A key binding system based on $n$-nearest minutiae structure of fingerprint

Eryun Liua,b, Heng Zhaoc, Jimin Lianga,* Liaojun Pangb, Min Xiec, Hongtao Chena,b, Yanhua Lia,b, Peng Lid, Jie Tiana,d

*Life Sciences Research Center, School of Life Sciences and Technology, Xidian University, Xian, Shaanxi 710071, China
bSchool of Electronic Engineering, Xidian University, Xian, Shaanxi 710071, China
cMinistry of Education Key Lab. of Computer Network and Information Security, School of Telecommunication Engineering, Xidian University, Xian, Shaanxi 710071, China
dInstitute of Automation, Chinese Academy of Sciences, Beijing 100190, China

A R T I C L E   I N F O

Article history:
Received 17 December 2009
Available online 30 December 2010
Communicated by G. Borgefors

Keywords:
Key binding
Nearest minutiae structure
Wrap-around
Secure sketch
PinSketch
Shamir’s secret sharing

A B S T R A C T

Biometric cryptosystem has gained increasing attention in recent years. One of the difficulties in this field is how to perform biometric matching under template protection. In this paper, we propose a key binding system based on $n$-nearest minutiae structures of fingerprint. Unlike the traditional fingerprint recognition method, the matching of nearest structures are totally performed in the encrypted domain, where the template minutiae are protected. Three levels of secure sketch are applied to deal with error correction and key binding: (1) The wrap-around construction is used to tolerate random errors that happens on paired minutiae; (2) the PinSketch construction is used to recover nearest structures which are disturbed by burst errors; and (3) Shamir’s secret sharing scheme is used to bind and recover a key based on template minutia structures. The experimental results on PVC2002 DB1 and DB2 and security analysis show that our system is efficient and secure.

1. Introduction

Biometrics provide a convenient and reliable way of identity authentication (Jain et al., 2007), which is based on the intrinsic aspects of human body or activities (e.g., fingerprint (Maltoni et al., 2009), iris (Daugman, 2005), keystroke (Bleha et al., 1990) and voice (Jelinek, 1999)). The matching of features in a traditional biometric system needs the naked template data to be stored in a central server or smart card, which causes severe problems if the central server has been hacked by attackers (Jain et al., 2005, 2006, 2008). In the field of cryptography, cryptographic key is widely used in modern security systems (e.g., AES, RSA). The security of modern systems bases heavily on the secret key. A cryptographic key is usually a long and random bit string, which makes key management a challenging problem and adds burden to the users (Stallings, 2005). Combining biometrics and cryptography together will settle the problems of both fields (Cavoukian et al., 2007; Tuyls et al., 2008).

Dodis et al. (2004) provided a general framework on how to extract randomly and uniformly distributed cryptographic key from noise data, including biometric features. The techniques are called secure sketch and fuzzy extractor. Based on their method, secret keys can be extracted from biometrics represented as canonical forms, such as bit vector, real valued vector, set. Such forms of representations can be measured in a simple way, e.g., Hamming metric for bit vectors and Euclidean metric for real valued vectors. More recent progresses on secure sketch and fuzzy extractor are referred to Kanukurthi and Reyzin (2008), Dodis et al. (2006a, 2007, 2008), Boyen (2004), Buhan et al. (2007), Cramer et al. (2008).

As a popular biometric, fingerprint shares the most of the markets. Most fingerprint matching algorithms are based on minutiae. The matching of minutiae is not an easy task, especially when the fingerprint images are nonlinearly deformed during data acquisition. Although many progresses have been made for the biometric systems without templates protection (Maltoni et al., 2009), the matching of minutia set under secure sketch is still an open problem, because the original templates are secured and unavailable at the verification stage, also the way of similarity measure is limited. These drawbacks severely degrade the recognition accuracy of a fingerprint recognition system compared with systems without template protection.

A general and straightforward method to deal with the similarity measure problem is to transform the minutia set into a fixed-length feature vector because of its convenience to the existing secure sketch constructions, such as fuzzy commitment scheme (Juels and Wattenberg, 1999), code-offset scheme (Golić and Baltatu, 2008), quantization method (Linnartz and Tuyls, 2003; Li et al., 2006), or concatenated error correction method (Hao et al., 2006). Xu et al. (2008, 2009) proposed to use continuous Fourier–Mellin transform on minutia set to obtain a fixed-length

1. Introduction

Biometrics provide a convenient and reliable way of identity authentication (Jain et al., 2007), which is based on the intrinsic aspects of human body or activities (e.g., fingerprint (Maltoni et al., 2009), iris (Daugman, 2005), keystroke (Bleha et al., 1990) and voice (Jelinek, 1999)). The matching of features in a traditional biometric system needs the naked template data to be stored in a central server or smart card, which causes severe problems if the central server has been hacked by attackers (Jain et al., 2005, 2006, 2008). In the field of cryptography, cryptographic key is widely used in modern security systems (e.g., AES, RSA). The security of modern systems bases heavily on the secret key. A cryptographic key is usually a long and random bit string, which makes key management a challenging problem and adds burden to the users (Stallings, 2005). Combining biometrics and cryptography together will settle the problems of both fields (Cavoukian et al., 2007; Tuyls et al., 2008).

Dodis et al. (2004) provided a general framework on how to extract randomly and uniformly distributed cryptographic key from noise data, including biometric features. The techniques are called secure sketch and fuzzy extractor. Based on their method, secret keys can be extracted from biometrics represented as canonical forms, such as bit vector, real valued vector, set. Such forms of representations can be measured in a simple way, e.g., Hamming metric for bit vectors and Euclidean metric for real valued vectors. More recent progresses on secure sketch and fuzzy extractor are referred to Kanukurthi and Reyzin (2008), Dodis et al. (2006a, 2007, 2008), Boyen (2004), Buhan et al. (2007), Cramer et al. (2008).

