Global and Local Feature-based Transformations for Fingerprint Data Protection

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

Tohari Ahmad

B.Comp.Sc., MIT

School of Computer Science and Information Technology

College of Science, Engineering and Health

RMIT University

Melbourne, Australia

January 2012

Declaration

I certify that except where due acknowledgment has been made, the work is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program; any editorial work, paid or unpaid, carried out by a third party is acknowledged; and, ethics procedures and guidelines have been followed.

Tohari Ahmad

School of Computer Science and Information Technology

RMIT University

January 2012

Acknowledgments

First of all, my sincere gratitude must go to my supervisors: Dr. Fengling Han and Dr. Ron van Schyndel whom I had worked with him in a very short time, at the nearly end of my study. I am very grateful for their assistance and support for both academic and personal. Their guidance and motivation have given a direction to my research. I also would like to express my appreciation to Prof. Jiankun Hu and Mr. Kai Xi from UNSW@ADFA, and Dr. Song Wang from La Trobe University for their feedback to the research. All of these have been very substantial to successfully completing this thesis.

I would like to thank Prof. Zahir Tari, head of Distributed Systems & Networking Discipline, for his advice in finishing the research; and my fellow graduate students, including those who shared the office space with me: Mardi, Naimah, Ayman, Jian, Shaahin, Palka, Peng and Sunidhi.

Last, but not least, I am indebted to my parents, my wife and my sons: Rafif and Akmal for their support. It is really a difficult time being far away from them while completing the research and the thesis. This achievement is dedicated to them.

Credits

Portions of the material in this thesis have previously appeared in the following publications.

Journal:

- Tohari Ahmad, Jiankun Hu and Song Wang. Pair-polar coordinate based cancelable fingerprint templates. *Pattern Recognition*, 44(10-11):2555-2564, 2011. (regular paper) ([4])
- Kai Xi, Tohari Ahmad, Fengling Han and Jiankun Hu. A fingerprint based bio-cryptographic security protocol designed for client/server authentication in mobile computing environment. *Security and Communication Networks*, 4(5):487-499, 2011. ([109])

Conference:

- Tohari Ahmad and Fengling Han. Cartesian and polar transformation-based cancelable fingerprint template. In *The 37th Annual Conference of the IEEE Industrial Electronics Society (IECON 2011)*, pages 373-378, 2011. ([1])
- Yong Feng, Juan Li, Fengling Han and Tohari Ahmad. A Novel Image Encryption Method based on Invertible 3D Maps and its Security Analysis. In *The 37th Annual Conference of the IEEE Industrial Electronics Society (IECON 2011)*, pages 2186-2191, 2011. ([38])
- Tohari Ahmad, Jiankun Hu and Song Wang. String-based cancelable fingerprint templates. In *The 6th IEEE Conference on Industrial Electronics and Applications (ICIEA 2011)*, pages 1028-1033, 2011. ([5])
- Tohari Ahmad and Jiankun Hu. Generating cancelable biometric templates using a projection line. In *The 11th IEEE International Conference on Control Automation Robotics & Vision (ICARCV 2010)*, pages 7-12, 2010. ([2])

- Tohari Ahmad, Jiankun Hu and Song Han. An efficient mobile voting system security scheme based on elliptic curve cryptography. In *The 3rd IEEE International Conference on Network and System Security (NSS 2009)*, pages 474-479, 2009. ([3])

The thesis was written in the `TeXnicCenter` editor on Windows XP, and typeset using the LaTeX 2_ε document preparation system.

Note

Unless otherwise stated, all fractional results have been rounded to the displayed number of decimal figures.

Contents

List of Figures

List of Tables

List of Algorithms

Abstract

Due to its non-shareable characteristic, biometrics has been widely implemented for authenticating users. This characteristic asserts that biometrics meets the non-repudiation requirement which is one of the key factors in the authentication system. Among biometric modalities, such as iris, face and voice, fingerprints have the best capability for satisfying both technical and social aspects of an authentication system. Nevertheless, similar to those other modalities, once the stored fingerprint template has been compromised, the effect will be forever since the fingerprint pattern is permanent. So, a mechanism which can protect this fingerprint pattern is desired. Common cryptographic approaches, however, do not work due to uncertainty in the captured fingerprint image caused by disturbing factors either in the scanner or in the finger itself. While authenticating fingerprints in the plain format is not secure, in the cipher format it is impractical because slightly different inputs result in completely different outputs.

Therefore, a specific transformation mechanism is needed: one which is able to accept similar fingerprints and reject dissimilar fingerprints, while at the same time generating a relatively non-invertible fingerprint template. Most of the existing protection approaches, however, have high error rates which make them inappropriate to implement. The approaches proposed in this thesis are for addressing this problem, in particular.

According to the fingerprint authentication system architecture, the proposed approaches in this thesis comprise three modules: feature transformation, feature representation and fea-

ture comparison (matching). This thesis also evaluates the overall capability of the proposed approaches from various points of view to measure the accuracy, the capability for revoking the template and generating another template, and the capability for scrambling the fingerprint pattern. This measurement includes the accuracy degradation caused by the proposed transformations, particularly the local feature-based transformation.

Firstly, the global feature-based transformation is developed by exploring both the fingerprint singular point (i.e., core point) and minutiae points. In this case, the core point is to be the reference point for transforming minutiae points. A projection line crossing the core point is constructed after plotting the fingerprint image on a Cartesian coordinate space. Minutiae points are projected onto this line according to their coordinate and orientation. The similarity level between the fingerprint template and query, which is specified by the mean absolute error value, determines the decision of whether the template matches to the query. The experimental results show that this approach is able to improve the existing performance, despite the possible limitation (i.e., relying on the core point).

In order to eliminate possible drawbacks of that global feature-based transformation, a different approach: a local-based transformation, is implemented by extracting only minutiae points. This is to explore the relation between minutiae points themselves. Different from the previous approach, the transformation is performed by plotting the fingerprint image on a polar coordinate space. That space is further processed by dividing it into some sectors. Only selected minutiae points are taken to be the transformation input which means reducing the number of minutiae comparison in the matching stage. In general, this proposed approach has been able to eliminate the core-point dependency and, at the same time, to produce only a slightly higher error rate than the previous proposed approach.

To make further improvements, especially in terms of error rate and processing time, the transformation is designed in both Cartesian and polar coordinate spaces. In this proposed approach, the number of minutiae points being used in both fingerprint template and query

construction is specified, and the feature representation being implemented in the previous local feature-based transformation is redefined. Furthermore, the Cartesian space is divided into some quadrant-levels and the polar space is divided into some blocks (sectors-tracks). The experimental result shows that the performance significantly goes up. This approach has been able to take advantages of being core point independent and at the same time generates higher performance than most existing fingerprint template protection approaches.

Chapter 1

Introduction

1.1 Background

Knowledge- and token-based authentication systems have been widely researched and implemented in various applications, from complex systems, such as e-voting [3] to simple ones, such as computer account verification [97]. These two authentication systems have high reliability and accuracy levels so that only when the information provided by the user is exactly the same as what has been stored in the database, the authentication is successful. This simplicity has made it easy for the users to authenticate themselves.

Nevertheless, these two authentication systems have some drawbacks. Firstly, passwords (knowledge-based authentication) and ID-cards (token-based authentication) are easily shared or distributed between users, so, the system is not able to detect whether they are used by the legitimate users. This can result in breaking the non-repudiation property in the authentication process. Secondly, most users hold exactly the same passwords for various applications [75] which makes it easy for the adversary to compromise all applications since he/she only needs to break one password. Moreover, dictionary words or the word *password* itself has been commonly used as a password [75, 46] which actually does not comply with the security standard, especially in terms of length and randomness. A vulnerable situation

caused by the password-related issue was also described by Furnell et al. [39] where there were 34% of users who never changed their passwords at all and only about 46% of users who changed their passwords within six months. Therefore, in multiple applications, passwords can be the weakest point.

On the other hand, the biometrics-based authentication system has advantages over the existing knowledge- and token-based authentication ones. The fact that biometrics employs the human physical or behavioral traits has become its strength since a legitimate user must present when the authentication process is performed. Also, biometrics is not easily shared or distributed [75]. This makes it difficult for the users to repudiate. Furthermore, an advanced technology has been introduced to detect the authenticity of the biometrics, for example, the liveness detection of face [90] and fingerprint [52]. In addition, the combination of biometrics and either passwords or ID-cards in multiple applications potentially increases security.

Conceptually, the biometrics-based authentication system is similar to both knowledge- and token-based ones. It needs to process the biometric data so that it is appropriate (in terms of size, format, etc) to be stored in the database. This biometric data, called biometric template, is to be compared (matched) with the biometric query which is presented by the user in the authentication process.

Among existing biometric modalities, the fingerprint has been the most popular to be used in any authentication or identification system [84]. Fingerprints as identification have a long history [64, 40]. They have been proposed to be a marker of identity by ancient people and have been researched scientifically since sixteenth century. In addition, fingerprints have relatively good characteristics, at least, based on them, users can be distinguished by using their unique fingerprint pattern which will not change over a long period of time (distinctiveness and permanence properties [46, 64]). It should be noted that in rare cases, the fingerprint pattern may change due to some reasons, for example, occupational and aging factors. It is also shown in [74] that the possibility of different individuals being falsely

matched is low.

In fact, each biometric module has different characteristics. Fingerprints, in general, hold properties which are suitable for various aspects required by a biometrics-based authentication system. Other biometric modalities may be better in one aspect but worse in others. For example, the iris is the best in terms of potentiality for circumvention but it is the worst in terms of acceptability [46, 64]. The superiority of fingerprints have made them a potential candidate to be used either in single or in multiple authentication systems. In the latter, fingerprints are combined with the existing knowledge- and token-based systems or other biometric modalities.

Similar to the other biometric modalities, however, the permanence characteristic has made fingerprints problematic. This is because once they are compromised, the effect will be forever. On the other hand, fingerprints are not attack-proof and fully private in spite of their strength. The fact that a copy of fingerprints is easily left in the surface where the finger has contacted with, called a latent print, has made fingerprints vulnerable although this latent print cannot be recognized easily due to its invisibility [111]. Nevertheless, the difficulty of reconstructing a fingerprint from its latent form or of copying and distributing the fingerprints, can be bypassed by directly compromising the stored fingerprint template in the database which results in breaching the security and privacy properties. Therefore, there must be a mechanism to protect the fingerprint data so that in case its template is compromised, the fingerprint data is still safe.

This thesis focuses on how to protect this fingerprint data by transforming it. This transformed data is then the secure template which is stored in the database. In the rest of the thesis, the terms *transformed template* and *secure template* are used interchangeably.

1.2 Research Problems

The obstacles to protect the fingerprint data are mainly caused by the intra-user variability that in every scan, a finger is very likely to produce a similar but non-identical image pattern due to some reasons such as ambient and imaging conditions [75]. This has made the conventional cryptographic algorithms unable to protect the fingerprint data well. This is because, if the fingerprint comparison is performed after the template is decrypted (i.e., in a plain format), then its original data will be disclosed. On the other hand, performing fingerprint comparison in the encrypted structure (i.e., in a cipher format) is very difficult because slightly different fingerprint data can lead to completely different transformed template. It can be inferred that there is a contradiction between the exactness of cryptography and the uncertainty of fingerprints. As a result, using conventional cryptographic or hashing algorithms for securing the fingerprint data is impractical.

Therefore, a mechanism which is able to perform matching in the transformed (secure) domain while at the same time still has a capability for identifying the similarities and differences between fingerprints is highly desirable. This leads to eliminating the need of storing the raw (non-transformed) fingerprint template data such that the privacy of the users is protected. It is worth to note that for the transformation, the key (password) somewhat similar to that of conventional cryptography is required to make the transformed fingerprint template revocable in case it is compromised. So, in this case, the use of a fingerprint-based authentication system is not to replace either the knowledge- or token-based system as such but it is to firstly prevent both legitimate and illegitimate users from violating the non-repudiation property due to the difficulty in copying or distributing the fingerprints, and also to shelve the use of non-standard passwords as it was found in [75, 46, 39].

At the matching process, it is very likely that the inter-user similarity (i.e., different fingers may produce similar fingerprint images) will also arise. In the raw fingerprint domain

matching, the intra- and inter-user issues have made it impossible for an authentication system to achieve perfect authentication results. It is expected that in the transformed fingerprint domain matching, this performance will even decrease; however, this degradation must be kept as low as possible. Thus, there should be an effective approach to distinguish a fingerprint pattern from the others. One possible step is by representing the fingerprint into a form which only contains the unique fingerprint point properties or the relation between those properties themselves. In particular, this can be between the singular point (especially core) and minutiae points or between minutiae points themselves.

Yet, there are at least two issues regarding those points. First, it has been widely known that the core point detection is unstable, particularly in terms of the orientation. In case the core point is employed to be the reference to the transformation, it is predicted that the performance may not be high due to the intra-user issue, despite its simplicity. Second, the number of minutiae points is relatively high, which can be more than a hundred [64, 40]. As each point may not be exactly reproducible, a higher number of minutiae points can lead to a higher intra-user variability. Moreover, a higher number of minutiae points can also make it possible for minutiae points from different fingerprints to overlap. It means that the inter-user similarity can also be higher.

So, in this fingerprint data protection research, some questions have arisen out of those issues, which are: How the fingerprint features are transformed (secured)? What feature representation should be used? If the raw fingerprint data is not stored, how to perform fingerprint matching in the transformed domain? How to minimize intra- and inter-user issues? What is the transformation impact on the performance and how to measure it? What if the secure template is compromised?

In this thesis, there are three approaches taken to address those questions. The first is to develop a global feature-based (i.e., core-based) transformation function which can produce a relatively high performance. The second is to investigate a local feature-based

(non-core-based) transformation function by utilizing only the minutiae information in a polar coordinate system along with designing the matching method according to its feature representation. The third is to extend the second approach by employing both Cartesian and polar coordinate systems for the minutiae transformation in order to obtain a better performance and to reduce the possible drawbacks of the second approach. Overall, the feature representation of each approach is developed in accordance with its transformation characteristics. Some scenarios will also be provided to measure the performance in various cases. In addition, the performance can also be evaluated by comparing it with one without transformation as well as that of other securing techniques.

1.3 Limitations of Existing Solutions

In order to protect the fingerprint data, many transformation functions have been proposed recently. Most of their performance, however, is not satisfying. This is reflected by their error rate value which can be as high as 15% or even more, for instance, 6.8%, 9.5% and 10.3% [56], 13% [113], more than 10% [49], 15% [11] and 16.8% [8]. It is worth mentioning that the data used for the evaluation process may vary among that research; however, those values have indicated the common accuracy of the respective method.

In addition to the performance issue experienced by most existing fingerprint data protection methods, the non-invertibility property can also be another drawback. Various attacking techniques have been implemented, from conventional approaches, such as the brute force attack [88], to mathematical problem solving approaches [76]. It is hard to have a secure system which is able to defend against all types of attacks. A feasible solution is to make the attack as complex as possible to be successful.

Regardless of the assumption being made, some attack techniques have been able to reveal the fingerprint data. For example, Quan et al. [76] are able to recover about 90% of minutiae points which have been protected by using the functional transformation proposed in [80].

Similar to this attacking technique, some others are carried out by assuming that either fully or partially, transformation data have been compromised, including the transformed fingerprint template, the transformation function, as well as transformation parameters and keys. It means that the attack does not consider the difficulty of compromising those transformed template, transformation function, parameters and keys themselves. Therefore, the attacks may work in specific circumstances only. For example, the attack using the method proposed in [76] may not be applicable to other cancelable template schemes, such as that proposed by Lee et al. [57] whose secret keys or parameters are assumed to be highly secure. Nevertheless, fingerprint security and privacy should not rely on this assumption because it is impractical.

1.4 Overview of Contributions

Three transformation approaches are proposed to deal with the problems which have not been fully addressed by the existing fingerprint data protection ones. These approaches are particularly designed to address the performance problem such that they are likely to be able to accurately recognize various fingerprints. Nevertheless, other requirements, such as an ability to revoke the fingerprint template and to generate different templates are also considered.

In general, the contributions of the thesis can be highlighted as follows:

- The local feature-based authentication approach is designed such that it is able to work on either a secure mode or an insecure mode (without protection). This characteristic makes it flexible to use.

- Global and local feature-based authentication approaches are proposed with respect to their own characteristics. These have given options for the different implementation environments.

- The representation of extracted fingerprint features which is invariant to translation

and rotation is developed along with a matching algorithm which is able to accept intra-user variability and reject inter-user similarity. Those characteristics are to be reflected by the results of the experiments. The global feature representation is constructed by exploring minutia properties referring to the core point whilst the local feature representation is structured by examining both the relation among minutiae and the properties of the minutiae themselves.

1.5 Thesis Organization

The remaining chapters in this thesis are structured as follows.

Chapter 2: Fingerprint Biometrics and its Vulnerabilities investigates basic and advanced fingerprint characteristics and the general concept of the fingerprint-based authentication system. This includes fingerprint classes, singularities, features and their representation, as well as the terminologies used in the authentication system. A detailed literature survey of fingerprint template vulnerabilities, including attack models, and existing techniques implemented to protect fingerprint data are presented. This literature review reveals the key research problems and possible approaches to address them.

Chapter 3: Transformed Fingerprint Template Environment explains how the experiments for this research are carried out. Terminologies used in the evaluation process are defined. In addition, this chapter also provides the scenarios to be implemented in the experiments. These reflect real world cases, for example, situations in which a secure template is safe or lost. Scenarios to evaluate the ability of each proposed approach to revoke both the key and the fingerprint template, and to evaluate the effect of the transformation on the fingerprint authentication system performance are also provided.

Chapter 4: Projection-based Transformation (Approach 1) presents a global feature-based (core-based) transformation function. The chapter starts with an examination of the limitations of the current approaches, including that of singular point detection, followed by description of the proposed one. Then, the results of experiments on fingerprint features with different parameter settings are plotted on graphs and analyzed according to the previous designed scenarios.

Chapter 5: Pair-polar Coordinate-based Transformation (Approach 2) proposes a new local feature-based transformation function that employs a polar coordinate system. This is implemented by following the analysis of recent related approaches provided in the beginning of the chapter. The proposed approach which consists of three parts: minutiae points selection, minutiae points transformation and fingerprint matching, is described. Here, the description of minutiae points transformation also covers that of minutiae points representation and template generation. The non-transformed fingerprint template is also evaluated in order to find out the performance degradation caused by the transformation. In addition, the separability of genuine-imposter fingerprint distribution is also analyzed.

Chapter 6: Cartesian and Polar Coordinate-based Transformation (Approach 3) introduces an improved local feature-based transformation function in both Cartesian and polar coordinate systems. The minutiae points selection and fingerprint matching techniques used in Chapter 5 are also implemented in this approach. The transformation function comprising two steps, namely, the rotation of the minutiae points in the Cartesian system, and the rotation and translation minutiae points in the polar system, is explained. This is followed by an analysis of the experimental results performance comparison with existing approaches, including the two new approaches investigated in the previous chapters.

Chapter 7: Conclusion provides the summary of the thesis contributions, and discusses the possibility of further research to increase the performance of transformation functions.

Chapter 2

Fingerprint Biometrics and its Vulnerabilities

This chapter surveys recent research in fingerprint authentication systems and possible vulnerabilities of stored fingerprint templates to security and privacy breaching. Existing approaches to address these vulnerability issues are also surveyed. In addition, this chapter investigates advances in fingerprint concepts, which form the foundation of fingerprint-based authentication systems.

This chapter is structured as follows. Section 2.1 describes fingerprint biometrics and its properties. This section consists of a survey on fingerprint feature classification and its characteristics. Section 2.2 presents an architecture of fingerprint-based authentication systems. This includes explanation of processes in these authentication systems and factors that influences those processes: fingerprints image capture, fingerprints feature representations and fingerprint matching. Section 2.3 depicts potential threats against the fingerprint template stored in a database. Some fingerprint template-attacking models are investigated. Section 2.4 surveys two state-of-the-art fingerprint data protection approaches: the biocryptosystem and feature transformation. Finally, Section 2.5 summarizes the key information and

highlights the direction of the research.

2.1 Fingerprint Biometrics

A fingerprint is the result of regeneration of fingertip epidermis [64] that constructs *ridges* and *valleys* [12]. In fingerprint images, ridges and valleys are represented by dark and bright areas, respectively. Because ridges increase the friction between the fingers and surfaces of other objects, they are useful for grip as well as maximizing the capability for recognizing different textures [111].

The two main characteristics that make fingerprints useful for authentication are permanence and uniqueness (in [74], the terms *persistence* and *individuality* are used, respectively). Permanence refers to the stability of ridge patterns, which fingerprints of individuals do not change throughout life; while uniqueness refers to the singularity of fingerprint patterns - there is no exact same ridge pattern on any other finger. While the permanence of fingerprints can be proved by intensively analyzing the ridge pattern of individual fingers, the uniqueness of fingerprints is not easy to validate. This is because fingerprints originating from different fingers may have similar appearances, in some cases. Nevertheless, they are very likely to be different if the analysis is undertaken using high resolution images [74]. This, however, requires high cost. Therefore, simpler fingerprint features are needed as a reference to recognize and distinguish among fingerprints.

There are some features that can be extracted from fingerprints according to ridge configuration. Based on their scale, those fingerprint features are analyzed and categorized into three different levels [64, 103]: global level (level one), local level (level two) and very fine level (level three).

The global level classifies fingerprints based on the general ridge or valley pattern which establishes distinctive configuration. This classification leads to three general classes, those are loop, arch and whorl. These classes can be further divided into left loop, right loop,

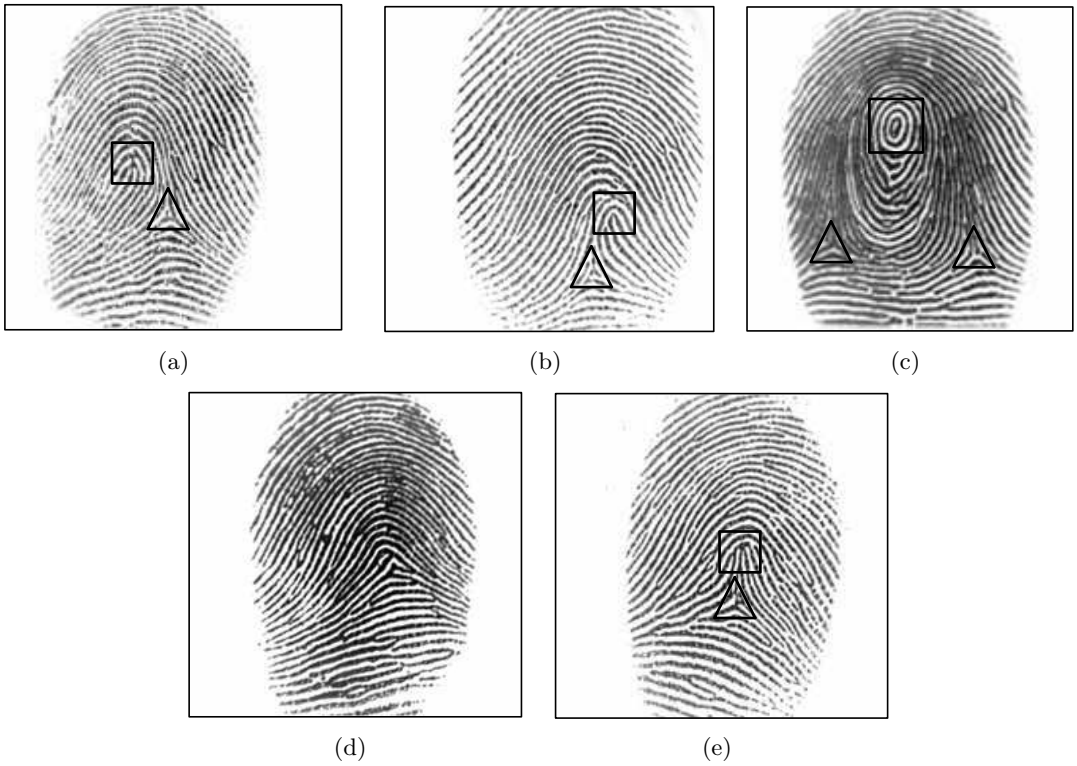
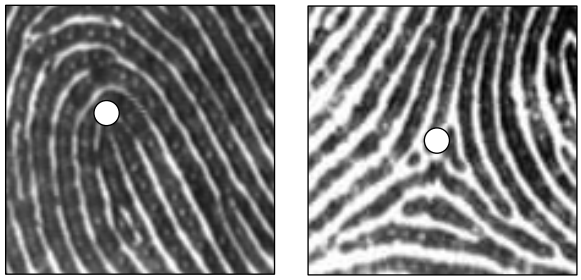


Figure 2.1: Ridge characteristics derived from the global level, where (□) *and* (△) *represent loop and delta, respectively (a) left loop (b) right loop (c) whorl (d) arch (e) tented arch (fingerprint images are taken from FVC2002 [61]).*

arch, tented arch and whorl (presented in Figure 2.1). Wilson et al. [107] summarized their a priori distribution probability to be 0.037, 0.338, 0.317, 0.029 and 0.279 for arch, left loop, right loop, tented arch and whorl, respectively. Based on this distribution, it can be inferred that those classes do not have uniform distribution and about 93.4% of fingerprints fall into one of only threee classes: left loop, right loop or whorl [111].

In most fingerprints, there is a small number of unique regions, called *singularities*, whose location determines the corresponding class of the fingerprint. Therefore, these singular regions are usually used for fingerprint classification purposes. The unique region can fall into either loop, delta or whorl, as depicted in Figure 2.1. In this case, whorl can also be defined as two loops which front each other. In more detail, it is found that left loop, right



(a) (b)

Figure 2.2: Singularity regions and points (a) core (b) delta.

loop and tented arch classes have one loop and one delta; the arch class has neither loop nor delta; the whorl class has one whorl and two deltas [116]. Each singular region contains a singular point which acts as the unique feature in the respective region. This can be a core or delta point. The former is located at the peak of the inner most ridge which can be viewed as the center of the fingerprint; while the latter is located at the divergent point of the ridges, which constructs a "triangle", as depicted in Figure 2.2. Not all singular points, however, can be easily identified, particularly those in the arch class.

By relying only on the information generated from fingerprint classes and singular points themselves, however, the authentication process does not work well. This is because, in spite of their fast detection, both of them provide only a small amount of distinctive fingerprint information. Further, it is difficult to capture a fixed location for singular points in all fingerprint images [80, 64]. This means that the location of singular points is difficult to accurately detect. Therefore, this feature level cannot be used in a fingerprint-based authentication system without combining it with other feature levels.

At the local level, a fingerprint is described based on its ridge points (points constructed by ridge lines). These ridge points indicate local fingerprint features, which are more stable and discriminable than singular points of the global level. Based on this stability and discriminability, ridge points can be a promising tool in a fingerprint authentication system. Moreover, most commercial authentication systems and forensic experts have adopted this

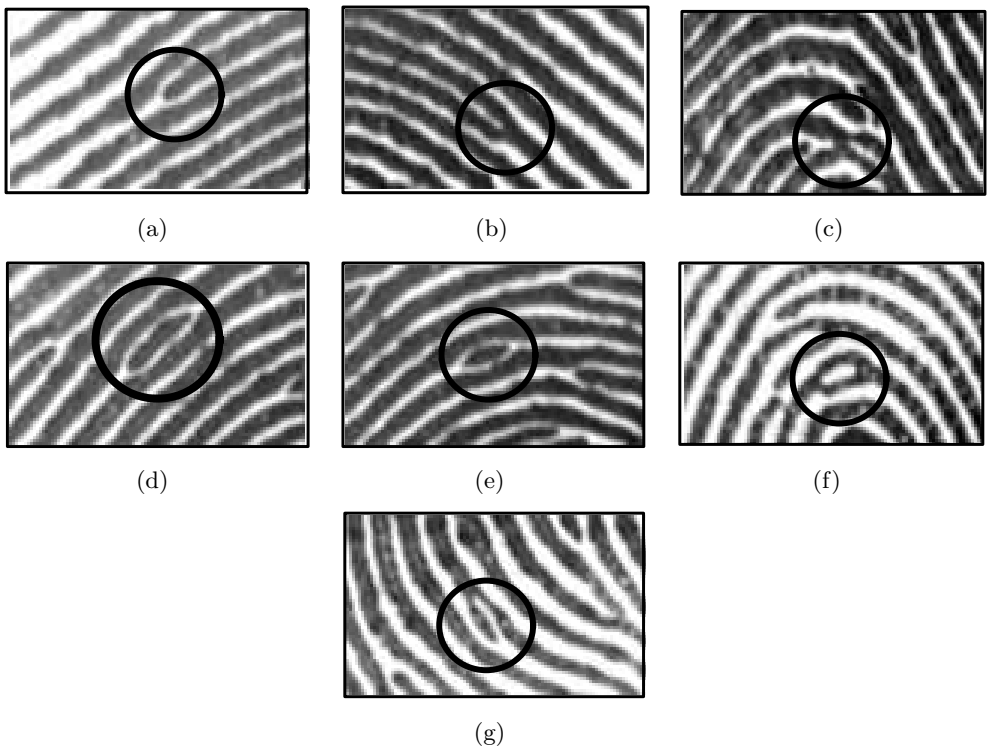


Figure 2.3: Ridge characteristics derived from the local level (a) ridge ending (b) bifurcation (c) crossover (d) independent (e) island (f) lake (g) spur (fingerprint images are taken from FVC2002 [61]).

feature [74]. Likewise, academic research has also used it in many fingerprint authentication systems, such as in [56, 4, 1].

The frequently used feature generated from this local level is *minutiae*, which is the representation of how the ridge ends. Some minutiae classification methods have been introduced that result in different minutiae numbers and types, such as that in [72, 9]. Examples of possible minutiae types are depicted in Figure 2.3. A relatively simpler method than others is proposed by ANSI/NIST-ITL [9] which classifies the minutiae types into four categories. The first is type A (ridge endings), where the ridge ends suddenly without diverging into other ridges. The length of this ridge must be greater than its width. The second is type B (ridge bifurcations), where the ridge line diverges into two ridges. The third is type C

(compound), which can be either crossover or trifurcation. The former is the point where two ridges intersect each other while the latter is the point where a ridge diverges into three ridges. The last is type D (undetermined), which consists of all other ridge types that cannot be classified into those three categories. Among those proposed minutiae types, ridge endings and ridge bifurcations are the most commonly used in fingerprint-based authentication systems [64]. A good quality fingerprint image usually has 20 - 70 minutiae points [46], and it can be up to more than one hundred [40]. The example of how both global and local level features can be generated from a single fingerprint is depicted in Figure 2.4. The generated global level features are a core point, loop and delta; while those of the local level are ridge ending, bifurcation, independent and spur (independent and spur are commonly included in bifurcation).

By considering that in the fingerprint authentication, 12 matched minutiae points of two fingerprints are enough to determine that both are derived from the same finger, Pankanti et al. [74] have investigated the possibility of a fingerprint matching to another randomly chosen fingerprint. They found that the possibility of 12 out of 36 minutiae points in a fingerprint match to 12 minutiae points of other fingerprints containing also 36 minutiae points are 6.10×10^{-8}. By using the same token, Zhu et al. [117] conducted an experiment on a fingerprint with 46 minutiae points. They obtained 2.25×10^{-6} of possibilities. These results show that there is still a possible error in the minutiae-based authentication system, even though those error rates are small. In other words, despite the uniqueness of fingerprints, it is still difficult to achieve error-free fingerprint-based authentication using these local level features alone.

The very fine level is the highest level at which the detail of ridges, such as the width, pores, scars and creases is analyzed. This level generates more distinctive features than the other levels and is useful in evaluating fingerprints in specific conditions, such as latent prints [64]. Other fingerprint features have been outlined in [74] which include minutiae

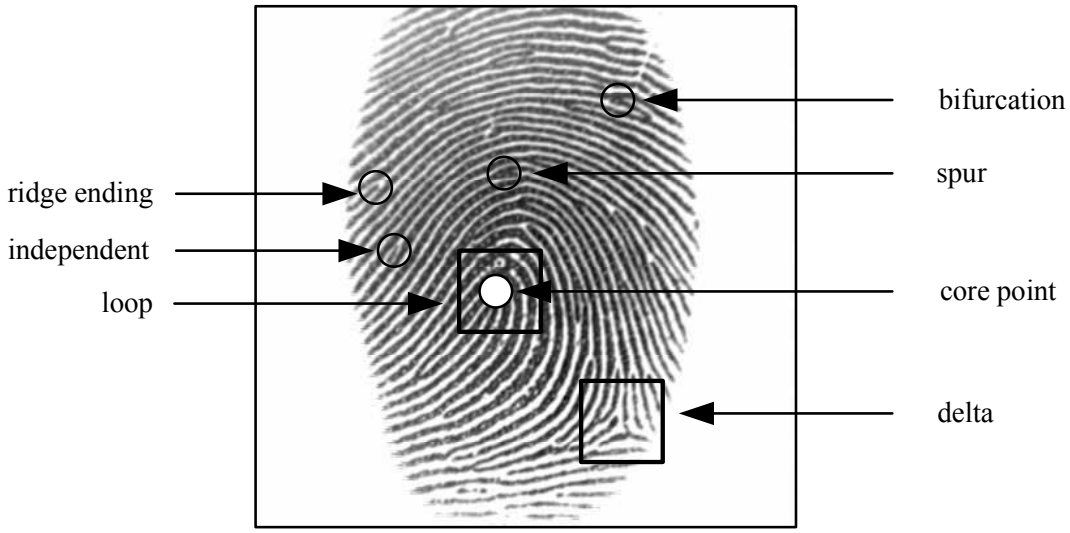
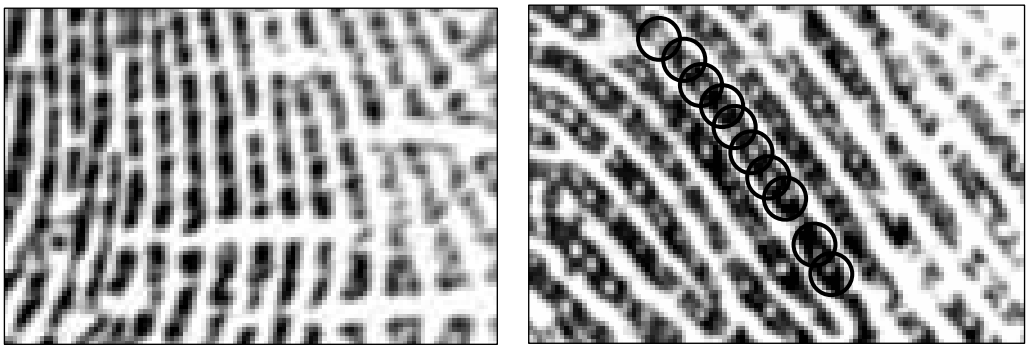


Figure 2.4: Both global and local features in a fingerprint (the fingerprint image is taken from FVC2002[61]).

distribution, minutiae area and fingerprint quality. Yet, in practice, not all of these features are utilized. The selection of which features should be used depends on many factors, such as the purpose of matching and the required accuracy level. In general, the disadvantage of this level is that it requires a good quality of high resolution fingerprint images, for example, 1000 dpi [64], which may make this level features inefficient to apply. Consequently, these features are rarely implemented in fingerprint-based authentication systems [111]. Examples of features that can be generated from this level are depicted in Figure 2.5, where scars and pores are analyzed.

The best authentication performance may be achieved by combining all three level features; however, this will result in a higher cost. A possible solution is to combine global and local level features as has been implemented in [5, 113, 8]. Wang [103] outlines key functions of the three levels: (i) global level features are appropriate for pattern description, for instance, classification and indexing; (ii) local level features are used in the matching process, especially in the general environment; (iii) very fine level features are useful in a



(a) (b)

Figure 2.5: Ridge characteristics derived from the very fine level (a) scars in a fingeprrint (b) sweat pores in a ridge line, represented by circles.

specific matching process, for example, in cases where the available fingerprint information is minimal.

2.2 Fingerprint Authentication System

Based on their characteristics, fingerprints have been implemented in various authentication systems. In general, these authentication systems are equivalent to both knowledge- and token-based authentication ones that all of them require to store data (template) in a database. In the fingerprint-based authentication systems, however, there should be additional modules to process the data. Despite their complexity, fingerprint authentication systems have made it easy for users since they do not have to worry about forgetting the password or losing the key which may happen in knowledge- and token-based authentication systems, respectively.

The fingerprint authentication system consists of two steps [4]:

1. Enrollment: constructing and storing a fingerprint template by firstly extracting the fingerprint features and/or other related data.

2. Recognition: measuring the similarity (difference) level between the fingerprint query

and the stored fingerprint template based on the specified features and/or other related data.

Based on their purpose, fingerprint authentication systems can be grouped into two categories: fingerprint identification and fingerprint verification [64]. A fingerprint identification system (FIS) recognizes a user by comparing the fingerprint query with all enrolled fingerprint templates in the database; therefore, there is a one-to-many comparison. A fingerprint verification system (FVS) recognizes a user by comparing the fingerprint query with an enrolled fingerprint template according to the identity (or other user's properties) he/she claims to be; therefore, there is a one-to-one comparison. Sometimes, the term *authentication* also refers to the *verification*.

As explained by Maltoni et al. [64], overall entities (modules) involved in a fingerprint authentication system can be depicted in Figure 2.6. The modules and their use are described as follows:

- Fingerprint capture (scanning) module is to convert fingerprint biometrics into digital representation (image). It means that this module converts a three-dimensional object to a two-dimensional one.

- Feature extraction module is to generate a set of features from the fingerprint image. This set (represented by feature set 1) contains compact and good (e.g., stable) fingerprint data.

- Feature representation module is to generate a fingerprint template based on the feature set 1. The template (feature set 2) contains specific data which is sent to the data storage.

- Matching module is to compare the fingerprint template with the fingerprint query. This module decides whether these two fingerprints are from the same finger.

