Fingerprint retrieval based on database clustering

Liu, Manhua

2008

Fingerprint Retrieval Based on Database Clustering

Liu Manhua

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University

in fulfillment of the requirements for the degree of

Doctor of Philosophy

2007

Statement of Originality

I hereby certify that the content of this thesis is the result of work done by me and has not been submitted for a higher degree to any other University or Institution.

....

Date Liu Manhua

Acknowledgments

I wish to express my sincere gratitude to all the people who have assisted me during the years of my postgraduate study at Nanyang Technological University. I am most grateful to my supervisors, Prof. Alex Kot and Asst. Prof. Xudong Jiang, for their professional guidance, useful discussions, constant support and trust. Their insightful comments have been an inspiration for my Ph.D research work.

I would like to specially thank the staffs in Workstation Resource Lab., Steward Chu and Tan Pengping, for their assistance and technical support. I would also like to thank all my friends in Workstation Resource Lab, Yang Huijuan, Cheng Jun, Bappaditya Mandal and Cui Shi etc., who have been very supportive over these years.

At last, my sincere thanks go to all the members of my family. Especially, I am very grateful to my husband, Li Yanggen, for his love, care, understanding and encouragement. I would not have accomplished my Ph.D thesis if there were not their supports.

This thesis is dedicated to my parents and my husband for their never-fading love and support.

Summary

Fingerprints as a kind of human biometric feature have been widely used for personal recognition in the forensic, commercial and civilian areas. A lot of works have been done for the fingerprint verification and identification. Fingerprint identification which requires the search of database for a match is more complex than fingerprint verification. Although good performances have been reported for fingerprint verification in the literature, the accuracy and efficiency of the identification deteriorate seriously if the 1:1 verification algorithm is directly extended to the 1:N fingerprint identification. How to facilitate an efficient and effective search of database is still a great challenge in developing an automatic fingerprint identification system. The multi-level search of fingerprint database is usually employed to incorporate both the coarse and fine level features of fingerprint for the efficient and effective automatic identification. The coarse level search of database is introduced to reduce the search space of time consuming fine matching and alleviate the performance deterioration resulted by the exhaustive fine search in an automatic fingerprint identification system.

The fingerprint classification and indexing techniques have been widely investigated for the coarse level search of fingerprint database. However, most of them still do not meet with the performance requirements of wide applications for fingerprint identification. This thesis investigates the efficient and accurate fingerprint retrieval techniques based on the coarse level features to facilitate the search of fingerprint database in automatic identification. The proposed fingerprint retrieval algorithms are the coarse level search of database independent of the fine matching

algorithm. Hence, they can be coupled with the existing verification algorithm as the fine matcher to design an automatic fingerprint identification system. The following several issues are extensively studied in this thesis to develop the automatic fingerprint retrieval algorithms.

Firstly, some techniques are investigated to robustly and reliably estimate two local parameters of fingerprint: ridge orientation and ridge distance, which play important roles for fingerprint analysis and retrieval. An effective method is proposed to consistently locate a reference point and compute a corresponding reference direction for all types of fingerprints which can be used for the fingerprint alignment. Secondly, we develop a fingerprint retrieval algorithm based on continuous classification that uses the orientation field as the main retrieval feature and the dominant ridge distance as an auxiliary feature. A new distance measure is proposed to quantify more effectively the dissimilarity between two orientation vectors than the traditional measures. A regional feature weighting scheme is introduced on the distance measure to visibly improve the retrieval performance. Thirdly, a multi-prototype clustering algorithm is developed to discover the clusters of arbitrary shape and size. A separation measure is proposed to evaluate how two prototypes are separated by a sparse region. Multiple prototypes with small separation are grouped into a given number of clusters in the agglomerative method. Finally, a clustering-based fingerprint retrieval algorithm is developed to speed up the retrieval process without compromising the retrieval accuracy. A nonuniform spacing of fingerprint by a circular tessellation is proposed to compute a multi-scale orientation field as the main retrieval feature. A modified K-means clustering is proposed to partition the high dimensional orientation feature space into a number of clusters. It outperforms the traditional K-means clustering for the fingerprint retrieval. The proposed fingerprint retrieval algorithms have been evaluated on the NIST fingerprint database which contains a significant number of poor quality fingerprints. The extensive experimental results and comparisons demonstrate the effectiveness and superiority of the proposed approaches.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Biometrics

The information society is developing very fast during the past decades and some electronically interconnected applications are evolving rapidly such as e-commerce, smart cards, cellular phones and electronic banking etc.. Identity fraud is reaching unprecedented proportions so that more emphasis is put on the privacy and security of information stored in various database. Hence, the reliable automatic personal recognition becomes crucial in many daily transactions and access controls. Traditional means of personal recognition are usually based on (i) knowledge such as passwords and Personal Identification Number (PIN) which only the individuals know and (ii) token such as passport, identification (ID) card and licenses which the individuals possess [78, 83]. However, PINs and passwords may be forgotten and passport, ID card and licenses may be forged, stolen or lost. Due to these problems, the traditional means cannot meet with the growing demands of the privacy and security in the information society.

Biometrics-based recognition system recognizes a person based on his/her physiological and/or behavioral characteristics such as fingerprint, face, voice, hand geometry, DNA and iris etc. [78, 83]. With the increasing use of computers as the

vehicles of information technology, automatic personal recognition based on biometrics is a rapidly evolving technology. It outperforms the traditional means of personal recognition in that the biometric features are distinctive, reliable and cannot be lost or forgotten. Thus, the biometrics-based personal recognition systems have been widely investigated and used in the forensic, commercial and civilian areas. For example, they are used to authorize a person to access the ATMs, cellular phones, smart cards, desktop PCs and computer networks etc. and to replace keys in automobile with key-less entry and key-less ignition. Due to the increase of security threats, some countries such as United States of America (USA) have started using the biometrics for border control and national ID cards. The personal recognition system based on single biometric feature may have some limitations such as low reliability. Recently, multiple biometrics are integrated to further improve the performance of the biometric-based recognition systems [41]. Biometrics is becoming a dominant and popular solution of automatic personal recognition for the increasing demands of the privacy and security in the information society.

1.2 Fingerprints

Fingerprints are a kind of human biometrics on the tips of finger. Since the beginning of the 20th century, fingerprints have been extensively used for the identification of criminals by the various forensic departments around the world [1]. Fingerprint-based personal recognition is one of the most mature and proven biometric technologies. With the availability of cheap and compact solid state sensors, fingerprint-based recognition systems are becoming popular in the commercial and civilian areas such as welfare disbursement and laptop computer access. Comparing to other human biometric features, fingerprints have some advantages. Firstly, it has been generally accepted that fingerprints can uniquely identify a person, which gives it a crucial advantage over other biometric features such as DNA, face and voice that cannot distinguish the identical twins. In addition, fingerprints are

fully formed at about seven months of fetus development and finger ridge configurations do not change throughout the life of an individual except due to the accidents such as bruises and cuts on the finger tips [3]. This immutability gives fingerprint another important advantage over other biometric features such as face and voice which will change over lifetime. Finally, fingerprint is easily accessible and its acquisition is inexpensive comparing to other biometrics techniques such as iris and retinal scan. Because of the above advantages: uniqueness, immutability and low cost, fingerprints become one of the most widely used human biometric features for personal recognition in the forensic, commercial and civilian areas. It has the potential to stay as a dominant biometric technique in the future [83].

Fingerprint is composed of the flow patterns of parallel ridges (black) and valleys (white). Figure 1.1 shows an example of fingerprint image. The anomalies of the ridge flows in the local regions vary for different fingerprints to form their individualities. Thus, the position and orientation of these anomalies are usually used to represent fingerprint for personal recognition. To develop a practical personal recognition system based on fingerprints, a first and important issue is to determine how a person is recognized. In general, fingerprint-based recognition systems work in two modes: verification (authentication) and identification [73]. Each mode has its own complexity and can be best solved by some techniques.

Figure 1.1: A fingerprint image.

1.3 Fingerprint Verification

Verification mode answers the question: "Is this person actually who he/she claims to be?". A verification system either accepts or rejects a user's identity by matching the input with the pre-stored one. In fingerprint verification, the user inputs his fingerprint and claims an identity, and the system verifies whether the user's fingerprint is consistent with the claimed identity by comparing the input fingerprint with the one already provided or stored in the database. Figure 1.2 shows the block diagram of a fingerprint verification system. In general, fingerprint verification system consists of two stages: enrollment and verification. In the enrollment stage, each user claims an identity such as name or PIN through a keyboard or keypad and presses his/her finger on a sensor to capture a digital fingerprint image. The captured fingerprint is further processed to extract the feature for its representation which is often called a *template*. Both the representation feature and the claimed identity are stored in the database by the system. During the verification stage, the user claims an identity and the sensor captures the user's input fingerprint image which is further processed in the same way as the enrollment stage. The representation feature of the input fingerprint is compared with the template selected from the database according to the user's identity. The output of fingerprint verification system is a match (accept) or non-match (reject) of the user.

In practice, the output of a fingerprint verification system is a matching score quantifying the similarity between the feature of the input fingerprint and a database template. The larger the matching score is, the more confident the verification system is to determine that two fingerprints are matched (i.e., come from the same finger). The decision of the system to accept or reject the user is determined by a threshold. If the matching score is larger than the threshold, these two fingerprints are matched and the user is accepted to access the system. Otherwise, they are non-matched and the user is rejected. There are two types of errors produced by a fingerprint verification system. One error is the false match of an input finger-

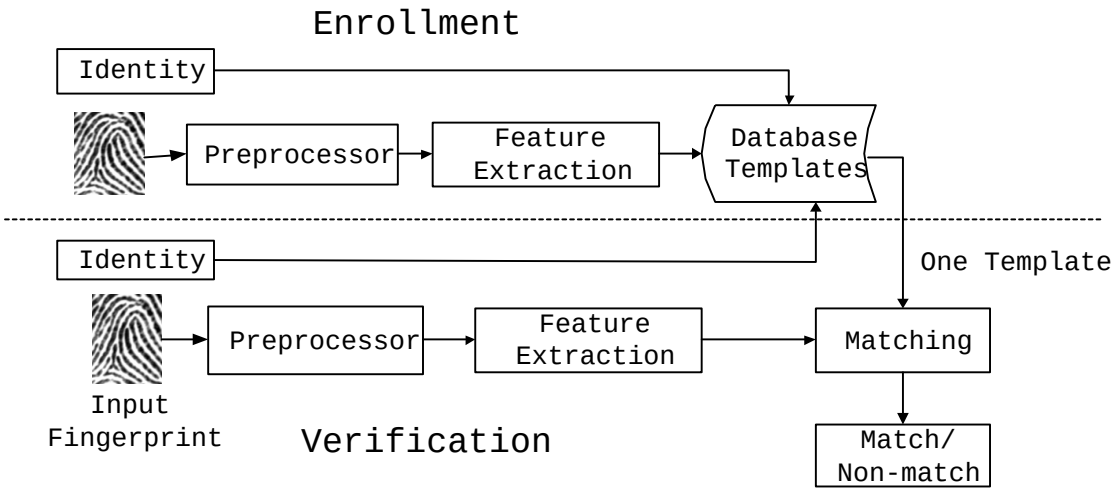


Figure 1.2: The block diagram of a fingerprint verification system

print which does not come from the same finger as the template. Another error is the false non-match of a genuine fingerprint which should be accepted. The false match error rate (FMR) and false non-match error rate (FNMR), which are the probabilities of the false match and non-match errors, respectively, are often used to evaluate the accuracy of fingerprint verification system. A tradeoff exists between FMR and FNMR and can be adapted by adjusting the matching threshold. Since the year 2000, international fingerprint verification competition (FVC) [13] is being held every two years such as FVC2000, FVC2002 and FVC2004. This competition focuses on the assessment of various fingerprint verification algorithms and attracts many researchers and engineers working on fingerprint verification. Results are reported after each competition [70–72]. A more detailed description for the performance evaluation of fingerprint verification algorithms can be found in the literature [20, 70].

With the availability of cheap and compact solid state fingerprint sensors, fingerprint verification becomes popular in the civilian and commercial areas such as the entry and access control systems, identity verification and attendance, etc.. For example, fingerprint verification is employed to verify and grant a person's access to cellular phones, desktop PCs and computer networks in some entry and access control systems.

1.4 Automatic Fingerprint Identification

Identification mode answers the question of "Is this person a member of a specified group?". Fingerprint identification is to establish a user's identity from the fingerprint database of a group. Without the claimed identity information, the query fingerprint of a user needs to be compared with a large number of fingerprints from the database and a match found indicates that the person belongs to the group. It is very labor intensive and may be impractical to manually compare a query fingerprint with thousands of fingerprints in a large database. Automatic fingerprint identification can efficiently identify a person and is very helpful to save the labor cost.

Let N be the number of fingerprints stored in the database. Figure 1.3 shows the block diagram of an automatic fingerprint identification system (AFIS). In general, an AFIS also consists of two stages: enrollment and identification. It differs from the fingerprint verification system in that the user does not provide any identity information in the identification stage. The enrollment stage is similar to that of fingerprint verification system. In the identification stage, a sensor captures the query fingerprint of the user which is further processed to extract the representation feature in the same way as the enrollment stage. Since no identity information of the user is provided, it is usually required to search the fingerprint database for a match of the query fingerprint. The output of fingerprint identification system is either the identity of an enrolled user or an warning message such as "user not identified".

A simple solution to design an AFIS is to directly extend a fingerprint verification algorithm of 1:1 matching to the automatic identification system of $1 : N$ matching. To infer the error rates of fingerprint identification from those of verification, we assume that the query fingerprint is required to match with all the N database templates and only one template is pre-stored in the database for each user. Let $FMR_{1:N}$ and $FNMR_{1:N}$ be the false match and non-match error rates

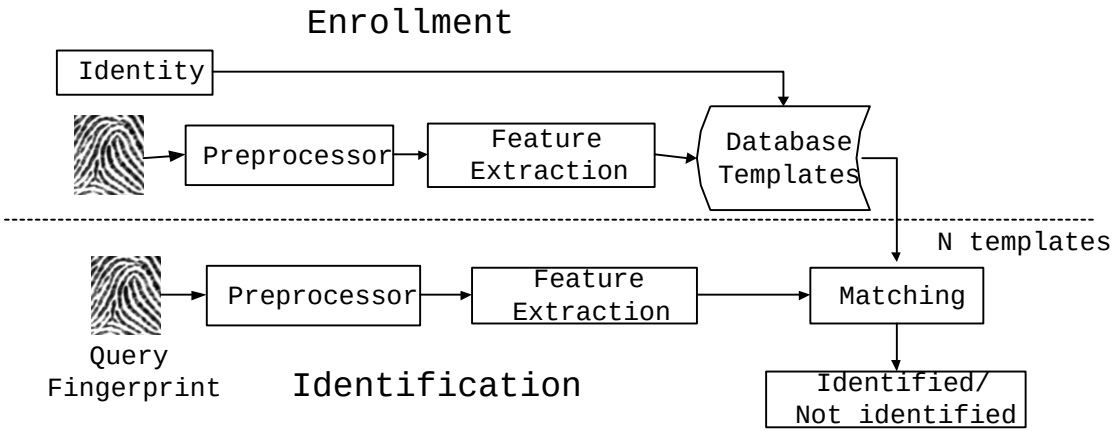


Figure 1.3: The block diagram of an automatic fingerprint identification system

of fingerprint identification, respectively. They are computed by [73]:

$$FMR_{1:N} = 1 - (1 - FMR)^N \tag{1.1}$$

$$FNMR_{1:N} = FNMR \tag{1.2}$$

where FMR and $FNMR$ are the false match and non-match error rates of fingerprint verification, respectively. We can see that $FMR_{1:N}$ increases with large N. In addition, fingerprint verification is usually a kind of fine matching algorithm so that the fingerprint identification is very time consuming by repeating it N times (scale up to the size of database). Thus, both the accuracy and time efficiency of fingerprint identification deteriorates seriously if the 1:1 verification algorithm is simply extended to the 1:N identification system especially for large database.

Multi-level matching (search) approaches are proposed to facilitate the search of database by incorporating the global and local information of fingerprint in an AFIS [89,96]. If two fingerprints are from the same finger, they must have the same global information first. The coarse level search, which is also called "fingerprint retrieval" in [73], is introduced to reduce the search space of the time-consuming fine matching for efficient fingerprint identification. After the fingerprint retrieval is applied to narrow down the search space of fingerprint verification (i.e., fine

matching), the false match rate $FMR_{1:N}$ and false non-match rate $FNMR_{1:N}$ are computed by [73]:

$$FMR_{1:N} = 1 - (1 - FMR)^{PR \times N} \tag{1.3}$$

$$FNMR_{1:N} = RER + (1 - RER) \times FNMR \tag{1.4}$$

where PR (i.e., Penetration Rate) is the average percentage of the template fingerprints retrieved over all query fingerprints and RER is the retrieval error rate (i.e., the percentage of query fingerprints with false retrieval). Assume that FMR of a fingerprint matching algorithm is 10^{-5} for an acceptable $FNMR$, Figure 1.4 shows the effects of the different penetration rates PR on the the false match rates $FMR_{1:N}$ against the size of database. For a database consisting of 10,000 fingerprints ($N = 10,000$), the $FMR_{1:N}$ is almost up to 10% by repeating the fine matching N times. It can be reduced to about 1% if a fingerprint retrieval with $PR = 10\%$ is employed to narrow down the fine matching (i.e., the query fingerprint is only required to be compared with 10% of the database templates in the fine matching). Thus, by reducing the search space of fine matching to only a small subset of the entire database, fingerprint retrieval not only reduces the response time of the AFIS but also alleviates the deterioration of $FMR_{1:N}$. Therefore, fingerprint classification and indexing have been proposed and investigated for the coarse level matching of database templates to narrow down the search space of fine matching in an AFIS.

1.5 Motivation

As discussed in the above sections, although satisfactory performances (i.e., accuracy and time efficiency) have been reported for the fingerprint verification [20, 72], the performance of fingerprint identification will deteriorate seriously if the 1:1 verification algorithm is directly extended to an AFIS. Fingerprint identification,

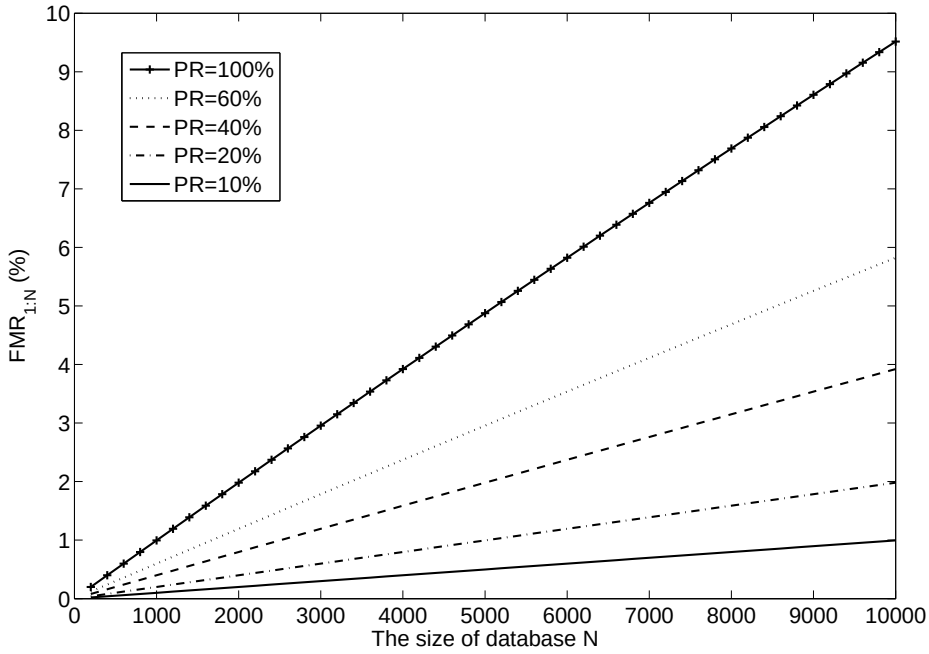


Figure 1.4: The effects of the penetration rate PR on the false match rate $FMR_{1:N}$ against the size of database in fingerprint identification system.

which requires the search of database for a match, is more complex than the fingerprint verification, especially when the database is large. Its performance is still not acceptable for wide employment in the large scale identifications. A fast and accurate fingerprint retrieval is expected to facilitate the search of fingerprint database and alleviate the performance deterioration of an AFIS. The motivation of this thesis is to review the current published approaches and techniques for the coarse level search of fingerprint database and fill the gaps in the following areas where inefficiency is observed:

- Fingerprint is composed of parallel ridge and valley flows with a certain orientation and frequency in a local region. In general, there are two local parameters widely used for analyzing fingerprints, which are the local ridge orientation and the local ridge distance. The ridge orientation describes the ridge-valley flow pattern while the inverse of ridge distance represents the ridge density or frequency. They play important roles for fingerprint

processing and feature extraction. Some methods have been proposed for the estimation of them in the literature. However, a significant number of captured fingerprint images are of poor quality. It is still a challenging problem to robustly and reliably estimate these local parameters in the poor quality fingerprints corrupted by different kinds of noise caused by sensor error, elasticity of skin, scars, ridge breaks and too dry or wet finger, etc..

- Pose transformation usually exists in the fingerprint impressions originated from the same finger. To achieve the invariance of such transformation, translation and rotation are needed to bring two fingerprints into alignment, which is one of the crucial parts for fingerprint retrieval and identification. Fingerprint alignment based on the minutiae structure and ridge shape is time consuming and is thus unsuitable for the fingerprint retrieval. One common and efficient solution for the retrieval is the alignment of fingerprints based on a reference scheme. Some landmark points in a fingerprint are often used as the reference point for fingerprint alignment [12, 46, 49, 66, 84]. However, consistent and reliable detection of the reference point for all types of fingerprints is not an easy task and thus need continuous effort for improvement.

- Fingerprint classification, which is also called "exclusive fingerprint classification" [66], is a traditional approach to provide an efficient indexing mechanism for large-scale fingerprint identification. It assigns each fingerprint into one of predefined classes and the query fingerprint only needs to be compared with the fingerprints of the same class in the fine matching. An ideal automatic classification algorithm should be able to consistently classify fingerprints into a significant number of classes with a desirable accuracy. However, most existing classification algorithm classify fingerprints into 4 or 5 classes. In addition, fingerprints are unevenly distributed in these classes. Thus, this approach cannot sufficiently reduce the search of fingerprint database. Moreover, there are some ambiguous fingerprints whose class

membership cannot be consistently stated even by human experts. For example, about 17% of fingerprints in the NIST database-4 [102] are labelled as two classes by human experts. An attractive approach called "continuous fingerprint classification" is proposed to avoid the difficult problems of exclusive classification [66]. This approach represents each fingerprint with the numerical feature vectors and similar fingerprints are mapped into close points in the multi-dimensional feature space by a given distance measure. The database templates close to the query fingerprint are retrieved for the fine matching of fingerprint identification. This approach can achieve better retrieval performance than the exclusive classification. Some algorithms on continuous classification have been proposed in the literature [11, 14, 17, 53], but their retrieval performances are still not very attractive. Further works on this approach such as extraction of compact and representative features and better quantifying the similarity between two representations are still of great interest to improve the retrieval performance.

- Clustering is a crucial technique widely used in discovering the underlying structure in a data set by unsupervisedly grouping the similar patterns. It is often used to facilitate the information retrieval from large database by exploiting the underlying data structure. Most partitional clustering algorithms represent each cluster with a single prototype such as the centroid and medoid of the cluster. This may not adequately model the clusters of arbitrary shape and size and hence limits the clustering performance on the complex data structure. Using multiple prototypes to model the clusters is expected to improve the clustering performance on discovering the clusters complex in shape and size.

- Although continuous fingerprint classification can avoid the problems of exclusive classification and achieve good retrieval performance, it only ranks the database templates according to their similarities to the query fingerprint while neglecting the similarities among the database templates. The

retrieval speed is still prohibitive for large database by the exhaustive comparison of the query fingerprint with all database template. The clustering can be applied to partition the fingerprint database into a number of clusters with more flexibility. The fingerprint retrieval is performed by comparing the query fingerprint with the cluster prototypes instead of all database templates. Thus, the similarities among the database templates can be exploited to facilitate the efficient search of fingerprint database. A good clustering technique can be explored to speed up the fingerprint retrieval process without compromising the retrieval accuracy.

1.6 Objectives

The primary objective of this thesis are to develop an automatic fingerprint retrieval algorithm which is able to sufficiently reduce the search space of fine matching with desirable accuracy and efficiency. The proposed fingerprint retrieval algorithm is independent of the fine matcher and can be coupled with an existing verification algorithm to obtain an AFIS. In developing an automatic fingerprint retrieval algorithm, two most important issues should be considered: (i) how to extract the representation feature from fingerprint and (ii) how to search the fingerprint database. More specifically, the objectives of the research works reported in this thesis can be summarized as:

- Propose some techniques to robustly and reliably estimate the two local parameters: local ridge orientation and local ridge distance, which are important for fingerprint analysis.
- Propose an effective method to consistently locate a reference point and compute a corresponding reference direction for all types of fingerprints which can be used for fingerprint alignment.
- Develop a fingerprint retrieval algorithm based on the continuous classifi-

cation. The representation features other than the minutiae features are extracted for the retrieval. New distance measure is investigated to quantify effectively the dissimilarity of the fingerprint representations.

- Develop a multi-prototype clustering algorithm to discover the clusters of arbitrary shape and size. Multiple close prototypes are grouped to represent one cluster.

- Develop a clustering-based fingerprint retrieval algorithm. The nonuniform spacing of fingerprint is exploited to produce the compact and representative feature vector. The clustering technique is investigated to exploit the similarities among the database templates for the efficient fingerprint retrieval.

1.7 Major Contributions of the Thesis

The major contributions in this thesis can be summarized as follows:

- Some techniques have been proposed to robustly and reliably estimate two local parameters of fingerprint: local ridge orientation and local ridge distance, which play important roles for fingerprint analysis and retrieval. A new orientation smoothing method based on the adaptive neighborhood is proposed to not only attenuate the noise but also maintain the orientation localization in the high curvature area. Instead of computing only one x-signature in the oriented window of each block, we propose to estimate the local ridge distance based on more than one x-signatures which is more robust to noise and irregular ridge flows.

- An effective method has been proposed to consistently locate a reference point and compute a corresponding reference direction for all types of fingerprints. The reference point is located based on multi-scale analysis of the orientation consistency, while the reference direction is computed by analysis of the orientation differences between 16 redial directions from the reference

point and the local ridge orientations along these radii. They can be used for fingerprint alignment to achieve the invariance of pose transformation in fingerprint retrieval.

- We have developed a fingerprint retrieval algorithm based on continuous classification that uses the orientation field as the main retrieval feature and the dominant ridge distance as an auxiliary feature. The dominant ridge distance of a fingerprint is more robust to noise than the simple average ridge distance and consistently improve the retrieval performance. These two coarse level features are not closely correlated with the minutiae features so that the fingerprint retrieval approach can cooperate with the minutiae based matching algorithms to design an identification system. A new distance measure is proposed to quantify the distance between two orientation vectors more effectively than the conventional Euclidean and Manhattan distance measures. In addition, we propose a regional feature weighting scheme on the distance measure to improve the retrieval performance. Furthermore, a variable search tolerance is introduced for more effective retrieval than those by the fixed distance and fixed order. Experimental results on the NIST database-4 and FVC2000 demonstrate that our proposed approach outperforms some existing approaches in terms of retrieval efficiency v.s. accuracy.

- A multi-prototype clustering algorithm has been developed to discover the clusters of arbitrary shape and size. The squared-error clustering is used to produce a number of prototypes to locate the regions of high density. A separation measure is proposed to evaluate how two prototypes are separated by a sparse region and is used to organize the prototypes into a given number of clusters. New prototypes are iteratively added to improve the poor cluster boundary resulted by the poor initial settings. The proposed algorithm requires less memory space and computation cost than some hierarchical clustering algorithms such as *Single-link* and *Complete-link* while

preserves much of the speed and efficiency of the squared-error clustering algorithm. Experimental results on both synthetic and real data sets show the effectiveness of the proposed clustering algorithm.

- A clustering-based fingerprint retrieval algorithm has been developed to speed up the retrieval process without compromising the retrieval accuracy. A nonuniform spacing of fingerprint by a circular tessellation is proposed to compute a multi-scale orientation field as the main retrieval feature. The nonuniform spacing not only produces more compact orientation vector but also achieves better retrieval performance than the uniform spacing. The dominant ridge distance as an auxiliary feature not only reduces the orientation comparisons in the query process but also consistently improves the retrieval accuracy. A modified K-means clustering approach is proposed to partition the high dimensional orientation feature space into a number of clusters. It outperforms the traditional K-means clustering for the fingerprint retrieval. Based on the offline database clustering, a hierarchical query processing is proposed to perform the cluster search followed by the continuous fingerprint classification. It not only reduces the retrieval complexity but also improves the retrieval accuracy. The extensive experimental studies and comparisons consistently demonstrate the effectiveness and superiority of the clustering-based fingerprint retrieval algorithm.

1.8 Organization of the Thesis

This thesis is divided into seven chapters which are organized as below:

Chapter 1 gives a brief introduction of the history and technologies in the personal recognition based on fingerprints. The motivation, objectives and contributions of the thesis are also presented in this chapter.

In Chapter 2, literature review is given on some common and published approaches of the coarse level search of fingerprint database in automatic personal

identification. They are broadly classified according to different feature extractions and search strategies used in these approaches.

In Chapter 3, some techniques are proposed to robustly and reliably estimate the two important local parameters: ridge orientation and ridge distance. In addition, an effective method is proposed to robustly and consistently locate a reference point and compute a corresponding reference direction for all types of fingerprints. The reference point and direction can be used for the fingerprint alignment to achieve the invariance of pose transformation.

In Chapter 4, a fingerprint retrieval algorithm based on continuous classification is presented to incorporate two coarse level features: orientation vector and dominant ridge distance. A new distance measure is proposed to better quantify the similarity between two orientation vectors than the traditional Euclidean and Manhattan distance measures. In addition, a feature weighting scheme by the entropy is exploited on the distance measure to achieve more effective fingerprint retrieval. We perform our algorithm on the NIST database-4 and FVC2000 and compare it with some published approaches in the field.

In Chapter 5, a multi-prototype clustering algorithm is presented that can discover the clusters of arbitrary shape and size. This algorithm begins using the squared-error clustering to produce a number of prototypes. A separation measure is proposed to evaluate how well two prototypes are separated. The multiple prototypes with small separation are organized into a given number of clusters. New prototypes are iteratively added to improve the poor cluster boundaries. We perform the proposed clustering algorithm on both synthetic and real data sets and compare it some well-known clustering algorithms.

In Chapter 6, a clustering-based fingerprint retrieval algorithm is presented for an efficient search of fingerprint database. A nonuniform spacing of fingerprint is proposed to compute a multi-scale orientation field as the main retrieval feature and the dominant ridge distance is used as an auxiliary feature. A modified K-means clustering is proposed to partition the high dimensional orientation feature

space into a number of clusters represented by prototypes. Based on the offline database clustering, a hierarchical query processing is proposed to facilitate an efficient fingerprint retrieval. We perform our algorithm on the NIST database-4 and the results are compared with those of some published approaches in the field.

Chapter 7 gives the conclusions of this thesis and recommends some research directions which can be further pursued in the future.

Chapter 2

Literature Review

2.1 Introduction

Identifying a person based on fingerprints requires the search of database for a match of the query fingerprint. A lot of approaches are proposed in the literature to facilitate the database search in automatic fingerprint identification. Most of them are based on the multi-level search of database by incorporating the global and local information of fingerprint. Fingerprint retrieval and indexing are introduced as a coarse level search of database to reduce the search space of the time-consuming fine matching in an AFIS [8,73]. An accurate and efficient fingerprint retrieval can greatly reduce the response time and improve the accuracy of an AFIS, especially for a large database. A typical fingerprint retrieval algorithm usually extracts a representation feature set to capture the individuality of each fingerprint and does some comparisons (search strategies such as classification and indexing) to select a subset of database as the candidates of a query fingerprint for the fine matching. In this chapter, we will give a literature review of some common and published approaches of fingerprint retrieval in automatic personal identification. They are broadly classified according to the feature extractions and search strategies used in developing the automatic fingerprint retrieval algorithms.

2.2 Feature Extraction

The captured fingerprint image by a sensor is a grey level image. It is well known that the grey values of fingerprint are unstable and of high dimension so that they cannot be directly used to reliably and efficiently represent fingerprint. Feature extraction is an important stage to identify representative features to capture the individuality of each fingerprint. Fingerprint is composed of parallel ridge and valley flows with a strong orientation tendency and well-defined spatial frequency in the local region that does not contain irregular ridge flows. In general, there are two main types of features for fingerprint representation: coarse level features which globally describe the ridge-valley flow patterns and fine level features with the minute details of the ridge flow anomalies. To facilitate the search of fingerprint database, the coarse level features are usually used for fingerprint retrieval while the fine level features are employed in the fine matching. A good feature set for the fingerprint retrieval usually has the following properties: easily computable and compact, translation and rotation invariant, noise robust and discriminating over a large number of fingerprints. In addition, the retrieval feature set should be independent on or loosely correlated with that used in the fine matching to avoid redundant representation of fingerprint in an AFIS. Most existing algorithms of fingerprint classification and indexing are based on one or more of the following features: singular points, local ridge orientations, filter responses, minutiae points and others.

2.2.1 Singular Points

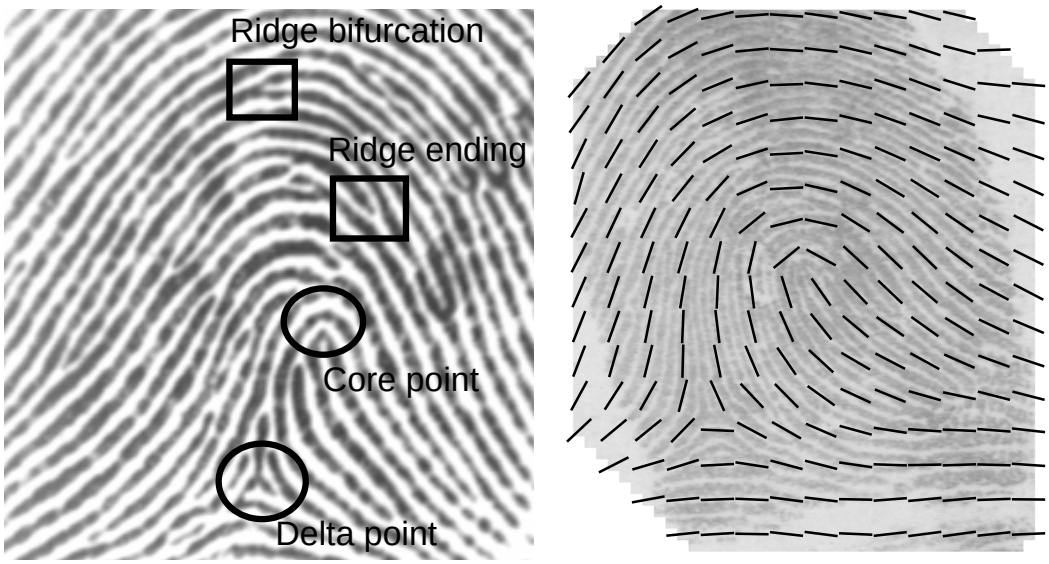
Singular points are the points in a fingerprint with sharp changes of ridge flows. Core and delta points are two types of singular points. The core point is defined as the topmost point of the innermost curving ridge and the delta point is defined as the center of triangular regions where three different direction flows meet [39]. The positions of a core and a delta point in an example fingerprint is given in

Figure 2.1(a). Core and delta points are two landmarks of a fingerprint and the global ridge and valley flow structure of fingerprint can be determined by the position and rotation of them. The numbers and positions of core and delta points are closely related to the human-predefined fingerprint class types. For example, whorl fingerprint often contains two pairs of core and delta points, loop fingerprint usually has one pair of core and delta points while no singular point exists in plain arch fingerprint. In addition, this kind of feature (the numbers and positions of core and delta points) is invariant to the translation, rotation, and moderate amounts of scale changes in fingerprint image. It is often used to generate some rules for fingerprint classification [30, 55, 108]. However, it is not an easy task to reliably and consistently detect the singular points in some partial or poor quality fingerprints. The singular points are usually outside of the partial fingerprint images. For example, the delta point is easily left outside of the fingerprint due to the small size of the sensor, especially of the dab fingerprint captured by the solid-state sensor. Inconsistent or unreliable detection of singular points may result in inconsistent classification of fingerprints from the same finger and increase the retrieval error.

2.2.2 Local Ridge Orientation

In addition to the singular points, the global ridge flow structure of a fingerprint can be inferred by extraction of the local ridge orientation field. To reduce the computational complexity, fingerprint is often divided into blocks and the local ridge orientation of each block is computed. The local ridge orientation field of fingerprint is composed of all the block-wise orientations. It is also called "directional image" in some literature. Figure 2.1(b) shows an example of fingerprint orientation field. The orientation field is widely used as the main feature for fingerprint classification and indexing [12, 14, 15, 53, 66, 67, 104].

An orientation vector is often constructed by concatenating the local ridge orientations and associating a dimension to each orientation element for fingerprint



(a) (b)

Figure 2.1: (a) Singular points (black circles) and minutia points (black boxes) and (b) local ridge orientation field.

representation [12, 53, 66, 104]. To be invariant to the translation of fingerprint image, the orientation field is usually registered with respect to the core points. Each local ridge orientation is represented by a unit vector of the doubled angle $[\cos(2\theta), \sin(2\theta)]$ to facilitate the feature transform and weighting. The constructed orientation vector is usually of high dimension (e.g., 1680 in [66]). The Karhunen-Loeve transform (KLT) is one of the most widely used statistical framework for reduction of dimensionality with the minimum loss of information. It is used to reduce the dimensionality of orientation vector and produce a more compact representation of fingerprint. However, the efficacy and efficiency of KLT progressively deteriorate with the increase of the database size. To overcome this problem, a multi-space generalization of KLT (MKL) is proposed to create some subspaces to represent different sets of patterns with common characteristics [16, 19]. It is imposed on the enhanced orientation vectors to produce multiple subspaces to represent fingerprint for classification [15].

A structural feature based on the relational graphs of the orientation field is proposed for fingerprint classification [67]. The orientation field is segmented into

several homogenous regular-shaped regions by using a dynamic clustering technique according to some well-suited criteria, i.e. minimization of the variance of the orientations within the regions. A relational graph is constructed by creating a node for each region and an arch for each pair of adjacent regions to summarize the topological structure of fingerprint. The inexact graph matching techniques are employed to compute the distances between the obtained graph and the prototype graphs of the predefined classes which are used to produce a feature vector to represent fingerprint for classification. A further improvement of this structural feature is proposed to use a set of dynamic masks to guide the segmentation of orientation field into regions [14]. The dynamic mask of each fingerprint class is generated using the same clustering algorithm as that in constructing the relational graph. It represents a general segmentation of regions for the class. A numerical feature vector is produced by the adaption of the masks to represent each fingerprint for classification. These structure features based on relational graph of orientation field are invariant to the translation and rotation of fingerprint image without any position alignment or normalization. In addition, they can be directly applied for partial fingerprints.