As a popular biometric, fingerprint shares the most of the markets. Most fingerprint matching algorithms are based on minutiae. The matching of minutiae is not an easy task, especially when the fingerprint images are nonlinearly deformed during data acquisition. Although many progresses have been made for the biometric systems without templates protection (Maltoni et al., 2009), the matching of minutia set under secure sketch is still an open problem, because the original templates are secured and unavailable at the verification stage, also the way of similarity measure is limited. These drawbacks severely degrade the recognition accuracy of a fingerprint recognition system compared with systems without template protection.

A general and straightforward method to deal with the similarity measure problem is to transform the minutia set into a fixed-length feature vector because of its convenience to the existing secure sketch constructions, such as fuzzy commitment scheme (Juels and Wattenberg, 1999), code-offset scheme (Golić and Baltatu, 2008), quantization method (Linnartz and Tuyls, 2003; Li et al., 2006), or concatenated error correction method (Hao et al., 2006). Xu et al. (2008, 2009) proposed to use continuous Fourier–Mellin transform on minutia set to obtain a fixed-length
feature vector which is invariant under all kinds of rigid transformation. The fixed-length feature vector can be conveniently combined with secure sketch. But in (Xu et al., 2008, 2009), no concrete implementation of secure sketch on this feature were provided, and much of the discriminative ability is lost during the feature transformation. In (Chang and Roy, 2007), the authors proposed to extract bits from minutiae by comparing the number of minutiae of two sides of considerable random lines. A similar method was presented in (Sucu et al., 2008), where the random lines were substituted by random cuboids.

Another approach to deal with the minutia set matching problem under secure sketch is to construct multiple secure sketches and directly operate on the minutiae set, where the similarity measure has to consider both Euclidean distance and set difference at the same time. Fingerprint-based fuzzy vault (Nandakumar et al., 2007) is a common way to deal with minutia matching problem. The main problem of the fingerprint fuzzy vault in (Nandakumar et al., 2007) is cross-matching through applications (Scheirer and Boult, 2007), and the number of input and output minutiae must be fixed to a certain value, which makes it hard to select the high noises measured in different metric spaces. The advantage of this way of matching is performed by multiple secure sketch constructions. In consideration of the diversity of noises in minutia set, the number of minutiae in l. For a minutia \( m_i \) in \( M \), nearest minutiae \( \{m_1, m_2, \ldots, m_n\} \) are found, where \( m_{ik} \in M, k = 1, \ldots, n \). Fig. 1 shows an example of the nearest minutiae topological structure of \( m \). Feature vector \( (d_{ik}, x_{ik}, \beta_{ik}) \) is used to describe the relationship between \( m_i \) and \( m_{ik} \), where \( d_{ik} = (0 \leq d_{ik} \leq d_{max}) \) denotes the distance between \( m_i \) and \( m_{ik} \), \( x_{ik} = (0 \leq x_{ik} \leq 359) \) denotes the minimal angle by clockwise rotating the radial line \( m_i m_{ik} \) to the orientation of minutia \( m_i \), and \( \beta_{ik} = (0 \leq \beta_{ik} \leq 359) \) denotes the minimal angle by counterclockwise rotating radial line \( m_i m_{ik} \) to the orientation of minutia \( m_{ik} \). The angular values, \( x_{ik} \) and \( \beta_{ik} \), are measured in degrees and rounded to the nearest integers. The radius \( d_{ik} \) are rounded to the nearest integers too. \( d_{max} \) is the maximum radius value and can be determined by experiments. \( (d_{ik}, x_{ik}, \beta_{ik}) \) is invariant under translation and rotation of the fingerprint image. The nearest topological structure of \( m_i \) is defined as

\[
NS_i = (u_i, v_i, \theta_i, d_{i1}, x_{i1}, \beta_{i1}, \ldots, d_{in}, x_{in}, \beta_{in}).
\]

The coordinate feature \( (u_i, v_i, \theta_i) \) nearest structure feature \( (d, x, \beta) \) are combined together to characterize minutia \( m_i \), where \( 1 \leq u_i \leq W, 1 \leq v_i \leq H, \) and \( 0 \leq \theta_i \leq 359 \). \( u_i, v_i \), and \( \theta_i \) are represented as integers. \( m_i \) is called the reference minutia of structure \( NS_i \).

Because coordinates are used as part of the feature vectors of minutiae, an extra step is needed for the verification stage to align the template fingerprint images and the query fingerprint images. In this paper, we use the core point along with its direction to perform accurate alignment (Liu et al., 2010).

3. Proposed key binding system

The proposed key binding system is driven by the error pattern of minutia’s n-nearest structure. The errors on minutia structure vector also can be categorized into two types. The first is the background of random error. Many factors can lead to this errors, such as inaccurate alignment, sensor background noise, humidity of fin-

The rest of the paper is organized as follows: Section 2 describes the feature extraction and representation. The proposed key binding system is presented in Section 3. The experimental results are showed in Section 4. Section 5 analyzes the security of the proposed system, and conclusions are drawn in Section 6.

2. Minutiae extraction, representation and alignment

Given a fingerprint image \( I \) of size \( W \times H \), the short time Fourier transform (STFT) based enhancement algorithm proposed in (Chikkurur et al., 2007) is adopted to obtain an enhanced binary fingerprint image. The minutiae set \( M = \{m_i\} \) is extracted by the method described in (Shi and Govindaraju, 2006), where \( N \) is the number of minutiae in \( I \). For a minutia \( m_i \) in \( M \), nearest minutiae \( \{m_1, m_2, \ldots, m_n\} \) are found, where \( m_{ik} \in M, k = 1, \ldots, n \). Fig. 1 shows an example of the nearest minutiae topological structure of \( m_i \). Feature vector \( (d_{ik}, x_{ik}, \beta_{ik}) \) is used to describe the relationship between \( m_i \) and \( m_{ik} \), where \( d_{ik} = (0 \leq d_{ik} \leq d_{max}) \) denotes the distance between \( m_i \) and \( m_{ik} \), \( x_{ik} = (0 \leq x_{ik} \leq 359) \) denotes the minimal angle by clockwise rotating the radial line \( m_i m_{ik} \) to the orientation of minutia \( m_{ik} \), and \( \beta_{ik} = (0 \leq \beta_{ik} \leq 359) \) denotes the minimal angle by counterclockwise rotating radial line \( m_i m_{ik} \) to the orientation of minutia \( m_{ik} \). The angular values, \( x_{ik} \) and \( \beta_{ik} \), are measured in degrees and rounded to the nearest integers. The radius \( d_{ik} \) are rounded to the nearest integers too. \( d_{max} \) is the maximum radius value and can be determined by experiments. \( (d_{ik}, x_{ik}, \beta_{ik}) \) is invariant under translation and rotation of the fingerprint image. The nearest topological structure of \( m_i \) is defined as