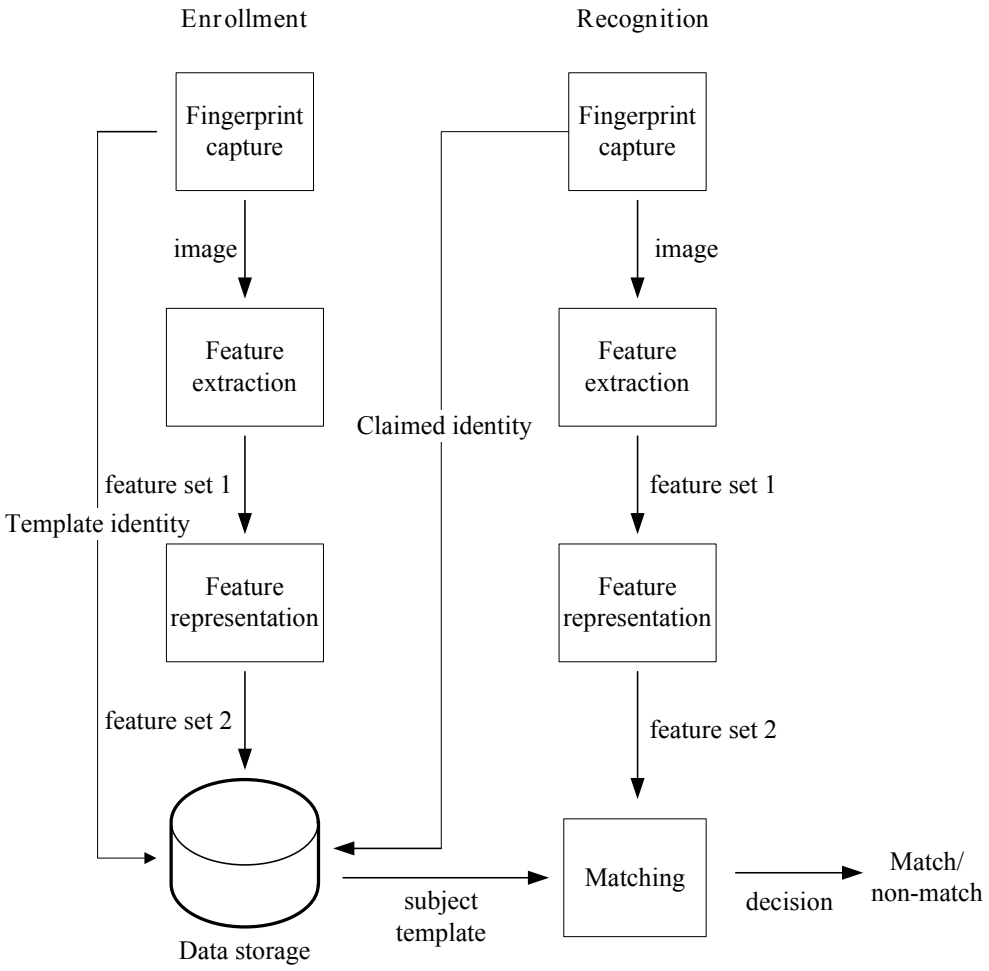


Figure 2.6: The architecture of fingerprint authentication systems, which comprises some processing steps (modules) (adapted from [64]).

The details of these modules are described in the following sections.

2.2.1 Fingerprint Capture and Uncertainty

There have been various advanced fingerprint scanner technologies introduced that have a capability for capturing the detail of fingerprints [20, 43] and even equipped with spoofing protection [93]. Furthermore, the specification of fingerprint image qualities has also been defined [7]. In spite of these advanced technologies and the permanency of fingerprint patterns

itself, however, the result of fingerprint scanning is still not so stable. It is very unlikely to reproduce exactly the same fingerprint images from a finger. In other words, there is still uncertainty in generating fingerprint images from a finger. This is because many factors influence the fingerprint image capturing process [64, 44], which include:

- Impression (pressure) of fingers to a scanner.
- The position of fingers on a scanner.
- Different shapes due to some reasons, such as drought or wetness.
- Cleanliness of a scanner.

These all factors (condition of both fingers and scanners) are very likely to vary from time to time. This inevitable condition causes the amount of finger surfaces contacting with the scanner is also varied. Consequently, exactly the same fingerprint images are difficult to obtain. Moreover, due to the fact that fingerprint scanning is actually mapping a three-dimensional finger onto a two-dimensional image, there must be non-linear deformation introduced [44, 111]. This has an effect on accuracy of the subsequent fingerprint image processing.

This uncertainty can be represented in the forms of insertion (a minutia point appears only in the query), deletion (a minutia point appears only in the template), reordering (a minutia point appears in both the template and the query but their location is not identical) or combination of them as illustrated in Figure 2.7. More than that, not only the location of the minutiae points in the fingerprint image can change but also the type of minutiae points itself [74]. Eventually, all of these variations lead to the intra-class and inter-class problems as depicted in Figure 2.8.

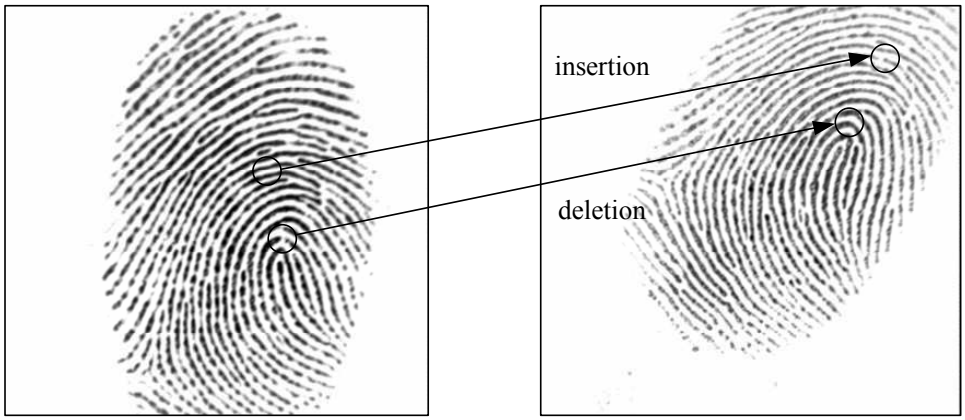


Figure 2.7: A fingerprint template and a fingerprint query that suffer from insertion and deletion of minutiae points (fingerprint images are adapted from FVC2002Db2a [61]).

2.2.2 Feature Extraction

Fingerprint images captured by scanners should be enhanced before being extracted. This is because their quality is varied (Figure 2.9) due to possible noises appearing on them. It is difficult to extract adequate features (e.g., core and minutiae points) commonly used in the authentication system if the image quality is low. This condition is very likely to cause an error in determining feature location (coordinate) and feature orientation, or even cause failure to detect those features. Both of these cases lead to missing the singularity of the fingerprints. Consequently, the overall authentication result is affected. Some enhancement approaches have been introduced, for example, by using a log-Gabor filter [101], a short-time Fourier transform (STFT) [25] and a multi-scale operator [71]. These have been able to recover some missing ridge lines; nevertheless, the quality of enhanced fingerprint images still depends on their pre-processed condition.

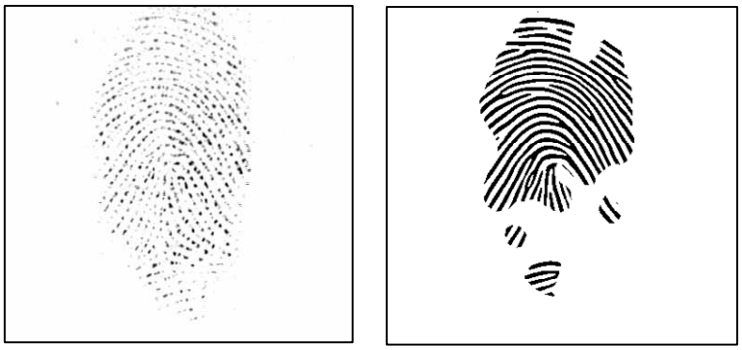
In order to accurately detect and extract the features, the enhanced fingerprint images are usually further pre-processed by converting them into binary ones (called the binarization step) and, in turn, by reducing the width of ridge lines to one pixel (called the thinning step) on which the minutiae and core point detection is performed. There have been some

Figure 2.8: The effect of uncertainty (e.g., minutiae insertion, deletion, reordering) of the fingerprint images. Fingerprints (a) and (b) which are originating from the same finger may have differences whilst fingerprints (c) and (d) which are originating from different fingers may have similarities (fingerprint images are taken from FVC2002 [61]).

binarization and thinning approaches introduced, such as one in [115] and in [47], respectively. The skeleton images (resulted from the thinning step) can produce more accurate minutiae points (in terms of coordinate and orientation). Nevertheless, as depicted in Figure 2.10, the enhancement and the subsequent processing steps are still unable to refine certain parts of fingerprints due to the low quality of the corresponding fingerprint images. It is argued that the image pre-processing stage should not be implemented because of some reasons, for example [64], (i) binarization and thinning steps remove important information, which may increase the number of spurious minutiae points; (ii) binarization and thinning steps require

(a) (b) (c)

Figure 2.9: The fingerprint images (a) a high quality fingerprint (b) and (c) low quality fingerprints caused by noises, which result in inaccurate or missing some features (fingerprint images are taken from FVC2002 [61]).



(a) (b)

Figure 2.10: Fingerprint image enhancement (a) before enhancement (b) after enhancement (fingerprint images are taken from FVC2002 [61]).

additional time to obtain the features.

2.2.3 Feature Representations

In this thesis, fingerprint feature representation is defined as a set of objects extracted from the fingerprint features, for example, properties of points (type and orientation) and the relation between points. Here, the *point* refers to both singular and minutiae points such that the relation between core and minutiae points or between minutiae points themselves can be covered by this definition. Ideally, a feature representation should only contain the invariant

and unique objects so that the effect of translation, rotation and noises can be minimized. As the result, the effect of fingerprint intra-user variability and inter-user similarity can also be minimized. This means that the form of how fingerprint features be represented highly influences matching performances. In addition, the representation of features should be generated, stored, and matched easily [64].

As has been previously described, fingerprint features can be categorized into three levels. By considering the strength and the weakness of features in each of those levels along with fingerprint authentication environments, this thesis explores both global and local features, whose common representations are provided as follows.

In a Cartesian coordinate space, a minutia point can be represented as a triplet [10], which consists of its relative position to the core point. Suppose T, m_i, and n are the fingerprint template, the minutia point i^{th} and the total number of minutiae points, respectively; (x_i, y_i) and α_i are the coordinate (i.e., abscissa, ordinate) and orientation of the minutiae m_i with respect to those of core point respectively. In this case, the core point is the center of the coordinate space and its orientation is aligned with $x-$axis. The template of a fingerprint can be denoted as:

$$\left.\begin{aligned} m_i &= (x_i, y_i, \alpha_i) \\ T &= m_i \end{aligned}\right\} \tag{2.1}$$

where $1 \leq i \leq n, (x_i, y_i) \in \mathbb{R}$ and $0 \leq \alpha_i < 360^o$. The definition of this triplet structure is depicted in Figure 2.11(a). This feature representation has been implemented in some research, for instance, in [77, 80, 8, 92, 5].

Similar to that of the Cartesian coordinate space, a minutia point in a polar one can also be represented by a triplet [10]. In this case, the template of a fingerprint is formulated as:

$$\left.\begin{aligned} m_i &= (r_i, \theta_i, \alpha_i) \\ T &= m_i \end{aligned}\right\} \tag{2.2}$$

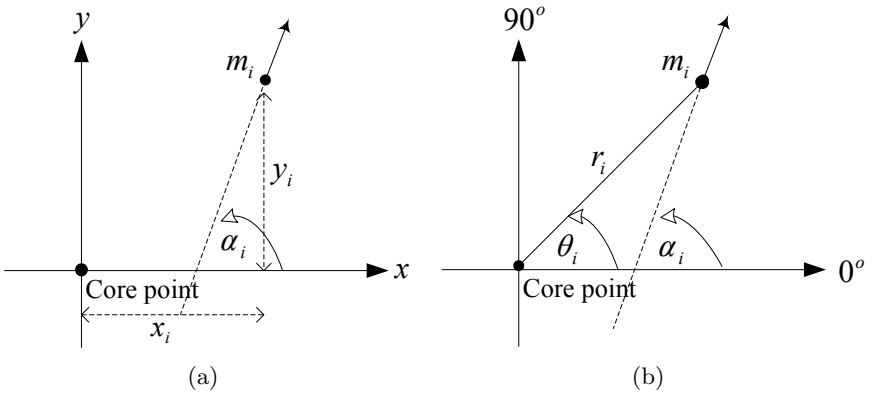


Figure 2.11: Feature representation (a) cartesian (b) polar.

where r_i and θ_i are the radial and angular coordinates of minutia m_i, respectively; $r_i \in \mathbb{R}$ and $0 \leq \theta_i, \alpha_i < 360^o, 1 \leq i \leq n$. Here, the (r_i, θ_i) coordinate is also relative to the core point whose orientation is aligned with the 0^o, as illustrated in Figure 2.11(b).

By implementing the similar concepts of those global feature-based representations, local features may be superior due to their stability and reliability. A local feature-based representation can be constructed by assigning each minutia point m_i a descriptor, which contains information of its k-nearest neighboring minutiae points. Suppose r_{i_j} is the distance between a reference minutia m_i and its neighboring minutia m_j; and θ_{i_j} is the angle between the orientation of m_i (x-axis) and the edge connecting m_i to m_j in counterclockwise. The template can be depicted as:

$$\left.\begin{aligned} m_i &= \{(r_{i\_1}, \theta_{i\_1}), (r_{i\_2}, \theta_{i\_2}), ..., (r_{i\_k}, \theta_{i\_k})\} \\ T &= m_i \end{aligned}\right\} \tag{2.3}$$

where $1 \leq i \leq n, k \in \mathbb{N}^*$. The definition of this local feature-based representation is provided in Figure 2.12(a). Other local feature-based representation techniques have also been introduced, for example, one which explores the relation between three minutiae points forming a triangle [49, 34, 14].

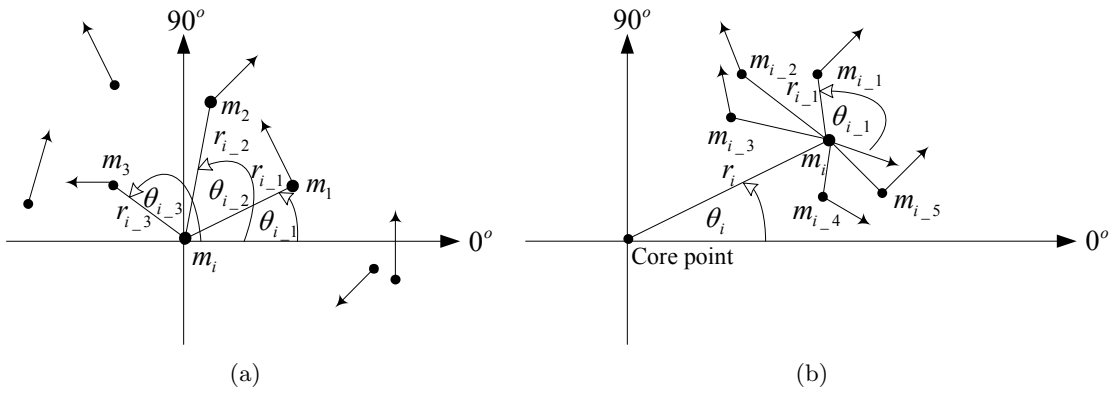


Figure 2.12: Feature representation (a) k-Nearest Neighbor with k = 3 (b) combination of global and local structure.

In addition, global and local feature-based representations can be combined such that a minutia point is described based on its relative location to both the core point and neighboring minutiae points. In [11], this combination is represented in a polar coordinate space. At the global level, every minutia point is described by its polar position, i.e. every minutia point has (r_i, θ_i), while at the local level, every point is described by its five nearest neighboring minutiae points, as depicted in Figure 2.12(b). This structure can be denoted in Equation 2.4.

$$\left.\begin{aligned} &(F_{global})_i = (r_i, \theta_i) \\ &(F_{local})_i = \{(r_{i\_1}, \theta_{i\_1}), (r_{i\_2}, \theta_{i\_2}), ..., (r_{i\_5}, \theta_{i\_5})\} \\ &m_i = ((F_{global})_i, (F_{local})_i) \\ &T = m_i \end{aligned}\right\} \tag{2.4}$$

where $1 \leq i \leq n$. Another possible combination is proposed by Yang et al. [113], where global features are represented by their respective position in a Cartesian space while local features are represented by minutia triangular structures. Overall, combining the global and local feature-based representations not only taking the strength of each of them but also their weakness.

2.2.4 Feature Comparison (Matching)

As the final stage in the fingerprint authentication system, the matching module decides whether the fingerprint query is either accepted or rejected. The decision is taken by comparing the fingerprint template, which has been stored in the database, with the fingerprint query being processed. Matching can be performed effectively if the fingerprint template and query images largely overlap so that a large number of features can be extracted and compared maximally from those two images. This gives the matching module enough information before taking the decision, whether they are similar (in case genuine users) or dissimilar (in case imposters). This is, however, not easy because many factors influence the process of capturing the fingerprint images (see Section 2.2.1).

According to Maltoni et al. [64], fingerprint matching approaches can be categorized into three groups: correlation-based, minutiae-based (local) and non-minutiae-based (global) matching. Additionally, Ceguerra and Koprinska [22] and Yang et al. [114] have implemented neural network-based matching, which requires a higher number of data training than the others.

Correlation-based matching is performed by lying a fingerprint image over the other. Then, by using a correlation filter, the relation between their corresponding pixels is analyzed. Therefore, in this approach, matching is done by directly correlating the images.

The minutiae-based matching approach, also called local matching, has been widely implemented in authentication systems due to its reliability and accuracy. Here, after being extracted from both template and query images and appropriately represented in the specified format, the minutiae points are compared such that the fingerprint template and the fingerprint query have as many as possible matching minutia pairs. Different from this, non-minutia-based matching (global matching) is performed after aligning the global features.

There is a trade-off among those matching approaches in terms of distinctiveness, com-

plexity and distortion-tolerance [103, 64]. In general, minutiae-based matching is considered to be more accurate than the others, even though the actual authentication performance also depends on how the minutiae points are represented, in particular what invariants extracted from the features, as discussed in Section 2.2.3. In certain environments, such as in a poor-quality fingerprint image, however, minutiae-based matching is inferior due to a low number of minutiae points can be extracted. In this case, global matching is more appropriate even though the performance may not be high [103].

2.3 Template Vulnerabilities

Like knowledge- and token-based authentication systems, fingerprint-based authentication systems are also vulnerable to attacks. Depending on its purpose, the attack may exploit various vulnerable points in the system. For describing this possible attack, some threat models have been proposed recently. Jain et al. [46] explain the fish-bone model by identifying some possible system attacks and analyzing the cause-effect relation of them. They found four factors: administration, intrinsic, biometric overtness and non-secure infrastructure that potentially caused intrusion and denial of service to the system. Cukic and Bartlow [29] provide an attack tree model. This explains that the attacks can be carried out in several stages. Relating to Figure 2.6, Ratha et al. [79] show that there are eight points to which the attack can be launched. These points comprise the process in each entity (module) and process between those entities. Roberts [81] argues that there are three dimensions of the attacks on the systems: threat agents, threat vectors and system vulnerabilities, of which needs different countermeasures. In addition, there are six defensive measures which were also explained: input device protection, input data protection, system data protection, data storage, system tamper resistance and secure communication.

From those threat models, it can be inferred that compromising the fingerprint template stored in the database is a common threat that greatly affects the security and privacy of

the users. For example, the compromised fingerprint data can be used by the adversary to compromise other systems or applications where the users have registered.

Furthermore, the stored fingerprint templates usually contain sensitive fingerprint information, which can be used to reconstruct the fingerprint patterns. Ross et al. [83] have depicted that there are three types of fingerprint information can be inferred from the template: the orientation field, the class or the type, and the friction ridge structure. Based on this information leakage, the raw (original) fingerprint data can be revealed illegitimately. This is common cases that some research has demonstrated it. For example, Feng et al. [38] successfully reconstructed a fingerprint with only a small number of spurious minutiae. In this case, a gray scale image was resulted from the phase image. Also, by using a standard template, Cappelli et al. [19] were also able to generate a fingerprint image. Recently, Wang and Hu [104] have proposed an advanced technique to reproduce a fingerprint image based on a partial image. In spite of the fact that it is still not easy to deceive the image-based authentication system by using only minutia information [68], this research has proven that the stored fingerprint template in the database is vulnerable to attacks, which can lead to disclosing the fingerprint data of the users.

2.4 Protected Fingerprint Template

As described by Jain et al. [45], there are two different approaches can be taken in protecting the fingerprint data. The first is the fingerprint cryptosystem-based approach which deals with fingerprint key binding and key generation. The second is the feature transformation-based approach which deals with invertible and non-invertible transformations. Alternatively, those two approaches can also be combined so that one approach will minimize the weakness of the other. However, the complexity may increase accordingly.

2.4.1 Fingerprint Cryptosystems

Fingerprint cryptosystem [100] or fingerprint encryption [21] is actually to secure a cryptographic key by using fingerprint features. So, its purpose is different from that of fingerprint data protection designs. However, fingerprint cryptosystems can also be applied for protecting fingerprint data because their authentication process is conducted by comparing the cryptographic key, which is obtained from a secure version of fingerprint data. It means that the original fingerprint data does not need to be stored in the database.

In its implementation, fingerprint cryptosystem usually needs to provide additional non-secret fingerprint information (called helper data) in order to extract the key. Therefore, this helper data should not contain too much fingerprint information so that a raw fingerprint cannot be recovered given helper data alone. Since it needs to extract an exact binary key string from a fingerprint, designing fingerprint cryptosystem may be more difficult than that of fingerprint authentication system itself, especially in overcoming the intra-user variation. Fingerprint cryptosystem can be developed by using either a key binding approach, which an independent key is binded to fingerprint features, or a key generation approach, which a key is directly extracted from the fingerprint itself.

Key Binding

Binding biometrics (i.e., fingerprints) with a key was initially proposed by using the fuzzy commitment scheme [51]. The overall process is carried out by combining a codeword w generated from an error correcting code (ECC) with the biometrics B in the enrollment step; and generating the codeword w' resulted from a query biometrics B'. If only the difference level between w and w' is less than the threshold, then it is inferred that $B = B'$. This concept has been implemented in some biometric modalities, such as fingerprints [110] and iris [41]. However, a difficulty may arise in the ECC implementation due to a possible high error rate caused by the intra-user variability [91].

Juels and Sudan [50] further improved that concept by developing the fuzzy vault scheme. Like the previous approach, the biometrics B hides the secret key κ. This is performed by projecting each element of B onto the polynomial P. On the other hand, P is constructed according to κ. The projection produces a set of points S_1 which is then combined with chaff points (S_2), where $S_2 \notin P$. The combination of S_1 and S_2 is to be the vault V, which is used by the biometric query B' as the helper data for retrieving κ. The reconstruction of κ is done according to the coefficients of P.

This fuzzy vault scheme has also been implemented in various biometric modalities, such as iris [58] and face [37]. In further development, helper data is automatically generated from the fingerprint and is added to the fuzzy fingerprint vault [99]. This helper data is extracted according to the orientation field information. In order to achieve a better performance, Nandakumar et al. [65] refined this fuzzy fingerprint vault by applying a minutia matcher using multiple impressions and decoding. Still working on the fingerprint biometrics, Xi and Hu [108] proposed the fuzzy composite feature-based fingerprint vault scheme, which matching between minutiae points is performed in two layers. This is intended to create multiple matching processes such that not only the corresponding minutiae points to be compared but also their neighboring minutiae points.

Yet, this fuzzy vault scheme suffers from a key inversion attack [85]. Assuming that helper data and the key are compromised, the original fingerprint information can be recovered. Furthermore, fuzzy vault may also have problems in revoking the vault; and cross-matching the vault among databases [45]. This is because multiple vaults generated from the same biometric data will project the genuine points identically.

Key Generation

Different from that of key binding scheme, a key and helper data (called a sketch) in the key generation scheme are directly generated from biometric data. Dodis et al. [31; 32] proposed

the fuzzy extractor scheme, which consists of secure sketch and strong extractor modules to derive a secret key from biometric data. This key generation process can be depicted in Figure 2.13. In the registration step, the secure sketch module produces a sketch s based on the biometric data B and seed r; while the strong extractor module produces the key κ based on B and seed x. The value of $P = \{x, s\}$ is public and is used in the authentication step. Next, the biometric query B' and the sketch s are to recover the original biometrics B such that the value κ can be generated. Here, κ can be recovered if only $B \approx B'$ given $P = \{x, s\}$. In order to measure the similarity level between the template and the query, they proposed three different metrics: Hamming distance, set difference and edit distance metrics. Overall, the main goals of the fuzzy extractor scheme are to achieve [30]:

- Uniformity (generating κ from B and x): This is based on fact that biometric data is not distributed uniformly.
- Fuzziness (reproducing κ if only $B \approx B'$): Due to the intra-user variation and inter-user similarity problems, the same fingers should generate similar fingerprints while different fingers should generate dissimilar fingerprints. This also works on other biometric modalities.
- Robustness: Different sketches s produce different keys κ, so that $\kappa' \Leftarrow (s', B')$ and $\kappa \Leftarrow (s, B')$

Arakala et al. [11] implemented this fuzzy extractor concept in the fingerprint modality by using descriptors. In this case, each descriptor is constructed by global and local features presented in a polar coordinate space. Similar to other global feature-based implementation, the singular point (i.e., core point) is to be the center of the space. Global features are represented by their position relative to the center; while local features are represented by 5-nearest neighboring minutiae points centering on the point being compared. Secure sketches

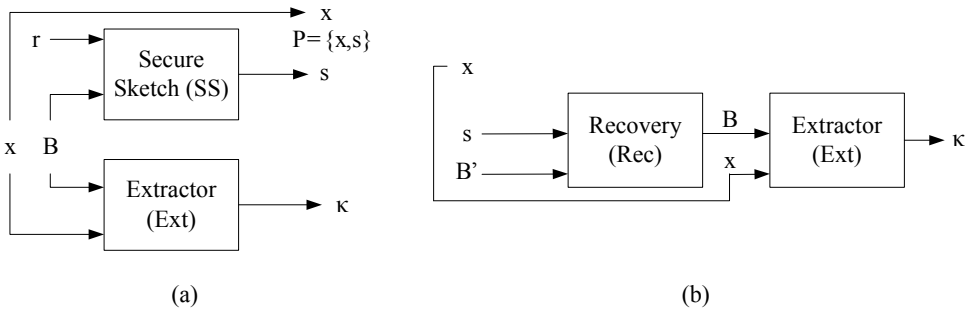


Figure 2.13: The fuzzy extractor scheme [31, 32, 33] (a) generating template (b) generating query (adapted from [31, 32]).

are developed by employing PinSketch [31]. For measuring the distance between the template and the query, and correcting the minutiae points error location, they implemented the set different metric and the BCH error correcting code [106], respectively. The experimental results, however, show that the error rate is relatively high (10% and 15% for private and public databases, respectively). It is believed that these error rates are affected by non-accurate singular point detection and extraction processes.

Despite its excellent concept, fuzzy extractor [31, 32] may not preserve the privacy. It is because the identity of users can be revealed once the sketch is compromised, regardless of the strength of the generated key [59]. In order to alleviate this problem, Li et al. [59] developed an asymmetric fingerprint representation of fuzzy extractor by combining the proposed concepts in [31, 32] and [23].

Overall, the sketch developed in the fuzzy extractor [31, 32] suffers from information leakage [17]. In particular, this happens in multiple uses of the sketches, assuming that they are generated from the same fingerprint data. Furthermore, the result of the experiment conducted by Simoens et al. [89] supports this argument. Some possible issues of secure sketches, which are used in various fuzzy extractor implementations, have also been analyzed in [91].

Different from the previous approaches, Shibata et al. [87] used a statistical analysis approach to extract a bit string. This is performed by dividing the fingerprint region into some subregions. In this approach, there are two fingerprint feature definitions, which are constructed based on: (i) the ridge orientation of these subregions; (ii) the ridge orientation of neighboring subregions. Nevertheless, some weakness is identified from this approach, for example, (i) raw fingerprint data (x and y minutia coordinates) is stored in the database for the alignment purpose, which make this approach inappropriate to implement for a fingerprint data protection scheme; (ii) similar to [59], at the registration phase, each user is required to provide multiple fingerprint templates (each finger is scanned several times) to produce stable features. This is very likely to reduces the user acceptance of the system.

2.4.2 Feature Transformations

Based on their invertibility, feature transformation approaches can be classified into two categories: invertible (two-way) and non-invertible (one-way). Both of them transform a fingerprint in such a way that the result still has a correlating pattern to its non-transformed version but difficult enough to recover this non-transformed template given only its transformed version. The transformation itself can be carried out either in the signal or in the feature domain.

The invertible approach usually has low error rates, is easy to revoke and is able to generate different transformed fingerprint templates given fingerprint data and several different keys [63, 45]. However, it much relies on security of the transformation functions, the transformed templates and the keys. In case all of them are compromised, the fingerprint data can be revealed because of its invertibility property.

Like the previous approach, protecting fingerprint data by using existing cryptographic algorithms (e.g., a symmetric key algorithm) also requires secure key storages. The fingerprint data is stored in a database after being encrypted (this is to be a template); therefore, as

long as the key is secure, the fingerprint data is also secure, assuming that the cryptographic algorithm is unbreakable. The authentication is carried out by comparing the fingerprint query with the decrypted fingerprint template. At this state, it can be inferred that the fingerprint data is not secure anymore. As described in Section 1.2, doing authentication in a cipher format is impossible because of the uncertainty of fingerprints. Therefore, existing cryptographic algorithms are inappropriate to protect fingerprint data.

Based on this invertibility characteristic, a fingerprint data protection technique, called BioHashing [94, 95], was proposed (in [45], it is also called *salting*). This technique provides protection for fingerprint data against illegitimate reconstruction by implementing random multi-space quantization (RMQ). This comprises several steps: (i) biometric projection by using a linear transformation, such as FDA [13] or PCA [98]; (ii) projection on subspaces; (iii) quantization. BioHashing has been able to achieve the ideal condition (zero EER). Nevertheless, it is found that this ideal error rate is mainly caused by an unrealistic assumption that each user holds a unique seed value to generate a tokenized random number [54, 24]. This assumption indicates that the number is highly secure and must never be compromised. Although it is possible, in many cases this assumption may not work. Since the introduction of this technique, some variants of BioHashing have been proposed, such as that in [67]. In general, BioHashing is more applicable to use by using the singular point in the image-based transformation [94, 66]. Similar approaches have been implemented in different biometric modalities, such as in palm print [28] and iris [27].

Another approach was proposed by Lee and Kim [56], which generates bit strings by projecting minutiae points on a three-dimensional array (3D array). In this design, a minutia is alternately chosen to be the mapping reference to the other minutiae points, and a bit string is generated according to the number of the corresponding mapped points in each 3D cell. It is performed such that an element is set to 1 if the respective cell contains more than one point, and set to 0 if otherwise. Using different keys for each transformation, they were

able to achieve a low error level. However, when the same keys were applied, the error rate increased significantly.

The non-invertible transformation function, which is also known as the cancelable biometric template function, was introduced by Ratha et al. [78] and Bolle et al. [15]. Different from its invertible counterpart, this non-invertible function does not much rely the security on the secrecy of keys itself. The difficulty of reversing the transformation function has been the major protection for the fingerprint data. In case the adversary has been able to compromise either the keys or the transformed template, both of them are just revoked and the new ones are easily re-generated. Thus, the raw fingerprint data is still safe.

Theoretically, this approach requires not only non-invertibility but also discriminability at the same time. It means that the transformation function has to be non-invertible and is able to recognize and distinguish the transformed templates. Those two factors refer to the fingerprint security and accuracy, respectively. In practice, it is difficult for a transformation function to meet those two requirements simultaneously [45]. As the result, a transformation function may offer the security but at the same time suffers from much performance degradation, and vice versa. For example, while providing non-invertibility, the approach proposed in [8] and [49] obtained a significant error rate increase that the error rate difference between before and after the transformation is more than 10%. Further research on trade-off between security and performance (accuracy) factors is conducted in [35].

The transformation function itself may use either texture information, such as in [26] or geometric (minutiae points) information, such as in [77, 80]. The latter has been a *de-facto* standard in the fingerprint applications [34] and has been implemented in most of the fingerprint-based authentication systems [74].

Similar to those of non-transformed authentication system, feature representation and matching modules of the transformed authentication system can be developed based on either global or local features. According to the research conducted by Thomas et al. [96],

it can be inferred that, in general, the global feature-based system is more appropriate to resource-constrained devices[1] than the local feature one due to its simplicity. On the other hand, local feature-based system shows a better performance in terms of non-invertibility and discriminability.

Generally speaking, several characteristics that should be satisfied in designing a fingerprint transformation function are outlined as follows [64, 53, 8, 80]:

- The function can be used to transform two feature sets B and B', which are derived from the same fingerprint. In this case, B and B' may be positioned (e.g., rotated, translated) in the same workspace.
- Given a fingerprint and a transformation function, different transformed templates can be generated by using different keys. The new template must be different enough from the old one, so that, there is no cross-matching between them.
- It is infeasible to reconstruct the original fingerprint data given the transformed template.
- Considering that the transformation function gives rise to decreasing the performance, the degradation must be kept as low as possible.
- Similar transformed fingerprint templates are successfully authenticated if only their difference is relatively small.

2.5 Summary

This chapter has described the general concept of both fingerprints and fingerprint authentication systems from various points of view. Fingerprints have two main characteristics which make them appropriate to be an authentication tool, those are uniqueness and permanence.

[1] A resource-constrained device is any device whose resources are constrained intentionally [55].

There are three feature levels at which the fingerprints can be classified. The first is the global level, which provides the global ridge patterns. The second is the local level, which describes fingerprints based on the local information (minutiae points) and the third is the very fine level, which gives detail information of ridge composition. Overall, all those feature levels can be used in an authentication system, depending on the authentication purpose and the availability of fingerprint features.

Due to some factors, a finger is very unlikely to reproduce exact fingerprint images. This intra-user variability problem has meant fingerprint-based authentication systems are not able to completely eliminate the authentication error. In other words, the ideal (error-free) condition may not be achieved. Other than this image capturing problem, the overall authentication result is also influenced by other factors: feature extraction, feature representation and matching modules. Therefore, in order to have a good result, the captured fingerprint images should be in a high quality so that the features can be completely extracted, appropriately represented and accurately matched.

Despite some advantages offered by the use of fingerprint-based authentication systems, the raw fingerprint template stored in the database of the system is vulnerable to attacks which threat the security and privacy of the users. This is because the permanence of fingerprint patterns also means that once a fingerprint is compromised, the effect will be forever. Therefore, the fingerprint data must be protected and in case the fingerprint template is compromised, the original data is still safe. Many securing approaches have been introduced to protect the fingerprint data. Among them, feature transformation, which does not have to correlate the template with public helper data, can be a potential solution. Most of the proposed feature transformation functions, however, have a performance drawback which is reflected by their error rate.

In this thesis, fingerprint data protection is performed by transforming minutiae points based on either global or local data. In particular, the geometrical function is examined.

The relation between fingerprint features is also investigated to further determine a stable feature representation in either Cartesian, polar or both coordinate spaces. Accordingly, a matching module is developed to deliver the final result.

Chapter 3

Transformed Fingerprint Template Environment

This chapter presents the scope of the thesis and depicts how the experiments are carried out. This includes testing approaches, which are used for evaluating the performances of proposed schemes as well as some terminologies used for representing both environment and the results of the experiments.

This chapter is structured as follows. Section 3.1 describes the focus of the thesis. This is to detail the research position in the existing fingerprint-based authentication system architecture. Section 3.2 specifies the environment in which the experiments are conducted, including what data is used in the experiments, and how the experimental results are analyzed. More detail scenarios of the experiments are provided in Section 3.3. Finally, the summary of this chapter is provided in Section 3.4.

3.1 Research Focus

General architecture of fingerprint authentication systems has been previously explained in Section 2.2, whose detail is depicted in Figure 2.6. It shows that fingerprint data is processed

in several steps before being stored in the database. Those steps are: data captured, feature extraction and feature representation. In this thesis, that architecture is modified so that the data sent to the database is in a secure (transformed) version. This is carried out by inserting a feature transformation module and redesigning the feature representation module. Accordingly, the feature comparison (matching) module is also redesigned (see Figure 3.1). These three modules are the focus of this thesis.

The feature representation module explores the properties of both singular points (i.e., core point) and minutiae points, or relation among them to have as stable as possible invariants against translation, rotation or any other ambient and imaging conditions. Here, a feature representation is defined such that it is appropriate for the other modules. Likewise, the matching module will also employ that representation in deciding the authentication result.

The feature transformation module itself is designed so that it meets not only non-invertibility but also discriminability (refer to Section 2.4.2). The ideal characteristic to achieve is that similar (unnecessary to be identical) transformed fingerprint data derived from same inputs (fingers) produces exactly the same outputs, whereas dissimilar transformed data derived from different inputs (fingers) produces different outputs. Let a and a' be respectively the fingerprint template and the fingerprint query; and $\Gamma(a, \kappa_1, \kappa_2, ..., \kappa_q)$ be the template a which has been transformed by using a function Γ and keys $\kappa_1, \kappa_2, ..., \kappa_q$, where q is the number of keys being used in the transformation. The results of a transformed fingerprint authentication process can be formulated as follows:

$$\left.\begin{array}{ll} \Gamma(a, \kappa_1, \kappa_2, ..., \kappa_q) = \Gamma(a', \kappa_1, \kappa_2, ..., \kappa_q) & \text{if } a \sim a' \\ \Gamma(a, \kappa_1, \kappa_2, ..., \kappa_q) \neq \Gamma(a', \kappa_1, \kappa_2, ..., \kappa_q) & \text{if } a \nsim a' \end{array}\right\} \tag{3.1}$$

In this thesis, a key is defined as a value that is used for securing (transforming) the fingerprint data such that some identical data transformed by different keys results in different

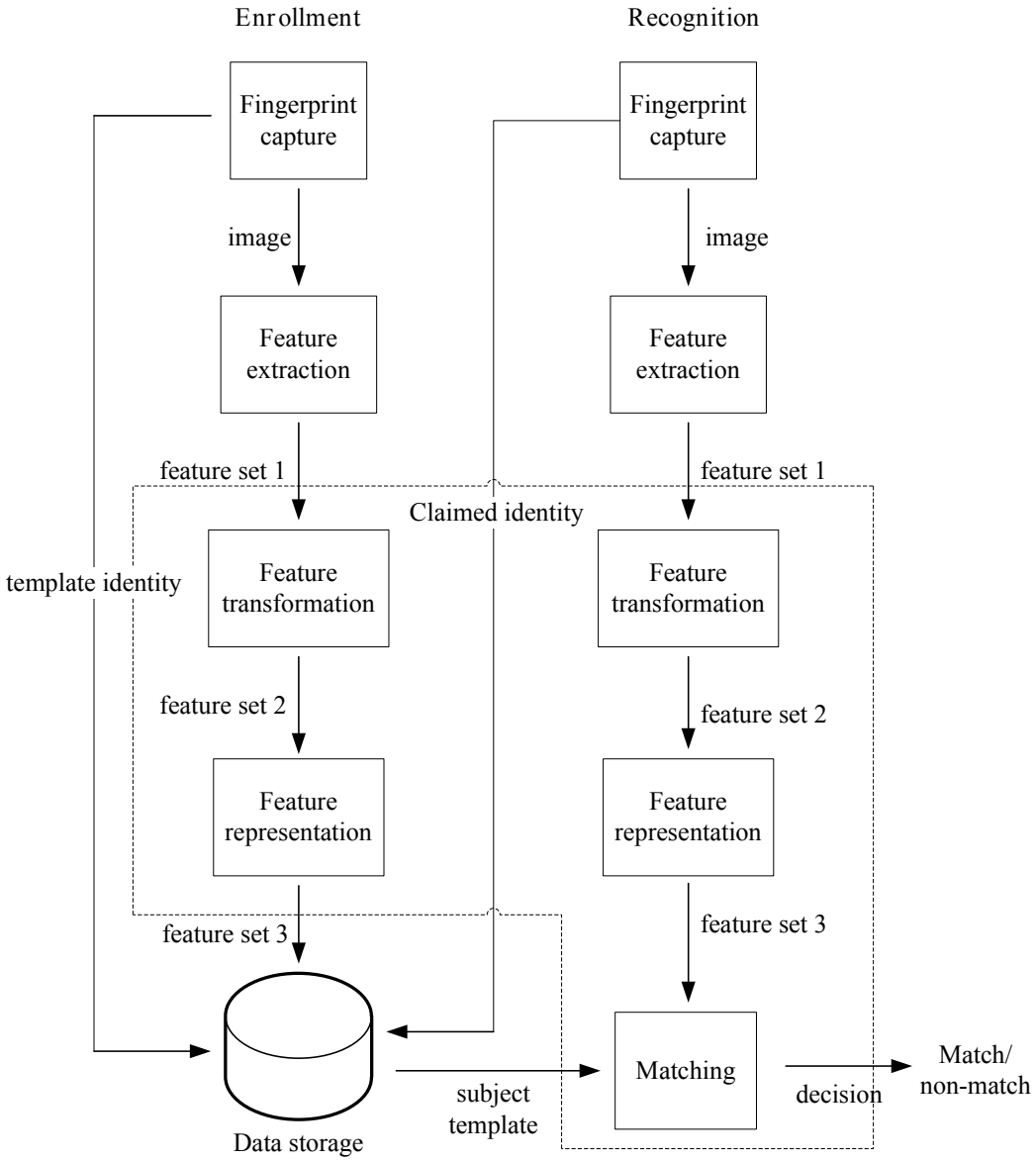


Figure 3.1: The transformed fingerprint authentication system. In this system, the feature transformation module is inserted and both the feature representation and matching modules are redesigned.

secure templates. In addition, the transformation may also require a parameter, represented by ρ (to be used in the transformation function proposed in Chapter 4). Different from a key whose value can be dynamically changed for every fingerprint template-query pair

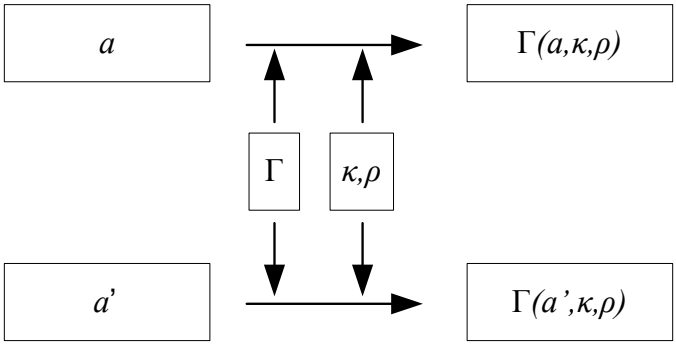


Figure 3.2: The process of fingerprint transformation by using both key κ and parameter ρ.

transformation, a parameter is statically set to a certain value.