The orientation feature extracted by a single method captures limited information of fingerprint orientation field. Various methods are used to extract the orientation features which are integrated to give a more comprehensive representation of fingerprint [17, 82]. The structural feature on the relational graph [14] and the MKL based statistical feature [15] are integrated to capture more information of fingerprint orientation field, thus improving the performance of fingerprint classification [17]. The structural and statistical approaches are combined to extract the representation feature from the local ridge orientation field for classifying fingerprints into six classes [82]. In this approach, the orientation field is binarized by a set of direction codes and its bifurcations are removed by a set of filter operators. A one-dimensional binary string is produced to form the structural feature set. The statistical feature set is extracted from the Euclidean distances between the second moment of an unknown pattern and the mean of the second moment of each fin-

gerprint class. The hybrid orientation feature extracted by different methods can compensate for the limitations of the respective single extraction strategy to give a more comprehensive representation of fingerprint for more effective retrieval.

Although the local ridge orientation field has been widely used for the coarse level search of fingerprint database, it is still a challenging problem to robustly and reliably estimate the local ridge orientation for the poor quality fingerprints corrupted by heavy noise. In addition, how to extract a compact and representative feature from the orientation field is an important issue for fingerprint retrieval.

2.2.3 Filter Responses

The ridge and valley flows of fingerprint have strong orientation tendency and well-defined spatial frequency in the local neighborhoods that do not contain irregular ridge flows. Gabor filters are directional band-pass filters which not only have the properties of both orientation and spatial frequency selectivity but also have optimal joint resolution in both spatial and frequency domains [27]. The Gabor filters are properly tuned and applied on the fingerprint image and a feature vector is extracted from the responses to represent fingerprint for classification [49]. In this approach, the region of interest around a registration point is unevenly partitioned by a circular spatial tessellation which is composed of six bands and eight even sectors in each band. The fingerprint is convolved with four Gabor filters with the orientation equal to 0, $\frac{\pi}{4}$, $\frac{\pi}{2}$ and $\frac{3\pi}{4}$ and the responses of the four filters are produced. A feature vector called "FingerCode" is constructed by the standard deviations of the Gabor filter responses in all sectors. It consists of 192 elements and is used for fingerprint classification. A set of complex filters are proposed for the detection of patterns with radial symmetries [10]. Two of them are applied on the orientation field of fingerprint to extract the local singularities, i.e. the similarities to singular points, for fingerprint retrieval [65]. The feature extraction from the filter responses are dependent on setting the parameters of the filters. Selecting an appropriate parameter for some filters is not an easy task. For example, the local

ridge orientation and distance are two important parameters for the Gabor filters. However, it is not an easy task to reliably estimate them, especially for the poor quality fingerprints.

2.2.4 Minutiae Points

Minutiae points in a fingerprint are the minute details of ridge flow anomalies. Ridge ending and bifurcation (see Figure 2.1(a)) are two most common types of minutiae points. The minutiae triplets (i.e. three minutiae points are constructed into a triangle which is called a minutiae triplet) are employed to index fingerprints to facilitate the search of database [9,36,96]. Using the minutiae feature for fingerprint indexing is proposed in [36] which is further improved in [9,96]. In [36], all the minutiae triplets in a fingerprint are identified with each triplet defining a triangle. This approach extracts the geometric features of each triplet: the length of each side, the angles and the ridge count between each pair as the feature set. A geometric hashing table is built by quantizing all the possible triplets for fingerprint indexing which can avoid the time consuming minutiae comparison between the query fingerprint and all database templates. A variant of this representation is presented to extract more robust geometric features such as handedness and maximum side of the triangle in [9]. Some geometric constraints are introduced to more effectively index fingerprints. Two new features of the minutiae triplet, minutiae density and ridge counts in each side of triangle, are added to the geometric feature set and the performance of fingerprint indexing is further improved in [96]. The fingerprint indexing based on minutiae triplets is attractive for the fast search of fingerprint database. However, the minutiae feature is the most important fine level feature and widely used in the fine matching of fingerprint verification and identification algorithms [44, 45, 52, 89]. One should try to avoid a redundant representation of fingerprint between the coarse level search and fine level matching in designing an AFIS.

2.2.5 Others

Fingerprint representation by a single feature captures limited information of fingerprint and thus limits the effectiveness of fingerprint retrieval. Various features are combined to give a more comprehensive representation of fingerprint for the effective retrieval [11, 91]. Three features, local ridge orientation field, FingerCode [49] and minutiae triplets [9], are combined to represent fingerprint to achieve more effective retrieval than that based on a single feature [11]. In [91], a set of fiducial lines are laid across the skeleton image of the ridges and the local ridge features are extracted at each intersection. The extracted local features consist of four measurements: the angle of the intersection, the ridge curvature, the changes of angle and distance since the last intersection. This representation can capture more information about a fingerprint which allows a representation of higher resolution. But it involves several local ridge features with different metrics which result in complex processing. The extracted local ridge features are slowly varying across the fingerprint with the stationary distributions over some spatial or temporal periods.

Hidden Markov models are a form of stochastic finite state automata which is successfully applied in speech recognition [87]. Thus, two-dimensional hidden Markov models are applied to statistically model the local ridge features by accumulations of the evidence across the whole fingerprint. To extract the hybrid representation feature, one should try to avoid redundant representation of fingerprint for the effective retrieval. A good processing method is also important for combining several features with different metrics.

In addition, some representation features for fingerprint classification are beyond the above mentioned features or their combinations. A Hexagonal Fast Fourier Transform is applied to extract a feature vector by utilization of the hexagonally sampled data and extension of the output data in a rectangular scheme for fingerprint classification [33]. A novel fingerprint representation is proposed by modelling fingerprint images as the amplitude-modulated (AM) and frequency-

modulated (FM) functions [86]. In the AM-FM model, ridge variations are represented as a FM function while the variations in the ridge intensity are modelled as an AM function. The dominant FM component is employed as the classification feature.

2.3 Search Strategies

After extraction of the representation features, some comparisons are needed to select a subset of database as the candidates for fine matching in fingerprint retrieval. Thus, an accurate and efficient search strategy is also crucial for the fingerprint retrieval. In general, three search strategies: exclusive fingerprint classification, continuous fingerprint classification and fingerprint indexing are often used in the literature to retrieve a subset of database according to the extracted features for the fine matching of fingerprint identification.

2.3.1 Exclusive Fingerprint Classification

Exclusive fingerprint classification is also called as "fingerprint classification" in the literature. It is denoted by adding "exclusive" to be different from another approach, i.e., "continuous classification", in [66]. Exclusive fingerprint classification assigns each fingerprint exclusively into one of predefined classes based on the extracted features and the database templates in the same class as that of query fingerprint are retrieved for the fine matching. It is a traditional approach of coarse level search to accelerate the search of fingerprint database in large-scale identification system [89].

Exclusive fingerprint classification has been studied for more than one century. Francis Galton defines three large general classes of fingerprint patterns which are Arch, Loop and Whorl with each class containing the similar general characteristics or family resemblance [35]. Edward Henry further refined this fingerprint

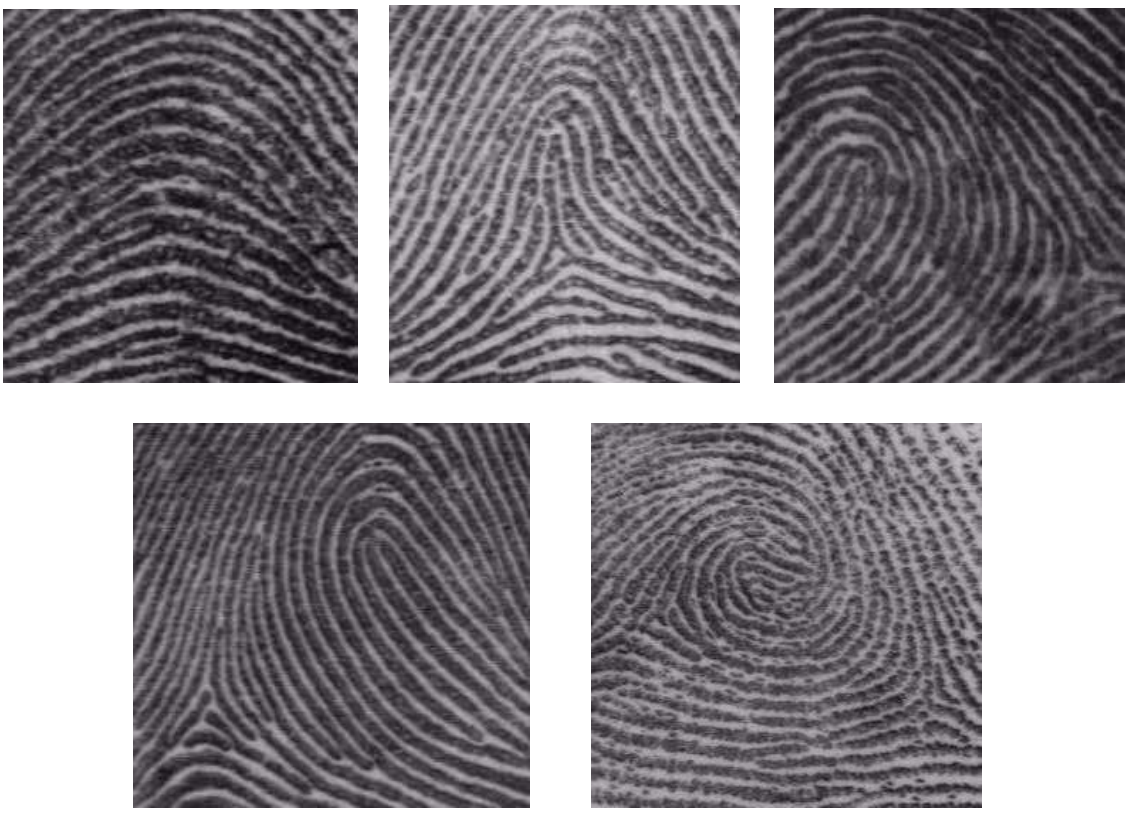


Figure 2.2: The five common fingerprint classes: Arch, Tented Arch, Left Loop, Right Loop and Whorl.

classification by dividing the general classes into more subclasses according to the smaller differences existing in the patterns of the classes [39]. The five most common fingerprint classes of the Galton-Henry classification scheme are Arch, Tented Arch, Left Loop, Right Loop and Whorl as shown in Figure 2.2. Consistently and reliably classifying fingerprints is a difficult problem due to the small inter-class variability and the large intra-class variability of the fingerprint patterns. Exclusive fingerprint classification has generated great interest because of its importance and intrinsic difficulty. A lot of fingerprint classification approaches are proposed in the literature [12,15,49,55,74,85,91,97,105]. Most of them are based on the five common fingerprint classes and can be broadly grouped as: rule-based, syntactic, structural, statistical, neural network based and multi-classifier approaches [73].

The rule based approaches usually adopt the techniques and rules commonly used in the manual classification by human experts for the automatic fingerprint

classification. For example, forensic experts often visually classify fingerprints based on the singular points. An approach is proposed to generate rules according to the number and positions of singular points for the automatic fingerprint classification [55]. Another rule based classification approach is proposed to classify fingerprints based on the type and orientation of core points [25]. These fingerprint classification approaches highly depend on the reliable and consistent detection of singular points. Although they work well for the rolled fingerprint impressions scanned from card, they are not suitable for the dab fingerprints captured by solid-state sensor in which the delta points are often outside of the images. Instead of only relying on the singular points, the traced pseudoridge analysis is used to compensate for the missing singular points to improve the classification performance [108]. In addition, the robust geometrical shapes of the ridge lines are extracted to produce rules for fingerprint classification [26, 47]. B-spline curves are extracted to model fingerprint ridges and the classification is conducted by tracing the resulting B-spline curves to detect the turns [26]. A fingerprint kernel is defined to model the general geometrical shape of the fingerprints in each class and the classification is conducted by hierarchical kernel fitting [47]. In general, the rule based approaches of fingerprint classification are straightforward and easy to understand. But it is still difficult to generate rules which can consistently and reliably classify the poor quality or partial fingerprints.

The syntactic classification approaches usually represent each fingerprint by some symbols, define a grammar for each fingerprint class, and finally use a parsing process to classify fingerprints. Moayer and Fu [79, 80] propose two syntactic classification approaches. In these approaches, terminal symbols are associated with small groups of orientation elements in the orientation field and fingerprint patterns are described with the stochastic grammars [79] and tree grammars [80]. A syntactic classification approach is proposed based on the global distribution of 10 basic ridge patterns, the analysis of the ridge shapes and a sequence of ridge distributions [22]. Due to the great variance of fingerprint patterns, complex grammars are often required that may result in a complicated and unstable inference

in the syntactic approaches.

Structural approaches of fingerprint classification are based on the relational organization of the information in a fingerprint which is often represented by graphs or trees. A state-of-the-art structural approach is proposed based on the relational graphs of some connected regions which are obtained by partitioning the fingerprint orientation field to minimize the orientation variance within each region [67]. Instead of giving the clustering algorithm total freedom, a set of dynamic masks, together with an optimization criterion, are used to guide the partitioning of orientation field into homogeneous regions [14]. The dynamic mask represents a general partition of the homogeneous regions for each fingerprint class. The structural classification approaches are usually based on the global structural information of fingerprint so that they can work on partial and poor quality fingerprints. However, it is not an easy task to obtain a stable and reliable structure of a fingerprint.

Statistical approaches of fingerprint classification represent each fingerprint by a numerical feature vector and use some general statistical classifiers to classify fingerprints. The most widely used statistical classifier, K-nearest neighbor classifier, is employed for fingerprint classification [33, 49]. In addition, the statistical Karhunen-Loeve (KL) and multi-space generalization of KL (MKL) transforms are used to reduce the dimensionality of orientation vectors for fingerprint classification [12, 15, 66].

Some approaches of fingerprint classification determine the fingerprint class by using neural networks. The probabilistic neural network is often employed for the determination of fingerprint classes [12, 86, 103, 104]. A fingerprint classification approach based on neural network is proposed to construct a pyramidal architecture consisting of several multi-layer perceptions with each perception trained to discriminate one fingerprint class [54]. A multi-layer artificial neural network is constructed to be composed of six independent sub-networks which are used to discriminate six fingerprint classes (with each subnetwork for one class) [82]. The self-organizing neural networks are employed for fingerprint classification [38]. In

general, the fingerprint classification approaches based on neural networks take some feature sets for training the classifiers to discriminate different fingerprint classes. The more feature sets are provided for training the neural networks, the more the classifiers are able to learn about the classes and accurately discriminate the fingerprint classes. Thus, these approaches can work well if sufficiently large and representative fingerprints of all different classes are provided for training.

It is well known that the fingerprint classification approaches have limited classification performance by using a single method of class determination. Two or more classifiers, which produce somewhat uncorrelated classification errors, are combined to offer complementary information of class determination and improve the performance of classification with a single classifier [49, 91, 106]. A hybrid multi-channel approach of fingerprint classification assigns the FingerCode of each fingerprint into one of the five common classes using the K-nearest neighbor classifier followed by a set of neural networks for further class determination [49]. A hybrid fingerprint classifier is proposed to combine a hidden Markov model classifier and a decision tree classifier [91]. Two ways of classifier combination: linear likelihood and neural network combinations are also discussed for this hybrid fingerprint classifier [91]. In [106], two machine learning approaches: support vector machines and recursive neural networks are combined for the determination of fingerprint classes.

The exclusive fingerprint classification has been widely studied to facilitate the search of fingerprint database because of some advantages such as human-interpretable, fast query process and rigid database partitioning. It provides an efficient indexing mechanism for large-scale fingerprint identification. An ideal automatic classification algorithm for fingerprint retrieval should be able to classify fingerprints into a large number of classes in a fast, consistent and reliable way. However, most automated classification algorithms are able to classify fingerprints into only 4 or 5 classes. Some researchers explores to further divide the fingerprint classes into more specific sub-classes [30]. But the automatic sub-classification

is more difficult than the first-level classification due to the small inter-class variance and large intra-class variance. Furthermore, fingerprints are unevenly distributed in these classes. The natural fingerprint distribution among the five common classes is approximately 3.7% plain arch, 2.9% tented arch, 33.8% left loop, 31.7% right loop and 27.9% whorl, which are estimated on the classification summary of more than 222 million fingerprints [104]. More than 90% of fingerprints belong to only three classes. On average, the query fingerprint still needs to be compared with about 29.48% of database templates in the fine matching of identification after the exclusive classification is applied. Thus, the exclusive classification cannot sufficiently narrow down the search of fingerprint database. Another problem of this approach is that there are many ambiguous fingerprints, whose class membership cannot be exclusively stated even by human experts. It is more difficult to consistently classify them by the automatic fingerprint classification algorithm. For example, there are about 17% of fingerprints in the NIST special database 4 (NIST-4) [102] labelled as two classes by human experts. These intrinsic difficulties in exclusive fingerprint classification lead some researchers to investigate other strategies that are not based on the human predefined classes but can effectively narrow down the search of fingerprint database.

2.3.2 Continuous Fingerprint Classification

The continuous fingerprint classification is proposed to avoid the problems of exclusive classification for the effective fingerprint retrieval [66]. Instead of classifying fingerprints into predefined classes, continuous classification represents each fingerprint by the numerical feature vectors and similar fingerprints are mapped into close points in the multi-dimensional feature space by a given distance measure. Fingerprint retrieval is performed by comparing the query fingerprint with all database templates and the closer ones are retrieved as the candidates for the fine matching. The tradeoff between the penetration rate and retrieval accuracy can be easily adapted by adjusting the size of retrieval neighborhood.

The local ridge orientation field of fingerprint is often used as the main feature for continuous classification [11, 14, 17, 53, 66]. A numerical feature vector is constructed by concatenating the orientation elements of the registered orientation field to represent fingerprint [53, 66]. In these approaches, each local ridge orientation is represented by a unit vector of the doubled angle $[\cos(2\theta), \sin(2\theta)]$ to facilitate the feature transform and comparison. The constructed orientation vector is usually of high dimension which results in high computation cost and memory space in fingerprint retrieval. The KL transform is used to reduce the dimensionality of orientation vector without loss of much information. Euclidean distance measure is used to compare two orientation vectors. Although the representation by the orientation vector is suitable for continuous fingerprint classification, the retrieval accuracy highly depends on the registration of orientation field by the reference points whose detection is sensitive to noise and partial images. The fingerprint orientation field is characterized by the relational graphs and a numerical feature vector is produced by the adaption of the masks to represent each fingerprint for continuous classification [14]. This approach captures the global structure information of fingerprint orientation field without the registration by reference point. Thus, it can work on the partial and poor quality fingerprint images. To further improve the retrieval accuracy, some approaches are proposed to combine different representation features for continuous fingerprint classification [11, 17]. Two numerical features on the relational graph [14] and the statistical MKL [15] are integrated to capture more information of fingerprint for continuous classification [17]. Three numerical features: orientation vector, FingerCode and minutia triplets are combined to represent fingerprint for continuous classification [11]. These combined representation features can compensate for the limits of individual feature and give more comprehensive representations of fingerprint so that better retrieval performances are often achieved in the continuous classification.

The continuous fingerprint classification can avoid the intrinsic problems in exclusive classification and achieve better retrieval performance. However, it only

ranks the database templates according to their similarities to the query fingerprint while neglecting the similarities among the database templates. Although the comparison between the query fingerprint and database template is a coarse level matching and is much faster than the fine matching, the fingerprint retrieval by exhaustive comparison of the database templates is still time consuming for large databases. Further works to facilitate an effective and efficient search of database are still of great interest to the researchers in the area of automatic fingerprint retrieval and identification.

2.3.3 Fingerprint Indexing

In most applications, it is not necessary to classify fingerprints into human interpretable classes in an AFIS. Instead of classifying fingerprints into a small number of human predefined classes, some researcher propose to partition fingerprint database into a number of bins based on the minutia triplets for fingerprint indexing [9,36,96]. This approach builds a geometric hashing table by quantizing all the possible minutiae triplets to index fingerprints which avoids the time consuming minutiae comparison between the query fingerprint and all database templates. In addition, it can classify fingerprints into more groups (classes or bins) than the exclusive classification as it exploits the more discriminating features, minutiae features. This indexing approach is attractive to speed up the search of fingerprint database. However, minutia points are the most important local features and widely used in the fine matching of fingerprint verification and identification algorithms [44, 45, 52]. To use the minutiae features to index fingerprints, one should try to avoid a redundant representation of fingerprint in designing an AFIS. This is because the accuracy deterioration of large-scale fingerprint identification system from the 1:1 verification can be hardly alleviated if the features used in fingerprint indexing (coarse level search) and fine matching (verification) are strongly correlated.

2.4 Summary

This chapter presents a brief review of some common and published fingerprint retrieval approaches in the literature. They are broadly classified according to the feature extractions and search strategies used. The exclusive fingerprint classification has been widely studied to facilitate the search of fingerprint database. However, most of the existing approaches classify fingerprints into only 4 or 5 classes and more than 90% of fingerprints belong to only three classes. Thus, exclusive classification cannot sufficiently reduce the search space of fingerprint database. Continuous classification is proposed to avoid the problems of exclusive classification and can achieve better retrieval performance. Most of the existing methods are based on the orientation feature and Euclidean distance measure. Their retrieval performances are still not very attractive. In the following chapters of this thesis, some research works have been done to further improve the retrieval performance.

The local ridge orientation and the local ridge distance are two important local parameters widely used for analyzing fingerprints and most existing methods cannot do a good job in the presence of high noise level in a fingerprint image. In the first part of this thesis, focus is on the investigation of robust and reliable estimations of these two parameters. In addition, to achieve the invariance of the pose transformation of fingerprint images, fingerprint alignment based on the minutiae structure and ridge shape is time consuming and thus unsuitable for real-time retrieval. We also present an algorithm to consistently detect a reference point for fingerprint alignments in the first part.

In the second part, we develop a fingerprint retrieval algorithm based on the continuous classification. Besides the orientation feature, other discriminating feature and more effective distance measure are explored to further improve the retrieval performance in database search.

The continuous classification only ranks the database templates according to

their similarities to the query fingerprint while neglecting the similarities among the database templates in fingerprint retrieval. The retrieval speed by the comparison of query fingerprint with all database templates is still time consuming for large database. In the last part of the thesis, we explore the database clustering for an efficient fingerprint retrieval.

In the next chapter, we investigate some techniques to robustly and reliably estimate the local ridge orientation and local ridge distance, which play important roles for fingerprint analysis and retrieval. In addition, an effective approach will be presented to consistently locate a reference point and compute a corresponding reference direction for all types of fingerprints which can be used for the fingerprint alignment to achieve the invariance of pose transformation.

Chapter 3

Parameter Estimation and Reference Point Detection

3.1 Introduction

Fingerprint is a grey level image captured by a sensor after pressing a finger against the smooth surface. It is composed of the ridge (low grey value) and valley (high grey value) flows which are interleaved at a certain frequency and run parallel with strong orientation in a local region that does not contain irregular ridge flows such as minutiae and singular points. In general, there are two local parameters widely used for analyzing fingerprints, which are the local ridge orientation and the local ridge distance. The local ridge orientation of fingerprint describes the ridge-valley flow pattern while the local ridge distance is another important intrinsic property of fingerprint as its inverse represents the ridge density or frequency. These two local parameters vary not only across fingerprint impressions of different fingers but also across different regions in the same fingerprint. They play important roles for fingerprint processing and feature extraction. Some methods have been proposed for the estimation of these local parameters in the literature. However, it is still a difficult task to reliably estimate them in the poor quality fingerprints.

In this chapter, some techniques are investigated to robustly and reliably estimate these two local parameters.

In addition, pose transformation usually exists in different fingerprint impressions originated from the same finger. To achieve the invariance of such transformation, translation and rotation are needed to bring two different fingerprints into alignment, which is one of the crucial parts for fingerprint retrieval and matching. Fingerprint alignment based on the minutiae structure and ridge shape is time consuming and is therefore unsuitable for fingerprint retrieval. One common and efficient solution suitable for the retrieval is the alignment of fingerprints based on a reference scheme. Some landmark points in a fingerprint are often used as the reference point for the fingerprint alignment [12, 46, 49, 66, 84]. However, robust and consistent detection of the reference point is still a difficult task and thus needs continuous research effort for improvement. In this chapter, we propose an effective method to detect a reference point and compute a corresponding reference direction for all types of fingerprints which can be used for fingerprint alignments.

3.2 Fingerprint Segmentation

The captured fingerprint images by a sensor often consist of not only the foreground originated from the contact of fingertip with the sensor but also the background, i.e., the blank and heavy noisy areas on the border. In the background of fingerprint, the estimations of the local parameters are usually unreliable and spurious reference points may be produced. If the background is included in feature extraction, the within-finger variance of fingerprint representation is greatly increased which results in low matching score for the fingerprints from the same finger. Fingerprint segmentation is introduced to decide which part of the image belongs to the foreground and which part belongs to the background. A good segmentation plays an important role to reliably extract the local parameters and detect the reference point for fingerprint analysis and feature extraction.

There are some approaches proposed for fingerprint segmentation in the literature [5, 68, 88, 92]. The foreground of fingerprint is discriminated from the background based on the variance of gray values in the direction orthogonal to the local ridge orientation [88]. This approach is based on the observation that the foreground with the striped and oriented pattern contains higher variance of gray values than the background without the oriented pattern does. The average magnitude of the gradients in each image block is used for fingerprint segmentation [68]. The magnitude of the gradients in the foreground of fingerprint is larger than that in the background because the foreground contains the clear ridge and valley alterations. The Gabor filter responses are used to classify the regions of fingerprint into "good", "poor", "smudged" and "dry" [92]. Instead of using only one feature, three features: gradient coherence, the mean and variance of the grey values are combined by a linear classifier to separate the foreground of fingerprint from the background in [5]. A morphological filter is applied in the post-processing to obtain the compact foreground and background clusters. This method has achieved good segmentation results and is employed in this work for fingerprint segmentation. The estimation of local parameters and detection of reference point in the following are performed only on the foreground of fingerprint.

3.3 Estimation of Local Ridge Orientation

The local ridge orientation at a pixel (x, y) of fingerprint is the angle $\theta(x, y)$ that the ridges, crossing through an arbitrary neighborhood centered at the pixel, form with respect to the horizontal axis [73]. It belongs to $[-\pi/2, \pi/2)$ as the ridges of fingerprint are not directed. The local ridge orientation is an intrinsic property of ridge flows and plays an important role in fingerprint processing and representation. It is often estimated at the discrete positions by dividing the fingerprint image into blocks to reduce the computational complexity. The fingerprint orientation field is represented by a matrix composed of all the block-wise orientation elements.

3.3.1 Orientation Estimation

Most existing methods proposed for the estimation of local ridge orientation in the literature can be broadly classified as the pixel alignment based method [38, 55] and the gradient based method [6,29,42,51,68]. The pixel alignment based method estimates the local ridge orientation of each pixel based on the alignments of the pixels in a local neighborhood with respect to a fixed number of reference orientations. The total fluctuation of gray values is expected to be smallest along the local ridge orientation and largest along its orthogonal orientation. The orientation of each image block is estimated by averaging the unit vectors of doubled pixel-wise orientations in the block. However, the reliability of the estimated orientation in the pixel alignment based method may be limited due to the fixed number of reference orientations. Since the phase angle of the gradients is the orientation with maximum changes of gray values, it is perpendicular to the local ridge orientation of each pixel. The gradient based method is widely used to estimate the local ridge orientation because of its high efficiency and resolution [29,42,68]. Due to the periodicity of the phase angle, the orientation of each image block cannot be estimated by arithmetically averaging the phase angles of the pixels. To avoid the orientation ambiguity, it is estimated by averaging the squared gradients. It is proven that this method is mathematically equivalent to the principal component analysis of the auto-covariance matrix of the gradient vectors [6]. Therefore, the least mean square method based on the gradients [42] is employed in this work to estimate the local ridge orientation of each block. Its processing steps are summarized as follows:

(1) Divide the fingerprint image into non-overlapping blocks of size $w \times w$ pixels.

(2) Compute the gradients $G_x(x, y)$ and $G_y(x, y)$ of each pixel corresponding to the horizontal and vertical directions, respectively. The gradient operator varies from the simple Sobel operator to the complex Marr-Hildreth operator. The Sobel operator is used for simplicity.

(3) Estimate the orientation of each block (i, j) by averaging the squared gradients as:

$$A = \sum_{(x,y)\in W_{i,j}} G_x^2(x,y), B = \sum_{(x,y)\in W_{i,j}} G_y^2(x,y), C = \sum_{(x,y)\in W_{i,j}} G_x(x,y)G_y(x,y) \tag{3.1}$$

$$\theta(i,j) = \frac{1}{2}\arctan\frac{A-B}{2C}, \tag{3.2}$$

where $W_{i,j}$ is the block (i, j) of size $w \times w$ pixels.

3.3.2 Orientation Smoothing

After estimation of the local ridge orientation, the original grey level fingerprint image of size $X \times Y$ pixels is transformed into an orientation field of size $\lfloor \frac{X}{w} \rfloor \times \lfloor \frac{Y}{w} \rfloor$ blocks. The orientation field may still contain the corrupted elements resulted by the minutiae and heavy noise such as scars, ridge breaks and low grey value contrast. Orientation smoothing is often used to further attenuate the noise and partially restore the corrupted orientation elements in the orientation field. A statistical orientation smoothing method is proposed to attenuate the impulsive-like noise [58]. But the resolution of the estimated orientation is limited due to the quantified orientation value. The smoothing method by averaging the unit vectors of the doubled orientation over a neighborhood is widely used because of its high efficiency and resolution [42]. However, the effectiveness and efficiency of this orientation smoothing method highly depends on the determination of the smoothing neighborhood. Large smoothing neighborhood is often required to attenuate the heavy noise and produce a smooth orientation field. But it also blurs the orientations of the high ridge curvature area such as the singular region (i.e., the region near the singular point). The smoothing neighborhood is adaptively determined based on the orientation consistency [44, 45]. If the consistency level of the orientations is small in a local neighborhood of a block, the block is considered as noisy

area and its orientation is smoothed on a larger neighborhood. But the orientation consistency is also small in the areas of high ridge curvature. This method cannot differentiate the noisy area from the area of high ridge curvature where the small smoothing neighborhood is required to capture the orientation of the area center.

We propose an orientation smoothing method that adaptively determines the smoothing neighborhood based on the analysis of the reliability of orientation estimation. The orientation smoothing is performed on the block-wise orientation field. The smoothing neighborhood is adjusted to attenuate the noise while avoiding the orientation blurring. The large smoothing neighborhood is used only when the orientation estimation on a small neighborhood is considered to be unreliable. The reliability measure of orientation estimation is also important in adjusting the smoothing neighborhood. The coherence of the squared gradients is computed by [6]:

$$Coh = \sqrt{(A-B)^2 + 4C^2}/(A+B), \tag{3.3}$$

where A, B and C are computed in Equation (3.1). Coh gives a good measure of how well the gradients over a neighborhood are pointing in the same direction. Thus, it also indicates the reliability of the estimated orientation in Equation (3.2). However, not only the phase angle but also the modulus of the gradient vector affect the value of Coh. Although it is normalized by the average grey value contrast $A+B$, the inconsistent contrast of the grey values in the smoothing window W may drastically change the value of Coh. As a result, Coh may not correctly reflect the orientation consistency in W when the grey value contrasts in the image block are inconsistent.

An orientation consistency is introduced to measure the reliability of orientation estimation based on the preliminarily estimated orientation vectors instead of the gradient vectors. It can avoid the negative effect of the inconsistency of grey value contrasts. In a smoothing neighborhood $\Omega(s)$, the orientation consistency is

computed by:

$$Cons(s) = \frac{\sqrt{(\sum_{(i,j)\in\Omega(s)} \cos(2\theta(i,j)))^2 + (\sum_{(i,j)\in\Omega(s)} \sin(2\theta(i,j)))^2}}{M} \tag{3.4}$$

where M is the number of orientations $\theta(i,j)$ in $\Omega(s)$. If all the orientations in $\Omega(s)$ are exactly directed to one orientation, $Cons(s)$ obtains the highest value of 1. It achieves the lowest value of 0 if the orientations in $\Omega(s)$ are evenly distributed in $[-\pi/2, \pi/2)$. $Cons(s)$ varies between these two extreme situations to provide a quantitative measure of the orientation consistency and reliability. Larger $Cons(s)$ indicates that the estimated orientation is more reliable. If the gradient vector used in Equation (3.1) is converted to a unit vector $[G_x(x,y), G_y(x,y)]/\sqrt{G_x^2(x,y) + G_y^2(x,y)}$ and the $\theta(i,j)$ in Equation (3.4) is the phase angle of the gradient vector, the coherence in Equation (3.3) will be equal to the orientation consistency in Equation (3.4). This equivalence of Equations (3.3) and (3.4) does not hold in general. The suggested orientation consistency based on the normalized orientation vector of each block is more robust to the inconsistency of grey value contrast than the coherence of the squared gradients. Therefore, it is used to measure the reliability of the estimated orientation in a local neighborhood.

The fingerprint ridges flow slowly and smoothly in most areas of fingerprint so that the local ridge orientation usually varies slowly in a local neighborhood. The area with sharp orientation changes may be either the high-curvature area or noisy area. The difference between the high-curvature and noisy areas is that the orientation consistency in Equation (3.4) becomes larger with the increase of the size of the smoothing neighborhood for the noisy area while it keeps small in the high-curvature area. Based on this observation, the analysis of orientation consistencies on the neighborhoods of variant sizes is used to differentiate the high-curvature area from the noisy area of fingerprint. Only the orientations in the noisy area are further smoothed with a larger neighborhood. We increase the size of smoothing neighborhood and compute the orientation consistency with

Equation (3.4). The optimal smoothing neighborhood for each block is adaptively determined by analyzing not only the orientation consistency but also its changes on the different neighborhoods.

If all the orientations in the neighborhood are included for smoothing in the noisy area of fingerprint, the corrupted orientations still deteriorate the final orientation estimation even if a large smoothing neighborhood is used. The proposed smoothing method tries to circumvent the corrupted orientations and use the reliable orientations in its neighborhood to restore the corrupted orientations. If the orientation consistency on a larger neighborhood is better than that on the small neighborhood, the orientations on the outer blocks of the larger neighborhood are more reliable than those on the small neighborhood. To circumvent the corrupted orientations on the small neighborhood, the smoothing neighborhood $\Omega(s)$ only includes the outside surrounding blocks of its $(2s+1) \times (2s+1)$ neighborhood and consists of $8s$ elements. The processing steps of the proposed orientation smoothing method for each block are summarized as follows:

(1) Convert the doubled orientation of each block to a unit vector $[\cos(2\theta(i,j))$ $\sin(2\theta(i,j))]$ and compute $Cons(1)$ (s=1) with Equation (3.4) where $\Omega(1)$ is set to the outer 8 blocks of the 3×3 smoothing neighborhood;

(2) s=s+1. Compute $Cons(s)$ with Equation (3.4) where $\Omega(s)$ is set to the outer $8s$ surrounding blocks of the $(2s+1) \times (2s+1)$ neighborhood;

(3) If $Cons(s)$ is smaller than a threshold (0.5 in our experiment) or smaller than $Cons(s-1)$, go to step (2) until s reaches its maximum (5 in our experiment);

(4) If s equals to its maximum, $\Omega(s)$ is reset to the neighborhood of size 3×3 blocks;

(5) Compute the local ridge orientation of the block by:

$$\theta = \frac{1}{2}\arctan\left(\frac{\sum_{(i,j)\in\Omega(s)} \sin(2\theta(i,j))}{\sum_{(i,j)\in\Omega(s)} \cos(2\theta(i,j))}\right). \tag{3.5}$$

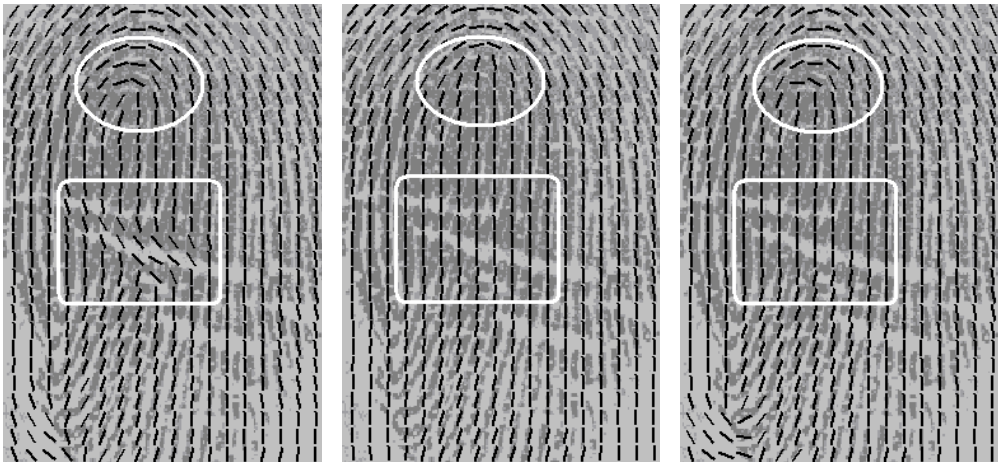
In step (3), the orientation estimation based on $\Omega(s)$ is considered reliable if $Cons(s)$ is larger than both $Cons(s-1)$ and the threshold. If $Cons(s)$ of all scales (s=1, 2, 3, 4) are smaller than the threshold, $\Omega(s)$ is considered as a high curvature area and we reduce the smoothing neighborhood to the minimum $\Omega(1)$.

Finally, an orientation field of fingerprint is composed of the local ridge orientations estimated by uniformly dividing the fingerprint into blocks and smoothed on the adaptively varying neighborhoods.

3.3.3 Experimental Results

Figure 3.1 shows the orientation fields of a poor quality fingerprint smoothed with different settings of neighborhood. From Figure 3.1(a), we can see the corrupted orientation elements have not been well restored using a small smoothing neighborhood although the local ridge orientations near the core point are not blurred. Figure 3.1(b) shows that the corrupted orientation elements are well restored using a larger smoothing neighborhood, but the local ridge orientations near the core point are blurred comparing to those of Figure 3.1(a). Figure 3.1(c) shows that the corrupted orientation elements are well restored while the orientation localization near the core point is maintained by using the proposed adaptive smoothing method. From these results, we can see that the proposed adaptive smoothing method not only maintains the orientation localization of high-curvature area but also has good performance in attenuating noise.

