Cancelable fingerprint templates using minutiae-based bit-strings

Chulhan Lee a,*, Jaihie Kim b

a Jongno Technology Appraisal Center, Kibo Technology Fund, #22F, Samil Building, 10-2, GwanCheol-Dong, Jongno-Goo, Seoul, Republic of Korea
b School of Electrical and Electronic Engineering, Yonsei University, Biometrics Engineering Research Center (BERC), Republic of Korea

ARTICLE INFO

Article history:
Received 31 July 2009
Received in revised form 22 November 2009
Accepted 14 December 2009

Keywords:
Cancelable biometrics
Fingerprint recognition
Security and privacy concerns

ABSTRACT

It has become critical to protect biometric templates in the current biometric community. One way for doing this is using a cancelable biometric method, which transforms original biometric templates in a non-invertible way and uses those transformed templates to verify a person’s identity. In this paper, we propose a new method to generate cancelable bit-strings (templates) from fingerprint minutiae. Our method is to provide a simple mean to generate cancelable templates without requiring pre-alignment of fingerprints. The main idea is to map the minutiae into a predefined 3 dimensional array which consist of small cells and find out which cells include minutiae. To do this, we choose one of minutiae as a reference minutia and other minutiae are translated and rotated in order to map the minutiae into the cells based on the position and orientation of the reference minutia. After mapping, we set the cells in the 3D array to 1 if they include more than one minutia otherwise the cells are set to 0. A 1D bit-string is generated by sequentially visiting the cells in the 3D array. The order of the 1D bit-string is permuted according to the type of reference minutiae and user’s PIN so that we can regenerate new templates when we need them. Finally, cancelable bit-strings are generated by changing the reference minutia into another minutia in turn. In the experiments, we evaluate our method using the FVC2004 database and show that the performance is better than that of a previous method.

© 2009 Elsevier Ltd. All rights reserved.

1. Introduction

When biometric templates are compromised, privacy violations may occur. Therefore, biometric template protection has become a critical issue in the current biometric community. Several researchers have shown that an unknown original biometric image can be reconstructed from a fingerprint or face template (Adler, 2003; Mohanty et al., 2006; Ross et al., 2005). For example, Hill (2001) proposed a way of determining fingerprint structures from the coordinates of minutiae templates (including singular points). This method used the location of singular points to estimate the orientation field of the fingerprint, based on the method proposed in Sherlock and Monro (1993). A line drawing algorithm was used to generate the ridge patterns that passed through the minutiae points.

Ross et al. (2007) showed that three levels of information about the original fingerprint could be obtained from minutiae templates: the orientation field, the class or type of information, and the friction ridge structure. The local ridge orientation was estimated using the minutiae triplets. This was then used to predict the class of the fingerprint. Finally, the ridge structure of the original fingerprint was generated using streamlines that were based on the estimated orientation field.

Cappelli et al. (2007) described the idea that fingerprint images could be reconstructed from ISO standard templates. The orientation field was estimated by minimizing a cost function that was based on singular points. The fingerprint patterns were reconstructed with the position of the minutiae and an orientation map was estimated by iterative Gabor filtering. In their experiments, nine different systems were tested and the average percentage of successful attacks was 81% at a high security level and 90% at a medium security level. Recently, Nanni and Lumini (2009) experimentally showed that minutiae based matcher could be faked using reconstructed minutiae but image based matcher could not be faked.

Furthermore, traditional methods for identifying persons, for example, ID and personal identification numbers (PINs), can be canceled and re-issued if the above privacy issues are compromised. But this is not possible with biometric data because biometric data do not vary much over time and are very rarely shared by two people. Therefore, when the same biometric data are used in multiple security applications, biometric data can be shared between commercial companies and law enforcement or government agencies. This may lead to the possibility of tracking personal biometric data stored in one security application by getting access to another security applications through cross-matching.
1.1. Previous ways of protecting biometric templates

In general biometric systems, templates are stored fairly insecurely in databases. To protect them better, many alternate solutions have been proposed by both biometric and cryptographic researchers. These solutions can be roughly divided into two categories: biometric cryptosystems and cancelable biometrics.

