Secure Fingerprint Matching with External Registration

James Reisman[1], Umut Uludag[2], and Arun Ross[3]

[1] Siemens Corporate Research, 755 College Road East, Princeton, NJ, 08540
james.reisman@siemens.com
[2] Computer Science and Engineering, Michigan State University, East Lansing, MI, 48824
uludagum@cse.msu.edu
[3] Lane Department, West Virginia University, Morgantown, WV, 26506
ross@csee.wvu.edu

Abstract. Biometrics-based authentication systems offer enhanced security and user convenience compared to traditional token-based (e.g., ID card) and knowledge-based (e.g., password) systems. However, the increased deployment of biometric systems in several commercial and government applications has raised questions about the security of the biometric system itself. Since the biometric traits of a user cannot be replaced if compromised, it is imperative that these systems are suitably secure in order to protect the privacy of the user as well as the integrity of the overall system. In this paper, we first investigate several methods that have been proposed in the literature to increase the security of the templates residing in a biometric system. We next propose a novel fingerprint matching architecture for resource-constrained devices (e.g., smart cards) that ensures the security of the minutiae templates present in the device. Experimental results describing the impact of several system parameters on the matching performance as well as the computational and storage costs are provided. The proposed architecture is shown to enhance the security of the minutiae templates while maintaining the overall matching performance of the system.

1 Introduction

Fingerprint-based biometric systems are among the most popular personal authentication systems partly because of their long history. Further, fingerprint sensors are relatively small and cheap and, hence, can be readily incorporated into devices such as cellular phones, laptop computers, computer keyboards, PDAs, etc. This has resulted in a proliferation of these devices in the market.

A generic fingerprint system consists of four main modules: (a) the acquisition module which senses the fingerprint of a user and produces a raw image; (b) the feature extraction module which processes the raw image and extracts a compact set of features representing the fingerprint; (c) the matching module which compares the extracted feature set with the templates residing in the database by generating match scores; and (d) the decision module which determines or verifies the identity of the user based on the match scores. The performance of a fingerprint matcher hinges on the accurate derivation of transformation parameters (typically, the translation and rotation values) relating two feature sets. This process, known as registration or alignment, impacts the matching performance of minutiae-based systems[1].

[1] Some algorithms avoid alignment. See [1], for example

T. Kanade, A. Jain, and N.K. Ratha (Eds.): AVBPA 2005, LNCS 3546, pp. 720–729, 2005.

Ratha et al. [2] discuss the various types of attacks that can be launched against a fingerprint system which can undermine the integrity of the system and, subsequently, result in the loss of privacy of the user. Therefore, it is necessary to design techniques that either prevent or neutralize the effect of these attacks. In this paper we focus on template protection schemes in resource-constrained systems such as smart cards where matching can potentially be performed in an external host computer.

Yang and Verbauwhede [3] describe a fingerprint system consisting of a secure and a non-secure part. The biometric template resides in the so-called secure part and the matching operation, which requires access to the stored biometric template, is executed in this part. In their formulation, the authors augment local minutiae information with minutiae neighborhood information. The distances (d) and angles (φ) between a specific minutia M and its nearest N (typically 6) neighbors, and the directions (ϑ) of these neighbors define the local structure for M. A collection of local structures representing all the minutiae in the image constitutes the template. During the matching process, no alignment (registration) information describing the translation and rotation between the template and input feature set is provided. Rather, the matcher compares the local structures of all minutiae in the input set against those in the template and two local structures are deemed to be a match if at least 3 of their 6 neighbors are similar in terms of the (d, φ, ϑ) triplet; the final match score is a function of the total number of matched structures. Experiments conducted on a very small database (of 10 different fingers) show that a 1% FRR (False Reject Rate) can be achieved at a 0% FAR (False Accept Rate).

Pan et al. [4] introduce a match-on-card system for fingerprints. Their system uses (x, y, θ) values for representing the minutiae set. The 3-dimensional transformation space (describing the alignment) is discretized into a set of candidate transformations defined by an accumulator array. The transformation between the input feature set and the template is obtained by comparing all the minutiae in the two sets. The authors employ a hierarchical scheme that evolves from a coarse-to-fine scale to determine the final transformation. First, a coarsely quantized accumulator array is used to compute the approximate transformation. The array cell corresponding to the best transformation is further discretized and the process repeated several times. The iterative nature of the algorithm ensures that only a limited amount of memory is required at any instance. An EER (Equal Error Rate) of ~6% is reported for the experiments carried out with 100 different fingers.