The general concept of this fingerprint transformation process can be illustrated in Figure 3.2. This shows that in the recognition step, the fingerprint query is transformed by using the same function, keys and parameters which have been used to transform the corresponding template.

In the rest of the thesis, the fingerprint transformation and the minutia transformation refer to different definitions. The former means the transformation of all minutiae points of a fingerprint while the latter means the transformation of an individual minutia point of a fingerprint.

3.2 Design of the Experiment

3.2.1 Error Rates

Due to the intra-class variability and inter-class similarity factors, match and non-match decisions produced by a matching module cannot be free from errors. The match or non-match decision itself is generated by comparing the matching score with the specified threshold, whose result, in turn, reflects the difference (or similarity) level between the fingerprint template and the fingerprint query. In addition, the appropriate selection of threshold values

determine the overall accuracy (performance) of the authentication system.

In this thesis, the accuracy of the transformed fingerprint authentication system is measured by calculating the number of *false reject* and *false accept* decisions. The former occurs when the genuine query is incorrectly rejected (actually, both the fingerprint template and the fingerprint query are derived from the same fingers) whilst the latter happens when the non-genuine query is incorrectly accepted (actually, the fingerprint template and the fingerprint query are derived from different fingers). In general, false reject is inversely proportional to *genuine accept* (the case when the fingerprint query is correctly accepted), whilst false accept is inversely proportional to *genuine reject* (the case when the fingerprint query is correctly rejected). Evaluation by using a set of data produces a false rejection rate (FRR) and a false acceptance rate (FAR) along with their corresponding genuine acceptance rate (GAR) and genuine rejection rate (GRR), respectively. The ideal condition is achieved when both FRR and FAR are equal to zero, which means all fingerprint queries are correctly accepted and rejected. Yet, it is impossible to have that value at the same time [78] since there is a trade off between them.

Plotting the FAR and FRR at the same time results in an equal error rate (EER) curve. In general applications, it is common to select the environmental setting based on an EER value, i.e., the error rate when the FRR and FAR are equal. The matching threshold selection, of course, depends on the purpose of the system. If high security is preferred for example, a low FAR is recommended instead of neither a low FRR nor a low EER. The performance evaluation is also performed by plotting the FAR and GAR into a curve, which results in the receiver operating characteristic (ROC) curve [36]. These EER and ROC curves have made it easier to evaluate the performance along with the respective thresholds. In addition to the accuracy level (represented by an error rate), the performance can also be viewed from other factors, for instance, the speed of the matching process. Different applications may have different preferences in what performance factors being required. In this thesis,

the performance is more focused on the accuracy; nevertheless, other factors may also be identified.

In addition, a fingerprint image may not meet the quality standard to be a template due to some reasons. For example, it does not contain enough features (the singular point or minutiae points). In this case, the failure to enroll (FTE) status will be generated. Practically, both the "actual FRR" (due to the weakness of the feature representation, the feature transformation or the matching algorithm/implementation) and this FTE (due to either the weakness of the feature extractor algorithm/implementation or the quality of fingerprints itself) constitute the final FRR value. Overall, the total FRR will affect the GAR.

3.2.2 Experimental Environment

Similar to that of other research, such as in [108, 65, 45], the evaluation of the proposed approaches is conducted in a public database of Fingerprint Verification Competition 2002 (FVC2002) [61]. This database contains fingerprint images whose quality is varied. Like other FVC databases [60, 62], this database consists of four sub-databases. The first three sub-databases were obtained from real fingers whilst the last one was generated synthetically. The FVC2002 database itself is suitable to any size of touch sensors, in particular large-area sensors [64]. Some other databases are also available for research, such as that of National Institute of Standards and Technology [70]. However, they may not be appropriate to evaluate the algorithms which should run on the specific environment, for example, the live-scan operation [64].

The feature extraction from fingerprints is carried out by using VeriFinger software [69] to collect a set of minutiae points; and FOMFE model [105] to obtain the core point. The set of features obtained from this extraction process can be denoted in Equation 3.2. This set is to be the input to the proposed fingerprint data protection approaches.

$$\left.\begin{aligned} & B \in \psi \\ & B = \{m_i\}_{i=1}^{n} \\ & m_i = (x, y, \theta, type)_i \\ & m_{sp} = (x, y, \theta)_{sp} \end{aligned}\right\} \tag{3.2}$$

where ψ is the fingerprint domain space; B is a set of minutiae points extracted from a fingerprint image; m_i and m_{sp} are the i^{th} minutia point and the singular point (i.e., core point), respectively; (x, y) is the point coordinate in the Cartesian space; θ is the point orientation, *type* is the minutia type (in this case is either ridge ending or ridge bifurcation), and n is the total number of minutiae points in B.

The research is run based on the assumption that users willingly present their fingerprint data to be verified, similar to [108, 45, 65], such that, the fingerprint data should have a relatively good quality. In order to comply with this circumstance, a Db2a sub-database is selected among those in FVC2002. This sub-database contains 100 fingers for the testing. By implementing the similar testing approach to that in [108, 45, 65], the fingerprint template and the fingerprint query in the experiment are represented by the first and second sets of impression images available in that sub-database, respectively. In addition, other sub-databases (i.e., Db1a and Db3a) will also be used in the experiment for a performance comparison purpose (described in Section 3.3.1).

Suppose there are two sets of fingerprint impression images (i.e., the first is the set of fingerprint templates and the second is the set of fingerprint queries). For all scenarios used in the experiment, the genuine user evaluation is carried out by comparing each image in the second set with its respective image in the first set, whilst the imposter evaluation is conducted by comparing each image in the second set with all images in the first set except its respective pair image. From this evaluation setting, there are 10,000 fingerprint comparisons in total, which consist of 100 genuine user and 9900 imposter evaluations.

The error rate obtained from the experiment represents the capability of the modules involved in the authentication system (in this thesis, those are feature transformation, feature representation and matching modules). It can be interpreted as follows. Firstly, by comparing the error rate obtained from the evaluation of transformed fingerprint template-query pairs with that of non-transformed ones, the amount of the performance degradation caused by the transformation can be inferred. This can be determined if only the format of both transformed and non-transformed fingerprint data is same, such as the one proposed in Chapter 5. In case the protection approach has made the fingerprint data format changes, such as the one proposed in Chapter 4, the performance degradation cannot be determined. This is because the specific matching module does not work on that non-transformed template format. Secondly, the error rate obtained from the evaluation of transformed templates means the performance (accuracy) of the overall system itself. The evaluation can be done by comparing this error rate with that of other transformation approaches.

3.3 Evaluation Scenarios

The evaluation scenarios are designed such that they reflect real world cases, for example, the transformed template has been compromised and a new set of keys is issued to generate a new transformed template. For the following evaluation scenarios, let $\delta_d(a, a')$ and $\delta_s(a, a')$ be the difference level and the similarity level of the fingerprint template a and the fingerprint query a', respectively; Φ_d and Φ_s be the maximum difference level and the minimum similarity level allowed for a and a' to be recognized as successfully "matched" or "verified", respectively; and $\Gamma(a, \kappa_1, \kappa_2, ..., \kappa_q)$ be the transformation of a by using a function Γ and some q keys $\kappa_1, \kappa_2, ..., \kappa_q$, where $q \geq 0$.

The selection of which $(\delta_d(a, a'), \Phi_d)$ or $(\delta_s(a, a'), \Phi_s)$ pair should be used depends on the matching algorithm being implemented. For example, in Chapter 4, the matching algorithm verifies the difference between the fingerprint template and the fingerprint query, so, in order

to make a "matched" decision in the transformed domains, a legitimate template and query pair has to satisfy $\delta_d(\Gamma(a,\kappa_1),\Gamma(a',\kappa_1)) \leq \Phi_d$. On the other hand, in Chapters 5 and 6, the matching algorithm verifies the similarity between the fingerprint template and the fingerprint query, so, in the non-transformed and transformed domains, a legitimate template and query pair has to meet $\delta_s(a,a') \geq \Phi_s$ and $\delta_s(\Gamma(a,\kappa_1),\Gamma(a',\kappa_1)) \geq \Phi_s$, respectively.

Additionally, evaluation terminologies used in [8, 77, 80, 34, 18] are redefined and combined. For simplicity purpose, the notations used in the following sections refer to $(\delta_d(a,a'),\Phi_d)$ pair.

3.3.1 Accuracy

It is to measure the effect of the transformation on the capability to accept similar fingerprints and to reject dissimilar fingerprints, that is, how the transformation tolerates the intra-user variability (same fingers/users) and does not tolerate the inter-user similarity (different fingers/users). This scenario also represents the worst case that the adversary has been able to steal the user's transformation key. By using this stolen key, along with his/her own fingerprint data, the adversary tries to authenticate himself/herself to the system. It is assumed that the raw fingerprint data of the users is safe and the adversary does not have this information at all.

The testing is conducted by transforming both the legitimate and illegitimate fingerprints using the same set of keys, as defined in [8, 80]. It is to evaluate whether these conditions are fulfilled:

$$\left.\begin{aligned} &\delta_d(a_1,a_1') \leq \Phi_d \\ &\delta_d(\Gamma(a_1,\kappa_1),\Gamma(a_1',\kappa_1)) \leq \Phi_d \end{aligned}\right\} \tag{3.3}$$

$$\left.\begin{aligned} &\delta_d(a_1, a_2') > \Phi_d \\ &\delta_d(\Gamma(a_1, \kappa_1), \Gamma(a_2', \kappa_1)) > \Phi_d \\ &a_1 \nsim a_2 \end{aligned}\right\} \tag{3.4}$$

The Equations 3.3 and 3.4 mean that if before the transformation the query matches to the template, then after the transformation the query should match to the template, too; and if before the transformation the query does not match to the template, then after the transformation the query should not match to the template, either. This is also to measure how much the performance degradation caused by the transformation function is. It is very likely that after the transformation, the error rate goes up. Ideally, Equation 3.3 works only on the fingerprint pairs derived from the same fingers whereas Equation 3.4 works on the fingerprint pairs derived from different fingers.

In this testing scenario, for the purpose of comparison, the experiment is also done on the sub-databases Db1a and Db3a whose image quality is lower than that of Db2a. Furthermore, both of them contain spurious and missing minutiae points. This is appropriate to represent the practical situation where the legitimate users do not willingly provide their fingerprint data to be verified or the adversary attempts to pretend to be a legitimate user by presenting the legitimate but incomplete fingerprint data. Since this is out of the defined assumption, the thesis still focuses the experiment on Db2a. Therefore, in the following chapters, otherwise it is explicitly mentioned, the experiment is carried out in Db2a.

3.3.2 Revocability and Diversity

In case the set of keys or the transformed template is compromised as previously discussed in Section 3.3.1, it must be canceled and a new set of keys is issued to generate a new transformed template. The capability for canceling and issuing the new transformed template is called revocability. The new transformed template itself must be different enough from the old one

even though it is actually derived from the same finger. This is called diversity [80, 18]. In particular, the new transformed template must not authenticate the old transformed query and the old transformed template must not authenticate the new transformed query, either. This means that there is diversity between those transformed fingerprint data, which also means that the privacy is maintained. In order to evaluate this case, p_1-FAR is defined. This is to be a pseudo FAR value resulted from the evaluation of the fingerprint template and the fingerprint query after being transformed by using different keys, as formulated in Equation 3.5. It is worth mentioning that both the fingerprint template and the fingerprint query are originated from the same fingers that in the non-transformed domain, they authenticate each other [8, 80].

$$\left.\begin{aligned} &\delta_d(a_1, a_1') \leq \Phi_d \\ &\delta_d(\Gamma(a_1, \kappa_1), \Gamma(a_1', \kappa_2)) > \Phi_d \\ &\kappa_1 \neq \kappa_2 \end{aligned}\right\} \tag{3.5}$$

In an ideal case, each user has a unique set of keys, which is not compromised. It means that the adversary does not have knowledge about the key and both the transformed and non-transformed fingerprint data. This situation can be represented in Equation 3.6 that in both non-transformed and transformed domains, different fingerprints do not authenticate each other. In this case, the transformation is also conducted by using different sets of keys. For the evaluation, r-FAR is defined. This is the proportion number resulted from incorrectly accepting illegitimate transformed queries. In addition, those testings in Equations 3.5 and 3.6 also indirectly evaluate the effect of the key selection on the transformation.

$$\left.\begin{aligned} &\delta_d(a_1, a_2') > \Phi_d \\ &\delta_d(\Gamma(a_1, \kappa_1), \Gamma(a_2', \kappa_2)) > \Phi_d \\ &a_1 \nsim a_2 \\ &\kappa_1 \neq \kappa_2 \end{aligned}\right\} \tag{3.6}$$

3.3.3 Changeability

The fingerprint data protection approaches should make the transformed fingerprint data different from its non-transformed version. It means that there should exist differences such that transformed and non-transformed fingerprint data do not authenticate each other. In order to evaluate this case, p_2-FAR is specified. This is to be the pseudo FAR resulted from the comparison between the transformed fingerprint template and the non-transformed fingerprint query. This can be denoted by Equation 3.7 [80].

$$\left.\begin{aligned} &\delta_d(a_1, a_1') \leq \Phi_d \\ &\delta_d(\Gamma(a_1, \kappa_1), a_1') > \Phi_d \end{aligned}\right\} \tag{3.7}$$

Equation 3.7 also represents the assumption that in the authentication system, the enrollment is done under supervision in a high secure environment. This implies that the adversary is unable to bypass the transformation system. In this case, the fingerprint is automatically transformed into a secure mode. On the other hand, the recognition step is unsupervised and distributed to the clients. This means that the security is likely to be lower than that of the enrollment step. It indicates that in a bad case, the adversary may be able to bypass the transformation. As a result, the fingerprint query, which is not transformed, is compared with the transformed template as represented by Equation 3.7. In addition, this evaluation scenario also depicts the case when the adversary has been able to compromise the non-transformed fingerprint data and use it as a query in the authentication process. Because the set of keys is safe, the adversary cannot transform the data into an appropriate secure version.

Fulfilling the requirements in Equation 3.7 means that transforming fingerprint data is equivalent to generating a new fingerprint. In other words, the transformed fingerprint can be viewed as a different fingerprint. This assumption also works on the diversity property

(Equations 3.5 and 3.6) as previously discussed.

3.3.4 Non-Invertibility

The original (non-transformed) fingerprint data, ideally, cannot be recovered given the transformation function, transformed template and even the set of keys, that is, the transformation is a one-way function (also called a non-invertible function). In the real world, in addition to this non-invertibility factor, it is likely that there are other security mechanisms to protect the databases (smart cards) containing the transformed template, assuming that the transformation function is public. Yet, this thesis does not discuss such mechanisms as it more focuses on the template transformation itself.

In this case, non-invertibility (security) is defined as the difficulty in reconstructing the original fingerprint data given its transformed version [18, 80]. A function is categorized as highly secure if there is no way recovering the original fingerprint data but carrying out a brute force attack, even though all transformation function properties (i.e., the transformation function, the transformed template and the set of keys) have been known. This ideal case can be formulated in Equation 3.8 that the original fingerprint template a is transformed into b by using the transformation function Γ and key κ_1 such that there is no inverse function $\Gamma^{-1}(b, \kappa_1)$ can be used to generate a from b. However, this is just the maximum security level. The practical transformation function is very likely to have a lower non-invertibility level than that, because of the trade-off between non-invertibility and discriminability (see Section 2.4.2).

$$\left.\begin{aligned} b &= \Gamma(a, \kappa_1) \\ a &\neq \Gamma^{-1}(b, \kappa_1) \\ a &\neq b \end{aligned}\right\} \tag{3.8}$$

3.4 Summary

In this chapter, the research scope of the thesis has been described. It covers three modules of the transformed fingerprint-based authentication system, those are: feature transformation, feature representation and matching modules. The research is conducted by inserting that first module and redefining those second and third modules in the fingerprint-based authentication system architecture. Those three modules determine the overall performance of the authentication system. In other words, a poor design or implementation of either one of those modules may significantly decrease the performance.

In protecting the fingerprint data, the key or parameter is required, mainly for the revocation purpose. Particularly, in case the transformed template is compromised, the new key is issued to generate the new transformed template. In addition, the same keys and parameters are applied to transform fingerprint template-query pairs in the authentication process, such that, similar fingerprints (inputs) result in the same outputs and dissimilar fingerprints result in different outputs.

Various verification testing scenarios have been designed, whose goal is, in general, to evaluate the performance of the system. Those testing scenarios themselves represent the real world cases, which include the evaluation of (i) the accuracy, which is done by authenticating both legitimate and illegitimate transformed queries whose transformation key is same as that of the transformed template; (ii) the revocability, which is conducted by re-issuing the transformation key and generating its corresponding transformed template; (iii) the diversity, which is performed by authenticating both the legitimate and illegitimate transformed queries whose transformation key is different from that of the transformed template; (iv) the changeability which is carried out by authenticating the legitimate non-transformed query to its transformed template counterpart.

The experimental results obtained from those scenarios (i.e., scenarios (i)-(iv)) are rep-

resented by true and false positive/negative rate values (i.e., GAR, GRR, FRR and FAR). These values are then plotted on either ROC or EER curve to make it easier to analyze. Additionally, in certain cases, the features cannot be extracted from fingerprints which lead to a failure to enroll (FTE). The performance level, specifically the matching accuracy of both non-transformed and transformed fingerprint data, can be used to indicate the performance degradation as well as its relative performance to other fingerprint data protection approaches. This is considering that the transformation is very likely to cause an increase of the error rate. Some properties of the transformation function, such as non-invertibility and the verification speed, are also evaluated.

In the real application, the selection of the authentication system settings (e.g., matching threshold, maximum error rate) depends on the purpose of the system, whether low FAR, low FRR or low EER is required. The evaluation of the proposed transformation functions itself is performed on the public sub-database FVC2002Db2a, by considering the assumption being used in the research. For the comparison purpose, the other sub-databases (i.e., FVC2002Db1a and FVC2002Db3a) are also used in the certain testing scenario.

Chapter 4

Projection-based Transformation

This chapter proposes a global feature-based transformation function (cancelable template design), that minutiae points in the fingerprint are transformed with respect to the singular point (i.e., core point). In particular, both the location and the orientation of the core point is to be the reference to the minutiae point transformation.

This chapter is structured as follows. Section 4.1 depicts state-of-the-art global feature-based cancelable fingerprint template approaches along with their potential problems. A general survey on singular point detection is also provided in this section. Section 4.2 describes the minutiae points projection design. The results of experiments, which were conducted in various scenarios and databases, are provided and analyzed in Section 4.3. Finally, this chapter is summarized in Section 4.4.

4.1 Global Feature-based Cancelable Templates

Fingerprints mostly fall into classes which contain singular points [107], whose type, number and location depend on their corresponding class (neither core nor delta points are available in the arch fingerprint class, refer to Section 2.1). These singular point characteristics represent a global fingerprint pattern, which in turn provides global fingerprint information. Indeed, the

uniqueness of fingerprints cannot rely on them alone because all fingerprints in a same class share similar characteristics. Therefore, in global fingerprint feature-based authentication systems, a singular point is usually utilized along with minutiae points. In this case, a singular point is used as a reference point to registering a fingerprint query. This registration process deals with translation and rotation settings being applied for aligning fingerprint template and fingerprint query [112]. For this registration, actually, any stable point is suitable, regardless of its type. However, it is found that core point detection is more stable than that of delta point [116], which makes it more appropriate to use.

Overall, global fingerprint feature-based authentication systems have some advantages [96], which make it appropriate for them to apply for resource constrained devices. Likewise, global feature-based transformation hold these advantages in spite of having to rely on the accurate core point data (e.g., location and orientation). This is crucial because the transformation of fingerprint features (e.g., minutiae points) is also conducted based on this data.

Research in finding accurate singular point detection has been conducted. For example, Zhou et al. [116] detected fingerprint singularities by using the orientation field. Specifically, they used zero-pole model [86] to get an accurate and efficient reconstructed orientation image. The experimental result, however, shows that the total number of incorrect detection is about 20% which is relatively high. Wang et al. [102] proposed a singular point detection method by defining the relation between that point and its corresponding neighbours. They are able to achieve 7.19% of EER. A better result is shown in [105], which singular point detection is performed by utilizing the 2D fourier expansion method called FOMFE. It is claimed to be able to work well in noisy fingerprint images. The experimental result, which is obtained from a matching scenario, exhibits lower error rates than those of other methods, that FAR = 1% and FRR = 3.6% can be achieved simultanously.

Nevertheless, a small difference of singular point data can lead to much transformed fin-

gerprint template deviation. Therefore, despite the improvement of singular point detection method as shown in [105], there still should be another mechanism to minimize the effect of inaccurate singular point detection. This mechanism can be implemented in each module of the authentication system architecture.

By relying on the proposed singular point detection methods, many global feature-based transformation functions have been introduced. These can be either an image-based or geometrical-based approach. However, most of them suffer from performance (accuracy) and even reversibility issues. Moreover, their EER can be more than 15%, which is considered to be high as described below.

A transformation function proposed by Ang et al. [8] was developed by firstly constructing a line crossing the core point. There are two purposes of the use of this line. First, its angle is to be the transformation key. Second, it is to be a transformation line, which reflects the minutiae points of the first half image onto another half such that this second half contains all minutiae points of the fingerprint. The fingerprint verification is performed by evaluating this combined space (the second half space) using a matching algorithm in [48]. After the transformation, around 16.8% of EER was obtained. This error rate is relatively much higher than that of without transformation, which was found to be 4%. These results depict that there is an EER increase of about 13%.

In terms of non-invertibility, this transformation may not be high. The fact that there are only two subspaces (one below and one above the reflection line) makes it easy for an adversary to recover the original fingerprint data in the event that this transformed template is compromised. Moreover, it is known that one of those two subspaces contain information of all minutiae points, that about half number of them is the transformed (reflected) minutiae points and the rest is non-transformed (unreflected). The distance between minutiae points in the combined subspace and the reflection line provides information of non-transformed minutiae points location.

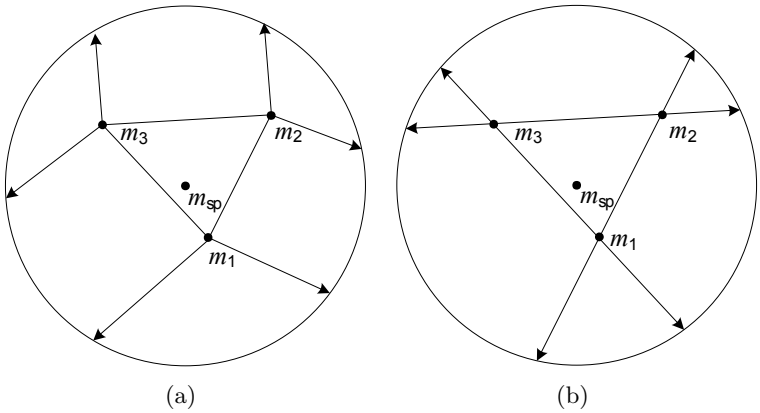


Figure 4.1: Mapping points onto the circle (a) perpendicular mapping of [113] (b) straight mapping of [92].

Yang et al. [113] developed a scheme by utilizing both local and global features of fingerprints. In this scheme, a circle with a certain radius is drawn centering at the fingerprint core point. Each minutia point pair in the circle is connected with a line and is mapped onto the circle in the perpendicular direction (Figure 4.1(a)). While global features are obtained from each minutia point relative position to the core point, local features are from each triangular properties formed by a set of three minutiae points, which include the difference angle of two minutiae orientation, and the angle between the line connecting two minutiae and their minutiae orientation. This configuration is intended to improve the performance of a scheme proposed by Sutcu et al. [92]. In this case, each minutia point pair was mapped onto the circle in a straight instead of perpendicular direction (Figure 4.1(b)). This straight mapping causes arbitrary distances of the transformed minutiae points, which affect the overall performance. Yang et al. [113] show that their approach have been able to reach 13% of EER. This is about 19.8% lower than the EER obtained by [92].

A relatively high error rate obtained by Yang et al. [113] and Sutcu et al. [92] is mainly caused by incapability of the transformation function to accommodate the fingerprint translation and rotation issues. A small position change (reordering) of minutiae points can result in different mapping point location as illustrated in Figure 4.2. Insertion and deletion of

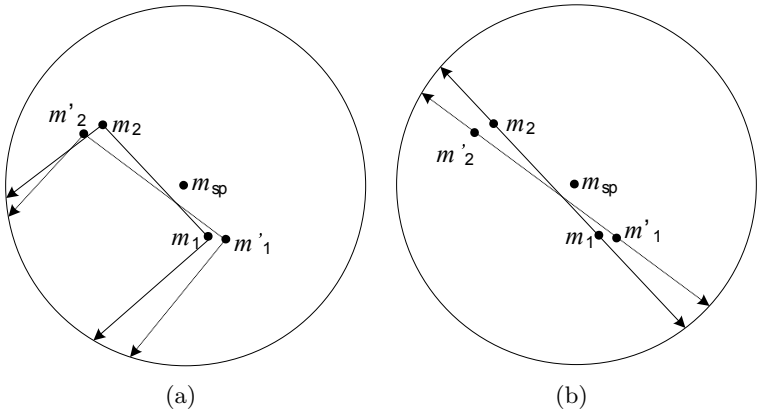


Figure 4.2: The effect of the reordering minutiae points (a) the mapping function of [113] (b) the mapping function of [92].

minutiae points make the mapping results even worse because there is no corresponding point in the template/query.

4.2 Minutiae Point Projection Design

In this proposed scheme, the transformed fingerprint template is stored in the form of a vector string. This is different from that of its original fingerprint data, where both core and minutiae points are represented in the forms of the coordinate, orientation and type. The vector string generation process itself consists of a number of steps, that each of them is given a parameter or a set of keys, as depicted in Figure 4.3. In this case, the singular point (i.e., core point) is also to be the transformation reference point.

4.2.1 Quantization

A fingerprint is aligned with the Cartesian coordinate space by locating its core point (m_{sp}) at the center of the space, whose orientation is to be the x-axis. In this coordinate space, cells (squares) are constructed, as depicted in Figure 4.4. Shibata et al. [87] also utilized cells and extracted ridge orientation from each of them. In their approach, the number of cells is

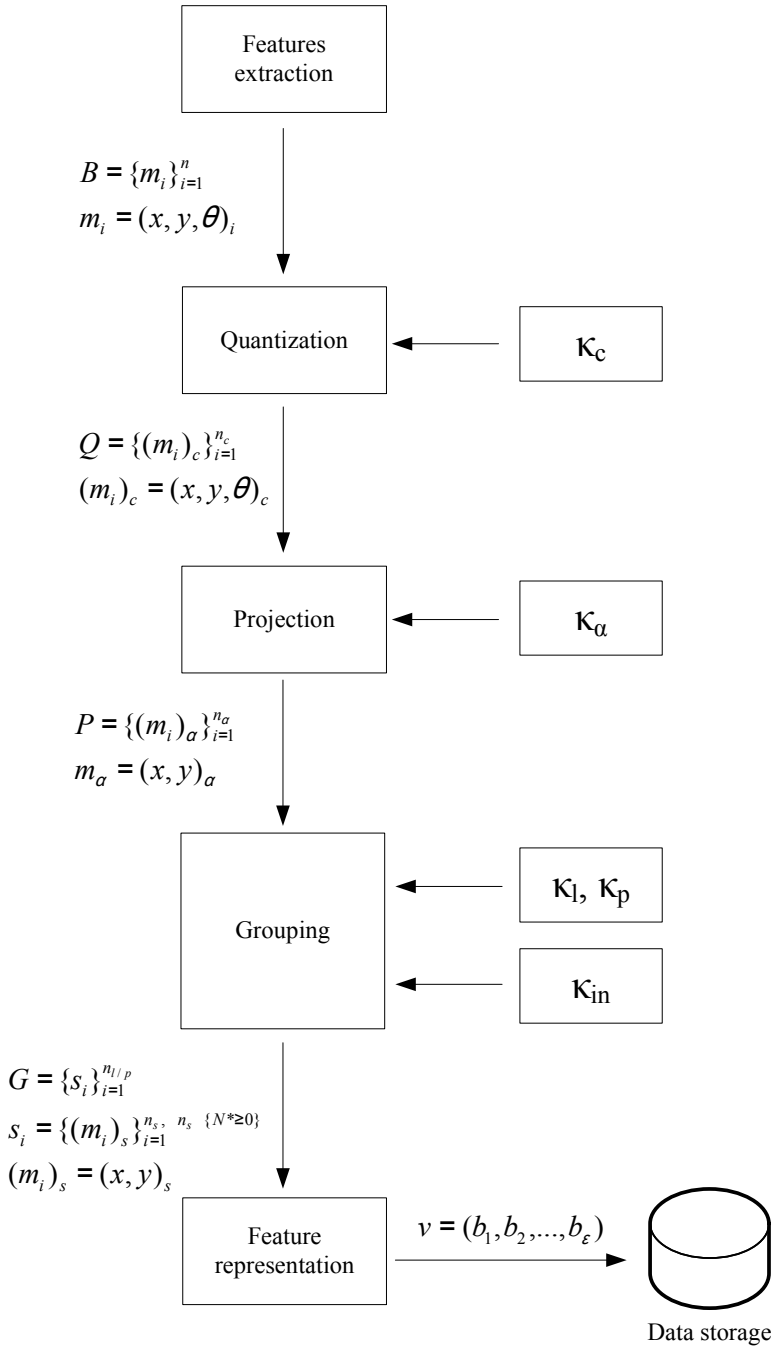


Figure 4.3: The projection-based transformation architecture.

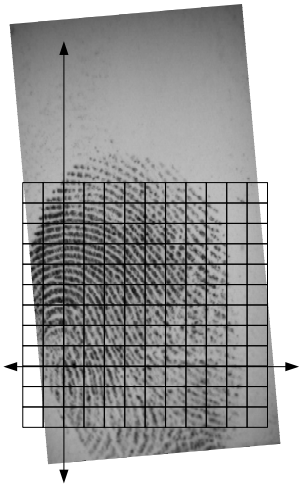


Figure 4.4: In the quantization step, the fingerprint coordinate space is divided into subspaces (cells or squares).

limited to 8×8 blocks whose size is fixed at 16×16 pixels. Different from that approach, this thesis constructs the cells such that all or most the minutiae points in a fingerprint are covered depending on the total number and size of the cells. These cell construction characteristics (for example, the size of cells) are determined by the parameter ρ_c.

All minutiae points in each cell, if any, are mapped by a function Γ_c onto the center of the corresponding cell. Therefore, in the event that the cell contains more than one minutia point, there is a many-to-one mapping in that cell. It is worth noting that the information of minutia point total number of each cell is not stored. As a result, there is no information about which cell performs either many-to-one or one-to-one mapping process. An empty cell, of course, represents that there is no minutia point in the corresponding cell both before and after the mapping. In this case, each point being mapped still keeps its minutia orientation and type information such that it is independent from the coordinate mapping. This means that a point may change its coordinate but not its orientation and type. This is to maintain the uniqueness of the fingerprint pattern. Note that in this transformation design, the minutia type information is not used; therefore, in the next sections it will not be referred. Globally,

this step is denoted as:

$$\{(m_i)_c\}_{i=1}^{n_c} = \Gamma_c(\{m_i\}_{i=1}^{n}, \rho_c) \tag{4.1}$$

where m_i and n are respectively the minutia point which has been extracted from a fingerprint and the number of minutiae points in the corresponding fingerprint (refer to Equation 3.2); $(m_i)_c$ and n_c are the minutia point which has been mapped by the function Γ_c and the number of mapped points, respectively. In this step, $n_c = n$ and each mapped point is independently processed in the subsequent steps. Specifically, all points within a cell, if any, are mapped according to Algorithm 4.1.

The effect of reordering on minutiae point location is minimized. This is because all minutiae points within a cell are translated to a point, regardless of their location in the cell; while at the same time, the uniqueness of each minutia point is maintained by the unchanged orientation information. As a trade-off, the reordering problem is not fully solved. For example, in the subsequent fingerprint scan, minutiae points whose locations are close to the cell boundary may move to a next cell, which leads to reducing the minutiae number in the original cell and increasing that of the other. Minutiae insertion and deletion problems are equivalent to this case.

Algorithm 4.1 Quantization step

Input: $\{m_i\}_{i=1}^{n}$
Output: $\{(m_i)_c\}_{i=1}^{n_c}$

```
1: for p ← 1 to total_cells_in_B do
2:    (x_p, y_p) ← center_point_coordinate_of_cell_p
3:    if total_minutiae_in_cell_p ≠ ∅ then
4:       for i ← 1 to total_minutiae_in_cell_p do
5:          (m_i)_c ← (x_p, y_p, θ_i, t_i)
6:       end for
7:    end if
8: end for
```

4.2.2 Projection

A line L_α, which is denoted by $(y_i)_\alpha = s_\alpha \times (x_i)_\alpha + c_\alpha$, where s_α is the slope and c_α is the $(y_i)_\alpha$-intercepts of the line, is drawn in the coordinate space. In this case, it is defined that this line crosses the axis at $m_{sp}(0,0)$; therefore, $c_\alpha = 0$. Suppose α is the rotational distance of L_α from x-axis in counterclockwise. The slope is defined as $s_\alpha = tan(\alpha)$, where $tan(\alpha)$ is the tangent function against α. The information of α itself is stored in the key κ_α.

All mapped minutiae points in the previous step ($\{(m_i)_c\}_{i=1}^{n_c}$) are projected onto the line L_α with respect to the x-axis and y-axis as represented in Figure 4.5(a) according to Equation 4.2.

$$\{(m_i)_\alpha\}_{i=1}^{n_\alpha} = \Gamma_\alpha(\{(m_i)_c\}_{i=1}^{n_c}, \kappa_\alpha) \tag{4.2}$$

where $(m_i)_\alpha$, n_α and Γ_α are the projected minutia point, the number of projected minutiae points and the projection function, respectively. This projection, called (x,y)-projection, specifies that $n_\alpha = 2n_c$ and the resulted minutiae points projection spread over the line L_α. According to Figure 4.5(a), the projection of $(m_1)_c$ onto $(m_1)_\alpha$ and $(m_2)_\alpha$ is performed based on the fact that their abscissa or ordinate is identical. Suppose $(m_1)_c = ((x_1)_c, (y_1)_c), (m_1)_\alpha = ((x_1)_\alpha, (y_1)_\alpha), (m_2)_\alpha = ((x_2)_\alpha, (y_2)_\alpha)$, it can be inferred that:

$$\left.\begin{aligned} (x_1)_c &= (x_1)_\alpha \\ (y_1)_\alpha &= s_\alpha \times (x_1)_c \\ (y_1)_c &= (y_2)_\alpha \\ (x_2)_\alpha &= (y_1)_c / s_\alpha \end{aligned}\right\} \tag{4.3}$$

In addition, the orientation of minutiae is to be the third point generated by this projection step. This is obtained according to the point at which the line crosses with the corresponding minutia orientation line, as depicted in Figure 4.5(b). This projection, called (x,y,θ)-projection, results in $n_\alpha = 3n_c$. It is worth mentioning that the variation of the

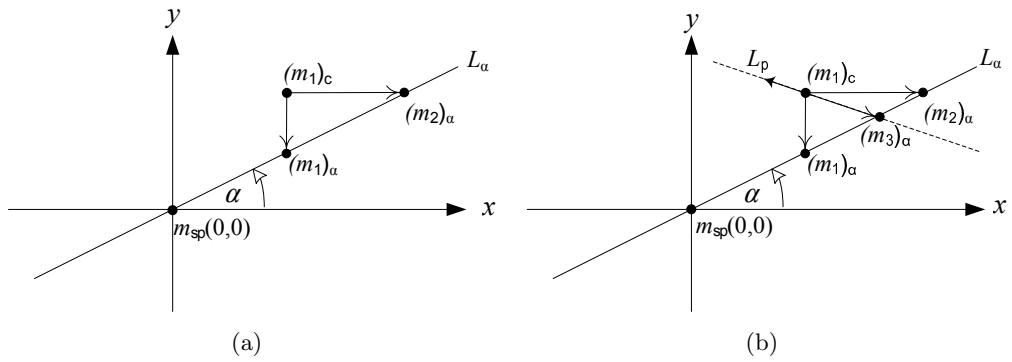


Figure 4.5: An example of the minutia projection step. A minutia point $(m_1)_c$ is projected onto the line L_α whose slope is determined by κ_α (a) projection with respect to both $x-axis$ and $y-axis$ (b) projection with respect to $x-axis$, $y-axis$ and θ.

minutiae orientation is higher than that of minutiae location [65]. Therefore, each projected point $((m_i)_\alpha)$ is assigned a weight ω_i, such that:

$$\forall i \in \{\mathbb{N}^* \leq n_\alpha\} : \omega_i = \begin{cases} \omega_{ori} \text{ if the point is from the orientation } (\theta)\text{-projection} \\ \omega_{cor} \text{ if the point is from the coordinate } (x,y)\text{-projection} \end{cases} \tag{4.4}$$

where ω_{ori} and ω_{cor} represent the weight of θ- and (x,y)-projections, respectively. In this case, points resulted from θ- projection have smaller weight than that from (x,y)-projection.