1.1.1. Biometric cryptosystems

Biometric cryptosystems (Uludag et al., 2004) combine cryptographic keys with biometric templates so that the keys cannot be revealed without successful biometric authentication. Soutar et al. (1998) proposed a key-binding algorithm based on fingerprints. This algorithm created a correlation filter that combined cryptographic keys with fingerprint images. To reduce variations of fingerprint features, they applied a majority coding scheme. However, they assumed that the input and template fingerprint images were completely aligned, which is not always feasible in practical systems.

Davida et al. (1999) proposed a scheme using an iris feature as a cryptographic key, which was able to bind with authorization information. To decrease variations, an error correction code was applied. However, since error correction bits were stored in the database, some leakage of the biometric information might be existed. Juels and Wattenberg (1999) proposed the fuzzy commitment scheme to tolerant more intra-class variation between enrolled and query biometries by binding a codeword of an error-correcting code (ECC) using biometric data.

Monrose et al. (1999) proposed a password-hardening method based on keystroke biometrics. This method extracted binary bits from typing patterns and combined them with passwords. However, this yielded a low level of security since low entropy features were added to the passwords. In Monrose et al. (2001a, b), an improved method (based on voice biometrics) was proposed.

Linnartz and Tuyls (2003) proposed the use of shielding functions to protect the biometric templates. These researchers used delta-contracting and epsilon-revealing functions to pre-process the biometric data acquired from an individual. These functions make it computationally prohibitive to estimate the original data.

Dodis et al. (2004) proposed fuzzy extractor to use biometric data to securely derive cryptographic keys. They introduced two primitives, a secure sketch which allowed the recovery of a shared secret given a close approximation, and a fuzzy extractor which extracted a uniformly distributed string from this shared secret in an error-tolerant manner. This method has been applied to fingerprints (Arakala et al., 2007; Chang and Roy, 2007) and 3D face (Zhou, 2007).

Hao et al. (2006) proposed a method that combined biometrics with cryptography. These researchers used an iris code and two kinds of error correcting codes, the Reed–Solomon code and the Hadamard code. The error correcting codes were used to encrypt/decrypt cryptographic keys and reduce the variations between the registered and the query iris codes. Kanade et al. (2009) adopted this method by using a shuffling scheme to improve the performance.

Draper et al. (2007) and Vetro et al. (2009) proposed a distributed sourcing code method and implemented template protection methods using iris and fingerprint. To overcome the variation (movement, deletions, and insertions) of fingerprint minutiae, they used a statistical model. Binary feature vectors are extracted from minutiae and secure template (syndrome) is created using the LDPC code. However, this method requires pre-alignment of fingerprint to avoid greater computational complexity and enhance performance.

One of the most popular approaches is fuzzy vault scheme proposed by Juels and Sundan (2002). This scheme encodes a secret key in the coefficients of a polynomial. The secret key is locked and unlocked by biometric features. Because this scheme is robust to intra-variations in the biometric data, it has been applied on fingerprint, Face (Feng and Yuen, 2006), and iris (Lee et al., 2008). Also, multibiometric fuzzy vaults based on fingerprint and iris (Nandakumar and Jain, 2008), fingerprint and voice (Camlikaya et al., 2008), and multiple fingerprints (Yanikoglu and Kholmatov, 2004) were proposed. Recently Kholmatov and Yanikoglu (2008) empirically assessed the vulnerability of the fuzzy vault scheme against correlation based attacks. They showed the genuine points can be identified using two different vaults obtained from the same biometric data.

Based on the fuzzy vault scheme, several methods using fingerprint minutiae have been proposed by Clancy et al. (2003) and Uludag et al. (2005). In these methods, the minutiae positions were used to encode and decode secret codes. However, this method inherently assumed that the fingerprints were aligned (Kotlarchyk et al., 2008). Several works have been proposed to overcome this issue. Yang and Verbauwhede (2005) proposed a way of determining a reliable reference point based on the similarity indices of minutiae pairs by using several enrolled fingerprints. Based on this reference point, each minutia was represented by a polar coordinate and then used as locking and unlocking sets in the fuzzy vault scheme. Uludag and Jain (2006) handled the alignment issue by using helper data, which was generated using ridge orientation curves and used to align input fingerprint images. Finally, Chung et al. (2005) proposed an automatic fingerprint alignment approach that used geometric hash tables. Also, Li et al. (2009) and Nagar et al. (2008) proposed fingerprint fuzzy vault using transformed minutiae and combining with local minutiae descriptors to enhance the security.

