Fingerprint Template Protection using Fuzzy Vault

Daesung Moon[1], Sungju Lee[2], Seunghwan Jung[2], Yongwha Chung[2], Kiyoung Moon[1]

[1]Biometrics Technology Research Team, ETRI, Daejeon, KOREA
{daesung, kymoon}@etri.re.kr
[2]Department of Computer and Information Science, Korea University, KOREA
{peacfeel, sksghksl, ychungy}@korea.ac.kr

Abstract. Biometric-based authentication can provide strong security guarantee about the identity of users. However, security of biometric data is particularly important as the compromise of the data will be permanent. To protect the biometric data, we need to store it in a non-invertible transformed version. Thus, even if the transformed version is compromised, the actual biometric data remains safe. In this paper, we propose an approach to protect fingerprint templates by using the idea of the *fuzzy vault*. Fuzzy vault is a recently developed cryptographic construct to secure critical data with the fingerprint data in a way that only the authorized user can access the secret by providing the valid fingerprint. We modify the fuzzy vault to protect fingerprint templates and to perform fingerprint verification with the protected template at the same time. This is challenging because the fingerprint verification is performed in the domain of the protected form. Based on the experimental results, we confirm that the proposed approach can perform the fingerprint verification with the protected template.

Keywords: Crypto-Biometric, Template Protection, Fuzzy Vault

1 Introduction

The increasing demand for more reliable and convenient security systems generates a renewed interest in human identification based on biometric identifiers such as fingerprints, iris, voice, signature and gait[1-2]. Since biometrics cannot be lost or forgotten like passwords, biometrics has the potential to offer higher security and more convenience for the users.

A common approach to biometric authentication is to capture the biometric measurements of users, transform the measurements into templates, and store the templates in a reference database during the *enrollment phase*. During the *verification phase*, a new measurement is captured and matched against the database information. Note that, biometric measurements are rarely identical and each measurement is environment- and device-dependent. Also, a template provides an approximate version of the biometric data, and verification succeeds if the second measurement is close to the stored template. And, in this paper, the fingerprint has been chosen as the biometrics for user authentication because it is more mature in terms of the algorithm availability and feasibility[1].

The fact that biometric templates are stored in a database introduces a number of security risks, and the following threats can be identified[1-6]:

1. *Impersonation*. An attacker steals templates from a database and constructs artificial biometrics that pass authentication.
2. *Irrevocability*. Once compromised, biometrics cannot be updated, reissued or destroyed.

For example, when an authentication system is used on a large-scale, the reference database has

to be made available to many different verifiers, who, in general, cannot be trusted. Especially, in a networked environment, attacks on the database pose a serious threat. It was shown explicitly by Matsumoto et al.[7] that using information stolen from a database, artificial biometrics could be constructed to impersonate people. Construction of artificial biometrics is possible even if only part of the template is available. Hill[8] showed that if only minutiae templates of a fingerprint are available, it is still possible to successfully construct artificial biometrics that pass authentication. The second threat was first addressed by Schneier[9]. Even without database compromise, biometric possession by government provides the potential for information misuse as the data belonging to the same user can be easily linked. In order to protect against the threats given above, we need to store a transformed version of the biometric data. The transformation is one-way and so knowledge of a transformed biometric does not leak information about the actual biometric data.

In this paper, we propose an approach to protect fingerprint templates by using the idea of the fuzzy vault[10]. *Fuzzy vault*, as explained in later sections, is a recently developed cryptographic construct, and has the characteristics that make it suitable for applications combining biometric authentication and cryptography: the advantages of cryptography(*e.g.*, proven security) and fingerprint-based authentication(*e.g.*, user convenience, non-repudiation) can be utilized in such systems.

We modify the fuzzy vault to protect fingerprint templates and to perform fingerprint verification with the protected template at the same time. This is challenging because the fingerprint verification is performed in the domain of the protected form. Based on the idea of the fuzzy vault, we first generate a minutiae table, which includes the non–invertible transformed version of the enrolled fingerprint data, and then perform fingerprint verification on the minutiae table with the input fingerprint data. Based on the experimental results, we confirm that the proposed approach can perform the fingerprint verification with the protected template securely, and cryptography and biometrics can be merged to achieve high security and user convenience at the same time.