Suppose $L_p : y_p = s_p \times x_p + c_p$ the orientation line according to the orientation of $(m_i)_c$; $(m_3)_\alpha = ((x_3)_\alpha, (y_3)_\alpha)$ the projected point resulted by θ-projection (refer to Figure 4.5(b)). This projected point, $(m_3)_\alpha$, is defined by assuming that L_α and L_p are crossing at $(m_3)_\alpha$, such that:

$$\left.\begin{aligned} (y_3)_\alpha &= y_p \\ s_\alpha \times (x_3)_\alpha &= s_p \times x_p + c_p \text{ where } (x_3)_\alpha = x_p \\ (x_3)_\alpha \times (s_\alpha - s_p) &= c_p \\ (x_3)_\alpha &= \tfrac{c_p}{s_\alpha - s_p} \\ (y_3)_\alpha &= s_\alpha \times (x_3)_\alpha \end{aligned}\right\} \tag{4.5}$$

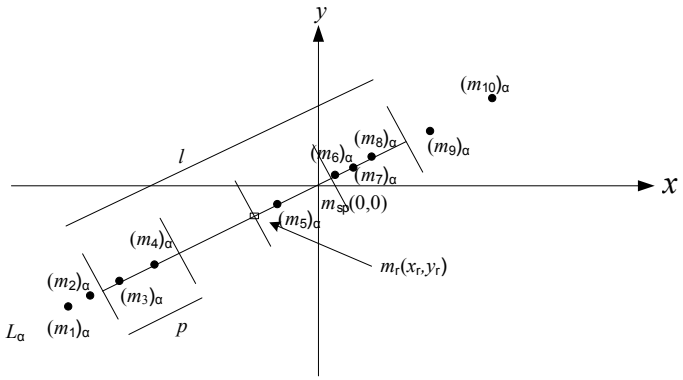


Figure 4.6: The grouping step. In this example, projected points $\{(m_i)_\alpha\}_{i=1}^{n_\alpha}$, $n_\alpha = 10$ *are grouped into 4 partitions.*

4.2.3 Grouping

The projected points $\{(m_i)_\alpha\}_{i=1}^{n_\alpha}$, which spread over the line L_α, are divided into (l/p) groups, where l and p values are derived from the keys κ_l and κ_p, respectively. Specifically, κ_l determines the length of line L_α involving in this grouping step, whilst κ_p specifies the length of each group, as depicted in Figure 4.6. The weight ω of points in each group is summed up and mapped onto a vector v whose total number of elements is also (l/p). Therefore, there is a one-to-one mapping between (l/p) groups in L_α and (l/p) elements in v. It is worth mentioning that some projected points in $\{(m_i)_\alpha\}_{i=1}^{n_\alpha}$ may be beyond the range of l. As the result, those points are not involved in the grouping process.

The rule of how to map the total weight of points in each group onto the element of vector v is specified by the key κ_{in}; that is, this key arranges the index (permutation) of each group in L_α, such that it contains a set of q possible indices (see Equation 4.6). There are at least two advantages offered by introducing this permutation. First, it is useful for revocation. Second, it makes the secure fingerprint template more unique. As the result, this reduces

the error rate caused by the inter-user similarity.

$$q = \frac{(l/p)!}{((l/p) - (l/p))!} = (l/p)! \tag{4.6}$$

In addition to the length of both the projection line and the group, the position of groups itself is also specified. Suppose $m_r(x_r, y_r)$ is the midpoint of L_α, which is unnecessary to be same as $m_{sp}(0, 0)$. This point, $m_r(x_r, y_r)$ is defined as:

$$\left.\begin{aligned} x_r &= \tfrac{max(\Pi x_\alpha) - min(\Pi x_\alpha)}{2} + min(\Pi x_\alpha) \\ y_r &= tan(\alpha) x_r \end{aligned}\right\} \tag{4.7}$$

where Πx_α is the set of abscissas of $\{(m_i)_\alpha\}_{i=1}^{n_\alpha}$. Considering that minutiae location is more stable than minutiae orientation [65], this Πx_α is restricted to only contain points generated from (x, y)-projection.

The transformation process of this step can be denoted as follows:

$$v = \Gamma_{in}(\{\Gamma_{lp}(\{(m_i)_\alpha\}_{i=1}^{n_\alpha}), \kappa_l, \kappa_p\}_{i=1}^{n_{lp}}, \kappa_{in}) \tag{4.8}$$

where Γ_{lp}, Γ_{in} and n_{lp} are the grouping function, permutation function and number of groups, respectively. In this case, $n_{lp} = (l/p)$. An example of this grouping step is illustrated in Figure 4.6. For simplicity, suppose all points in L_α are generated by (x, y)-projection, whose $\omega_{cor} = 1$. It is defined that there are four groups in L_α. By referring to Equation 4.6, κ_{in} specifies the indexing number of each group, for example, (0, 1, 2, 3). It means that group 0, group 1, group 2 and group 3 contain 2 points $((m_3)_\alpha, (m_4)_\alpha)$, 0 point, 1 point $((m_5)_\alpha))$ and 3 points $((m_6)_\alpha, (m_7)_\alpha, (m_8)_\alpha)$, respectively. It is worth pointing out that there are 4 points in L_α which are not covered by these groups because they are beyond the line length l. These are: $(m_1)_\alpha, (m_2)_\alpha, (m_9)_\alpha, (m_{10})_\alpha$. The total point weight of each group is mapped onto vector v such that $v = (2, 0, 1, 3)$.

4.2.4 Matching

The vector v is to be the template, which is stored in the database. Matching (verification) is conducted by comparing the fingerprint template v with the fingerprint query v'. If v' is similar enough to v, then the verification is successful.

The similarity between v and v' is determined by using mean absolute error [42, 92]. It measures the average of differences between the corresponding vector element pair in v and v', as denoted in Equation 4.9.

$$\delta(v, v') = \frac{1}{\epsilon}\Sigma_{i=1}^{\epsilon}|s_i - s'_i| \tag{4.9}$$

where ϵ, s_i and s'_i are the total number of elements in v, the i^{th} element in vector v and v', respectively. In this case, the value of ϵ must be exactly same as that of n_{lp}. Additionally, in order to make it verifiable, both v and v' must have the same ϵ. The value of $\delta(v, v')$ less than or equal to the specified threshold τ means that v' is similar enough to v and otherwise.

In summary, suppose Γ and κ are the set of all transformation functions and the set of keys used in Γ, respectively. Securing the raw fingerprint B generates the template v, which is denoted as $v = \Gamma(B, \kappa)$. A different set of keys κ is required to generate a different template v which may be used in a different system.

4.3 Experiments and Analysis

The proposed scheme is evaluated in accordance with evaluation designs which have been provided in Section 3.3. Here, the changeability testing is not applicable because the transformation changes the data format. As the result, matching can only be performed after transforming the fingerprint data. This has made the transformed and non-transformed fingerprints incomparable; it means that the transformation function meets the changeability (distortion) property by itself.

4.3.1 Accuracy

In this evaluation, it is assumed that the set of keys $\kappa = \{\kappa_\alpha, \kappa_l, \kappa_p, \kappa_{in}\}$ has been compromised. As previously discussed, both the fingerprint template and the fingerprint query must have a same total group number (l/p) and a same total vector element number (ϵ), which are determined by combination of κ_l and κ_p, to make them verifiable (refer to Section 4.2.4). It means that compromising only κ_α and κ_{in} may not be adequate for the adversary to break the system.

The results of the experiment, which was conducted in FVC2002Db2a, are described in an ROC curve shown in Figure 4.7. It depicts the performance obtained by varying the orientation weight (ω_{ori}) and fixing the location (coordinate) weight (ω_{cor}) to 1. In this case, $\omega_{ori} = 0$ refers to (x, y)-projection, which means that the orientation information is not used.

From Figure 4.7, it is found that $\omega_{ori} = 0.06$ delivers the best result. Specifically, when GAR=94%, its FAR is the lowest among others, which is about 2.39%. It is only slightly lower than that of $\omega_{ori} = 0$ and 0.03. Increasing ω_{ori} leads to decreasing the performance as reflected by $\omega_{ori} = 0.5$ and $\omega_{ori} = 1$. Furthermore, $\omega_{ori} = 0.06$ can achieve a slightly lower EER than that of $\omega_{ori} = 0$, which are about 5.5% and 5.6%, respectively. On the other hand, assigning minutiae orientation the same weight as that of minutiae location leads to a higher EER as represented by an EER curve in Figure 4.8. It can be inferred that this requires a higher threshold (τ). This result has supported the previous assumption that minutiae orientation is more varied than minutiae location.

In more specific, it is found that 3% of genuine user testings result in failure-to-enrol (FTE) because of the unavailability of the core point in the corresponding fingerprints. In this case, the extractor may deliver incorrect core point data, whose example is depicted in Figure 4.9. The FRR depicted in both Figures 4.7 and 4.8 include this FTE rate.

The summary of genuine and false acceptance rates for certain thresholds according to

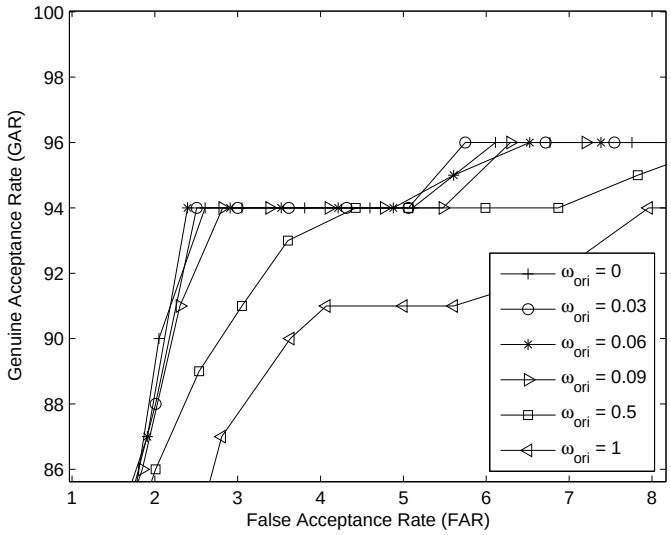


Figure 4.7: The ROC curve of various orientation weights (ω_{ori}). In this case, the location weight (ω_{cor}) is fixed to 1.

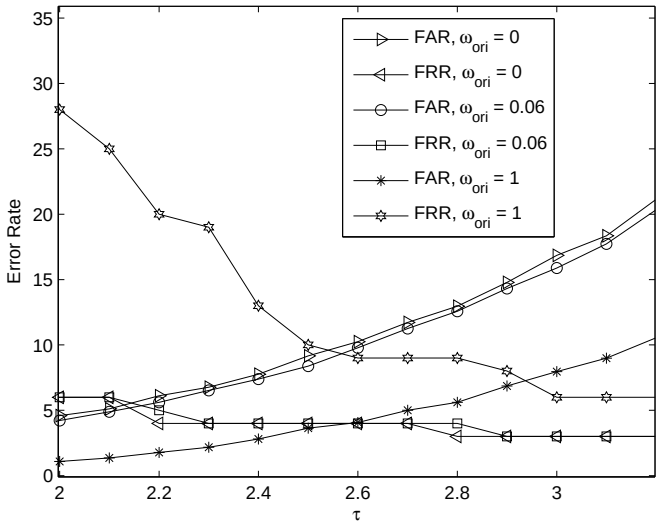


Figure 4.8: The EER of some ω_{ori} values, which reflect the performance of (x, y)- and (x, y, θ)-projection. In this experiment, ω_{cor} is fixed to 1.

Figure 4.9: A fingerprint which does not have the core point. The circle represents the point incorrectly recognized by the extractor to be the core point (the fingerprint image is taken from [61]).

those graphs is provided in Table 4.1. Overall, it can be inferred that in the specific ω_{ori} values, (x, y, θ)-projection gives better results than (x, y)-projection. Selection of an appropriate τ, of course, depends on the implementation, whether security (low FAR) or convenience (low FRR) is preferred.

It is found that the main reason of the false rejection is the small overlapping area between fingerprint templates and fingerprint queries. Figure 4.10 depicts an example of a legitimate fingerprint pair, which fails to authenticate. By referring to their core point, it can be inferred that the template and the query are obtained from relatively different finger sides. As depicted in Figure 4.11, in spite of their small overlapping area, there are still some matched minutiae pairs can be identified; however, their number is significantly lower than that of non-matched minutiae pairs.

In this case, the decision of whether the query matches to the template can be explained as follows. Suppose n and n' are the number of minutiae points in the fingerprint template and query, respectively; there are i minutiae overlapped and l/p groups in L_α (in the experiment, $l/p = \epsilon = 26$). In order to make the fingerprint template-query pair is authenticated, the amount of differences between corresponding elements of template and query vectors

Table 4.1: The summary of experimental results on FVC2002Db2a according to the accuracy scenario for certain thresholds.

τ	FTE (%)	$\omega_{ori} = 0$		$\omega_{ori} = 0.06$		$\omega_{ori} = 0.5$		$\omega_{ori} = 1$	
		GAR (%)	FAR (%)	GAR (%)	FAR (%)	GAR (%)	FAR (%)	GAR (%)	FAR (%)
1.7	3	94	2.60	94	2.39	79	1.19	51	0.34
2.2	3	96	6.11	95	5.60	93	3.60	80	1.78
3	3	97	16.85	97	15.88	96	12.13	94	7.96

(a) (b) (c)

Figure 4.10: A fingerprint pair whose overlapping area is small (a) template (b) query (c) the overlapping area between template and query (fingerprint images are taken from [61]).

should not exceed τ, on average. This amount of differences not only depends on i matched points but also depends on $(n - i)$ and $(n' - i)$ non-matched points of template and query, respectively. Greater i results in smaller both $(n - i)$ and $(n' - i)$, which means that the difference between the template and the query is also smaller. The minimum number of i should be held, however, can not be fixedly defined because n and n' are likely to be different from scanning to scanning.

In fact, relatively small $(n - i)$ and $(n' - i)$ are still acceptable. For example, (x, y, θ)-projection is applied to transform the minutiae points; suppose $\tau = 1.7$, $\epsilon = 26, n = 32, n' = 31$, $i = 30$, $\omega_{cor} = 1$ and $\omega_{ori} = 0.06$. Three non-matched minutiae points ((32-30) + (31-30)

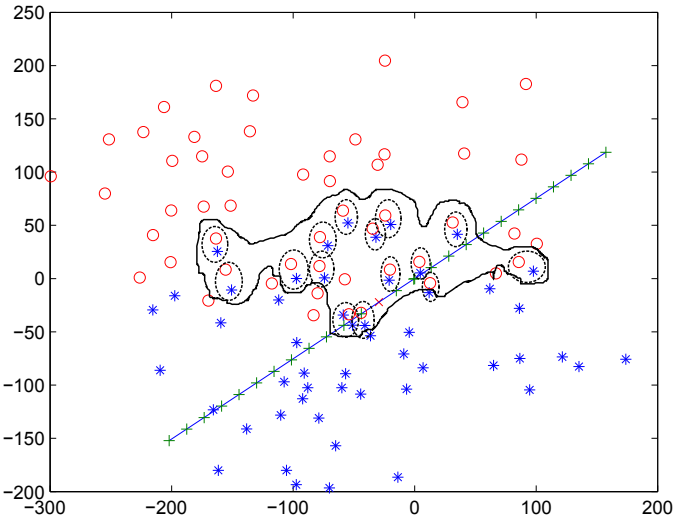


Figure 4.11: Minutiae points of the template and the query to be transformed to the projection line. Partition boundaries, minutiae points of template and query are represented by $+, o$ and $*$, respectively. The corresponding (matched) minutia point pairs are put in the ellipse.

$= 3$) result in $\delta(v, v') = ((3 \times 2 \times 1) + (3 \times 1 \times 0.06))/26 = 0.2377$, assuming that those projected points are covered by the line (within the range specified by κ_l). This value is still lower than the threshold. On the other hand, 20 matched points ($i = 20$) will leave 23 points to be non-matched. Still assuming that all projected points are on the line range, this may generate $\delta = (23 \times 2.06)/26 = 1.8223$ which is greater than the threshold.

For a comparison purpose, the experiment was also conducted in Db1a and Db3a of FVC2002. It is found that in Db1a, there are 2% of fingerprint pairs whose core point is not available. The extractor alternatively delivered a relatively stable point which can be a transformation reference. However, these fingerprints are excluded and are categorized as FTE. This is because such points are core extraction method-dependent. A similar case occurs to Db3a, where there is 1% of fingerprint pairs which does not have the core point. Different from that in Db1a, the alternate point delivered by the extractor is not stable. In addition, it is also found that there are 3% of fingerprint pairs whose core point exists but it cannot be detected and 2% of fingerprint pairs whose minutiae points cannot be extracted

Table 4.2: The GAR and FAR obtained from various databases, where $\omega_{cor} = 1$ *and* $\omega_{ori} =$ 0.06.

FVC2002	τ	FTE (%)	GAR (%)	FAR (%)
Db1	1.7	2	94	4.30
	2.2	2	97	11.22
	3	2	98	34.32
Db2	1.7	3	94	2.39
	2.2	3	95	5.60
	3	3	97	15.88
Db3	1.7	6	92	14.04
	2.2	6	94	33.93
	3	6	94	69.89

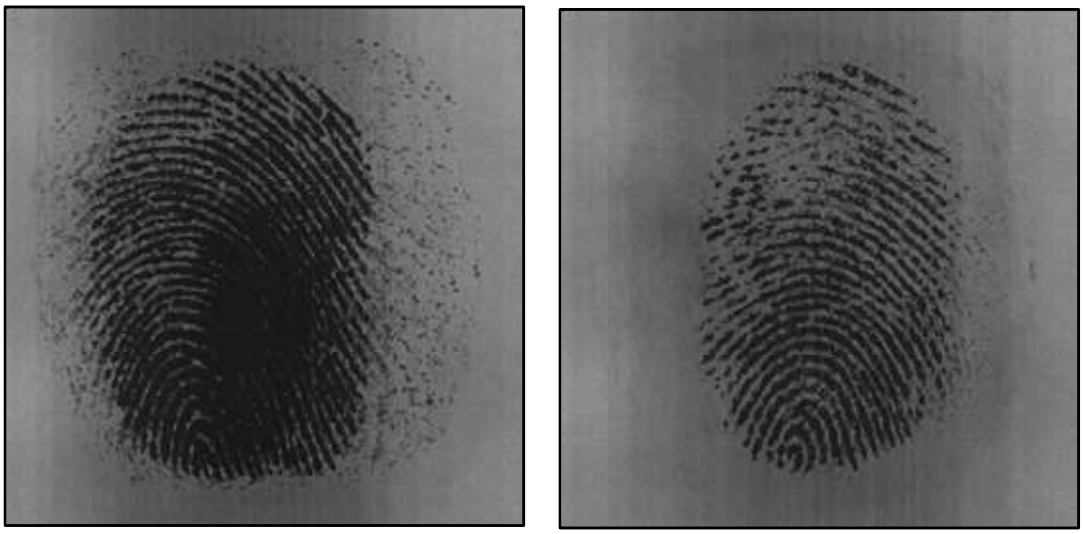
at all, either. So, there are 6% of fingerprint pairs which are classified as FTE.

Along with that of Db2a, the results of the experiments carried out in Db1a and Db3a are provided in Table 4.2. Similar to that in Table 4.1, the total FTE and FRR numbers lead to reducing the GAR. It is shown that the transformation conducted in Db2a generates the highest performance whilst that on Db3a produces the lowest one. Although Db2a depicts a higher number of FTE than Db1a, its performance is still relatively better than that of Db1a. Overall, this is appropriate to the assumption which has been discussed in Section 3.2.2.

In addition, examples of fingerprints with undetectable core and minutiae points are shown in Figure 4.12. It can also be inferred that difficulties in detecting the core point occurring in those databases are caused by following reasons:

- The fingerprint does not have the core point.
- The fingerprint does have the core point but it cannot be detected.

That first reason results in failing to generate the template at any time while the second causes inaccurate core point data. This second problem can be overcome by requiring the users to scan their finger several times until an appropriate fingerprint image is obtained.



(a) (b)

Figure 4.12: Fingerprint with undetectable points (a) undetectable core point (b) undetectable minutiae points (fingerprint images are taken from [61])

Many other global features-based cancelable fingerprint template designs have a relatively higher EER value than that of proposed scheme. For example, the approaches proposed by Yang et al. [113] and Sutcu et al. (cited in [113]) which obtained 13% and 35% of EER, respectively. Both of them were also evaluated using FVC2002Db2a. Additonally, on average, the time taken by this proposed scheme to match a template-query pair is about a second which is low.

4.3.2 Revocability and Diversity

Revoking a transformed template is performed by generating a new template by using a different set of keys κ. In order to evaluate the revocability and diversity properties, 99 random sets of keys $\kappa = \{\kappa_\alpha, \kappa_{in}\}$ were generated for each query (a template-query pair has to have a same (l/p) value to make it verifiable, refer to Section 4.2.4). This leads to 9900 pseudo imposter testings (p_1-FAR). This is to measure the FAR of template-query pairs derived from the same finger but transformed by using different sets of keys.

Table 4.3: The p_1-FAR and r-FAR values, which respectively represent legitimate and illegitimate fingerprint pairs transforming by using different sets of keys.

τ	p_1-FAR (%)	r-FAR (%)
1.7	0.27	0.03
2.2	2.16	0.28
3	13.72	5.92

The experimental results of revocability and diversity are provided in Table 4.3. It is found that this pseudo FAR is low, specifically for $\tau = 1.7$, which is close to zero. It is also denoted that if template and query pairs originating from different fingers are transformed by using random keys (the template and the query have different sets of keys), the false acceptance (r-FAR) is even lower than that of pseudo imposters (Table 4.3). The value of p_1-FAR and r-FAR is close, especially for $\tau = 1.7$ and $\tau = 2.2$. This means that transforming the same fingerprints by using different keys results in different templates, as if they are from different fingers. In other words, this condition leads to a low possibility of cross-matching among databases [34, 49].

4.3.3 Non-invertibility

In the event that v is compromised, the total weight of points in each group involved in the transformation can be revealed. The group permutation number (the relation between groups in the line L_α and elements of the vector v), however, is still unknown. In the worst case when all κ, ρ_c and v are compromised, the adversary is able to reveal the projection line information but not the exact coordinate of projected points $\{(m_i)_\alpha\}_{i=1}^{n_\alpha}$ on this line. Assuming that these projected point coordinates can be found (by performing a trial and error method, the probability of a minutia point location in a partition is $\frac{1}{\lfloor l/p \rfloor} \times 100\%$); and referring to Section 4.2.2, that each point in $\{(m_i)_c\}_{i=1}^{n_c}$ derives three new points $(m_i)_\alpha$ because of the (x, y, θ)-projection; the adversary should find each of these three points to obtain a corresponding point in $\{(m_i)_c\}_{i=1}^{n_c}$. The number of point combinations itself can be

denoted by $\frac{n_\alpha!}{(n_\alpha-3)!}$. It is worth mentioning that each point in $\{(m_i)_c\}_{i=1}^{n_c}$ must be constructed by two points resulted from (x, y)-projection and one point from θ-projection. Therefore, the number of possible $(m_i)_c$ points can be found is $((\sum_{i=0}^{k_1-1} k_1) \times k_2)$, where k_1 and k_2 are the points from (x, y)-projection and θ-projection, respectively. On the other hand, not all projected points $((m_i)_\alpha)$ are covered by l (specified by κ_l). For example, in Figure 4.6, points $(m_1)_\alpha, (m_2)_\alpha, (m_9)_\alpha$ and $(m_{10})_\alpha$ are beyond the range l. Therefore, no information about those points is available. The implementation of θ-projection has made finding $(m_i)_c$ more difficult. This is because, different from (x, y)-projection which always follows x and y axis, θ-projection follows the minutia orientation whose angle is relatively varied (even though it is not purely random).

Suppose all $(m_i)_c$ points can be found. The adversary may use the information in ρ_c to find $\{m_i\}_{i=1}^{n}$ (refer to Section 4.2.1). However, ρ_c only contains information of the range of where a minutia point m_i is originally located, without providing its exact location information; based on this, the possibility of finding a correct minutia point coordinate is $(\frac{1}{w^2} \times 100\%)$, where w is the width of the corresponding cell. Furthermore, if there is more than one point in the cell, then all those m_i points are mapped onto the same $(m_i)_c$. Therefore, in the worst case when $\{(m_i)_c\}_{i=1}^{n_c}$ can be revealed, the minutiae points in $\{m_i\}_{i=1}^{n}$ are still safe; while the possibility of finding $\{(m_i)_c\}_{i=1}^{n_c}$ can be represented as $\frac{1}{\lfloor l/p \rfloor} \times \frac{1}{(\Sigma_{i=0}^{k_1-1} k_1) \times k_2} \times \frac{1}{w^2} \times 100\%$.

Besides its role in revoking the template, a set of keys/parameters is also useful for minimizing the inter-user similarity. Furthermore, the use of a set of keys results in creating more key spaces than that of just a key. The size of a key space is proportional to the security (non-invertibility). Nevertheless, it is predicted that there is a trade-off between this key space size (security) and the performance. Specifically, the value of α which is represented by key κ_α, may not deliver the same performance for all $0 \leq \alpha < 360^o, \alpha \in \mathbb{R}$.

In order to evaluate this condition, an experiment was performed by varying α in the first two quadrants ($0^o \leq \alpha < 180^o$). This was carried out by limiting the performance to a certain

Table 4.4: The ranges of an appropriate α value should be used in order to obtain GAR $\geq 90\%$ and FAR $\leq 10\%$. These are limited to the first two quadrants ($0^o \leq \alpha < 180^o$).

τ	Ranges of α ($\alpha \in \mathbb{R}$)
1.7	$[15^o, 70^o], [110^o, 165^o]$
2.2	$[20^o, 40^o], [140^o, 160^o]$
3	-

level, in this case is GAR $\geq 90\%$ and FAR $\leq 10\%$, whose results are provided in Table 4.4. It is found that not all of values in those two quadrants satisfy that required performance. In addition, the same experiment conducted in the other two quadrants also delivered an equivalent results. Overall, those experimental results on all quadrants (depicted in Figure 4.13, in the even that $\tau = 1.7$) recommend the ranges where the value of α should be chosen from. In other words, in order to maintain the performance, α should not be freely chosen. Implementing α beyond the ranges specified in Figure 4.13 for example, results in dropping the performance. This is because the projected point will be beyond the line lenght l of L_α as illustrated in Figure 4.14.

This restriction has reduced the α space. For example, for $\tau = 1.7$, there is a decrease of about 39% and even greater for $\tau = 2.2$. Furthermore, there is no α available for $\tau = 3$. Consequently, there is also a smaller number of κ_α available to use. It is worth talking into consideration that the effect of decreasing the α space is minimized by introducing the other keys, for example, κ_{in}.

As it has been discussed in Section 4.2.4, in order to make the vector query verifiable, its total element number must be same as that of the vector template. It is defined by $\epsilon = (l/p)$ whose value is derived from $\{\kappa_l, \kappa_p\}$. Compromising only either l or p is not enough to reveal ϵ. Nevertheless, the same ϵ can also be derived from different (κ_l, κ_p) pairs, such that $\epsilon = (l/p) = (l'/p')$ where $l \neq l', p \neq p'$. So, the adversary can obtain ϵ by compromising v or (κ_l, κ_p), or by trying all $\{l', p'\} \in \mathbb{R}$ combination.

A greater ϵ means increasing the number of possible permutation of the vector element

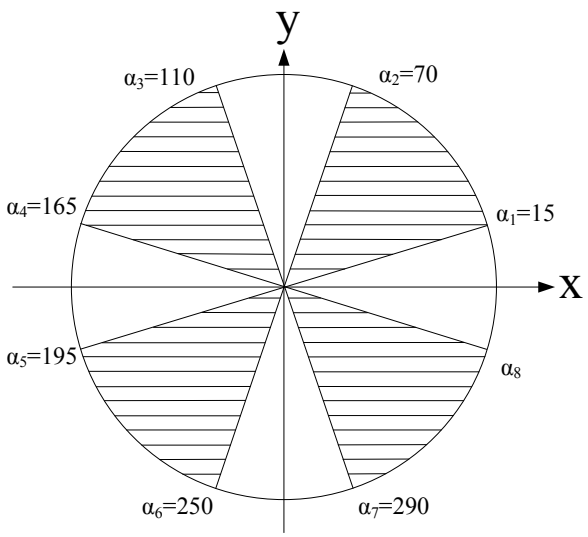


Figure 4.13: The ranges of an appropriate α value in all quadrants with $\tau = 1.7$.

indices (refer to Equation 4.6), which is proportional to the entropy ($log_2\epsilon!$). In other words, a greater ϵ enlarges the κ_{in} space, which has an effect on the difficulty in revealing B given v. In addition, a greater ϵ also means better scalability that the number of users obtaining a unique κ_{in} is likely to be higher. As the result, the authentication system is able to accommodate a larger number of enrolling users.

However, as depicted in Figure 4.15, too big ϵ decreases the performance. Figure 4.15(a) shows performances in various ϵ values when κ_p is fixed, while Figure 4.15(b) is when κ_l is fixed. Both figures show that $\epsilon = 26$ reaches GAR = 94% when FAR $\approx$ 2.3%. This GAR level can only be achieved by other ϵ values with higher FAR, for example $\epsilon = 22$ at about 3% and $\epsilon = 30$ at about 5% (refer to Figure 4.15(a)) and $\epsilon = 34$ at about 2.4% and $\epsilon = 22$ at about 5.5% (refer to Figure 4.15(b)). So, in terms of performance, $\epsilon = 26$ is better than the others. This generates $26! = 4.0329 \times 10^{26}$ indexing possibilities. Assuming that the machine can procees a million verification per second, a brute force attack will take about 6.3941×10^{12} years on average to break this κ_{in} only.

In the event that p is fixed, increasing ϵ results in increasing l. It is worth pointing out

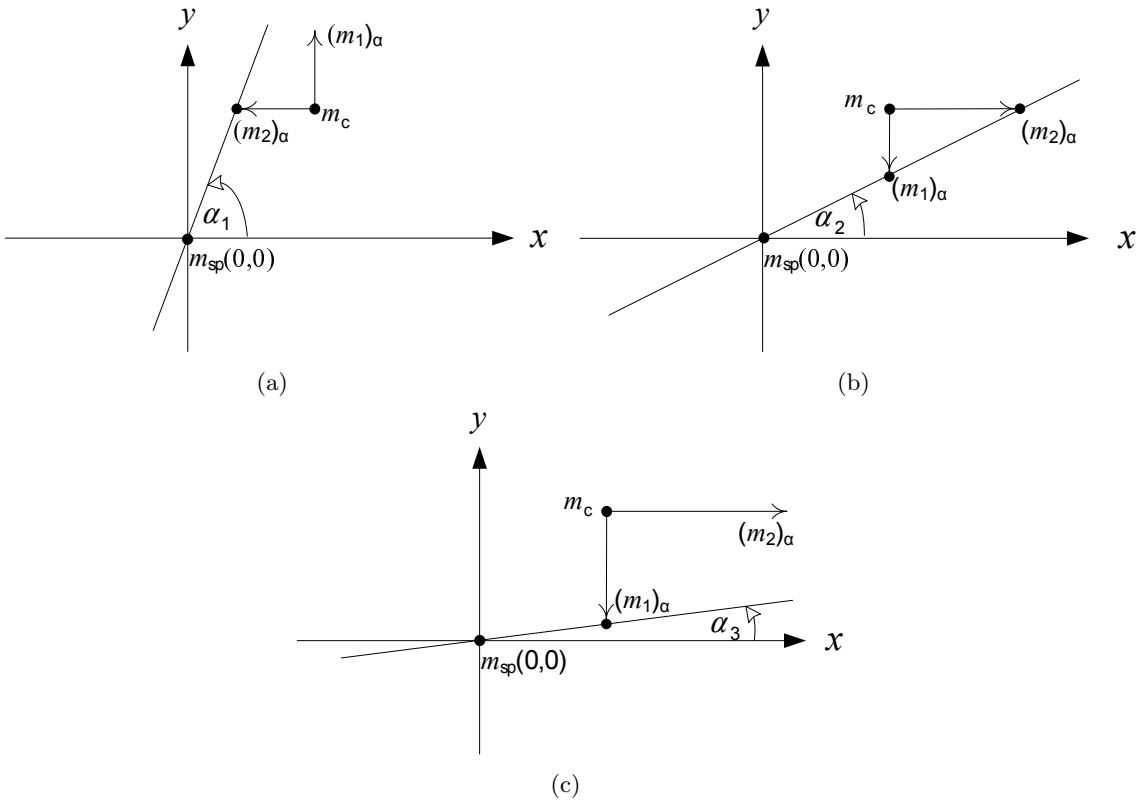
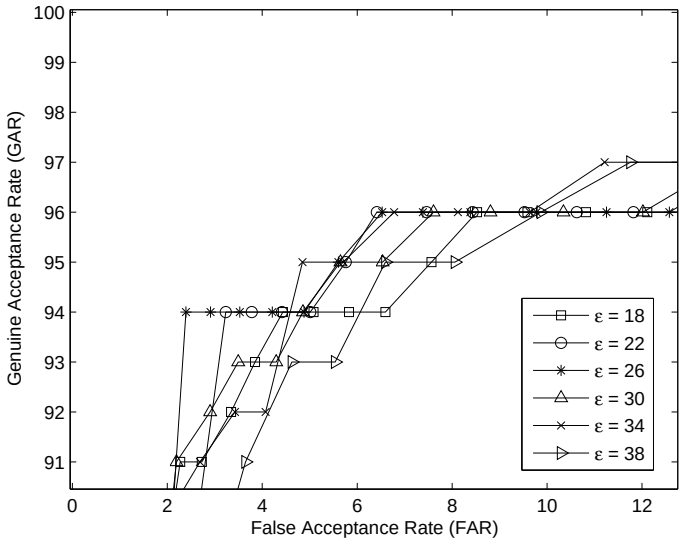
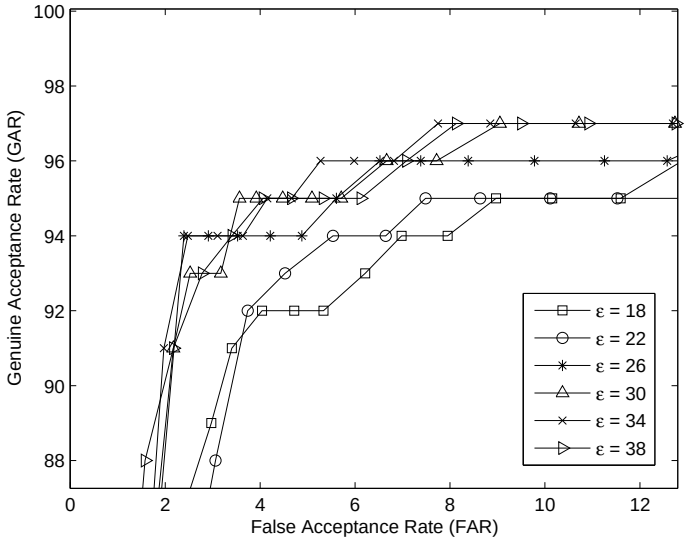


Figure 4.14: A projection line which is relatively close to either x or y axis produces points whose coordinate is beyond the coverage line. In this example, $\alpha_1 > \alpha_2 > \alpha_3$ (a) relatively close to y axis (b) relatively close to neither x nor y axis (c) relatively close to x axis.

that the length of the projection line involved in the grouping is determined by the length of each group and the total number of those groups, i.e., $l = p \times \epsilon$. The increasing of l means a larger projection space is covered. On the other hand, most minutiae points are located at 50 - 150 pixels around the core [80]. As the result, most projected points are also around the core, especially those produced by (x, y)-projection. Furthermore, on average, the center of L_α is about 47 pixel from the core. Consequently, most of those projected points are covered by L_α. In this case, $\epsilon = 26$ has been an optimal value as previously discussed. It is also found that implementing more than 26 groups in L_α (elements in v) is very likely to increase the inter-user similarity. This is because after the 26^{th}, groups in L_α mostly contain either no point (empty) or points produced by θ-projection only whose weight is less than that of



(a)



(b)

Figure 4.15: The ROC curve of various ϵ. Too low or too high ϵ decreases the performance. In this example, $\epsilon = 26$ gives better performance than 18, 22, 30, 34 or 38 for certain error levels (a) the key κ_p is fixed (b) the key κ_l is fixed.

(x, y)-projection.

Equivalently, in the case of l is fixed, increasing ϵ results in decreasing p. It means that

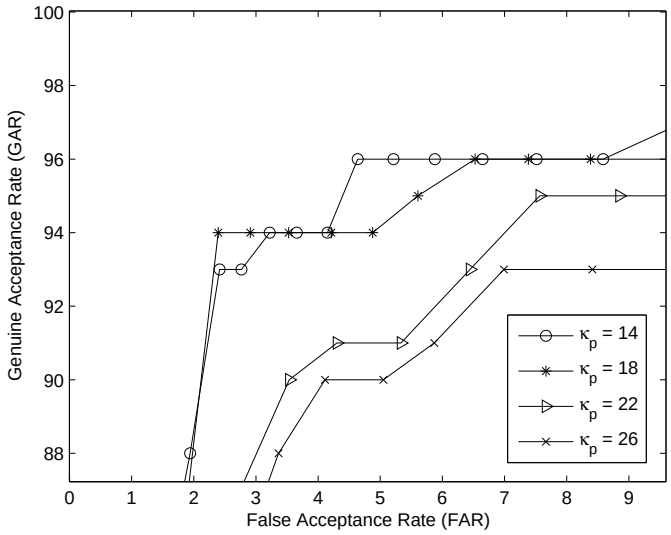


Figure 4.16: The ROC curves of various $\{\kappa_l, \kappa_p\}$ pairs, while the ϵ is fixed.

each group may contain relatively unique points. Due to intra-user variability issue, however, this causes a higher FRR and a lower GAR. On the contrary, a smaller ϵ (higher p) reduces the uniqueness of the fingerprint patterns. This leads to a higher FAR and a lower GRR.

In addition, too small or too big l and p also affects the performance as depicted in Figure 4.16, where a fixed ϵ is constructed by various (l, p) pairs. It is shown that $p = 18$ generates a better performance than that of 14, 22 or 26, especially when the FAR is between about 2% and 4%. Assigning 14 to p results in a better performance when FAR is between about 4% and 6.5%. It can also be inferred that a bigger p leads to a lower performance as depicted by $p = 22$ and $p = 26$.

Overall, it has been argued that there is a trade-off between security (non-invertibility), which is represented by the key size, and the performance, which is represented by the accuracy in particular. There are two possible options can be taken to deal with this trade-off. First, the performance is allowed to be slightly lower for compensating the security. For example, the performance is kept at GAR $\approx 90\%$ and FAR $\approx 10\%$ such that $\{\kappa_\alpha, \kappa_l, \kappa_p\}$ can

be varied. Second, the performance is maintained at the highest level by reducing the key spaces. For example, by converting the keys $\{\kappa_\alpha, \kappa_l, \kappa_p\}$ to parameters $\{\rho_\alpha, \rho_l, \rho_p\}$ such that only their optimum value is used. In this case, κ_{in} is to be the only key, as in the previous discussion.