1.1.2. Cancelable biometrics

Cancelable biometrics (Ratha et al., 2001) uses transformed or intentionally-distorted biometric data instead of original biometric data for identification. Because the transformation is non-invertible, the original biometric templates cannot be recovered from the transformed templates. When a set of biometric templates is found to be compromised, it can be discarded and a new set of biometric templates can be regenerated.

Savvides et al. (2004) proposed cancelable biometrics for face recognition using minimum average correlation energy (MACE) filters and random kernels. In the training stage, training face images were convolved with a random kernel that was generated by a random number generator. A MACE filter was created using these convolved images. In the verification stage, a test face image was convolved with the same random kernel that was used in the training stage and the convolved test image was cross-correlated with the MACE filter. The resulting correlation outputs were examined for authentication and it was found that performance was not affected by the random kernel. Recently, Bodde et al. (2009) applied modified MACE filter on face recognition and Hirata and Takahashi (2009) proposed correlation-based matching algorithm using Number Theoretic Transform for cancelable biometrics.

Also, Ang et al. (2005) proposed a key-based transformation method for fingerprint minutiae. A core point of an input fingerprint image was detected and then a line through the core point was specified. The angle of the line depended on the key, where $0 \leq \text{key} \leq \Pi$. The transformed fingerprint templates were
generated by reflecting the minutiae under the line into those above the line. The new transformed fingerprint template was then generated by changing the key (angle). A disadvantage of this method is that it required core point detection as well as the alignment of the input fingerprint image into a canonical position. Also, since the minutiae above the line were not transformed, the transformed template still retained some information from the original fingerprint.

Next, Tulyakov et al. (2005) proposed a hash-based transformation method. For each minutia, the N nearest neighbor minutiae were found and M (M < N) hashed minutiae were generated using symmetric hash functions. The hashed minutiae were then stored in a database and compared to the query hashed minutiae. Unlike common hash functions, these hash functions showed good biometric properties. In the hash space, these researchers discovered the geometric relationship (translation and rotation) between the query and the enrolled fingerprint. However, they did not describe how the newly hashed minutiae could be generated when stored minutiae were compromised.

Teoh et al. (2006) proposed an authentication approach called Biohashing, which combined user-specific tokenized random vectors with biometric feature vectors to generate biometric codes. From the user token, an M × N (M > N) random matrix was created and orthonormalized using the Gram–Schmidt method. The inner product between the N-dimensional biometric feature vectors and a set of column vectors of the orthonormal matrix was calculated, and a biometric code was generated with a predefined threshold. The biometric code was canceled and re-issued by varying the tokenized random vectors. However, this method also required alignment and it could not be applied to unordered biometric features such as fingerprint minutiae. Related to this, Kong et al. (2006) experimentally showed that the performance of the Biohashing was degraded when token were duplicated. To overcome this problem, muti-Biohashing based on multiple hashing was proposed by Nanni and Lumini (PRL) (2008), Maio and Nanni (2005), and Nanni and Lumini (2008).

Ratha et al. (2007) described three transformation methods such as Cartesian, polar, and functional transformation. The Cartesian and polar transformation methods divided a fingerprint into sub-blocks and then scrambled those sub-blocks. In the functional transformation method, transformation was based on a Gaussian function. However, all three methods required alignment before transformation. To align the fingerprints, these methods used singular points. However, it was not always possible to detect the singular points.

Boult (2006) proposed a transformation method for face recognition with a robust distance measurement. In this method, a face feature was transformed via scaling and translation, and then it was separated into a fraction and an integer part, and the integer part was hidden using one-way transformation or encryption. This method was recently applied to fingerprints (Boult et al., 2007).

Jeong et al. (2006) proposed an approach for appearance-based face recognition. Two feature vectors were extracted with PCA (principal component analysis) and ICA (independent component analysis) architecture from a face image, and these vectors were normalized. The vectors were then scrambled and added to obtain a final transformed feature vector. If this was compromised, a newly transformed feature vector was generated by changing the previous scrambling rule.