Moon et al. [5] describe another minutiae-based match-on-card system. Again, the minutiae are represented as (x, y, θ) triplets. The registration between the input feature set and the template is accomplished *outside* the smart card in order to reduce its workload. To facilitate this, the smart card stores the average horizontal and vertical coordinate values (μ_X^E, μ_Y^E) and the direction (μ_θ^E) of all minutiae in the enrolled template. When an input minutiae set is presented to the smart card for matching, these average values are calculated ($\mu_X^I, \mu_Y^I, \mu_\theta^I$) by the host computer. The parameters $\mu_X^E, \mu_Y^E, \mu_\theta^E$ are next transferred from the smart card to the host, where the minutiae of the input set is translated and rotated such that

$\mu_X^I = \mu_X^E, \mu_Y^I = \mu_Y^E, \mu_\theta^I = \mu_\theta^E$. The transformed input minutiae are sent to the smart card, where point matching yields the number of matched minutiae. The authors only report the genuine matching results on a very small database (of 10 different fingers) and, hence, it is difficult to assess the overall matching performance of the system. However, in our opinion, the use of average coordinates and angle provides a rather coarse (and possibly, incorrect) registration that will degrade system performance.

Ishida et al. [6] present a matching scheme for smart cards. The smart card stores the binary fingerprint image as well as the core point of a gray-scale fingerprint image. When an input fingerprint is presented for matching, the host computer transmits the location of its core to the smart card which determines the correct translation parameter. Next, the smart card selects some coordinates (typically around the minutiae points of the enrolled template), and after modifying them using the translation parameter, sends these coordinates to the host. The host then transmits a rectangular image patch (called *chip*, which is centered around the received coordinate) of the input image to the smart card, which proceeds to calculate the similarity between the received image chip and the corresponding chip of the enrolled image. To account for small variations in translation, multiple input chip images obtained via small shifts around the initial chip image are sent to the smart card. If, for a particular chip location, no match can be found in the input image, the smart card selects another chip location, and repeats the process. For a database with 576 fingers, the authors report that a 2% FRR is achieved at 0.1% FAR.

Yang et al. [7] discuss a method for fingerprint verification in another resource-constrained device – the Personal Digital Assistant (PDA). The authors implement the minutiae extraction, core point detection and minutiae matching operations in fixed-point arithmetic (32 bit words with [-32768, 32767] range), citing that majority of processors in mobile applications do not support floating-point arithmetic. In their scheme, minutiae (represented via (x, y, θ) triplets) are extracted using a modified version of the ridge line tracing algorithm in [8]. The transformation parameters are recovered using the location of core point in the two images. To optimize the computation time, only a subset of minutiae in the input set is used for matching; in fact, only minutiae closest to the fingerprint core are used during the matching process, which in itself employs a simple bounding box technique. The authors report that on a database of 383 different fingers, an EER of nearly 7% is achieved (the corresponding floating-point EER is reported as 6%).

2 System Architecture

The goal of automatic fingerprint matching systems is to accurately determine or validate an individual's identity. However, these systems are vulnerable to spoofing, registration template theft, and other attack methods [10]. In order to discourage identity theft, the registered fingerprint feature set needs to be held in a secure location. One possibility is to store the registered fingerprint information and accomplish matching on a *smart chip* device (e.g., smart card, USB dongle) kept in the possession of the individual. The chip releases the secure information (e.g., cryptographic keys), or a verification signal, upon being presented with a fingerprint feature representation matching the registered representation stored on the chip. The difficulty in

this application scenario is that the smart chip processor is generally incapable of handling fingerprint matching duties with satisfactory accuracy.

We propose a method where sufficient information is passed to the fingerprint scanning unit's processor (i.e., an external processor) to allow alignment of the template, but which would be insufficient to reconstruct the user's fingerprint representations. This is accomplished by using a multi-step approach in which certain selected verification tasks are moved off chip. The steps needed for fingerprint verification – fingerprint representation extraction, alignment and matching – are decoupled into separate processes.

The method is built upon the previously proposed ridge-texture based matching algorithm of Ross et al. [11]. Matching using this algorithm is fast and accurate once the templates are properly aligned, and can be handled by the smart chip's processor. More computationally intensive operations such as feature extraction are implemented in the scanning unit. Ridge map alignment is achieved by a comparison of minutiae information in the form of minutiae triplet sets. The minutiae triple sets are constructed in such a way that neither the individual's ridge map nor minutia map can be regenerated from it. This allows much of the computationally expensive ridge map alignment to be moved off-chip without loss of security.