In practice, we cannot know the ground truth of the local orientations in a real poor quality fingerprint. To quantitatively analyze the proposed orientation smoothing method, we add noise to corrupt a good quality fingerprint image. Since the impulsive-like oriented noise such as scar and ridge breaks often appears in real fingerprints [51], the noise is simulated by using white lines of 5 pixels in width and 20 pixels in length. Figure 3.2 shows a good quality fingerprints and the corrupted ones by noises of 15 and 20 random lines. The orientation field of



(a) (b) (c)

Figure 3.1: The orientation fields of a poor quality fingerprint smoothed on different settings of neighborhood: (a) 5×5 blocks; (b) 9×9 blocks; (c) the proposed adaptive smoothing neighborhood. The significant orientation errors caused by the 5×5 and 9×9 neighborhoods are shown in the white rectangle in (a) and circle in (b), respectively.

the good quality fingerprint is considered as the ground truth. The smoothing methods used in [42, 44] are also employed to smooth the orientation fields of the corrupted fingerprints. 400 local ridge orientations are estimated for each image with each block of size 9×9 pixels. To compare the effectiveness of the smoothing methods, the average absolute error is computed over all smoothing regions of 10 images with independent noise. Table 3.1 shows the experimental results. We can see that our method performs better than others.

Table 3.1: The average error of smoothed orientations of fingerprints in Figure 3.2.

Average error	no smoothing	method [42]	method [44]	Our method
Figure 3.2b	5.05°	4.54°	4.06°	2.43°
Figure 3.2c	5.99°	5.12°	4.68°	2.99°

(a) (b) (c)

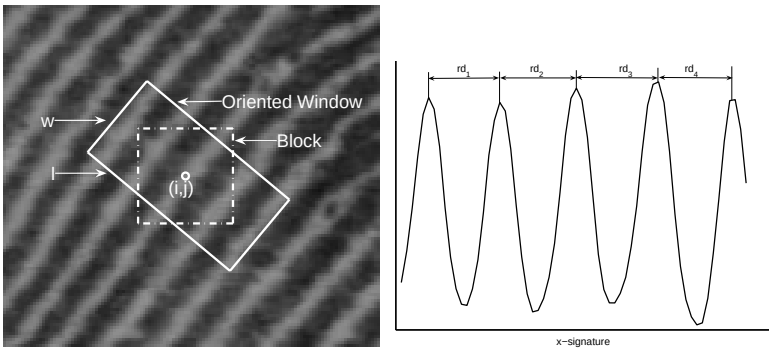
Figure 3.2: (a) A good quality fingerprint, (b) the corrupted fingerprint by noise of 15 lines and (c) the corrupted fingerprint by noise of 20 lines.

3.4 Estimation of Local Ridge Distance

Another evident and important property of fingerprint image is the texture pattern of the interleaved ridges and valleys. In a local region without irregular ridge flows, the ridge and valley are interleaved at a certain frequency and run in parallel. The local ridge distance of fingerprint is defined as the distance between the center points of two adjacent ridges along a line perpendicular to the local ridge orientation. It is another important local parameter of fingerprint as its inverse represents the ridge density or frequency, i.e., the number of ridges per unit length along a line perpendicular to the local ridge orientation. For the fingerprints scanned at 500 dpi, the local ridge distance varies from 3 to 25 pixels [42]. It varies not only across the fingerprints from different fingers but also across the different regions in the same fingerprint. The local ridge distance has been used as an important parameter for fingerprint image enhancement [42].

Similar to the estimation of local ridge orientation, the local ridge distance is estimated by dividing the fingerprint image into blocks of the same size. Several methods have been proposed for the estimation of local ridge distance in the literature [42, 50, 69]. In [69], the local ridge pattern of fingerprint is modelled as a sinusoidal-shaped surface and the variation theorem is exploited to estimate the local ridge distance. In [42], the local ridge distance is estimated by averaging the

distances between two consecutive peaks of the x-signature which is defined as the average grey values along the direction orthogonal to the local ridge orientation. In this method, an oriented window is defined at the center of each block and rotated to align the y-axis with the local ridge orientation firstly (see Figure 3.3). Next, the x-signature is computed by averaging the grey values of the pixels in each column of the oriented window. Finally, the local ridge distance of the block is estimated as the average distance between two consecutive peaks of the x-signature. For example, the ridge distance of the block (i, j) in Figure 3.3 is computed as $\frac{rd_1+rd_2+rd_3+rd_4}{4}$. The method in [50] further makes use of the high-order spectrum technique to estimate the local ridge distance based on the x-signature. The information contained in the second and third harmonics is exploited to enhance the fundamental frequency of the signal.



(a) (b)

Figure 3.3: (a) The oriented window centered at the center of each block and (b) the corresponding x-signature with five peaks.

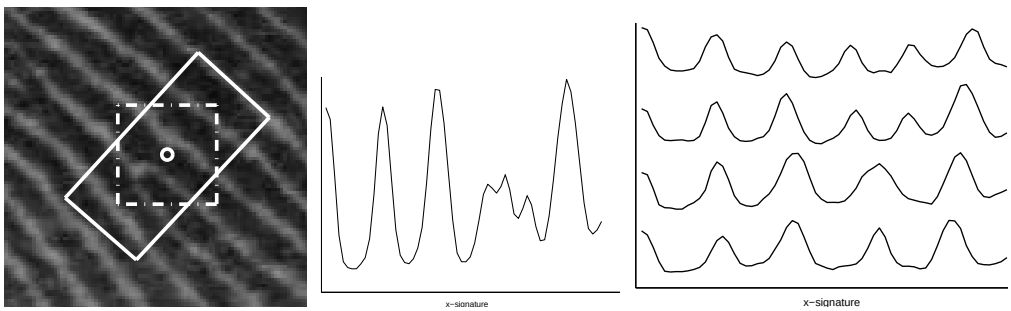
3.4.1 Ridge Distance Estimation

The estimation methods based on the x-signature can perform well when the ridges and valleys in the oriented window have distinct contrast and consistent ridge directions. However, these methods are not very effective to estimate the local ridge distance in the regions with high curvature, irregular ridge flows such as ridge ending and bifurcation, scars and low grey-value contrast because the peaks of the

x-signature can be easily corrupted by them. Figure 3.4(b) shows a corrupted x-signature by a ridge ending. We propose an improvement of this estimation method by dividing the width of the oriented window w into B segments. A x-signature is produced for each segment by averaging the grey values in each column of the segment instead of the oriented window. Therefore, we can get B x-signatures for each oriented window. Figure 3.4(c) shows an example of 4 x-signatures for an oriented window in Figure 3.4(a). We locate the peak points of each x-signature and compute the distances between two consecutive peak points of all x-signature denoted as rd_1,rd_2,...,rd_n. To attenuate the noise, the local ridge distance of each block (i, j) is finally estimated by

$$rgd_{i,j} = mean\{\forall rd_q | (1 \leq q \leq n) \wedge (|rd_q - \frac{1}{n}\sum_{q=1}^{n} rd_q| \leq b) \wedge (3 \leq rd_q \leq 25)\}, \quad (3.6)$$

where b is a threshold determining the valid distances of consecutive peak points for the ridge distance estimation (b is set 2 pixels in our experiments). For the block corrupted by a minutiae point in Figure 3.4(a), the estimated ridge distance based on 4 x-signatures is 12 pixels while the estimated ridge distance based on one x-signature is 21 pixels.



(a) (b) (c)

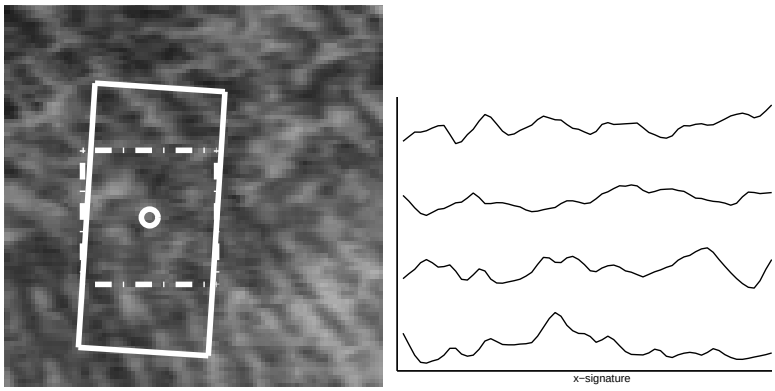
Figure 3.4: (a) The oriented window with a ridge ending, (b) the corrupted x-signature and (c) the 4 x-signatures.

In addition, fingerprint may be corrupted by heavy noise so that the corrupted areas will have no clear ridge and valley flows (see Figure 3.5(a)). The estimation of the local ridge distance cannot be reliable in these areas. We identify such blocks

with the unreliable estimation of the local ridge distance by computing the ratio:

$$r = \frac{\#\{\forall rd_q | (1 \leq q \leq n) \wedge (|rd_q - \frac{1}{n}\sum_{q=1}^{n} rd_q| \leq b) \wedge (3 \leq rd_q \leq 25)\}}{n} \quad (3.7)$$

If r is larger than a threshold, the estimated local ridge distance is reliable. Otherwise it is considered to be unreliable (see Figure 3.5).



(a) (b)

Figure 3.5: (a) The oriented window and (b) four x-signatures of a heavily corrupted block.

Let I be the grey values of a fingerprint image. The fingerprint image is divided into blocks of size $w \times w$ to estimate the local ridge distances. The process steps for the estimation of the local ridge distance in each block are summarized as follows:

(1) Define an oriented window of size $w \times l$ at the center of block (i, j) with the v-axis rotated to align with the local ridge orientation $\theta_{i,j}$ (see Figure 3.3 (a)). The grey values of the oriented window in the transformed coordinate system (u, v) are computed as:

$$\begin{aligned} W(d,k) &= I(u,v), \text{for } 0 \leq d \leq w, 0 \leq k \leq l \\ u &= i + (d - w/2)\cos\theta_{i,j} + (k - l/2)\sin\theta_{i,j} \\ v &= j + (d - w/2)\sin\theta_{i,j} + (l/2 - k)\cos\theta_{i,j} \end{aligned} \quad (3.8)$$

(2) Compute the B x-signatures of each oriented window as:

$$X(m,k) = \frac{B}{w} \sum_{d=(m-1)w/B+1}^{mw/B} W(d,k), m = 1,2,...,B, k = 1,2,...,l \quad (3.9)$$

(3) Compute the distances between two consecutive peak points of all x-signatures and denote them as rd_1,rd_2,...,rd_n.

(4) Compute r using Equation (3.7).

(5) If r is larger than a threshold (0.6 in our experiments), the local ridge distance is computed with Equation (3.6). Otherwise, it is denoted as an unreliable element.

3.4.2 Experimental Results

Similar to the evaluation of orientation smoothing, we also add noise to a good quality fingerprint to test the effectiveness of the proposed method. The test fingerprints in Figure 3.2 are also used to test the estimation of local ridge distance. The local ridge distances of the good quality and the corrupted fingerprints are estimated with both the proposed method and the method in [42]. Similarly, to evaluate the estimation of local ridge distance, the average absolute estimation error is computed over all estimation regions of 10 images with independent noise. The experimental results are shown in Table 3.2. We can see that the average absolute estimation errors by our proposed method are smaller than those by the method in [42].

Table 3.2: The estimation error of local ridge distance of fingerprints in Figure 3.2.

Estimation error (pixels)	method in [42]	Our method
Figure 3.2b	0.57	0.30
Figure 3.2c	0.68	0.41

3.5 Reference Point Detection

A good reference point for fingerprint alignments should be consistently detected for all types of fingerprints including the plain arch fingerprint in which no singular points exist. In addition, the detection of reference point should be robust to noise such as ridge cracks and scars etc.. To reflect the pose transformation of fingerprint, the detection of reference point in this section includes locating a reference point and computing a reference direction corresponding to the reference point. The reference point is located based on multi-scale analysis of the orientation consistency. A reference direction is determined by finding the radial direction from the reference point that is most parallel to the local ridge orientations along the radial. For plain arch fingerprints, two such directions can be found and the average of them serves as the reference direction.

3.5.1 Reference Point Locating

The singular points, i.e., core and delta points, are the landmark points in a fingerprint image where the local ridge orientation changes more rapidly than that in other areas (see Figure 2.1(a)). However, the delta point is easily left outside of the fingerprint due to the small size of the sensor, especially of the dab fingerprints captured by the solid-state sensor. The core point often exists in the central area of fingerprint and is thus more reliable for the alignment purpose. It is widely employed as the reference point for the translational alignment of fingerprints [12, 49, 66]. However, the number of core points differs in different types of fingerprints [55]. For example, there are no core points in strict sense in plain arch fingerprints and two core points usually exist in whorl fingerprints. For the consistent detection of a reference point in all types of fingerprints, we define the reference point as the point with maximum curvature on the convex ridge. If core points exist in a fingerprint, the core point on the convex ridge is the reference point (see Figure 3.8(b)). Otherwise, the point with maximum curvature is always on the convex

ridge and can be consistently detected (see Figure 3.8(c)).

Some approaches are proposed to detect the singular points in the literature [55, 60, 107]. Most of them are based on the orientation field of fingerprint. The Poincare index (PI) method is one of the commonly used methods for the detection of singular points [55, 107]. It computes the PI of each block by summing up the direction changes around a closed digital curve of the block. Although this method is efficient, it is not effective to detect the reference point in plain arch fingerprint as it is not a core point in strict sense. The singular points are detected based on searching the point with maximum curvature [60]. This method cannot work well in poor quality fingerprints as the curvature computation is sensitive to noise. A sine map based method is proposed to detect a reference point based on multi-resolution analysis of the differences of sine component integration between two defined regions in the orientation field [46]. This method is robust to noise, but the two defined regions are sensitive to the rotation of fingerprint. A novel method is proposed for the detection of reference point based on orientation pattern labelling [84]. Although this method is computationally efficient, the orientation pattern labelling is sensitive to the rotation of fingerprint.

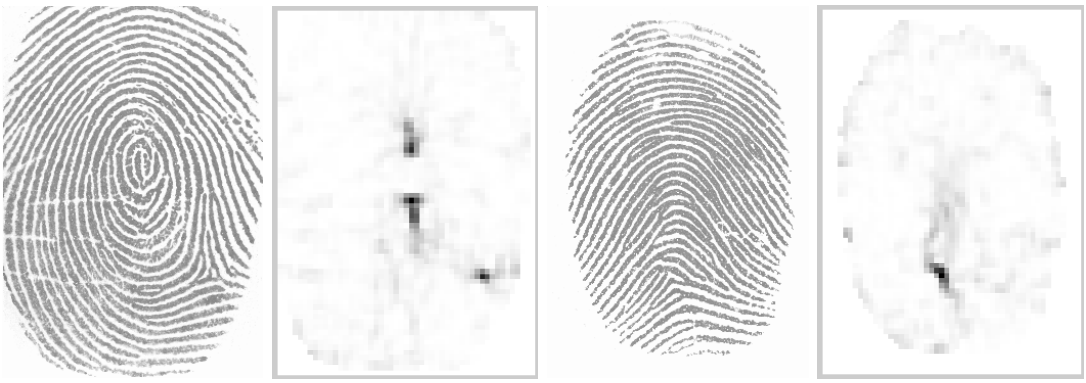
We propose to locate the reference point based on multi-scale analysis of the orientation consistency, which indicates how well the orientations in a neighborhood are consistent with the dominant direction. The introduced orientation consistency in Equation (3.4) is based on the normalized orientation vector of each block. It is more robust to the variance of grey value contrasts than the coherence of squared gradients when evaluating how well the orientations in a neighborhood are consistent. As analyzed in orientation smoothing, the orientation consistency in the high-curvature area is smaller than that of homogeneous area (see Figure 3.6). Hence, it is used for the detection of reference point in which the orientation consistency is minimum in a local neighborhood. However, delta points and the core point in the concave ridge of whorl fingerprints also have minimum orientation consistency in a local neighborhood (see Figure 3.6(a)). We discriminate the

reference point on the convex ridge from those spurious points according to the direction of curvature. In the $(2s+1) \times (2s+1)$ neighborhood of block (i,j), we compute:

$$dx_{i,j}(s) = \sum_{t=-s}^{s} \cos\left(2\theta(i-s, j+t)\right) - \sum_{t=-s}^{s} \cos\left(2\theta(i+s, j+t)\right), \tag{3.10}$$

$$dy_{i,j}(s) = \sum_{t=-s}^{s} \sin\left(2\theta(i+t, j-s)\right) - \sum_{t=-s}^{s} \sin\left(2\theta(i+t, j+s)\right), \tag{3.11}$$

If both $dx_{i,j}(s)$ and $dy_{i,j}(s)$ are larger than 0, the block (i, j) is considered to be on the convex ridge and selected as candidate blocks for the reference point. From the selected blocks on the convex ridge, we choose the one with minimum orientation consistency as the reference point.



(a) (b)

Figure 3.6: (a) A whorl fingerprint and its orientation consistency field, (b) A plain arch fingerprint and its orientation consistency field. White block denotes high orientation consistency.

In addition, the consistency of the corrupted orientation elements in the noisy area of fingerprint may also be small which will result in producing the spurious reference points. As discussed earlier in orientation smoothing, we can differentiate the high curvature area from the noisy area based on the analysis of both the orientation consistency and its changes on the neighborhoods of varying sizes. The orientation consistency in the high curvature area is always small on the neighborhoods of varying sizes while the orientation consistency in the noisy area on a large

neighborhood is usually larger than that on a small neighborhood. To reliably detect the reference point, we search the block with minimum orientation consistency from large scale to fine scale (see Figure 3.7). Similar to orientation smoothing, the orientation consistency on each scale is computed based on the outer surrounding blocks of the neighborhood to circumvent the corrupted orientations. The spurious reference points in the noisy areas are eliminated in the large scale while the accurate reference point with high ridge curvature is finally lacated in the finest scale.

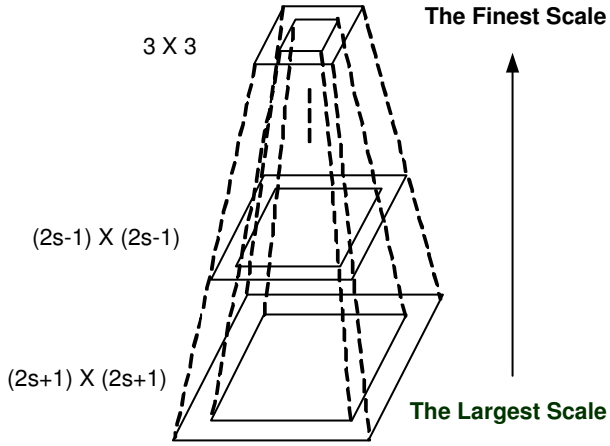


Figure 3.7: The multi-scale analysis of the orientation consistency.

As a result, the block on the convex ridge, which contains the minimum orientation consistency from both large and fine scales, is located as the reference point. Given an orientation field of fingerprint, the processing steps of the proposed approach for reference point locating are summarized as below:

(1) Set s to the largest scale of the neighborhood considered (4 in our experiments);

(2) Compute $dx_{i,j}(s)$ and $dy_{i,j}(s)$ of each block with Equation (3.10) and (3.11), respectively;

(3) Select the blocks with both $dx_{i,j}(s)$ and $dy_{i,j}(s)$ larger than 0 as the candidate blocks for the reference point;

(4) Compute the orientation consistencies $Cons(s)$ of all the selected candidate blocks with Equation (3.4) based on the outer $8s$ surrounding blocks of the $(2s+1) \times (2s+1)$ neighborhood;

(5) Find the block with the minimum orientation consistency $Cons(s)$;

(6) Set $s = s - 1$ and go to step (2) in the selected blocks until $s = 1$;

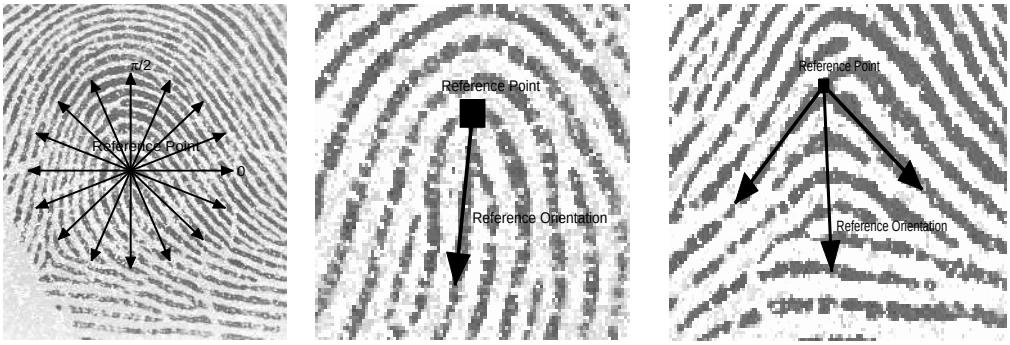
(7) Locate the block with the minimum orientation consistency $Cons(1)$ as the reference point.

3.5.2 Reference Direction Computation

There are few papers concerned with the computation of a reference direction for fingerprint alignment. A good reference direction should be consistently computed for all types of fingerprint to reflect the rotation of fingerprint. An orientation associated with a singular point is computed by comparing the orientation field near the singular point and a standard orientation model [6]. This method performs well if the orientation field near the singular point is very similar to the standard reference model. But it cannot work well for plain arch fingerprints since this type of fingerprint does not belong to any one of the proposed two reference models.

To define a consistent reference direction for all types of fingerprints, 16 directions are radiated from the reference point with $\frac{\pi}{8}$ equal interval (see Figure 3.8(a)). In a neighborhood of the reference point, the local ridge orientation most parallel to the nearest radial direction is consistent for the fingerprints whose reference points are core point and thus can reflect the fingerprint rotation (see Figure 3.8(b)). Thus, this local ridge orientation is defined as the reference direction of the core point. For plain arch fingerprint whose reference point is not core point in strict sense, there exists two different such local ridge orientations. The average of them is defined as the reference direction (see Figure 3.8(c)).

To consistently and reliably compute the reference direction, we propose to



(a) (b) (c)

Figure 3.8: (a) The 16 radial directions from the reference point, (b) The reference point and direction for core point, (c) The reference point and direction for plain arch fingerprint.

analyze the differences between the radial directions from the reference point and the local ridge orientations along the corresponding radial. The absolute sine component of the orientation difference is employed to approximate the difference between the radial direction and the local ridge orientations. The absolute sine components of the differences between the radial direction θ_k and the local ridge orientations $\theta(i,j)$ along the corresponding radial are averaged as:

$$Var(k) = \frac{1}{M} \sum_{(i,j)\in\Omega_k} |\sin(\theta(i,j) - \theta_k)|, \theta_k = k\pi/8, k = 0, 1, ..., 15 \qquad (3.12)$$

where Ω_k is a set of M local ridge orientations along the radial with direction θ_k. It is a rectangle with its length side parallel to the radial and symmetric with respect to the radial. Obviously, $Var(k)$ of range [0 1] equals to 1 when θ_k is orthogonal to all the orientations in Ω_k and equals to 0 when θ_k is parallel to them. Therefore, the dominant local ridge orientation in Ω_k with the minimum $Var(k)$ is considered to be most parallel to the radial direction θ_k. For the fingerprint whose reference point is core point, only one local minimum exists in $Var(k)$ for $k = 0, 1, ..., 15$ (see Figure 3.9(a)). For plain arch fingerprint in which the reference point is not core point in strict sense, two local minimums exist in $Var(k)$ (see Figure 3.9(b)). In order to effectively search the minimum of $Var(k)$, the size of the orientation

set Ω_k is adaptively changed. The number of blocks along the side orthogonal to and centered with the radial, i.e., the width of Ω_k is set to 5. The length of Ω_k is initialized to 4 blocks (this depends on the minimum distance between two core points) and is adjusted by the analysis of $Var(k)$. The processing steps of the proposed approach to compute the reference direction are summarized as below:

(1) The length and width of Ω_k are initially set to 4 and 5 blocks, respectively;

(2) Compute $Var(k)$ with respect to the 16 radial directions with Equation (3.12) and find the minimum of $Var(k)$ as $Var(k_{min})$;

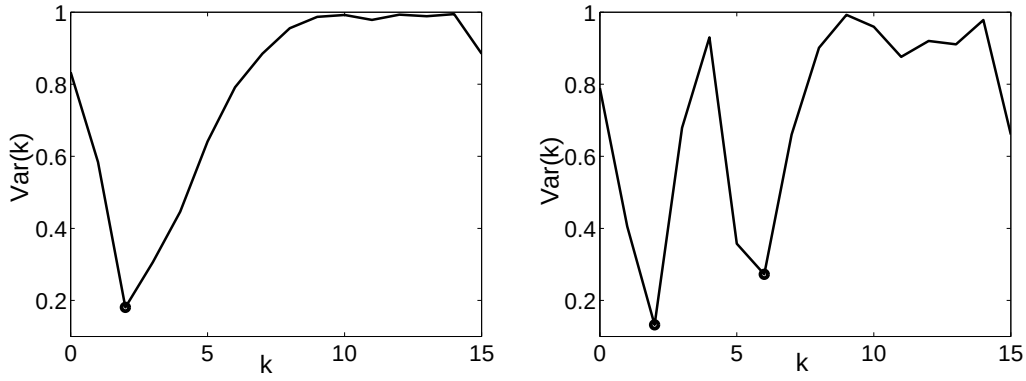
(3) Select the radial directions θ_k with $Var(k) < Var(k_{min}) + 0.1$ as the candidate directions. If two or more such directions are continuous with k, select the radial direction with the minimum $Var(k)$ from them as one candidate direction;

(4) If more than two candidate directions exist and the length of Ω_k does not reach the boundary of image or its maximum (15 in our experiment), increase the length of Ω_k by 1 and go to step (2);

(5) Compute the dominant local ridge orientations of Ω_k with respect to the candidate radial directions using the least mean square averaging method. The average of these dominant orientations is computed as the reference direction.

3.5.3 Experimental Results

The proposed algorithm for the reference point detection has been tested on the FVC2000 DB2 set A, in which 800 fingerprint images from 100 fingers (with 8 images from each finger) are captured using a low cost capacitive sensor. The image size is 364×256 pixels and the resolution is 500 dpi. This database contains many poor quality fingerprints such as the partial images with the reference point



(a) (b)

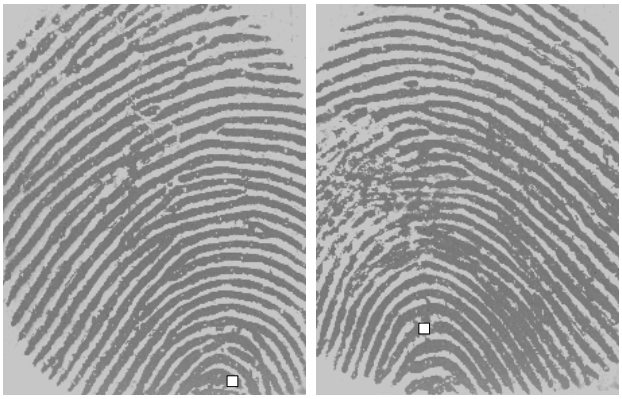
Figure 3.9: $Var(k)$ against k where circle denotes the minimum: (a) one minimum and (b) two local minimums.

left outside and the noisy images corrupted by scars, ridge breaks and too wet or dry etc.. The desired position and orientation of the reference point in each fingerprint are not detected previously by the experts. To evaluate the performance of our proposed algorithm quantitatively, we manually locate the desired reference point and the desired reference orientation for each fingerprint.

There are 13 partial fingerprint images in the test database with the reference point left outside. For these images, no reference point should be detected. In our experiments, one point with minimum orientation consistency is usually detected for each fingerprint. If the detected point is on the boundary of image (see Fig. 3.10a), this image is identified as partial fingerprint image. In this way, we can correctly identify 11 partial fingerprint images while two partial fingerprints fail as the detected reference points are not located on the image boundary (see an example of Fig. 3.10b).

Reference Point Locating

In our experiments, the position of the reference point is the center pixel of the finally located block. The Euclidean distance between the manually located position and the position located by the algorithm is computed as the distance error of the reference point. Since the manually located reference point may have some



(a) (b)

Figure 3.10: The identification of partial fingerprint image: (a) the correct identification with the detected point on the boundary of image, (b) the false identification with the detected point in the internal region of image.

deviation from the true one, four coarse level measures of the distance error are defined to evaluate the accuracy of reference point locating. If the distance error is not larger than 10 pixels (about 1 inter-ridge), the reference point is considered to be accurate as the error may be caused by human vision. If the distance error is larger than 10 pixels but not larger than 20 pixels, it is considered as small error which may be caused by both human vision and algorithm. If the distance error is larger than 20 pixels but not larger than 40 pixels, it is considered as significant error which may have negative effect on the subsequent processing steps. If the distance error is larger than 40 pixels, most likely a spurious or false reference point is detected that cannot be used for subsequent processing steps. Table 3.3 shows the experimental results in the test database. We can see that the accuracy of the reference point arrives 95.18% if small distance errors can be tolerated in the subsequent processing steps. It can be further improved by reducing the step size of overlapping blocks. Almost all the significant errors and false detections are in the poor quality fingerprints (see Fig. 3.11).

The definition of reference point in [46] is same as that in this work. We compare our approach with the sine map based approach proposed in [46]. Fig. 3.12 shows two examples of reference points located in our approach and the sine map based approach. From Fig. 3.12a, we can see that the detected reference

Table 3.3: The accuracy of the reference point for non-partial fingerprints

Distance Error (pixels)	The Number of fingerprints	The probability
≤ 10	659	0.8374
> 10 and ≤ 20	90	0.1144
> 20 and ≤ 40	25	0.0318
> 40	13	0.0165

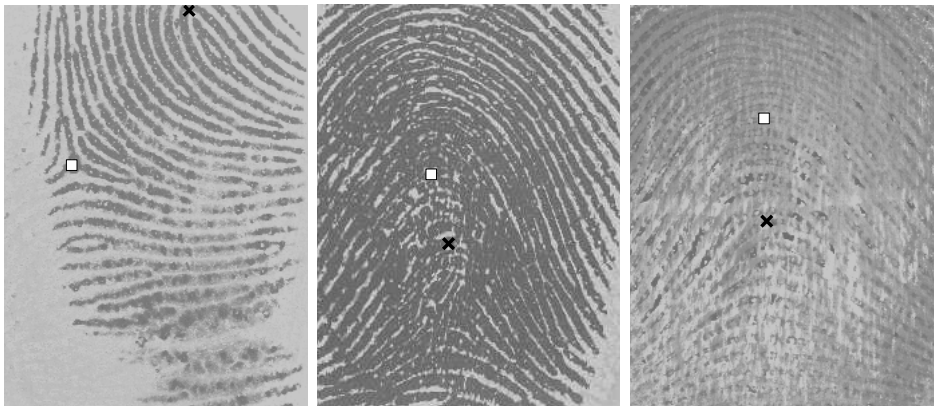


Figure 3.11: Examples of false reference point detection and the detection with significant error ("×" denotes the manually located reference point and "□" denotes the reference point located by the algorithm).

points by the sine map based approach are not very consistent in two fingerprints from the same finger due to a slight rotation between them. Fig. 3.12b shows that the detected reference points by our approach are more consistent in these two fingerprints than those in Fig. 3.12a. Furthermore, the standard deviations of the reference points are computed to compare their consistencies. Let $Z_r(i)$ and $Z(i)$ be the positions of the manually located reference point and the detected reference point of the fingerprint i from the same finger, respectively. The standard deviation σ is computed as:

$$\sigma = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\|dZ(i) - \frac{1}{M}\sum_{i=1}^{M}dZ(i)\|^2} \tag{3.13}$$

$$dZ(i) = Z(i) - Z_r(i) \tag{3.14}$$

where M is the number of fingerprints (8 in our test database) from the same finger. As for the 8 fingerprints from the same finger as those shown in Fig. 3.12,

the σ of the detected reference points by the sine map based approach is 6.0228, while it is 2.4708 by our approach. The average standard deviation of the whole test database (800 fingerprints) by our approach is 11.5109, which is smaller than that by the sine map based approach (19.7800). Therefore, the consistency of the detected reference points by our approach is better than that by the sine map based approach.

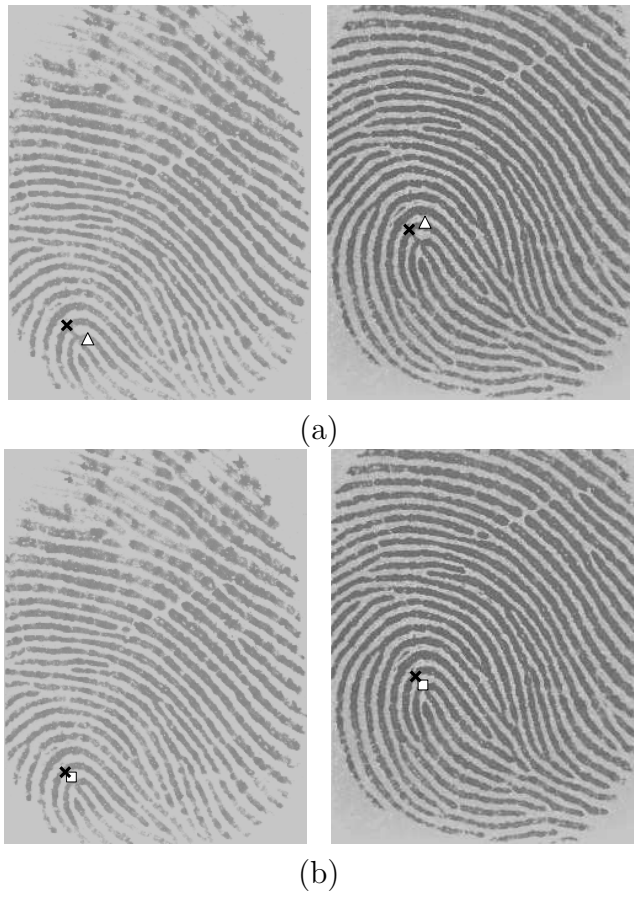


Figure 3.12: Examples of the reference points located by (a) the sine map based approach and (b) our approach ("×", "△" and "□" denote the reference points detected by human expert, sine map based approach and our approach, respectively).

Reference Direction Computation

The difference between the manually detected reference orientation and the reference orientation computed by the proposed approach is calculated as the error

Table 3.4: The accuracy of the reference orientations on the test database.

OE	Number of fingerprints	Probability
$\leq\pi/16$	690	87.67%
$>\pi/16$ and $\leq\pi/8$	47	5.97%
$>\pi/8$	50	6.35%

of reference orientation computation. Similarly, three coarse level measures of the orientation errors are defined to evaluate the accuracy of the reference orientation. The orientation error larger than $\pi/8$ rads is considered as significant error while the orientation error larger than $\pi/16$ but not larger than $\pi/8$ rads is considered as small error. The orientation error not larger than $\pi/16$ rads is considered as accurate computation. To avoid the ambiguity of the orientation difference, the orientation error (OE) is computed as below:

$$OE = \min\{|\theta_p - \theta_r|, \pi - |\theta_p - \theta_r|\}, \quad -\pi/2 \leq \theta_p, \theta_r < \pi/2, \tag{3.15}$$

where θ_p and θ_r are the computed and the manually detected reference orientations, respectively. Table 3.4 shows the experimental results on the test database. We can see the accuracy is 93.65% if the reference orientation can tolerate error up to $\pi/8$. It can be further improved if we can improve the accuracy of reference point.

3.5.4 Alignment of Fingerprints

Pose transformation often exists in the fingerprint impressions originated from the same finger. Figure 3.13 shows examples of two fingerprints from the same finger with pose transformation. To achieve the invariance of such transformation, translation and rotation are needed to bring two fingerprints into alignment, which is one of the crucial parts for fingerprint verification and identification. Fingerprint alignment based on the minutiae structure and ridge shape used in the verification is time consuming and is thus unsuitable for fingerprint identification. One

common and efficient solution is the alignment of fingerprints based on a reference scheme. Let (x_r, y_r) and θ_r, $-\pi < \theta_r \leq \pi$, be the position of the reference point and the corresponding reference direction, respectively. They can be used for the translational and rotational alignments of fingerprints to achieve the invariance of pose transformation. Figure 3.14 shows some examples of fingerprint orientation fields after smoothing and segmentation with the corresponding detected reference point and direction superimposed on the fingerprints.

Figure 3.13: Examples of two fingerprints from the same finger with pose transformation.

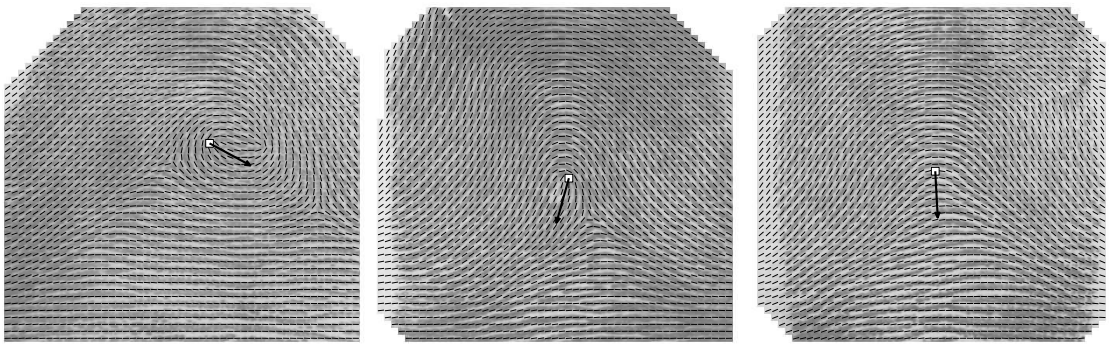


Figure 3.14: Examples of orientation field, reference point and reference direction superimposed on the fingerprint, where the white area without orientation is segmented as the background and the small white square and black arrow denote the reference point and direction, respectively.

3.6 Summary

In this chapter, some techniques have been proposed to robustly and reliably estimate two local parameters of fingerprint: local ridge orientation and local ridge distance, which will play important roles for fingerprint analysis and retrieval in the following chapters. We propose a new orientation smoothing method based on the adaptive neighborhood which not only attenuates the noise well but also maintains the orientation locating in the high curvature area. Instead of computing only one x-signature in the oriented window of each block, we propose to estimate the local ridge distance based on more than one x-signatures which is more robust to noise and irregular ridge flows. In addition, an effective method has been proposed to consistently locate a reference point and compute a corresponding reference direction for all types of fingerprints. The reference point is located based on multi-scale analysis of the orientation consistency, while the reference direction is computed by analysis of the orientation differences between 16 redial directions from the reference point and the local ridge orientations along these radii. They can be used for the alignment of fingerprints to achieve the invariance of pose transformation. In the next chapter, we will develop a fingerprint retrieval algorithm based on the continuous classification and using the results obtained in this chapter.