The rest of the paper is structured as follows. Section 2 explains previous results to protect biometric data and the overview of the fuzzy fingerprint vault, and Section 3 describes the proposed approach to perform fingerprint verification with the protected template. The experimental results are given in Section 4, and conclusions are made in Section 5.

2 Background

2.1. Fingerprint Verification

A fingerprint authentication system has two phases: *enrollment* and *verification*[11]. In the off-line enrollment phase, an enrolled fingerprint image is preprocessed, and features, called as *minutiae*, are extracted and stored. In the on-line verification phase, the similarity between the enrolled template minutiae and the input minutiae is examined.

`Pre-Processing` refers to the refinement of the fingerprint image against the image distortion obtained from a fingerprint sensor. It consists of three stages. Binary conversion stage applies a low-pass filter to smooth the high frequency regions of the image and threshold to each subsegment of the image. Thinning stage generates a one-pixel-width skeleton image by considering each pixel with its neighbors. In positioning stage, the skeleton obtained is transformed and/or rotated such that valid minutiae information can be extracted.

`Extraction` refers to the extraction of minutiae in the fingerprint image. After this step, some of the minutiae are detected and stored into a template file. A minutia can be specified by its coordinates, angle, and its type(ending/bifurcation).

Based on the minutiae, the input fingerprint is compared with the template fingerprint. Actually, `Match` is composed of alignment stage and matching stage. In order to match two fingerprints captured with unknown direction and position, the differences of direction and position between two fingerprints are detected, and alignment between them needs to be accomplished. Therefore, in alignment stage, transformations such as translation and rotation between two fingerprints are estimated, and two minutiae are aligned according to the estimated parameters. If alignment is performed accurately, matching stage is referred to point matching simply. That is, two minutiae are compared based on their position, orientation, and type. Then, a matching score is computed.

As explained in Section 1, some problems with fingerprint verification systems can arise when the template minutiae has been compromised. For verification systems based on physical tokens such as keys and badges, a compromised token can be easily canceled and the user can be assigned a new token. Similarly, user IDs and passwords can be changed as often as required. Yet, the user only has ten fingers. Thus, if the minutiae are compromised, the user may quickly run out of the minutiae to be used for authentication and cannot re-enroll forever. To solve this problem, some results have been reported.

2.2. Protection of Biometric Information

Ratha, Connell, and Bolle[12] introduced the term "cancelable biometrics" to protect biometric templates. Designing cancelable biometrics had many objectives. First, a cancelable template stored in a database of certain application cannot be used as a template in another application. Second, if a database record(a fingerprint template) is compromised, a new database record can be issued(just like a new password can be issued). Finally, altering a database record(replacing a fingerprint template) is unfeasible because the template can be digitally signed by the issuer, or some privileged information(*e.g.*, an encryption key) can be stored in the template in such a way that it can be released only through biometric recognition. Although they explained their idea with a high-order polynomial function, it was conceptual and they did not implement their idea.

There were other innovative, yet similar methods that did not perform biometric matching. The first is the *fuzzy commitment* scheme[13]. Here, a secret(presumably a private key used for later authentication) is encoded using a standard error correcting code such as Hamming or Reed-Solomon, and then XOR-ed it with a biometric template. To retrieve the secret, a slightly different biometric template can again be XOR-ed, and the result put through an error correcting decoder. Some small number of bit errors introduced in the key can be corrected through the decoding process. The major flaw of this system is that biometric data is often subject to reordering and erasures, which cannot be handled by using this simple scheme.

A second paper[14] based on previous work on a fuzzy commitment scheme had a similar theoretical foundation to this work, but aimed toward a completely different application. Dodis, et al., demonstrated how such data could be used to generate strong keys for any kind of cryptographic application. They used the notion of a fuzzy extractor to describe the process of extracting a random string U from a biometric input b, in such a way that a certain amount of error was allowed for. If the input changes slightly to b, then the extracted U will be the same. To enable the recovery of U from b, the fuzzy extractor also outputs a public string V. Additionally, Dodis described three metrics to measure the variation in the biometric reading: Hamming Distance, Set Distance and Edit Distance.