4.4 Summary

In this chapter, a projection-based cancelable fingerprint template approach has been proposed. This utilizes the core point to be the reference to transform minutiae points. A vector string is generated which is to be the template and is stored in the database or smart card. The experimental result shows that the proposed approach meets both performance and non-invertibility requirements. More specifically, it satisfies the accuracy, revocability, diversity and changeability. It has relatively low error rates, and even lower than that of the surveyed global feature-based schemes.

As discussed and shown in the experiments, in rare cases, the core point may not be available in a fingerprint for some reasons. Failing to detect the core point caused by noises can be solved by scanning the finger several times until the expected point appears. However, this does not work on fingers whose core point is physically missing, such as happening in the arch finger class (the a priori distribution probability of arch finger class is 0.037, refer to Section 2.1). On the other hand, as a global feature-based approach, this proposed scheme relies on the existence of the core point. Therefore, in spite of its performance and security (non-invertibility) superiority, the proposed scheme experiences limitation in this certain case. Consequently, the reliability of the proposed secure authentication system is affected.

There are two possible solutions for dealing with this drawback. First, it needs to generate an alternate stable point which should be available in all fingerprint classes. The other existing singular point: delta point, is also not available in the fingerprint arch class. Moreover, a delta point is more unstable than the core point itself. Therefore, while finding new

stable points is still an issue, the other existing singular point (i.e., delta point) cannot be an alternative to replace the core point. Second, it needs to develop other cancelable fingerprint template algorithms which completely eliminate the need of the core point detection. In this case, the existence of a core point does not have an effect on the registration and verification processes. This second option is to remove the transformation dependency not only on the core point but also on the other singular points (e.g., delta point).

Chapter 5

Pair-polar Coordinate-based Transformation

Most existing fingerprint data protection methods, including feature transformation (cancelable template), rely on the global feature (i.e., core point) information. In spite of their strength, those methods do not work well if the core point is not accurately detected. On the other hand, accurate core point data is difficult to obtain [80]. Consequently, they will be core point extractor capability-dependent. A small change in the core point information can greatly affect the performance. Moreover, in some cases, the core point is not physically available [26, 116] as described in Chapter 2. The experiments conducted in Chapter 4 have shown that global feature-based protection methods suffer from the unavailability of core point information.

In order to address this problem, this chapter proposes a scheme which only employs fingerprint local features. In this scheme, each minutia point is described by its neighboring minutiae points. In particular, information of both the minutiae property and the relative position of a minutia point to other minutiae points in the polar coordinate space is explored.

This chapter is structured as follows. Section 5.1 describes the concept of polar coordinate

system-based and local feature-based transformations. Section 5.2 depicts the pair-polar transformation design. The experiment and its results are provided in Section 5.3. Finally, Section 5.4 summarizes this proposed local feature-based fingerprint data protection scheme.

5.1 Polar Coordinate System-based and Local Feature-based Transformations

5.1.1 Polar Coordinate System-based Transformation

Among transformation functions introduced by Ratha et al. [77; 80], the Cartesian transformation, based on the experimental results obtained from a private database IBM-99 [16], exhibits the lowest fingerprint matching performance. The other two transformation functions (i.e., polar and functional functions) show a slightly lower performance level than that of without transformation. In addition, based on the experimental results on the public database FVC2002Db2a [61], Jain et al. [45] report that with 5% of FAR, they are able to obtain about 92%, 97% and 98% of GAR for polar transformation, functional transformation and without transformation, respectively (there is no report about the experimental result of the Cartesian transformation). These results show that the functional transformation generates a higher performance than that of polar transformation.

Furthermore, the functional transformation is designed to solve the reordering problem experienced by Cartesian and polar transformations. This is done by folding the fingerprint image surface such that the fingerprint global minutiae structure changes but its local structure does not. In other words, the transformation crumples a sheet which contains minutiae points. Despite its high performance, the functional transformation suffers from attacks, such as one that has been described by Shin et al. [88] and Quan et al. [76]. Therefore, the functional transformation may not be appropriate to use if security and privacy are preferred. In this case, the polar coordinate system-based transformation can be an alternative, considering that its performance (presented in [80, 45]) is only slightly lower than that of

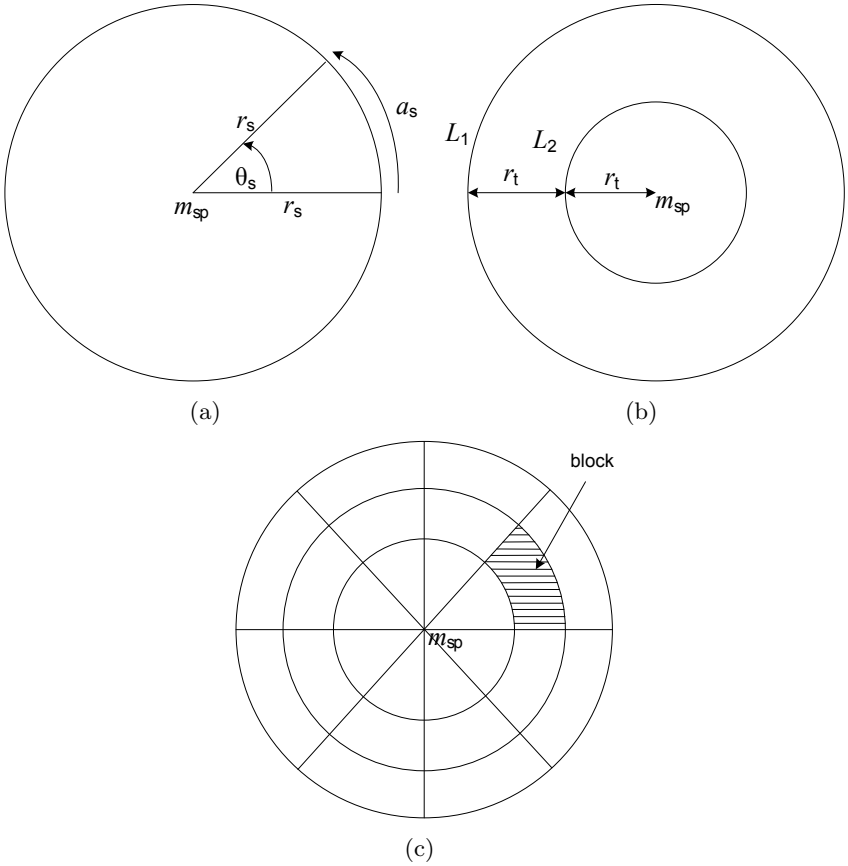


Figure 5.1: An example of sectors and tracks (a) sector (b) track (c) block.

the functional transformation. Moreover, there are still many opportunities to increase its performance by, for example, re-designing its transformation function.

In their implementation of the polar transformation function, Ratha et al. [77; 80] explored the global features, that is, the core point is to be the reference, similar to other global feature-based fingerprint data protection methods. In this polar transformation, the fingerprint space is divided into some subspaces centering at the core point, whose orientation is to be the reference to its angular coordinate. Figure 5.1 depicts the definition of subspace attributes, called *sector* and *track*. In this case, a sector refers to a subspace enclosed by an arc (a_s) and two radii (r_s) while a track refers to a subspace enclosed by circles L_1 and L_2 or by L_2 and m_{sp} (the centroid). In this figure, the size of sectors, denoted by ω_s, is determined

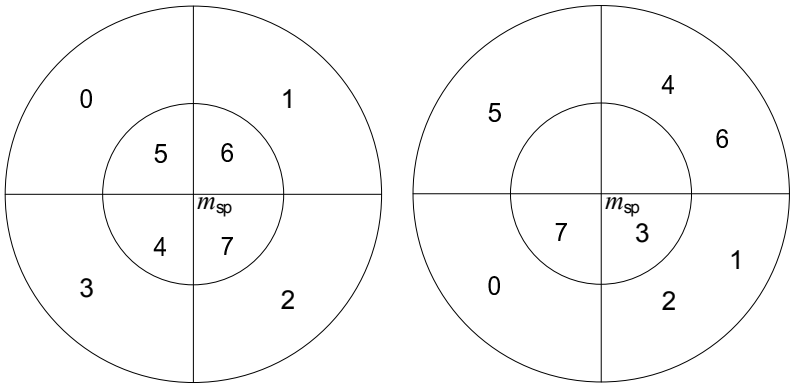


Figure 5.2: Blocks in a polar coordinate space (a) before the transformation (b) after the transformation.

by the angle θ_s, which is measured from 0^o or x-axis; and the size of tracks, denoted by ω_t, is defined by r_t. Additionally, the intersection of a sector and a track is called a block or sector-track.

Each block is given a sequential number to be its identity as depicted in Figure 5.2(a). Based on their identity number, the transformation is performed by rearranging the position of blocks, along with their corresponding minutiae points. An example of this rearrangement result is shown in Figure 5.2(b). It is possible that after the transformation, a block location is occupied by more than one blocks or even no block at all. On the other hand, it is also possible that a block remains at its original location. Therefore, in terms of this block moving, the transformation may practically result in either many-to-one or one-to-one mapping, depending on the key being used for the transformation.

Since the transformation is applied to a set of minutiae points in a block all together, the structure of minutiae points within the block is maintained, especially for a one-to-one mapping. This means that after the transformation, the relative positions of those minutiae within a block do not change. In the event of a many-to-one mapping, the structure may change because minutiae points originated from more than one block are combined.

It is worth noting that the size of each block in the same sector is not identical. This may cause a problem if a block is transformed into either a smaller or a larger block (i.e., a block in a different track), as the distances between minutiae points in this block are affected. For example, in Figure 5.2, block 3 moves to block 7 whose size is different. Overall, the block transformation is carried out based on the transformation matrix [80], such that:

$$C' = C + M \tag{5.1}$$

where C', C and M are the new block indexing number, the old block indexing number and the transformation matrix, respectively. For example, given C = [0 1 2 3 4 5 6 7] and M = [+3 +1 +0 +4 -3 -5 -5 -3]. The transformed block C' is [3 2 2 7 1 0 1 4] whose illustration is depicted in Figure 5.2. It is also shown that after the transformation, more than one block may occupy the same block (e.g., blocks 4 and 6 to 1) and there are blocks which do not move at all (e.g., block 2). In the event that transformed blocks must not remain in their original location, the transformation matrix must not contain 0.

Despite its good result in satisfying the non-invertibility requirement, the polar transformation design in [77, 80] has two potential drawbacks. These relate to the discriminability property caused by the intra-user variability. First, in the subsequent fingerprint scanning, minutiae points, particularly the ones which are close to the block border, may be located in different blocks. Second, as it has been discussed in Chapter 2 and shown in Chapter 4, in certain situations, the core point may not be extracted from fingerprints, especially from an arch fingerprint class. Therefore, this polar transformation concept does not work at a certain fingerprint class.

5.1.2 Local Feature-based Transformation

Unlike a global feature-based design (in which the polar coordinate-based transformation was originally implemented), a local feature-based design only explores the minutia point structure. Instead of the core point, local feature-based transformations utilize minutiae points themselves as the reference to the transformation, such as that implemented by Lee and Kim [56]. In their transformation, each minutia point is subsequently selected as a reference to its neighboring minutiae points. In general, local features are relatively invariant to translation and rotation, compared with the global features. Nevertheless, the superiority of local features may also depend on the quality of fingerprint images, such as shown in [57]. The local features can be constructed by either the relation between minutiae points or the property of minutiae points itself, such as the orientation information.

In order to increase the accuracy of the matching process in the local feature-based transformation (e.g., reducing the effect of the intra-class variability), the relation between minutiae points can be constructed in several layers (hierarchical). This means that verification of a minutia point in the query is carried out by comparing its neighboring points with those in the template, and those neighboring points are in turn verified by comparing their neighboring points. An example of this is given in Figure 5.3. Let m_1 in Figure 5.3(a) and m'_1 in Figure 5.3(b) be the minutiae points being verified. Their respective neighboring points, m_2, m_3, m_4, m_5 and $m'_2, m'_3, m'_4, m'_5, m'_6$ are compared. Suppose m_2 matches with m'_2. These two minutiae points are verified by comparing their neighboring points, m_{2a}, m_{2b}, m_{2c} and $m'_{2a}, m'_{2b}, m'_{2c}$. Similar comparisons apply to the other neighboring points. If those comparisons satisfy the threshold, then m_1 matches with m'_1. This concept has been implemented for a fuzzy vault scheme [50] by Xi and Hu [108] in their composite feature-based design. The disadvantage is that this hierarchical verification increases the computation level.

Instead of individually verified as in the previous approaches, minutiae points can be

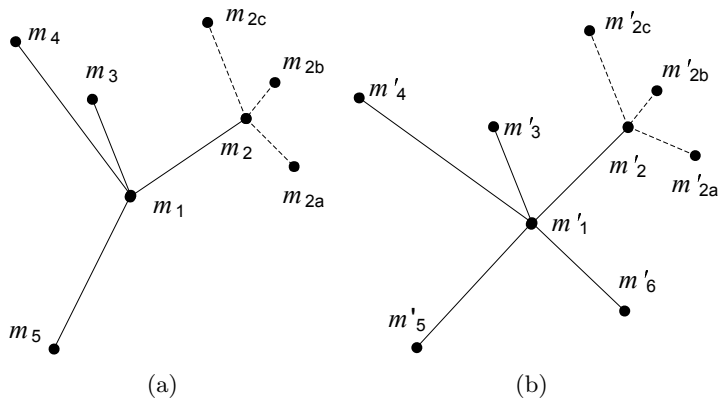


Figure 5.3: Hierarchical minutiae point verification (a) template (b) query.

concurrently verified in a group. For example, every three minutiae points are grouped to form a triangle from which the feature representation is obtained [49, 6, 34, 14, 40]. However, in some cases, this approach eliminates the individual property of minutiae such that it reduces the uniqueness of fingerprints. As a result, it can cause an FRR increase.

5.2 Pair-polar Transformation Design

Based on that polar coordinate-based transformation design and those characteristics of fingerprint local features, the pair-polar coordinate-based transformation is proposed. This approach is inspired by research conducted, specifically that in [108, 56, 6, 77] and [80].

In general, this approach takes the input only from a set of selected minutiae points. As in the concept of cancelable template, matching between fingerprint template and the fingerprint query is carried out in the transformed domain; therefore, in case the template is compromised, the fingerprint data is still safe. The overall process of this approach is depicted in Figure 5.4, which consists of the following stages:

1. Minutia point selection.

2. Template generation.

3. Minutia point comparison (fingerprint matching).

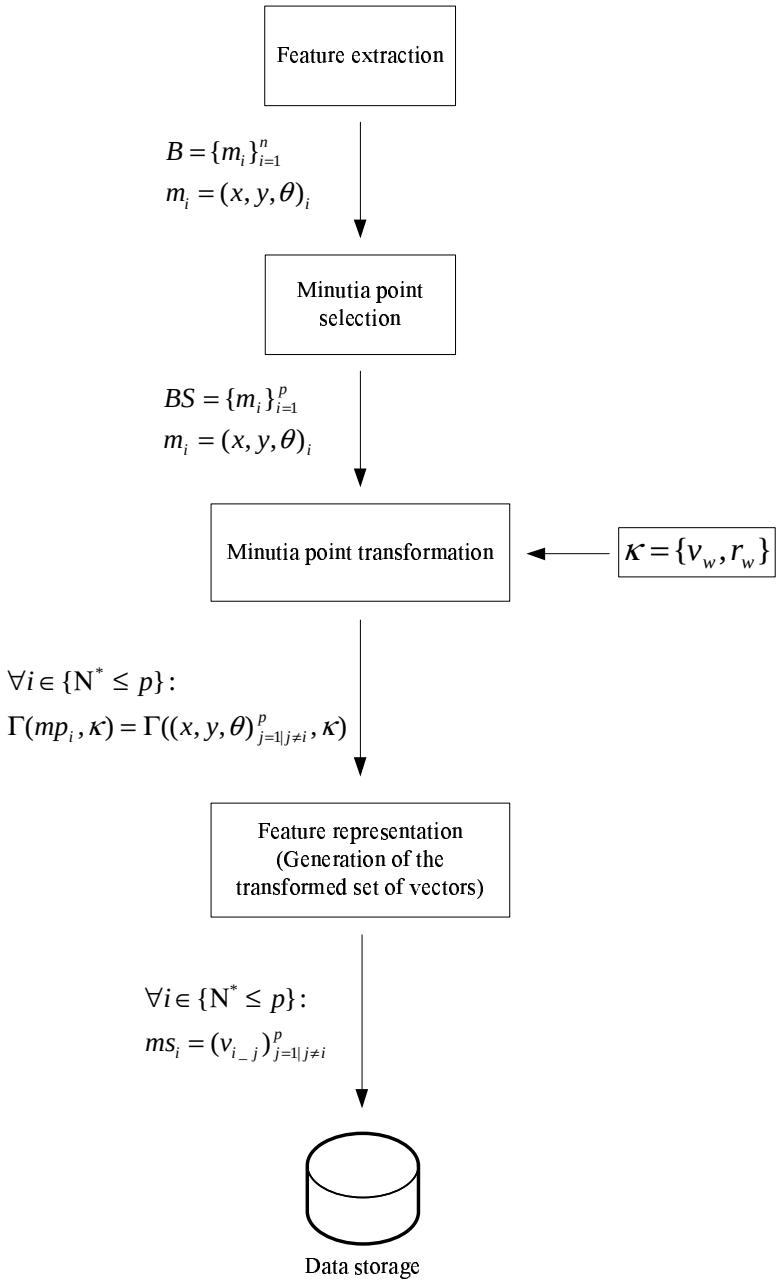


Figure 5.4: The pair-polar coordinate-based transformation architecture.

5.2.1 Minutia Point Selection

Suppose B is a set consisting of n minutiae points extracted from a fingerprint, as denoted in Equation 3.2; BS is the subset of B consisting of at least k minutiae points whose distances to other minutiae points are greater than the defined threshold τ_1; p is the factual total minutiae number in BS, and $dis(m_i, m_j)$ is the distance between minutiae m_i and m_j. The set BS, which is to be the input to the transformation function, is denoted as:

$$BS = \{m_i\}_{i=1}^{p}, k \leq p \leq n \tag{5.2}$$

Minutia m_j will be included in BS if it satisfies the requirement in Equation 5.3. The details of this minutia point selection process is depicted in Algorithm 5.1.

$$dis(\{m_i\}_{i=1}^{j-1}, m_j) > \tau_1, 1 < j \leq p \tag{5.3}$$

If a fingerprint image fails to generate at least k points, it is ineligible to be either a secure template or a secure query. In this case, τ_1 and k may be changed to accommodate the corresponding fingerprint image, or even the original fingerprint image may be used to avoid failure to enroll (FTE).

Similar to the minutia point selection process in [65], which is used for the fuzzy vault [50] implementation, in this feature transformation (cancelable template) design, the distance between minutia m_i and minutia m_j is determined by considering both the minutia coordinate and minutia orientation. This definition can be denoted in Equation 5.4.

$$dis(m_i, m_j) = t_1 \times \Delta r + t_2 \times \Delta a \tag{5.4}$$

where $\Delta r = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$, $\Delta a = min(|\theta_i - \theta_j|, (360 - |\theta_i - \theta_j|))$, $t_1 = 1$ and $t_2 = 0.2$.

Algorithm 5.1 Select minutiae points from a fingerprint

Input: B
Output: BS

$s \leftarrow 1$
while $s \leq total\_minutiae\_in\_B$ **do**
  $m_i \leftarrow m_s$
  $BS \leftarrow m_s$
  increment p
  if $dis(m_i, m_j) > \tau_1$ **then**
    for $r \leftarrow 1$ to p **do**
      if $dis(m_j, m_r) \leq \tau_1$ **then**
        break
      end if
    end for
    $BS \leftarrow m_j$
    increment p
  end if
  if $p \geq k$ **then**
    break {total number of selected minutiae is greater than threshold}
  else
    increment s
    $BS \leftarrow \emptyset$ {reset BS}
  end if
end while

5.2.2 Template Generation

Generally speaking, a secure fingerprint template is developed after the minutiae points are transformed and their features are represented as a vector. This template development is performed according to the set of selected points (BS) obtained from the previous step. Each point in the set holds a descriptor which contains information about its relative position to the neighboring points. This information is stored in the form of vectors [108] whose details are provided as follows.

Vector Definition

Suppose m_i is the minutia point being processed and m_j is a neighboring minutia point of m_i. The minutia m_i is positioned at the center of the polar coordinate space, whose orientation acts as the 0^o axis and is to be the reference to both radial and angular distances of m_j. The relation between m_i and m_j is represented by vector v_{i_j} whose definition is provided in Equation 5.5 and depicted in Figure 5.5(a).

$$v_{i\_j} = (r_{i\_j}, \alpha_{i\_j}, \beta_{i\_j}) \tag{5.5}$$

The elements of vector v_{i_j} are described as follows:

- r_{i_j} : the radial distance between the center $m_i(0,0)$ and a neighboring minutia $m_j(x_j, y_j)$ such that $r_{i\_j} = \sqrt{x_j^2 + y_j^2}$
- α_{i_j} : the angle between the orientation of the center m_i and the edge r_{i_j} in the counterclockwise direction such that $\alpha_{i\_j} = arctan(\frac{y_j}{x_j})$
- β_{i_j} : the angle between the orientation of the neighboring minutia m_j and the edge r_{i_j} in the counterclockwise direction such that $\beta_{i\_j} = arctan(\frac{y_i'}{x_i'})$

From this definition, it can be inferred that $r_{i\_j} = r_{j\_i}$, $\alpha_{i\_j} = \beta_{j\_i}$ and $\beta_{i\_j} = \alpha_{j\_i}$.

The center m_i and each of its neighboring minutiae (represented by m_j) in BS, $\{m_j\}_{j=1|j\neq i}^{p}$, where p is the total number of minutiae in BS, form a set of minutiae points mp_i. Each of these m_i and m_j pairs constructs a vector v_{i_j}. Since all the minutiae points in BS are to be the center, each minutia point has a set of $(p-1)$ v_{i_j} vectors. Suppose ms_i is the set of vectors constructed by the center m_i, the non-transformed template of BS is represented by:

$$\forall i \in \{\mathbb{N}^* \leq p\} : ms_i = \{v_{i\_j}\}_{j=1|j\neq i}^{p} \tag{5.6}$$

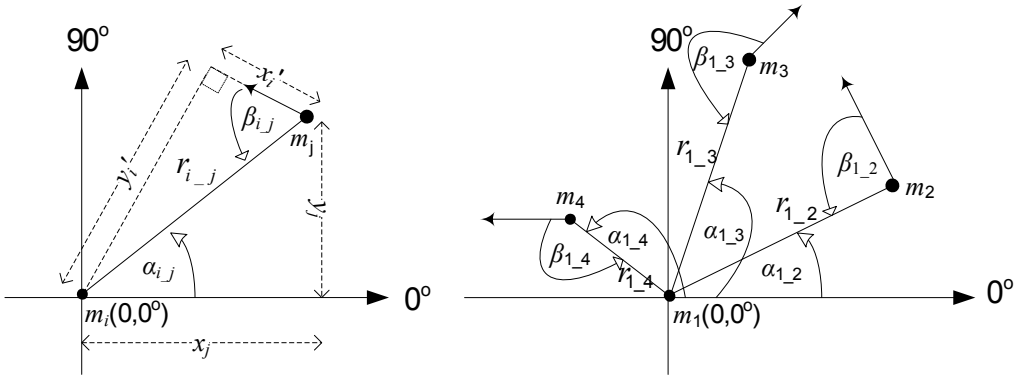


Figure 5.5: Vector generation process in the polar coordinate system (a) definition of vector properties (b) the example of vector set of points, $ms_1 = \{v_{1\_2}, v_{1\_3}, v_{1\_4}\}$.

Referring back to the assumption that each minutia point in BS has $(p-1)$ vectors; this implies that the location information of each minutia point is determined by its $(p-1)$ neighboring minutiae. The example of ms_i construction process, as presented in Figure 5.5(b), can be illustrated as follows. Suppose there are four minutiae points in BS ($p = 4$), i.e., $BS = \{m_1, m_2, m_3, m_4\}$, where m_1 is the minutia point being processed, m_2, m_3 and m_4 are the neighboring minutiae points of m_1. The set of vectors of m_1 is $ms_1 = \{v_{1\_2}, v_{1\_3}, v_{1\_4}\}$, where $v_{1\_2} = (r_{1\_2}, \alpha_{1\_2}, \beta_{1\_2})$, $v_{1\_3} = (r_{1\_3}, \alpha_{1\_3}, \beta_{1\_3})$ and $v_{1\_4} = (r_{1\_4}, \alpha_{1\_4}, \beta_{1\_4})$. Equivalently, $ms_2 = \{v_{2\_1}, v_{2\_3}, v_{2\_4}\}, ms_3 = \{v_{3\_1}, v_{3\_2}, v_{3\_4}\}$ and $ms_4 = \{v_{4\_1}, v_{4\_2}, v_{4\_3}\}$. The set $\{ms_1, ms_2, ms_3, ms_4\}$ has been the non-transformed (insecure) template of BS. In turn, this set has also been the non-transformed template of B because BS itself is the representation of B.

Transformation

The transformed (secure) fingerprint template and query are generated by re-arranging the minutiae points in the polar coordinate space according to the transformation function and the set of keys being used. For this transformation process, as in Ratha et al. [77; 80], the

polar coordinate space is divided into blocks. Different from [77, 80], however, blocks are developed with respect to the reference point m_i because this transformation design does not employ the core point. In this case, there is one track being generated; therefore a block and a sector refer to the same definition and those two terms are interchangeable.

The general process of the transformation is denoted as follows:

$$BS_{sec} = \Gamma(BS, \kappa) \tag{5.7}$$

where BS_{sec} is the result of the transformation of BS, Γ is the transformation function and κ is the set of keys being used in the transformation. BS_{sec} is either the template being stored in the database or the query to be authenticated with the corresponding stored template. It contains sets of vectors which describe transformed minutiae points in BS. Additionally, the process of generating BS_{sec} is provided in Algorithm 5.2. In case the non-transformed template or query is preferred, the transformation step (line 2 - 8) of Algorithm 5.2 is skipped.

In the transformation, minutiae points are re-arranged in two directions, those are: (i) angular direction, which is performed by transforming the minutia m_j according to its corresponding sector with respect to the orientation of m_i; (ii) radial direction, which is performed by modifying the distance between m_j and m_i.

The angular transformation (sector transformation) is carried out by considering a random vector v_w, which specifies the location to where a sector is mapped. All minutiae points within each sector are moved according to the following equation:

$$new\_sect = \text{abs}(old\_sect + v_w)\text{mod}(total\_sect) \tag{5.8}$$

where *new_sect*, *old_sect* and *total_sect*, are indices of transformed sectors, indices of original (non-transformed) sectors and the total number of sectors, respectively .

The radial transformation is performed by utilizing the transformed-radial factor r_w. In

Algorithm 5.2 Transform minutiae points using Pair-polar method

Input: BS
Output: BS_{sec}

for $i \leftarrow 1$ to $total\_minutiae\_in\_BS$ **do**
  {Transformation}
  for $s \leftarrow 0$ to $total\_sector$ - 1 **do**
    $angular\_transformation$
    for $j \leftarrow 1$ to $total\_minutiae\_in\_sector\_s$ **do**
      $radial\_transformation$
    end for
  end for

  {Generating minutiae vectors}
  for $j \leftarrow 1$ to $total\_minutiae\_in\_BS$ **do**
    if $j \neq i$ **then**
      $ms_i \leftarrow v_{i\_j}$
    end if
  end for
  $BS_{sec} \leftarrow ms_i$
end for

particular, the radial distance of points in the specified transformed sector locations, which is represented by r_{i_j}, is modified. Let r_{i_j} and r'_{i_j} be the radial distance of before and after the radial transformation, respectively, and μ be a variable being used for the modulo operation. This radial transformation is denoted by:

$$r'_{i\_j} = \frac{(r_{i\_j} \times r_w) \mathrm{mod}(\mu)}{r_w} \tag{5.9}$$

From the transformation functions in Equations 5.8 and 5.9, there are variables whose values can be varied such that an insecure fingerprint can be transformed into some different secure ones. Specifically, the combination of v_w, r_w and μ are applied to the transformation keys. Therefore, every transformation Γ needs a set of keys $\kappa = \{v_w, r_w, \mu\}$. In order to eliminate the possible linkage in BS_{sec}, each $\{ms_i\}_{i=1}^{p}$ is transformed by using a different κ.

5.2.3 Minutia Point Comparison (Fingerprint Matching)

As discussed in the previous sections, matching process (verification) between a fingerprint template and a fingerprint query is done in the transformed domain, whose inputs are sets consisting of certain (selected) minutiae points only. Although the fingerprint template and the fingerprint query are derived from the same finger, it is still possible that those sets contain relatively different minutiae points. In order to address this possibility, some thresholds are defined in the matching process to eliminate the effect of those non-overlapping minutiae points on the matching result.

It is worth pointing out that the proposed matching algorithm is independent of the minutiae selection and transformation designs. Therefore, it can be implemented with or without those two steps, as long as the data format is identical. Also, this matching design has made it possible to combine the matching algorithm with other transformation algorithms.

Verification of the fingerprint query is accomplished in two comparison levels: point and vector. Specifically, the verification is done by comparing each point in the query with all points in the template whilst the point comparison itself is performed by comparing each vector in the vector set associated with a point in the query with all vectors in the vector set associated with a point in the template (an example is depicted in Figure 5.6). In order to be recognized as a matched vector pair, a vector in the query and its corresponding vector in the template have to satisfy the following conditions:

1. Their similarity level is greater than or equal to the specified thresholds.

2. Their similarity level is greater than when they are paired to other vectors.

Equivalently, these conditions also have to be fulfilled by the point in the query and its corresponding point in the template to allow them to be recognized as a matched point pair.

Different from that in [108], this matching algorithm utilizes neither the conditional matched state nor the primary matching rate for comparing minutiae points. Instead, a com-

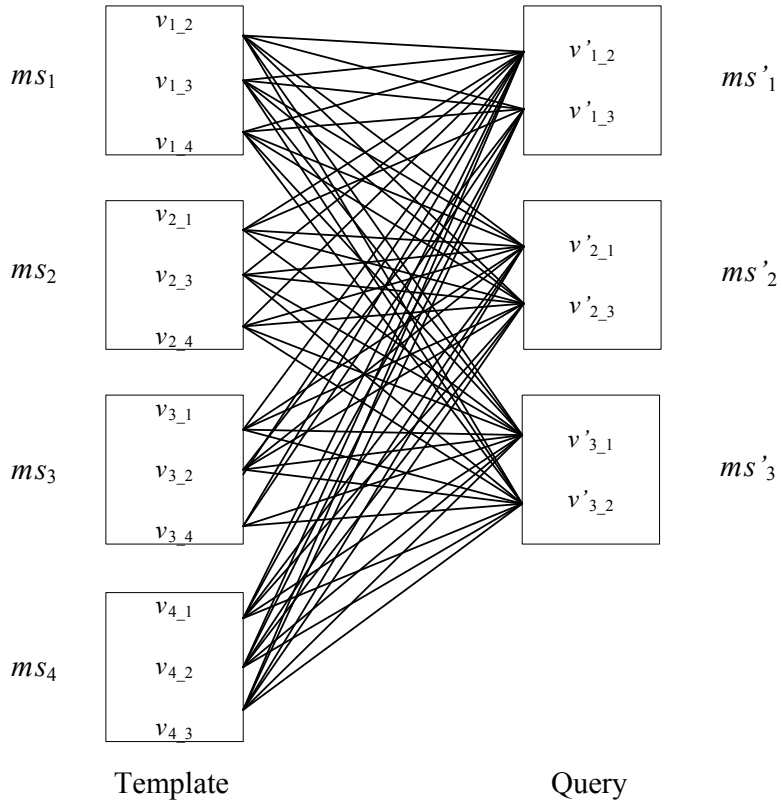


Figure 5.6: An example of a fingerprint verification process. The template and the query consist of four and three minutiae points, respectively. Verification is carried out by implementing a many-to-many comparison to their vectors.

plement similarity threshold is applied. Moreover, in both transformed and non-transformed domains, all neighboring points are counted. In other words, a point descriptor is constructed by all points derived from those in BS (for a transformed template) or all points in BS (for a non-transformed template) except the point being compared itself (the center of the polar space, m_i). Therefore, regardless of their location in the polar coordinate space, all neighboring points equally contribute to the authentication result.

The matching procedure, including some further refinement of the algorithm in [108], is presented in Algorithm 5.3. This covers those two comparison levels which are carried out sequentially. By referring to Figure 5.7, this matching procedure along with its example can

Algorithm 5.3 Match the query to the template

Input: BS'_{sec}, BS_{sec}
Output: R

for $i' \leftarrow 1$ to p' **do**
  for $i \leftarrow 1$ to p **do**
    if $type'_i = type_i$ **then**
      compare ms'_i to ms_i
      remove_duplicate_matched_vectors
      if $total\_matched\_vectors \geq \lambda$ **then**
        $possibly\_matched\_points \leftarrow \{m_i, m'_i\}$
      end if
    end if
  end for
end for
for $i \leftarrow 1$ to $total\_possibly\_matched\_points$ **do**
  remove_duplicate_possibly_matched_points
  $total\_matched\_points \leftarrow non\_duplicate\_possibly\_matched\_points$
end for

{Authentication decision}
if $total\_matched\_points \geq \eta$ **then**
  $R \leftarrow 1$ {matched}
else
  $R \leftarrow 0$ {not matched}
end if

be illustrated as follows.

Constructing vector sets. The sets of vectors corresponding to each point is constructed according to the vector definition in Equation 5.5. Denote BS and BS' as the set of selected minutiae template and query, respectively; p and p' the total number of minutiae points in BS and BS', respectively. This generates the vector sets $\{ms_i\}_{i=1}^{p}$ for the fingerprint template and the vector sets $\{ms_i\}_{i=1}^{p'}$ for the fingerprint query.

Let m_i and m'_i be respectively the minutia points in the template and in the query being processed by the matching module. These minutiae points are located at the center of the polar coordinate space. From the example shown in Figure 5.7, it can be deduced that the template has $p = 6$, $i = 1$ whilst the query has $p' = 5$, $i' = 1$. Accordingly, m_1 has five and

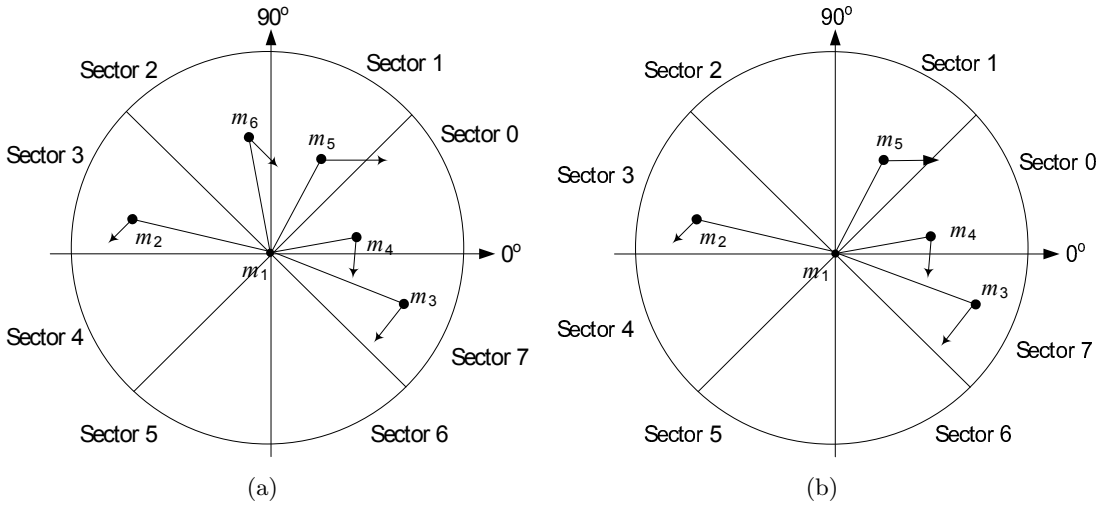


Figure 5.7: An example of matching process of two transformed fingerprint data (a) template (b) query.

m'_1 has four neighboring points. In addition, the corresponding vector sets of m_i and m'_i are $\{v_{1\_2}, v_{1\_3}, v_{1\_4}, v_{1\_5}, v_{1\_6}\}$ and $\{v'_{1\_2}, v'_{1\_3}, v'_{1\_4}, v'_{1\_5}\}$, respectively.

Comparing vector sets. This step is performed by firstly checking the type of the point pair being processed (i.e., m_i and m'_i). If both of them are of the same type, then the vector matching is proceeded by comparing each vector in $\{ms'_i\}$ with all vectors in $\{ms_i\}$. This m_i and m'_i matching step will generate $((p'-1)\times(p-1))$ comparisons. In the event that the type of m_i and m'_i is different, the next query point $(i'+1)$ is taken and the matching process is back to the previous *constructing vector sets* step. In this thesis, only two common minutiae types are used: ridge ending and bifurcation.

Suppose that both m_1 and m'_1 in Figure 5.7 have the same minutia type. Each query vector v'_{1_k}, where $2 \leq k \leq 5$ is compared with all template vectors v_{1_j}, where $2 \leq j \leq 6$.

Referring back to the vector definition in Equation 5.5, the difference (similarity) between v' of ms'_i and v of ms_i can be represented as vector difference components: $\Delta r_{i,k\_i,j}$, $\Delta\alpha_{i,k\_i,j}$,

$\Delta\beta_{i,k\_i,j}$, whose definition is provided as follows:

$$\left.\begin{aligned}\Delta r_{i,k\_i,j} &= \frac{|r'_{i\_k}-r_{i\_j}|}{r_{i\_j}} \times 100\% \\ \Delta\alpha_{i,k\_i,j} &= \frac{min(|\alpha'_{i\_k}-\alpha_{i\_j}|,360-|\alpha'_{i\_k}-\alpha_{i\_j}|)}{360} \times 100\% \\ \Delta\beta_{i,k\_i,j} &= \frac{min(|\beta'_{i\_k}-\beta_{i\_j}|,360-|\beta'_{i\_k}-\beta_{i\_j}|)}{360} \times 100\%\end{aligned}\right\} \tag{5.10}$$

Finding pair-matched vectors. Finding the pair-matched vector is performed based on the vector difference level, which has been defined in the previous step. It can be expected that smaller differences leads to a higher possibility to be a pair-matched vector.

In order for a vector v'_{i_k} to match with v_{i_j}, all their vector component differences, represented by $\{\Delta r_{i,k\_i,j}, \Delta\alpha_{i,k\_i,j}, \Delta\beta_{i,k\_i,j}\}$ in Equation 5.10, have to satisfy the following requirements [108]:

$$\left.\begin{aligned}\Delta r_{i,k\_i,j} &< \tau_r \\ \Delta\alpha_{i,k\_i,j} &< \tau_\alpha \\ \Delta\beta_{i,k\_i,j} &< \tau_\beta\end{aligned}\right\} \tag{5.11}$$

where $\tau_r, \tau_\alpha, \tau_\beta$ are the thresholds of each vector difference component. Accordingly, their total difference, represented by Δf, has to satisfy:

$$\left.\begin{aligned}\Delta f &\leq \tau_2 \\ \Delta f &= \Delta r_{i,k\_i,j} \times \omega_r + \Delta\alpha_{i,k\_i,j} \times \omega_\alpha + \Delta\beta_{i,k\_i,j} \times \omega_\beta\end{aligned}\right\} \tag{5.12}$$

where τ_2 is the threshold of the total vector difference; $\omega_r, \omega_\alpha, \omega_\beta$ are the weight factors of the corresponding vector difference components.