\[
NS_i = (u_i, v_i, \theta_i, d_{i1}, x_{i1}, \beta_{i1}, \ldots, d_{in}, x_{in}, \beta_{in}).
\]

The coordinate feature \( (u_i, v_i, \theta_i) \) nearest structure feature \( (d, x, \beta) \) are combined together to characterize minutia \( m_i \), where \( 1 \leq u_i \leq W, 1 \leq v_i \leq H, \) and \( 0 \leq \theta_i \leq 359 \). \( u_i, v_i \), and \( \theta_i \) are represented as integers. \( m_i \) is called the reference minutia of structure \( NS_i \).

Because coordinates are used as part of the feature vectors of minutiae, an extra step is needed for the verification stage to align the template fingerprint images and the query fingerprint images. In this paper, we use the core point along with its direction to perform accurate alignment (Liu et al., 2010).

3. Proposed key binding system

The proposed key binding system is driven by the error pattern of minutia’s n-nearest structure. The errors on minutia structure vector also can be categorized into two types. The first is the background of random error. Many factors can lead to this errors, such as inaccurate alignment, sensor background noise, humidity of fin-
ger tips. The other is burst error due to missing or spurious minutiae, which are caused by partial observation, scars, etc. In our system, the first type of errors are handled by wrap-around secure sketch (Golić and Baltatu, 2007, 2008), and the second type of errors are handled by PinSketch (Dodis et al., 2006b). Their concatenation is a special case of soft two-level construction proposed in (Golić and Baltatu, 2008).

3.1. Soft two-level construction by wrap-around and pinsketch

In (Golić and Baltatu, 2007, 2008), the authors proposing an improved code-offset secure sketch construction for Euclidean metric space, named as wrap-around subtraction/addition. Suppose the biometric feature length is \( h \), we can obtain the following code-word space by choosing two vectors \( K = (k_1, k_2, \ldots, k_t) \) and \( q = (q_1, q_2, \ldots, q_t) \)

\[
C = \{(s_1q_1 + q_1/2, \ldots, s_tq_t + q_t/2)|s_i \in Z, 0 \leq s_i \leq k_i - 1, 1 \leq i \leq h\}.
\]  

(2)

The parameter \( K \) is an integer vector to decide the size of codeword space and \( q \) is the quantization step to set the error tolerance ability. Given a real valued biometric feature vector \( x = (x_1, x_2, \ldots, x_t) \), the wrap-around sketching procedure can bind a random codeword \( z \in C \) and produce a sketch vector \( \omega = (\omega_1, \omega_2, \ldots, \omega_h) \). The code-word \( z \) is used as a secret and sketch \( \omega \) is published. When a query biometric feature vector \( x' = (x'_1, x'_2, \ldots, x'_t) \) and the sketch \( \omega \) are given, the \( i \)-th component \( z_i \) of \( z \) can be recovered if \(-q_i/2 < x'_i - x_i < q_i/2 \), where \( [x'_i] \) equal to the unique real number in \([0,k_i] \) such that \( k_i \) divides \( x_i - [x'_i] \), which can be called the residual of \( x'_i \) modulo \([0,k_i] \). When \( x'_i < 0, k_i \), the above inequation can be simplified as \(-q_i/2 < x'_i - x_i < q_i/2 \). So, the wrap-around construction can handle the random errors falling into the range bound by quantization steps. For more details on wrap-around, we referred to Golić and Baltatu (2007), Golić and Baltatu (2008).

Wrap-around only is usually not enough to tolerate all the errors on \( x \). Supposing the probability of correctable random errors that happens on each component of \( x \) is \( p_r \), the probability to tolerate all the errors is \( p_r^t \). When \( h \) is very large, \( p_r^t \) will be very small. So, a second layer of error tolerance scheme is needed, such as Reed–Solomon code. In this paper, we use PinSketch (Dodis et al., 2008) to accomplish this task, which handles set difference more conveniently.

PinSketch is suitable for set symmetric difference metric. Given the input set \( x = (x_1, x_2, \ldots, x_t) \), where \( x_i \in \mathbb{F} = GF(2^m) \) and \( GF \) represents Galois field, the sketch data \( \omega = (\omega_1, \omega_2, \ldots, \omega_h) \) is obtained by applying \( \omega_i = \sum_{j=1}^{t} a^{j-1}. \omega_j \). \( \omega \) does not threat the security of \( x \), and \( t \) is the error tolerance ability. Given a corrupted set \( x' \) that the size of symmetric difference with \( x \) is less than or equal to \( t \), \( x \) can be recovered by finding a difference set \( D \) and output the symmetric difference of \( D \) and \( x' \) as \( x \). This is the recovery procedure of PinSketch. When \( x \) and \( x' \) have the same size, \( t \) must be set to even number. More details about PinSketch refer to Dodis et al. (2008).

3.2. Enrollment

At enrollment stage, each nearest structure is processed by soft two-level secure sketch and a binary vector is obtained. All the binary vectors are used to construct Shamir’s secret sharing system.