Lee et al. (2007) proposed a cancelable fingerprint template using fingerprint minutiae. Translation and rotation invariant values were extracted using orientation information around each minutia. The obtained invariant value was input into two changing functions (which output translational and rotational movement) to transform each minutia. Final cancelable templates were generated by moving each minutia according to the calculated movements. When the cancelable templates were compromised, new templates were regenerated by replacing the changing functions.

Finally, Farooq et al. (2007) generated a binary-string from fingerprint minutiae. Based on minutiae triplets (triangle), translational and rotational invariants were extracted. These researchers used the length of three sides, the three angles between the sides and minutiae orientations and the height of the triangle as invariants. The invariants were quantized and hashed to generate a binary-string (2^{24} bit). The binary-string was permuted and encrypted to re-issue and enhance the security of the template.

1.2. Motivation and scope

One problem of previous efforts to protect fingerprint templates is that they required the alignment of fingerprint images to make protected templates. The alignment issue is different to the alignment issue for matching between registered and query fingerprint images in a conventional non-cancelable environment. A matching algorithm for non-cancelable fingerprints tries to align a query fingerprint with a registered fingerprint. However, when making protected fingerprint templates, registered fingerprints have to be transformed so that they do not provide any cue to a query fingerprint for alignment. Some previous methods (Boult et al., 2007; Lee et al., 2007; Farooq et al., 2007) have proposed the way to overcome the alignment issue. However, the method discussed in Boult et al. (2007) required an iterative searching approach to find parameters, which were used to scale and translate (transform) the minutiae. The method discussed in Lee et al. (2007) was affected by the quality of the fingerprints because the invariant values for moving minutia were extracted from the orientation information around each minutia. Therefore, performance degraded seriously when the quality of images was poor. The method discussed in Farooq et al. (2007) required additional computation costs to generate templates, because all possible triple minutiae had to be selected and length, angles and height had to be calculated. In this paper, we propose simple and effective method to generate cancelable templates without requiring for alignment.

2. Generation of bit-strings from minutiae

In this section, we describe how to generate bit-strings (cancelable templates) from minutiae. Fig. 1 shows the overall method for generating bit-strings. First, we define a 3 dimensional array which consists of cells (Section 2.1). Second we choose one of minutiae as a reference minutia and map other minutiae into the 3D array based on the position and orientation of the reference minutia (Section 2.2.1). By mapping the minutiae into the 3D array, we find out which cells in the 3D array include minutiae. Then, we set a cell in the 3D array to 1 if it contains more than one minutia, and otherwise, the cell is set to 0 (Section 2.2.2). Finally, a 1D bit-string (cancelable template) is generated by sequentially visiting the cells in the 3D array and the order of the array is permuted. Finally, cancelable bit-strings are generated by changing the reference minutia into another minutia in turn.

2.1. 3 dimensional array

The first step in our proposed method is to define a 3 dimensional array consisting of small cells. The width (W_X) and height (W_Y) of the array are two times of the size of an input
fingerprint image and the depth ($W_Z$) is $2\pi$. And, the width ($C_X$), height ($C_Y$), and depth ($C_Z$) of a cell are predetermined experimentally. The units of $C_X$ and $C_Y$ are pixels and the unit of $C_Z$ is radian. The total number of cells in the 3D array is $d = M/C^2 \times N/C^2 \times L$, where $M = [W_X/C_X], N = [W_Y/C_Y], L = [2\pi/C_Z]$, and $\lfloor \cdot \rfloor$ represents the floor function. Fig. 2 shows the 3D array schematically.

2.2. Mapping minutiae into 3 dimensional array and generation of bit-strings

2.2.1. Transformation of minutiae

Let $M_i = [x_i, y_i, \theta_i, t_i]$ denote the ith minutia. Here, $x_i$ and $y_i$ are pixels, and $\theta_i$ represents the position, the orientation, and the type of minutiae (bifurcation or endpoint), respectively. One of the minutiae is selected as the reference minutia $M_r = [x_r, y_r, \theta_r, t_r]$. Other minutiae are rotated and translated to align the orientation of the reference minutia into the $x$-axis of the 3D array and the position of the reference minutia at the center of the 3D array. Fig. 3 shows the transformation of minutiae. A transformed minutia $M'_i = [x'_i, y'_i, \theta'_i, t'_i]$ is obtained as follows:

$$
\begin{bmatrix}
    x'_i \\
    y'_i \\
    \theta'_i
\end{bmatrix} =
\begin{bmatrix}
    \cos \theta - \sin \theta & 0 & x_i - x_r \\
    \sin \theta & \cos \theta & y_i - y_r \\
    0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    W_X/2 \\
    W_Y/2 \\
    t_i
\end{bmatrix}
$$

(1)

This idea about using the coordinate of reference minutiae to make transformed minutiae is inspired from Benhammadi et al. (2005) and Wang et al. (2006).

