vol. 77, pp. 284-316, 2000.
- [80] P. Maragos, "Pattern Spectrum and Multiscale Shape Representation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, pp. 701-716, 1989.
- [81] J. Serra, *Image analysis and mathematical morphology*. New York: Academic, 1982.
- [82] A. Loui, A. Venetsanopoulos, and K. Smith, "Two-dimensional shape representation using morphological correlation functions," *Proceedings of International Conference on ASSP*, pp. 2165-2168, 1990.
- [83] H. Blum, "A transformation for extracting new descriptors of shape," in *Models for the perception of Speech and Visual Forms*, W. Whaten-Dunn, Ed. Cambridge, MA: MIT Press, 1967, pp. 362-380.
- [84] W. Niblack, "Updates to the QBIC system," *Proceedings of International Conference on Storage and Retrieval for Image and Video Databases VI, SPIE Proceedings*, vol. 3312, pp. 150-161, 1998.
- [85] M. Flickner, "Query by image and video content: The QBIC system," *Computer*, vol. 9, pp. 23-32, 1995.
- [86] J. R. Smith and S.-F. Chang, "Visually searching the web for content," *IEEE Multimedia*, pp. 12-19, 1997.
- [87] M. S. Lew, "Next-generation web searches for visual content," *IEEE Computer*, vol. 33, pp. 46-53, 2000.
- [88] A. Franco, A. Lumini, and D. Maio, "Eigenspace merging for model updating," *Proceedings of 16th International Conference on Pattern Recognition*, vol. 2, pp. 156-159, 2002.

- [89] Y. Yamashita, Y. Ikeno, and H. Ogawa, "Relative Karhunen-Loeve transform method for pattern recognition," *Proceedings of Fourteenth International Conference on Pattern Recognition*, vol. 2, pp. 1031-1033, 1998.
- [90] V. Popovici and J. P. Thiran, "PCA in autocorrelation space," *IEEE Conference on Pattern Recognition*, vol. 2, pp. 132-135, 2002.
- [91] S. K. Chang, Q. Y. Shi, and C. W. Yan, "Content based image retrieval using optimum peano scan," *IEEE Conference on Pattern Recognition*, vol. 4, pp. 130-133, 2002.
- [92] R. Elmasri and A. B. Navathe, *Fundamentals of database systems*. Redwood City, CA: Benjamin Cummings, 1994.
- [93] 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.
- [94] 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.
- [95] 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.
- [96] S. Sclaroff, L. Taycher, and M. L. Cascia, "Imager over: a content base Image browser for the world wide web," *Proc. Workshop Content-based Access to Image and Video Libraries*, pp. 1000-1006, 1997.
- [97] 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.
- [98] M. Charikar, C. Chekur, T. Feder, and R. Motwani, "Incremental clustering and dynamic information retrieval," *The 29th Annual ACM Symposium on Theory of Computing*, pp. 626-635, 1997.
- [99] A. Berman and L. G. Shapiro, "Efficient retrieval with multiple distance measures," *Conference of Storage and Retrieval for Image and Video Database*, vol. 3022, pp. 12-21, 1997.
- [100] R. Ng and A. Sedighian, "Evaluating multi-dimensional indexing structures for images transformed by principal component analysis," *Proceedings of SPIE Storage and Retrieval for Image and Video Databases*, 1996.
- [101] C. Faloutsos and K.-I. Lin, "Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets," *Proceedings of SIGMOD*, pp. 163-174, 1995.
- [102] 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.