Given a nearest topological structure vector \( NS = (u, v, t, \theta, D_1, x_1, x_1, \ldots, D_m, x_m, \theta) \) of minutia \( m = (u, v, t, \theta, D_1, x_1, x_1, \ldots, D_m, x_m, \theta) \), where \( h = 3(n + 1) \), soft two-level construction operates as follows:

1. **Initialization**: Choosing parameters \( K \) and \( q \) for \( NS \) to obtain a codeword space by (2). From the codeword space, a code-word \( z \) is randomly selected. The quantization steps \( q_i \) is determined by the corresponding standard deviation of errors and the choice of \( k_i \) satisfy \( x_i \in [0,k_i] \) for ease of security analysis. In our implementation, \( k_i = 64 \) for \( i = 1, 2, \ldots, h \). In the rest of paper, we use \( K \) to represent all of them.

2. **Wrap-around construction for NS**: Taking \( NS \) as input of the sketching procedure of wrap-around. This sketching procedure outputs sketch \( \omega \) and binds codeword \( z \). The vector length of sketch \( \omega \) and codeword \( z \) is \( 3(n + 1) \). \( \omega \) is published, and \( z \) will be further processed by Shamir’s secret sharing scheme.

3. **PinSketch construction for local feature**: The codeword \( z = (z_1, z_2, \ldots, z_{3n+3}) \) uniquely determines an integer vector \( s = (s_1, s_2, \ldots, s_{3n+3}) \) by Eq. (2). Convert \( s \) into another form of set representation by \( sc = \{s_1|s_2|s_3|s_4|s_5|s_6|s_7|s_8|s_9|s_{10}|s_{11}|s_{12}|s_{13}|s_{14}|s_{15}|s_{16}|s_{17}|s_{18}|s_{19}|s_{20}|s_{21}|s_{22}|s_{23}|s_{24}|s_{25}|s_{26}|s_{27}|s_{28}|s_{29}|s_{30}|s_{31}|s_{32}|s_{33}\} \), where \( s_1|s_2|s_3 \) sequentially concatenating the binary representation of \( s_1, s_2 \) and \( s_3 \). The bit length of elements in \( sc \) is \( 3 \times \log_2 K \), because \( 0 < s_i < K \). Set \( sc \) is fed to sketching procedure of PinSketch. The parameter \( t \) for error toleration is set to 4, which means that among the \( n + 1 \) elements in \( sc \), two burst errors are tolerable. The resulting sketch by PinSketch is represented by \( y \). All the elements in \( sc \) are then concatenated together in ascending order to get a longer bit vector \( SC \), which is \( L = 3(n + 1) \log_2 K \) bits in length. Compute the hash value of \( SC \) by \( H = Hash(SC) \), where \( Hash() \) is any cryptographic hash function such as SHA256, MD6.

The above steps are performed on each minutia. Suppose the input template minutia number is \( N \). \( (SC)_i^{n+1} \) denotes all the binary vectors obtained by step 3 corresponding to each of minutiae. Given a key provided by the user, and if we expect \( num \) of \( SC_i \) is enough to recover the key, then the key can be bound by \((SC)_i^{n+1}\) as follows:

1. Divide key into \( num + 1 \) segments, such that \( key = key_0||key_1|| \ldots ||key_{num} \), with each element \( L \) bits in length. So the length of key should be \((num + 1) \times L\).

2. Construct a polynomial \( p(x) \) by taking \( key_0, key_1, \ldots, key_{num} \) as coefficients

\[
p(x) = key_0 + key_1x + \ldots + key_{num}x^{num}.
\]  

(3)

3. Compute \( A_i = p(SC_i) \) for each \( i = 1, 2, \ldots, N \), store \( A_i \) and discard \( SC_i \). The computation is operated in Galois field \( GF(2^L) \).

4. Randomly choose an element \( SC_0 \) in \( GF(2^L) \), and compute \( A_0 = p(SC_0) \). \((A_0, SC_0)\) is then published.

Here we summarize the sketches produced at the enrollment stage which are needed to store explicitly. There are three sketches in total. The first one is the wrap-around based sketch data \( \omega \). This is the first layer of sketch to tolerate the background random errors on each element of \( NS \). The second one is the PinSketch data \( y \), which is used to tolerate the burst errors happening among the \( n + 1 \) minutiae (including \( n \) nearest minutiae and one reference minutia). The third part of sketch is \((A_i)^{n+1}_i\) and \((A_0, SC_0)\), used for polynomial reconstruction. Apart from these sketch data, a hash value is computed for each minutia structure and need to be stored for verification. In Section 5, we will discuss the information leakage issues of these sketches and hash values.

3.3. Verification

Given the warp-around sketch data \( \omega \), PinSketch data \( y \), hash values of a template minutia structure and the query nearest minutia structure feature \( NS \), the verification stage outputs the bound key or a failure signal by exhaustive searching enough valid
codewords. Because the elements in sc are randomly distributed except for the first one which corresponds to the reference minutiae, in order to correct the burst errors in sc, we need to determine the corresponding relationship between two sets of nearest minutiae. So, our soft two-level is a little different from that in (Golić and Baltaci, 2008). The details are described as follows:

1. Initialize: Let Π be a set with each element a permutation of (1, 2, 3, ..., n). The size of Π is n! Set iterator k = 0.
2. Rearrange elements order in NS': Let k − k + 1. If k > n, structure recovery failure, proceed to recover this structure by next query structure feature vector. Take the kth element πk ∈ Π as a corresponding relationship, rearrange the elements order of NS' by applying

\[ NS' = (u, v, \theta, d_{\pi_1(1), \alpha_{\pi_1(1)}}, \beta_{\pi_1(1)}, d_{\pi_2(2), \alpha_{\pi_2(2)}}, \beta_{\pi_2(2)}, \ldots, d_{\pi_n(n), \alpha_{\pi_n(n)}}, \beta_{\pi_n(n)}) \]  

(4)