2.2.2. Generation of a bit-string

After transforming the minutiae, we know which cell in the 3D array include the minutiae as follows:

$$
\begin{bmatrix}
    x_i^C \\
    y_i^C \\
    z_i^C
\end{bmatrix} =
\begin{bmatrix}
    \lfloor x'_i/C_X \rfloor \\
    \lfloor y'_i/C_Y \rfloor \\
    \lfloor z'_i/C_Z \rfloor
\end{bmatrix}
$$

(2)

where $\lfloor \cdot \rfloor$ represents the floor function. $x_i^C, y_i^C,$ and $z_i^C$ are the $x, y,$ and $z$ indices in the 3D array and $C_X, C_Y,$ and $C_Z$ are width, height, and depth of the cell.

A 1-D bit-string is generated by sequentially visiting the cells in the 3D array. If a cell in the 3D array included more than one transformed minutia, the cells are set to 1; otherwise they are set to 0. The number of bits for the 1-D bit-string is $d = M \times N \times L$. This is the same as the number of cells in the 3D array.

2.2.3. Permutation of the 1D bit-string

Mapping the minutiae into the 3D array based on the reference point ($M_r$), a 1D bit-string ($b_i$) is generated. Next, the order of the
A 1D bit-string is permuted based on the type of reference minutiae \((t_r)\) and user’s PIN. We define the two 2D bit-strings of length \(M \times N \times L\) as \(p^{\text{PIN}}_{\text{E}}\) and \(p^{\text{PIN}}_{\text{B}}\), which are generated by random number generator with a seed given by the user’s PIN. Using these matrices, the 1D bit-string is permuted as follows:

\[
B_r = \begin{cases} 
    p^{\text{PIN}}_{\text{E}}, & \text{if } t_r \text{ is endpoint} \\
    p^{\text{PIN}}_{\text{B}}, & \text{if } t_r \text{ is bifurcation}
\end{cases} 
\]  

where \(B_r\) represents a permuted 1D bit-string. The two different permutation matrices are created randomly and seeded by the user’s PIN from random number generator. Therefore, for users with the same user’s PIN, the same permutation matrices are created. The 1D bit-string is permuted because:

(i) The final cancelable templates are composed of the 1D bit-strings (Section 2.2.4). By replacing the permutation matrices or the user’s PIN, new cancelable templates are regenerated from the same fingerprint, when cancelable templates are compromised.

(ii) Because a 1D bit-string is created by sequentially visiting all cells of the 3D array, the quantized positions and orientations can be found from a 1D bit-strings. Using the type of reference minutiae to permute the 1D bit-strings, it makes more intractable to find out the quantized positions and orientations because the type is not stored. To give high security, we describe a way in Section 3.4.

2.2.4. Changing the reference minutiae

As was described in the previous section, we generate a 1D bit-string based on a reference minutia. If only one minutia is used as the reference minutia, the minutia should be detected from query fingerprints and stored in database with cancelable template. But it is practically impossible to detect the same reference minutia from query fingerprints. Therefore, by changing the reference minutia to another minutia in turn, we make a final template consisting of \(m\) 1D bit-strings (where \(m\) represents the total number of minutiae). Because of this scheme, our method does not require pre-alignment to create cancelable templates.

Because the minutiae are aligned into the 3D array based on the reference minutia and bit-strings are generated, we can make the same 1D bit-string when the reference point is the same without requiring for pre-alignment of fingerprints.

2.3. Similarity score between bit-strings

Ideally, two 1D bit-strings generated from the same reference minutiae are the same. However, we cannot know which 1D bit-strings are generated from the same reference point. Therefore, we compare all 1D bit-strings generated by changing reference minutiae and calculate maximum similarity score.