2.1 Basic Matching Algorithm

Here, a simple description of the algorithm that forms the basis for the smartcard based fingerprint matching is provided. For more details about this algorithm, please see [11]. A set of Gabor filters, whose spatial frequencies correspond to the average inter-ridge spacing in a fingerprint, is used to capture the ridge strength at equally spaced orientations. The number of these filters is an important system parameter: e.g., if 4 filters are used, the filters are oriented at 0^o, 45^o, 90^o and 135^o; for 8 filters, the orientation difference between filters is 22.5^o, allowing a finer representation of the fingerprint ridges. A square tessellation (based on the block size used for grid generation, e.g., 8x8 blocks) of the filtered images is then used to construct a multi-dimensional *ridge feature map*. This map is expressed as an array of values, where each value can be represented with a fixed number of bits (e.g., a 4-bit representation allows 16 different levels for each feature while an 8-bit representation allows 256 levels, increasing the granularity). A similarity score of two fingerprints can be generated by finding the Euclidean, Hamming or similar vector distance between the two ridge feature maps. Further, multiple templates (e.g., 3) can be used instead of a single template during verification: using multiple templates helps in better capturing the variability of the fingerprint, hence increasing the matching accuracy.

When the ridge feature map is extracted, the 2 images need to be aligned and tessellated in the same manner. This requires the prior determination of the relative transformation between the two images. It is this step that is moved off the smart chip in the proposed algorithm.

2.2 Triplet-Based Fingerprint Registration

First, the minutiae of the fingerprint are extracted. Then, a minutia triplet (Figure 1) is used to describe a set of 3 minutia points.

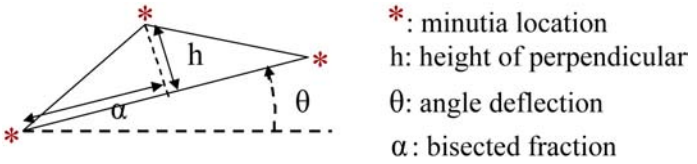

Fig. 1. A minutiae triplet triangle

Four parameters describe the shape of the minutia triplet (h, α, s, θ):

Height of Perpendicular (h): The height in pixels from the longest side to the vertex opposite it.

Bisected Fraction (α): The fraction (from 0.0 to 1.0) of the longest side from the perpendicular to its clockwise vertex (the "Primary Vertex").

Deflection Angle (θ): The angle of deflection that the longest side (running from its clockwise vertex to its counterclockwise vertex) makes with the horizontal ray running left to right in the fingerprint image.

Side Length (s): The length associated with α.

The location of the triplet in the image is defined by the (x,y) location (in pixels) of the primary vertex.

The full triplet list contains all possible combinations of the minutiae in groups of three. The extraction of the parameters depends upon the accurate determination of the longest side of the triplet triangle and the ordering of the vertices. The following qualifying criteria are applied to avoid unsuitable triplets:

- The longest side of a triplet triangle must be greater than the 2nd longest side by a certain threshold.
- The height of the triplet triangle must be greater than a certain threshold.

The selected triplets may still be more than what can be stored in a given smart chip. This number can be reduced by ordering the triplets using a scoring function and retaining the N highest scoring triplets (N is defined by the storage limitations of a given smart chip). The scoring function ranks the triplets according to various factors including:

- Repeatability of the three minutia in a given triplet in subsequent fingerprint samples (less likely if the triplet uses minutia on the periphery of the print, or far away from each other), and
- Repeatability of parameter measurement between samples (less likely if a triplet used minutia that are close together).

The scoring function is maximized when the triangle is centrally located and of "medium size", best satisfying both parameter stability and minutia repeatability. Finally, the pruned triplet list and the ridge feature map for the enrollment template are stored in the smart card.

2.3 Fingerprint Verification

Firstly, the minutiae are extracted from the query fingerprint and a complete triplet list generated in the scanning unit. The smart chip device is then requested to transmit