Chapter 4

Fingerprint Retrieval Based on Continuous Classification

4.1 Introduction

As mentioned in Chapter 2, the traditional exclusive classification provides an efficient indexing mechanism to speed up the search of fingerprint database for large-scale identification. However, it cannot sufficiently reduce the search of fingerprint database due to the small number of predefined classes and uneven fingerprint distribution in the classes. In addition, it is still a difficult problem to consistently and exclusively classifying fingerprints, especially the poor quality fingerprints, due to the small inter-class variance and large intra-class variance. Thus, the retrieval efficiency and accuracy are limited on exclusive fingerprint classification. In fact, it is usually not a practice to classify fingerprints into human-interpretable classes for an AFIS in some applications.

To avoid the difficult problems of exclusive fingerprint classification, an attractive approach called "continuous classification" is proposed to facilitate the search of fingerprint database [66]. Instead of classifying fingerprints into a small number of human interpretable classes, this approach represent each fingerprint with the

numerical feature vectors. Similar fingerprints are mapped into close points in the multi-dimensional feature space by a given distance measure. The database templates close to the query fingerprint are retrieved for the fine matching of fingerprint identification. The tradeoff between the penetration rate (the portion of retrieved database) and the retrieval accuracy can be easily adapted by adjusting the size of retrieval neighborhood. This approach can achieve better retrieval performance than the exclusive classification. Some algorithms are proposed on the continuous classification in the literature [11, 14, 17, 53], but their retrieval performances are still not very attractive. Further works on this technique are still of great interest to the researchers in the fields of fingerprint retrieval and identification.

In this chapter, we develop a fingerprint retrieval algorithm based on the continuous classification. An orientation vector of fixed size is constructed from the local ridge orientation field of fingerprint and is used as the main retrieval feature. A dominant ridge distance is computed as an auxiliary retrieval feature. A new distance measure is proposed to quantify the distance between two orientation vectors and is compared with the conventional Euclidean and Manhattan distance measures. In addition, we propose a new regional feature weighting scheme which puts larger weights on the orientation elements with the more powerful variation in the test database to compare the orientation vectors. As a result, the representation power of the orientation feature is improved for the fingerprint retrieval. Furthermore, a variable search tolerance is introduced for the effective retrieval. Finally, the proposed fingerprint retrieval algorithm is performed on NIST database-4 [102] and compared with some state-of-the-art approaches to test its effectiveness.

4.2 Feature Extraction

As mentioned in Chapter 2, fingerprint retrieval is a coarse level search to reduce the search space of the fine matching in an AFIS. An ideal retrieval feature set should have the following properties: easily computable and compact, translation

and rotation invariant, noise robust and discriminating over a large number of fingerprints. In addition, it should be independent on or loosely correlated with that used in the fine matching to avoid redundant representation in an AFIS. The local ridge orientation and local ridge distance are two important local parameters for fingerprint analysis. In Chapter 3, we have proposed some techniques to robustly and reliably estimate them. These two local parameters are the coarse level features of fingerprint and are low correlated with each other. Moreover, both of them have low correlation with the fine level feature, minutiae points, that are often used in the fine matching. Thus, we combine these two features to represent fingerprint for the retrieval. An orientation vector of fixed size is constructed from the local ridge orientation field of fingerprint as the main retrieval feature. A dominant ridge distance is computed from the local ridge distances of fingerprint as an auxiliary retrieval feature.

4.2.1 Construction of Orientation Vector

The orientation field of fingerprint is a discrete matrix composed of the local ridge orientations estimated at the discrete positions of fingerprint. It describes the global information of the fingerprint ridge and valley flows and varies across different fingerprint patterns. In addition, the orientation field is loosely correlated with the minutiae features that are often used in the fine matching. As mentioned in Chapter 2, it is widely used as the main feature for fingerprint classification. It is tested in [12, 104] that the performance of fingerprint classification is improved by replacing the uniform spacing orientation field (i.e., orientation field computed by dividing fingerprint into blocks of same size) with an orientation field computed on a nonuniform spacing of the fingerprint image. The nonuniform spacing concentrates the orientation measurements more densely on the regions more likely to contain core and delta points. It is further found that performance improvement similar to that of nonuniform spacing is achieved by applying regional weights to the uniform spacing orientation field in comparing the orientation vectors [12,104].

We adopt the uniform spacing orientation field same as that used in the fingerprint image enhancement and minutia extraction to avoid the double computation of the orientation field. A regional weighting scheme will be investigated to compare the orientation vectors. The proposed method in Chapter 3 is used to estimate each local ridge orientation.

For construction of orientation vector, each local ridge orientation is often represented by a unit vector of the doubled orientation $[\cos(2\theta)\ \sin(2\theta)]$ instead of the phase angle θ [12,53,66]. This representation is able to facilitate the feature transform and weighting. But it doubles the size of the feature vector and increases the storage and computation cost in the fingerprint retrieval. For example, the orientation vector consists of 1680 elements if the fingerprint of size 512×480 pixels is divided into blocks of 16×16 pixels to estimate the orientation [66]. The Karhunen-Loeve (KL) or multi-space KL transform is used to reduce the dimensionality of orientation vector into a much lower one in such a way that the Euclidean distances between two vectors are approximately preserved [12, 15, 53, 66]. The basic idea of the KL transform is to select the combinations of elements which contain the most representation power in the test database. To implement the KL transform, each element of the original feature vectors is valid to produce a sample covariance matrix. However, in the practical applications, the orientation vectors may have quite different valid and invalid elements due to the variant translation, rotation and background areas in different fingerprints. This may negatively affect the reliability of the covariance matrix and the obtained transform matrix. Furthermore, the transform of the feature vector into the eigen-space combines all elements of the feature vector in the original feature space. If the original feature vector has a substantial number of invalid elements, the transformation may result in large error. Therefore, we do not apply such transform technique although it may bring some advantages such as reducing the dimensionality of feature vector and facilitating the feature weighting. As we will see later, if the Euclidean distance measure is applied to compare two orientation vectors, the orientation representation by the unit vector has similar function as that by the phase angle θ. Therefore, we repre-

sent each local ridge orientation with the phase angle to reduce the dimensionality of the feature vector by half.

To achieve the invariance of pose transformation, the position of reference point (x_r, y_r) and the corresponding reference direction θ_r $(-\pi < \theta_r \leq \pi)$ detected in Chapter 3 are used for the translational and rotational alignments of the orientation fields. The new coordinates of the aligned orientation field, represented by a complex variable $m + jn$, are calculated by

$$m + jn = [x - x_r + j(y - y_r)]e^{-j\theta_r} \tag{4.1}$$

where (x, y) are the coordinates of the orientation field before alignments. Due to the periodicity and discontinuity of the orientation $\theta_{m,n}$ at $\pm\pi/2$ and the direction θ_r at $\pm\pi$, the aligned local ridge orientation $\hat{\theta}_{m,n}$ is computed by

$$\hat{\theta}_{m,n} = \begin{cases} \Delta\theta, & \text{if } -\pi/2 < \Delta\theta \leq \pi/2, \\ \Delta\theta - \pi, & \text{if } \Delta\theta > \pi/2, \\ \Delta\theta + \pi, & \text{if } \Delta\theta \leq -\pi/2, \end{cases} \tag{4.2}$$

where $\Delta\theta = \theta_{m,n} - \theta_r$. Obviously, $-\pi/2 < \hat{\theta}_{m,n} \leq \pi/2$.

For the fingerprint retrieval based on continuous classification, an orientation vector of fixed size is constructed by concatenating the aligned local ridge orientations $\hat{\theta}_{m,n}$ to represent each fingerprint. Thus, all orientation vectors should be cropped into a canonical size. In practice, the canonical size of orientation vector depends on the size of the fingerprint image and the validity of the orientation elements. If the reference point is located at the center of fingerprint image, the orientation vector covers most crucial area of the fingerprint. Otherwise, it includes only a part of the fingerprint and a substantial number of orientation elements are from the noisy background or the outside of the image if the reference point is located near the background or the border of image such as in the partial image. The segmentation algorithm is therefore used to label the valid and invalid elements

of each orientation vector. The segmentation result of fingerprint q is denoted by a vector $S^q = [s^q_{1,1}, s^q_{1,2}, ..., s^q_{X,Y}]$ where $s^q_{m,n} \in \{0, 1\}$ and $s^q_{m,n} = 1$ indicates that element (m, n) is valid for feature selection. For example, we estimate the valid probability of each element of the orientation field aligned by the reference point and reference direction pointing to south-most using the first fingerprint instances of the NIST database-4. Figure 4.1 shows the probability map. We can see that the elements far from the reference point is more likely to be invalid than those near the reference point. This is because the border of fingerprint is more possible to be background or corrupted by the heavy noise than the central region.

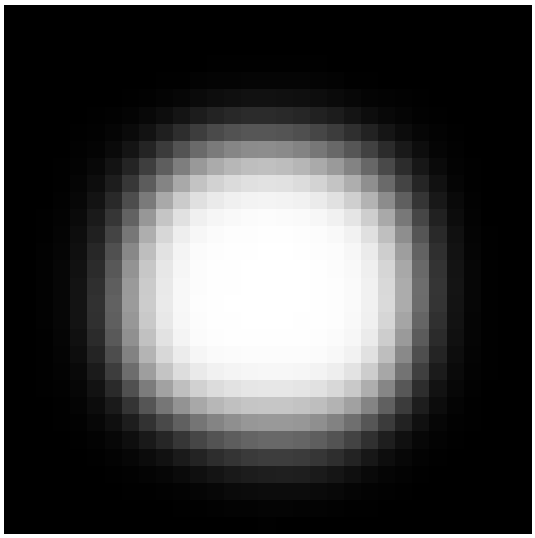


Figure 4.1: The valid probability map of the aligned orientation field estimated on the NIST database-4 (white area indicates high probability).

Obviously, the more valid values exist in one element for different fingerprints, the more important it is for the fingerprint representation. Therefore, the orientation vector of fixed size $O = \{o_k, k = 1, 2, ..., M\}$ is constructed by concatenating the aligned local orientations within a circle of radius R, excluding the one nearest to the reference point, i.e.,

$$O = \{\forall \hat{\theta}_{m,n} | (m^2 + n^2 < R^2) \wedge ((m, n) \neq (0, 0))\}. \tag{4.3}$$

The orientation nearest to the reference point is excluded from the orientation

vector because this element is of high curvature and its orientation estimation is unreliable. Determination of the parameter R depends on the size of fingerprint image. For the application to the NIST database-4, the orientation field is computed by dividing the fingerprint into blocks of size 27×27 pixels and R is set to 8. The constructed orientation vector consists of $M = 192$ elements, which covers the fingerprint image of size $27[2(8-1)+1] \times 27[2(8-1)+1] = 405 \times 405$. The dimensionality of the feature vector is equal to that of "FingerCodes" [49]. Figure 4.2 shows some examples of the smoothed fingerprint orientation fields, the valid orientation elements for the construction of feature vector and the corresponding detected reference point and direction.

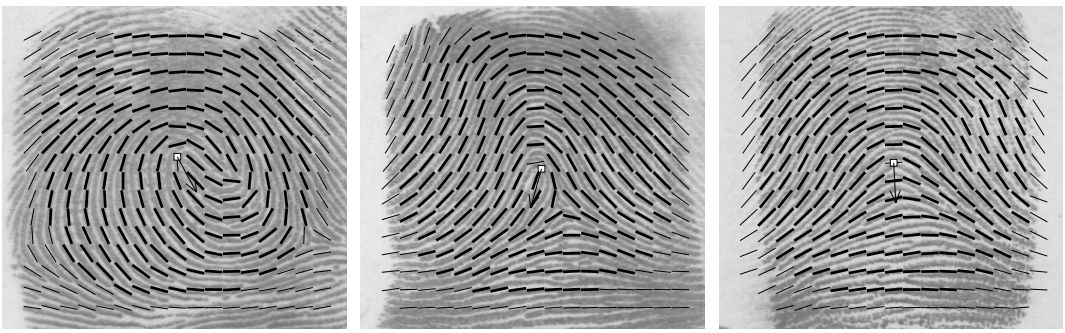


Figure 4.2: Examples of orientation field, reference point and direction superimposed on the fingerprint, where short (thicker / thinner) line represents for the local orientation (included in / excluded from the feature vector), the small square and arrow for the reference point and direction, respectively.

4.2.2 Computation of Dominant Ridge Distance

In addition to the local ridge orientation, the local ridge distance is another intrinsic property of fingerprint image. The local ridge distance map $\{rgd_{x,y}\}$ of fingerprint can be robustly estimated by the proposed method in Chapter 3. However, the local ridge distance may vary across different fingerprints from the same finger due to the different manners a elastic finger presses on a plane sensor. In addition, the local ridge orientation is an important parameter for the estimation of the local ridge distance so that unreliable local ridge orientation can result in large estimation

error of the local ridge distance. Moreover, noise and image deterioration may also result in large estimation error of the local ridge distance. Thus, the estimated local ridge distance is much less stable than the local ridge orientation. It seems not viable to construct a high dimensional feature vector using the local ridge distances for the fingerprint retrieval.

Nevertheless, the average ridge distance ARD over the fingerprint foreground shows a stable yet discriminating scalar feature. The average within-finger variance of the ARD over the NIST database-4 is 0.033 and its between-finger variance is 0.494, which lead to a discriminant value of 0.494/0.033=14.97. We further find that some local ridge distances largely deviate from the mean due to the existence of minutia, ridge break, bad ridge separation and other heavy noise. These outliers negatively affect the stability of the ARD. Therefore, we define a dominant ridge distance DRD as the second mean over the local ridge distances within a bin centered at the first mean over the fingerprint foreground $\mathcal{F}$ by:

$$DRD = mean\{\forall rgd_{x,y} | ((x,y) \in \mathcal{F}) \wedge (|rgd_{x,y} - ARD| < b)\}, \tag{4.4}$$

where

$$ARD = mean\{\forall rgd_{x,y} | (x,y) \in \mathcal{F}\}. \tag{4.5}$$

With a proper choice of threshold b, outliers of the local ridge distance will be excluded from the computation of the dominant ridge distance DRD. The average within-finger variance of the DRD with $b = 2$ over the NIST database-4 is 0.028 and its between-finger variance is 0.665, which lead to a discriminant value of 0.665/0.028=23.75. We can see that it is much better than the discriminant value 14.97 of the ARD. Obviously, the DRD is discriminating, loosely correlated with the orientation field and the minutia features. It can bring new information to represent fingerprint for the retrieval of an AFIS with the minutia features used in the fine matching. In addition, this scalar feature is invariant to the translation and

rotation of fingerprint image without the requirement of the alignments. Therefore, the DRD computed with Equations (4.4) and (4.5) is employed as an auxiliary scalar feature for our fingerprint retrieval based on continuous classification.

4.3 An Orientation Distance Measure

Euclidean and Manhattan distance measures have been widely used in quantifying the distance between two numerical vectors. The constructed orientation vector to represent fingerprint in the above section is composed of phase angles which are periodic, discontinuous at $\pm\pi/2$ and can be valid or invalid. Let $O^p = \{o_k^p, k = 1, 2, ..., M\}$ denote the orientation vector of fingerprint p and $S^p = \{s_k^p, k = 1, 2, ..., M\}(s_k^p \in \{0, 1\})$ denote the corresponding segmentation result where $s_k^p = 1$ indicates that the element k of fingerprint p is valid for feature selection. Therefore, the Euclidean distance $d_E(O^p, O^q)$ and Manhattan distance $d_M(O^p, O^q)$ between the orientation vectors of two fingerprints p, q (i.e., O^p and O^q) can be computed by

$$d_E(O^p, O^q) = \sqrt{\frac{1}{\sum_{k=1}^{M} s_k^p s_k^q} \sum_{k=1}^{M} s_k^p s_k^q (\Delta_k^{p,q})^2} \tag{4.6}$$

$$d_M(O^p, O^q) = \frac{1}{\sum_{k=1}^{M} s_k^p s_k^q} \sum_{k=1}^{M} s_k^p s_k^q \Delta_k^{p,q} \tag{4.7}$$

where

$$\Delta_k^{p,q} = min(|o_k^p - o_k^q|, \pi - |o_k^p - o_k^q|). \tag{4.8}$$

If the unit vector $[\cos(2o_k)\ \ \sin(2o_k)]$ is employed to represent each local ridge orientation for construction of the orientation vector, from the trigonometry, we

can easily have

$$\begin{aligned}&(\cos 2o_k^p - \cos 2o_k^q)^2 + (\sin 2o_k^p - \sin 2o_k^q)^2 \\ &= 4\sin^2(o_k^p - o_k^q) = (2\sin \Delta_k^{p,q})^2. \end{aligned} \tag{4.9}$$

Obviously, if Euclidean distance measure is applied, this unit vector representation that doubles the feature vector size maps $0 \leq \Delta_k^{p,q} \leq \pi/2$ in (4.6) to $0 \leq 2\sin(\Delta_k^{p,q}) \leq 2$, which produces very similar distance to that of the angle o_k.

If all the orientation elements between two feature vectors consistently have a constant difference, the fingerprint orientation fields of such two vectors are very similar just with a rotation in human perception. Therefore, the distance between these two orientation vectors should be zero. However, the Euclidean and Manhattan distances of them may achieve a large deviation as a constant difference exists in all vector elements. To overcome this problem, we propose a new orientation distance measure that is based on the inconsistency of the orientation differences among all valid elements. The proposed distance between the orientation vectors of fingerprint p and q (i.e., O^p and O^q) is computed as

$$d_C(O^p, O^q) = 1 - \frac{\left|\sum_{k=1}^{M} v_k e^{j2(o_k^p - o_k^q)}\right|}{\sum_{k=1}^{M} v_k} \tag{4.10}$$

where $v_k = s_k^p s_k^q$ ($v_k \in \{0, 1\}$ and $v_k = 1$ means that the element k is valid for both fingerprints p and q.), $j = \sqrt{-1}$ and $|z|$ computes the magnitude of the complex variable z. Instead of averaging the absolute or squared difference over all the valid orientations with Equation (4.6) or (4.7), the proposed distance measure in Equation (4.10) averages the unit vectors whose phases are the doubled orientation differences. Thus, it quantifies the distance between two orientation vectors based on the inconsistency of the orientation differences among all the valid elements. We can see that the proposed distance $d_C(O^p, O^q)(\in [0\ 1])$ achieves the minimum of zero for a constant orientation difference $o_k^p - o_k^q$ (constant to k) and it increases

when all the valid orientation differences of two fingerprints are inconsistent.

Let a and b be two scalars and I be a unit vector of the same size as O, it is easy to verify that the proposed distance measure satisfies:

$$d_C(O^p + aI, O^q + bI) = d_C(O^p, O^q), \tag{4.11}$$

because

$$\begin{aligned} \left| \sum_{k=1}^{M} v_k e^{j2(o_k^p + a - o_k^q - b)} \right| &= \left| e^{j2(a-b)} \right| \left| \sum_{k=1}^{M} v_k e^{j2(o_k^p - o_k^q)} \right| \\ &= \left| \sum_{k=1}^{M} v_k e^{j2(o_k^p - o_k^q)} \right|. \end{aligned} \tag{4.12}$$

This means that a constant amount of orientation difference has no effect on the proposed distance measure in Equation (4.10). Therefore, the proposed orientation distance measure is invariant to a slight rotation between two aligned fingerprints resulted by a small estimation error of the reference direction. Another merit of the proposed distance measure is that we need not subtract the reference direction from the original orientation field to construct the orientation vector, i.e., Equation (4.2) is not necessary and we can use $\theta_{m,n}$ instead of $\hat{\theta}_{m,n}$ in the feature vector construction of Equation (4.3).

4.4 Regional Feature Weighting

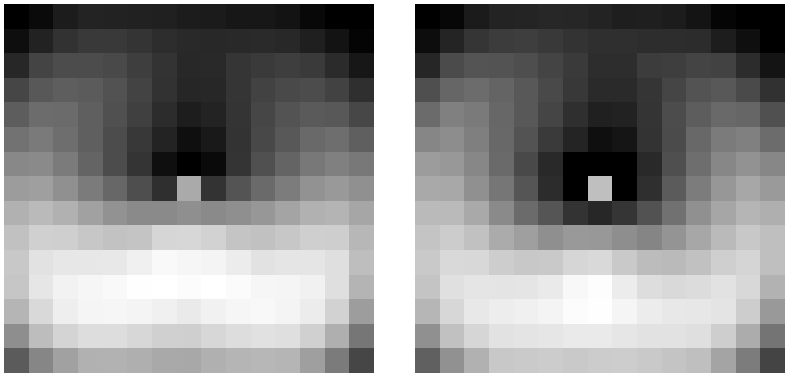
In the aligned orientation fields, the local orientation elements may have variant representation powers to discriminate fingerprints. If all elements are put equal weights to compare the orientation vectors, it will obscure those elements which have more power to discriminate fingerprints. A nonuniform pattern of the regional weights, which assigns greater weights to the elements in the central region of fingerprint, is applied on the orientation elements before performing the KL

transform and computing distances [12]. The results of fingerprint classification are improved comparing to those without regional weighting. In [14], the regional weights are used to attenuate the orientation elements near the border by applying a Gaussian-like function and strengthen those in the irregular regions by imposing the singularity. However, we find that the discriminating power of a local orientation element may not be simply decreased with the increase of its distance to the singular points. As the entropy of a random variable measures its uncertainty, we propose to weight a local orientation by its entropy. The entropy of each orientation element is estimated by:

$$w_k = -\frac{\sum_{l=1}^{L} p(o_k \in B_l) \log p(o_k \in B_l)}{log(L)}, \tag{4.13}$$

where L is the number of bins used to partition the orientation range of $(-\pi/2, \pi/2]$, B_l is the lth bin and $p(o_k \in B_l)$ is the occurrence frequency of fingerprints whose local orientations o_k fall in the bin B_l. Note that $p(o_k \in B_l) \log p(o_k \in B_l)$ is set to be zero if $p(o_k \in B_l) = 0$ in Equation (4.13). The weight w_k obtains the minimum of zero if the local orientations o_k of all fingerprints fall in a same bin. It reaches to the maximum of one if o_k are evenly distributed over all bins.

We compute the regional weights with Equation (4.13) using the first fingerprint instances from NIST database-4. All the test fingerprints are aligned by the reference point on the central point and reference direction pointing to south-most. Figure 4.3 (a) and (b) show the regional weights based on the first instances of all 2000 fingerprints from NIST database-4 and 1204 fingerprints selected from NIST database-4 according to the natural distribution, respectively. From Figure 4.3, we can see that the orientation elements below the reference point contain more variant values and thus are more discriminating than those above that. This is expected because the ridge flows in the upper region (i.e., the region above the reference point) have more variant patterns than those in the below region. The center block is bright due to the unstable orientation of the highest curvature and it is excluded from the feature vector.



(a) (b)

Figure 4.3: Gray level representation of the regional weights computed by the entropy estimates on the first fingerprint instances from NIST database-4: (a) all fingerprints and (b) 1204 fingerprints selected according to the natural distribution.

Due to the periodicity and discontinuity of the orientation, we can neither weight the orientation nor its difference directly during the computation of the distance measure in Equation (4.10). Our proposed distance measure with the incorporation of the regional weights is defined by

$$d_W(O^p, O^q) = 1 - \frac{\left|\sum_{k=1}^{M} w_k v_k e^{j2(o_k^p - o_k^q)}\right|}{\sum_{k=1}^{M} w_k v_k}. \tag{4.14}$$

Similar to the property (4.11), we have

$$d_W(O^p + aI, O^q + bI) = d_W(O^p, O^q). \tag{4.15}$$

Let $\alpha_k = w_k v_k$ and $\delta_k^{pq} = 2(o_k^p - o_k^q)$, the proposed distance measure in Equation (4.14) can be computed by

$$d_W(O^p, O^q) = 1 - \frac{\sqrt{\left(\sum_k \alpha_k \cos\delta_k^{pq}\right)^2 + \left(\sum_k \alpha_k \sin\delta_k^{pq}\right)^2}}{\sum_k \alpha_k}. \tag{4.16}$$

4.5 Fingerprint Retrieval

The purpose of fingerprint retrieval is to reduce the search space of the fine matching with a desired accuracy and facilitate the search of database in an AFIS. Given a query fingerprint, retrieval is to select a subset of candidate fingerprints for the fine matching by coarsely searching the database. The average percentage of the retrieved fingerprints from the database over all query fingerprints measures how much the fingerprint retrieval can narrow down the fine search space and thus represents the retrieval efficiency. At a given retrieval efficiency, if one of the retrieved candidates originates from the same finger as the query fingerprint, the retrieval is successful for this query. Otherwise, it is a failure. The retrieval accuracy/error rate is calculated by the percentage of the query fingerprints with retrieval success/failure. The performance of a retrieval technique is thus measured by the retrieval accuracy and efficiency. The retrieval on the continuous classification retrieves fingerprints in the database whose features fall in a neighborhood centered at that of query fingerprint. Its retrieval efficiency and accuracy can be balanced more easily by adjusting the size of the retrieval neighborhood than the traditional exclusive classification. Thus, it usually provides good retrieval performance. Since our retrieval feature set consists of a 1-D dominant ridge distance DRD and a 192-D orientation vector, we investigate the performance of fingerprint retrieval by incorporating these two numerical features on the continuous classification in this chapter.

As the DRD is a scalar feature, we sort the fingerprints in the database according to their DRDs computed with Equations (4.4) and (4.5). In the online query processing of fingerprint retrieval, we first narrow down the search of the database by selecting a subset of fingerprints whose DRDs are within a bin centered at the DRD of the query fingerprint. This is a coarse level retrieval in which the bin width is chosen sufficiently large so that there is almost no error occurring in this stage at a price of low retrieval efficiency. The bin width is set to 2 pixels in our experiments.

In the retrieved subset of database by the DRD, the 192-D orientation vector is then used to search for the candidate fingerprints of the query fingerprint. Fingerprints in the subset are retrieved if their distances (4.16) between their orientation vectors O^p and the query orientation vector O^q are less than a threshold r_n^q, i.e.,

$$d_W(O^p, O^q) < r_n^q, \tag{4.17}$$

where r_n^q represents for the retrieval threshold of query fingerprint q that results in retrieving n fingerprints. The tradeoff between retrieval accuracy and efficiency can be adapted by adjusting the threshold r_n^q. In an AFIS, the retrieval and fine matching can be integrated so that the retrieval threshold increases from a small value until a successful match is found. The threshold can increase by a fixed step or based on a fixed number of new fingerprints retrieved [18, 66]. The incorporation of the fingerprint matching in the retrieval may greatly increase the retrieval performance if a good matching algorithm is applied. As the retrieval performance of this incremental search depends on the matching algorithm, it is not further studied in this work. We will just give the experimental result of retrieval efficiency by assuming a perfect matching algorithm is applied, i.e., the search of database stops once the corresponding database template is found.

Let N be the number of fingerprints in the database. Without applying the matching algorithm, we can only set the retrieval threshold to a constant $r_n^q = c$ (fixed distance) or some value that results in retrieving a constant portion of fingerprints $r_n^q|(n = cN)$ (fixed order) for all query fingerprints q. c is usually determined by a given target of retrieval efficiency or accuracy in practice. Our experiments show that the variable threshold $r_n^q|(n = cN)$ achieves better retrieval performance than the constant one $r_n^q = c$ at a price of longer retrieval time caused by sorting the distances between the query fingerprint and database templates. To avoid the time consuming database sorting, we propose to vary the threshold based on the nearest few fingerprints in the database. If a query fingerprint has smaller distance to its nearest few fingerprints in the database, the possibility that one

of them is the right candidate is higher than the case of larger distance. It is very likely that the former case needs smaller retrieval threshold than the latter. Therefore, we set the retrieval threshold as

$$r_n^q = r_{bN}^q + c, \tag{4.18}$$

where b $(1/N \leq b < 1)$ and c are two constants to all query fingerprint q. b is often set a small value (b is simply fixed to the minimum value $1/N$ in our experiments). Different values of c are used to test the retrieval accuracies at different retrieval efficiencies. Our experiments show that the proposed threshold setting $r_n^q = r_1^q + c$ achieves better retrieval performance than the fixed distance $r_n^q = c$ and surprisingly even better than the fixed order $r_n^q|(n = cN)$.

4.6 Experimental Results

In this section, we apply the proposed fingerprint retrieval algorithm on the well-known data sets and compare it with some existing approaches to demonstrate its performance and advantages. Retrieval efficiency and accuracy/error are two main evaluations of the retrieval performance. In our experiments, these two performances are evaluated to be well scaled to the size of database. The retrieval efficiency is indicated by a "penetration rate", which is the average percentage of database retrieved for the fine matching over all query fingerprints. The penetration rate can be adapted by changing the values of c. It indicates how much the fingerprint retrieval can narrow down the search of database. For a query fingerprint, the retrieval is successful if one of the retrieved candidate fingerprints is from the same finger as the query one. It is more likely to retrieve the correct one if more candidate fingerprints are retrieved from the database. The retrieval error rate is thus calculated by the percentage of the query fingerprints with false retrieval at a given penetration rate. These performance estimates are independent of the size of database used, although the uncertainties in the estimates depend

on the size of the sample pair populations.

4.6.1 Data Sets

The NIST fingerprint special database 4 (NIST database-4) [102] is often used as a benchmark to test the performance of fingerprint retrieval. Most published results of fingerprint classification and indexing are based on this database. It contains 2,000 pairs of fingerprints of size 512×480 pixels with the resolution of 500 dpi. These fingerprints are taken from 2,000 different fingers with two instances per finger and are numbered from $f0001$ to $f2000$ for the first instances and from $s0001$ to $s2000$ for the second instances. To have a comprehensive comparison, we also perform our fingerprint retrieval approach on this database. This fingerprint database is collected for the purpose of testing the exclusive classification so that the five common classes (arch, tented arch, left loop, right loop and whorl) are pre-labelled for each fingerprint and are of the same occurrence frequency. However, the natural distribution of fingerprints among these five classes is significantly different with 3.7% for arch, 2.9% tented arch, 33.8% left loop, 31.7% right loop and 27.9% whorl, respectively. We reduce the number of fingerprints of less frequent classes according to this real distribution and obtain 1204 pairs of fingerprints from NIST database-4. This reduced database, called data set 2, and the original NIST database-4, are both applied in our experiments. The first fingerprint instances form the database templates to be retrieved and the second instances serve as query fingerprints. In addition, our proposed algorithm are also performed on the databases of Fingerprint Verification Competition (FVC) [70] to test the retrieval performance on the online fingerprints captured under less controlled conditions.

4.6.2 Fingerprint Retrieval on NIST Database

In our developed fingerprint retrieval algorithm, a new distance measure is proposed to compare two orientation vectors and a regional feature weighting scheme

is imposed on the proposed distance measure. In addition, the dominant ridge distance is employed as an auxiliary feature for fingerprint retrieval. The first experiment is implemented to test the improvements of fingerprint retrieval performance by these contributions. Data set 2 is applied in this experiment as it resembles the real fingerprint distribution. Figure 4.4 shows the retrieval error rate against the penetration rate by using the Euclidean, Manhattan distance measures, the proposed distance measure in Equation (4.10) and the proposed weighted distance measure in Equation (4.16) as well as the proposed approach adding the dominant ridge distance. While there is little difference between Euclidean and Manhattan distance measures, the proposed distance measure (4.10) consistently improves the retrieval performance. Moreover, the proposed weighted distance measure (4.16) consistently reduces the retrieval error by more than two percents at all penetration rates. Comparing to the fingerprint retrieval by applying the Euclidean and Manhattan distance measures on the orientation vectors alone, the proposed fingerprint retrieval approach significantly reduces the retrieval error by about five percents at all penetration rates.

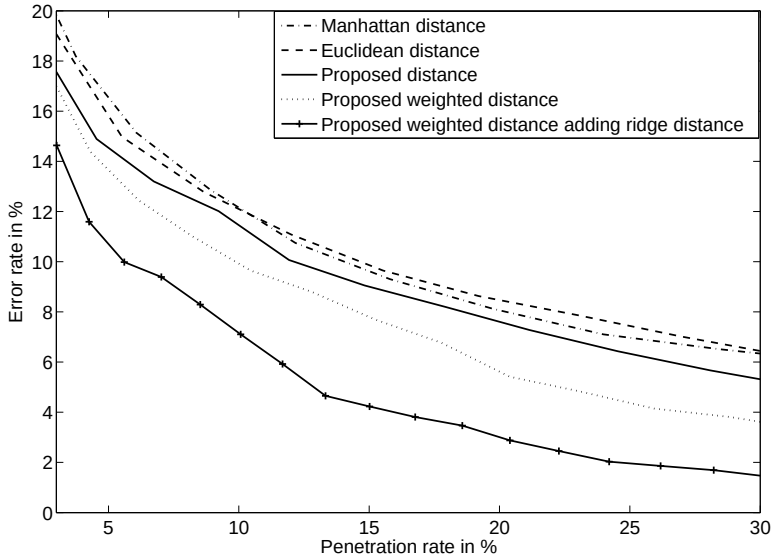


Figure 4.4: Comparison of retrieval results with different distance measures and the effect by adding the dominant ridge distance in the retrieval feature set.

The second experiment tests the retrieval performance with different ways of setting the retrieval threshold r_n^q. The dominant ridge distance is not used in this experiment as the threshold r_n^q is only applied to the retrieval on the orientation vector. Figure 4.5 shows the retrieval error rate against the penetration rate by using the retrieval thresholds of fixed distance ($r_n^q = c$), fixed order ($r_n^q|(n = cN)$) and the proposed retrieval threshold ($r_1^q + c$). Error rates at different penetration rates are computed by using different values of c, which are constant to all query fingerprints. We can see that the retrieval performance of fixed order is better than that of fixed distance. This performance improvement is at a price of longer retrieval time caused by sorting the distances of all template fingerprints to the query fingerprint. In contrast, our proposed threshold setting only needs to find the minimal distance. From Figure 4.5, we can see that the proposed threshold setting $r_n^q = r_1^q + c$ achieves better retrieval performance than the fixed distance $r_n^q = c$ and surprisingly even better than the fixed order $r_n^q|(n = cN)$.

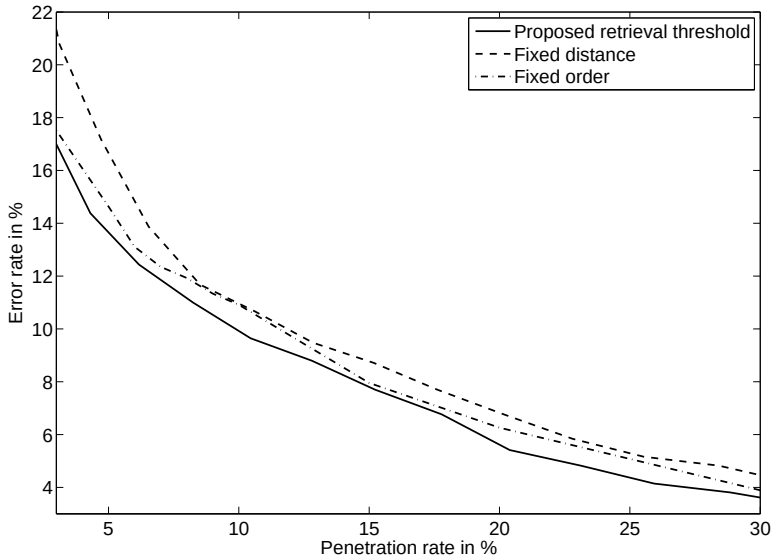


Figure 4.5: Comparison of retrieval results with different threshold setting methods.

The third experiment tests the effect of the database size on the retrieval performance. A model is proposed to predict fingerprint biometrics performance from

a small gallery based on the feature of minutiae triplets [101]. But our features for the fingerprint retrieval are orientation vector and dominant ridge distance. It is difficult to derive theoretically the retrieval performance on a large database from that on a small one. In the absence of a general theory, we test the retrieval performance on different subsets picked randomly from the NIST database-4. Figure 4.6 shows the retrieval error rate against the penetration rate on the databases of five different sizes. We can see no general trend of performance deterioration exist with the increase of database size. We further test the error rates at the penetration rate of 10% for 40 different database sizes (from 50 to 2000). Fig 4.7 shows the retrieval error rate against the database size. The oscillation of the curve at small database sizes indicates that the performance evaluated on small database is unstable. As the database size is larger than 1000, the retrieval performance tends to be stable and no general trend of performance deterioration exists with the increase of database size. This is expected because the performance is estimated by the percentage of fingerprints in the database. These performance estimates are independent of the size of database used, although the uncertainties in the estimates depend on the size of the sample pair populations.

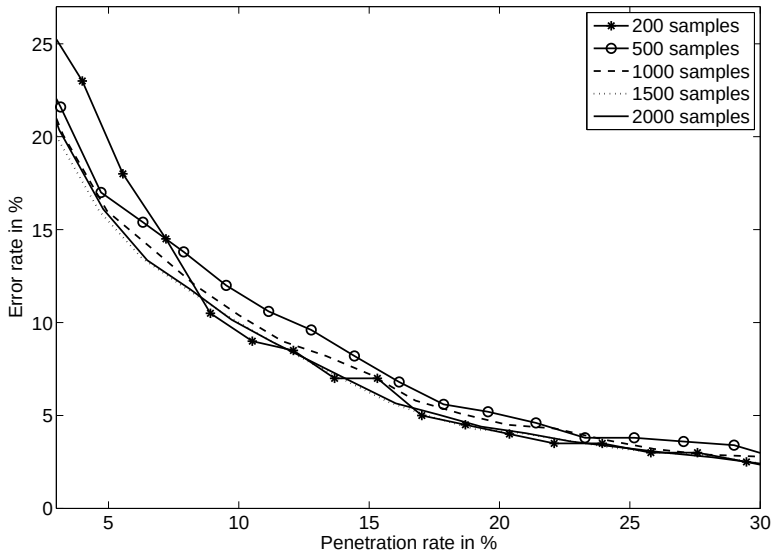


Figure 4.6: Comparison of retrieval results on the databases of different sizes.

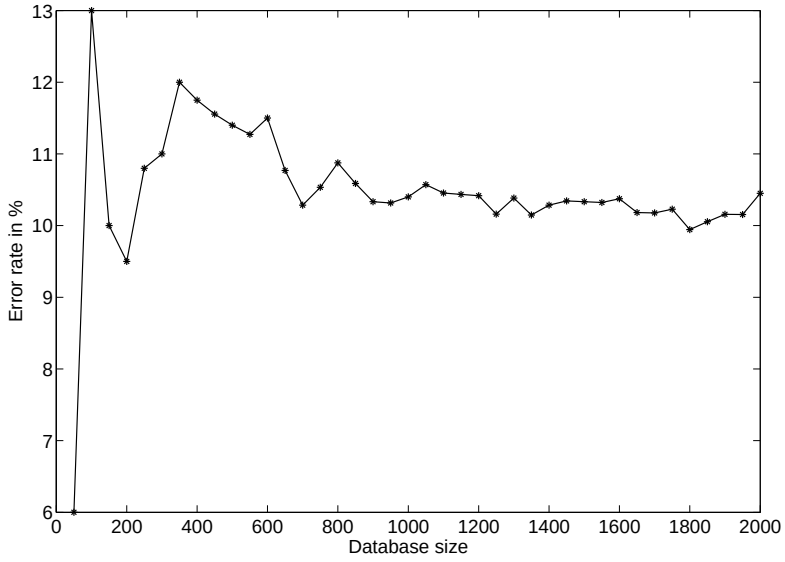


Figure 4.7: The retrieval error rates at the penetration rate of 10% against the number of fingerprints in the database.