3. Wrap-around recovery: Taking NS' and wrap-around sketch vector ω as input, wrap-around recovery procedure outputs codeword z. z determines integer vector \( s = (s_1, \ldots, s_{3n-1}) \) by Eq. (2).
4. PinSketch recovery: \( s \) is grouped and concatenated by applying \( sc = \{s_1, s_2, s_3, s_4, \ldots, s_5, s_6, s_7, \ldots, s_{20}, s_{21}, s_{22}, s_{23}, \ldots, s_{3n-1}\} \). Taking \( sc = \{sc_1, sc_2, \ldots, sc_{n-1}\} \) and \( \gamma \) as inputs, PinSketch recovery procedure outputs a recovered set \( sc' = \{sc'_1, sc'_2, \ldots, sc'_{n-1}\} \). Sort the elements in \( sc' \) in ascending order and sequentially concatenate together obtaining a longer vector \( SC \). Compute the hash value \( H = Hash(SC) \) and compare it with \( H = Hash(SC) \). If \( H = H \), the codeword \( z \) of this template minutia structure is recovered successfully, otherwise goto step (2).

For the \( i \)th sketch \( \gamma_i \) generated during enrollment, we find the first query structure that successfully recovers \( SC \) and add \((\gamma_i, SC)\) into unlock set C. The stored \((\gamma_i, SC)\) is also added into C. When num of unduplicated \( SC \) in C are recovered, the secret can be obtained by the following interpolation equation

\[ p(x) = \sum_{i=0}^{num} A_i \prod_{j=0}^{num} \frac{x - SC_j}{SC_i - SC_j}. \]  

(5)

The secret key is obtained by sequentially concatenating the coefficients \( c_{nx+1}, \ldots, c_{num} \) of \( p(x) \).

4. Experiments

The proposed system is evaluated on FVC2002 DB1 and DB2 (Maio et al., 2002). This database contains 100 fingers with each having 8 impressions. For each finger, any two impressions are used to simulate genuine attempts once, which yields 100 × \( \binom{8}{2} \) = 2800 genuine attempts. The imposter attempts are simulated between any two fingers, impressions 1 are used, which yields \( \binom{100}{2} \) = 4950 imposter attempts. Three indexes are used for evaluation: (1) genuine accept rate (GAR); (2) false accept rate (FAR); and (3) zero false accept rate (ZeroFAR). GAR is defined as the percentage of accepted number of attempts among all simulated genuine attempts. FAR is defined as the percentage of false accepted number of imposter attempts. ZeroFAR is defined as the largest GAR at which no false accepts occur.

4.1. Quantization step, q

The quantization step vector \( q \) has key effects on the system's performance and security. In our experiments, all pairings of minutiae between impressions 1 and 2 are selected and aligned for both DB1 and DB2. From these samples of aligned paring minutiae, a large number of samples of aligned paring structure features can also be obtained. These samples of nearest structure features are used to estimate the error distributions of \( u, v, \theta, d, \alpha, \beta \). Each quantization step, \( q_i \), is set to the value such that \( p \) component of the corresponding feature errors can be corrected (e.g., in the range \([0, \frac{1}{2}]\)). In our experiments, all the quantization steps for \( u, v, \theta \) are set to values based on the same \( p_m \), and all the quantization steps for \( d, \alpha, \beta \) are set to values based on the same \( p_v \).

It is worth noting that because of angle discontinuity (360° ↔ 0°), some type of errors about \( \theta \), \( \alpha \) and \( \beta \) can not be handled by wrap-around. Fortunately, the percentages of the number of angle differences falling into the range \([180, 360)\) or \([-360, -180)\) are 0.45%, 0.93% and 0.94% for \( \theta \), \( \alpha \), \( \beta \), respectively on DB2.

Fig. 2 shows the ZeroFAR of the proposed system on DB1 and DB2 under different configurations of \( p_m \) and \( p_v \). The corresponding quantization steps are shown in Table 1. We get a conclusion from Fig. 2 that \( p_m \) has larger effects on ZeroFAR than \( p_v \). As \( p_m \) increasing, the ZeroFAR also increases. While as \( p_m \) increasing, the ZeroFAR only changes slightly. This phenomenon

![Fig. 2. ZeroFAR performances on (a) FVC2002 DB1 and (b) FVC2002 DB2 of the proposed system under different configurations of \( \rho_m \) and \( \rho_v \). \( \rho_m \) takes values from 0.6 to 0.88 with step 0.04 and \( \rho_v \) takes values 0.6, 0.68, 0.76, 0.84, and 0.88. The corresponding quantization steps are shown in Table 1. \( n \) and \( t \) are 5 and 4, respectively.](image-url)
can be explained as that $\rho_{ns}$ influences more feature elements in NS than $\rho_{m}$ and the correctness property of PinSketch can account for alignment errors to some extent. Although large $\rho_{ns}$ can improve the ZeroFAR performance, large $\rho_{ns}$ also decreases the security level as analyzed in the next section.

4.2. Required number of nearest structure, $num$

The required number of nearest structure $num$ for reconstruction of polynomial $p(x)$ affects system’s GAR and FAR drastically as shown in Fig. 3. The system GAR decreases quickly as $num$ increasing and the system FAR keeps at a very low level under different $\rho_{ns}$ and $num$ values. In most cases, one structure is enough to distinguish all the imposter attempts (FAR = 0), which means that $n$-nearest structure feature is discriminating enough. Large $num$ increases the searching time. By considering performances and computation time, $num$ = 1 or 2 are both good choices, which can achieve a very low FAR.

4.3. Number of nearest minutiae, $n$

Fig. 4 shows the GAR and FAR of the proposed system when $n$ = 4, 5, 6 on FVC2002 DB1 and DB2. The GAR curves show that when $n$ = 4, the system has a high recognition rate, but the FAR is higher than $n$ = 5, 6. When $n$ = 5 or 6, the FARs all equal to 0, and $n$ = 5 have a higher GAR than $n$ = 6. Table 2 shows that when $n$ = 5, the system’s ZeroFAR is higher than $n$ = 4 or 6 on both DB1 and DB2. When $n$ = 4, the required number of nearest structure $num$ equal 3, higher than that when $n$ = 5, 6. The results suggest that $n$ = 5 is a good choice.