Recently, Juels and Sudan[16] proposed the *fuzzy vault*, a new architecture with applications similar to Juels and Wattenberg's fuzzy commitment scheme, but is more compatible with partial and reordered data. In the fuzzy commitment, Alice can place a secret value k(*e.g.*, private encryp-

tion key) in a vault and lock(secure) it using an unordered set A. Bob, using an unordered set B, can unlock the vault (access k) only if B overlaps with A to a great extent. The procedure for constructing the fuzzy vault is as follows: First, Alice selects a polynomial p of variable x that encodes k(*e.g.*, by fixing the coefficients of p according to k). She computes the polynomial projections, $p(A)$, for the elements of A. She adds some randomly generated "chaff" points that do not lie on p, to arrive at the final point set R. When Bob tries to learn k (*i.e.*, finding p), he uses his own unordered set B. If B overlaps with A substantially, he will be able to locate many points in R that lie on p. Using error-correction coding, it is assumed that he can reconstruct p(and hence k). The security of the scheme is based on the infeasibility of the polynomial reconstruction problem(*i.e.*, if Bob does not know many points that lie on p, he can not feasibly find the parameters of p, hence he cannot access k). Note that, since this fuzzy vault can work with unordered sets(common in biometric templates, including fingerprint minutiae data), it is a promising candidate for crypto-biometric systems.

Based on the fuzzy vault, some implementations results for fingerprint have been reported. For example, Clancy, et al.[15] and Uludag, et al.[16] proposed a *fuzzy fingerprint vault*. Similarly, Barral, et al[17] added the chaff points to the user's fingerprint information. Note that, their systems inherently assume that fingerprints(the one that locks the vault and the one that tries to unlock it) are pre-aligned. This is not a realistic assumption for fingerprint-based authentication schemes, and limits the applicability of their schemes[16]. For the purpose of explanation, the fuzzy fingerprint vault is shown in Fig. 1, and the details of it can be found in [16].

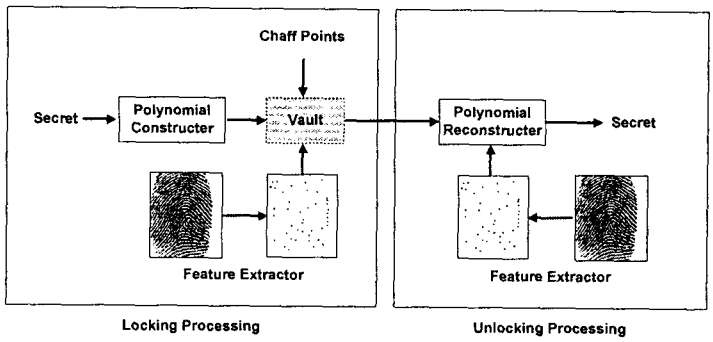

Fig. 1. Illustration of the Fuzzy Fingerprint Vault.

3 Integration of Template Protection with Fingerprint Verification

This section explains a proposed approach to generate protected fingerprint templates and perform fingerprint verification with the protected templates. To explain our approach, we first describe the fuzzy vault in more detail. As explained in Section 2, we assume that Alice can place a secret value m in a vault and lock it using an unordered locking set $\boldsymbol{L}$. Bob, using an unordered unlocking set $\boldsymbol{U}$, can unlock the vault only if $\boldsymbol{U}$ overlaps with $\boldsymbol{L}$ to a great extent. The procedure for constructing the fuzzy vault is as follows: Secret value m is first encoded as the coefficients of some degree k polynomial in x over a finite field $GF(q)$. This polynomial $f(x)$ is now the secret to protect. The locking set $\boldsymbol{L}$ is a set of t values $l_i \in GF(q)$ making up the fuzzy encryption key, where $t > k$. The locked vault contains all the pairs $(l_i; f(l_i))$ and some large number of chaff points (α_j, β_j), where $f(\alpha_j) \neq \beta_j$. After adding the chaff points, the total number of items in the vault is

r. In order to crack this system, an attacker must be able to separate the chaff points from the legitimate points in the vault. The difficulty of this operation is a function of the number of chaff points, among other things. A legitimate user should be able to unlock the vault if they can narrow the search space. In general, to successfully interpolate the polynomial, they have an unlocking set U of t elements such that $L \cap U$ contains at least $k + 1$ elements. The details of the fuzzy vault can be found in [10,15-16].