If the requirements in Equations 5.11 and 5.12 are satisfied, then v'_{i_j} possibly matches with v_{i_j}. The matched vector relation between ms'_i and ms_i itself is one-to-one. This means that a vector in ms'_i cannot have more than one pair-matched vector in ms_i. Likewise, a vector in ms_i cannot have more than one pair-matched vector in ms'_i. Considering that there

is a possibility of some vector pairs satisfying the thresholds (see Equations 5.11 and 5.12), only the vector pair whose Δf is the least is selected, as depicted in Figure 5.8. It means that in every ms'_i and ms_i verification, there are at most $min((p'-1),(p-1))$ vector-matched pairs generated. In other words, the relation between ms'_i and ms_i is injective but is not necessarily bijective.

If there are at least λ vector-matched pairs between ms'_i and ms_i, then the point m'_i possibly matches with the point m_i. This vector comparison is carried out such that all vector sets in $\{ms'_i\}_{i=1}^{p'}$ and $\{ms_i\}_{i=1}^{p}$ are processed.

Determining whether the query matches with the template. Assuming that there are q possibly point-matched pairs between BS' and BS have been found in the previous step. Due to the one-to-many, many-to-one or many-to-many minutia point comparison, there may be duplicate matched points among those q pairs. Similar to the previous duplicate pair-matched vector case, those duplicate matched points are also removed to obtain the actual (non-duplicate) pair-matched points. This is done according to the following criteria:

1. The total number of the corresponding vector-matched pairs. Only the minutia point with the highest number of vector-matched pairs is selected and classified as the pair-matched point. If this highest number is obtained by more than one pair points, then

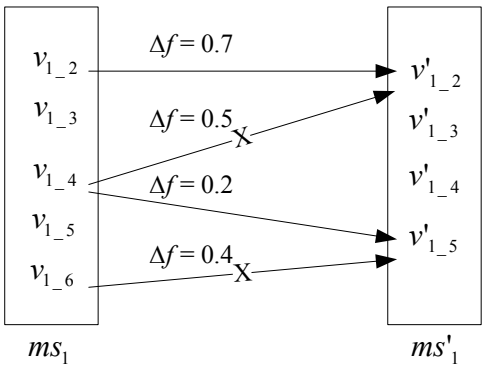


Figure 5.8: The pair-matched vectors of ms_1 and ms'_1 are (v_{1_2}, v'_{1_2}) and (v_{1_4}, v'_{1_5}).

.

proceed to the next criterion.

2. The level of Δf values (Equation 5.12) of each vector-matched pair associated with the corresponding point-matched pair. Denote f_v and L an array vector and its length (the number of elements in f_v), respectively. Each Δf is mapped according to its level onto f_v. In this case, *level* is defined as the specified range of values to which Δf is classified. For example, suppose L is 5 (f_v has 5 elements) and there are 7 vector-matched pairs in the corresponding point-matched pair whose set of Δf values are {0.8, 2.7, 0.2, 1.6, 2.4, 1.9, 0.1}. The first element of f_v is determined by the first level of L, that is, the subset of Δf whose values fall into [0,1). This results in {0.8, 0.2, 0.1}. In this case, the subset has 3 members; therefore, the first element of f_v is set to 3. By the same token, the subset {1.6, 1.9} is in [1,2), which is the second level of L. So, the second element of f_v is set to 2. Likewise, the third, fourth and fifth elements are 2, 0, and 0, respectively. This is because {2.7, 2.4} is in [2,3) and no value in the Δf set is in both [3,4) and [4,5). So, for this set of Δf and L, the resulted array vector is $f_v = (3, 2, 2, 0, 0)$. Based on the fact that the smaller Δf the more similar the points, an array vector f_v whose first element is the highest is selected. If this highest number is obtained by more than one array vectors, then that with the highest second element is selected, and so on. Similar to the first criterion, if there are more than one point-matched pairs having exactly the same f_v element values, then proceed to the next criterion. Otherwise, the pair-matched points have been found.

3. The average of Δf. Only the point-matched pair with the smallest average Δf value is selected. If there are more than one point-matched pairs having exactly the same average of Δf, then any of those pairs can be used. Nevertheless, this is unlikely to happen because the average of Δf should be different for a certain digit number.

From these criteria, it can be inferred that the actual pair-matched points should have as

many as possible pair-matched vectors. The process of removing duplicate pair-matched points is analogous to that of removing duplicate pair-matched vectors, which has been illustrated in Figure 5.8. Additionally, if there are at least η actual (non-duplicate) point-matched pairs between BS' and BS, then the fingerprint query B' is classified as matching with the fingerprint template B.

5.3 Experiments and Analysis

The proposed approach is evaluated according to the scenarios which have been described in Section 3.3: accuracy, revocability, diversity and changeability. This is done by measuring the degree of similarity between the template and the query. In addition, the security (non-invertibility) of the proposed approach is also provided.

The minutia selection process (refer to Section 5.2.1) has been able to reduce the number of minutiae points in each fingerprint to about 65% of the original number. The set of selected minutiae points (BS) is to be the input to the transformation function. For the transformation itself, the parameters are obtained by deriving them from [65, 108] as well as by doing an experiment on a smaller database of FVC2002Db2a, comprising both genuine and imposter testings to find the best value combination. This smaller database comprises 10 randomly selected image pairs. Once the required minimum performance level has been obtained, the corresponding parameter values are implemented in the formal experiment. The experimental result of this small database (represented by an EER value) corresponds to that of the bigger database. In more detail, these experimental results are plotted on an ROC curve (depicted in Figure 5.9). It is shown that the combination between $\tau_2 = 4$ and $\lambda = 6$ delivers the best performance, especially for small FAR. For example, when FAR is less than about 1%, the parameters ($\tau_2 = 4, \lambda = 6$) achieve about 90% of GAR, which is higher than the others. Moreover, from FAR $\approx$ 1% to FAR $\approx$ 8%, GAR is the highest. Hence, the parameter pair ($\tau_2 = 4, \lambda = 6$) is implemented to the formal testing on all scenarios and

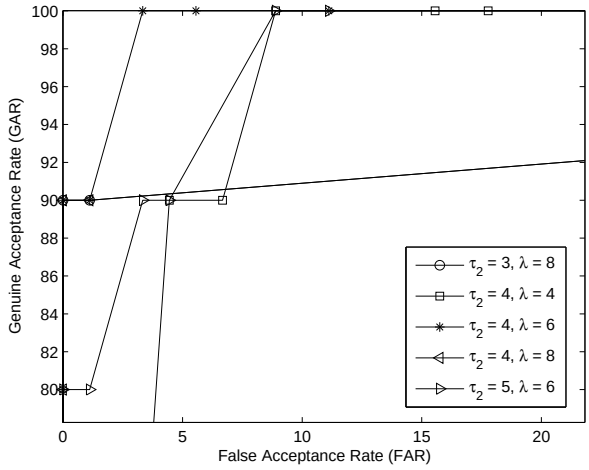


Figure 5.9: ROC in the transformed domain of various τ_2, λ combination using small database.

databases (FVC2002Db1a, FVC2002Db2a and FVC2002Db3a [61]).

5.3.1 Accuracy

As described in the evaluation design (refer to Chapter 3), it is assumed that the key is compromised. The experimental results, which are obtained from FVC2002Db2a [61], show that the EER of both non-transformed and transformed fingerprints are 5% and 6%, respectively, whose graphs are provided in Figure 5.10. It can be inferred that the transformation has increased the EER by 1%, which is relatively small comparing to that of other transformation functions; for example, about 13% in [8] (the EER of non-transformed and transformed fingerprints are 4% and 16.8%, respectively). Furthermore, as summarized in Table 5.1, 6% of EER is lower than that obtained by using other parameter settings (more detail acceptance rates are depicted in Figure 5.11). In addition, this EER itself is also lower than that of most other transformation functions, for example, 6.8% of [56] and more than 10% of [49].

When the experiment was performed in FVC2002Db1a, the EERs obtained from non-transformed and transformed fingerprints are 3% and 9%, respectively; whilst those of

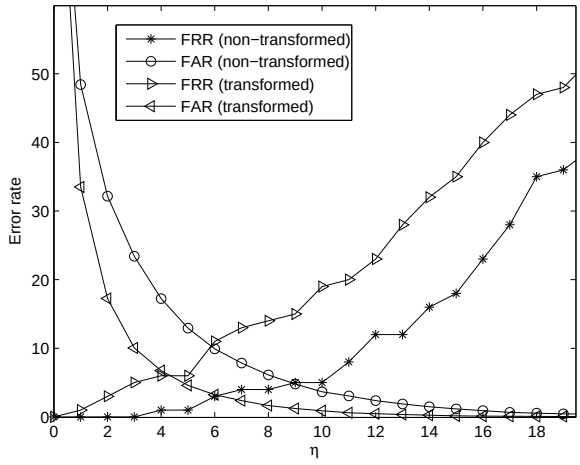


Figure 5.10: Equal Error Rate (EER) in both transformed and non-transformed domains.

Table 5.1: EER obtained by varying the parameters.

(τ_2, λ)	EER (%)
(2,7)	16
(4,4)	15
(4,6)	6
(4,8)	7.8
(6,7)	8
(6,8)	8

FVC2002Db3a are 16% and 27%, respectively. All these EERs of transformed fingerprints are higher than that of FVC2002Db2a. As discussed in Section 4.3.1, the number of extracted minutiae points from FVC2002Db1a images is smaller than that from FVC2002Db2a; while that from FVC2002Db3a is even smaller than that from FVC2002Db1a. Furthermore, 2% of fingerprint image pairs in FVC2002Db3a are unextractable, which result in failure to enroll. It also means that, in this case, fingerprint transformation and authentication processes do not work. According to the assumption that in the real world users have willingness to authenticate (provided in Section 3.3.1), the experimental results obtained from FVC2002Db2a are more representative.

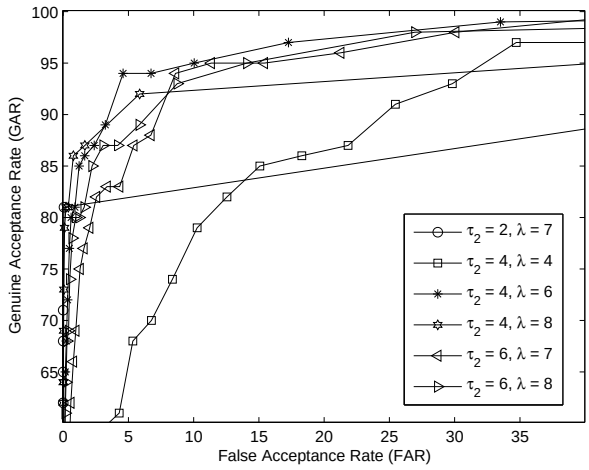


Figure 5.11: ROC curve of various parameters.

Nevertheless, there is a trade-off between this excellent accuracy performance and the computation speed. On average, the system takes about a minute to authenticate a transformed template-query fingerprint pair. In the real application, actually, this depends on what environment is being used and how it is implemented. For example, a hardware-based implementation should run faster than a software-based implementation [73].

5.3.2 Revocability and Diversity

In case the transformed fingerprint template is compromised, it is revoked by issuing a new set of keys and generating a new transformed fingerprint template based on the new keys. In order to evaluate this capability, the testing was conducted in FVC2002Db2a by taking the pseudo-imposter from the legitimate query set to measure the false acceptance level (p_1-FAR). The experimental results exhibit that all queries are rejected. In other words, (p_1-FAR) = 0. It means that the transformation function has been able to generate different transformed templates from exactly the same finger such that those templates do not authenticate each other. Similar to that in Chapter 4, this characteristic results in solving not

only the revocability but also cross-matching between databases issues [34, 49]. Hence, the same finger can be used to concurrently enroll on various applications.

Furthermore, the experiment to measure the r-FAR (the false acceptance rate when the fingerprint template-query pair is derived from different fingers and is transformed by using different keys, refer to Section 3.3) also results in no error (r-FAR = 0). It means that the discriminability property is still preserved in the transformed domain.

5.3.3 Changeability

The evaluation was conducted by matching the non-transformed query with the transformed template. In this case, both were derived from the same finger. The experimental result represents that all queries are rejected (p_2-FAR = 0). It can be deduced that, as in the previous testing scenario, the non-transformed query and the transformed template are sufficiently different; therefore, they do not authenticate each other.

In order to find the differences between genuine and imposter fingerprint pairs, the distribution of their matched minutiae points are plotted and presented in Figures 5.12 and 5.13; these represent both genuine and imposter pairs of non-transformed and transformed fingerprints, respectively. It is shown that, in general, non-transformed genuine fingerprints have more matched minutiae points than transformed ones. In order to further analyze their distribution, the separability factor [57] is measured, which is defined as:

$$Separability = \frac{|\mu_G - \mu_i|}{\sqrt{(\sigma_G^2 + \sigma_i^2)/2}} \tag{5.13}$$

where (μ_G, σ_G^2) and (μ_i, σ_i^2) represent respectively the mean and variance pairs of both the genuine and imposter distributions. It is found that with EER $\approx$ 5%, the separability of matched minutiae points of non-transformed genuine and imposter distributions is $\approx$ 3.37; on the other hand, with EER $\approx$ 6%, the separability of matched minutiae points of transformed

genuine and imposter distributions is ≈ 2.73.

Further separability evaluation of the matched minutia point distribution was conducted based on the following scenarios, whose results are depicted in Figure 5.14:

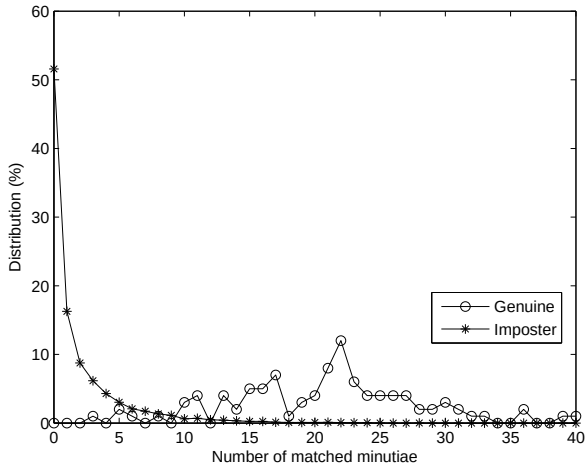


Figure 5.12: Distribution of matched insecure (non-transformed) genuine and imposter.

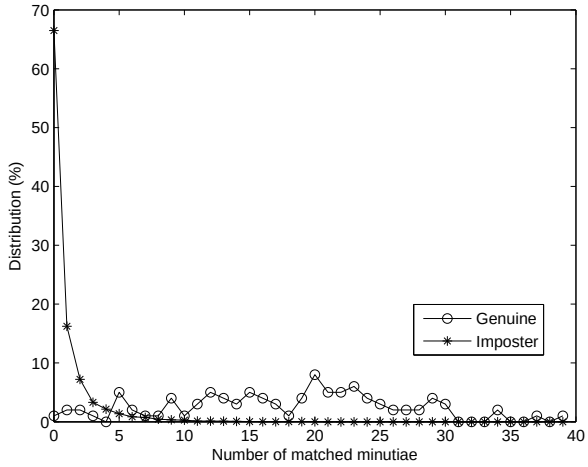


Figure 5.13: Distribution of matched secure (transformed) genuine and imposter.

1. Both genuine and imposter fingerprints are not transformed.

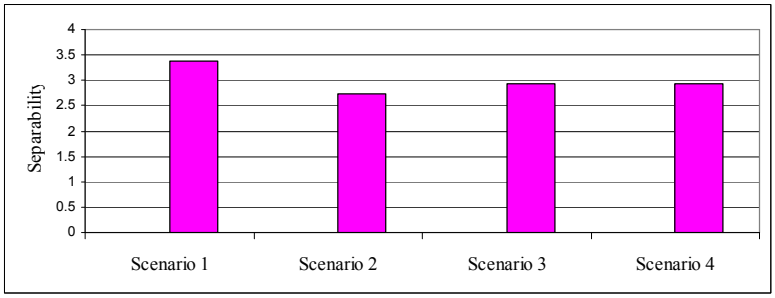


Figure 5.14: Separability of the specified scenarios.

2. Both genuine and imposter fingerprints are transformed by using the same key.

3. Genuine fingerprints are transformed by using the same key whilst imposter fingerprints are transformed by using different keys.

4. Genuine fingerprints are transformed by using the same key whilst imposter fingerprints are not transformed.

Based on Figure 5.14, it can be inferred that the separability of the scenarios 3 and 4 is approximately same. This means that the case of transforming both template and query fingerprints using different keys is analogous to the case where the template is transformed while the query is not transformed. On the other hand, non-transformed template and non-transformed query fingerprints (scenario 1) have the highest separability level and transforming both template and query fingerprints using the same key (scenario 2) leads to the lowest separability level. This means that, by the nature, the raw (original) fingerprints have a relatively high distinctiveness level; and transforming different fingerprints using the same key results in reducing this distinctiveness more than that of using different keys.

For a comparison purpose, the experiment on measuring the separability level of non-transformed and transformed fingerprints was also conducted in other databases (i.e., Db1a and Db3a of FVC2002) whose results are given in Figure 5.15. From those three databases, it is found that non-transformed fingerprints have higher separability level than transformed

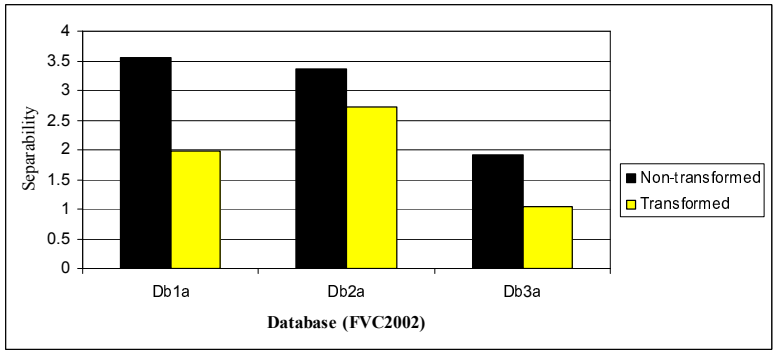


Figure 5.15: Separability of both non-transformed and transformed fingerprints from different databases.

ones for each corresponding database. The smallest separability decrease is obtained when the transformation is implemented in FVC2002Db2a. In addition, the highest separability of transformed fingerprints is also achieved by applying it to FVC2002Db2a. This supports the assumption which has been made in Section 3.2 that the proposed transformation function is more appropriate to implement in FVC2002Db2a because of its characteristics. In general, all these separability values are inversely proportional to their corresponding EER value.

5.3.4 Local Smoothness

As described in [77, 80], a transformation function should preserve local smoothness. This means that a transformation function should preserve the local minutiae structure but distorts the global structure. This is to make the pattern of the transformed template "scrambled" but still distinguishable. In this proposed approach, the local minutia structure is defined as the relation between the point at the center of the coordinate space (m_i) and its neighboring minutiae points $\{m_j\}_{i=1}^{p-1}$, instead of between neighboring points themselves. Therefore, the relative location of neighboring points before and after the transformation does not have to be similar. The most important thing is that after being transformed by using the same function and set of keys, the same fingerprint should have similar structure.

This concept is different from that of polar transformation in [77, 80] where a block is moved to another block. This leads to relatively stable local structures, that is, a transformed block has a similar pattern to its corresponding non-transformed version. It is worth pointing out that in each fingerprint scanning, minutiae points around the block boundaries may be moved to other blocks, resulting in the reordering issue. In this proposed design, this is minimized because of the use of multiple transformations in the template generation step (each minutia point is to be the transformation reference, refer back to Section 5.2.2).

5.3.5 Non-invertibility

Suppose s, r_w, μ, p, n are the total number of sectors, the transformed-radial factor, the modulo number for randomizing radial distance, the total number of selected minutiae points (in Section 5.2.1) and the total number of minutiae points in a fingerprint, respectively. It is known that in the transformed domain, each minutia point in BS is described by its $(p-1)$ neighboring minutiae points. Each of these $(p-1)$ neighboring minutiae points itself has $(s\mu)$ possible locations in this transformed domain. Therefore, for $(p-1)$ minutiae points, there are $(s\mu)^{p-1}$ possibilities. In other words, the probability of reconstructing the transformed template is $\frac{1}{(s\mu)^{p-1}} \times 100\%$. This means that given the transformed template, in order to reveal those $(p-1)$ minutiae points, a brute force attack should perform about $(s\mu)^{p-1}$ attempts $(((p-1) \times log_2(s\mu))$ bits). It is worth mentioning that in order to minimize the linkage between transformed templates, they are transformed by using different keys. Thus, they are independent of each other.

In addition, in order to reconstruct a minutia radial distance in BS, r_{i_j} has to be obtained from $r'_{i\_j} = \frac{(r_{i\_j} \times r_w) \bmod(\mu)}{r_w}$ (refer to Equation 5.9). This is difficult because both $(r_{i\_j} \times r_w)$ and the modulo operation $mod(\mu)$ give rise to many possible combinations. On the other hand, the authentication process is straightforward because the parameters are given for the sector mapping, scaling and other operations.

Therefore, in the event that the transformed template is compromised, the mapping operation being implemented in both sector and radial transformation functions brings about many combinations of the original minutia property. Moreover, the transformed template contains only p transformed minutiae points because of the minutia point selection process (refer to Section 5.2.1). This has been useful not only for the performance but also for the security (non-invertibility). In the worst case when BS can be revealed, the $(n - p)$ non-selected minutiae points are still safe because there is no stored information of those minutiae points at all.

5.4 Summary

In this chapter, a local feature based-cancelable template scheme has been proposed. Overall, the template is constructed by assigning a descriptor to each minutia point. This design consists of three modules: minutia point selection, template generation (including feature transformation and feature representation) and minutia point comparison (fingerprint matching). Those three are independent of each other; therefore, each module can be implemented with or without the others or even be combined with other schemes.

The first module has been able to reduce a significant number of minutiae points in a fingerprint. There are at least two advantages in this number reduction. First, it decreases the computational time because it does not have to evaluate all minutiae points in the fingerprint. Second, it prevents the adversary from revealing all minutiae points in the fingerprint, in case the transformed template is compromised. The second module performs multiple transformations in a polar coordinate space such that each minutia point is to be the reference to the others. An advantage of this design is that it can minimize the effect of intra-user variability caused by minutia reordering. Additionally, the transformation employs a modulo operation that is useful for giving uncertainty in revealing the fingerprint data, given the transformed template. The third module carries out fingerprint matching by comparing

descriptors of each minutia point. This module performs two matching levels: vector and point. So, the non-invertibility property is mainly maintained by the second module and supported by the first. Skipping the first module can decrease this non-invertibility level; however, reconstructing the original fingerprint data given a transformed template and its keys is still infeasible because of both sector and radial transformations.

In terms of performance, this local feature-based cancelable template scheme is similar to the global feature-based one, which has been proposed in Chapter 4. However, in terms of reliability, this pair-polar coordinate-based transformation is superior because it is not affected by the absence of singular point information (e.g., core point) of the fingerprint. This means that the scheme has eliminated the singular point dependency problem. The experimental results also show that it has better performance than that of most existing schemes and at the same time, it also meets the revocability, diversity, changeability and security (non-invertibility) requirements.

In spite of its excellence in performance and security properties, its authentication speed can be a drawback. Although this trade-off may be acceptable in some cases, this limitation can affect the users acceptability. Therefore, a local feature-based cancelable template scheme which does not suffer from this processing time drawback needs to be developed. The next chapter intends to minimize the drawback by exploring the geometrical fingerprint transformation in both Cartesian and polar coordinate spaces.

Chapter 6

Cartesian and Polar Coordinate-based Transformation

The research in this chapter is motivated by the fact that there is a trade-off among capabilities of local feature-based cancelable fingerprint template schemes. For example, the polar coordinate-based transformation proposed in Chapter 5 is able to remove the need of singular point detection, which makes it more reliable; however, it experiences an increase in processing time. Even though accuracy and security factors may be preferred in some cases, the processing speed is still an important factor. This has a significant effect on certain implementation environments, such as resource-constrained devices[1].

In this chapter, another local feature-based transformation is proposed. This transformation is designed such that it eliminates the use of multiple sets of keys while at the same time still considers the accuracy, revocability, diversity, changeability and security (non-invertibility) properties. In more specific, it utilizes both Cartesian and polar coordinate spaces to construct a transformation function. As in the previous chapter, this design is also motivated by other research, particularly the one in [80].

[1] A resource-constrained device is any device whose resources are constrained intentionally [55].

This chapter is organized as follows. Section 6.1 describes the cartesian-polar transformation design. The experiments and their results are depicted in Section 6.2 and the summary of this chapter is provided in Section 6.3.

6.1 Cartesian-polar Transformation Design

The overall transformation design, as depicted in Figure 6.1, takes a set of selected minutiae points BS (refer to Section 5.2.1) as the input to the transformation. This is similar to the pair-polar transformation design (Chapter 5). Nevertheless, taking the set of all minutiae points B as the input is also acceptable; however, as previously discussed, a smaller number of input points reduces both template generation and matching complexities.

Each minutia point in BS is given a descriptor, which is constructed by its transformed neighboring minutiae points. In turn, the descriptors of each minutia point is to be the secure fingerprint template. In general, the process of generating these descriptors can be denoted as follows:

$$BS_{sec} = \Gamma_2(\Gamma_1(BS, \kappa_{rot}), \kappa_{rad}, \kappa_\alpha, \kappa_\beta) \tag{6.1}$$

where BS_{sec}, Γ_1 and Γ_2 are the secure (protected/transformed) template, the transformation function in Cartesian and in polar coordinate systems, respectively. The transformation requires the set of keys κ which consists of a key κ_{rot} for the Cartesian transformation and $\{\kappa_{rad}, \kappa_\alpha, \kappa_\beta\}$ for the polar transformation. All these keys can be easily generated from a hashed pass phrase; therefore, it is very likely to be random. The use of a pass phrase itself has made it easy for users to memorize. The transformation design can be denoted in Algorithm 6.1 whose detail is described in the following sections.

In the verification process, after the fingerprint query and the fingerprint template are transformed, their similarity level is measured by using the matching algorithm proposed in

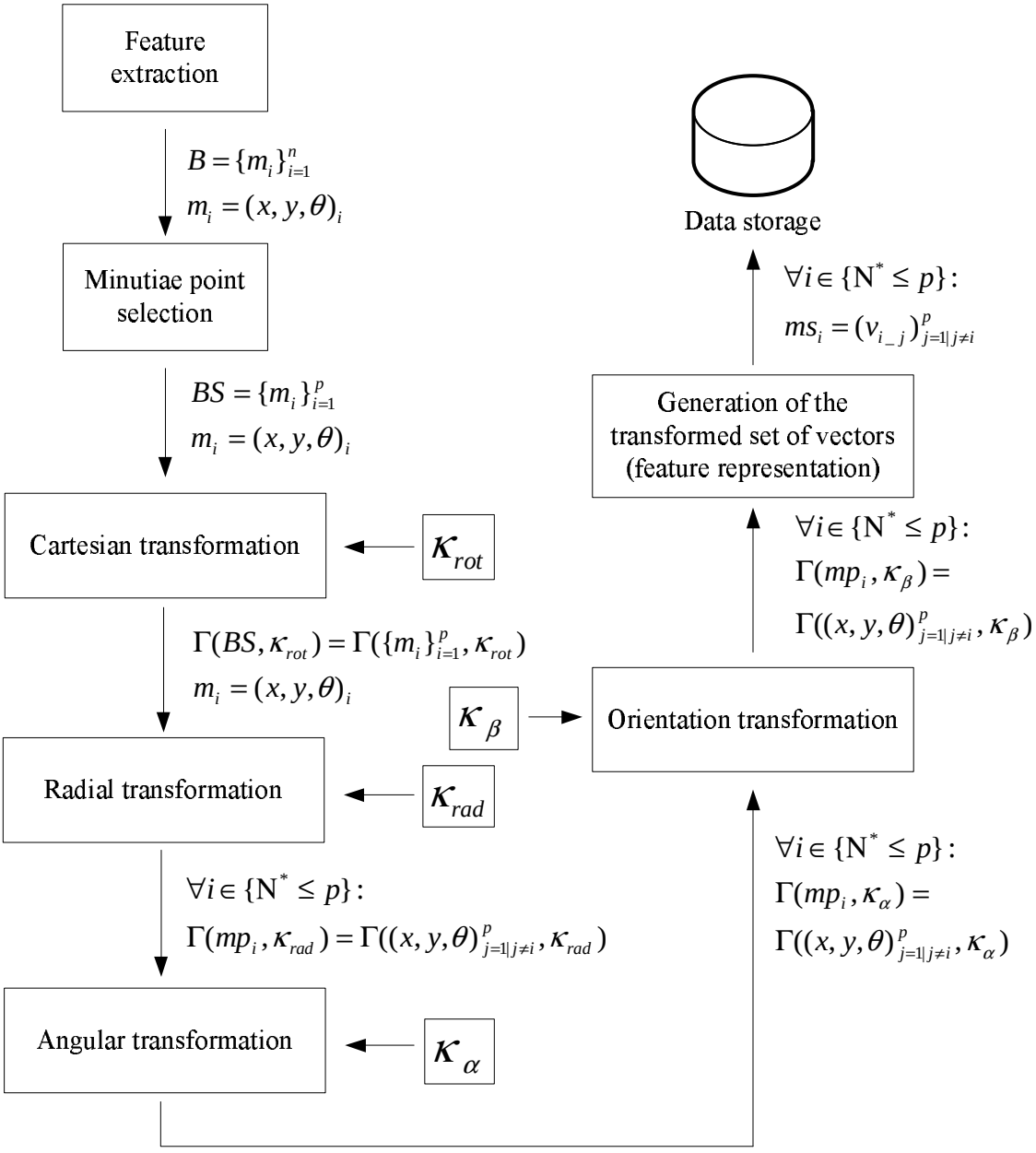


Figure 6.1: The Cartesian-polar transformation architecture.

Section 5.2.3. Only if this similarity level is high enough, are the template and the query considered to be from the same finger.

6.1.1 Cartesian-based Transformation

Let m_i be a minutiae point on the consideration (being processed). By using $m_i(0,0)$ as the center, the Cartesian coordinate space is divided into four quadrant-squares, which cover the minutiae points in BS. The size of the squares is parameterized, as a result, the number of minutiae points in the square may be different from application to application. The first resulting quadrants are to be the squares level 0. These squares can be re-divided several times, depending on the specified parameter, such that the original coordinate space is divided $(l+1)$ times (until level l). The total number of generated squares can be denoted by $\sum_{i=0}^{l} 4^{i+1}$. An example of squares in this Cartesian coordinate space with $l = 2$ is depicted in Figure 6.2(a).

The transformation is done by independently rotating all squares in each level centering on the midpoint of the corresponding square r times (rounds). It means that the total number of rotation is $(r \sum_{i=0}^{l} 4^{i+1})$, and each minutia point is rotated $((l+1) \times r)$ times. Let $(x, y), (x_p, y_p)$ and ϕ be the minutia point coordinate of pre- and post-point rotation, and the degree of rotation, respectively. Each quadrant at all levels is given a weight q_a, q_b, q_c, q_d for respectively quadrant 0, quadrant 1, quadrant 2 and quadrant 3. This transformation can be represented as:

$$\left.\begin{aligned} x_p &= x \times cos(\phi) - y \times sin(\phi) \\ y_p &= x \times sin(\phi) + y \times cos(\phi) \end{aligned}\right\} \tag{6.2}$$

where ϕ is generated from multiplication between the key κ_{rot} and the quadrant weight. In this case, κ_{rot} may be different from square to square. Furthermore, the value of ϕ can be restricted to a certain range of values by using a modulo operation; therefore, there is a many-to-one mapping. For example, if the rotation is restricted to multiples of 90^o, the

Algorithm 6.1 Transform minutiae points using Cartesian-polar method

Input: BS
Output: BS_{sec}

for $i \leftarrow 1$ to $total\_minutiae\_in\_BS$ **do**
  {Cartesian transformation}
  for $r \leftarrow 0$ to $total\_rounds$ - 1 **do**
    for $l \leftarrow 0$ to $total\_levels$ - 1 **do**
      for $j \leftarrow 1$ to $total\_minutiae\_in\_each\_level$ **do**
        $rotation\_transformation$
      end for
    end for
  end for

  {Polar-radial transformation}
  for $r \leftarrow 0$ to $total\_rounds$ - 1 **do**
    for $q \leftarrow 0$ to $total\_tracks$ - 1 **do**
      for $j \leftarrow 1$ to $total\_minutiae\_in\_(track_q, track_{q+1})$ **do**
        $radial\_transformation$
      end for
    end for
  end for

  {Polar-angular transformation}
  for $r \leftarrow 0$ to $total\_rounds$ - 1 **do**
    for $q \leftarrow 0$ to $total\_sectors$ - 1 **do**
      for $j \leftarrow 1$ to $total\_minutiae\_in\_(sector_q, sector_{q+1})$ **do**
        $angular\_transformation$
      end for
    end for
  end for

  {Polar-orientation transformation}
  for $r \leftarrow 0$ to $total\_rounds$ - 1 **do**
    for $q \leftarrow 0$ to $total\_sectors$ - 1 **do**
      $orientation\_transformation$
    end for
  end for

  {Generating transformed minutiae vector}
  for $j \leftarrow 1$ to $total\_minutiae\_in\_BS$ **do**
    if $j \neq i$ **then**
      $ms_i \leftarrow k\_shortest(v_{i\_j})$
    end if
  end for
  $BS_{sec} \leftarrow ms_i$
end for

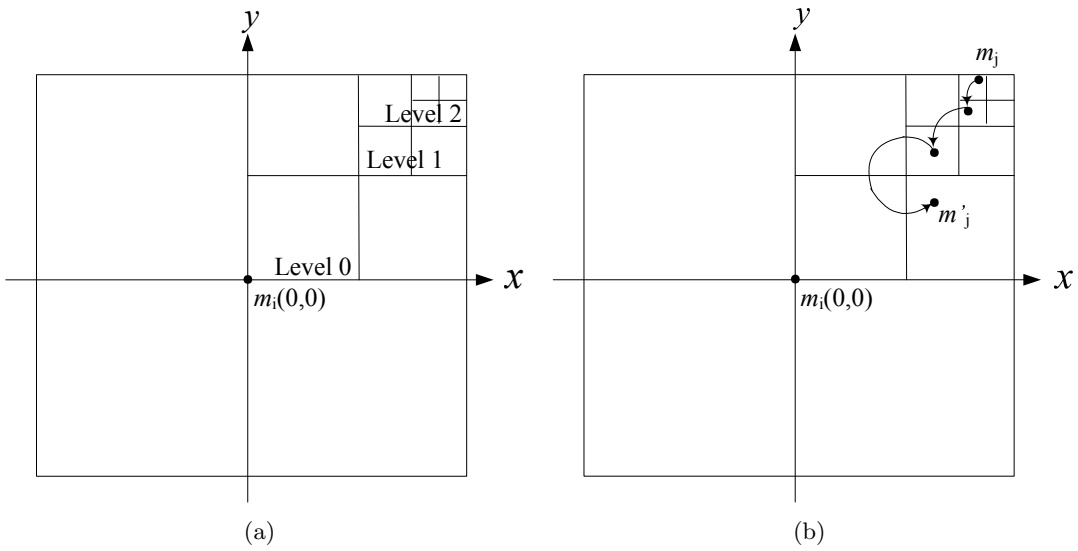


Figure 6.2: Square levels in the Cartesian coordinate space (a) definition of square level (b) an example of rotation when l=2, r=1.

following formula can be implemented to define the rotation of quadrant 3 at each square level (including the minutiae points in it): $\phi = ((q_d \times \kappa_{rot}) \bmod(4)) \times 90$.

An example of the rotation is shown in Figure 6.2(b), which for a simplicity purpose, the rotation properties are defined as $l = 2$ and $r = 1$. This rotates a neighboring point m_j three times. Since the transformation function rotates a set of minutiae points all together, its local structure does not change. It means that in the new location, a point still maintains its close neighboring structure. In this case, the term *close* refers to the smallest square size (the square level l). As occurred in other cell- and block-based transformations [77, 80], it is possible for the minutiae points near the square boundary to move to the other square in the next fingerprint scanning process. It is worth pointing out that this rotation is inspired by research in [80]; however, this kind of transformation is not practically implemented there.

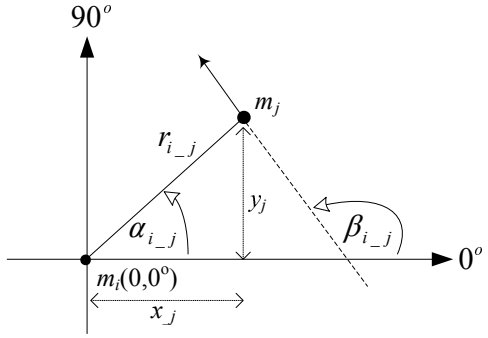


Figure 6.3: Definition of vector $v_{i\_j} = (r_{i\_j}, \alpha_{i\_j}, \beta_{i\_j})$.

6.1.2 Polar-based Transformation

In the polar-based transformation, the relation between a minutia point on the consideration (at the center of the space) m_i and its neighboring point m_j is represented as a vector v_{i_j}. The set of vectors constructed by minutiae points centering at m_i is denoted by $ms_i = \{v_{i\_j}\}_{j=1|j\neq i}^{k}, 1 \leq i \leq p$, where k and p are the specified number of the nearest neighboring minutiae points and the number of minutiae points in BS, respectively. In this case, k does not have to be same as p (this is different from that of Equation 5.6, where k is same as p). The elements of vector v_{i_j}, as depicted in Figure 6.3, are defined as follows:

- r_{i_j}: distance between the center (m_i) and its neighboring point (m_j).
- α_{i_j}: angle between 0^o axis (horizontal axis) and the edge (the line connecting the center (m_i) and the neighboring point (m_j)) in counterclockwise.
- β_{i_j}: angle between 0^o axis (horizontal axis) and the orientation of its neighboring point in counterclockwise.

Here, r_{i_j} and α_{i_j} have the same definition as that of the transformation scheme proposed in Chapter 5 while β_{i_j} does not. In the implementation level, this new β_{i_j} definition is simpler than the previous one which has an effect on the overall performance, specifically on the authentication speed.

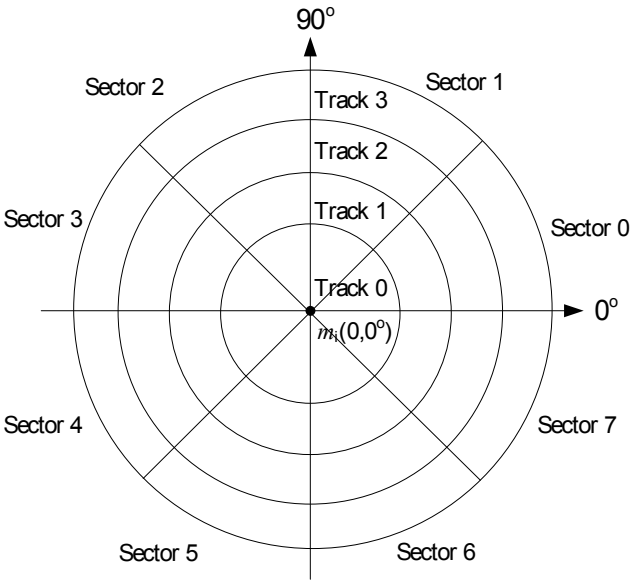


Figure 6.4: A polar space whose center is m_i, is divided into 8 sectors and 4 tracks.