- [103] 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.
- [104] A. Vellaikal and C.-C. J. Kuo, "Hierarchical clustering techniques for image database organization and summarization," *SPIE Conf. Multimedia Storage and Archiving System III*, vol. 3527, pp. 68–79, 1998.
- [105] R. Elmasri and A. B. Navathe, *Fundamentals of database systems*, 2nd ed. Redwood City, CA: Benjamin Cummings, 1994.
- [106] S. Dao, Q. Yang, and A. Vellaikal, "MB⁺-tree: An index structure for content-based retrieval," in *Multimedia Database Systems: Design and Implementation Strategies*, K. C. Nwosu, B. Thuraisingham, and P. B. Berra, Eds. Norwell, MA: Kluwer, 1996.
- [107] C. A. Shaffer, *A practical introduction to data structures and algorithm analysis*. NJ: Prentice-Hall, 1997.
- [108] O. Guttman, "R-tree: A dynamic index structure for spatial searching," *ACM SIGMOD International Conference on Management of Data*, pp. 47–57, 1984.
- [109] T. Sellis, N. Roussopoulos, and C. Faloutsos, "the R⁺-tree: a dynamic index for multi-dimensional objects," *The 13th Conference on Very Large Database (VLDB)*, pp. 507–518, 1987.
- [110] 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 Conference on Management of Data*, pp. 322–331, 1990.
- [111] D. A. White and R. Jain, "Similarity indexing with the SS-tree," *the 12th IEEE International Conference on Data Engineering (ICDE)*, pp. 516–523, 1996.
- [112] R. Kurniawati, J. S. Jin, and J. A. Shepard, "SS⁺-tree: An improved index structure for similarity searches in a high-dimensional feature space," *Conference on Storage and Retrieval for Image and Video Database V*, vol. 3022, pp. 110–120, 1997.
- [113] S. Berchtold, D. A. Keim, and H.-P. Kriegel, "The X-tree: an index structure for high-dimensional data," *Proceedings of the 22nd International Conference on Very Large Databases*, pp. 28–39, 1996.
- [114] 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.
- [115] P. Ciaccia, M. Patella, and P. Zezula, "M-tree: An efficient access method for similarity search in metric spaces," *Proceedings of 23rd International Conference on Very Large Data Bases (VLDB'97)*, pp. 426–435, 1997.
- [116] S. Berchtold, C. Bohm, and H. P. Kriegel, "The pyramid-technique: towards breaking the curse of dimensionality," *ACM SIGMOD International Conference on Management of Data*, pp. 142–153, 1998.

- [117] J. Nievergelt, H. Hinterberger, and K. C. Sevcik, "The grid file: An adaptive symmetric multikey file structure," *ACM Trans. Database Syst.*, vol. 9, 1984.
- [118] A. Sajjanhar and G. Lu, "Indexing 2-D nonoccluded shape for similarity retrieval," *SPIE Conf. Applications of Digital Image Processing XX*, vol. 3164, pp. 188–197, 1997.
- [119] A. Berman and L. G. Shapiro, "Efficient retrieval with multiple distance measures," presented at Conf. Storage and Retrieval for Image and Video Databases V, San Jose, CA, 1997.
- [120] 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.
- [121] H. Tagare, "Increasing retrieval efficiency by index tree adaption," *Proceedings of IEEE Workshop on Content-based Access of Image and Video Libraries, in Conjunction with IEEE CVPR'97*, 1997.
- [122] Y. Gong, *Intelligent image databases-towards advanced image retrieval*. Boston: Kluwer Academic Publishers, 1998.
- [123] S. Antani, R. Kasturi, and R. Jain, "A survey on the use of pattern recognition methods for abstraction, indexing and retrieval of images and video," *Pattern Recognition*, vol. 35, pp. 945–965, 2002.
- [124] Y. Rui, T. S. Huang, and S.-F. Chang, "Image retrieval: current techniques, promising directions, and open issues," *Journal of Visual Communication and Image Representation*, vol. 10, pp. 39–62, 1999.
- [125] K. Chakrabarti, M. Ortega-Binderberger, K. Pokaew, and S. Mehrotra, "Similar shape retrieval in MARS," *IEEE International Conference on Multimedia and Expo, 2000. ICME 2000*, vol. 2, pp. 709–712, 2000.
- [126] K. Pokaew, M. Ortega, and S. Mehrotra, "Query reformulation for content based multimedia retrieval in MARS," *IEEE International Conference on Multimedia Computing Systems*, vol. 2, pp. 747–751, 1999.
- [127] T. Gevers and A. W. M. Smeulders, "Pictoseek: combining color and shape invariant features for image retrieval," *IEEE Trans. Image Processing*, vol. 9, pp. 102–119, 2000.
- [128] Z.-N. Li, O. R. Zaiane, and B. Yan, "C-BIRD: content-based image retrieval from digital libraries using illumination invariance and recognition kernel," *Proceedings of 9th International Workshop on Database and Expert Systems Applications*, vol. 9, pp. 361–366, 1998.
- [129] Z.-N. Li, O. R. ZaTiane, and Z. Tauber, "Illumination invariance and object model in content-based image and video retrieval," *Journal of Visual Communication and Image Representation*, vol. 10, pp. 219–244, 1999.
- [130] D. Zhong and S.-F. Chang, "An integrated approach for content-based video