3.1. Generation of Protected Templates

As explained in Section 2, a fingerprint minutia represented by $m_i = (x_i, y_i, \theta_i, t_i)$ is composed of three elements such as *coordinates*, *angle*, and *type*. In typical fingerprint verification systems, minutiae are stored in the template file, and input minutiae are compared with the template minutiae after aligning them. In this typical fingerprint verification system, however, an attacker can reuse the template minutiae once he steals it. For protecting template minutiae from the attacker, we use the idea of the fuzzy vault scheme that stores a number of chaff minutiae generated randomly as well as the real minutiae. That is, we consider the template(real and chaff minutiae) and the input minutiae as the locking and the unlocking set, respectively. Then, the attacker cannot reuse the fuzzy fingerprint vault directly without separating the real minutiae from the chaff minutiae even if he steals the fingerprint fuzzy vault. Note that, a legitimate user also should be able to separate the real minutiae to compute alignment parameters between the enrolled real minutiae and the input minutiae. However, the legitimate user cannot separate the real minutiae from the chaff minutiae, either. Therefore, we need a new alignment technique for the fuzzy fingerprint vault that does not need this separate operation, and we perform this alignment by modifying the geometric hashing technique[18]. Fig. 2 shows an example of a generated template consisting of real minutiae and chaff minutiae.

Fig. 2. Example of Generating Protected Templates[19]; (a) Example of a Fingerprint, (b) Example of a Typical Template(circles: real minutiae extracted from (a)), (c) Example of a Protected Template(circles: real minutiae, rectangles: chaff minutiae).

3.1.1. Selection of Chaff Minutiae

As in [16], we assume the number of the chaff minutiae as 200. However, it is challenging to perform fingerprint verification with the protected template added by this number of chaff minutiae. To increase the accuracy of our fuzzy fingerprint vault system, we determine the coordinates and the angles of the added chaff minutiae carefully. For example, if the coordinates and the angles of

the added chaff minutiae are similar to those of a real minutia of a legitimate user, then the corresponding input minutia of the legitimate user can be aligned with the added chaff minutiae resulting in an incorrect alignment. Therefore, we first define acceptable margin Δd and $\Delta\theta$ for coordinate and angle, respectively. Since two minutiae within Δd and $\Delta\theta$ are considered as matched minutiae pair, the coordinates and the angles of the added chaff minutiae need to be determined outside the acceptable margin from any real minutiae for accurate fingerprint match. Additionally, newly added chaff minutiae need to consider the coordinates and the angles of the already added chaff minutiae in order not to reveal them as chaff minutiae. However, the type information of the chaff minutiae is selected randomly, because it is less important than other information.

3.1.2. Creation of Enrollment Minutiae Table

Let $m_i = (x_i, y_i, \theta_i, t_i)$ represent a minutia and $\boldsymbol{L} = \{m_i \mid 1 \le i \le r\}$ be a locking set including the real and chaff minutiae. In $\boldsymbol{L}$, the real and chaff minutiae can be represented by $\boldsymbol{G} = \{m_i \mid 1 \le i \le n\}$ and $\boldsymbol{C} = \{m_i \mid n+1 \le i \le r\}$, respectively. Note that, the enrollment minutiae table is generated from $\boldsymbol{L}$.

In the *enrollment minutiae table generation stage*, an enrollment table is generated in such a way that no alignment is needed in the verification process for unlocking vault by using the geometric hashing technique. That is, alignment is pre-performed in the enrollment table generation stage. In verification process, direct comparisons without alignment are performed in 1:1 matching between the enrollment minutiae table and an input fingerprint in order to select the real minutiae($\boldsymbol{G}$) only. Each step in the enrollment minutiae table generation stage is explained in detail in the following.

1) Reference Point Selection Step

In reference point selection step, a minutia is selected as the first minutia from the set of enrollment minutiae($\boldsymbol{L}$). The first minutia is denoted by m_1 and the other remaining minutiae are denoted as $m_2, m_3, \ldots, m_n$. At this moment, the minutia, m_1, is called *basis*.

2) Minutiae Transform Step

In minutiae transform step, minutiae $m_2, m_3, \ldots, m_n$ are aligned with respect to the first minutia m_1 and quantized. Let $m_j(1)$ denote the transformed minutiae, *i.e.*, the result of the transform of the jth minutia with respect to m_1. Also, let T_1 be the set of transformed minutiae $m_j(1)$, *i.e.*, $T_1 = \{m_j(1) = (x_j(1), j_j(1), \theta_j(1), t_j(1)) \mid 1 < j \le r\}$, and T_1 is called the m_1-transformed minutiae set. (Eq. 1) represents the translation and rotation such that features $(x_1, y_1, \theta_1, t_1)$ of m_1 is translated and rotated into $(1,1,1,t_1)$. Let ${}_{TR}m_j(1)$ denote the minutia translated and rotated from the jth minutia with respect to m_1.