4.6.3 Comparison with Other Approaches

The state-of-the-art approaches of continuous classification proposed in the literature [14, 66] are tested on the data set 2. The approach [14] achieves better retrieval performance than the approach [66]. Figure 4.8 compares the retrieval results reported in [14] with those of our proposed approach on the same data set. Consistent performance improvement of our approach is visible over all penetration rates in Figure 4.8 where significant performance enhancement is achieved in the low penetration rate (i.e., high retrieval efficiency).

In addition, the fingerprint retrieval on continuous classification can be incorporated with a fine matching algorithm which is used to halt further retrieval just when the right candidate is retrieved for each query fingerprint. Obviously, this incremental retrieval depends on the fine matching algorithm. An important penetration rate of 5.22% is given in [14] by assuming a perfect matching algorithm is applied for the incremental retrieval, i.e., the search of database stops once the corresponding database template is found. In the implementation, it is to the av-

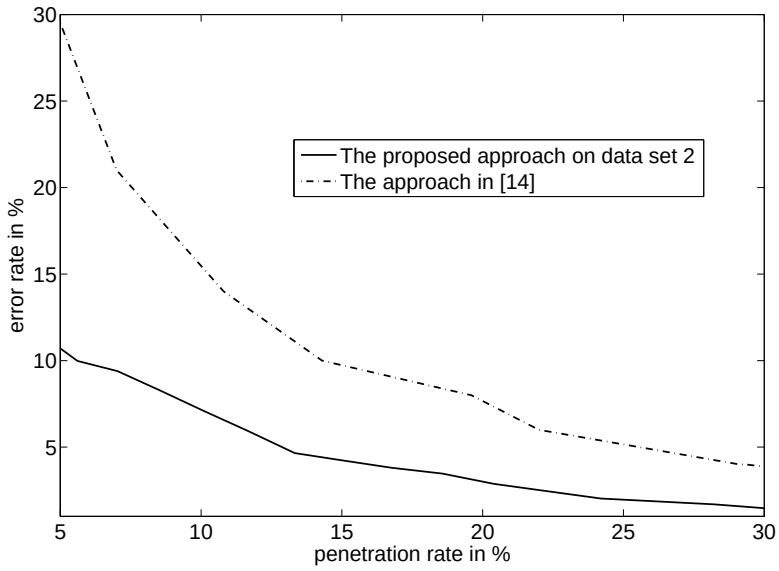


Figure 4.8: Retrieval results of the proposed approach and the approach in [14].

erage order of the corresponding database templates for all query fingerprints after sorting the distances of the database templates to each query fingerprint. Our proposed approach achieves 2.93% for such an indicative penetration rate.

The fingerprint indexing approach proposed in [9] is based on the minutia triplets. The retrieval performance is further improved in [96] by adding two new features. The better performed approach [96] is tested on the second 1000 pairs of fingerprints from the NIST database-4. Figure 4.9 shows the results reported in [96] and our approach on the same data set. The retrieval accuracy of our approach is consistently about two percents better than that in [96] for all penetration rates in Figure 4.9. Moreover, the test database is not of natural distribution, which negatively affects the performance of the proposed approach mainly based on the local ridge orientation feature. The minutia based approaches can be less affected as the minutia features have less correlation with the fingerprint class types than the orientation feature. Therefore, we also plot the retrieval results of our approach on the data set 2 that has natural distribution for a further comparison in Figure 4.9. In addition, the minutia based indexing approaches tend to bring redundant information in the identification system if the minutia feature

is also used in the fine matching algorithm. This will limit its ability to alleviate the accuracy deterioration of an automatic identification system.

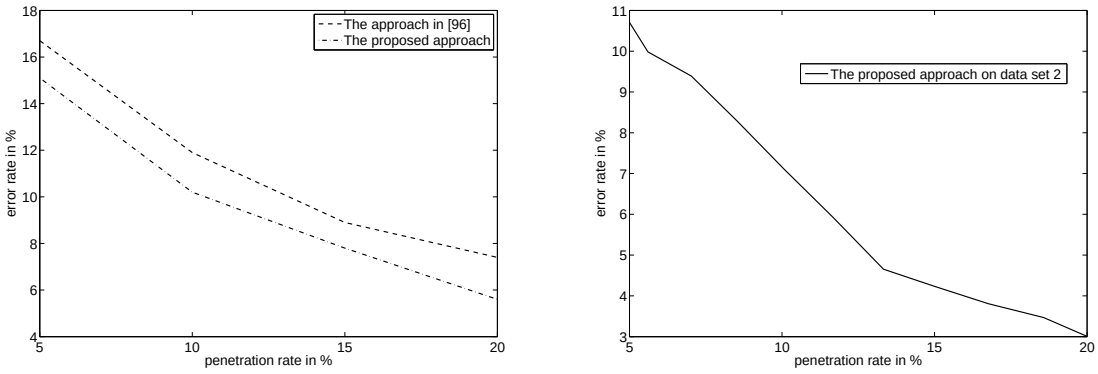


Figure 4.9: Retrieval results in [96] and the proposed approach on the same database as in [96] and on the data set 2.

4.6.4 Fingerprint Retrieval on FVC Database

This experiment tests our proposed approach on the on-line fingerprint databases. Two databases: FVC2000 Db2_a and Db3_a [70], which contain 1600 fingerprints from 200 fingers (8 impressions per finger), are used in this experiment. Figure 4.10 shows the retrieval error rate against the penetration rate of our algorithm performed on these databases. Fingerprints of the Db2_a have higher image quality than those of the Db3_a. At the low penetration rate, successful retrieval needs closer similarity between the query and the template fingerprints, which is more sensitive to the image quality. Therefore, the retrieval performance on Db2_a is better than that on Db3_a at the low penetration rates. However, the retrieval performance on Db2_a is worse than than that on Db3_a at the high penetration rates. Partial fingerprint whose core point is near the image border or out of the image is the culprit. Db2_a contains more such partial fingerprints than Db3_a, which will fail to be retrieved even at the high penetration rates.

We further test the retrieval performance on the combined database that contains all fingerprints from both Db2_a and Db3_a. Figure 4.10 shows that the

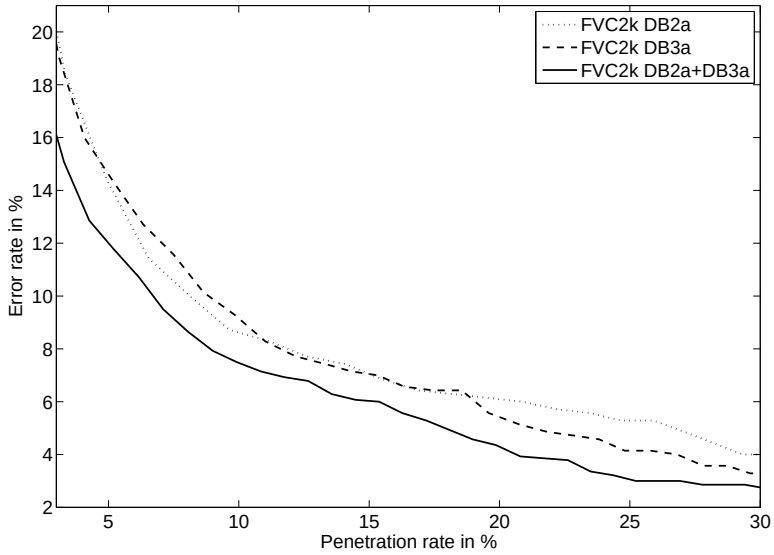


Figure 4.10: Retrieval results on the on-line fingerprint databases: FVC2000 Db2_a and Db3_a.

retrieval performance on the combined database is better than those on the separate Db2 a and Db3_a. This is not a surprise because fingerprints of a same finger are captured by the same sensor but fingerprints from different fingers may originate from the different types of sensors which will enhance the discriminability of the feature set.

4.6.5 Computational Complexity Analysis

The proposed fingerprint retrieval algorithm is implemented by C programming language and is executed under Windows XP Professional O.S. on a Comaq Evo D510CMT (Intel Pentium 4 at 2.26GHZ) PC. To test the computational complexity, we compute the average time for searching the database over all query fingerprints. The average search time for retrieving a subset from the 2000 templates of the NIST database-4 is 0.067 seconds (over 2000 query fingerprints from the NIST database-4). It is indeed a fast search process comparing to the time consumption for matching a query fingerprint with 2000 database templates using the minutiae features.

4.7 Summary

In this chapter, we have developed a fingerprint retrieval algorithm based on the continuous classification that uses the orientation field as the main retrieval feature and the dominant ridge distance as an auxiliary feature. These two coarse level features are not closely correlated with the minutiae features that are often used for the fine matching in the automatic fingerprint verification and identification systems. Consequently, the proposed retrieval approach will not only speed up the identification process but also alleviate the accuracy deterioration of fingerprint identification from that of verification. The introduced auxiliary feature, dominant ridge distance, is more robust to noise than the simple average ridge distance of a fingerprint and brings new information of fingerprint for more effective retrieval. The proposed orientation distance measure, which evaluates the inconsistency of orientation differences, quantifies more effectively the distance between two orientation vectors than the traditional Euclidean and Manhattan distance measures. The proposed regional weighting scheme imposed on the distance measure also visibly improves the retrieval performance. In addition, the suggested retrieval threshold setting slightly improves the retrieval performance comparing to those by the fixed distance and fixed order. We have performed our algorithm on the NIST database-4 and FVC2000 Db2_a and Db3_a. Experimental results and comparisons demonstrate that the proposed retrieval approach outperforms the previous continuous classification and minutia-based indexing approaches in terms of retrieval efficiency v.s. accuracy.

However, the continuous classification only ranks the database templates according to their distances to the query fingerprint while neglecting the similarities among the database templates for the fingerprint retrieval. Therefore, our proposed fingerprint retrieval algorithm suffers from the time consuming exhaustive search of database in the continuous classification. This will limit its application to large fingerprint database. Other indexing techniques which can discover the underlying structure of the fingerprint database such as database clustering will be

explored to facilitate more efficient fingerprint retrieval in the following chapters.

Chapter 5

A Multi-Prototype Clustering Algorithm

5.1 Introduction

Clustering is a crucial technique widely used to discover the underlying structure of a given data set by unsupervisedly grouping the similar patterns. It groups the data set into a number of clusters such that the patterns within cluster are more similar than those from different clusters. Clustering techniques have been widely investigated in the literature [7, 21, 24, 34, 37, 48, 56, 59, 61, 64, 81, 94, 98, 110] and are used in many applications such as data mining, information retrieval, image segmentation and pattern recognition, etc.. Most clustering algorithms can be broadly classified into hierarchical or partitional clustering [43].

Hierarchical clustering is a procedure of transforming the proximity matrix of the data set into a sequence of nested groups in an agglomerative or divisive manner. The agglomerative hierarchical clustering has been widely studied since it allows for more feasible segments to be investigated [34, 37, 56, 59, 94]. The *single-link* [94], *complete-link* [59] and *average-link* [94] algorithms produce a sequence of clusterings based on the rank order of proximities. The *single-link* and *complete-*

link algorithms use the distance between two closest and farthest points from two clusters as the cluster distance, respectively. Dependence on only a few data points to measure the cluster distance makes these algorithms sensitive to noise. The *average-link* algorithm measures the cluster distance with the average distance of all pairs of patterns from different clusters. It is more robust to noise than the *single-link* and *complete-link* algorithms. A CURE algorithm [37] represents each cluster with a certain fixed number of well scattered points and shrinks these points toward the cluster center by a specified fraction. This algorithm achieves an improvement of noise robustness over the *single-link* algorithm through shrinking. A Chameleon algorithm partitions a constructed k-nearest neighbor graph into a number of subclusters followed by dynamically merging the subclusters [56]. A cluster isolation criterion is proposed for the agglomerative hierarchical clustering in [34]. In general, the hierarchical clustering algorithms can provide an easy understanding of the inherent structure of the data set. But they often require high computation cost and large memory space which make them inefficient for large data sets.

Partitional clustering produces a single partition of the data set which aims to optimize a certain cluster criterion function. Some partitional clustering algorithms have been proposed based on different cluster criterions [2, 31, 32, 75, 76, 90, 93]. In fact, each cluster criterion imposes a certain structure on the data set. The model-based clustering algorithms [32, 75] assume that the data distribution of a cluster fit a given probability density model such as Gaussian model. They can discover the hyper-ellipsoidal clusters. But assumption of a static model makes them ineffective to adequately capture the characteristics of individual clusters, especially when the data set contains the clusters of diverse shapes and densities and the model parameters are not properly selected. Some nonparametric clustering algorithms based on density and grid are proposed to identify clusters by searching the regions of high data density separated by sparse valleys [2, 31, 93]. Although these algorithms can find clusters of arbitrary shape, their performances usually degrade for the high dimensional data set. The squared-error clustering algorithm produces

a partition of the data set based on the minimization of squared error [43, 76]. It represents each cluster with the mean vector of the cluster and assign each pattern to the closest cluster iteratively. It assumes the clusters are hyper-spherical and of similar size. This algorithm terminates when the cluster labels do not change. But there is no guarantee to reach the global minimum of the squared error. Some variants of the squared-error clustering algorithm are proposed in the literature [4, 23, 57, 76, 90, 95, 100]. The K-means clustering algorithm [76] assumes the number of clusters is given. It is widely used because of its low computational cost and memory space requirement, but the clustering result is sensitive to the initialization of partition. A good initialization method can improve the performance of K-means clustering algorithm [57]. The Euclidean distance used in the K-means clustering makes it only effective to discover hyper-spherical clusters. Some new distance measures are proposed to detect clusters with specific characteristics [23, 95]. An ISODATA (Iterative Self-Organizing Data Analysis Techniques) clustering algorithm is proposed to automatically adjust the number of clusters by merging similar clusters and splitting clusters with large standard deviation during the iteration [4]. Besides the squared error, other criterions such as the Davies-Bouldin index [28] and cluster variance are imposed as a global criterion to determine the optimum number of clusters [90, 100]. Most partitional clustering algorithms require less memory space and computation cost than the hierarchical clustering algorithms. But their clustering results are usually not as good as those of hierarchical clustering since they often represent each cluster with a single prototype which may not adequately model the complex cluster. Recently, support vector clustering is proposed to generate the cluster boundaries of arbitrary shape by transforming the original space to a high dimensional space with a kernel function [7]. Although this algorithm can solve some difficult clustering problems, it is not easy to choose a suitable kernel parameter and the clustering result cannot provide information about the representation of cluster such as prototypes.

The hybrid clustering algorithms are proposed to combine the merits of partitional and hierarchical clustering algorithms for better data grouping [24, 64, 81,

99,110]. They usually partition the data set into a relatively large number of small subclusters and construct a hierarchical structure for them based on a certain cluster distance (similarity) measure. A given number of clusters can be found on the hierarchical structure. A BIRCH algorithm [110] arranges the patterns of a data set into a number of subclusters represented by *cluster feature* (CF) vectors in a tree structure. The centroid, radius and diameter of subcluster are easily obtained from its CF vector. The representation of each subcluster by a CF vector makes it efficient for very large data sets. The agglomerative method based on a cluster distance measure is often performed on the subclusters to obtain a desired number of clusters [64, 81, 99, 110]. The hybrid clustering algorithms usually work well to discover clusters of arbitrary shape and size. But the clustering results through a single scan of the data set may be sensitive to the number of subclusters and the initial settings of the partition.

Most partitional clustering algorithms represent each cluster with a single prototype such as the centroid and medoid of the cluster. This may not adequately model the clusters of arbitrary shape and size and hence limits the clustering performance on the complex data structure. In this chapter, we develop a clustering algorithm which use multiple prototypes to represent a cluster. The proposed multi-prototype clustering algorithm begins with an initial partitioning of the data set into a relatively large number of small subclusters using the squared-error clustering. Each subcluster is represented by a prototype to locate the region of high density. A separation measure is proposed to evaluate how well two prototypes are separated. Multiple prototypes with small separation are grouped to model a given number of clusters in agglomerative method. After a single scan of the data set, the cluster separating boundary may still reside in the region of high density due to the poor initial settings. New prototypes are iteratively added to improve the poor cluster boundaries until all of them move to the sparse region. The details of our proposed clustering algorithm are presented in the following sections. We perform the proposed algorithm on both synthetic and real data sets and compare it with some existing clustering algorithms to demonstrate its effectiveness.

5.2 Squared-error Clustering Algorithm

The squared-error clustering algorithm produces a partition of the data set based on the minimization of squared error [43, 76]. Let $X = \{X_1, X_2, ..., X_N\}$ where $X_i = [x_{i,1}, x_{i,2}, ..., x_{i,M}] \in \mathbb{R}^M$ be a set of N patterns in the data set and K be the given number of clusters. The cluster prototypes are denoted by a set of vectors $Z = \{Z_1, Z_2, ..., Z_K\}$. The squared error of clustering is computed as:

$$E(U, Z) = \sum_{l=1}^{K} \sum_{i=1}^{N} u_{i,l} d^2(X_i, Z_l), \tag{5.1}$$

subject to

$$\sum_{l=1}^{K} u_{i,l} = 1, 1 \leq i \leq N, \tag{5.2}$$

where $u_{i,l} \in \{0, 1\}$ and $u_{i,l} = 1$ indicates that pattern i belongs to cluster l; $d(X_i, Z_l)$ is the distance between pattern i and the prototype of cluster l and Euclidean distance measure is often used. The squared error can be reduced by iteratively solving the following two minimization problems:

- Fix $Z = \widehat{Z}$ and solve the problem $E(U, \widehat{Z})$ by:

$$u_{i,l} = \begin{cases} 1 & \text{if } d^2(X_i, Z_l) = \min_{t=1}^{K} d^2(X_i, Z_t) \\ 0 & \text{otherwise} \end{cases} \tag{5.3}$$

- Fix $U = \widehat{U}$ and solve the problem $E(\widehat{U}, Z)$ by:

$$Z_l = \frac{\sum_{i=1}^{N} u_{i,l} X_i}{\sum_{i=1}^{N} u_{i,l}}, \text{for } 1 \leq l \leq K \tag{5.4}$$

The processing steps of the squared-error clustering algorithm can be summarized as follows [43]:

(1) Initialize the K cluster prototypes;

(2) For each pattern, compute its distance to the K prototypes and assign it to the closest cluster using Equation (5.3);

(3) Compute the new prototype of each cluster with Equation (5.4);

(4) Go to step (2) until the cluster labels do not change between two successive iterations.

The squared-error clustering algorithm has been widely used in data partitioning because of its high computational efficiency, low memory space and yet good performance in finding the regions of high density. It tends to work well with the hyper-spherical clusters of similar size. However, two basic problems exist in this algorithm. One is that the clustering result is sensitive to the initialization of prototypes since there is no guarantee to reach the global minimum of the squared error. Another problem is that using single prototype to represent each cluster may not adequately model the clusters of arbitrary shape and size even if the cluster prototypes are properly initialized. In this algorithm, the separating boundary for each pair of clusters is the hyperplane through the midpoint of the cluster prototypes and perpendicular to the line connecting these prototypes. Figure 5.1 shows the separating hyperplane (HP) between cluster C_q and C_l. Thus, the cluster separating boundary may reside in the region of high density between two clusters of complex structure. For example, the separating hyperplanes reside in the high-density region between two clusters of different size in Figure 5.2 (a) and two clusters of arbitrary shapes in Figure 5.2 (c). We will propose a clustering algorithm which can use multiple prototypes to model the clusters of complex structure.

5.3 The Multi-prototype Clustering Algorithm

In this section, we first propose a separation measure to evaluate how well two cluster prototypes are separated. Next, we present the proposed multi-prototype

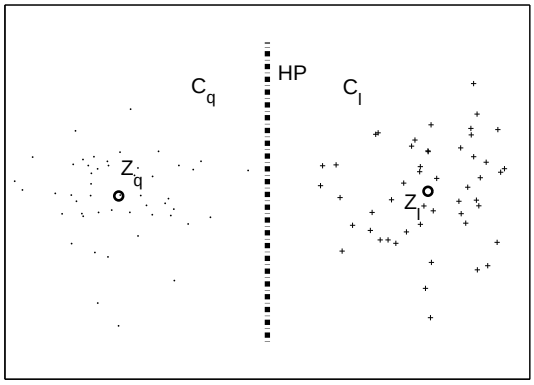


Figure 5.1: The separating hyperplane of cluster C_q and C_l represented by prototype Z_q and Z_l, respectively, in squared-error clustering.

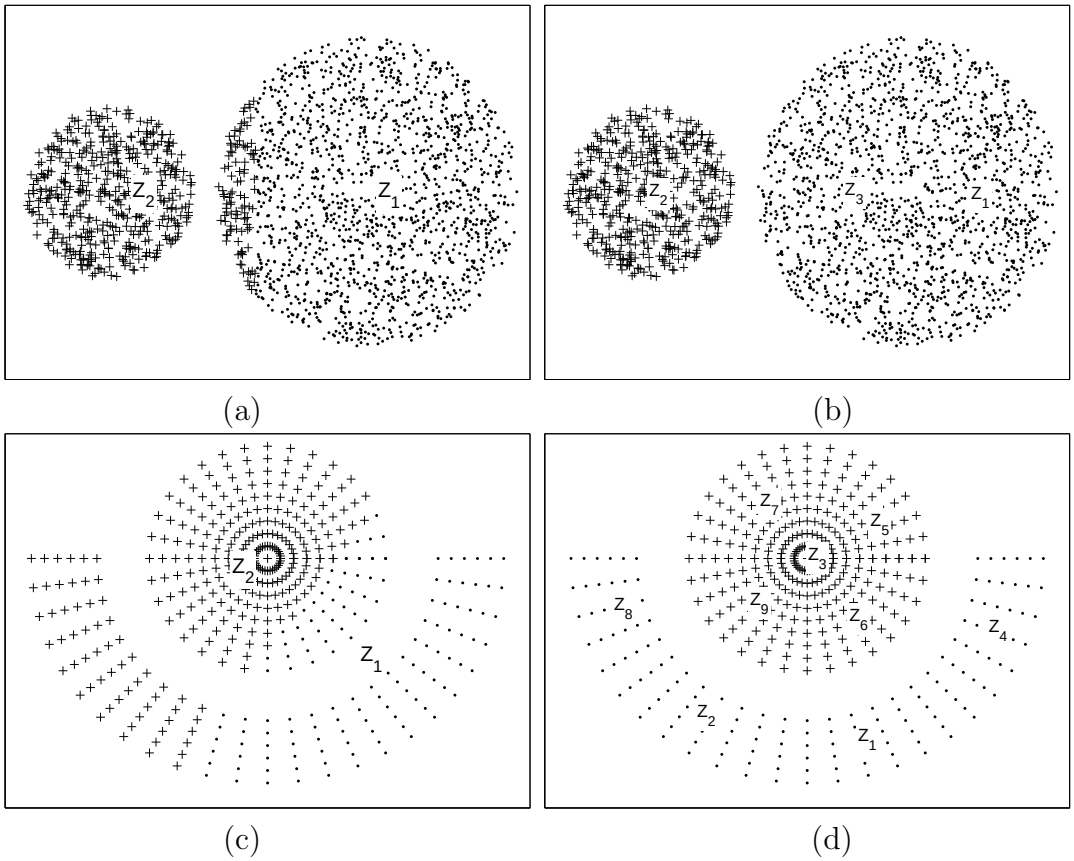


Figure 5.2: Clustered data represented by different marks and the prototypes denoted by text 'Z_l': (a) poor result with two clusters of different size modelled by 'Z_1' and 'Z_2', (b) good result with the small cluster modelled by 'Z_2' and the large cluster modelled by 'Z_1' and 'Z_3', (c) poor result with two clusters of arbitrary shape modelled by 'Z_1' and 'Z_2' and (d) good result with one cluster modelled by 'Z_1,Z_2,Z_4,Z_8' and another cluster modelled by 'Z_3,Z_5,Z_6,Z_7,Z_9'.

clustering algorithm based on the separation measure. Finally, the complexity analysis of the proposed clustering algorithm is provided.

5.3.1 Separation Measure

The separation of two clusters measures how well the clusters are separated. Conceptually, large separation of two clusters indicates less inclination of integrating these clusters into a larger one. It is also called cluster distance or similarity in the literature [37, 56, 59, 94]. The distances between the closest or farthest data points of two clusters are used to measure the cluster separation in the agglomerative clustering algorithms [59, 94]. They are not only computation expensive but also sensitive to noise due to the dependence on a few points. In the prototype-based clustering algorithms, the separation of two clusters (or prototypes) is often measured using the distance between their prototypes. Although this measure is computationally efficient and robust to noise, it cannot distinguish the clusters of different sizes and shapes. For example, four pairs of clusters in Figure 5.3 have equal prototype distances, but their separations are obviously different. A measure of within-to-between cluster spread $R_{q,l} = \frac{e(Z_q)+e(Z_l)}{d(Z_q,Z_l)}$, where $e(Z_l)$ is the within-cluster variance and $d(Z_q, Z_l)$ is the distance between the prototype Z_q and Z_l, is introduced to evaluate the cluster separation [28]. Including the information of within-cluster variance solves the problem of cluster size but leave the cluster shape problem unsolved. For example, the R of two clusters in Figure 5.3 (d) is 0.2931 which is larger than that in Figure 5.3 (c), 0.2455. But two clusters in Figure 5.3 (d) are obviously better separated than those in Figure 5.3 (c). A similarity measure of two clusters is proposed by assuming that the data distribution of each cluster follows a static model [64]. Although this measure is effective in some cases, it may not adapt to the internal characteristics of the clusters especially when the data set contains the clusters of diverse shapes and distributions.

The cluster prototypes produced by the squared-error clustering are often located in the high-density regions. Two prototypes connected by a region of high

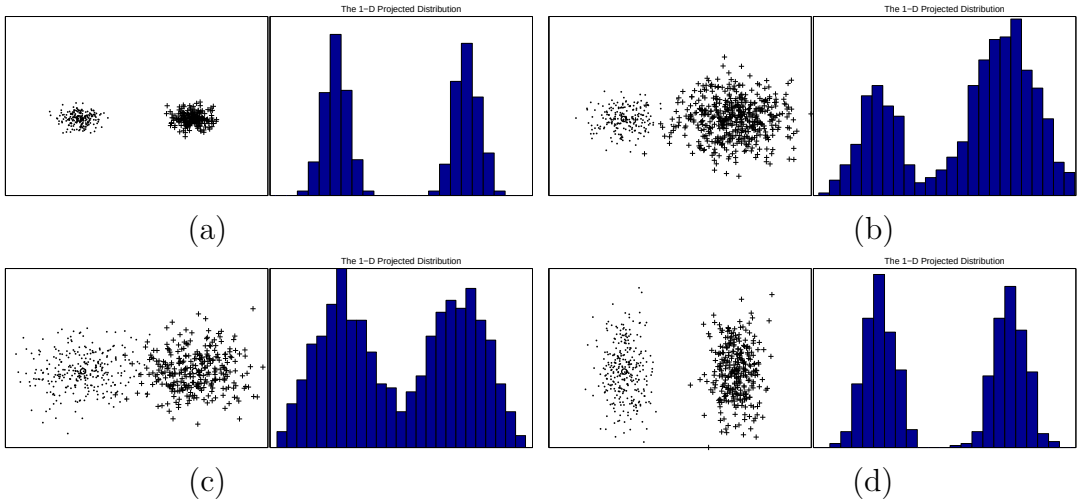


Figure 5.3: Four pairs of clusters and their 1-D projected distributions. Their separations are: (a) *sp*=1 (b) *sp*=0.8219 (c) *sp*=0.6372 and (d) *sp*=0.9930.

density are more possible to belong to one cluster than those connected by a sparse region. We propose a separation measure based on the data distribution between two cluster prototypes, which evaluates how well the prototypes are separated by a sparse region. Firstly, two cluster prototypes are connected by a line segment. The data points of two clusters are projected onto the line connecting the prototypes since the separating hyperplane is perpendicular to it. Let Z_q and Z_l denote the prototypes of cluster C^q and C^l (column vectors), respectively. The projections of the data points are computed by:

$$x' = \frac{(X - m_0)^T (Z_q - Z_l)}{\|Z_q - Z_l\|^2}, X \in C^q \cup C^l \tag{5.5}$$

where $m_0 = \frac{Z_q + Z_l}{2}$ is corresponding to the origin of the projections x'. Two cluster prototypes are projected at the positions $-\frac{1}{2}$ and $\frac{1}{2}$ of the line.

Subsequently, we compute the distribution of the projections between the prototypes. For simplicity, we use the histogram to compute the 1-D projected data distribution. The bin center of the histogram is confined in the range of $[-1\ 1]$. To obtain the data densities at the prototypes and origin, the positions -1, $-\frac{1}{2}$, 0, $\frac{1}{2}$ and 1 are specified as the bin centers. $4B + 1 (B \geq 1)$ bins of equal size (i.e.,

$\frac{1}{2B}$) are formed and the number of projections x' falling in each bin is counted to produce the histogram. Figure 5.3 shows some examples of the 1-D projected distribution where B is set to 6. Let $f(c)$ be the data density at the position c. The data distribution between the prototypes is denoted by the $2B+1$ densities $\{f(-\frac{1}{2}), f(-\frac{1}{2}+\frac{1}{2B}), ..., f(\frac{1}{2})\}$. The smoothness of the data distribution depends on the bin size. To obtain a smooth distribution, Gaussian filter is repetitively applied on these data densities until only one local minimum exists on them.

Finally, the separation is computed based on the projected data distribution between the prototypes. If the minimum of the $2B+1$ densities between two prototypes is large, the prototypes are connected with a relatively high density region and hence are inclined to belong to one cluster. Based on the minimum density normalized by the average of those at two prototypes, the separation is computed by:

$$sp_{q,l} = 1 - \frac{2\min_{k=1}^{2B+1} f(-\frac{1}{2}+\frac{k-1}{2B})}{f(-\frac{1}{2})+f(\frac{1}{2})} \tag{5.6}$$

Large $sp_{q,l}$ indicates that cluster C^l and C^q or their prototypes are well separated by a sparse region. Some examples of the separations between two clusters are shown in Figure 5.3. Instead of assuming a static distribution model, the data distribution is automatically estimated in this separation measure which can adapt to the internal characteristics of individual clusters.

The separation measure in Equation (5.6) is based on the minimum data density so that the cluster separating boundary is assumed at the hyperplane through the most sparse region between two prototypes. However, two clusters are separated by the hyperplane through the midpoint of the prototypes in practice. By replacing the minimum density with the density at the midpoint of two prototypes in Equation (5.6), we compute the separation of two prototypes based on the current

separating hyperplane as:

$$sp^0_{q,l} = 1 - \frac{2f(0)}{f(-\frac{1}{2}) + f(\frac{1}{2})} \tag{5.7}$$

If $sp^0_{q,l}$ is small, the separating hyperplane resides in the high-density region between two prototypes. Obviously, $sp^0_{q,l} \leq sp_{q,l}$.

5.3.2 The Proposed Clustering Algorithm

For a given data set, the natural clusters often exist in the continuous regions of relatively high density separated by the sparse areas in the pattern space. Using single prototype to represent each cluster often result in the cluster boundaries residing in the region of high density (see Figure 5.2 (a) and (c)). By adding one or more prototypes to model the clusters, the cluster boundary can move to the sparse region of the pattern space (see Figure 5.2 (b)) or a more complex boundary can be constructed to separate the complex clusters (see Figure 5.2 (d)). We propose a clustering algorithm which groups multiple prototypes coexisting within a continuous region of relatively high density into one cluster and adds new prototypes iteratively to improve the poor cluster boundaries.

Firstly, we partition the data set into a relatively large number of small subclusters with each one represented by a prototype. Let P ($K \leq P < N$) be the number of prototypes for the K clusters. The squared-error clustering has good performance in finding the regions of high density with high computational efficiency and low memory space usage. It is thus employed in this stage to produce P prototypes. However, the P prototypes may not be appropriately distributed in the high-density regions. Some prototypes may represent the large subclusters consisting of the patterns from different clusters which can form a connection between two natural clusters. In addition, some prototypes may reside in the outliers which do not belong to any cluster. Thus, we try to add prototypes in the large subclusters and remove the noise prototypes. For each subcluster l, we compute

the within-cluster squared error E_l and the number of patterns N_l. The large subcluster usually has both large N_l and E_l while the noise subcluster often has small N_l. We remove the noise prototypes with $N_l < N_{min}$ ($N_{min} = 0.3\frac{N}{P}$ in our experiments) and add a new prototype in the large subcluster with $N_l > 2.5N_{min}$ and $E_l > E_{max}$ ($E_{max} = \frac{\sum_{k=1}^{P} E_k}{P} + \eta std(E_k), \eta > 0$). The squared-error clustering is repeated until no prototypes are added or removed.

Next, multiple prototypes coexisting within a continuous region of relatively high density are grouped to model the cluster based on the separation measure in Equation (5.6). We compute the separation between each pair of prototypes and form a separation matrix of $P \times P$. If $sp_{q,l}$ is small, prototype q and l coexist in a high-density region and can be grouped into one cluster. The separation of two multi-prototype clusters C^m and C^n is defined as the smallest separation of two prototypes from different clusters:

$$csp_{m,n} = \min_{Z_l \in C^m, Z_q \in C^n} sp_{l,q} \tag{5.8}$$

In cluster organization, each prototype forms a cluster initially and two clusters with the smallest separation are iteratively grouped until K clusters are obtained. This process is similar to the agglomerative *single-link* clustering [94]. The P prototypes are finally organized to represent K clusters in the agglomerative method. Thus, the separating boundary of two clusters is composed of multiple hyperplanes determined by the pairs of prototypes from different clusters. For example, the separating boundary of two clusters in Figure 5.2 (d) is composed of five hyperplanes determined by the pairs of prototypes: $\{Z_7,Z_8\}$, $\{Z_9,Z_8\}$, $\{Z_9,Z_2\}$, $\{Z_6,Z_1\}$, $\{Z_6,Z_4\}$ and $\{Z_5,Z_4\}$. By grouping multiple prototypes to represent a cluster, complex boundaries can be obtained to separate the non-linearly separable clusters.

The last step is to improve the poor cluster boundaries. The cluster boundary may not reside in the sparse region between two clusters due to the poor initial settings. Adding new prototype can push the poor cluster boundary move to the sparse region or construct a more complex boundary to separate the clusters. The

$sp^0_{q,l}$ in Equation (5.7) is used to check the separation of cluster hyperplane. If $sp^0_{q,l} < T$, the cluster separating hyperplane is poor. We compute $sp^0_{q,l}$ for each pair of prototypes from different clusters and sort the poor ones ($sp^0_{q,l} < T$) in increasing order. For each pair of prototypes sorted by $sp^0_{q,l}$, if $N_q + N_l > 3N_{min}$ and the separating boundary of the clusters the prototypes belong to has no new prototype already been added to, a new prototype is added to improve the poor cluster boundary. The new prototype is computed as the mean vector of the patterns whose projections x' locate in $[-\frac{1}{4}, \frac{1}{4}]$.

After adding new prototypes, the multi-prototype clustering algorithm repeats the above steps until the prototypes do not change. Each pattern in the data set belongs to the cluster consisting of the closest prototype. The processing steps of the proposed algorithm can be summarized as:

(1) Initially set $P \geq K$ and randomly choose the P cluster prototypes from the data set;

(2) Apply the squared-error clustering on the data set to obtain P subclusters with each one represented by a prototype;

(3) Remove the prototypes of subclusters with $N_l < N_{min}$ and decrease P accordingly. Add a new prototype in the large subclusters with $N_l > 2.5N_{min}$ and $E_l > E_{max}$ and increase P accordingly. If there are prototypes removed or added, go to step (2);

(4) Calculate the separations between each pair of prototypes with Equation (5.6) and produce a separation matrix;

(5) Organize the P prototypes into K clusters based on the separation matrix. Two clusters with the smallest separation are iteratively grouped into one cluster until K clusters are obtained;

(6) Compute the separation $sp^0_{q,l}$ between each pair of prototypes from different clusters with Equation (5.7) and sort the poor ones ($sp^0_{q,l} < T$) in increasing

order;

(7) For each pair of prototypes sorted by $sp^0_{q,l}$, if $N_q + N_l > 3N_{min}$ and the separating boundary of the clusters the prototypes belong to has no new prototype already been added to, add a new prototype between these two prototypes and increase P accordingly. The new prototype is computed as the mean vector of the patterns whose projections x' locate in $[-\frac{1}{4}\ \frac{1}{4}]$;

(8) Go to step (2) until the prototypes do not change;

(9) Output the clustering result.

5.3.3 Complexity Analysis

Let m be the number of iterations in the squared-error clustering. The time complexity is $O(NPm)$ for partitioning the data set to produce P prototypes. It is $O(P^2 logP)$ for the prototype organization which is similar to the *single-link* algorithm. Thus, the time complexity of the proposed clustering algorithm is $O(NPm + P^2 logP)$. The space complexity is $O(N)$ for the partitioning of the data set in the squared-error clustering. In the prototype organization, the space complexity is $O(P^2)$ because a separation matrix of size $P \times P$ has to be stored. Thus, the space complexity of the proposed clustering algorithm is $O(N+P^2)$. The number of prototypes P is much smaller than N. Thus, the proposed clustering algorithm requires less memory space and computation cost than the commonly used hierarchical clustering algorithms such as *Single-link* and *Complete-link* while preserves much of the speed and efficiency of the squared-error clustering algorithm.

5.4 Experimental Results

The proposed multi-prototype clustering algorithm can be applied for any numerical data set. We conduct a series of experiments on both synthetic and real data

sets to demonstrate the clustering performance of the proposed algorithm. The results are compared with those of some existing clustering algorithms.

5.4.1 Synthetic Data Sets

Clusters on two dimensional (2D) data sets are easy to visualize and compare. The proposed clustering algorithm is tested on five synthetic 2D data sets. For good visualization of the clustering results, we represent the clustered data by different marks and denote the prototypes with texts (e.g., 'Z_l' denotes prototype l).