- object segmentation and retrieval," *IEEE Trans. Circuits Systems Video Technology*, vol. 9, pp. 1259–1268, 1999.
- [131] C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik, "Blobworld: A system for region-based image indexing and retrieval," *The 3rd International Conference on Visual Information and Information Systems (VISUAL'99)*, vol. 1614, pp. 509–516, 1999.
- [132] J. K. Wu, B. M. Mehtre, C. P. Lam, A. D. Narasimhalu, and J. J. Gao, "Core: a content-based retrieval engine for multimedia information systems," *Multimedia Systems*, vol. 3, pp. 25-41, 1995.
- [133] J. R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, and C. F. Shu, "the Virage image search engine: an open framework for image management," *SPIE Conf. Storage and Retrieval for Image and Video Databases IV*, vol. 2670, pp. 76-87, 1996.
- [134] J. Dowe, "Content-based retrieval in multimedia imaging," *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1993.
- [135] W. Ma and B. Manjunath, "Netra: A toolbox for navigating large image databases," *IEEE International Conference on Image Processing*, pp. 568--571, 1997.
- [136] M. A. Smith and T. Kanade, "Video skimming for quick browsing based on audio and image characterization," *Technical Report CMU-CS-95-186 Carnegie Mellon University*, 1995.
- [137] V. Ogle and M. Stonebraker, "Chabot: retrieval from a relational database of images," *IEEE Computer*, vol. 28, pp. 40-48, 1995.
- [138] S.-F. Chang, J. R. Smith, M. Beigi, and A. Benitez, "Visual information retrieval from large distributed online repositories," *Commun. ACM*, vol. 40, pp. 62-71, 1997.
- [139] Y. Gao and M. K. H. Leung, "Face recognition using line edge map," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, pp. 764-779, 2002.
- [140] R. Nevatia and K. R. Babu, "Linear feature extraction and description," *Computer Graphics and Image Processing*, vol. 13, pp. 257-269, 1980.
- [141] M. K. Leung and Y. H. Yang, "Dynamic two-strip algorithm in curve fitting," *Pattern Recognition*, vol. 23, pp. 69-79, 1990.
- [142] M. D. Heath, S. Sarkar, T. Sanocki, and K. W. Bowyer, "Comparison of edge detectors: a methodology and initial study," *Computer Vision and Image Understanding*, vol. 69, pp. 38-54, 1998.
- [143] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, pp. 4-37, 2000.
- [144] I. Biederman and J. Gu, "Surface versus edge-based determinants of visual recognition," *Cognitive Psychology*, vol. 20, pp. 38-64, 1988.