$$ {}_{TR}m_j(1) = \begin{pmatrix} {}_{TR}x_j(1) \\ {}_{TR}y_j(1) \\ {}_{TR}\theta_j(1) \\ {}_{TR}t_j(1) \end{pmatrix} = \begin{pmatrix} \cos(\theta_1) & \sin(\theta_1) & 0 & 0 \\ -\sin(\theta_1) & \cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_j - x_1 \\ y_j - y_1 \\ \theta_j - \theta_1 \\ t_j \end{pmatrix}, \text{where } 1 < j \le r \quad \text{(Eq. 1)} $$

$$ m_j(1) = \begin{pmatrix} x_j(1) \\ y_j(1) \\ \theta_j(1) \\ t_j(1) \end{pmatrix} = \begin{pmatrix} \lfloor {}_{TR}x_j(1)/\alpha + 0.5 \rfloor \\ \lfloor {}_{TR}y_j(1)/\alpha + 0.5 \rfloor \\ \lfloor {}_{TR}\theta_j(1)/\beta \rfloor \\ {}_{TR}t_j(1) \end{pmatrix} \quad \text{(Eq. 2)} $$

To reduce the amount of information, quantization is required both in coordinates and angles. This quantization is summarized in (Eq. 2), where α is the quantization parameter for coordinates and β is the quantization parameter for angles. α is determined by the range of coordinates in the extraction stage in the enrollment process, whereas β is determined by the required precision in the verification process.

3) Repeat Step

Step 1) and step 2) are repeated for all the remaining minutiae. When step 1) and step 2) are completed for all the minutiae of the enrollment user, the *enrollment minutiae table* is generated completely.

Finally, the locked vault contains all the pairs $(x_i, y_i, \theta_i, t_i, f(x_i, y_i))$ and some large number of the chaff minutiae $(x_j, y_j, \theta_j, t_j, \beta_j)$, where $f(x_j, y_j) \neq \beta_j$.

3.2. Fingerprint Verification with the Protected Template

After the enrollment process, the verification process to separate the chaff minutiae($\boldsymbol{C}$) from the real minutiae($\boldsymbol{G}$) in the enrollment minutiae table should be performed. In the verification process, minutiae information(unlocking set $\boldsymbol{U}$) of a verification user is obtained and a table, called *verification table*, is generated according to the geometric characteristic of the minutiae. Then, the verification table is compared with the enrollment minutiae table, and the subset of real minutiae is finally selected. Note that, the verification table generation stage is performed in the same way as in the enrollment process.

In comparing the enrollment and verification minutiae tables, the transformed minutiae pairs with the same coordinates, the same angle, and the same type are determined. The minutiae pairs having the maximum number and the same basis are selected as the subset of real minutiae($\boldsymbol{G}$). Also, any additional alignment process is not needed because pre-alignment with each minutia is executed in the enrollment and verification minutiae table generation stage.

Note that, because of the noises and local deformation during acquisition, extracted minutiae from the same finger may have different coordinates and angles over each acquisition. To solve this problem, an adaptive elastic matching algorithm in which tolerance levels are determined according to the polar coordinates of the minutiae was proposed in [1]. In this paper, the coordinate plane is divided into several fields according to the distance from the origin. Each field has its own level of tolerance for x- and y-coordinates. The first field has tolerance level of [-3, 3] which means errors between -3 and 3 in x- or y-coordinate are tolerated. Two transformed minutiae in the first field are considered to have matching coordinates if their coordinates do not differ more than this error range. Tolerance level for angles is 22.5° for all transformed minutiae. Note that, this reduction of the search space required in a straightforward implementation of the geometric hashing can reduce the execution time significantly.

If the unlocking minutiae set $\boldsymbol{U}$ overlaps with the real minutiae of the locking minutiae set $\boldsymbol{L}$ in at least (polynomial degree d + 1) minutiae, for some combinations, the correct polynomial of degree d can be reconstructed. This represents the desired outcome when the locking and the unlocking minutiae sets are from the same finger. Since an attacker cannot separate the real minutiae from the chaff minutiae, he cannot reconstruct the correct polynomial without brute-force attacks.