Fig. 4 shows an enrolled template \((B^E)\) and a query template \((B^Q)\), both of which consist of 1D bit-strings. In this figure, \(B^E_i\) \((i=1,\ldots,n; \ n \text{ is the number of the minutiae for the enrolled fingerprint})\) denotes the \(i\)th 1D bit-string of \(B^E\), and \(B^Q_j\) \((j=1,\ldots,m; \ m \text{ is the number of minutiae for query fingerprint})\) denotes the \(j\)th 1D bit-string of \(B^Q\). Each 1D bit-string has the same bits \(d=M \times N \times L\). The similarity score between the enrolled bit-strings \((B^E)\) and the query bit-strings \((B^Q)\) is calculated as follows:

\[
\begin{align}
(i) \text{ Comparing } B^Q_i \text{ to } B^E: \\
SA(i,j) &= \frac{(N^Q_i + N^Q_j \sum_{k=1}^{d} (B^Q_i \Theta B^E_{k,j}))}{(N^Q_i)^2 + (N^Q_j)^2} \\
N^Q_i &= \sum_{k=1}^{d} B^Q_{i,k}, & N^Q_j &= \sum_{k=1}^{d} B^E_{i,k} 
\end{align}
\]

where \(\Theta\) represents a bit-wise and operator, \(B^Q_{i,k}\) and \(B^E_{i,k}\) denote the \(k\)th bit in \(B^Q_i\) and \(B^E_i\), and \(N^Q_i\) and \(N^Q_j\) represent the number of 1’s bits of \(B^Q_i\) and \(B^E_i\). The \(SA(\in [0,1])\) in Eq. (4) means how many bits with a value of 1 are the same between enrolled and query bit-strings and is normalized because the number of minutiae extracted from fingerprint is different.

(ii) Finding a maximum SA value for the 1D query bit-string \((B^Q_i)\).

By comparing the \(j\)th 1D query bit-string \((B^Q_j)\) to all the 1D enrolled bit-strings, we find the most similar 1D enrolled
bit-string and calculate the maximum SA value (SAmax(j)):

\[
\text{Index}(j) = \arg\max_i \text{SA}(i, j)
\]

\[
SA_{\text{max}}(j) = \text{SA}(\text{Index}(j), j)
\]  

(iii) Calculating the final similarity score

The final similarity score between the enrolled and query templates is calculated as follows:

\[
\begin{align*}
\text{Score}(E, Q) &= \frac{\sum_{j=1}^{m} S(j)}{T} \\
S(j) &= \begin{cases} 
SA_{\text{max}}(j) & \text{if } SA_{\text{max}}(j) > Th \\
0 & \text{otherwise}
\end{cases}
\end{align*}
\]

where \(T\) represents the number of \(S(j)\) whose value is not zero and \(Th\) represents the threshold which is determined experimentally. The final score (\(\text{Score}(E, Q)\)) between the query and enrolled template has a value between 0 and 1.

3. Experiments and analysis

3.1. Experimental design

The proposed method is evaluated using the FVC2004 database (DB1, DB2, and DB3) (FVC2004). Each database consists of two sets, “Set A” and “Set B”. “Set B” includes 80 fingerprint images for 10 fingers. Each finger has 8 fingerprint images. This set is used for parameter tuning before testing. “Set A” includes 800 fingerprint images for 100 fingers and it is used to evaluate the fingerprint system. In our experiments, using “Set B”, we discover the cell size in the 3D array considering both performance and template size. Fig. 5 shows the EER when cell size is varied using FVC2004DB1. Table 1 shows the cell size that were used in these experiments.

3.2. Evaluation criteria

(i) Performance comparison: We compare the proposed method with Lee’s method (Lee et al., 2007). It is found that both Lee’s method and the proposed method generate cancelable fingerprint minutiae without requiring alignment. Two cases are considered: when each subject has a different PIN (the permutation matrices are different for each finger) and when all subjects has the same PIN. The first case verifies the performance of the proposed method and the latter case is used to simulate what would happen if a user’s PIN was stolen and duplicated.

(ii) Changeability: In cancelable biometric systems, when transformed templates are compromised, new transformed templates are generated. The new transformed templates are usually very different from the compromised templates. Changeability measures the degree of difference. To analyze changeability, we generated two bit-strings from the same fingerprint by replacing the permutation matrix and then compared them.