The polar coordinate space is divided into t geometrical tracks (from track 0 to track $t-1$) and s sectors (from sector 0 to sector $s-1$). In this case, each minutia point is transformed individually based on its position in the sector or in the track; therefore, the structure between neighboring points (the local structure) may change. This is different from the previous transformation function proposed in Chapter 5, which concurrently transforms minutiae points per track-sector (block). Figure 6.4 illustrates the polar coordinate space when $s = 8$ and $t = 4$.

The transformation itself is performed in three steps. First, the minutiae points are transformed based on their radial distance, which employs tracks as the reference. Second, the minutiae points are transformed based on their angular distance, which uses sectors as the reference. Therefore, it can be viewed as a track- and sector-based transformation instead of a block-based transformation as implemented in [77, 80]. Third, the orientation of each minutia point is transformed in a similar way as that of the second step. Overall, the proposed function transforms the minutia point coordinate and orientation in several rounds.

Let the polar space be divided into t tracks and s sectors whose sizes are ω_t and ω_s, respectively. The process of this polar transformation is denoted in Algorithm 6.1 whose detail and example are provided in the following sections.

Radial Distance Transformation

In this radial distance transformation, the transformation function in Equation 6.3 is applied to the minutiae points in two consecutive tracks. In other words, minutiae points in track q are transformed along with those in track $(q+1)$. If track q is the last track (track $(t-1)$), then the next one is track 0 (defined in Equation 6.4).

$$\begin{aligned} &\forall q \in \{\mathbb{N}^0 < t\} \text{ and } \forall j \in \{\mathbb{N}^* \leq p, j \neq i\} : \\ &r'_{i\_j} = ((r_{i\_j} \times \kappa_{rad}) \bmod(2 \times \omega_t) + (q \times \omega_t)) \bmod((t-1) \times \omega_t) \end{aligned} \tag{6.3}$$

where (r_{i_j}) and (r'_{i_j}) are the radial distance of pre- and post-radial distance transformation between the center (m_i) and its neighboring point (m_j).

$$m_j \in \begin{cases} \{track_q, track_0\} \text{ if } q = t-1 \\ \{track_q, track_{q+1}\} \text{ if } q \neq t-1 \end{cases} \tag{6.4}$$

Suppose the polar coordinate space is divided into 4 tracks. The transformation starts from the first two tracks: track 0 and track 1, and iteratively moves to the subsequent tracks. The transformed minutiae points originating from track 0 and track 1 are very likely to hold new coordinate points, spreading over those two tracks. This means that some or all minutiae points in track 0 will either move to track 1 or remain in track 0. The same applies to minutiae points in track 1. These transformed minutiae points in track 1 and those of non-transformed in track 2 are processed whose results spread over both tracks, and so on. Finally, the minutiae points in the last track (track 3) are transformed along with those in the first track (track 0) and the results spread over track 3 and track 0. This will construct

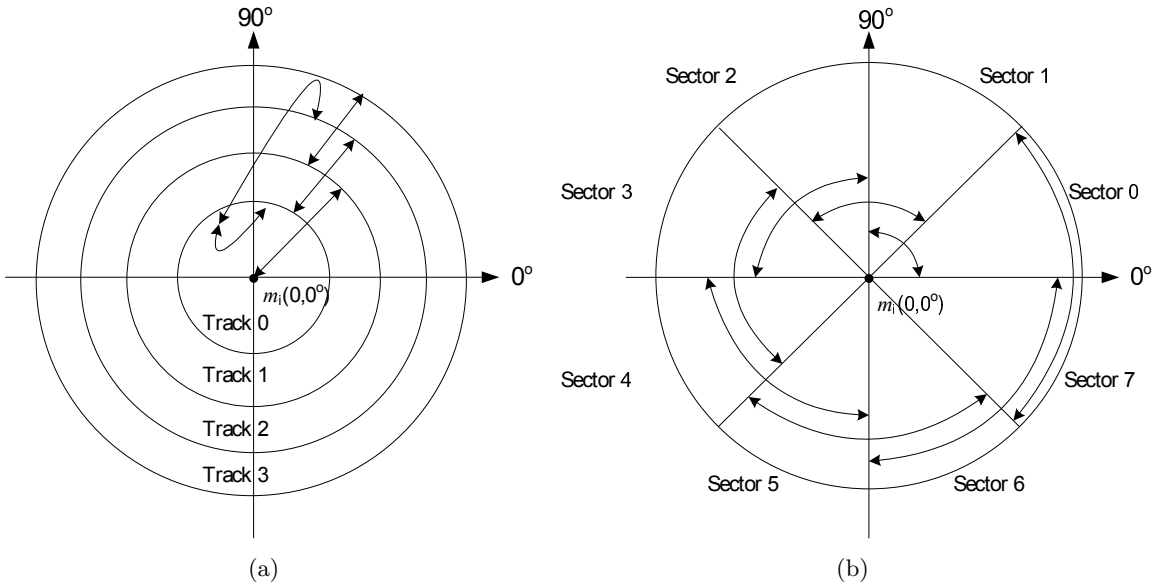


Figure 6.5: An example of polar transformation, when t=4, s=8 (a) a round of radial transformation is performed from track 0 and going back to track 0 (b) a round of angular transformation is performed from sector 0 and going back to sector 0.

one round radial transformation, as depicted in Figure 6.5(a).

The transformation is performed for at least 2 rounds to make it possible for each point to move to or stop at any track. In other words, a minutiae point in track 0 may move to tracks 1, 2, 3 and vice versa. It is also possible that a minutia point will finally arrive back at its original track but most likely with different r_{i_j} value. Some minutiae points originating from different tracks may result in the same track. It is difficult to determine the original radial distance or even the track where the minutiae points originated from, because each point is transformed by using a many-to-one mapping due to the implementation of the modulo operation.

A more detail illustration of this radial distance transformation is shown in Figure 6.6, given m_i with $i \in \{\mathbb{N}^* \leq 6\}$ and $t = 4$. For a clarity purpose, it is assumed that the original minutiae points spread only over track 0 and track 1. After minutiae points in track 0 and track 1 are transformed, m_1 is still in track 0, m_2 moves to track 0, m_3 and m_5 are still in

track 1, m_4 and m_6 moves to track 1. Transforming minutiae points in tracks 1 and 2 leads to locate m_3, m_4 and m_6 in track 2 while m_5 remains in track 1. The next transformation of tracks 2 and 3 results in moving m_3 and m_6 to track 3 while m_4 is still in track 2. After a round transformation, m_3 is in track 3 and m_6 is back to track 0.

Angular Transformation

The angular transformation, as shown in Figure 6.5(b), is analogous to the radial distance transformation, which has been previously discussed. Different from it, the minutiae points are transformed in angular instead of radial direction. A many-to-one mapping is also applied to this transformation. In this case, the minutiae angle α is transformed according to

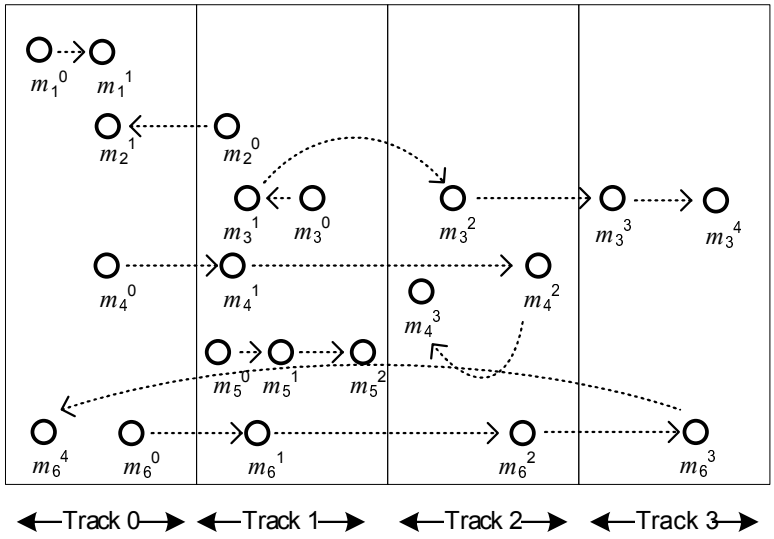


Figure 6.6: An example of a round radial transformation where minutiae points originated from track 0 and track 1 end up in various tracks. The superscript and subscript numbers represent the transformation step being applied to the minutiae and the minutia identity, respectively.

Equations 6.5 and 6.6.

$$\begin{aligned} &\forall q \in \{\mathbb{N}^0 < s\} \text{ and } \forall j \in \{\mathbb{N}^* \leq p, j \neq i\} : \\ &\alpha'_{i\_j} = ((\alpha_{i\_j} \times \kappa_\alpha) \bmod(2 \times \omega_s) + (q \times \omega_s)) \bmod((s-1) \times \omega_s) \end{aligned} \tag{6.5}$$

$$m_j \in \begin{cases} \{sector_q, sector_0\} \text{ if } q = s-1 \\ \{sector_q, sector_{q+1}\} \text{ if } q \neq s-1 \end{cases} \tag{6.6}$$

where (α_{i_j}) and (α'_{i_j}) are the angle of pre- and post-transformation of a neighboring point (m_j) and $sector_q$ is the q^{th} sector. A round angular transformation is defined as the transformation from the first sector (sector 0) to the last sector (sector $s-1$) and back to the first. Similar to the radial distance transformation, this angular transformation is also performed for at least two rounds, such that, every minutiae point has an opportunity to move within 360^o. After the transformation, a sector may contain minutiae points coming from various sectors.

Orientation Transformation

The minutia point orientation is transformed in the same way as the angular transformation (see Equation 6.5) by using the key κ_β instead of κ_α. Both transformations are conducted based on the angle definition which has been described in Section 6.1.2 and illustrated in Figure 6.3. In this transformation, however, each minutia orientation is transformed separately because each minutia point has only one orientation angle in its polar space, namely its orientation itself, as shown in Figure 6.7. In this example, β_{1_2} and β_{1_3} are independently transformed based on its corresponding center of space: m_2 and m_3, respectively.

As previously discussed, after the Cartesian and polar transformations are complete, each minutia m_i is described by a set of vectors $ms_i = \{v_{i\_j}\}_{j=1|j\neq i}^{p}, 1 \leq i \leq p$ where p is the total number of minutiae points in BS. Among $(p-1)$ of m_i's transformed neighboring minutiae

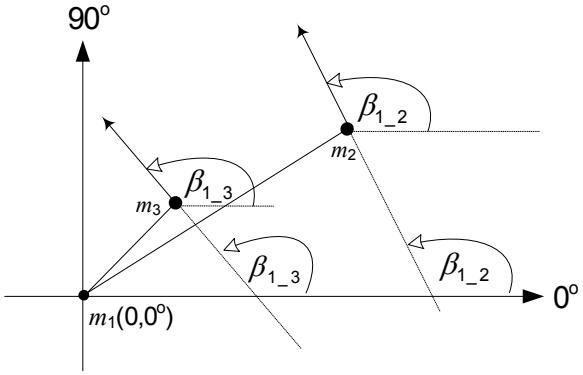


Figure 6.7: Orientation transformation.

points, only the k-nearest of them are selected to be the m_i's descriptor. In other words, the set of vectors ms_i is redefined, such that, it consists of only k out of $(p-1)$ possible vectors as denoted in Equation 6.7. These k-nearest neighboring transformed points are selected according to their radial distance from the center, m_i (denoted by r_{i_j}).

$$\forall i \in \{\mathbb{N}^* \leq p\} : ms_i = \{v_{i\_j}\}_{j=1|j\neq i}^{k} \tag{6.7}$$

Let BS_{sec} be the secure fingerprint template consisting of all sets of vectors resulted from the transformation ($\{ms_i\}_{i=1}^{p}$). Since $\{ms_i\}_{i=1}^{p}$ is to be the transformed version of BS where BS itself is the representation of B, BS_{sec} is the transformed fingerprint of B (see Equation 6.8). This can be the template to be stored in the database or the query to be matched to the stored template in the verification process.

$$\left.\begin{aligned} BS_{sec} &= \{ms_i\}_{i=1}^{p} \\ &= \{\{v_{i\_j}\}_{j=1|j\neq i}^{k}\}_{i=1}^{p} \end{aligned}\right\} \tag{6.8}$$

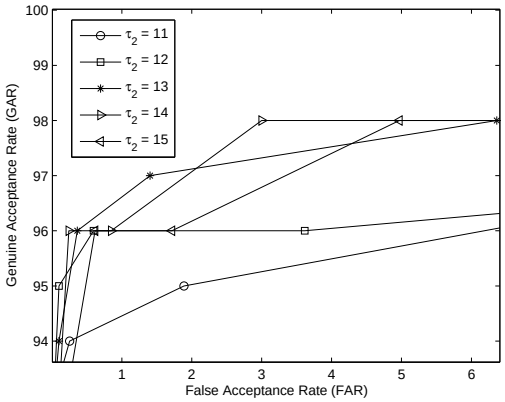


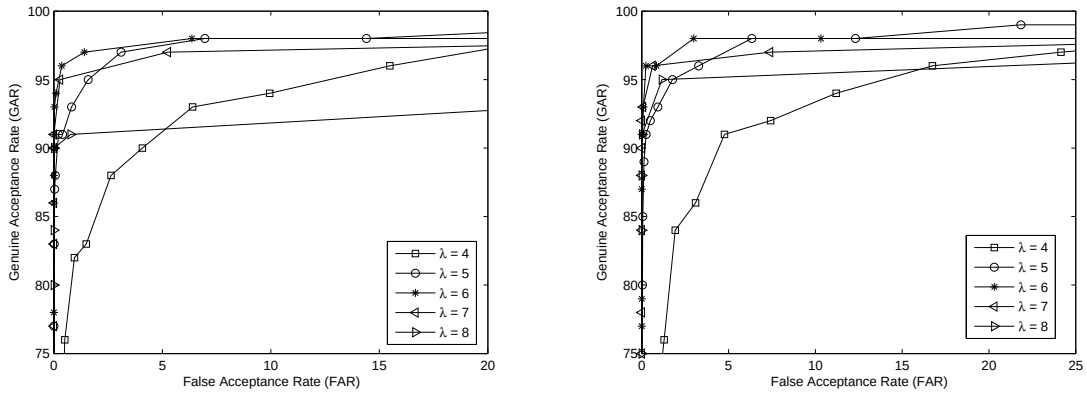
Figure 6.8: The ROC curve when $\lambda = 6, 11 \leq \tau_2 \leq 15$. Both templates and queries are transformed by using the same key.

6.2 Experiments and Analysis

As in the evaluation design, the proposed approach is evaluated based on its accuracy (performance), revocability, diversity and changeability. In addition, security (non-invertibility) is also evaluated.

6.2.1 Accuracy

In this scenario, the performance was firstly evaluated by varying τ_2, which is the upper bound value of the vector difference, Δf (the definition of τ_2 is in Equation 5.12, Section 5.2.3). It is found that $\tau_2 = 13$ and $\tau_2 = 14$ give the best performance, as shown in Figure 6.8. In general, implementing those two threshold values produce a higher GAR than that of the others, particularly when the FAR is less than 4%. A higher τ_2 (i.e., 15) exhibits a good performance, as that of $\tau_2 = 14$ if only the FAR is more than 5%, while a lower τ_2 (i.e., 11, 12) denotes a lower performance than the others. It can be inferred that this proposed design generates a more varied transformed fingerprint pattern than that of the previous pair-polar method proposed in Section 5, whose τ_2 is 4.



(a) (b)

Figure 6.9: ROC curves when both templates and queries are transformed by using the same key (a) ROC curve for $\tau_2 = 13, 4 \leq \lambda \leq 8$ (b) ROC curve for $\tau_2 = 14, 4 \leq \lambda \leq 8$.

Further performance evaluation is carried out by varying the minimum number of matched vectors, λ (defined in Section 5.2.3), and fixing τ_2 to 13 and 14. The ROC curve of $\tau_2 = 13$, which is depicted in Figure 6.9(a), shows that $\lambda = 6$ has the highest GAR level when FAR is between 0 and about 14%. In particular, it achieves about 98% of GAR when its FAR is between around 6% and 14%, same as that of $\lambda = 5$. A slightly lower GAR is obtained when its FAR is less than 6%. The ROC curve of $\tau_2 = 14$ is depicted in Figure 6.9(b). It also points out that $\lambda = 6$ has the best performance, especially when the FAR is less than 6%, whose GAR is 98%. Starting at FAR = 6%, $\lambda = 5$ reaches this GAR level and even higher when its FAR is more than 11%. However, FAR greater than 10% may not be acceptable even though the GAR is close to 100%. Therefore, $\lambda = 6$ is preferred over $\lambda = 5$. This λ level is same as that of the previous polar method in Section 5. Denote η the minimum number of pair-matched minutiae points between the template and the query. From these curves, the GAR and FAR of both (τ_2, λ) pairs can be summarized in Table 6.1. In the remaining sections of this chapter, $(\tau_2 = 13, \lambda = 6)$ and $(\tau_2 = 14, \lambda = 6)$ are referred by $transform_1$ and $transform_2$, respectively.

In order to evaluate the effect of the transformation on the overall performance degrada-

tion, the EER curve is generated. As depicted in Figures 6.10 (a) and 6.10 (b), which represent the EER curves of transform_1 and transform_2, respectively, the performance degradation of before and after the transformation are about 2.65% and 2.25%, respectively. These are lower than that of most other research, such as [8]. In comparing this performance degradation level with that of the previous approach proposed in Chapter 5, it is shown that this approach is higher; however, all EER values obtained by this approach are actually lower. Therefore, in terms of EER, this proposed approach is better than the previous. It is also shown that despite having different degradation levels, transform_1 and transform_2 generate similar EER levels for both before and after the transformation.

Table 6.1: Summary of GAR and FAR of both (τ_2, λ) pairs when template-query pairs are transformed by using the same key.

Transformation	η	GAR (%)	FAR (%)
transform_1	2	98	6.36
	3	97	1.40
	4	96	0.36
transform_2	3	98	3.00
	4	96	0.85
	5	96	0.24

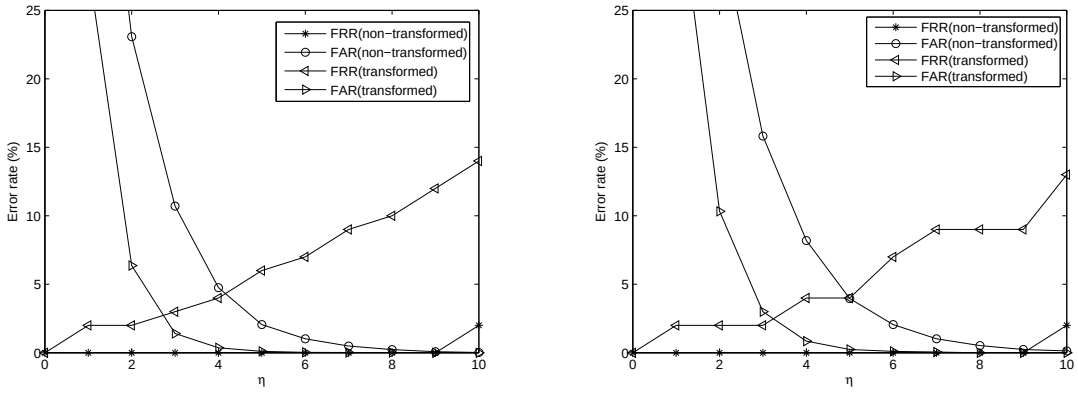


Figure 6.10: The EER curves of both non-transformed (unprotected) and transformed (protected) templates (a) the EER curve of transform_1; there is an EER difference of about 2.65% (b) the EER curve of transform_2; there is an EER difference of about 2.25%.

Transform_1 and transform_2 generate the same EER values when the experiment is carried out in FVC2002Db1a and FVC2002Db3a, that the EER values generated from FVC2002Db1a for non-transformed and transformed fingerprint templates are 1.2% and 4.2%, respectively; and those of FVC2002Db3a are 11% and 13%, respectively. It is worth pointing out that among fingerprint pairs in FVC2002Db3a, there are 2% of them whose minutiae points cannot be extracted at all, which lead to failure to enrol (FTE) (refer to Section 4.3.1). Overall, based on the experimental results obtained from those sub-databases, it can be inferred that the smallest performance degradation, which is represented by an EER increase, is obtained from FVC2002Db3a. However, its EER for both before and after the transformation is actually the highest. The smallest EER of transformed data is obtained from FVC2002Db2a. This means that this transformation is more appropriate to use in this sub-database, as in the research assumption (refer to 3.2.2). The EER comparison between this proposed approach and surveyed fingerprint data protection ones are provided in Table 6.2. This shows that its EER is lower than that of the others. Note that, in the implementation, the EER value is not the only consideration. The preferred parameter setting depends on the application characteristics, whether the concern is security (low FAR/high GRR) or convenience (low FRR/high GAR).

The summary of the EER and GAR of the proposed approach along with those of the previous chapters is given in Table 6.3. It is also shown that the proposed approach has a better performance than the others.

In order to further evaluate the transformation function performance, the experiment is conducted by generating 10 random sets of keys for each template-query pair such that there are 1000 genuine and 99000 imposter testings whose results are shown in Table 6.4. This is also used to measure the effect of the key variation on the performance. By comparing those experimental results with that provided in Table 6.1, it can be inferred that in terms of GAR, the variation of keys does not affect the performance significantly. Overall, the best

Table 6.2: EER comparison of the proposed methods with some existing fingerprint template protection ones.

Reference	EER (%) per database			
	FVC2002 Db1a	FVC2002 Db2a	FVC2002 Db3a	other public/ private Db
Yang et al. [113]	–	13	–	–
Sutcu et al. cited in [113]	–	35	–	–
Arakala et al. [11]	–	–	–	15
Ang et al. [8]	–	–	–	16.8
Lee and Kim [56]	–	–	–	6.8; 9.5; 10.3
Jin et al. [49]	–	–	–	>10
Proposed methods:				
- transform$_1$	4.2	2.7	13	–
- transform$_2$	4.2	2.5	13	–

Table 6.3: The summary of EER and GAR of the proposed methods when FAR = 1% and FAR = 5%; the experiment is conducted in FVC2002Db2a.

Proposed methods	EER (%)	GAR (%)	
		FAR = 1%	FAR = 5%
Chapter 4	5.5	76.9	94.0
Chapter 5	6.0	85.0	94.0
This chapter			
- transform$_1$	2.7	96.5	97.5
- transform$_2$	2.5	96.0	98.0

performance of transform$_1$ and transform$_2$ are provided by ($\eta = 4$) and ($\eta = 5$), respectively. The (GAR, FAR) pairs generated by those two η values are respectively (95.1%, 2.31%) and (94.70, 2.54%).

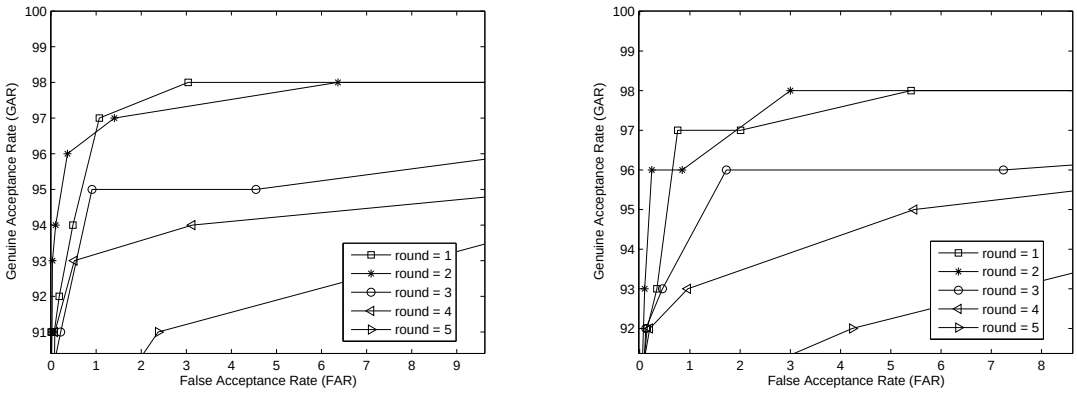
The use of several *rounds* (refer to Section 6.1.2) on the polar transformation is predicted to affect the performance. This is because it enlarges the transformed minutia coordinate and orientation spaces from a track and a sector to all tracks and all sectors. In other words, the resulted minutia coordinate is more varied. This predicted performance impact was evaluated by varying the number of rounds used in the transformation. The experimental results are presented in Figure 6.11. It can be inferred that, in general, a smaller number of rounds

results in a higher performance. In certain FAR ranges, however, $round = 1$ and $round = 2$ have the same GAR levels. Furthermore, $round = 2$ is able to generate a higher GAR when the FAR is less than 1%,(shown in Figure 6.11(a)) or between 2% and 5% (shown in Figure 6.11(b)). From this, $round = 2$ may be preferred, considering that it gives an opportunity for each minutia point to move to other tracks or sectors that $round = 1$ does not.

Although the use of rounds has made it possible to vary the location of transformed minutiae points, the final transformation is also determined by the size of tracks (ω_t) and sectors (ω_s). It is expected that smaller tracks or sectors generate a better performance because the transformed minutiae points may not move far from their original location due

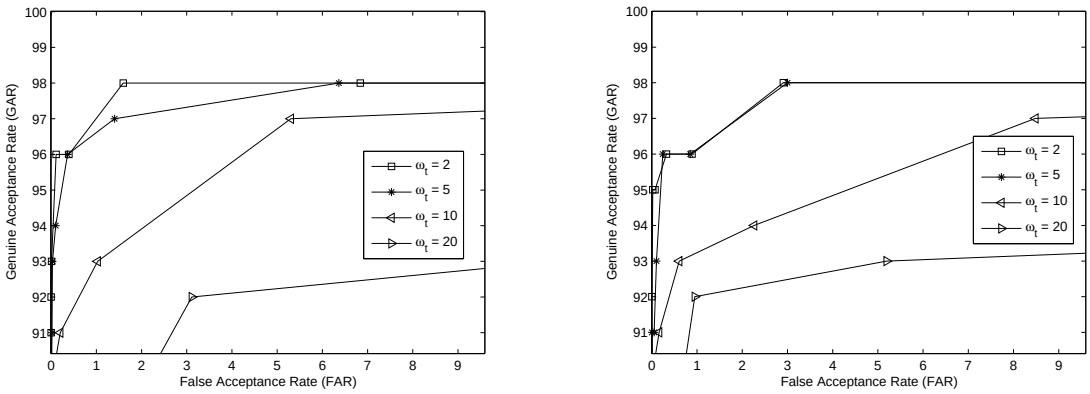
Table 6.4: The mean (μ) and standard deviation (σ) of GAR and FAR. The testing was carried out over 1000 genuine and 99000 imposter pairs.

Transformation	η	GAR (μ, σ) (%)	FAR (μ, σ) (%)
transform_1	2	(98.3, 1.42)	(17.06, 7.29)
	3	(96.6, 1.71)	(6.26, 3.65)
	4	(95.1, 1.66)	(2.31, 1.64)
transform_2	3	(97.40, 1.58)	(11.06, 9.99)
	4	(96.00, 1.94)	(5.14, 6.37)
	5	(94.70, 1.57)	(2.54, 3.92)



(a) (b)

Figure 6.11: The ROC curve for various round parameter values (a) the ROC curve for $transform_1$ (b) the ROC curve for $transform_2$.



(a) (b)

Figure 6.12: The ROC curve for various ω_t parameter values (a) ROC curve for transform$_1$. (b) ROC curve for transform$_2$.

to the track or sector space (refer to Equations 6.3 and 6.5). Figures 6.12 and 6.13 represent the performance generated by some different track and sector sizes for both transform$_1$ and transform$_2$. As expected, those figures indicate that a smaller track or sector size has a relatively higher GAR. It is worth noting that, however, there is a trade-off between performance and security. A relatively small ω_t and ω_s may also have a lower security level (discussed in Section 6.2.4).

In terms of the computation time, this proposed approach is also better than the previous approach in Chapter 5. On average, it needs about six seconds to perform an authentication process, which shortens the time for transforming and authenticating the template and the query fingerprints. Overall, the existing and proposed fingerprint protection data methods can be summarized in Table 6.5.

6.2.2 Revocability and Diversity

In case the transformed template BS_{sec} or the set of keys $\kappa = \{\kappa_{rot}, \kappa_{rad}, \kappa_{\alpha}, \kappa_{\beta}\}$ is compromised, the keys and the corresponding transformed template are revoked. A new set of keys is created to generate a new transformed template.

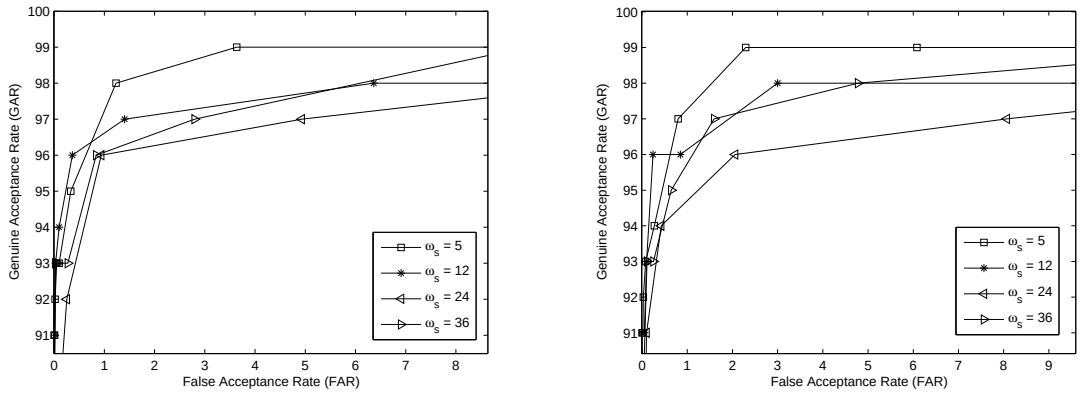


Figure 6.13: The ROC curve for various ω_s parameter values (a) the ROC curve for transform$_1$ (b) the ROC curve for transform$_2$.

To evaluate the capability of the transformation function for diversifying the transformed fingerprint, the query and the template derived from the same finger were transformed by using different keys. For this purpose, 99 different sets of keys were randomly generated for each template-query pair. This produces 9900 testings, which is equivalent to that used for measuring the false acceptance in the previous evaluation (see Figures 6.8, 6.9, 6.10). This is to be a pseudo imposter testing (p_1-FAR), that is, the transformed query which is generated from a legitimate finger is treated as if it is generated from an illegitimate one. This pseudo imposter is then matched to its corresponding transformed template.

The experimental results are presented in Table 6.6 along with the GAR and FAR generated from the previous scenario (Table 6.1). It is shown that there is a small FAR increase, especially at transform$_1$ with $\eta = 4$ and transform$_2$ with $\eta = 5$ whose FAR difference is less than 2%. As in the other evaluations, a greater η results in a smaller p_1-FAR as well as a smaller GAR and FAR. This also shows that the transformation function is relatively insensitive to the key variation, which is good. In addition, it has supported the assumption that generating transformed templates by using different keys is similar to giving the user new fingerprints. This capability means accomplishing diversity as well as non-cross matching

Table 6.5: Summary of fingerprint data protection methods (where 1: biometric cryptosystem, 2: feature transformation, a: global features, b: local features).

No	Ref.	Method	Feature	Note
1	Yang et al. [113]	2	a,b	each pair of minutiae points is connected and is mapped onto a perpendicular direction
2	Sutcu et al. cited in [113]	2	a,b	same as [113] but the mapping is in a straight direction
3	Arakala et al. [11]	1	a,b	secure sketches are generated by using Pinsketch [31]; it implements the set different metric and BCH for measuring and correcting the errors
4	Ang et al. [8]	2	a,b	a fingerprint space is divided into 2 subspaces where the first is reflected onto the second; the error rate is relatively high
5	Lee and Kim [56]	2	b	generating bit string by projecting minutiae points on 3D array
6	Jin et al. [49]	2	a,b	triangles are developed over a fingerprint space, which covers a certain number of minutiae points; this number is processed to be the finger identity
	Proposed methods:			
7	- Chapter 4	2	a,b	minutiae points are projected onto a line crossing the core point
8	- Chapter 5	2	b	minutiae points are transformed in a Cartesian space
9	- This chapter	2	b	minutiae points are transformed in Cartesian and polar spaces

among databases issues [49, 34].

The diversity between real imposters (r-FAR), i.e., a template-query pair which is originated from different fingers and is transformed by using different keys, is presented in Table 6.7. It is found that the (r-FAR) is close to zero which is equivalent to the situation when the adversary tries to break the system but he/she does not have knowledge about the key.

Table 6.6: The mean and standard deviation of pseudo false acceptance rate (p_1-FAR), where the template and query are derived from the same finger and transformed by using different keys. The corresponding GAR and FAR are also provided.

Transformation	η	GAR (%)	FAR (%)	p_1-FAR (μ, σ) (%)
transform_1	2	98	6.36	(13.55, 17,05)
	3	97	1.40	(5.35, 10.93)
	4	96	0.36	(2.30, 6.53)
transform_2	3	98	3.00	(8.01, 13.82)
	4	96	0.85	(3.68, 8.87)
	5	96	0.24	(1.56, 4.94)

Table 6.7: The r-FAR of both transform_1 and transform_2 when different fingers are transformed by using different keys.

Transformation	η	r-FAR (%)
transform_1	2	0.27
	3	0.00
	4	0.00
transform_2	3	0.01
	4	0.00
	5	0.00

6.2.3 Changeability

For this evaluation, each fingerprint template is transformed by 99 random sets of keys, and was matched to its corresponding non-transformed fingerprint query to measure its pseudo FAR (p_2-FAR). This leads to 9900 template-query pair testings whose results are shown in Table 6.8. It is depicted that the difference between p_2-FAR and its corresponding FAR is small (less than 1% for $\eta = 3, 4, 5$). Compared with the p_1-FAR evaluation results, this transformation generates lower false acceptance levels (less than 5% of p_2-FAR is obtained for all of those specified η).

Table 6.8 also depicts that the transformation has made the fingerprint features relatively different from their original version such that the secure (transformed) template does not authenticate the insecure (non-transformed) query as in the assumption made in Section 3.3.3. Likewise, this means that the transformation is relatively insensitive to the set of keys

Table 6.8: The mean (μ) and standard deviation (σ) of the pseudo false acceptance rate of transformed template and non-transformed query pairs (p_2-FAR). The template and the query are derived from the same finger. The corresponding GAR and FAR are also provided.

Transformation	η	GAR (%)	FAR (%)	p_2-FAR (μ, σ) (%)
transform_1	2	98	6.36	(4.83, 7.34)
	3	97	1.40	(1.09, 2.67)
	4	96	0.36	(0.17, 0.64)
transform_2	3	98	3.00	(2.08, 4.65)
	4	96	0.85	(0.43, 1.50)
	5	96	0.24	(0.10, 0.48)

and the transformed fingerprint itself can be seen as a new fingerprint.

The separability [57] of fingerprint minutia distribution is measured based on the following scenarios, whose results are provided in Figures 6.14 and 6.15:

1. Both genuine and imposter fingerprints are not transformed.

2. Both genuine and imposter fingerprints are transformed by using the same key.

3. Genuine fingerprints are transformed by using the same key whilst imposter fingerprints are transformed by using different keys.

4. Genuine fingerprints are transformed by using the same key whilst imposter fingerprints are not transformed.

Those two graphs show that for the same evaluation scenario, the fingerprint separability values in Figures 6.14 and 6.15 are very close, in spite of the different (τ_2, λ) parameter values. It is also depicted that non-transformed fingerprints have the highest separability and transforming fingerprints using the same key results in the lowest separability. It means that the transformation has decreased the uniqueness of each fingerprint. In fact, the error rate obtained from transformed fingerprints (represented by EER) is higher than that of non-transformed.

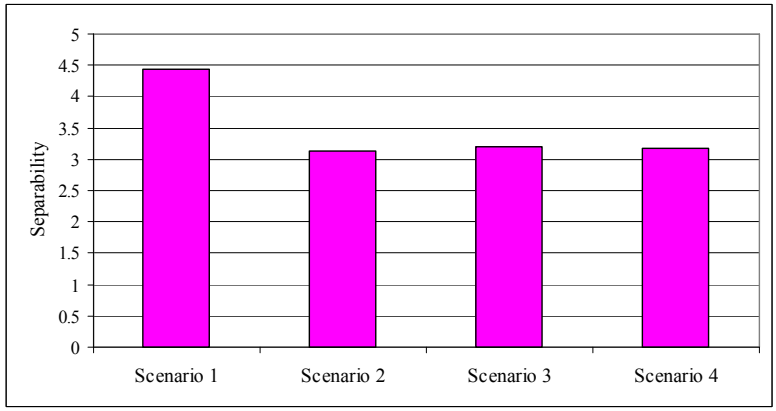


Figure 6.14: Separability of the specified scenarios for transform$_1$.

The separability of various databases are depicted in Figures 6.16 and 6.17. It is shown that a similar trend exists for the corresponding evaluation scenario and database. It is also depicted that Db2a and Db3a databases present the highest and the lowest separability for both non-transformed and transformed fingerprints, respectively. In addition, these separability values are inversely proportional to the EER values. This supports the assumption that users willingly provide their fingerprint data to be authenticated (refer to Section 3.2.2).

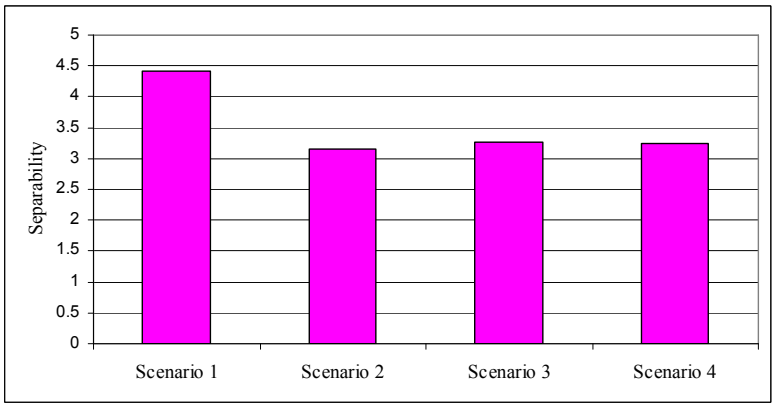


Figure 6.15: Separability of the specified scenarios for transform$_2$.

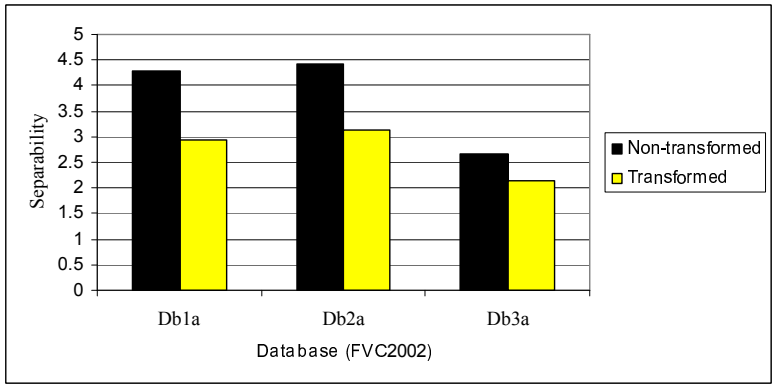


Figure 6.16: Separability of both non-transformed and transformed fingerprint from different databases for transform$_1$.