To illustrate the processing steps of the proposed algorithm, we first consider the data set shown in Figure 5.2 (c) which consists of two clusters of arbitrary shape and size. The clustering result with $P = K$ ($K = 2$) is shown in Figure 5.2 (c). K prototypes cannot adequately model the clusters. Our clustering algorithm initializes P as 5 and set T to 0.8. Since only two clusters exist in the data set, a new prototype is added in each iteration. Figure 5.4 (a) shows an initial clustering state of 5 prototypes produced by the squared-error clustering. The separation $sp_{q,l}$ between each pair of the 5 prototypes is computed and Table 5.1 shows the separation matrix. In cluster organization, prototype Z_1 and Z_2 with the smallest separation are first grouped into one cluster $\{Z_1, Z_2\}$. The second smallest separation is $sp_{3,4}$ so that Z_3 and Z_4 are organized into another cluster $\{Z_3, Z_4\}$. The left prototype Z_1 are organized into the same cluster as Z_5, i.e., $\{Z_1, Z_2, Z_5\}$, since $sp_{1,5}$ is the third smallest separation. The cluster boundary is composed of the hyperplanes separating four pairs of prototypes, $\{Z_3,Z_2\}$, $\{Z_3,Z_1\}$, $\{Z_4,Z_1\}$ and $\{Z_4,Z_5\}$. We compute the $sp^0_{q,l}$ between these pairs of prototypes. A new prototype is added between Z_1 and Z_4 since the $sp^0_{1,4} = 0.4253$ is the smallest among them. If the above steps are repeated, an intermediate clustering state of 8 prototypes is obtained as shown in Figure 5.4 (b). Since the smallest separation $sp^0_{q,l}$ between two clusters is $sp^0_{4,5} = 0.7460 < T$, a new prototype is added between Z_4 and Z_5. The clustering algorithm terminates at $P = 10$ when the smallest separation between two clusters $sp^0_{4,10} = 0.8400 > T$. Figure 5.4 (c) shows the

final clustering result of 10 prototypes. We can see that two clusters are correctly discovered by using 6 prototypes to model the circle cluster and 4 prototypes to model the other cluster.

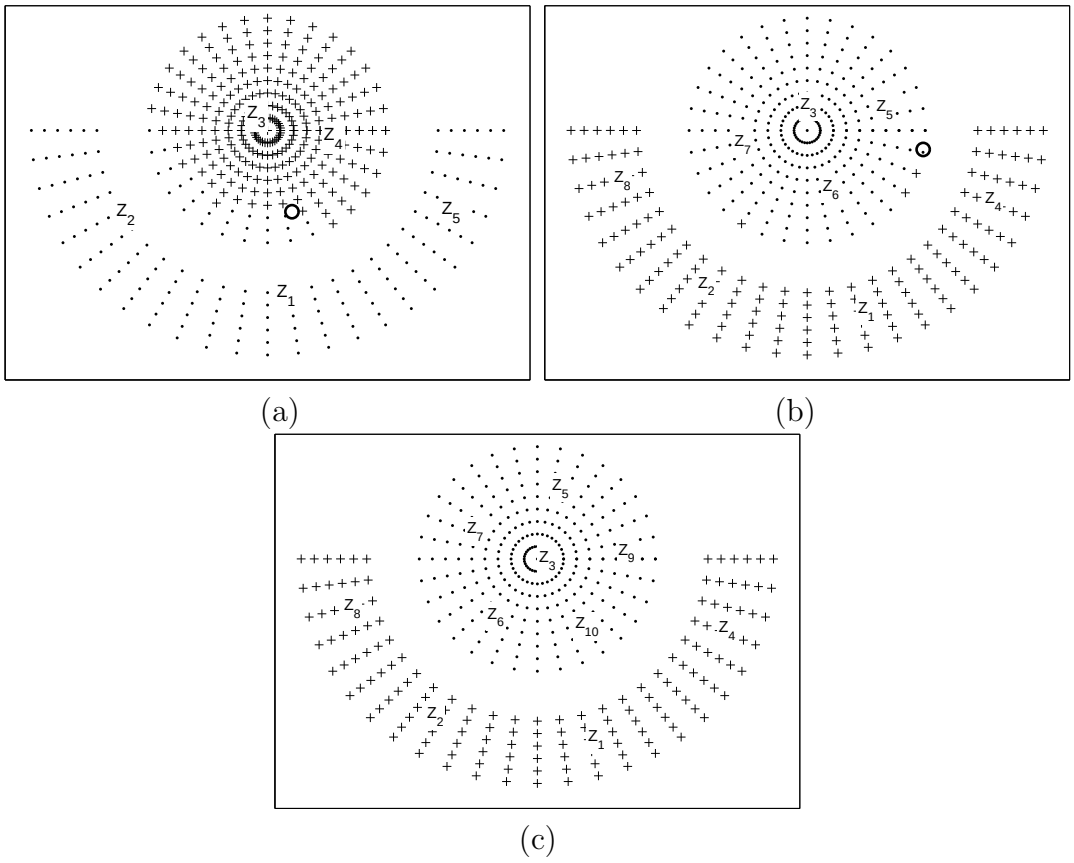


Figure 5.4: An illustrative example of our clustering algorithm where '○' denotes the location of new added prototype: (a) initial clustering state of 5 prototypes, (b) an intermediate clustering state of 8 prototypes and (c) final clustering result of 10 prototypes.

Table 5.1: The separation matrix of 5 prototypes in Figure 5.4a.

sp	Z_1	Z_2	Z_3	Z_4	Z_5
Z_1	-	0.0330	0.8388	0.5747	0.3474
Z_2	0.0330	-	0.8426	0.9008	1.0000
Z_3	0.8388	0.8426	-	0.2663	1.0000
Z_4	0.5747	0.9008	0.2663	-	0.8579
Z_5	0.3474	1.0000	1.0000	0.8579	-

In addition, we perform the proposed clustering algorithm on the four 2D data

sets which are often used to test the clustering algorithms [31, 37, 56, 64, 110]. Figure 5.5 shows the four 2D data sets denoted as DB1, DB2, DB3 and DB4, respectively. DB1 is obtained from [37]. It contains one big and two small circles and two ellipsoids connected by a chain of outliers. The other three data sets DB2, DB3 and DB4 are obtained from [56]. These four data sets with 8,000 to 10,000 data points consist of the clusters of complex shape and arbitrary size and the outliers are scattered on the data sets, which represent some difficult clustering instances.

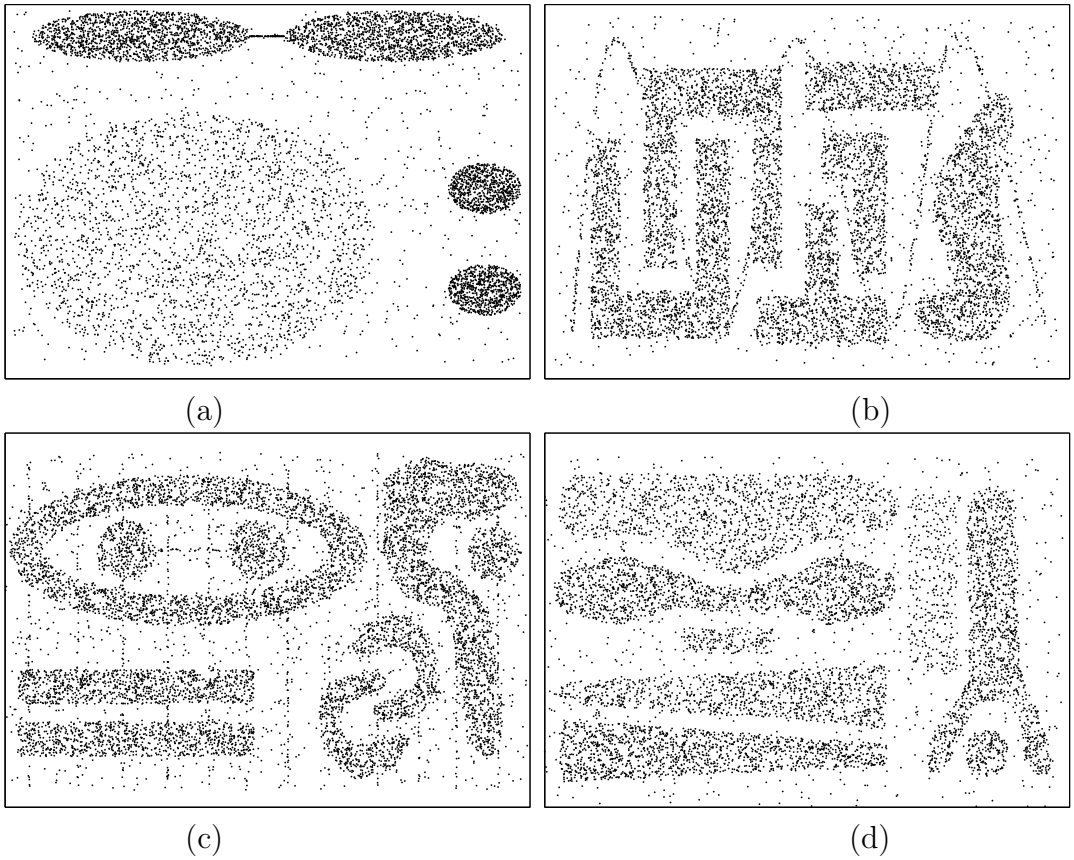


Figure 5.5: Four 2D data sets used in our experiments: (a) DB1 with 8,000 data points, (b) DB2 with 8,000 data points, (c) DB3 with 10,000 data points and (d) DB4 with 8,000 data points.

Figure 5.6 shows the clustering results of the proposed multi-prototype clustering algorithm on these 2D synthetic data sets. The total number of prototypes finally obtained to model the clusters are 11, 29, 38 and 59 for the DB1, DB2, DB3 and DB4, respectively. In addition, the clustering results with the grouped

prototypes for the clusters are illustrated in Table 5.2. From these results, we can see that the proposed clustering algorithm successfully discovers the clusters on these data sets by using multiple prototypes to model the complex clusters.

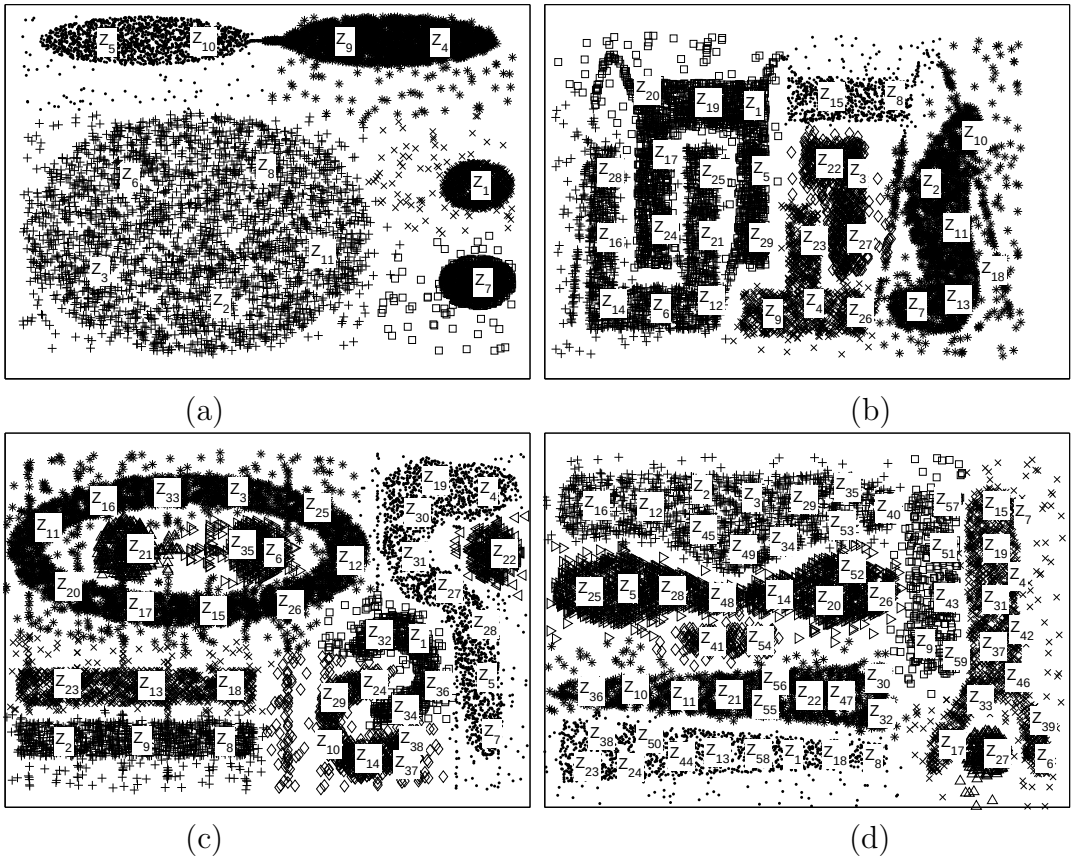


Figure 5.6: The clustering results of our proposed algorithm on the four 2D data sets: (a) DB1 with 11 prototypes, (b) DB2 with 29 prototypes, (c) DB3 with 38 prototypes and (d) DB4 with 59 prototypes, where the clustered data are represented by different marks and the prototypes are denoted with texts.

To show the robustness to initial settings, we perform the proposed clustering algorithm on a poor initialization. DB1 and DB3 are used in this experiment. The initial number of prototypes is set to 20 for both data sets. The separation threshold T is set to 0.45. Figure 5.7 (a) and (c) show the initial clustering results of 19 prototypes on the DB1 and 20 prototypes on the DB3, respectively. DB3 has more complex data structure than DB1 so that the initial 20 prototypes cannot adequately model the clusters on DB3. Although 20 prototypes are enough to model the clusters on DB1, the inappropriate initial prototypes also result in

Table 5.2: The clustering results with grouped prototypes in Figure 5.6.

	# clusters	Grouped prototypes
DB1	5	$\{Z_5,Z_{10}\};\{Z_4,Z_9\};\{Z_2,Z_3,Z_6,Z_8,Z_{11}\};\{Z_1\};\{Z_7\}$
DB2	6	$\{Z_6,Z_{12},Z_{14},Z_{16},Z_{21},Z_{25},Z_{28}\};\{Z_1,Z_5,Z_{17},Z_{19},Z_{20},Z_{24},Z_{29}\};$ $\{Z_8,Z_{15}\};\{Z_3,Z_{22},Z_{27}\};\{Z_4,Z_9,Z_{23},Z_{26}\};\{Z_2,Z_7,Z_{10},Z_{11},Z_{13},Z_{18}\}$
DB3	9	$\{Z_3,Z_{11},Z_{12},Z_{15},Z_{16},Z_{17},Z_{20},Z_{25},Z_{26},Z_{33}\};\ \{Z_{21}\};\ \{Z_6,Z_{35}\};$ $\{Z_{13},Z_{18},Z_{23}\};\{Z_2,Z_8,Z_9\};\{Z_4,Z_5,Z_7,Z_{19},Z_{27},Z_{28},Z_{30},Z_{31}\};$ $\{Z_{22}\};\{Z_1,Z_{32},Z_{34},Z_{36}\};\{Z_{10},Z_{14},Z_{24},Z_{29},Z_{37},Z_{38}\}$
DB4	8	$\{Z_2,Z_3,Z_{12},Z_{16},Z_{29},Z_{34},Z_{35},Z_{40},Z_{45},Z_{49},Z_{53}\};$ $\{Z_5,Z_{14},Z_{20},Z_{25},Z_{26},Z_{28},Z_{48},Z_{52}\};\ \{Z_{41},Z_{54}\};$ $\{Z_{10},Z_{11},Z_{21},Z_{22},Z_{30},Z_{32},Z_{36},Z_{47},Z_{55},Z_{56}\};$ $\{Z_1,Z_8,Z_{13},Z_{18},Z_{23},Z_{24},Z_{38},Z_{44},Z_{50},Z_{58}\};\{Z_9,Z_{43},Z_{51},Z_{57},Z_{59}\};$ $\{Z_4,Z_6,Z_7,Z_{15},Z_{17},Z_{19},Z_{31},Z_{33},Z_{37},Z_{39},Z_{42},Z_{46}\};\{Z_{27}\}$

poor clustering result where two small circles on the right are modelled into one cluster by a prototype. Our proposed algorithm iteratively adds new prototypes to improve the poor cluster boundaries resulted by the inappropriate initial settings. Figure 5.7 (b) and (d) show the final clustering results of 21 prototypes on DB1 and 47 prototypes on DB3, respectively. We can see the clusters on these data sets are successfully discovered. Table 5.3 further illustrates the clustering results with the grouped prototypes for the clusters.

Table 5.3: The clustering results with grouped prototypes in Figure 5.7.

	# clusters	Grouped prototypes
(a)	5	$\{Z_4,Z_8,Z_{12},Z_{13},Z_{14},Z_{18}\};\{Z_3,Z_5,Z_{11},Z_{15}\};$ $\{Z_1,Z_2,Z_6,Z_7,Z_{10},Z_{16},Z_{17},Z_{19}\};\{Z_9\}$
(b)	5	$\{Z_4,Z_8,Z_{12},Z_{13},Z_{14},Z_{18}\};\{Z_3,Z_{11},Z_{15},Z_{20},Z_{21}\};$ $\{Z_1,Z_2,Z_6,Z_7,Z_{10},Z_{16},Z_{17},Z_{19}\};\{Z_5\};\{Z_9\}$
(c)	9	$\{Z_6\};\{Z_3\};\{Z_{10}\};\{Z_8\};\{Z_4,Z_{11},Z_{13},Z_{14},Z_{16},Z_{18},Z_{19}\};$ $\{Z_7,Z_{17},Z_{20}\};\{Z_2,Z_5,Z_9\};\{Z_{12},Z_{15}\};\{Z_1\}$
(d)	9	$\{Z_3,Z_6,Z_7,Z_{10},Z_{16},Z_{17},Z_{20},Z_{22},Z_{25},Z_{35},Z_{37},Z_{42}\};\{Z_{21},Z_{45}\};$ $\{Z_8,Z_{31}\};\{Z_9,Z_{27},Z_{33},Z_{41}\};\{Z_2,Z_5,Z_{29},Z_{39},Z_{47}\};$ $\{Z_{12},Z_{23},Z_{24},Z_{26},Z_{44}\};\{Z_{15},Z_{30},Z_{34},Z_{36},Z_{40}\};$ $\{Z_4,Z_{11},Z_{13},Z_{14},Z_{16},Z_{18},Z_{19},Z_{28},Z_{32},Z_{38},Z_{43}\};\{Z_1,Z_{46}\}$

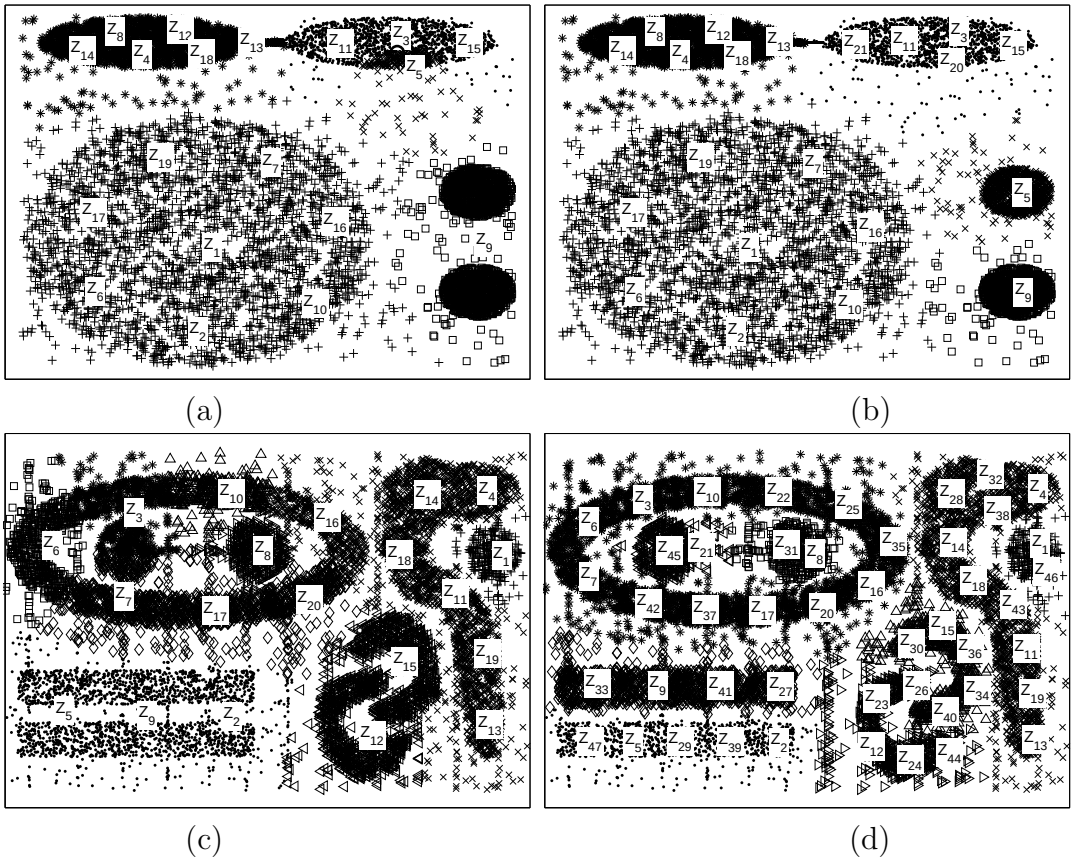


Figure 5.7: The clustering results of the proposed algorithm on poor initialization (a) the initial partition of DB1 and (b) final result of 21 prototypes on DB1; (c) the initial partition of DB3 and (d) final result of 47 prototypes on DB3.

We compare the proposed clustering algorithm with other clustering algorithms on the four 2D data sets. As these data sets contain the clusters of arbitrary shapes and sizes, most variants of the squared-error clustering algorithm such as K-means [76] and ISODATA [4], which use single prototype to represent each cluster, cannot correctly discover the clusters on the data sets. DB1 is used to test the CURE clustering algorithm [37]. As stated in [37], the BIRCH algorithm [110] cannot distinguish between the big and small clusters and the *single-link* algorithm cannot handle the chain of outliers connecting two ellipsoids. The hybrid clustering algorithm [64] is also tested on the four data sets and performs better than some existing clustering algorithms such as the *Single-link* [94], *Complete-link* [59], CURE [37], K-means [76], BIRCH [110] and DBScan [31] algorithms. As stated in [64], the *Complete-link*, K-means and BIRCH algorithms cannot discover the clusters on the four data sets, while the *Single-link* equipped with outlier elimination, DBScan and CURE algorithms can correctly discover the clusters on one or two of the first three data sets but all fail in the complex DB4. The hybrid clustering algorithm [64] is able to discover the clusters on the four data sets. However, its clustering result is sensitive to the initial settings. As reported in [64], the probabilities of successful partitions are about 95%, 90%, 65% and 40% on the DB1, DB2, DB3 and DB4, respectively, after performing this hybrid algorithm 20 times with random initialization on each data set. Similarly, the proposed algorithm is performed on each of the four 2D data sets over 20 random runs with the initial number of prototypes set to $3K$. The probabilities of the successful partitions are 100%, 100%, 95% and 90% on the DB1, DB2, DB3 and DB4, respectively. The average numbers of prototypes to finally model the clusters are 17, 26, 39 and 62 for DB1, DB2, DB3 and DB4, respectively. Thus, more prototypes are usually required to represent the clusters which are more complex in shape and size. Table 5.4 shows the clustering results of different algorithms on these four 2D data sets

Table 5.4: Clustering results of different algorithms on four data sets.

Algorithms	DB1	DB2	DB3	DB4
CURE	100%	100%	0	0
DBScan	0	100%	100%	0
Hybrid clustering [64]	95%	90%	65%	40%
Our algorithm	100%	100%	95%	90%

5.4.2 Real Data Sets

To show the practical applicability of our proposed multi-prototype clustering algorithm, we perform it on three real data sets: Iris data, Wine Recognition Data, Wisconsin Breast Cancer Data and image segmentation Data, which are available at UCI Machine Learning Repository [77]. The class label is given for each pattern in these data sets. It is ignored during the clustering but used for evaluation of clustering performance. The clustering error rate is used to evaluate the performance of clustering algorithm. It is computed by [57]:

$$Error = \frac{\text{the number of misclassified patterns}}{\text{the number of patterns in data set}} \times 100\% \tag{5.9}$$

To compute the clustering error rate, the major problem is the correspondence between the given class labels and the found clusters of the data sets by the clustering algorithm. We perform the matching between them. A cluster l corresponds to class label q if the number of patterns labelled as q in l is larger than those of other class labels. The best matching of the clusters is selected as the correspondence to the class labels.

We compare our proposed algorithm with some existing clustering algorithms such as K-means, agglomerative hierarchical clustering and hybrid algorithms on these real data sets. The proposed algorithm is performed on each data set over 20 random runs and the best clustering result is presented. We implement the K-means algorithm with a good initialization of the cluster centers in [57] on the Wisconsin Breast Cancer Data while the clustering results on the other two

real data sets are reported in [57]. We perform the three common agglomerative clustering algorithms: *Single-link* [94], *Complete-link* [59] and *Average-link* [94] on each data set and present the best clustering result of them. In addition, the hybrid algorithm [64] is also implemented on these real data sets with our best effort. We give the best clustering result after 20 random runs of it on each data set. For other clustering algorithms, we present the results reported in the literature.

The Iris data set consists of 150 patterns and each pattern is represented by four numerical features: sepal length, sepal width, petal length and petal width. Three types of Iris flowers: setosa, versicolor and virginica are labelled as class I, II and III, respectively. Each class consists of 50 patterns. This data set is often used to test the clustering algorithms and the clustering results of three clusters are reported in [24] and [57]. [24] also gives the clustering results by the *Single-link* and *Complete-link* algorithms. Our algorithm produces five prototypes to model the three clusters on this data set. One prototype is used to represent cluster C1 which is easier to be separated from others. Two prototypes are used to represent each of cluster C2 and C3. Table 5.5 summarizes the clustering results for this real data set. We can see our algorithm outperforms the other clustering algorithms.

Table 5.5: The clustering results for Iris data.

Found cluster	Given class			Clustering Error Rate (%)				
	I (50)	II (50)	III (50)	Our algorithm	[57]	[64]	[59]	[24]
C1	50	0	0	2.67	11.33	4.0	4.0	7.4
C2	0	47	1					
C3	0	3	49					

The Wine Recognition data set contains the results of a chemical analysis of the wines grown in the same region in Italy but derived from three different cultivars. The wines from three cultivars represent three types of wine data labelled as class I, II and III, respectively. This data set consists of 178 patterns and each pattern is represented by 13 features such as alcohol, magnesium, color intensity, etc.. The feature values are normalized to $[0, 1]$ to balance the effects of the features measured on different scales. This data set is used to test the clustering algorithms [57,93] and

the clustering results are reported in the literature. Five prototypes are produced in our algorithm to model the three clusters on this data set. Two prototypes are used to represent each of cluster C1 and C2. Cluster C3 is represented by one prototype. Table 5.6 shows the clustering results for this real data set. We can see that the clustering result of our algorithm is better than those of others.

Table 5.6: The clustering results for Wine Recognition data.

Found cluster	Given class			Clustering Error Rate (%)				
	I (59)	II (71)	III (48)	Our algorithm	[57]	[64]	[94]	[93]
C1	59	2	0	2.25	5.05	3.93	5.62	17.42
C2	0	67	0					
C3	0	2	48					

The Wisconsin Breast Cancer (WBC) data set consists of 683 patterns which belong to two types of patterns: 444 benigns and 239 malignants labelled as class I and II, respectively. Each pattern is represented by nine features. This data set is also used to test the clustering algorithm [34]. As stated in [34], its clustering result is better than those in [61] and [21]. Our clustering algorithm produces four prototypes to model the two clusters on this data set. One prototype is used to represent cluster C1 and the other three prototypes represent cluster C2. The clustering results for this real data set are shown in Table 5.7. Our algorithm outperforms the other clustering algorithms on this real data set.

Table 5.7: The clustering results for Wisconsin Breast Cancer data.

Found cluster	Given class		Clustering Error Rate (%)				
	I (444)	II (239)	Our algorithm	[57]	[64]	[94]	[34]
C1	427	2	2.78	3.95	3.66	8.20	3.37
C2	17	237					

The image segmentation (image) data set consists of 2,320 patterns from 7 classes. Each pattern consists of 19 features extracted from a 3×3 region taken from seven types of outdoor images: brickface, sky, foliage, cement, window, path, and grass. Due to the complexity of this data set, we only present the mean and standard deviation of the clustering error rates over 20 random runs in the Table

5.8. This data set is used to test the clustering algorithm in [62].

In addition, we present the mean and standard deviation (in the parenthesis) of the clustering error rates over 20 random runs on each real data sets. Table 5.8 shows the experimental results on the above data sets. We compare the results of the proposed algorithm with those of K-means and agglomerative algorithms and the hybrid clustering algorithm [64]. The clustering results in [59]/ [94] show the best of three common agglomerative algorithms: *Single-link* [94], *Complete-link* [59] and *Average-link* [94] on each data set. They are stable so that the standard deviations of the error rates are zero.

Table 5.8: The clustering results of different algorithms over 20 random runs.

Data Sets	Clustering Error Rate (in %)				
	Our algorithm	K-means	[64]	[59]/ [94]	[62]
Iris	2.67 (0)	14.30 (10.16)	9.18 (4.75)	4.0 (0)	-
Wine	2.75 (0.17)	5.23 (0.66)	4.88 (0.81)	5.62 (0)	6.61 (3.91)
WBC	2.78 (0)	3.88 (0.07)	3.10 (0.09)	8.20 (0)	-
image	19.98 (1.75)	31.07 (3.93)	23.80 (3.11)	57.14 (0)	20.19 (1.54)

From above experimental results and comparisons, we can see that the proposed multi-prototype clustering algorithm performs better than or comparable to some existing clustering algorithms on these real data sets.

5.4.3 Discussion on Parameter T

Finally, we discuss the effects of parameter T on the clustering results. The setting of T is empirical and has some effects on the clustering result. If T is set too small, the iterative clustering process stops earlier which may result poor cluster separation and false clustering result. Thus, T is usually set to larger than 0.5. If T is set too large, however, the iterative clustering process may not stop due to the strict requirement. This problem can be alleviated by analyzing the stability of the clustering results in the iterative process. For the data set shown in Figure 5.2, the separations of two clusters with new prototypes added are shown in Figure 5.8. The obtained two clusters tend to be stable with the number of prototypes

larger than 9. The cluster separation equals to 0.7522 with 9 prototypes. The clusters are correctly discovered with T set to 0.8.

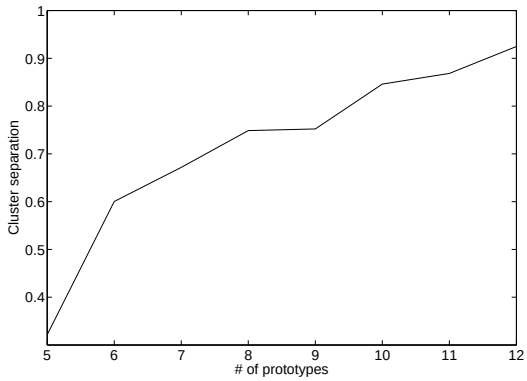


Figure 5.8: The cluster separation changes with the number of prototypes for the data set shown in Figure 5.2.

5.5 Summary

In this chapter, we have developed a multi-prototype clustering algorithm which can discover the clusters of arbitrary shapes and sizes. The squared-error clustering is used to produce a number of prototypes to locate the regions of high density because of its low computational cost and memory space and yet good performance. A separation measure is proposed to evaluate how well two prototypes are separated by a sparse region. Multiple prototypes with small separation are organized to model a given number of clusters in the agglomerative method. Instead of taking a single scan of the data set to produce a large number of prototypes for the clusters, our algorithm adds new prototypes iteratively to improve the poor cluster boundaries resulted by the poor initial settings. The proposed algorithm requires less memory space and computation cost than the commonly used hierarchical clustering algorithms such as *Single-link* and *Complete-link* while preserves much of the speed and efficiency of the squared-error clustering algorithm. Experimental results on both synthetic and real data sets show the effectiveness of the proposed clustering algorithm.

One of the important applications for clustering is to facilitate the information retrieval from the large database. Similarly, clustering can be used to explore the underlying data structure of the fingerprint database to facilitate the database search in an AFIS. Instead of classifying fingerprints into a small number of human interpretable classes, the clustering can partition the fingerprint database into a number of similar groups with more flexibility. In the next chapter, we will investigate to employ the clustering technique for the fingerprint retrieval.

Chapter 6

Clustering-based Fingerprint Retrieval

6.1 Introduction

As mentioned in Chapter 2, exclusive and continuous classifications are two common search strategies used for the fingerprint retrieval. The exclusive classification partitions the fingerprint database into non-overlapping and human-predefined classes with similar patterns. The query fingerprint is compared with the fingerprints of the same class in the fine matching of identification. It provides an efficient indexing mechanism to faciliate the search of fingerprint database for the large-scale identification system [89]. However, the exclusive fingerprint classification cannot sufficiently narrow down the search of database due to the small number of predefined classes and uneven fingerprint distribution. Moreover, it is sill a difficult problem to automatically and consistently classify fingerprints into predefined classes due to the small inter-class variability and large intra-class variability.

Instead of classifying each fingerprint into one of human interpretable classes, the continuous classification assigns the numerical feature vectors to each fingerprint [14, 53, 66]. Similar fingerprints are mapped into close points in the feature

space by a given distance measure. The database templates close to the the query fingerprint are retrieved for fine matching. The continuous classification can avoid the problems of exclusive classification and achieve better retrieval performance. In Chapter 4, we have developed a fingerprint retrieval algorithm based on continuous classification using two numerical features: orientation vector and dominant ridge distance, which has achieved better retrieval performance than some existing approaches. However, the continuous classification only ranks the database templates according to their similarities to the query fingerprint while neglecting the similarities among the database templates in fingerprint retrieval. The retrieval speed by the comparison of query fingerprint with all database template is still prohibitive for large database although this comparison is a coarse level search and is much faster than the fine matching.

In this chapter, we investigate to employ the database clustering for the efficient fingerprint retrieval. Instead of classifying fingerprints into a small number of human-predefined classes, the unsupervised clustering technique is used to partition the fingerprint database into a number of non-overlapping clusters with more flexibility. The query fingerprint is compared with a small number of cluster representatives instead of all database templates and the fingerprints from the close clusters are retrieved. Thus, the clustering technique can exploit the similarities among the database templates to speed up the retrieval process. But it deteriorates the retrieval accuracy and efficiency due to the quantization of the feature space and the number of fingerprints retrieved comparing to the continuous classification. Furthermore, we take advantages of both database clustering and continuous classification by performing the cluster retrieval followed by the continuous fingerprint classification (fingerprint search) to speed up the retrieval process without compromising the retrieval accuracy. Figure 6.1 shows the data-flow chart of the fingerprint retrieval framework based on database clustering. It differs from the continuous classification in that clustering is employed to partition the database into many groups for cluster search before fingerprint search (continuous fingerprint classification).

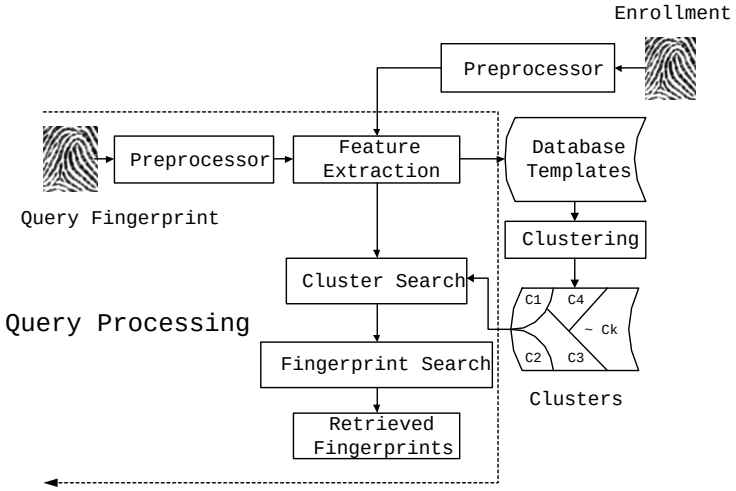


Figure 6.1: The overview of the clustering based fingerprint retrieval framework.

A clustering-based fingerprint retrieval algorithm is developed in this chapter. Similar to that of the proposed fingerprint retrieval algorithm in Chapter 4, the fingerprint orientation field and dominant ridge distance DRD are used as two retrieval features. Instead of uniformly dividing fingerprint to compute the orientation field, we propose to unevenly partition fingerprint image by a circular tessellation to compute a multi-scale orientation field as the main retrieval feature. The DRD same as that used in Chapter 4 is also employed as an auxiliary retrieval feature. The clustering-based fingerprint retrieval algorithm consists of two phases: offline database clustering and online query processing. During the offline database clustering, a modified form of K-means clustering is proposed to partition the high-dimensional orientation feature space into clusters and the fingerprints in each cluster are further divided into bins according to their DRDs. Based on the offline database clustering, a hierarchical online query processing is proposed to perform the cluster search before the fingerprint search to facilitate an efficient fingerprint retrieval. In the cluster search, each query fingerprint is compared with the cluster prototypes to retrieve the close clusters followed by searching the bins in the retrieved clusters. Fingerprint search is performed on the retrieved close bins and the database templates close to the query fingerprint are finally retrieved for the fine matching.

6.2 Feature Extraction

As mentioned in Chapter 3, the local ridge orientation and the local ridge distance are two important local parameters for fingerprint analysis. These two coarse level features have been combined for the fingerprint retrieval on continuous classification and have achieved good retrieval performance in Chapter 4. Similarly, they are employed for the clustering-based fingerprint retrieval algorithm. The dominant ridge distance DRD same as that used in Chapter 4, which is computed with Equations (4.4) and (4.5) from the local ridge distances of fingerprint, is also used as an auxiliary retrieval feature. In this section, we propose to unevenly partition each fingerprint by a circular tessellation to compute a multi-scale orientation field, from which an orientation vector of fixed size is constructed as the main feature for the clustering-based fingerprint retrieval.

To construct a compact orientation vector for the efficient retrieval, the local ridge orientation field of fingerprint are usually computed based on block wise instead of pixel by pixel. The orientation field computed by uniformly dividing fingerprint into blocks has been widely used as the feature for fingerprint classification [12, 53, 66]. However, this uniform spacing orientation field may obscure the representation power of the orientation elements in the important area which can be estimated in finer scale. Larger scale is often required in some areas for noise attenuation and dimensionality reduction. Since the singular regions of fingerprint (i.e., the regions near the singular points) have large orientation changes, a nonuniform spacing is proposed to concentrate orientation measurements more densely in the areas more likely to contain the singular points [104]. It outperforms the uniform spacing orientation field for fingerprint classification. A nonuniform partition of fingerprint by tessellating the region of interest is also proposed to extract the Gabor filter responses for construction of "FingerCodes" [49]. This tessellation puts finer scale measurements on the inner region of interest than the outer region. Thus, the nonuniform spacing of fingerprint can be used to strengthen the feature elements with large representation power without compromising the performance

of noise attenuation and the compactness of feature vector.