4 Implementation Details and Experimental Results

For the purpose of evaluating of the accuracy of the proposed approach, a data set of 1,600 fingerprint images composed of four fingerprint images per one finger was collected from 400 individuals by using the optical fingerprint sensor[20]. The resolution of the sensor was 500dpi, and the size of captured fingerprint images was 248×292. As in [16], we selected the number of the chaff minutiae as 200, and they were generated by using a random number generator. The average number of minutiae in our data set is 36, and the average size of the enrollment minutiae table is 236×235.

Table 1. Experimental Results with Genuine and Imposter Sets.

	Genuine	Imposter
Average # of Minutiae	36	36
Average # of Matched Minutiae	17	6
Average # of Matched Real Minutiae	16	2
Average # of Matched Chaff Minutiae	1	4

Table 2. Experimental Results for the Various Polynomials.

	FRR	FAR
Degree-9 Polynomial	0.08	About 0
Degree-10 Polynomial	0.12	0
Degree-11 Polynomial	0.18	0
Degree-12 Polynomial	0.26	0

First, we evaluate the capability of separating the real minutiae from the chaff minutiae with the genuine and the imposter sets, respectively. As shown in Table 1, the average number of the matched minutiae is 16 and 1 for the real and the chaff minutiae in the genuine experiment, respectively. On the contrary, 2 and 4 minutiae are matched for the real and the chaff minutiae in the imposter experiment, respectively. Also, we can consider that the input minutiae are from a genuine if the unlocking set reconstructs the correct polynomial. If we use a 9-degree polynomial with 10 coefficients, 10 unique projections are required for unlocking the 9-degree polynomial. Based on our experiment, 4,391 of the 4,800 attempts were able to successfully unlock the vault in the genuine experiment. The remaining 409 query fingerprints selected fewer than the required number of the real minutiae(*i.e.*, less than 10), hence they were unable to unlock the vault. Also, the False Reject Rate(FRR) of the proposed approach is 0.081 and the Genuine Accept Rate is 0.915.

For evaluating the corresponding False Accept Rate(FAR), we tried to unlock the vaults with fingerprint minutiae of the imposter. Hence, we have 399 imposter unlocking attempts for a distinct finger, and only one fingerprint of the attempts could unlock the vault. Hence, experimental FAR is about 0%. Table 2 shows the experimental FRR and FAR for the various polynomials. As shown in Table 2, the FAR is 0% once using from 9-degree to 12-degree polynomials.

We can also analyze the security of the system mathematically as in [16]. Assume that we have an attacker who does not use real fingerprint data to unlock the vault; instead he tries to separate real minutiae from chaff minutiae in the vault using brute-force. If we use a 12-degree polynomial with 13 coefficients, attacker needs at least 13 minutiae to reconstruct the correct polynomial. The vault has 236 minutiae(36 of them are real, remaining 200 are chaff); hence there are a total of $C(236,13) \approx 8.0\times10^{20}$ combinations with 13 elements. Only $C(36,13) \approx 2.3\times10^{9}$ of these combinations will reveal the secret(*i.e.*, unlock the vault). Thus, it will need an average of 3.4×10^{11} (=C(230,9)/C(30,9)) evaluations for an attacker to crack the vault.

To guarantee for higher security, we are developing an approach to handle a more number of chaff points. Because adding the number of chaff points in restricted to the size of the 2D hash table determined by a given fingerprint sensor, we are considering a 3D hash table. By using the 3D hash table, we can add more than 200 chaff points. However, it will take more execution time and memory space, and an efficient approach needs to be developed.

Finally, we compare the performance of the proposed approach and that of the Uludag's approach[16] as shown in Table 3. To compare two approaches, we show the performance with 8-degree polynomials that were selected in Uludag's experiment. The FRR and FAR of the proposed approach are 0.05 and 0.0016, respectively. As our fingerprint data set and variable parameters(*i.e.*, the number of the real minutiae) are different from ones used in Uludag's experiment, the reported FRR and FAR can be different with the environment of Uludag's experiment. Nevertheless, we should note that Uludag aligned two fingerprints manually while our proposed approach aligned automatically. To crack the vault, however, an attacker needs an average of 5.7×10^{7} evaluations with our approach and 5.3×10^{10} evaluations with the Uludag's approach. This is the reason that our experiment uses more real minutiae(36 minutiae) than ones(18 minutiae) used in Uludag's experiment.