4.4. Alignment issues

In our system, accurate alignment can improve the system performance. However, because of the correctness of PinSketch, the system can still work when inaccurate alignment occurs. Among the $n + 1$ minutiae in a nearest minutia structure, only reference minutia need be aligned during verification. If $n$ nearest minutiae are all close enough, PinSketch can recover the correct codeword no matter the reference minutia is successfully aligned or not.

We conduct two experiments to demonstrate that our system can work without alignment procedure with a slight cost of recognition accuracy decreasing. Table 3 shows the ZeroFAR comparison between the systems working with and without alignment. When system works without alignment, the ZeroFAR can achieve 77.21% on DB1, and on DB2, the ZeroFAR can even achieve 91.1%, which is very close to the system performance working with alignment. The GAR curves are shown in Fig. 5. On DB1, the difference of GAR be-

<table>
<thead>
<tr>
<th>$\rho_{ns}$ and $\rho_{m}$</th>
<th>0.6</th>
<th>0.64</th>
<th>0.68</th>
<th>0.72</th>
<th>0.76</th>
<th>0.80</th>
<th>0.84</th>
<th>0.88</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB1</td>
<td>(40.10,10)</td>
<td>(10,10,12)</td>
<td>(12,14,10)</td>
<td>(14,16,10)</td>
<td>(16,18,10)</td>
<td>(20,20,12)</td>
<td>(26,26,22)</td>
<td>(28,28,24)</td>
</tr>
<tr>
<td>DB2</td>
<td>(40.10,10)</td>
<td>(10,10,12)</td>
<td>(12,14,10)</td>
<td>(14,16,10)</td>
<td>(16,18,10)</td>
<td>(20,20,12)</td>
<td>(26,26,22)</td>
<td>(28,28,24)</td>
</tr>
</tbody>
</table>

Fig. 3. GAR and FAR performances on DB1 (first row) and DB2 (second row). The $\rho_{m}$ is fixed to 0.84 and 0.8 for DB1 and DB2, respectively and $n$ = 5.
between systems working with and without alignment is larger than
that on DB2. This phenomenon can be explained by the image size
and quality of DB1 and DB2. The image size and quality of DB2 are
both superior to that in DB1.

### 4.5. Comparison with other key binding systems

Our algorithm is compared with fuzzy vault based key binding
systems (Nandakumar et al., 2007; Nagar et al., 2010) and spectral
minutiae algorithm (Xu et al., 2008) on FVC2002 DB2 (see Table 4).
The matching of spectral minutia in (Xu et al., 2008) is not per-
formed under secure sketch, we still include it for comparing, be-
cause the fixed-length feature representation of minutia set is
convenient to existing secure sketch construction. In (Nandakumar
et al., 2007; Nagar et al., 2010), the number of genuine attempts are
100, only impressions 1 and 2 are used, and the number of impos-
ter attempts are 9900, any two different fingers are taken two
imposter attempts. In (Xu et al., 2008), impression 1, 2, 7 and 8
are used for evaluation. The number of genuine and imposter at-
ttempts is 600 and 4950, respectively, alinement needed. In our sys-
tem, taking two imposter attempts between two different fingers
reduces half duplicates, so only 4950 imposter attempts are simu-
lated. The genuine attempts of our system have been simulated on
three data sets. The first set only involved impressions 1 and 2, the
second set involved impressions 1, 2, 7 and 8, and the third one
used all the impressions in DB2. For each data set, we conduct
two experiments: the system working with and without align-
ment, respectively.

From Table 4, we see that our system, with or without align-
ment, has higher ZeroFAR performances than those in (Nandaku-
mare et al., 2007; Nagar et al., 2010) when
evaluated on the same data set. Our system can also work without
alignment, which makes our system more practical than Nandaku-
mare's and nagar's, which heavily depend on alignment accuracy.
Spectral minutia method for fixed-length feature vector extraction
is convenient to the existing template protection schemes, but it
suffers from global non-linear deformation and partial observation.
Our method uses local structure representation as the feature of
individual minutia, which effectively accounts for global non-lin-
ear deformation. Security will be analyzed in the next section.

### 4.6. Computation time

We implement the system using C++ programming language on
Intel (R) Core (TM) 2 × 1.86 GHz CPU with RAM 2 GB. Under
parameter setting $n = 5$, $t = 4$ and $num = 1$, the genuine matching
time is about 3.2 s and the imposter matching time is about 23 s,
and when \( n = 4, t = 4 \) and \( \text{num} = 1 \), the genuine matching time is about 0.433 s and the imposter matching time is about 5.2 s. The imposter matching time is far longer than the genuine matching time, this is because that imposter matching need to search almost all the nearest structures. Fortunately, in the genuine matching (the common case), it is usually much faster to search one paring nearest structure than imposter matching.

The majority computation time of our system is spent on searching the right codeword by hash checking. The computation complexity is \( O(N^2 \times N^2 \times n!) \). Supposing the numbers of query minutiae and template minutiae are both 40, in the worst case, when \( n = 4 \), the computational complexity is 38,400 and when \( n = 5 \), the computational complexity is 192,000. In a fingerprint fuzzy vault system (Nandakumar et al., 2007), the number of genuine points in the vault is 18–24, and the degree of encoding polynomial is 7–10, than in the worst case, their system complexity is 43,758–2,496,144. In a fuzzy vault system, in order to obtain a high system performance (GAR), more genuine points should be kept in the vault, while this will further increase the system complexity. From this point of view, our system is more practical than fingerprint fuzzy vault system. By storing some auxiliary information that are independent to nearest structure feature along with sketches, we could speed up the matching time by reducing the searching space.