The entropy of a random variable measures its uncertainty. In Chapter 4, the representation power of each orientation element is evaluated by the entropy of its orientation values over all the aligned orientation fields. The first fingerprint instances of NIST database-4 are used to test the representation power. Each orientation field is computed by uniformly dividing the fingerprint into blocks and aligned by the reference point at the central point and reference direction pointing to south-most. The entropy map of all the aligned orientation fields is shown in Figure 4.3 (a). Thus, the representation power of the orientation elements is unevenly distributed on the fingerprint image. The entropy values are used to weight the uniform spacing orientation elements in the orientation comparison of the continuous classification and better retrieval performance has been achieved than that without the feature weights in Chapter 4.

In this chapter, instead of imposing the regional feature weights on the uniform spacing orientation field, we propose to unevenly partition fingerprint image to compute a multi-scale orientation field for fingerprint retrieval. From Figure 4.3, we find that the elements in the region below the reference point have more variant orientation patterns than those above. To improve the discriminability of orientation vector, the elements with large representation power can be estimated in finer scale. In addition, the ridge curvature of the inner region around the reference point is usually larger than that of outer region in a fingerprint. The orientations of the high curvature area can be estimated in finer scale than those of low curvature area. Based on above observations, we construct a circular tessellation to unevenly divide fingerprint and compute a multi-scale orientation field. Let $I(x, y)$ be the gray value at pixel (x, y) of a fingerprint of size $X \times Y$. To achieve the invariance of pose transformation, translation and rotation are usually needed to bring two different fingerprints into alignment. The reference point (x_r, y_r) and the corresponding reference direction θ_r, $-\pi < \theta_r \leq \pi$ detected in Chapter 3 are employed for the translational and rotational alignments of fingerprints, respectively. The

circular tessellation is composed of sectors determined by the radius r from (x_r, y_r) and the rotation angle φ from θ_r. Sector $S_{i,j}$ $(1 \leq i \leq E, 1 \leq j \leq F)$, the jth sector of the ith band, is computed as:

$$S_{i,j} = \{(x,y)|(i-1)b+b_0 \leq r < i \cdot b+b_0, \varphi_{j-1} \leq \varphi < \varphi_j, 1 \leq x \leq X, 1 \leq y \leq Y\} \tag{6.1}$$

$$r = \sqrt{(x-x_r)^2 + (y-y_r)^2}, \varphi = tan^{-1}(\frac{y-y_r}{x-x_r}) - \theta_r \ mod \ 2\pi \tag{6.2}$$

where b is the band width and b_0 is the width of the innermost band which is not used for orientation extraction due to its large orientation inconsistency. The parameters φ_j, b, E, F are determined empirically to obtain the best performance of fingerprint retrieval. Each band is segmented into 13 nonuniform sectors $(F = 13)$ and finer scale estimation is put in the region with φ close to θ_r than that far from θ_r. $\varphi_j = j\pi/8$ for $1 \leq j \leq 5$, $(2j-5)\pi/8$ for $6 \leq j \leq 8$ and $(j+3)\pi/8$ for $9 \leq j \leq 13$, respectively. Parameter b depends on the dpi resolution of the sensor. It is set to 18 pixels for the fingerprints scanned at 500 dpi. Parameter E depends on the size of the contact area of fingertip with sensor. The region around the reference point is partitioned into 12 bands $(E = 12)$ for the fingerprints of 512×480 pixels in the NIST database-4. The circular tessellation is shown in Figure 6.2.

The local ridge orientation of each sector is estimated in the least square averaging method based on the gradients [42]. The unit vector of doubled angles $[\cos(2\theta), \sin(2\theta)]$ is often used to represent each local orientation that facilitates the feature transform and weighting [12, 53, 66]. But this representation doubles the size of feature vector and requires extra memory and computation time in the database searching process. Similar to Chapter 4, we choose the angle representation to reduce the dimensionality of orientation vector by half. Let $\hat{\theta}_{i,j}$ be the local ridge orientation of sector (i,j). Due to the periodicity and discontinuity of $\hat{\theta}_{i,j}$ at $\pm\pi/2$ and θ_r at $\pm\pi$, each aligned local orientation is computed with Equation

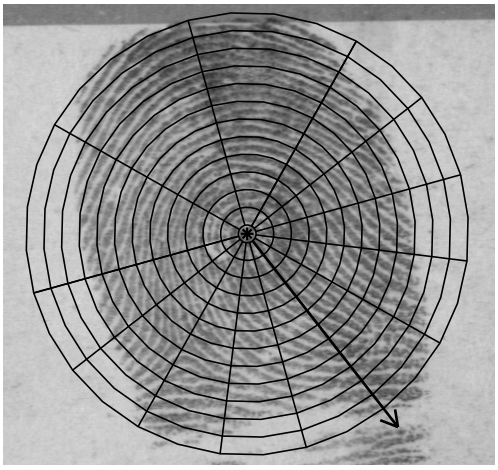


Figure 6.2: The circular tessellation of fingerprint aligned by the reference point ($*$) and direction (the line with arrow).

(4.2) in Chapter 4.

An orientation vector of fixed size is constructed for a fingerprint q by concatenating the aligned local orientations of all sectors. Let $\Theta_q = [\theta_{q,1}, \theta_{q,2}, ..., \theta_{q,M}]$ ($M = E \times F$) denote the orientation vector. The orientation in each sector captures the local ridge flow pattern, while the ordered enumeration of the tessellation describes the global relationships among the local patterns. This orientation vector consists of 156 elements. It covers most important areas of fingerprint if the reference point is located at the center of image. However, only a part of fingerprint is included in the tessellation due to the unfavorable position of the reference point so that a substantial number of feature elements are from the noisy background or the outside of fingerprint image. The segmentation is therefore used to label the valid and invalid elements of the feature vector. The segmentation result of fingerprint q is denoted with a vector $S_q = [s_{q,1}, s_{q,2}, ..., s_{q,M}]$ where $s_{q,k} \in \{0, 1\}$ and $s_{q,k} = 1$ indicates that sector k is segmented as foreground and valid for feature selection.

6.3 Clustering-based Fingerprint Retrieval

Clustering is a crucial technique widely used in discovering the underlying structure in a data set by unsupervisedly grouping the similar patterns. It groups the data set into a number of clusters such that the patterns within cluster are more similar than those from different clusters. Clustering has been widely used in many applications such as data mining, information retrieval, image segmentation and pattern recognition. It is also used to accelerate the content based image retrieval by comparing the query image with a small number of cluster representatives instead of all database templates [63, 109]. Although the fingerprint retrieval on continuous classification can achieve good retrieval performance, the retrieval process is time consuming due to the exhaustive database search. Thus, we propose to employ the database clustering to exploit the similarities among the database templates for the efficient fingerprint retrieval. The 156-D orientation vector constructed from the multi-scale orientation field is used as the main retrieval feature while 1-D DRD is used as an auxiliary feature. Clustering is used to partition the fingerprint database into a number of groups with the similar feature sets. The query fingerprint is coarsely compared with the prototypes of the groups the number of which is much smaller than that of database templates and the close groups are retrieved for further fingerprint search. The proposed clustering-based fingerprint retrieval algorithm consists of two phases: offline database clustering and online query processing.

6.3.1 The Offline Database Clustering

The orientation feature space is high-dimensional and unevenly distributed. For example, the orientation fields of whorl fingerprints are more variant than those of plain arch fingerprints as a whorl fingerprint contains more singularities with sharp orientation changes. The retrieval process by exhaustively searching the high-dimensional orientation feature space is time consuming especially when the

database is large. The clustering can be used to exploit the similarities among the database templates to speed up the retrieval process. We propose a clustering technique to partition the orientation feature space into non-overlapping clusters with more flexibility for the efficient fingerprint retrieval.

The K-means clustering algorithm [76] is the most widely used partitional clustering algorithm because of its high computational efficiency and low memory space requirement. Some of its variants have been proposed for the specific applications in the literature [23, 95]. This clustering algorithm represents each cluster with its mean vector and assign each pattern to the cluster with the closest prototype iteratively. It terminates when the cluster labels do not change. However, using one prototype to represent each cluster may not adequately model the clusters on the given data set especially when the clusters are of arbitrary shape and size. Recently, support vector clustering is proposed to generate the cluster boundaries of arbitrary shape by transforming the original space to a high dimensional space with a kernel function [7]. Although this algorithm can solve some difficult clustering problems, it is not easy to choose a suitable kernel parameter and the clustering result cannot provide information about the representation of cluster. In Chapter 5, a multi-prototype clustering algorithm has been developed to use multiple prototypes to represent the complex cluster. This clustering algorithm begins with an initial partitioning of the data set into a relatively larger number of small subclusters than the expected clusters and each subcluster is represented by one prototype. The multiple prototypes with small separation are organized to model a given number of clusters in the agglomerative method. New prototypes are iteratively added to improve the poor cluster boundaries resulted by the poor initial settings. As a result, the proposed algorithm can discover the clusters of complex structure and is robust to prototype initialization.

The clusters in the multi-dimensional orientation feature space of fingerprints may also be complex in shape and size. Moreover, some ambiguous fingerprints are located near the cluster boundaries no matter how well the database is parti-

tioned. Therefore, similar to the exclusive fingerprint classification, the retrieval efficiency and accuracy are limited if the fingerprint database is partitioned into a small number of clusters by the clustering technique and the fingerprints from the nearest cluster are retrieved. Instead, a modified form of the K-means clustering is developed to partition the orientation feature space into a relatively large number of clusters to avoid the low retrieval efficiency.

It is well known that the K-means clustering needs to specify the number of clusters previously. Instead of understanding the inherent data structure correctly, the main purpose of clustering for fingerprint retrieval is to exploit the similarities among the database templates and facilitate a fast and effective query process. Thus, the initial number of clusters is approximately determined to balance the time efficiency and accuracy of fingerprint retrieval. After grouping N patterns into K clusters, the average number of comparisons is approximately computed as $K + \frac{N}{K}$ for retrieving the nearest cluster followed by fingerprint search in the retrieved cluster. It is minimized when $K = \sqrt{N}$. However, multiple clusters close to the query fingerprint are often retrieved to improve the accuracy of fingerprint retrieval. To balance these effects, the initial number of clusters is approximately set to about $\tau\sqrt{N}(1 < \tau < 3)$.

Euclidean distance measure is often used in the traditional K-means clustering algorithm to assign each pattern to its closest cluster. This makes it only effective to discover the hyperspherical clusters. Since our feature vector for clustering is composed of the orientation angles, Euclidean distance cannot be directly applied to compare the orientation vectors due to the periodicity and discontinuity of orientations. In addition, if all elements between two orientation vectors consistently have a constant difference, the orientation fields of such two vectors are very similar just with a rotation in human perception. The distance between them is zero but their Euclidean distance may achieve a large deviation. In Chapter 4, to overcome these problems, a distance measure is proposed in Equation (4.10) by averaging the unit vectors of the doubled difference instead of the squared dif-

ferences over all valid orientations. The orientation distance measure in Equation (4.10) is based on the inconsistency of the orientation differences among all valid elements and is invariant to the constant amount of orientation differences caused by a slight rotation between two aligned fingerprints. It performs better to quantify the distance between two orientation vectors than the traditional Euclidean and Manhattan distance measures. The modified K-means clustering employs this distance measure to assign each orientation vector to the cluster with the closest prototype.

After all feature vectors are assigned to their closest clusters in each iteration, the mean vector of each cluster is computed as its new prototype in the traditional K-means clustering. However, the mean vector by directly averaging the orientation vectors is not applicable due to their periodicity and discontinuity. For example, the average orientation of 0 and π is 0 instead of the arithmetic mean value $\frac{\pi}{2}$. To avoid this problem, the orientation averaging is often performed by separately averaging two components of the unit vector of doubled angles $[\cos 2\theta \ \sin 2\theta]$. This method is used to compute the mean vector of each cluster. Let $\{Z_1, Z_2, ..., Z_K\}$ denote the K cluster prototypes where $Z_l = [z_{l,1}, z_{l,2}, ..., z_{l,M}]$ $(1 \leq l \leq K)$. In the modified K-means clustering, the mth element $z_{l,m}$ of the prototype Z_l of cluster C_l is updated as:

$$z_{l,m} = \frac{1}{2} \arctan \frac{\sum_{p \in C_l} s_{p,m} sin2\theta_{p,m}}{\sum_{p \in C_l} s_{p,m} cos2\theta_{p,m}}, 1 \leq m \leq M, 1 \leq l \leq K \tag{6.3}$$

The K-means clustering algorithm cannot guarantee the global minimum of the cluster criterion so that the clustering result is sensitive to the initial cluster prototypes. Cluster split and merge techniques are often used to alleviate this problem [23]. In our application, the skewed fingerprint distribution on the clusters may deteriorate the effectiveness of fingerprint retrieval. Let P_l $(1 \leq l \leq K)$ be the percentages of fingerprints assigned to cluster C_l. The average portion of fingerprints retrieved is about $\sum_{l=1}^{K} P_l^2$ if the nearest cluster is retrieved for each query fingerprint. It achieves its minimum on even distribution ($P_1 = P_2 = ... =$

P_K) and increases on the skewed distribution. To alleviate the above problems, the modified K-means clustering eliminate the small clusters and split the large clusters into two clusters. To split the large cluster, a new prototype is added by randomly choosing one feature vector from the cluster. In our experiments, the cluster with $P_l < \frac{1}{3K}$ is considered as small cluster while the cluster with $P_l > \frac{2}{K}$ is considered as large cluster.

The traditional K-means clustering is modified by replacing the Euclidean distance measure with the distance measure (4.10), computing the cluster prototype with equation (6.3), eliminating the small clusters and splitting the large clusters. The modified K-means clustering algorithm repeats above procedures and outputs the clusters when the cluster prototypes do not change. The processing steps of this algorithm are summarized as:

(1) Initialize the number of clusters $K = \tau\sqrt{N}(1 < \tau < 3)$ and randomly choose the initial cluster prototypes;

(2) Compute the distances between each orientation vector and the K cluster prototypes with equation (4.10) and assign the corresponding fingerprint to the closest cluster;

(3) Compute the new cluster prototypes with equation (6.3);

(4) Compute the distances between the new and old prototypes with equation (4.10). If the maximum one is larger than ε, go to step (2);

(5) If there are no small or large clusters, output the clusters. Otherwise, eliminate the small clusters, split the large clusters and go to step (2).

After partitioning the orientation feature space into clusters, the 1-D dominant ridge distance DRD is employed as an auxiliary feature to further divide the fingerprints of each cluster into bins. Over the whole range of the DRD, B bins of equal width are predefined in each cluster. The center of the kth bin is computed as $DRD_{min} + k \times (DRD_{max} - DRD_{min})/B$. Since more than 3 bins close to the

DRD of the query fingerprint are usually retrieved in the fingerprint retrieval, B is set to larger than 20. The fingerprint in each clusters is assigned to the bin with the closest center to its DRD. After the offline database clustering, the hierarchical data structure of the fingerprint database is shown in Figure 6.3.

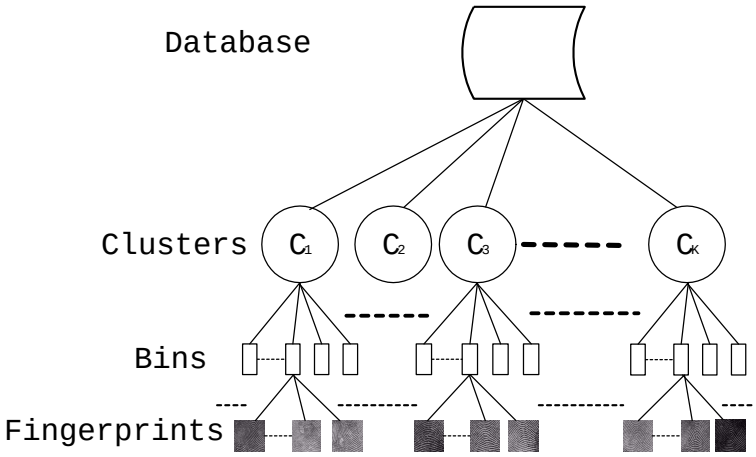


Figure 6.3: The hierarchical data structure of the fingerprint database.

6.3.2 The Online Query Processing

The online query processing of fingerprint retrieval is to search the fingerprint database and select a subset of database templates close to the query fingerprint for the fine matching. Based on the offline database clustering, the query fingerprint can be compared with the representative prototypes of the groups (clusters and bins) instead of all database templates to speed up the query process. However, comparing to the continuous fingerprint classification, the retrieval accuracy and efficiency will deteriorate due to the quantization of the feature space and the number of fingerprints retrieved. We propose a hierarchical query process for the clustering-based fingerprint retrieval that consists of three levels of database search (see Figure 6.4). In the first level, we search the clusters by comparing the query orientation vector with the cluster prototypes. Some ambiguous fingerprints are located near the cluster boundary no matter how well the database is partitioned. In Chapter 5, the multiple close prototypes are grouped to represent one cluster in

the proposed multi-prototype clustering algorithm. Similarly, we retrieve multiple clusters nearest to the query fingerprint instead of only the nearest one to alleviate this problem. In the second level, we search the bins of the retrieved clusters by comparing the query DRD with the bin centers and retrieve multiple nearest bins. These coarse level searches can efficiently narrow down the search of database because the number of groups is much smaller than the number of database templates. However, the retrieval performance is limited due to the quantization of the feature space and the number of fingerprints retrieved. In the third level, the continuous fingerprint classification is applied on the coarsely retrieved fingerprints (i.e., fingerprint search) to further improve the retrieval performance. The fingerprint retrieval is performed by comparing the query orientation feature with the database templates in the retrieved bins to further narrow down the search space of fine matching. In this way, the online query processing of fingerprint retrieval is accelerated by database clustering without compromising the retrieval accuracy and efficiency.

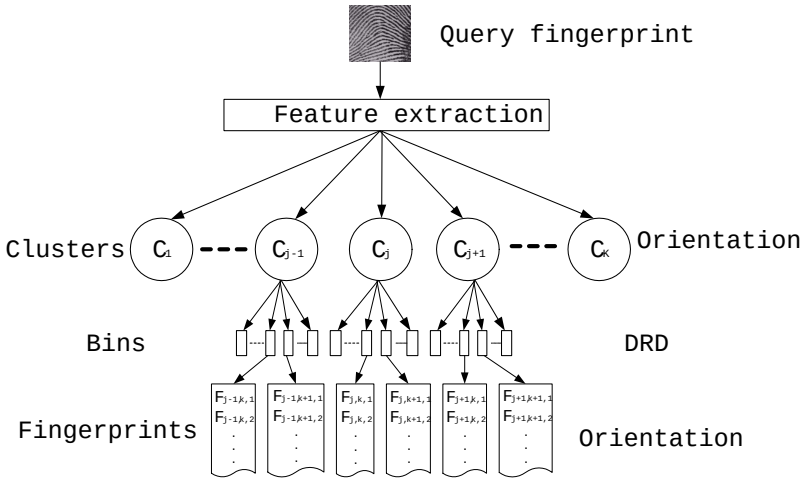


Figure 6.4: The hierarchical online query processing by incorporating two features.

For search of clusters, we compute the orientation distances $d_C(\Theta_q, Z_l)(1 \leq l \leq K)$ between the query orientation vector and K cluster prototypes with the equation (4.10) and retrieve the clusters with the distance smaller a threshold. Since the clusters may be unevenly distributed in the orientation feature space,

the retrieval threshold is adaptively determined as $\min_{l=1}^{K} d_C(\Theta_q, Z_l) + \sigma_1$ where σ_1 is tuned to adjust the size of retrieval neighborhood. It is more likely to retrieve the correct database template if more clusters are retrieved (larger σ_1). In the retrieved clusters, we compute the distances between the query DRD and the centers of all bins and retrieve the bins with the distance smaller than a threshold. This threshold is set to produce small retrieval error in this coarse level search. It is constantly specified as 1 pixels in our experiments on the NIST database-4.

In the final search of the candidate fingerprints, we compute the orientation distances $d_C(\Theta_q, \Theta_j)$ between the query fingerprint and all fingerprints in the retrieved bins with the equation (4.10). The fingerprints with the distance smaller than a threshold are finally retrieved for the fine matching. In Chapter 4, an adaptive setting method of the retrieval threshold has been introduced for the fingerprint retrieval on continuous classification and has slightly improved the retrieval performance comparing to those by the fixed distance and fixed order. Similarly, this retrieval threshold is adaptively determined as $\min_{j=1}^{N_q} d_C(\Theta_q, \Theta_j) + \sigma_2$ where N_q is the number of fingerprints in the retrieved bins and σ_2 is tuned to adjust the average portion of retrieved candidate fingerprints (i.e., the penetration rate).

6.4 Experimental Results

In this section, we present the experimental results and comparisons to demonstrate the performance and advantages of the proposed clustering-based fingerprint retrieval algorithm. The two data sets from the NIST database-4 used to test the fingerprint retrieval algorithm in Chapter 4 are also applied to test the performance of the proposed clustering-based fingerprint retrieval algorithm. The data set 1 is the whole NIST database-4 which contains 2,000 pairs of fingerprints of size 480×512 pixels. These fingerprints are taken from 2,000 different fingers with two instances per finger. Most published results of fingerprint retrieval on classification and indexing are based on the NIST database-4. To have a comprehensive com-

parison, we also implement our algorithm on this well-known database. However, NIST database-4 is collected mainly for testing the Henry five classification so that the classes: arch, tented arch, left loop, right loop and whorl are evenly distributed in the database. The natural fingerprint distribution in these five classes is significantly different. Therefore, the data set 2 contains 1204 pairs of fingerprints from the NIST database-4 which are obtained by reducing the number of fingerprints of less frequent classes according to the natural distribution. The first fingerprint instances of these data sets are used as the database templates for retrieval while the second instances serve as the query fingerprints.

As discussed in Chapter 4, retrieval efficiency and accuracy/error rate are two main evaluations of the retrieval performance. The retrieval efficiency is indicated by a so called "penetration rate", which is the average portion of database retrieved for the fine matching over all query fingerprints. It evaluates how much the fingerprint retrieval can narrow down the search of database and is controlled by the parameter σ_1 and σ_2 in our proposed approach. For a query fingerprint, the retrieval is successful if one of the retrieved candidate fingerprints is from the same finger as the query. It is more likely to retrieve the correct one if more database templates are retrieved as candidates. The retrieval error/accuracy rate is thus calculated by the percentage of the query fingerprints with false/correct retrieval at a given penetration rate. These performance estimates are independent of the size of database used, although the uncertainties in the estimates depend on the size of the sample pair populations. In addition, the computation complexity of the online fingerprint retrieval, which is called "retrieval complexity" in this work, is an important performance to evaluate how fast the online retrieval process is. The orientation comparisons with Equation (4.10) cost most of the computation in the online query processing of the proposed fingerprint retrieval algorithm. Hence, the retrieval complexity is evaluated by the average number of such comparisons required over all query fingerprints.

6.4.1 Experiments on Feature Extraction

This experiment is implemented on the NIST database-4 (data set 1) to test the effectiveness of two features for fingerprint retrieval: a 156-D orientation vector constructed from the multi-scale orientation field and a 1-D DRD. The retrieval result by the orientation feature computed on the proposed nonuniform spacing of fingerprint is compared with that by the uniform spacing of fingerprint. To fairly show the effectiveness of the orientation extraction for fingerprint retrieval, we apply the continuous fingerprint classification on the orientation vectors constructed from the uniform spacing and the proposed multi-scale orientation fields. Same as in Chapter 4, the uniform spacing orientation field is computed by dividing fingerprint into blocks of size 27×27 pixels and a 192-D orientation vector is constructed by concatenating the phase angles after fingerprint alignment. The distance measure of equation (4.10) is used to compare the query orientation vector with all database templates. The retrieval threshold is adaptively determined in the same way as that used in Chapter 4. In addition, to show the improvement of retrieval performance by adding the DRD, the fingerprints whose DRDs are close to that of query fingerprint (their distances are smaller than 1 pixel) are retrieved for the further fingerprint search based on the nonuniform spacing orientation vectors.

The results of fingerprint retrieval on different feature sets are shown in Figure 6.5 where the penetration rate is adapted by varying the parameter σ_2. We can see that the orientation extraction by our proposed nonuniform spacing of fingerprint not only produces more compact feature vector but also achieves better performance of fingerprint retrieval than that by the uniform spacing. Similar to Chapter 4, the 1-D DRD as an auxiliary feature consistently reduces the retrieval errors. It also reduces about 38% orientation comparisons in the query process of fingerprint retrieval since the average portion of the retrieved fingerprints by DRD is about 62% of the fingerprints in the database.

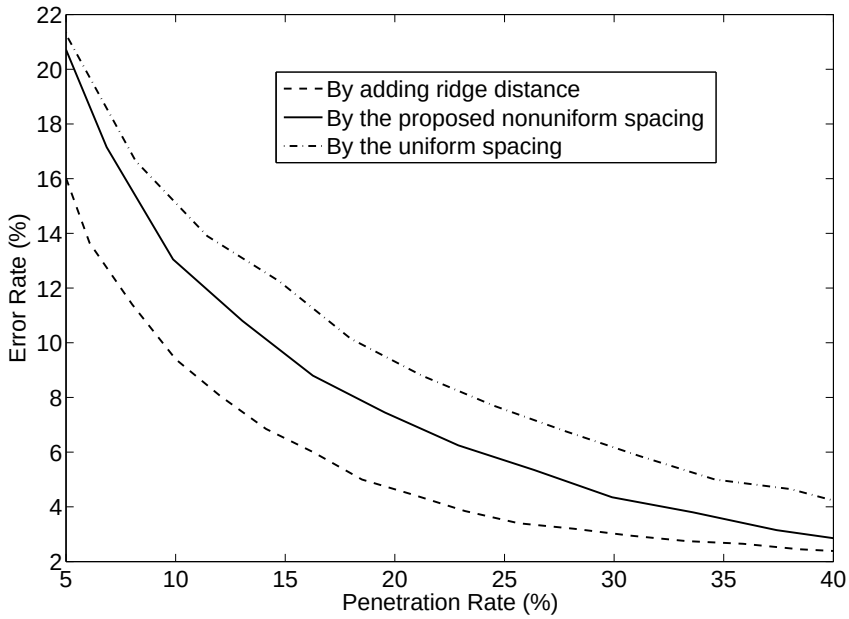


Figure 6.5: Results of fingerprint retrieval by the orientation feature computed on the uniform and proposed nonuniform spacing and by adding the dominant ridge distance.

6.4.2 Experiments on Clustering-Based Fingerprint Retrieval

Extensive experiments and comparisons are performed on the NIST database-4 for the proposed clustering-based fingerprint retrieval algorithm.

The first experiment is to show the effectiveness of the proposed modified K-means clustering technique for the fingerprint retrieval. We compare the retrieval performance based on the modified K-means clustering technique to that on the traditional K-means clustering technique. To apply the traditional K-means clustering algorithm, the orientation vector is constructed by the unit vectors $[\cos(2\Theta), \sin(2\Theta)]$ instead of the phase angles and Euclidean distance measure is used to assign each fingerprint to the closest cluster. The number of clusters and the cluster prototypes are initialized same for both clustering techniques. The final numbers of clusters are 90. To better reflect the effectiveness of clustering techniques on the fingerprint retrieval, we present the retrieval results by only retrieving the close clusters produced by clustering the orientation feature space. The 1-D DRD and the following fingerprint search are not used in this experiment.

Figure 6.6 shows the experimental results of the fingerprint retrieval based on the cluster search where the penetration rate is adapted by varying the parameter σ_1. We can see that our proposed modified K-means clustering outperforms the traditional K-means clustering for the fingerprint retrieval.

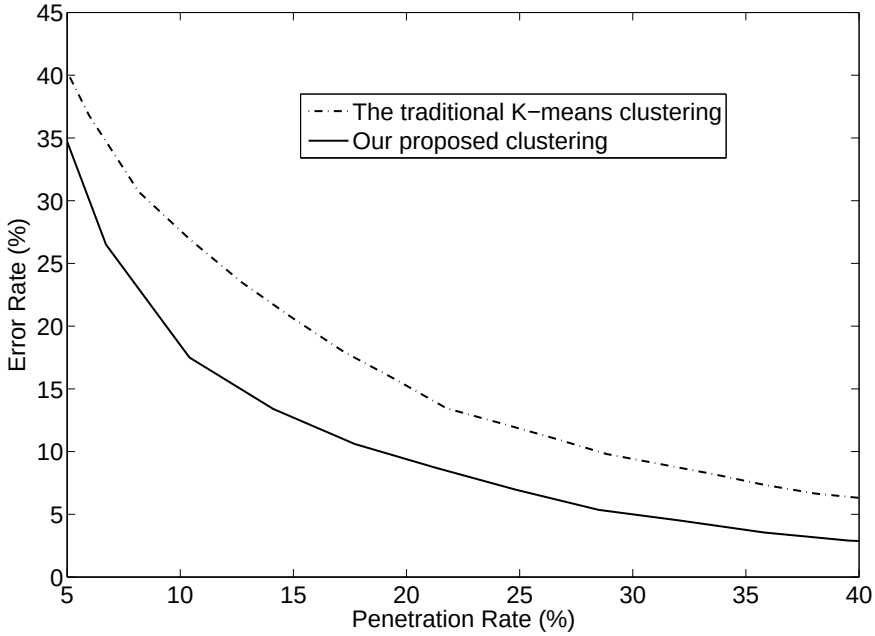


Figure 6.6: The results of fingerprint retrieval based on the traditional K-means clustering technique and our proposed modified K-means clustering technique.

The second experiment is to show the improvement of retrieval performance by applying the third level search of fingerprint database (i.e., continuous fingerprint classification) on orientation feature. In the query process of the proposed clustering-based fingerprint retrieval, we report the retrieval results on the first two coarse level search (cluster retrieval) and on the finest level search (cluster retrieval & continuous classification). Figure 6.7 shows these retrieval results. We can see that the retrieval accuracy is consistently improved, especially at the low penetrate rates, after applying the continuous classification of the third level search. These results demonstrate that cluster search produces the limited retrieval performance due to the quantization of the feature space and the number of fingerprints retrieved although it can speed up retrieval process. The following continuous classification can further improve the fingerprint retrieval performance.

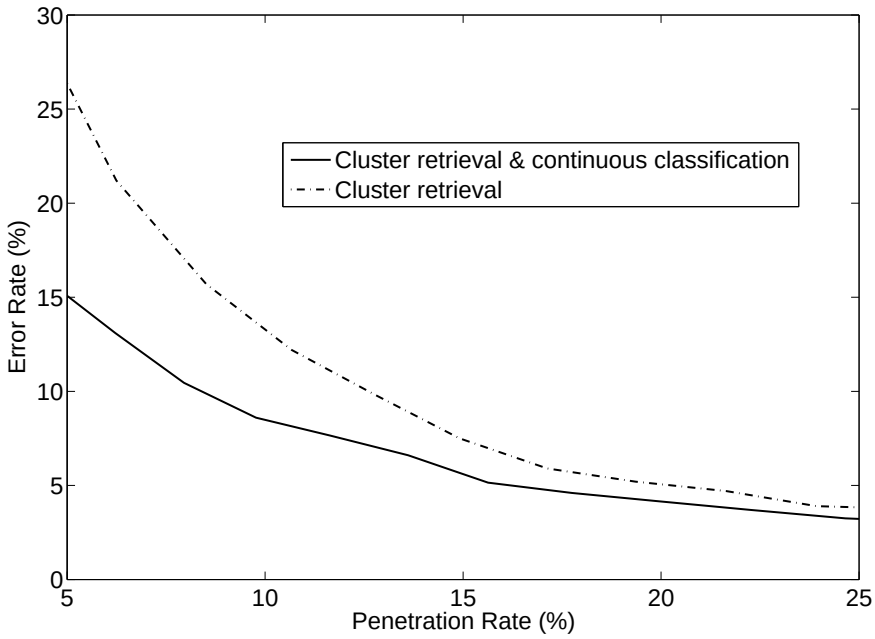
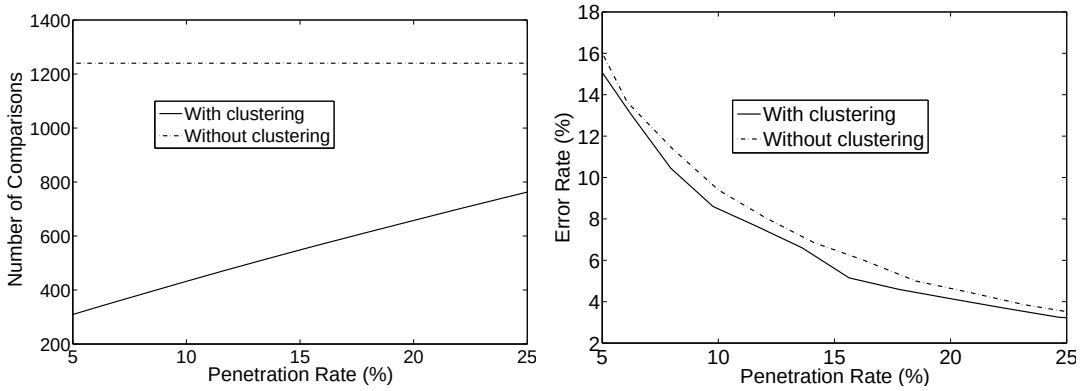


Figure 6.7: The results of fingerprint retrieval on the first two coarse level search (cluster retrieval) and on the finest level search (cluster retrieval & continuous classification).

The third experiment is to compare the performance of the clustering-based fingerprint retrieval (with clustering) with that of fingerprint retrieval based on continuous classification (without clustering). The same feature set is used in the fingerprint retrieval procedures with and without clustering. The continuous fingerprint classification is to exhaustively search the fingerprint database. Using the 1-D DRD as an auxiliary feature for continuous fingerprint classification will narrow down the search space to about 62% of database. In the query process of the clustering-based fingerprint retrieval, the number of orientation comparisons is the number of clusters (90 in our experiments) plus the number of fingerprints in the retrieved bins. It varies at different penetration rates. For example, a small number of clusters are retrieved so that the following fingerprint search is fast at the low penetrate rate. Figure 6.8 shows the results of fingerprint retrieval on the continuous classification (without clustering) and the clustering-based fingerprint retrieval (with clustering). From Figure 6.8 (a), we can see that the retrieval complexity is greatly reduced by using the clustering, especially at the low penetration

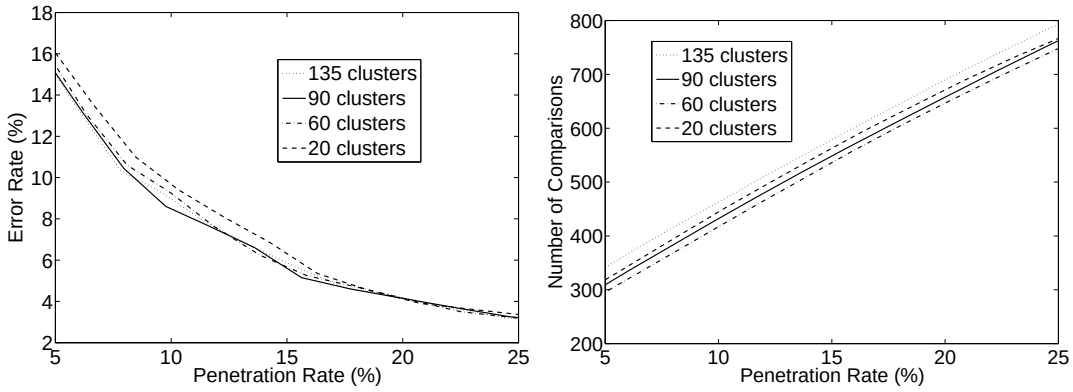
rates. For example, to retrieve 5% of database, we requires 300 ($N \times 15\%$) orientation comparisons in the query process of clustering-based fingerprint retrieval that is much smaller than 1240 ($N \times 62\%$) orientation comparisons in the continuous fingerprint classification. Moreover, the retrieval accuracy is slightly yet consistently improved by using the modified K-means clustering (see Figure 6.8 (b)). This may be resulted by exploiting the similarities among the database templates through the clustering to facilitate an effective fingerprint retrieval. These results demonstrate that the proposed clustering-based approach not only speeds up the retrieval process but also improves the retrieval accuracy.



(a) (b)

Figure 6.8: Results of the continuous fingerprint classification (without clustering) and our proposed fingerprint retrieval (with clustering): (a) retrieval complexity and (b) retrieval error rate.

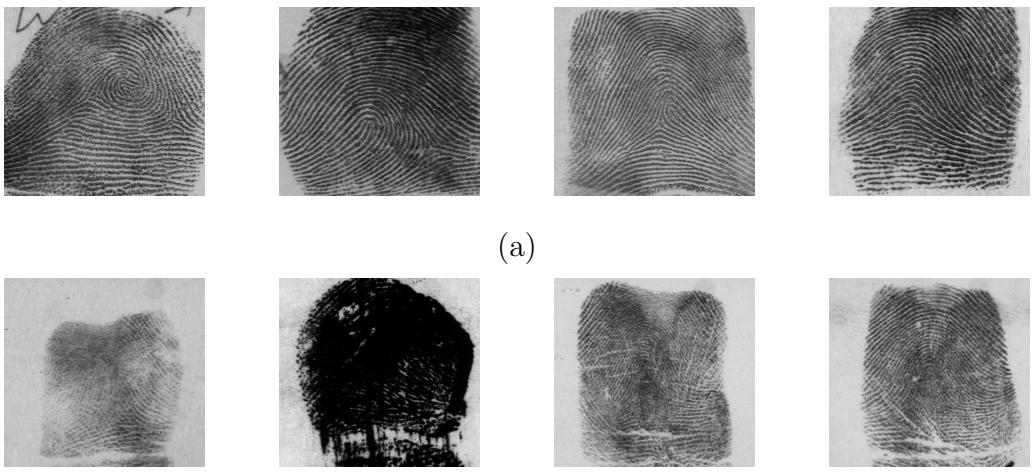
The forth experiment is performed to test the effects of different number of clusters on the performance of fingerprint retrieval. Although it is not necessary to specify the optimal number of clusters for our fingerprint retrieval, the final number of clusters may have some effects on the retrieval results. The number of clusters varies from 20 to 135 in our experiments and their results of fingerprint retrieval are shown in Figure 6.9. The retrieval error rate is reduced when increasing the number of cluster from 20 to 90 and cannot be further reduced with the number of clusters increased to 135 (see Figure 6.9a). From Figure 6.9b, we can see that the retrieval complexity is reduced by increasing the number of clusters from 20 to 60 and deteriorates by further increasing it to 90 and 135.