- [145] F. Attneave, "Some informational aspects of visual perception," *Psychol. Rev.*, vol. 61, pp. 183-193, 1954.
- [146] L. Zusne, *Visual perception of form*. New York: Academic Press, 1970.
- [147] F. Attneave and M. D. Arnoult, "the quantitative study of shape and pattern perception," *Pattern Recognition*, pp. 123-141, 1966.
- [148] D. Geiger, T.-L. Liu, and R. V. Kohn, "Representation and self-similarity of shapes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, pp. 86-99, 2003.
- [149] E. Saber, Y. Xu, and A. M. Tekalp, "Partial shape recognition by sub-matrix matching for partial matching guided image labeling," *Pattern Recognition*, vol. 38, pp. 1560-1573, 2005.
- [150] K. Koffka, *Principles of Gestalt Psychology*. New York: Harcourt, Brace and company, 1935.
- [151] S. Sarkar and K. L. Boyer, "Perceptual organization in computer vision: a review and a proposal for a classificatory structure," *IEEE Trans. Systems, Man and Cybernetics*, vol. 23, pp. 382-399, 1993.
- [152] M. Fisher and K. Smith-Gratto, "Gestalt theory: a foundation for instructional screen design," *Journal of Educational Technology Systems*, vol. 27, pp. 361-371, 1998-99.
- [153] D. Chang, L. Dooley, and J. E. Tuovinen, "Gestalt Theory in Visual Screen Design-A New Look at an Old Subject," presented at the Seventh World Conference on Computers in Education, Copenhagen, Denmark, 2001.
- [154] D. Lauer, *Design basics*. New York: Holt, Reinhart and Winston, 1979.
- [155] A. M. Finch, R. C. Wilson, and E. R. Hancock, "Matching delaunay graphs," *Pattern Recognition*, vol. 30, pp. 123-140, 1997.
- [156] Y. S. Kim and W. Y. Kim, "Content-based trademark retrieval system using a visually salient feature," *Image and Vision Computing*, vol. 16, pp. 931-939, 1998.

Appendix A 3-D stationery model images

In the following pages, the real object model images used for the experiments in this work are shown. Each logo is assigned a unique name, and is shown with its dimensions. For instance, $200 \times 300 \times 256$ indicates an image of dimensions of 200×300 pixels and 256 gray scales.

Appendix B Results of shape retrieval

In the following pages, the results of shape retrieval are shown.

1 Results for letter queries

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10
A	A	A	ʌ	Λ	ʌ	A	A	∟	∠	∟
A	A	A	A	ʌ	ʌ	Λ	A	∟	ʌ	V
B	ʌ	A	A	B	A	ʌ	A	ʌ	Λ	P
B	B	B	B	B	B	P	⊞	R	D	ʌ
A	A	A	ʌ	Λ	ʌ	A	ʌ	A	∟	Λ
ʌ	ʌ	Λ	ʌ	A	A	A	ʌ	∟	Λ	∟
ʌ	ʌ	A	A	V	Λ	ʌ	∟	∠	ʌ	v
A	A	ʌ	ʌ	A	A	A	∟	ʌ	V	Λ
ʌ	ʌ	ʌ	Λ	A	A	A	∟	ʌ	Λ	↗
A	A	A	ʌ	Λ	ʌ	A	A	ʌ	∟	V

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10
<	<	┐	┌	└	≡	↗	≡	L	/	√
L	L	└	┐	┌	<	/	≡	≡	E	≡
┐	┐	L	└	┌	<	≡	E	↗	/	≡
M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M
M	M	M	M	M	M	M	M	M	M	M
N	N	N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N	N	N
O	O	O	O	O	O	O	O	O	O	O
P	P	P	P	P	P	P	P	P	P	P
P	P	P	P	P	P	P	P	P	P	P
P	P	P	P	P	P	P	P	P	P	P
Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10
R	R	R	R	R	B	✓	P	B	B	✓
✓	✓	✓	R	✓	R	R	✓	B	P	R
✓	✓	R	✓	✓	✓	✓	✓	R	✓	P
S	S	S	S	S	S	S	G	G	G	G
S	S	S	S	S	S	S	U	U	U	U
S	S	S	S	G	S	S	S	H	Q	C
T	T	T	T	T	✓	✓	✓	✓	T	/
T	T	T	T	T	✓	✓	✓	✓	T	/
✓	✓	✓	T	✓	✓	/	✓	I	I	I
U	U	U	U	U	U	U	U	U	J	E
U	U	U	U	U	U	U	U	U	E	U
U	U	U	U	U	U	U	U	E	U	U
U	U	U	U	U	U	U	U	E	U	U
V	V	V	V	M	✓	✓	✓	✓	✓	V
✓	✓	✓	✓	V	✓	✓	✓	✓	✓	N
✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	A

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10

2 Results for 3D distortion queries

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10

3 Results for similarity based queries

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10

4 Results for partial queries

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10

5 Results for Mixed object queries

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10

6 Results for 3D real object image queries

Query	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9