3.3. Experimental results

3.3.1. Performance comparison

Fig. 6 shows the ROC curves when each user have a different PIN and Fig. 7 shows the ROC curves when each user have the same PIN. In both cases, the proposed method shows better performance than Lee’s method. One of the reasons is that Lee’s method is much affected by the quality of the fingerprint because the orientation information around each minutia has to be used to transform the minutiae. And, unlike Lee’s method, the performance of the proposed method when using different PINs is much improved (the EER is close to 0). This is because we generate the bit-strings that have order properties, and the order of the bit-string is permuted differently for each user. Therefore, the dissimilarity between different users (imposters) increases.

Fig. 8 shows the genuine and imposter distributions for both cases. The genuine distribution is the same in both cases but the imposter distributions are different. The similarity between
imposters when users have different PINs is lower than that when they have the same PINs.

When the proposed cancelable method is applied to a real system, all users have different PINs (Ratha et al., 2007), so the performance of the proposed method is almost ideal. However, the user’s PINs can be misused consciously or unconsciously. In this case, the performance is not satisfied to high level secure system (EER is 10.3%, 9.5%, and 6.8% for FVCDB1, FVCDB2, and FVCDB3). The reasons for this decrease in performance include the quantization error in the 3D array and the distortion of the fingerprint images. These make different minutiae position in the 3D array and generated different bit-strings.

### 3.3.2. Changeability

In general, when a cancelable template is compromised, a new cancelable template is regenerated and replaced by a cancelable biometric system. This new template should not match the compromised template. Changeability measures this situation. In the proposed method, we generated a new template by replacing the permutation matrix or user’s PIN. To measure the degree of changeability, we generated 100,000 different templates from the same finger by replacing the permutation matrices, and then compared them. From this comparison, we calculated a distribution called a pseudo-imposter distribution. Fig. 9 shows the three distributions, imposters for both cases (the same and different PINs) and pseudo-imposter distribution. Table 2 shows the mean and standard deviations of the distributions. From Fig. 9 and Table 2, the pseudo-imposter distributions are similar to the impostor distribution for different PINs. This means that even if two templates were generated from the same fingerprint, the two templates rarely matched when the corresponding permutation matrices are different. Therefore, the possibility for matching the new cancelable templates with the compromised templates is very low.

### 3.3.3. Computational complexity and template size

In this section, we discuss the computational complexity and template size of the proposed method. Computational cost is measured in terms of a number of additions and multiplications. Assume the number of extracted minutiae to be m. From Section 2, in order to generate cancelable template, our method translates and rotates minutiae based on one of minutiae (called reference point) and then the reference minutia is changed into another minutia in turn. The number of additions and multiplications to translate and rotate a minutia is 3 and 4. Based on the minutia (called a reference minutia), other minutiae \((m−1)\) is translated and rotated. Therefore, \(3 \times (m−1)\) additions and \(4 \times (m−1)\) multiplications are required. A reference minutia is changed into
another minutia in turn and permutation is applied (permutation needs \(m^2\) multiplication). Therefore, the total number of additions and multiplications for generating cancelable bit-string is \(3 \times m \times (m-1)\) and \(4 \times m \times (m-1)+m^2\). When \(m = 40\), \(4680\) additions and \(7840\) multiplications are required.

Using Lee's method, in order to transform the minutiae, invariant value is extracted from each minutia. The invariant value is calculated using orientation information surrounding each minutia. When gradient method is used to calculate orientation, the \(20\times w^2+1\) addition and \(21\times w^2\) multiplication are required. Here, \(w\) is the block size \((w \times w)\) for calculating gradient. From each minutia, orientations of \(p\) locations are extracted using gradient method. The number of addition and multiplications is \(\frac{p}{2} \times (20 \times w^2+1)\) and \(\frac{p}{2} \times 21 \times w^2\). After calculating orientation of \(p\) locations surrounding each minutiae, invariant value is calculated by inner product and then finally minutia are translated and rotated (transformed). The total number of additions and multiplications is \(m \times (p \times 20 \times w^2+p-1+3)\) and \(m \times (p \times 21 \times w^2+p+4)\). For example, when \(m = 40\) and \(w = 8\), and \(p = 16\), the number of additions and

---

**Fig. 8.** Genuine and imposter distributions for the same and different PINs: (a) FVC2004DB1, (b) FVC2004DB2, and (c) FVC2004DB3.
The above discussion shows that our method needs $O(m^2)$ additions and multiplications and Lee’s method needs $O(mp^w)$ additions and multiplication.