5. Security analysis

5.1. PinSketch security

We first analyze the security from the output of PinSketch of each nearest structure, how much information of the codewords contain. Based on a conclusion in (Dodis et al., 2004), for a variable \( A \) of \( \lambda \) bits, it can carry at most \( \lambda \) bits of information if and only if \( A \) is randomly and uniformly distributed. Then for any variable \( B \), the mutual information of \( A \) and \( B \) is at most \( \lambda \) bits. That is \( I(A,B) \leq H(A) = \lambda \). In our method, the stored sketch data (denoted by variable \( A \)) of PinSketch outputs is \( 18 \times t \) bits, and the input data (denoted by variable \( B \)) is \( 18(n + 1) \) bits. So, given sketch data \( A \), the residual entropy of \( B \) is \( H(B|A) = H(B) - I(B,A) \geq 18(n + 1) - 18t \) bits. For \( n = 5 \) and \( t = 4 \), the security level is 36 bits. It should be noted that this security level estimated here represents the difficulty for an attacker to crack the codeword from the sketch data produced by PinSketch procedure. From the codeword, the attackers get nothing about biometric features. However, once the biometric data known to attackers, the codeword is determined. So, to estimate the exact entropy of codeword, we still need to know the entropy of biometric features. In the next section, we estimate the entropy of the nearest structure feature vector from the statistical point of view.

5.2. Hash security

The storage of a hash value for each nearest structure feature \( NS \) is for checking the output value of PinSketch during verification. This checking and retrying policy also gives attackers a chance of brute force attack on nearest structures one by one until \( \text{num} \) structures are guessed. The best strategy for an attacker to guess the nearest structure is based on the feature distribution model. For a given list of feature vector, we use the Parzen window method (Duda et al., 2001) to estimate the probability density of any two elements of the nearest feature vector. The mutual information of two variables is an indicator of information leakage from
each other. This property can also be used as an indicator of inde-
pendency. Given a probability density function $f(x, y)$ estimated by
Parzen window, the mutual information of $x$ and $y$ can be com-
puted by

$$I(x, y) = H(x) + H(y) - H(x, y) \tag{6}$$

where

$$\begin{align*}
H(x) &= -\sum_x f_x(x) \log f_x(x), \\
H(y) &= -\sum_y f_y(y) \log f_y(y), \\
H(x, y) &= -\sum_{x, y} f(x, y) \log f(x, y)
\end{align*}$$

and $f_x(y)$ and $f_y(x)$ are marginal distribution density function of
$f(x, y)$.

Based on the mutual information, we can better understand the
distribution characteristic of our used feature. Table 5 reports the
mutual information among variables $u$, $v$, $\theta$, $d_1$, $\alpha$, and $\beta_1$, and Fig. 6
shows their joint probability density functions, which are esti-
mated from FVC2002 DB2 impression 1. The subscript denotes
the index number of the neighboring minutiae after being ordered
in ascending way by their distances to the reference minutia. We
see that between $u$ and $\theta$, $v$ and $\theta$, $\alpha$, and $\beta_1$, there is relatively large
mutual information. Other variable pairs have very low mutual
information. The dependencies between $u$ and $\theta$, $v$ and $\theta$
are caused by the global orientation distribution characteristic. For
example, in the top and bottom fingerprint image, orientations
are near to horizontal, in the left, orientations are mostly near to
$-45^\circ$ and in the right, orientations are mostly near to $45^\circ$. The
same type of fingerprint (e.g., arch, whorl fingerprint) has similar
orientation map. The dependency between $\alpha$ and $\beta$ can be ex-
plained by the fact that two neighboring minutiae are often very
similar in angle.

From Table 5, we get a conclusion that the reference minutia
feature values $(u, v, \theta)$ are independent to that of nearest feature
values $(d_1, \alpha, \beta_1)$ (the same to $(d_i, \alpha_i, \beta_i)$, $i = 2, \ldots, n$). In our method,
all the nearest minutia are represented as a polar coordinates with
respect to the reference minutiae. The local relative coordinates of
nearest minutiae are not dependent on the global coordinates of
reference minutiae. But among the nearest minutiae, some depen-
dencies still exist.

From Tables 6–9, we find out that between the feature values of
two nearest minutiae, the main dependency is between $d_1$ and $d_2$. 

<table>
<thead>
<tr>
<th>Table 5</th>
</tr>
</thead>
</table>
The mutual information between $u$, $v$, $\theta$, $d_1$, $\alpha$, and $\beta_1$. The bold values denote higher dependences.

<table>
<thead>
<tr>
<th></th>
<th>$u$</th>
<th>$v$</th>
<th>$\theta$</th>
<th>$d_1$</th>
<th>$\alpha$</th>
<th>$\beta_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u$</td>
<td>-</td>
<td>0.0198</td>
<td>0.1716</td>
<td>0.0207</td>
<td>0.0115</td>
<td>0.0098</td>
</tr>
<tr>
<td>$v$</td>
<td>0.0198</td>
<td>-</td>
<td>0.2624</td>
<td>0.0211</td>
<td>0.0188</td>
<td>0.0194</td>
</tr>
<tr>
<td>$\theta$</td>
<td>0.1716</td>
<td>0.2624</td>
<td>-</td>
<td>0.0087</td>
<td>0.0388</td>
<td>0.0268</td>
</tr>
<tr>
<td>$d_1$</td>
<td>0.0207</td>
<td>0.0211</td>
<td>0.0087</td>
<td>-</td>
<td>0.0025</td>
<td>0.0049</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.0115</td>
<td>0.0188</td>
<td>0.0388</td>
<td>0.0025</td>
<td>-</td>
<td>1.1097</td>
</tr>
<tr>
<td>$\beta_1$</td>
<td>0.0098</td>
<td>0.0194</td>
<td>0.0268</td>
<td>0.0049</td>
<td>1.1097</td>
<td>-</td>
</tr>
</tbody>
</table>

Fig. 6. The union probability density functions (PDF) estimated by Parzen window. (a) $u$ and $\theta$; (b) $v$ and $\theta$; (c) $\alpha$ and $\beta$; (d) $d_1$ and $d_2$. 