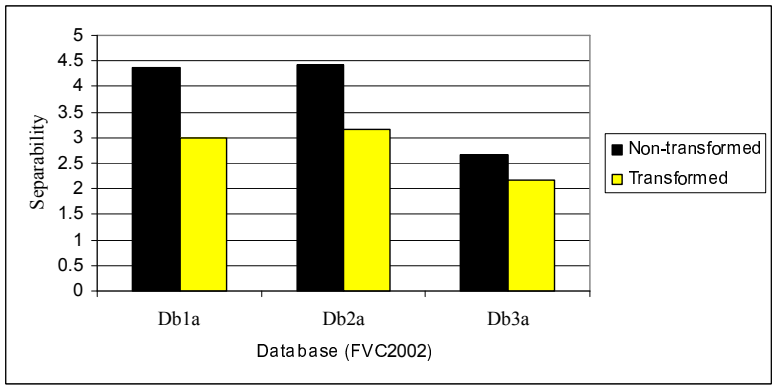


Figure 6.17: Separability of both non-transformed and transformed fingerprint from different databases for transform$_2$.

6.2.4 Non-invertibility

The non-invertibility evaluation of the proposed scheme is based on the assumption that minutiae points are uniformly distributed within a fingerprint, similar to that of Pankanti et al. [74]. It has been described that in the Cartesian-based transformation, the coordinate space is divided $(l + 1)$ times, where l is the specified square level. The transformation rotates the minutiae points $((l + 1) \times r)$ times, where r is the number of minutia rotation per level. The rotation itself is performed iteratively from the highest quadrant level (the

smallest quadrants) to the lowest level (the largest quadrants). Suppose a set of minutiae points refers to the minutiae points in the smallest quadrant. Each set of minutiae points experiences different transformation. As a result, the global structure of minutiae points after the transformation changes. Additionally, there is no information from which each minutia point is transformed. From this, the probability of finding the original location can be represented as $\frac{1}{4(l+1)\times r} \times 100\%$.

Referring to Section 6.1.2, the polar coordinate space consists of $t = \frac{r_{max}}{\omega_t}$ tracks and $s = \frac{s_{max}}{\omega_s}$ sectors, where r_{max} and s_{max} are the radial distance between the center of the polar coordinate space and the outer most track, and the specified maximum angular distance measured from 0^o in a counterclockwise direction, respectively; ω_t and ω_s are the size of tracks and sectors, respectively. A transformed minutia point in track $q, 0 \leq q < t$, is coming from either track q itself or track $(q-1)$; while that in track $(q-1)$ is from either track $(q-1)$ itself or track $(q-2)$ and so on. This is equivalent to a transformed minutia point resulted from the angular transformation. After the transformations, a minutia point has at least $(t-1) \times \omega_t \times (s-1) \times \omega_s$ possible location (coordinates) and $(s-1) \times \omega_s$ possible minutia orientation. It is worth mentioning that minutia orientation is actually not purely random.

Increasing the difficulty of revealing the fingerprint data given its transformed version can be done by increasing either the number of rounds (for both radial and angular transformations), the number of tracks or sectors, the size of tracks or sectors, or combination of them. It is worth pointing out that while r_{max} can be a variable, s_{max} is a constant, which is fixed to 360^0. Thus, increasing the number of sectors will decrease the size of sectors and vice versa; this is equivalent to the case when r_{max} is fixedly defined. Therefore, there is a trade-off between the security and the performance (see Section 6.2.1).

Suppose p is the number of minutiae points in BS and k is the specified number of neighboring minutiae points ($\{m_j\}_{j=1|j\neq i}^{p}$) nearest to the center of coordinate space m_i. The descriptor is derived from a set of k out of $(p-1)$ transformed neighboring minutiae

points. Each minutia point m_i is likely to have different these k neighbors, depending on its relative location to the other minutiae points. Therefore, each minutia point in BS has ${}_{(p-1)}C_k = \frac{(p-1)!}{k!(p-1-k)!}$ possible neighboring minutia point combinations. For example, given $k = 15, p = 30$, there are 77558760 possible descriptors for each minutia. The fact that there is no identity assigned to each minutia point in BS has increased the difficulty in carrying out cross-matching among sets of vectors in $\{ms_i\}_{i=1}^{p}$.

Overall, recovering r'_{i_j}, α'_{i_j} and β'_{i_j} are difficult because of the modulo operation introduced in the transformation (refer to Equations 6.3 and 6.5). Furthermore, if those modulo operations can be solved, then the adversary has to also find the relation between neighboring minutiae points of a center point and neighboring points of other center points (the relation among $\{\{v_{i\_j}\}_{j=1|j\neq i}^{k}\}_{i=1}^{p}$) in order to reconstruct the minutiae points in BS. Yet, to find all minutiae points in B given those in BS, the adversary has to also break Equation 5.3. On the other hand, there is no information leading to B or information of B itself being stored in the database. This has made it infeasible to find the whole information of B.

6.3 Summary

This chapter has proposed a local feature-based cancelable fingeprrint template approach by considering the trade-off between reliability and computation time, which is experienced by that in the previous chapter. This is designed to eliminate the need of singular points and to satisfy the accuracy, revocability, diversity, changeability and security (non-invertibility) factors.

Similar to that in the previous chapter, the proposed approach takes the input only well separated minutiae points whose distance to other minutiae points is greater than the specified threshold. In addition, matching is performed by comparing minutiae descriptors of the query to that of the template. In this case, the descriptor comprises information of only a certain number of transformed neighboring minutiae points.

The transformation itself is performed in two steps. First, the transformation is implemented in the Cartesian coordinate space by dividing this space into several quadrant levels. Minutiae points in every quadrant level are rotated centering on the respective quadrant. Second, the transformation is applied to the polar coordinate space by constructing tracks and sectors, which are used for radial distance and angular distance transformations, respectively. In addition, the minutiae orientation is also transformed based on sectors. Furthermore, the vectors which construct the descriptors are redefined such that the transformation is simpler to implement. This is useful for increasing the computation speed.

The experimental results indicate that the transformation has been able to produce a better performance than that discussed in the previous chapters. In particular, the transformation gives rise to a smaller performance degradation, which is represented by a smaller increase in EER. Furthermore, the EER itself is also lower than that of other surveryed approaches. This better performance is also apparent in both GAR and FAR levels. In terms of processing time, this approach is also better than the previous pair-polar one.

In case the stored template is compromised, a new template can easily be generated by using another set of keys. It is also shown that diversity between transformed fingerprint is high so that cross-matching across databases is infeasible. By compromising only the transformed template, or even both the transformed template and the set of keys, it is still difficult to reconstruct the original fingerprint data because of the use of multiple transformation stages as well as the modulo operation.

Chapter 7

Conclusion

A fingerprint has been a potential authentication tool as a result of its excellence in both performance and user acceptance. The permanence characteristic of a fingerprint has become both its strength and its weakness, in that it not only makes the fingerprint highly verifiable but it also suffers from the forever-effect; it means that once a fingerprint is compromised, it cannot be used again. In order to protect fingerprint data, several approaches have been introduced, including the cancelable fingerprint template design. Many of these, however, experience a relatively high performance degradation. This means that they do not satisfy the transformed fingerprint discriminability property. In general, this thesis has addressed this issue by proposing geometrical approaches of minutiae-based fingerprint authentication algorithms.

This thesis has studied the concept of fingerprint biometrics and state-of-the-art fingerprint data protection schemes. Feature transformation functions along with their corresponding feature representation and matching designs have been proposed and evaluated in terms of: accuracy, revocability, diversity and changeability. Additionally, the security (non-invertibility) of the proposed transformation functions has also been discussed. In more detail, the summary of thesis contributions to fingerprint data protection research is described

in Section 7.1, and some recommendations for future research are provided in Section 7.2.

7.1 Concluding Remarks and Contributions

Even though its focus is on fingerprint security and privacy, this thesis has contributed overall to fingerprint authentication systems. This is because several modules, which construct the system, are independent of each other, specifically those provided in Chapters 5 and 6. This has made it possible to skip the transformation module, thereby creating a common fingerprint authentication system (without fingerprint data protection). Furthermore, some real world scenarios have also been presented in order to measure the overall capability of the proposed approaches.

As expected, similar to that of existing fingerprint data protection approaches, the error rate obtained from implementing the proposed transformation approaches is higher than those that do not use transformation. Nevertheless, these transformation approaches show only a relatively small error rate increase, compared with most existing ones. Furthermore, the error rate itself is also lower. This reflects the strength of the three modules in the proposed fingerprint-based authentication systems: the minutia input selection, the transformation function, and the matching modules. It is worth mentioning that in this performance comparison, the error rate of the proposed global feature-based transformation approach (in Chapter 4) cannot be compared with the corresponding approach without transformation because of the differences in the respective template formats.

The proposed approaches provide the capability to revoke the transformed template in the event that it has been compromised. Revocation can be carried out by generating a new transformed template based on the new set of transformation keys. The research found that this new transformed template meets the diversity property, that is, it does not authenticate the old query derived from the same fingerprint. Furthermore, the proposed approaches also make the transformed template different enough from its non-transformed counterpart so

that they do not authenticate each other. This means that the transformation function has been able to generate new and different fingerprints from an original fingerprint.

In more detail, contributions of each proposed approach are described below.

7.1.1 Projection-based Transformation

The global feature-based (i.e., core-based) transformation function was developed based on the projection approach. Given that accurate core point information is difficult to obtain, this approach does not totally rely on the information provided by the existing extractor. Instead, it allows *tolerant spaces* to be used both before and after the projection.

The proposed approach has been able to deal positively with both performance and security factors, thereby achieving a lower error rate than most other core-based approaches. Some authentication errors found were due to the absence of the core point or minutiae points in the fingerprint. It is worth pointing out that in certain cases, fingerprints may not have the core point. The effect of the intra-user variability (e.g., reordering, insertion and deletion of minutia points) is minimized by quantisizing them through either many-to-one or one-to-one mapping. The discriminability of fingerprints is maintained by using several steps, such as preserving the minutiae orientation, selecting the projection line properties and implementing permutation indexing.

Revocability can be performed by simply changing the key and generating its corresponding transformed template. It has been demonstrated that the use of different keys in the transformation process results in different transformed templates. In general, the use of keys and parameters are useful not only for template revocation purposes, but also for increasing the uniqueness of the template itself. Furthermore, the application of the permutation function to the template vector has increased the uniqueness of the template.

Some techniques have been implemented in this transformation function in order to keep the non-invertibility property, for example, projecting minutiae points onto a line and group-

ing them. It has been indicated that the implementation of this transformation function delivers the difficulty of recovering the original fingerprint data in spite of having the transformed template and even the set of transformation keys and parameter settings.

7.1.2 Pair-polar Coordinate-based Transformation

Considering that not every finger can produce singular point (e.g., core point) information, this proposed approach explores only the relative relation among minutiae points in a pair-polar framework. In particular, a minutia point is described by its neighboring minutiae points. As it has been able to eliminate the need of any singular point information, the proposed scheme is more reliable than singular point-based approaches. Furthermore, it has been able to minimize the effect caused by the placing of a finger in different positions on the scanner, as shown in the experimental results. This means that the cancelable fingerprint template designed in this research is shift- and rotation-free.

In order to provide an input to the transformation function, a set of minutiae points is selected. This set of points reduces the system complexity since it only needs to process a smaller number of minutiae points. Selection is made according to the minutia radial and angular distances, that is, only minutiae points whose distance to other minutiae points is greater than the specified thresholds are included in the set.

The transformation is performed by referring to the sectors, which are constructed in a polar coordinate space. After being transformed, the minutiae points are represented by sets of vectors, which are to be the template stored in the database. The matching module compares the transformed template with the transformed query and determines their similarities.

Results show that there is only a relatively small error rate increase from before to after the transformation. Furthermore, the other requirements (i.e., revocability, diversity, changeability) have also been satisfied. The proposed approach offers security (non-invertibility)

in several ways. First, it performs a many-to-one mapping for transforming the sectors. In the event that the transformed template is compromised, the adversary will experience uncertainty in reversing that mapping. This means that the difficulty of determining the original data raises. Second, in the event that the adversary is able to break this mapping barrier, he/she has to solve the modulo operation, which has been implemented to protect the radial minutia distance. Therefore, it is infeasible that the original fingerprint data can be recovered, even though in the worst scenario of both the transformed template and the key being compromised.

7.1.3 Cartesian and Polar Coordinate-based Transformation

This local feature-based cancelable template approach is more advanced than the previous proposed pair-polar coordinate-based transformation. This is developed by employing both Cartesian and polar coordinate spaces. Specifically, the minutiae points are rotated in the Cartesian coordinate space, and rotated and translated in the polar coordinate space by firstly dividing those respective coordinate spaces into a certain level of squares (quadrants) and tracks/sectors.

Similar to the previous proposed pair-polar coordinate-based transformation, this approach assigns a descriptor to each minutia point. Different from that, this minutia descriptor is constructed from a certain number of the nearest neighboring minutiae points only. Therefore, these neighboring points may be different from one minutia point to another. There are at least two advantages in implementing this approach. First, it minimizes the possibility of cross-matching among descriptors, making it unnecessary to use multiple sets of keys in transforming the minutiae points. Second, it reduces the amount of the descriptor content, thereby increasing computation speed (the speed has been a possible drawback in the previous pair-polar transformation). Moreover, this is also an improvement in making the approach more implementable in resource constraint devices.

The transformation itself is carried out in several steps, each of them containing multiple rounds (in the experiments, it is set to two). This makes it more difficult for the adversary to recover the original fingerprint data even if both the transformed template and the set of transformation keys have been compromised. At the same time, the transformation further implements another securing technique: many-to-one mapping, in each step. Overall, this scheme provides substantial performance and security improvements to the secure fingerprint authentication system.

In terms of diversity and changeability, this approach presents low error rates (close to zero), similar to those in the previous polar-based transformation. A performance comparison with surveyed fingerprint data protection approaches has been provided. From this, it can be inferred that this proposed approach has lower error rates than those of the others.

7.2 Future Research

This thesis has proposed the implementation of both global and local feature-based transformation approaches which outperform surveyed schemes. Nevertheless, there are still opportunities to improve on these approaches. These can be classified into: distance of keys, feature extraction, the design of biometric multi-modalities and the execution time.

Although discussion about key variation has been provided in the proposed algorithms, it is useful to specify the minimum distance of keys such that same fingerprints lead to different transformed data as shown in Equations 7.1 and 7.2.

$$\left.\begin{aligned} & B_{sec} = \Gamma(B, \kappa) \\ & B'_{sec} = \Gamma(B, \kappa') \\ & \kappa = \{\kappa_1, \kappa_2, ..., \kappa_n\}, \kappa' = \{\kappa'_1, \kappa'_2, ..., \kappa'_n\} \\ & \kappa_1 \neq \kappa'_1, \kappa_2 \neq \kappa'_2, ..., \kappa_n \neq \kappa'_n \\ & B_{sec} \neq B'_{sec} \end{aligned}\right\} \tag{7.1}$$

$$\left.\begin{array}{l} |\kappa_1 - \kappa_1'| > \tau_{k1} \\ |\kappa_2 - \kappa_2'| > \tau_{k2} \\ ..... \\ |\kappa_n - \kappa_n'| > \tau_{kn} \end{array}\right\} \tag{7.2}$$

where B is a non-transformed fingerprint, B_{sec} and B'_{sec} are transformed fingerprints generated by a function Γ, and keys κ and κ', respectively; τ_{k1}, τ_{k2} and τ_{kn} are the minimum distance between corresponding keys.

Ideally, the minimum distance $\tau_{k1}, \tau_{k2}, ..., \tau_{kn}$ are very low. This means that small differences of corresponding keys result in different transformed fingerprint data. In other words, the security level of the transformed fingerprint data does not depend on the variation of keys. A possible way to achieve this condition is by hashing the keys, as applied in the Cartesian and polar coordinate-based transformation in Chapter 6.

On the one hand, the global feature-based transformation approach has an excellent performance in some aspects [96], including authentication speed (refer to Chapter 4). On the other hand, its reliability is challenged by the unavailability of the singular points in the arch fingerprint class. An alternate point could be defined to be an anchor replacing the role of the core point in this fingerprint class. This new point definition is likely to make the global feature-based transformation more reliable because the transformation does not rely just on the existence of the core point. This alternate point extraction module could then be combined with other modules in a transformed fingerprint authentication system. An example of alternate point is described in [64], where it is located in the maximum ridge curvature.

The drawbacks of fingerprint-based authentication systems, such as those occurring in the global feature-based transformation, can be minimized by combining fingerprints with other biometric modalities. The comparison between biometric modalities themselves (for an

authentication purpose) has been made by Jain et al. [46], Uludag et al. [100] and Maltoni et al. [64]. In general, it is shown that there is no biometrics superior in all aspects, including fingerprints themselves. Ideally, the strength of each biometric modality is combined to minimize the error level and at the same time to increase the security, privacy and suitability of the users. It is also expected that the biometric system is more reliable because of this multiple and independent biometric data [46, 82]. In combining these biometric modalities, there are specific topics, which should be defined. Those are: (i) what biometric modality is more appropriate to combine with fingerprints, whether face, iris, palm print or other modalities; (ii) how the biometric modalities are combined, whether single biometrics multiple matchers, single biometrics multiple representations, or biometric fusion, as specified in [103]; (iii) what level the combination is performed at, whether sensor, feature, match score, rank or decision levels, as described in [46].

Yet, the implementation of multi-modal biometrics not only affects the performance but also the execution time because, of course, the more biometrics involved in the process, the more complex the system. Even though advanced hardware and software technologies have made it possible for this authentication system running faster, the refinement of the proposed schemes is still an important factor, especially to the local feature-based transformation (refer to Chapters 5 and 6). In this case, a higher number of minutiae points involved in the authentication process significantly increases the execution time because fingerprint matching is carried out by comparing all minutiae points in the template with all minutiae points in the query (many-to-many minutia point comparison).

The local feature-based transformation functions proposed in this thesis have reduced this minutia number. Nevertheless, other methods which can further reduce this minutia number and concurrently increase the performance is desirable. This minutia selection could be done by, for example, finding the optimal value of minutiae distance and minutia orientation weights.

Bibliography

[1] T. Ahmad and F. Han. Cartesian and polar transformation-based cancelable fingerprint template. In *The 37th Annual Conference of the IEEE Industrial Electronics Society*, pages 373–378, 2011.

[2] T. Ahmad and J. Hu. Generating cancelable biometric templates using a projection line. In *The 11th IEEE International Conference on Control Automation Robotics & Vision*, pages 7–12, 2010.

[3] T. Ahmad, J. Hu, and S. Han. An efficient mobile voting system security scheme based on elliptic curve cryptography. In *The 3rd IEEE International Conference on Network and System Security*, pages 474–479, 2009.

[4] T. Ahmad, J. Hu, and S. Wang. Pair-polar coordinate based cancelable fingerprint templates. *Pattern Recognition*, 44(10-11):2555–2564, 2011.

[5] T. Ahmad, J. Hu, and S. Wang. String-based cancelable fingerprint templates. In *The 6th IEEE Conference on Industrial Electronics and Applications*, pages 1028–1033, 2011.

[6] D. Ahn, S. G. Kong, Y.-S. Chung, and K. Y. Moon. Matching with secure fingerprint templates using non-invertible transforms. In *Congress on Image and Signal Processing*, pages 29–33, 2008.

[7] A. Alessandroni, R. Cappelli, M. Ferrara, and D. Maltoni. Definition of fingerprint scanner image quality specifications by operational quality. In *Biometrics and Identity Management, LNCS 537*, pages 29–36, 2008.

[8] R. Ang, R. Safavi-Naini, and L. McAven. Cancelable key-based fingerprint templates. In *Information Security and Privacy, LNCS 3574*, pages 109–128, 2005.

[9] ANSI/NIST-ITL. American national standard for information systems-data format for the interchange of fingerprint, facial, & scar mark & tattoo (smt) information, 2000. URL http://www.nist.gov/itl/ansi/upload/sp500-245-a16.pdf.

[10] A. Arakala. *Secure and Private Fingerprint-based Authentication*. PhD thesis, School of Mathematical and Geospatial Sciences, RMIT University, Melbourne, Australia, November 2008.

[11] A. Arakala, J. Jeffers, and K. J. Horadam. Fuzzy extractors for minutiae-based fingerprint authentication. In *Advances in Biometrics, LNCS 4642*, pages 760–769, 2007.

[12] D. R. Ashbaugh. *QuantitativeQualitative Friction Ridge Analysis: An Introduction to Basic and Advanced Ridgeology*. CRC Press, Boca Raton, FL, 1999.

[13] P. Belhumeur, J. Hespanha, and D. Kriegman. Eigenfaces versus fisherfaces: Recognition using class specific linear projection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[14] 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, 2003.

[15] R. M. Bolle, J. H. Connell, and N. K. Ratha. Biometric perils and patches. *Pattern Recognition*, 35(12):2727–2738, 2002.

[16] R. M. Bolle, N. K. Ratha, and S. Pankanti. Error analysis of pattern recognition systems - the subsets bootstrap. *Computer Vision and Image Understanding*, 93(1): 1–33, 2004.

[17] X. Boyen. Reusable cryptographic fuzzy extractors. In *The 11th ACM Conference on Computer and Communications Security*, pages 82–91, 2004.

[18] J. Bringer, H. Chabanne, and B. Kindarji. Anonymous identification with cancelable biometrics. In *The 6th International Symposium on Image and Signal Processing and Analysis*, pages 494–499, 2009.

[19] R. Cappelli, D. Lumini, D. Maio, and D. Maltoni. Fingerprint image reconstruction from standard templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(9):1489–1503, 2007.

[20] R. Cappelli, M. Ferrara, and D. Maltoni. On the operational quality of fingerprint scanners. *IEEE Transactions on Information Forensics and Security*, 3(2):192–202, 2008.

[21] A. Cavoukian, A. Stoianov, and F. Carter. Biometric encryption: Technology for strong authentication, security and privacy. *IFIP International Federation for Information Processing*, 261:57–77, 2008.

[22] A. Ceguerra and I. Koprinska. Automatic fingerprint verification using neural networks. page 136, 2002.

[23] E.-C. Chang and S. Roy. Robust extraction of secret bits from minutiae. In *Advances in Biometrics, LNCS 4642*, pages 750–759, 2007.

[24] K.-H. Cheung, A. Kong, D. Zhang, M. Kamel, and J. You. Revealing the secret of facehashing. In *Advances in Biometrics, LNCS 3822*, pages 106–112, 2005.

[25] S. Chikkerur, A. N. Cartwright, and V. Govindaraju. Fingerprint enhancement using stft analysis. *Pattern Recognition*, 40(1):198–211, 2007.

[26] S. Chikkerur, N. Ratha, J. Connell, and R. Bolle. Generating registration-free cancelable fingerprint templates. In *The 2nd IEEE International Conference on Biometrics: Theory, Applications and Systems*, pages 1–6, 2008.

[27] C. S. Chin, A. T. B. Jin, and D. N. C. Ling. High security iris verification system based on random secret integration. *Computer Vision and Image Understanding*, 102(2):169–177, 2006.

[28] T. Connie, A. Teoh, M. Goh, and D. Ngo. Palmhashing: a novel approach for cancelable biometrics. *Information Processing Letters*, 93(1):1–5, 2005.

[29] B. Cukic and N. Bartlow. Biometric system threats and countermeasures: A risk based approach, 2005. URL http://www.biometrics.org/bc2005/Presentations/Conference/2%20Tuesday%20September%2020/Tue_Ballroom%20B/Cukic_Threats%20and%20countermeasures.pdf.

[30] Y. Dodis, J. Katz, L. Reyzin, and A. Smith. Robust fuzzy extractors & authenticated key agreement from close secrets. URL https://www.ipam.ucla.edu/publications/scws3/scws3_6769.ppt.

[31] Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *Advances in Cryptology (EUROCRYPT'04), LNCS 3027*, pages 523–540, 2004.

[32] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *Tech. Rep. 235, Cryptology ePrint Archive*, 2006.

[33] Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors, 2008. URL http://www.cse.psu.edu/~asmith/pubs/2008/fuzzysurvey.pdf.

[34] F. Farooq, R. Bolle, J. Tsai-Yang, and N. Ratha;. Anonymous and revocable fingerprint recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7, 2007.

[35] F. Farooq, N. Ratha, J. Tsai-Yang, and R. Bolle;. Security and accuracy trade-off in anonymous fingerprint recognition. In *The Ist IEEE International Conference on Biometrics: Theory, Applications and Systems*, pages 1–6, 2007.

[36] T. Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[37] Y. Feng and P. Yuen. Protecting face biometric data on smartcard with reed-solomon code. In *The IEEE Conference on Computer Vision and Pattern Recognition Workshop*, page 29, 2006.

[38] Y. Feng, J. Li, F. Han, and T. Ahmad. A novel image encryption method based on invertible 3d maps and its security analysis. In *The 37th Annual Conference of the IEEE Industrial Electronics Society*, pages 2186–2191, 2011.

[39] S. Furnell, P. Dowland, H. Illingworth, and P. Reynolds. Authentication and supervision: A survey of user attitudes. *Computers & Security*, 19(6):529–539, 2000.

[40] R. Germain, A. Califano, and S. Colville. Fingerprint matching using transformation parameter clustering. *IEEE Computational Science & Engineering*, 4(4):42–49, 1997.

[41] F. Hao, R. Anderson, and J. Daugman. Combining crypto with biometrics effectively. *IEEE Transaction on Computers*, 55(9):1081–1088, 2006.

[42] R. J. Hyndman and A. B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688, 2006.

[43] V. I. Ivanov and J. S. Baras. Authentication of fingerprint scanners. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1912–1915, 2011.

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

[45] A. Jain, K. Nandakumar, and A. Nagar. Biometric template security. *EURASIP Journal on Advances in Signal Processing*, 2008:1–17, 2008.

[46] A. K. Jain, A. Ross, and S. Pankanti. Biometrics: a tool for information security. *IEEE Transactions on Information Forensics and Security*, 1(2):125–143, 2006.

[47] L. Ji, Z. Yi, L. Shang, and X. Pu. Binary fingerprint image thinning using template-based pcnns. *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, 37(5):1407–1413, 2007.

[48] X. Jiang and W. Yau. Fingerprint minutiae matching based on the local and global structures. In *The 15th International Conference on Pattern Recognition*, volume 2, pages 1038–1041, 2000.

[49] Z. Jin, A. B. J. Teoh, T. S. Ong, and C. Tee. Secure minutiae-based fingerprint templates using random triangle hashing. In *Visual informatics: Bridging research and practice, LNCS 5857*, pages 521–531, 2009.

[50] A. Juels and M. Sudan. A fuzzy vault scheme. *Designs, Codes and Cryptography*, 38(2):237–257, 2006.

[51] A. Juels and M. Wattenberg. A fuzzy commitment scheme. In *The 6th ACM Conference on Computer and Communications Security*, pages 28–36, 1999.

[52] H. Kang, B. Lee, H. Kim, D. Shin, and J. Kim. A study on performance evaluation of the liveness detection for various fingerprint sensor modules. *Knowledge-Based Intelligent Information and Engineering Systems, LNCS 2774*, pages 1245–1253, 2003.

[53] T. Kevenaar. Protection of biometric information. *Security with noisy data: Private biometrics, secure key storage and anti-counterfeiting*, pages 169–193, 2007.

[54] A. Kong, K.-H. Cheung, D. Zhang, M. Kamel, and J. You. An analyzing of biohashing and its variants. *Pattern Recognition*, 39 (7):1359–1368, 2006.

[55] J. T. Kukunas, R. D. Cupper, and G. M. Kapfhammer. A genetic algorithm to improve linux kernel performance on resource-constrained devices, 2010. URL http://www.cs.allegheny.edu/~gkapfham/research/publish/Kukunas_gecco2010.pdf.

[56] C. Lee and J. Kim. Cancelable fingerprint templates using minutiae-based bit-strings. *Journal of Network and Computer Applications*, 33(3):236–246, 2010.

[57] C. Lee, J.-Y. Choi, K.-A. Toh, and S. Lee. Alignment-free cancelable fingerprint templates based on local minutiae information. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 37(4):980–992, 2007.

[58] Y. Lee, K. Bae, S. Lee, K. Park, and J. Kim. Biometric key binding: Fuzzy vault based on iris images. In *Advances in Biometrics, LNCS 4642*, pages 800–808, 2007.

[59] Q. Li, M. Guo, and E.-C. Chang. Fuzzy extractors for asymmetric biometric representations. In *The IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–6, 2008.

[60] D. Maio, D. Maltoni, R. Cappelli, J. Wayman, and A. K. Jain. Fvc2000: fingerprint verification competition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):402–412, 2002.

[61] D. Maio, D. Maltonil, R. Cappelli, J. Wayman, and A. K. Jain. Fvc2002: Second fingerprint verification competition. In *16th International Conference on Pattern Recognition, 2002*, volume 3, pages 811–814, 2002.

[62] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, and A. K. Jain. Fvc2004: Third fingerprint verification competition. In *Biometric Authentication, LNCS 3072*, pages 1–7, 2004.

[63] E. Maiorana, P. Campisi, J. Fierrez, J. Ortega-Garcia, and A. Neri. Cancelable templates for sequence-based biometrics with application to on-line signature recognition. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 40(3):525–538, 2010.

[64] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar. *Handbook of Fingerprint Recognition*. Springer, 2009.

[65] K. Nandakumar, A. K. Jain, and S. Pankanti. Fingerprint-based fuzzy vault: Implementation and performance. *IEEE Transactions on Information Forensics and Security*, 2(4):744–757, 2007.

[66] L. Nanni and A. Lumini. A deformation-invariant image-based fingerprint verification system. *Neurocomputing*, 69(16-18):2336–2339, 2006.

[67] L. Nanni and A. Lumini. Random subspace for an improved biohashing for face authentication. *Pattern Recognition Letters*, 29(3):295–300, 2008.

[68] L. Nanni and A. Lumini. Descriptors for image-based fingerprint matchers. *Expert Systems with Applications*, 39(10):12414–12422, 2009.

[69] Neurotechnology. Verifinger version 5.0, 2006. URL http://www.neurotechnology.com.

[70] NIST. National institute of standards and technology, 2010. URL http://www.nist.gov/index.html.

[71] M. Oliveira and N. Leite. A multiscale directional operator and morphological tools for reconnecting broken ridges in fingerprint images. *Pattern Recognition*, 41(1):367–377, 2008.

[72] J. W. Osterburg, T. Parthasarathy, T. E. S. Raghavan, and S. L. Sclove. Development of a mathematical formula for the calculation of fingerprint probabilities based on individual characteristics. *Journal of the American Statistical Association*, 72(360): 772–778, 1977.

[73] G. Ottoy, T. Hamelinckx, B. Prenee, L. D. Strycker, and J.-P. Goemaere. Aes data encryption in a zigbee network: Software or hardware? In *Security and Privacy in Mobile Information and Communication Systems, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, 47*, pages 163–173, 2010.

[74] S. Pankanti, S. Prabhakar, and A. Jain. On the individuality of fingerprints. *IEEE Transaction on Pattern Analysis Machine Intelligence*, 24(8):1010–1025, 2002.

[75] S. Prabhakar, S. Pankanti, and A. Jain. Biometric recognition: Security and privacy concerns. *Security & Privacy*, 1(2):33–42, 2003.

[76] F. Quan, S. Fei, C. Anni, and Z. Feifei. Cracking cancelable fingerprint template of

ratha. In *International Symposium on Computer Science and Computational Technology*, pages 572–575, 2008.

[77] N. Ratha, J. Connell, R. M. Bolle, and S. Chikkerur. Cancelable biometrics: A case study in fingerprints. In *The 18th International Conference on Pattern Recognition*, volume 4, pages 370–373, 2006.

[78] N. K. Ratha, J. H. Connell, and R. M. Bolle. Enhancing security and privacy in biometrics-based authentication systems. *IBM Systems Journal*, 40(3):614–634, 2001.

[79] N. K. Ratha, J. H. Connell, and R. M. Bolle. Biometrics break-ins and band-aids. *Pattern Recognition Letters*, 24:2105–2113, 2003.

[80] N. K. Ratha, S. Chikkerur, J. H. Connell, and R. M. Bolle. Generating cancelable fingerprint templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4):561–572, 2007.

[81] C. Roberts. Biometric attack vectors and defences. *Computers and Security*, 26(1): 14–25, 2007.

[82] A. Ross, K. Nandakumar, and A. Jain. *Handbook of Multibiometrics*. Springer, NY, 2006.

[83] A. Ross, J. Shah, and A. Jain. From template to image: Reconstructing fingerprints from minutiae points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4):544–560, 2007.

[84] E. Sano, T. Maeda, T. Nakamura, M. Shikai, K. Sakata, M. Matsushita, and K. Sasakawa. Fingerprint authentication device based on optical characteristics inside a finger. In *The Conference on Computer Vision and Pattern Recognition Workshop*, pages 354–358, 2006.

[85] W. J. Scheirer and T. E. Boult. Cracking fuzzy vaults and biometric encryption. In *2007 Biometrics Symposium*, pages 1–6, 2007.

[86] B. Sherlock and D. Monro. A model for interpreting fingerprint topology. *Pattern Recognition*, 26(7):1047–1055, 1993.

[87] Y. Shibata, M. Mimura, K. Takahashi, and M. Nishigaki. A study on biometric key generation from fingerprints: Fingerprint-key generation from stable feature value. In *Conference on Security and Management*, pages 45–51, 2007.

[88] S. W. Shin, M.-K. Lee, D. Moon, and K. Moon. Dictionary attack on functional transform-based cancelable fingerprint templates. *ETRI Journal*, 31(5):628–630, 2009.

[89] K. Simoens, P. Tuyls, and B. Preneel. Privacy weakness in biometric sketches. In *The 30th IEEE Symposium on Security and Privacy*, pages 188–202, 2009.

[90] L. Sun, G. Pan, Z. Wu, and S. Lao. Blinking-based live face detection using conditional random fields. *Advances in Biometrics, LNCS 4642*, pages 252–260, 2007.

[91] Y. Sutcu, Q. Li, and N. Memon. Protecting biometric templates with sketch: Theory and practice. *IEEE Transactions on Information Forensics and Security*, 2(3) part 2: 503–512, 2007.

[92] Y. Sutcu, H. Sencar, and N. Memon. A geometric transformation to protect minutiae-based fingerprint template. *SPIE: Biometric Technology for Human Identification IV*, 6539, 2007.

[93] B. Tan and S. Schuckers. Spoofing protection for fingerprint scanner by fusing ridge signal and valley noise. *Pattern Recognition*, 43(8):2845–2857, 2010.

[94] A. B. J. Teoh, D. C. L. Ngo, and A. Goh. Biohashing: two factor authentication

featuring fingerprint data and tokenised random number. *Pattern Recognition*, 37(11): 2245–2255, 2004.

[95] A. B. J. Teoh, A. Goh, and D. C. L. Ngo. Random multispace quantization as an analytic mechanism for biohashing of biometric and random identity inputs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):1892–1901, 2006.

[96] A. O. Thomas, N. Ratha, J. Connell, and R. Bolle. Comparative analysis of registration based and registration free methods for cancelable fingerprint biometrics. In *The 19th International Conference on Pattern Recognition*, pages 1–8, 2008.

[97] D. Ting. Under lock and key keeping sensitive data where it belongs. *Biometric Technology Today*, 2010(5):10–12, 2010.

[98] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[99] U. Uludag and A. Jain. Securing fingerprint template: Fuzzy vault with helper data. In *The IEEE Conference on Computer Vision and Pattern Recognition Workshop*, page 163, 2006.

[100] U. Uludag, S. Pankanti, S. Prabhakar, and A. K. Jain. Biometric cryptosystems: Issues and challenges. *Proceedings of the IEEE*, 92(6):948–960, 2004.

[101] W. Wang, J. Li, F. Huang, and H. Feng. Design and implementation of log-gabor filter in fingerprint image enhancement. *Pattern Recognition Letters*, 29(3):301–308, 2008.

[102] X. Wang, J. Lia, and Y. Niu. Definition and extraction of stable points from fingerprint images. *Pattern Recognition*, 40(6):1804–1815, 2007.

[103] Y. Wang. *Ridge Orientation Modeling and Feature Analysis for Fingerprint Identifi-*

cation. PhD thesis, School of Computer Science and Information Technology, RMIT University, Melbourne, Australia, August 2008.

[104] Y. Wang and J. Hu. Global ridge orientation modeling for partial fingerprint identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):72–87, 2011.

[105] Y. Wang, J. Hu, and D. Philip. A fingerprint orientation model based on 2d fourier expansion (fomfe) and its application to singular-point detection and fingerprint indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4):573–585, 2007.

[106] S. B. Wicker. *Error Control Systems for Digital Communication and Storage*. Englewood Cliffs, NJ : Prentice Hall, 1995.

[107] C. Wilson, G. Candela, and C. Watson. Neural network fingerprint classfication. *Journal of Artificial Neural Networks*, 1(2):203228, 1993.

[108] K. Xi and J. Hu. Biometric mobile template protection: A composite feature based fingerprint fuzzy vault. In *The IEEE International Conference on Communications*, pages 1–5, 2009.

[109] K. Xi, T. Ahmad, F. Han, and J. Hu. A fingerprint based bio-cryptographic security protocol designed for client/server authentication in mobile computing environment. *Security and Communication Networks*, 4(5):487–499, 2011.

[110] H. Xu, R. Veldhuis, A. Bazen, T. Kevenaar, T. Akkermans, and B. Gokberk. Fingerprint verification using spectral minutiae representations. *IEEE Transactions on Information Forensics and Security*, 4(3):397–409, 2009.

[111] N. Yager. *Hierarchical Fingerprint Verification*. PhD thesis, School of Computer Science and Engineering, University of New South Wales, Sidney, Australia, April 2007.

[112] N. Yager and A. Amin. Dynamic registration selection for fingerprint verification. *Pattern Recognition*, 39(11):2141–2148, 2006.

[113] H. Yang, X. Jiang, and A. C. Kot. Generating secure cancelable fingerprint templates using local and global features. In *The 2nd IEEE International Conference on Computer Science and Information Technology*, pages 645–649, 2009.

[114] J. Yang, J. W. Shin, B. Min, J. Park, and D. Park. Fingerprint matching using invariant moment fingercode and learning vector quantization neural network. In *International Conference on Computational Intelligence and Security*, pages 735–738, 2006.

[115] Y. Zhang and Q. Xiao. An optimized approach for fingerprint binarization. In *International Joint Conference on Neural Networks*, pages 391–395, 2006.

[116] J. Zhou, J. Gu, and D. Zhang. Singular points analysis in fingerprints based on topological structure and orientation field. In *Advances in Biometrics, LNCS 4642*, pages 261–270, 2007.

[117] Y. Zhu, S. C. Dass, and A. K. Jain. Statistical models for assessing the individuality of fingerprints. *IEEE Transactions on Information Forensics and Security*, 2(3):39– 401, 2007.