In Section 2.2.2, we explain the number of bits for the 1-D bit-string generated from one reference point is $M/C + N/C + L$ bit (in case of our implementation, 2048 bit for FVC2004DB1, 795 bit for FVC2004DB2, 1440 bit for FVC2004DB3). When the number of minutiae is $m$, the size of cancelable bit-strings is $m \times M \times N \times L$ bit. For example, when $m$ is 40, about 80 Kbyte for FVC2004DB1, 32 Kbyte for FVC2004DB2, and 57 Kbyte for FVC2004DB3 are required in our case. When MD5 is used to hash the address where the bit is 1 in cancelable bit-string, each address is hashed into 128 bit. More explanation about applying hash function is described in Section 3.4. Because the number of address where bit is 1 is $m \times (m - 1)$, the template size is $128 \times m \times (m - 1)$ bit. When $m$ is 40, the template size is 195 Kbyte.

In case of Lee’s method, because transformed minutiae (cancelable template) are generated by moving the position and orientation of original minutiae, the size of transformed minutiae is the same to original minutiae. When ISO standard format is used to represent minutiae $m \times 6$ Byte is required to store $m$ minutiae (Cappelli et al., 2007). When $m$ is 40, the template size is 240 bytes. Consequently, our method has less computational complexity than existing method but requires larger memory to store cancelable template.

### 3.4. Enhancing the security of the proposed method

Although the permutation function is determined based on the type of the reference minutiae and the type is not stored in final cancelable template, when attacker knows all possible information (final bit-string, user’s PIN, random number generator and structure of 3D array), attacker can find out the quantized position and orientation of original minutiae because the permutation function is invertible. Even attacker knows only the final bit-string; he can also discover the quantized original minutiae using brute force attack. Even though attacker can recover not original minutiae but quantized minutiae, our method is no more secure.

To increase the security of the proposed method, we can apply existing cryptographic hashing techniques. The addresses where the bit is 1 in bit-string are hashed (Farooq et al., 2007). We do not hash bit-string (template) but each address where the bit is 1 in bit-string because:

1. Bit-string (template) can be varied because fingerprint features are unstable. Therefore, even if one bit in the bit-strings is flipped, the hashed-bit-string is dramatically changed.
2. Our method to calculate the similarity score between two bit-strings counts how many addresses where bits are 1 and the
same. Therefore, only the addresses are important information to verify two templates.

Although the existing cryptographic hashing techniques such as MD5 and SHA-1 are very useful and have been employed in a wide variety of security applications, it is difficult to apply biometrics because biometric data are inherently unstable. However, our method can use cryptographic hashing techniques by hashing the address where the bit is 1. By hashing the each address and comparing two hashed bit-strings in hashed domains, there is no practical way to recover original bit-strings and quantized position and orientation of original minutiae.

4. Conclusions and future works

In this paper, a new method of generating cancelable fingerprint templates from fingerprint minutiae was described. We proposed a low complexity method that did not require pre-alignment. The proposed method generated bit-strings (cancelable templates) by mapping the minutiae into a predefined 3D array using the coordinates of each minutua (the position and orientation of each reference minutua). Therefore, the alignment of an input fingerprint was not necessary. In our experimental analysis, the performance was ideal when each user had a different PIN and the two templates from the same fingerprint were not matched when the corresponding PINS were different. We also proposed how to apply the cryptographic hashing technique on our method to enhance the security. However, when PINS were duplicated (for users with the same PIN), the performance was not highly secure. To solve this problem, there are following further studies to improve the performance. First, we used quantized position and orientation of minutiae without considering the distortion of the fingerprint. One of the solutions is to determine the quantization level (cell size) adaptively to reduce the effect of the distortion. Second, Biohashing proposed by Teoh et al. (2006) has the same problem. Because our method can extract the same length feature vector from fingerprint template, we can use the scheme of Biohashing. The improved Biohasing (Nanni et al., 2008) is one of the solutions to enhance performance when PINS were duplicated.

Acknowledgements

This work was supported by Korea Science and Engineering Foundation (KOSEF) through the Biometrics Engineering Research Center at Yonsei University.

References