(a) (b)

Figure 6.9: Results of fingerprint retrieval on the different number of clusters: (a) retrieval accuracy and (b) retrieval complexity.

In addition, we have manually checked the fingerprint images that can or cannot be correctly retrieved by our proposed algorithm. The retrieval performance can be affected by the image quality of both the query fingerprint and the corresponding database templates. Figure 6.10 shows some examples of the fingerprint images that can be correctly retrieved even at the low penetration rate of %5 while Figure shows some examples of the fingerprint images that cannot be correctly retrieved even at the high penetration rate of 40%.



(a)

(b)

Figure 6.10: 4 sample fingerprints with (a) correct retrieval at 5% penetration rate and (c) false retrieval at 40% penetration rate.

6.4.3 Comparisons with Other Approaches

In this subsection, the proposed clustering-based fingerprint retrieval approach is compared with some state-of-the-art approaches of fingerprint retrieval on exclusive and continuous classifications as well as fingerprint indexing in the literature in terms of retrieval accuracy and efficiency.

Two state-of-the-art approaches of continuous fingerprint classification in [14, 66] are tested on the data set 2. The approach [14] performs better than the approach [66]. To compare with them, we also perform our clustering-based fingerprint retrieval algorithm on the data set 2. Figure 6.11 shows the retrieval results reported in [14] and the experimental results of our proposed approach on the same data set. We can see that consistent performance improvement of our approach is visible at all penetration rates in Figure 6.11 and significant performance improvement is achieved at low penetration rates.

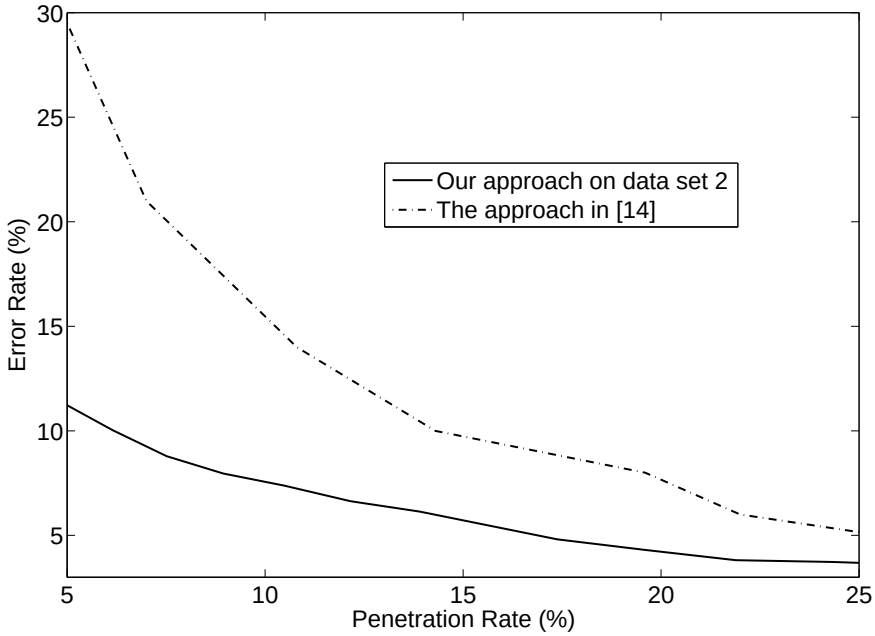


Figure 6.11: Results of fingerprint retrieval in our approach and the approach [14] on data set 2.

A fingerprint retrieval approach based on indexing the triplets of minutia points is proposed in [9]. The result of fingerprint retrieval is further improved by adding

two new features in [96]. The best performed approach [96] is tested on the second 1000 pairs of fingerprints from the NIST database-4. As stated in [96], the retrieval accuracies on the penetration rates of 5%, 10%, 15% and 20% are 83.3%, 88.1%, 91.1% and 92.6%, respectively. Figure 6.12 shows the results reported in [96] and the retrieval results of our proposed approach on the same data set. We can see that our proposed fingerprint retrieval approach outperforms the minutia-based fingerprint indexing approach [96].

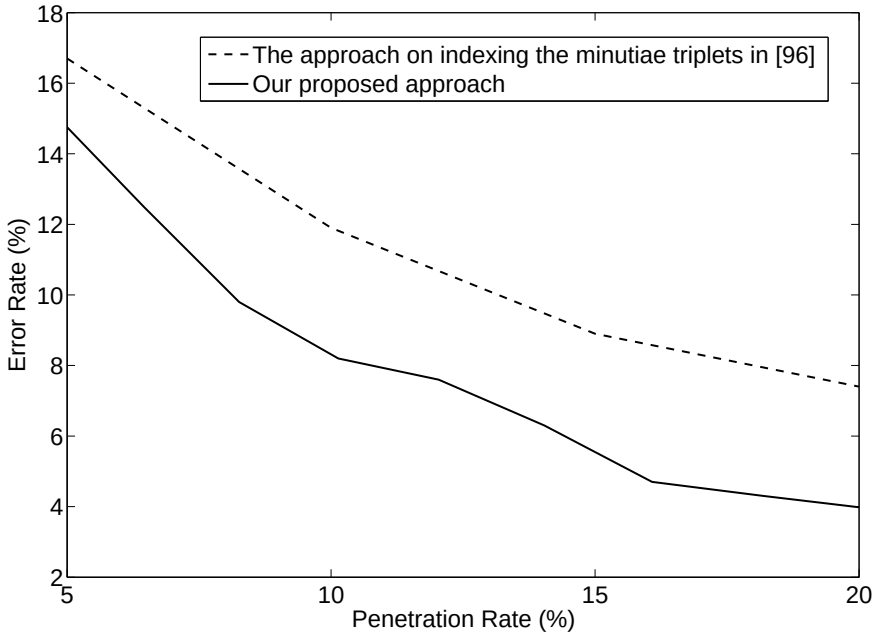


Figure 6.12: Results of fingerprint retrieval reported in [96] and our proposed approach on the same data set.

The exclusive fingerprint classification has been widely studied and many classification results are reported in the literature. However, the query process of exclusive classification is more efficient than that of our approach on the three levels of database search. To be comparable with the exclusive classification in terms of retrieval complexity, we perform our clustering-based fingerprint retrieval by only the first two coarse levels of search in the query process (denoted by "our approach on 2^{nd} level"). Without the continuous classification of the third level search, the retrieval accuracy and efficiency decreases due to the quantization. Our retrieval results on the three levels of hierarchical query process are also provided

for further comparison (denoted by "our approach on 3^{nd} level"). We compare the clustering-based fingerprint retrieval approach with some published approaches of exclusive fingerprint classification according to the error rate at about the same penetration rate. Most published exclusive classification approaches classify fingerprints into 4 (arch and tented arch are merged into arch) or 5 common classes. The penetration rate is 20% if a perfect classifier is applied to partition NIST database-4 into the five common classes which are evenly distributed in the database. It will increase up to 28% if two classes (arch and tented arch) are merged into one. As the fingerprint frequency in each class does not reflect the real distribution, many researchers weight the classification results or constructed a subset (the data set 2) according to the natural fingerprint distribution for testing. For the weighted classification results or on the data set 2, the penetration rates are 29.48% and 29.69% for the 4 and 5 common classes, respectively. To fairly compare our results with those of exclusive classification approaches, our approach retrieves fingerprints at the penetration rates of or slightly smaller than 20%, 28%, 29.48% and 29.69% in the experiments. Table 6.1 shows the error rates of 12 published exclusive classification approaches and our proposed approach on the NIST database-4 ('whole') or its second half . All error rates in the same column are at the same penetration rate (PR). The number of classes labelled by the exclusive classification approaches and whether weighting is used in the error calculation are denoted in the second row of Table 6.1 such as '4 w. class'.

It should be noted that the error of exclusive fingerprint classification is not fully equivalent to the error of fingerprint retrieval for the application to identification. The exclusive classification for fingerprint retrieval is successful only if the query fingerprint and the corresponding one in the database are consistently classified in the same class. There are about 17% of fingerprints in the NIST database-4 labelled as two classes by human experts. The error rates of the exclusive classification approaches [12, 15, 47, 49, 55, 105, 108] are calculated by assuming a correct classification if the fingerprint classifier output is any one of two class hypotheses. This assumption gives lower error rate than that obtained using only one class label.

Table 6.1: Error rates of some exclusive fingerprint classification approaches and our clustering-based fingerprint retrieval approach on the NIST database-4

method year and source	PR=20 5 class	PR=29.5 5 w. class	PR=28 4 class	PR=29.7 4 w. class	test set
Candela et al. 95 [12]	–	–	11.4	6.1	2^{nd} half
Karu & Jain 96 [55]	14.6	11.9	8.6	9.4	whole
Hong & Jain 99 [40]	12.5	10.6	7.7	–	whole
Jain et al. 99 [49]	10	7.0	5.2	–	2^{nd} half
Jain & Minut 02 [47]	–	–	8.8	9.3	whole
Cappelli et al. 99 [14]	–	12.9	–	–	set 2
Cappelli et al. 99 [15]	7.9	6.5	5.5	–	2^{nd} half
Senior 01 [91]	–	–	–	5.1	2^{nd} half
Yao et al. 01 [105]	10.7	9.0	6.9	–	2^{nd} half
Marcialis et al. 01 [74]	12.1	9.6	–	–	2^{nd} half
Zhang & Yan 04 [108]	15.7	–	7.3	–	whole
Park & Park 05 [85]	9.3	–	6.0	–	whole
Tan et al. 05 [97]	8.4	–	6.7	–	whole
Our approach on 2^{nd} level	5.2	3.3	3.5	3.2	whole
Our approach on 2^{nd} level	4.6	2.9	3.2	2.8	set 2
Our approach on 3^{rd} level	4.2	2.9	3.1	2.9	whole

In addition, the error rates of the exclusive classification approaches [49, 74, 105] are obtained at 1.8% rejection rate, which slightly increases their penetration rates. Nevertheless, the experimental results in Table 6.1 demonstrate that the proposed fingerprint retrieval approach achieves lower error rate than the various exclusive classification approaches.

6.5 Summary

The techniques that facilitate an efficient and effective search of fingerprint database in an AFIS have been extensively studied in the past decades. The exclusive fingerprint classification based on the Henry classification scheme cannot sufficiently narrow down the search of database due to the small number of classes and the uneven fingerprint distribution. Although the continuous fingerprint classification avoids the problems of exclusive classification and achieves better retrieval performance, it neglects the similarities among the database templates so that its retrieval performance is also limited. In this chapter, we have developed an clustering-based fingerprint retrieval algorithm which applies a clustering technique to exploit the similarities among the database templates for an efficient fingerprint retrieval. The local ridge orientation field and 1-D dominant ridge distance are used as the representation feature for fingerprint retrieval. In orientation extraction, we propose a nonuniform spacing of fingerprint by a circular tessellation to compute a multi-scale orientation field as the main retrieval feature. The orientation extraction by the nonuniform spacing not only produces more compact feature vector but also achieves more effective fingerprint retrieval than that by the uniform spacing. The 1-D dominant ridge distance is employed as an auxiliary retrieval feature which not only reduces the orientation comparisons in the query process but also consistently improves the retrieval accuracy. A modified K-means clustering is proposed to partition the multi-dimensional orientation feature space into clusters. It outperforms the traditional K-means clustering for the fingerprint

retrieval. Based on the offline database clustering, a hierarchical query processing is proposed to perform the cluster search followed by continuous fingerprint classification of the retrieved clusters. It not only greatly reduces the retrieval complexity but also improves the retrieval accuracy comparing to the continuous fingerprint classification. The extensive experimental studies and comparisons consistently demonstrate the effectiveness and superiority of the clustering-based fingerprint retrieval framework.

Chapter 7

Conclusions and Recommendations

7.1 Conclusions

In this thesis, we have presented some algorithms for fingerprint image processing, fingerprint retrieval and database clustering. The main contribution of this thesis is to present the fingerprint retrieval algorithms based on the continuous classification and database clustering by using the local ridge orientation field and ridge distance of fingerprint as the representation features. These algorithms are proved to be effective, efficient and promising to facilitate the search of database in an AFIS. In developing these fingerprint retrieval algorithms, we have investigated the following problems:

- We have proposed some techniques to robustly and reliably estimate two local parameters of fingerprint: local ridge orientation and local ridge distance, which play important roles for fingerprint analysis and effective fingerprint retrieval. A new orientation smoothing method based on the adaptive neighborhood not only attenuates the noise well but also maintains the orientation localization in the high curvature area. The local ridge distance is estimated

based on more than one x-signatures which results in more robustness to noise and irregular ridge flows than only one x-signature. In addition, an effective method has been proposed to consistently locate a reference point and compute a corresponding reference direction for all types of fingerprints. The reference point is located based on multi-scale analysis of the orientation consistency, while the reference direction is computed by analysis of the orientation differences between 16 redial directions from the reference point and the local ridge orientations along these radii. They can be used for the alignment of fingerprints to achieve the invariance of pose transformation in feature extraction.

- We have developed a fingerprint retrieval algorithm based on continuous classification which achieves good retrieval performance. An orientation vector is constructed from the local ridge orientation field as the main retrieval feature. The dominant ridge distance of fingerprint is proposed as an auxiliary feature. It is more robust to noise than the simple average ridge distance and consistently improves the retrieval accuracy. These two coarse level features are not closely correlated with the minutiae features. Thus, the proposed retrieval approach can be cooperated with the minutiae based matching algorithms to develop an efficient and effective fingerprint identification system. In addition, we have proposed an orientation distance measure based on the inconsistency of orientation differences, which quantifies more effectively the distance between two orientation vectors than the traditional Euclidean and Manhattan distance measures. A regional feature weighting scheme by the entropy also leads to a visible enhancement of the retrieval performance. The suggested retrieval threshold setting slightly improves the retrieval performance comparing to those by the fixed distance and fixed order. This fingerprint retrieval algorithm has been performed on the NIST database-4 and FVC2000 Db2_a and Db3_a. Experimental results and comparisons demonstrate that the proposed approach outperforms some state-of-the-art

approaches of continuous fingerprint classification and minutia-based fingerprint indexing in terms of retrieval efficiency and accuracy.

- Using one prototype to represent each cluster in most partitional clustering algorithms may not adequately model the clusters of complex structure. A multi-prototype clustering algorithm has been developed to discover the clusters of arbitrary shape and size. The squared error clustering is used to produce a number of subclusters represented by prototypes because of its low computational cost and memory space and yet good performance. A separation measure is proposed to evaluate how well two prototypes are separated by a sparse region. Multiple prototypes with small separation are organized to model a given number of clusters in the agglomerative method. New prototypes are iteratively added to improve the poor cluster boundaries resulted by the poor initial settings. The proposed algorithm requires less memory space and computation cost than the commonly used hierarchical clustering algorithms such as *Single-link* and *Complete-link* while preserves much of the speed and efficiency of the squared-error clustering algorithm. Experimental results on both synthetic and real data sets show the effectiveness of the proposed multi-prototype clustering algorithm.

- We have developed a clustering-based fingerprint retrieval algorithm which applies a clustering technique to exploit the similarities among the database templates for an efficient fingerprint retrieval. We propose a nonuniform spacing of fingerprint by a circular tessellation to compute a multi-scale orientation field as the main retrieval feature. The orientation extraction by the nonuniform spacing not only produces more compact feature vector but also achieves better retrieval performance than that by the uniform spacing. The 1-D dominant ridge distance is used as an auxiliary retrieval feature which not only reduces the orientation comparisons in the query process but also consistently improves the retrieval accuracy. A modified K-means clustering is proposed to partition the multi-dimensional orientation feature space into

a number of clusters. It outperforms the traditional K-means clustering for the fingerprint retrieval. Based on the offline database clustering, a hierarchical query processing is proposed to perform the cluster search followed by continuous fingerprint classification of the retrieved clusters. It not only greatly reduces the retrieval complexity but also improves the retrieval accuracy comparing to the continuous fingerprint classification. The extensive experimental studies and comparisons consistently demonstrate the effectiveness and superiority of the clustering-based fingerprint retrieval algorithm.

7.2 Recommendations for Further Research

A lot of works have been done to facilitate an efficient and effective search of database in an AFIS and have achieved good performances. With the growing demands of the efficiency and accuracy of personal identification from various fingerprint databases, there are still some problems available to be solved to make the automatic identification system more efficient and effective in practice. In the following, we will discuss and recommend some research directions which may be further pursued in the future.

- To achieve the invariance of pose transformation, translation and rotation are needed to bring two fingerprints into alignment. Although fingerprint alignment based on a reference point and a reference direction is an efficient solution, the reliable and consistent detection of the reference point and direction is not a easy task for the partial and poor quality fingerprint images. Therefore, it is beneficial to propose some techniques which can capture the global structure information of fingerprint ridge flow patterns invariant to the translation and rotation without alignment. The fingerprint representation based these techniques will be able to work on the partial and poor quality fingerprint images in the fingerprint retrieval.

- It is well known that the feature extraction plays an important role for the

effective fingerprint retrieval and identification. The local parameters: ridge orientation and distance used in our retrieval algorithm are two conventional coarse level features of fingerprint. Other features which can be robustly and reliably estimated and have more representative power of fingerprint can be investigated to further improve the effectiveness of the fingerprint retrieval and identification.

- A good clustering technique which can work well to discover the underlying structure of fingerprint database will be beneficial to facilitate an efficient and effective search of fingerprint database. A modified K-means clustering approach has been proposed to exploit the similarities among the database templates to facilitate an efficient and effective fingerprint retrieval in this thesis. Other clustering techniques can be investigated to further improve the performance of fingerprint retrieval from large database.

- The fingerprint retrieval algorithms proposed in this thesis are a kind of coarse level matching and can be coupled with a fine matching algorithm to obtain an AFIS. Since the retrieval features in our algorithms are not strongly correlated with the minutiae features, the existing 1:1 minutiae based verification (matching) algorithm can be used for the fine matching. It will be attractive and beneficial for the search of fingerprint database by incorporating the fingerprint retrieval with the fine matching algorithm. Some important issues can be further investigated for the effective incorporation. For example, one is that how much the search space of the fine matching is needed in order to benefit the most in the efficiency and accuracy of an AFIS. Another one is to show the correlation between the features used in fingerprint retrieval and fine matching affect the effectiveness of the fingerprint search in an automatic identification system.

Author's Publications

Journal Paper

(1) Manhua Liu, Xudong Jiang and Alex C. Kot, "A Multi-Prototype Clustering Algorithm", *submitted to Pattern Recognition*, November 2006.

(2) Manhua Liu, Xudong Jiang and Alex C. Kot, "Fingerprint Database Filtering Based on Complex Filter Responses", *submitted to International Journal of Pattern Recognition and Artificial Intelligence*, September 2006.

(3) Manhua Liu, Xudong Jiang and Alex C. Kot, "Efficient Fingerprint Search Based on Database Clustering", *Pattern Recognition*, Vol. 40, No. 6, pp. 1793-1803, June 2007.

(4) Xudong Jiang, Manhua Liu and Alex C. Kot, "Fingerprint Retrieval for Identification", *IEEE Transactions on Information Forensics and Security*, Vol. 1, No. 4, pp. 532-542, December 2006.

(5) Manhua Liu, Xudong Jiang and Alex C. Kot, "Fingerprint Reference Point Detection", *EURASIP Journal on Applied Signal Processing*, volume 2005, issue 4, pp. 498-509, April 2005.

Conference Paper

(1) Manhua Liu, Xudong Jiang and Alex C. Kot, "Database Clustering Based on Multi-prototype Representation of Cluster", *Proceedings of International Conference on Multimedia & Expo (ICME'07)*, 2 - 5 July, 2007, Beijing, China, pp. 2198-2201.

(2) Manhua Liu, Xudong Jiang and Alex C. Kot, "Fingerprint Retrieval by Complex Filter Responses", *Proceedings of the 18th International Conference on Pattern Recognition (ICPR)*, 20 - 24 August, 2006, Hong Kong, China, pp. 1042-1045.

(3) Xudong Jiang, Manhua Liu and Alex C. Kot, "Fingerprint Identification with Exclusive and Continuous Classification", *Proceedings of the 1st IEEE Conference on Industrial Electronics and Applications (ICIEA 2006)*, May 24 - 26, 2006, Singapore.

(4) Manhua Liu, Xudong Jiang and Alex C. Kot, "Nonlinear Fingerprint Orientation Smoothing by Median Filter", *Proceedings of the Fifth International Conference on Information, Communications and Signal Processing (ICICS 2005)*, 6-9 December 2005, Bangkok, Thailand, pp. 1439-1443.

(5) Xudong Jiang, Manhua Liu, Alex C. Kot, "Reference point detection for fingerprint recognition", *Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004)*, 2004, London, U.K, pp. 540-543.

(6) Manhua Liu, Xudong Jiang and Alex C. Kot, "Fingerprint reference point detection", *Proceedings of the International Conference on Biometric Authentication (ICBA 2004)*, Hong Kong, China, pp. 272-279.

Bibliography

[1] The science of fingerprints: Classification and uses. Federal bureau of investigation, U.S. Government Printing Office, Watchington D.C., 1984.

[2] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. OPTICS: Ordering points to identify the clustering structure. In *Proceedings of ACM SIGMOD 1999: International Conference on Management of Data*, pages 49–60, 1999.

[3] W. J. Balbler. Embryologic development of epidermal ridges and their configuration. *Birth Defects Original Article Series*, 27(2), 1991.

[4] G. H. Ball and D. J. Hall. ISODATA, a novel method of data analysis and classification. Technical report, Stanford University, CA, 1965.

[5] A. M. Bazen and S. H. Gerez. Segmentation of fingerprint images. In *Proceedings of ProRISC2001, 12th Annual Workshop Circuits, Systems and Signal Processing*, November 2001.

[6] A. M. Bazen and S. H. Gerez. Systematic methods for the computation of the directional fields and singular points of fingerprints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):905–919, July 2002.

[7] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik. Support vector clustering. *Journal of Machine Learning Research*, (2):125–137, 2001.

[8] B. Bhanu and X. Tan. *Computational Algorithms for Fingerprint Recognition*. Kluwer Academic Publishers, ISBN: 1-4020-7651-7, 2003.

[9] B. Bhanu and X. Tan. Fingerprint indexing based on novel features of minutiae triplets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):616–622, May 2003.

[10] J. Bigun. Pattern recognition in images by symmetries and coordinate transformations. *Computer Vision and Image Understanding*, 68(3):290–307, December 1997.

[11] J. D. Boer, A. M. Bazen, and S. H. Gerez. Indexing fingerprint database based on multiple features. In *Proceedings of ProRISC, 12th Annual Workshop on Circuits, Systems and Singnal Processing*, November 2001.

[12] G. T. Candela, P. J. Grother, C. I. Watson, R. A. Wilkinson, and C. L. Wilson. PCASYS - a pattern-level classification automation system for fingerprints. *Technique Report: NIST TR 5647*, August 1995.

[13] R. Cappelli. FVC: Fingerprint verification competitions. In *Proceedings of Second BSI-Symposium on Biometrics 2004*, Darmstadt, Germany.

[14] R. Cappelli, A. Lumini, D. Maio, and D. Maltoni. Fingerprint classification by directional image partitioning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):402–421, May 1999.

[15] R. Cappelli, D. Maio, and D. Maltoni. Fingeprint classification based on multi-space KL. In *Proceedings of Workshop on Automatic Identification Advanced Technologies*, pages 117–120, October 1999.

[16] R. Cappelli, D. Maio, and D. Maltoni. Similarity search using multi-space KL. In *Proceedings of the 10th International Workshop on Database & Expert Systems Applications*, pages 155–160, September 1999.

[17] R. Cappelli, D. Maio, and D. Maltoni. Combining fingeprint classifiers. In *Proceedings of International Workshop on Multiple Classifier Systems (1st)*, pages 351–361, 2000.

[18] R. Cappelli, D. Maio, and D. Maltoni. Indexing fingeprint database for efficient 1:N matching. In *Proceedings of International Conference on Control Automation Robotics and Vision (6th)*, 2000.

[19] R. Cappelli, D. Maio, and D. Maltoni. Multispace KL for pattern representation and classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(9):977–996, September 2001.

[20] R. Cappelli, D. Maio, D. Maltoni, J. L. Wayman, and A. K. Jain. Performance evaluation of fingerprint verification systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):3–18, January 2006.

[21] S. V. Chakravarthy and J. Ghosh. Scale-based clustering using the radial basis function network. *IEEE Transactions on Neural Networks*, 7:1250–1261, 1996.

[22] J.-H. Chang and K.-C. Fan. A new model for fingerprint classification by ridge distribution sequences. *Pattern Recognition*, 35:1209–1223, 2002.

[23] D. Charalampidis. A modified k-means algorithm for circular invariant clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1856–1865, December 2005.

[24] E. Y. Cheu, C. K. Kwoh, and Z. Zhou. On the two-level hybrid clustering algorithm. *The International Conference on Artificial Intelligence in Science and Technology (AISAT 2004)*, pages 138–142, 2004.

[25] B.-H. Cho, J.-S. Kim, J.-H. Bae, I.-G. Bae, and K.-Y. Yoo. Core-based fingerprint image classification. In *Proceedings of 15th International Conference on Pattern Recognition*, volume 2, pages 859–862, Sept 2000.

[26] M. Chong, T. NGEE, J. Liu, and R. Gay. Geometric framework for fingerprint image classification. *Pattern recognition*, 30(9):1475–1488, 1997.

[27] J. Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by twodimensional visual cortical filters. *Journal of the Optical Society of America*, 2(A):1160–1169, 1985.

[28] D. Davies and D. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(4):224–227, April 1979.

[29] M. J. Donahue and S. I. Rokhlin. On the use of level curves in image analysis. *Image Understanding*, 57(2):185–203, March 1993.

[30] G. Drets and H. Liljenstrom. Fingerprint sub-classification and singular point detection. *International Journal of Pattern Recognition and Artificial Intelligence*, 12(4):407–422, June 1998.

[31] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231, 1996.

[32] M. Figueiredo and A. Jain. Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):381–396, March 2002.

[33] A. P. Fitz and R. J. Green. Fingerprint classification using a hexagonal fast fourier transform. *Pattern Recognition*, 29(10):1587–1597, 1996.

[34] A. L. Fred and J. M. Leitao. A new cluster isolation criterion based on dissimilarity increments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):944–958, August 2003.

[35] F. Galton. Finger prints. *McMillan, London.*

[36] R. Germain, A. Califano, and S. Colville. Fingerprint matching using transformation parameters. *IEEE Transactions on Computational Science and Engineering*, 4(4):42–49, April 1997.

[37] S. Guha, R. Rastogi, and K. Shim. CURE: An efficient clustering algorithm for large databases. In *Proceedings of ACM SIGMOD 1998: International Conference on Management of Data*, pages 73–84, 1998.

[38] U. Hallici and G. Ongun. Fingerprint classification through self-organizing feature maps modified to treat uncertainties. *Proceedings of the IEEE*, 84(10):1497–1512, 1996.

[39] E. R. Henry. Classification and use of fingerprint. 1900.

[40] L. Hong and A. K. Jain. Classification of fingerprint images. In *Proceedings of Scandinavian Conference on Image Analysis (11th)*, 1999.

[41] L. Hong, A. K. Jain, and S. Pankanti. Can multibiometrics improve performance. In *Proceedings AutoID'99*, pages 59–64, Summit, NJ, October 1999.

[42] L. Hong, Y. Wan, and A. K. Jain. Fingerprint image enhancement: Algorithm and performance evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):777–789, August 1998.

[43] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, New Jersey 07632, 1988.

[44] A. K. Jain, L. Hong, and R. Bolle. On-line fingerprint verification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):302–314, April 1997.

[45] A. K. Jain, L. Hong, S. Pankanti, and R. Bolle. An identity-authentication system using fingerprints. *Proceedings of the IEEE*, 85(9):1365–1388, September 1997.

[46] A. K. Jain, S. P. L. Hong, and S. Pankanti. Filterbank-based fingerprint matching. *IEEE Transactions on Image Processing*, 9(5):846–859, May 2000.

[47] A. K. Jain and S. Minut. Hirerarchical kernel fitting for fingerprint classification and alignment. In *Proceedings of International Conference on Pattern Recognition (16th)*, volume 2, pages 469–473, 2002.

[48] A. K. Jain, M. Murthy, and P. Flynn. Data clustering: A review. *ACM Computer Surveys*, 31(3):264–323, September 1999.

[49] A. K. Jain, S. Prabhakar, and L. Hong. A multichannel approach to fingerprint classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4):348–359, April 1999.

[50] X. D. Jiang. Fingerprint image ridge frequency estimation by higher order spectrum. In *Proceedings of International Conference on Image Processing*, volume 1, pages 462–465, 2000.

[51] X. D. Jiang. On orientation and anisotropy estimation for online fingerprint authentication. *IEEE Transactions on Signal Processing*, 53(10):4038– 4049, October 2005.

[52] X. D. Jiang and W. Ser. On-line fingerprint template improvement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1121–1126, August 2002.

[53] T. Kamei and M. Mizoguchi. Fingerprint preselection using eigenfeatures. In *Proceedings of IEEE Conference Computer Vision and Pattern Recognition*, pages 918–923, 1998.

[54] M. Kamijo. Fingerprint classification using artificial neural networks: Deriving the classification state. In *Proceedings of the 3rd International Conference on Neural Network*, pages 1932–1937, 1993.

[55] K. Karu and A. K. Jain. Fingerprint classification. *Pattern Recognition*, 29(3):389–404, 1996.

[56] G. Karypis, E.-H. S. Han, and V. Kumar. CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *IEEE Computer*, 32(8):68–75, Auguest 1999.

[57] S. S. Khan and A. Ahmad. Cluster center initialization algorithm for k-means clustering. *Pattern Recognition Letters*, 25:1293–1302, 2004.

[58] B.-G. Kim, H.-J. Kim, and D.-J. Park. New enhancement algorithm for fingerprint images. *Proceedings of the 16th International Conference on Pattern Recognition*, 3:879 –882, August 2002.

[59] B. King. Step-wise clustering procedures. *Journal of the American Statistical Association*, 69:86–101, 1967.

[60] W. M. Koo and A. Kot. Curvature based singular points detection. *3rd International Conference on Audio- and Video-Based Biometric Person Authentication, Lecture Notes in Computer Since (LNCS)*, 2091:229–234, 2001.

[61] R. Kothari and D. Pitts. On finding the number of clusters. *Pattern Recognition Letters*, 20:405–416, 1999.

[62] M. H. Law, M. A. T. Figueiredo, and A. K. Jain. Simultaneous feature selection and clustering using mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1154– 1166, September 2004.

[63] K.-M. Lee and W. N. Street. Cluster-driven refinement for content-based digital image retrieval. *IEEE Transactions on Multimedia*, 6(6):817–827, December 2004.

[64] C.-R. Lin and M.-S. Chen. Combining partitional and hierarchical algorithms for robust and efficient data clustering with cohesion self-merging. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):145– 159, Feberuary 2005.

[65] M. Liu, X. D. Jiang, and A. C. Kot. Fingerprint retrieval by complex filter responses. In *Proceedings of the 18th International Conference on Pattern Recognition*, pages 1042–1045, Hong Kong, China, August 2006.

[66] A. Lumini, D. Maio, and D. Maltoni. Continuous versus exclusive classification for fingerprint retrieval. *Pattern Recognition Letter*, 18(10):1027–1034, October 1997.

[67] D. Maio and D. Maltoni. A structural approach to fingerprint classification. In *Proceedings of 13th International Conference of Pattern recognition*, volume 3, pages 578–585, Vienna, August 1996.

[68] D. Maio and D. Maltoni. Direct gray-scale minutiae detection in fingerprints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1):27–39, January 1997.

[69] D. Maio and D. Maltoni. Ridge-line density estimation in digital images. In *Proceedings of 14th International Conference of Pattern recognition*, volume 1, pages 534–538, Brisbane, Australia, Aug 1998.

[70] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, and A. K. Jain. FVC2000, fingerprint verification competition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):402–412, March 2002.

[71] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, and A. K. Jain. FVC2002: Second fingerprint verification competition. In *Proceedings of 16th International Conference of Pattern recognition*, volume 3, pages 811–814, Quebec City, Canada, Aug 2002.

[72] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, and A. K. Jain. FVC2004: Second fingerprint verification competition. In *Proceedings of International Conference on Biometric Authentication*, pages 1–7, Hong Kong, July 2004.

[73] D. Maltoni, D. Maio, A. K. Jain, and A. Prabhakar. *Handbook of Fingerprint Recognition*. Springer, New York, 2003.

[74] G. L. Marcialis, F. Roli, and P. Frasconi. Fingerprint classification by combination of flat and structural approaches. In *Proceedings of International Conference on Audio- and Video-based Biometric Person Authentication (3rd)*, pages 241–246, 2001.

[75] G. McLachlan and K. Basford. Mixture models: Inference and application to clustering. *New York: Marcel Dekker*, 1988.

[76] J. McQueen. Some methods for classification and analysis of multivariate observations. *Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1:281–297, 1967.

[77] C. J. Merz and P. M. Murphy. UCI repository of machine learning databases. http://www.ics.uci.edu/mlearn/mlrepository.html, Department of Information and Computer Scinence, University of California, 1996.

[78] B. Miller. Vital signs of identity. *IEEE Spectrum*, 31(2):22–30, 1994.

[79] B. Moayer and K. S. Fu. An application of stochastic languages to fingerprint pattern recognition. *Pattern Recognition*, 8:173–179, 1976.

[80] B. Moayer and K. S. Fu. Tree system approach for fingerprint pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(3):223–231, 1980.

[81] M. N. Murty and G. Krishan. A hybrid clustering procedure for concentric and chain-like clusters. *International Journal of Computer and Information Science*, 10(6):397–412, Dec 1981.

[82] K. A. Nagaty. Fingerprints classification using artificial neural networks: A combined structural and statistical approach. *Neural Networks*, 14:1293–1305, 2001.

[83] E. Newham. The biometric report. Technical report, SJB Services, New York, 1995.

[84] C.-H. Park, S.-K. Oh, D.-M. Kwak, B.-S. Kim, Y.-C. Song, and K.-H. Park. A new reference point detection algorithm based on orientation pattern labeling in fingerprint images. In *Proceedings of 1st Iberian Conference on Pattern Recognition and Image Analysis*, volume Puerto de Andratx (Mallorca, Spain), pages 697–703, June 2003.

[85] C. H. Park and H. Park. Fingerprint classification using fast fourier transform and nolinear discriminant analysis. *Pattern Recognition*, 38(4):495–503, April 2005.

[86] M. S. Pattichis, G. Panayi, A. C. Bovik, and S.-P. Hsu. Fingerprint classification using an AM-FM model. *IEEE Transactions on Image Processing*, 10(6):951–954, June 2001.

[87] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257C286, 1989.

[88] N. K. Ratha, S. Chen, and A. Jain. Adaptive flow orientation based feature extraction in fingerprint images. *Pattern Recognition*, 28(11):1657–1672, November 1995.

[89] N. K. Ratha, S. Chen, K. Karu, and A. Jain. A real-time matching system for large fingerprint databases. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):799–813, 1996.

[90] M. Sarkar, B. Yegnanarayana, and D. Khemani. A clustering algorithm using an evolutionary programming-based approach. *Pattern Recognition Letters*, 18:975–986, 1997.

[91] A. Senior. A combination fingerprint classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1165–1174, October 2001.

[92] L. Shen, A. Kot, and W. M. Koo. Quality measures of fingerprint images. In *Proceedings of the 3rd International Conference on Audio- and Video-Based Biometric Person Authentication*, pages 266–271, 2001.

[93] Y. Shi, Y. Song, and A. Zhang. A shrinking-based clustering approach for multidimensional data. *IEEE Transactions on Knowledge and Data Engineering*, 17(10):1389–1403, October 2005.

[94] P. H. A. Sneath and R. R. Sokal. *Numerical Taxonomy*. Freeman, San Francisco,London, 1973.

[95] M.-C. Su and C.-H. Chou. A modified version of the k-means algorithm with a distance based on cluster symmetry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):674–680, June 2001.

[96] X. Tan, B. Bhanu, and Y. Lin. Fingerprint identification: Classification vs. indexing. In *Proceedings of the IEEE Conference on Advanced Video and Siganl Based Surveillance (AVSS'03)*, pages 151–156, Miami, Florida, July 2003.

[97] X. Tan, B. Bhanu, and Y. Lin. Fingerprint classification based on learned features. *IEEE Transaction on System, Man and Cybernetics, Part C, Special issue on Biometrics*, 35(3):287–300, August 2005.

[98] A. Topchy, A. K. Jain, and W. Punch. Clustering ensembles: Models of consensus and weak partitions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1866–1881, December 2005.

[99] E. W. Tyree and J. A. Long. The use of linked line segments for cluster representation and data reduction. *Pattern Recognition Letters*, 20(1):21–29, January 1999.

[100] C. J. Veenman, M. J. Reinders, and E. Backer. A maximum variance clustering algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1273–1280, September 2002.

[101] R. Wang and B. Bhanu. Predicting fingerprint biometrics performance from a small gallery. *Pattern Recognition Letters*, 28(1):40–48, January 2007.

[102] C. I. Watson and C. L. Wilson. NIST special database 4, fingerprint database. Technical report, National Institute of Standards and Technology, March 1992.

[103] C. L. Wilson, J. L. Blue, and O. M. Omidvar. Improving neural network performance for character and fingerprint classification by altering network dynamics. In *World Congress on Neural Networks Proceedings*, volume II, pages 151–158, Washington, D.C., 1995.

[104] C. L. Wilson, G. T. Candela, and C. I. Watson. Neural network fingerprint classification. *Journal of Artificial Neural Networks*, 1(2):203–228, 1994.

[105] Y. Yao, P. Frasconi, and M. Pontil. Fingerprint classification with combination of support vector machine. In *Proceedings of 3rd International Conference on Audio- and Video-based Biometric Person Authentication*, pages 253–258, 2001.

[106] Y. Yao, G. L. Marcialis, M. Pontil, P. Frasconi, and F. Roli. Combining flat and structured representations for fingerprint classification with recursive neural networks and support vector machines. *Pattern Recognition*, 36(2):397–406, 2003.

[107] Q. Zhang, K. Huang, and H. Yan. Fingerprint classification based on extraction and analysis of singularities and pseudoridges. *Pan-Sydney Area Workshop Visual Information Processing (VIP2001)*, 11, 2001.

[108] Q. Zhang and H. Yan. Fingerprint classification based on extraction and analysis of singularities and pseudo ridges. *Pattern Recognition*, 37(11):2233–2243, November 2004.

[109] R. Zhang and Z. Zhang. A clustering based approach to efficient image retrieval. In *Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence*, pages 339–346, Washington DC, November 2002.

[110] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An efficient data clustering method for very large databases. In *Proceedings of ACM SIGMOD 1996: International Conference on Management of Data*, pages 103–114, 1996.