The attacker only has to guess the exact random variable \( (u, \bar{v}, \bar{t}) \) (or \( (d, \bar{x}, \bar{\beta}) \)) should fall in the cuboid \( (u \pm \frac{v}{2}, v \pm \frac{u}{2}, \bar{t} \pm \frac{\bar{v}}{2}) \) (or \( (d \pm \frac{\bar{v}}{2}, \bar{x} \pm \frac{u}{2}, \bar{\beta} \pm \frac{\bar{\bar{v}}}{2}) \)). The cuboid is called recoverable region of wrap-around. Given the entropy \( H_{u,v} \) of \( (x, \beta) \), the average probability of guessing a exact \( (x, \beta) \) pair is \( \frac{1}{2^n} \). We estimate the probability of guessing a point fall in recoverable region of wrap-around using \( \frac{q_u q_v}{2^n} \) (There are \( q_u \times q_v \) number of discrete \( (x, \beta) \) pairs in range \( (x \pm \frac{v}{2}, \beta \pm \frac{\bar{v}}{2}) \)). So the difficulty for an attacker to guess a recoverable point of \( (x, \beta) \) is

\[
H_{u,v} = \log \frac{q_u q_v}{2^n} = H_{x,\beta} - \log(q_u \times q_v).
\]

Similarly, we can compute the entropy of \((u, \bar{v}, \bar{t})\) by \( H_{u,v} = H_{u,v} - \log(q_u \times q_v) \), where \( H_{u,v} = H(u) + H(v) + H(\bar{t}) - H(u, \bar{v}, \bar{t}) \) (supposing \( u \) and \( v \) are independent to each other).

Table 9

The mutual information between \( x_i \) and \( \beta_j \) (\( i \neq j \)). The bold values denote higher dependences.

<table>
<thead>
<tr>
<th>( x_1 )</th>
<th>( x_2 )</th>
<th>( x_3 )</th>
<th>( x_4 )</th>
<th>( x_5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \beta_1 )</td>
<td>1.1097</td>
<td>0.0181</td>
<td>0.0128</td>
<td>0.0112</td>
</tr>
<tr>
<td>( \beta_2 )</td>
<td>0.0173</td>
<td>0.9124</td>
<td>0.0147</td>
<td>0.0151</td>
</tr>
<tr>
<td>( \beta_3 )</td>
<td>0.012</td>
<td>0.012</td>
<td>0.7696</td>
<td>0.0126</td>
</tr>
<tr>
<td>( \beta_4 )</td>
<td>0.0122</td>
<td>0.012</td>
<td>0.0106</td>
<td>0.6363</td>
</tr>
<tr>
<td>( \beta_5 )</td>
<td>0.0111</td>
<td>0.0133</td>
<td>0.0108</td>
<td>0.0121</td>
</tr>
</tbody>
</table>

(i \( \neq j \)). In the rest parts of security analysis, we take the following three types of dependencies into consideration:

1. The dependencies between \( u \) and \( \bar{v}, v \) and \( \bar{t} \).
2. The dependencies between \( x_i \) and \( \beta_j \) (\( i = 1, 2, \ldots, n \)).
3. The dependencies between \( d_i \) and \( d_j \) (\( i \neq j \), and \( i, j = 1, 2, \ldots, n \)).

The dependencies between \( d_i \) and \( d_j \) make it very difficult to accurately estimate the union probability density function of \( (d_1, d_2, \ldots, d_n) \). For the sake of easy security analysis, we consider a more conservative estimation of security. We take the feature values \( d_i \) as totally dependent to each other, that is to say that once \( d_i \) is known, \( d_j \) is completely determined. Then, we can omit the \( d_i \) and analyze the entropy of nearest minutiae one by one. This assumption will help to relief the security analysis and estimate the security level in a more worse condition.

Given the discrete union probability density functions \( f(x, \beta) \) estimated by Parzen window, we can compute the union entropy of \( (x, \beta) \) by \( H_{x,\beta} = -\sum_{x,\beta} f(x, \beta) \log f(x, \beta) \). The entropy measures how difficult for an attacker to guess the exact random variable of \( (x, \beta) \). But, we have to notice that the attacker does not have to guess the exact value of a NS feature vector, because the correctness of wrap-around and PinSketch gives him/her a powerful tool. The attacker only has to guess \( n + 1 - \frac{1}{n} \) closed minutiae of a structure and the rest \( \frac{1}{n} \) minutiae can be recovered by PinSketch. The closeness is measured by the quantization step, which means for a reference minutia \( (u, v, \bar{t}) \) (or a nearest minutia \( (d, x, \bar{\beta}) \)), a point

![Fig. 7. The system security level under different quantization steps. The numbers in brackets indicate \((q_u, q_v, q_{\beta})\).](image-url)
6. Conclusions and future work

We propose a key binding system using $n$-nearest minutiae structure. The soft two-level construction is used to tolerate noises in a minutia structure, and Shamir’s secret sharing scheme is adopted for key binding and recovering. The proposed system can work even with or without minutiae alignment process. Experiments conducted on FVC2002 DB1 and DB2 demonstrate the efficiency.

Our system can be improved in the following aspects: (1) speed up the computation time of key recovery, such as research on how to determine the order of nearest minutiae in a structure or transform the structure to a fixed length feature vector. In our system, the majority of matching time is spent on the searching of paring minutiae. By storing some information of minutiae, we could speed up the searching procedure. The stored information should be independent to that of nearest structure; (2) improve the security level against brute force attack of a structure. The lower bound security level is based on the distribution of structure feature. Longer and more random feature for a structure may strengthen structure security; and (3) seek a better trade-off between security and recognition accuracy. Different to a traditional fingerprint recognition system, a key binding system should consider both recognition accuracy and security issues.

Acknowledgement

This paper is supported by 863 program under Grant No. 2008AA012411, the National Natural Science Foundation of China under Grant Nos. 60902083, 60803151, 60875018, CAS Hundred Talents Program, Beijing Natural Science Fund under Grant No. 4091004.

References