Table 3. Performance Comparison with Degree-8 Polynomials.

	Uludag, et al.[16]	Proposed Approach
Alignment	Manual	Automatic
FRR	0.21	0.05
FAR	0	0.0016
Security	5.3×10^{10}	5.7×10^{7}

5 Conclusions

The use of biometrics in user authentication systems is very promising. However, without adequate security considerations, the compromise of such biometric data may make them useless for the user forever. In this paper, we proposed an approach to generate a non–invertible transformed version of the fingerprint data and to perform fingerprint verification with the protected fingerprint template. Based on the idea of the fuzzy vault, we generated a minutiae table that included the non–invertible transformed version of the enrolled fingerprint data, and then performed fingerprint verification on the minutiae table with the input fingerprint data.

To evaluate the effectiveness of our approach, we conducted experiments with actual fingerprint data. Based on the experimental results, our approach of using the minutiae table generated with the idea of the fuzzy vault can perform the fingerprint verification securely with automatic alignment.

References

[1] D. Maltoni, *et al.*, *Handbook of Fingerprint Recognition*, Springer, 2003.

[2] S. Elliott, "Differentiation of Signature Traits vis-à-vis Mobile-and Table-Based Digitizers," *ETRI Journal*, Vol. 26, No. 6, pp.641-646.

[3] R. Bolle, J. Connell, and N. Ratha, "Biometric Perils and Patches," *Pattern Recognition*, Vol. 35, pp. 2727-2738, 2002.

[4] S. Prabhakar, S. Pankanti, and A. Jain, "Biometric Recognition: Security and Privacy Concerns," *IEEE Security and Privacy*, pp. 33-42, 2003.

[5] U. Uludag, *et al.*, "Biometric Cryptosystems: Issues and Challenges," *Proc. of IEEE*, Vol. 92, No. 6, pp. 948-960, 2004.

[6] Y. Jeong, K. Yoon, and J. Ryou, "A Trusted Key Management Scheme for Digital Rights Management," *ETRI Journal*, Vol.27. No.1, pp.114-117, 2005.

[7] T. Matsumoto, et al., "Impact of Artificial Gummy Fingers on Fingerprint Systems," *Proc. of SPIE*, Vol. 4677, 2002.

[8] C. Hill, "Risk of Masquerade arising from the Storage of Biometrics," *BS Thesis - Australian National U.*, 2002.

[9] B. Schneier, "The Uses and Abuses of Biometrics," *Communications of the ACM*, Vol. 42, No, 8, pp. 136, 1999.

[10] A. Juels and M. Sudan, "A Fuzzy Vault Scheme," *Proc. of Symp. on Information Theory*, pp. 408, 2002.

[11] D. Moon, *et al.*, "Implementation of the USB Token System for Fingerprint Verification," *LNCS 2749*, pp. 998-1005, 2003.

[12] N. Ratha, J. Connel, and R. Bolle, "Enhancing Security and Privacy in Biometrics-based Authentication Systems," *IBM Systems Journal*, Vol. 40, No. 3, pp. 614-634, 2001.

[13] A. Juels and M. Wattenberg, "A Fuzzy Commitment Scheme," *Proc. of ACM Conf. on Computer and Comm. Security*, pp. 28-36, 1999.

[14] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data," *LNCS 3027*, pp. 523-540, 2004.

[15] T. Clancy, N. Kiyavash, and D. Lin, "Secure Smartcard-based Fingerprint Authentication," *Proc. of ACM SIGMM Multim., Biom. Met. & App.*, pp. 45-52, 2003.

[16] U. Uludag, S. Pankanti, and A. Jain, "Fuzzy Vault for Fingerprints," *LNCS 3546*, pp. 310-319, 2005.

[17] C. Barral, *et al.*, "Externalized Fingerprint Matching," *LNCS 3072*, pp. 309–315, 2004.

[18] H. Wolfson and I. Rigoutsos, "Geometric Hashing: an Overview," *IEEE Computational Science and Engineering*, Vol. 4, pp. 10-21, Oct.-Dec. 1997.

[19] A. Adler, "Vulnerabilities in Biometric Encryption Systems," *LNCS 3546*, pp. 1100-1109, 2005.

[20] D. Ahn, *et al.*, "Specification of ETRI Fingerprint Database(in Korean)," *Technical Report – ETRI*, 2002.