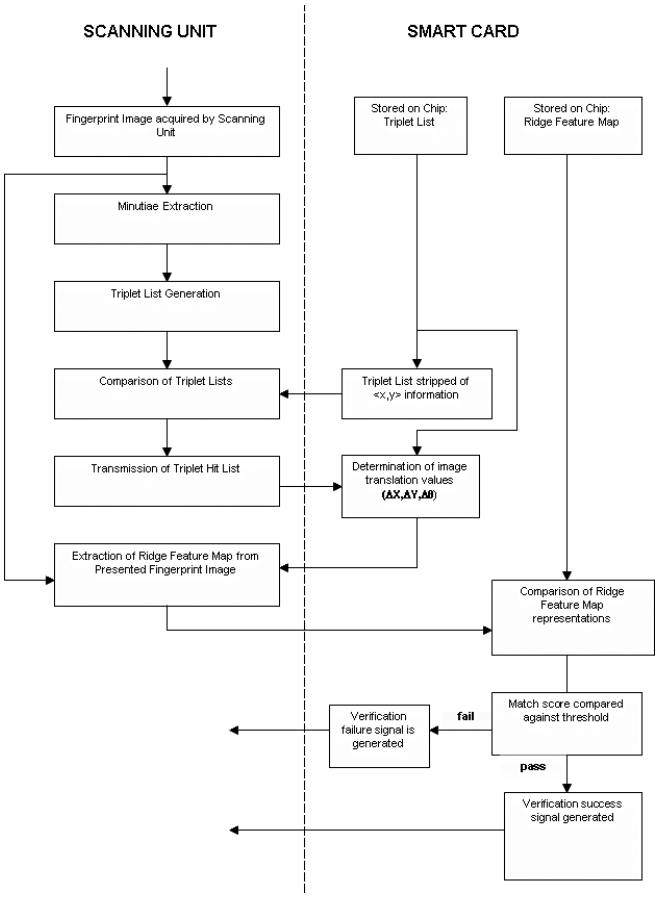

Fig. 2. Exchange of information between the smart card and the scanning unit

the parameters of the pruned triplet list, which is sent withholding the identifying (x,y) location information. Each transmitted triplet is tracked by an index number. The triplet list from the smart chip (list A) is next compared to the complete triplet list extracted by the scanning unit (list B). A triplet *hit* is said to occur when all triplet parameters from a triplet pair are within proscribed distances.

The resultant triplet hit list contains the following information: (index^a, x^b, y^b, $\Delta\theta^{ab}$) where $\Delta\theta^{ab} = \theta^a - \theta^b$. The hit list is transmitted to the smart chip device. Using the index identifier to reference $\mathbf{x}^a$ and $\mathbf{y}^a$, each triplet hit is converted into the perceived $\mathbf{\Delta x}^{ab} = \mathbf{x}^a - \mathbf{x}^b$, $\mathbf{\Delta y}^{ab} = \mathbf{y}^a - \mathbf{y}^b$, and $\mathbf{\Delta\theta}^{ab}$ transformations. Using the Parzen window voting technique, the global $\mathbf{\Delta X}$ and $\mathbf{\Delta Y}$ translations for the image is estimated from the individual $\mathbf{\Delta x}^{ab}$ and $\mathbf{\Delta y}^{ab}$ values. The hit list is then pruned to retain hits whose $\mathbf{\Delta x}^{ab}$ and $\mathbf{\Delta y}^{ab}$ fall within a threshold distance from the global $\mathbf{\Delta X}$ and $\mathbf{\Delta Y}$. Parzen window voting is next applied to the pruned list to determine the global $\mathbf{\Delta\theta}$ value. The global transformation values ($\mathbf{\Delta X}$, $\mathbf{\Delta Y}$, and $\mathbf{\Delta\theta}$) are sent from the smart chip to the scanning device, where they are used to "align" the ridge feature map of

the query fingerprint. The aligned ridge feature map is transmitted to the smart chip for matching, using the method described in Ross *et al*. [10]. If the match results in a distance score within the match tolerance threshold, a verification signal or the secret information (e.g., cryptographic key stored in smart card) is transmitted to the outside host.

3 Experimental Results

In this section, we provide results aiming to show the effect of different system parameters (block size, number of filters, feature bit quantization, and number of templates per user) on the overall performance. These results can be used to set the system parameters for a specific smart-card based fingerprint matching system, considering required accuracy and execution time, and the smart card capacity. Note that when using multiple templates, the conglomerate matching distance between a query and multiple template images is selected to be the minimum value of the associated distance set.

We used the Siemens Fingerprint Database, containing 100 images each of 36 distinct users. Images are 224x288 pixels, with 500 DPI resolution, and a 8 bit/pixel gray level quantization. ROC (Receiver Operating Characteristics) curves are generated by considering 13 images per user as template images, and 87 images per user as query images.

Effect of Block Size: Figure 3 shows the ROC curves for different block sizes, with other parameters fixed (number of filters is 4, bits per feature is 8). We see that 8x8 block size is the optimal one: smaller blocks (4x4) make the system too sensitive to block tessellation boundary artifacts. On the other hand, larger blocks (16x16) decrease the discriminability of templates, hence resulting in inferior performance.

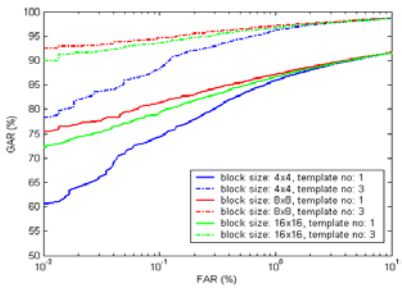

Fig. 3. Effect of block size

Also, using multiple templates per user increases the performance of the system considerably, at the cost of increased storage space and processing time (especially critical for smart-card based applications).

Effect of Number of Filters: Figure 4 shows the ROC curves for different number of filters, with the other parameters fixed (block size is 8x8, bits per feature is 8). It is observed that increasing the number of filters results in increased performance, reaching saturation at 4 filters. Hence, choosing 4 filters is a good tradeoff between performance, and computational and storage costs.

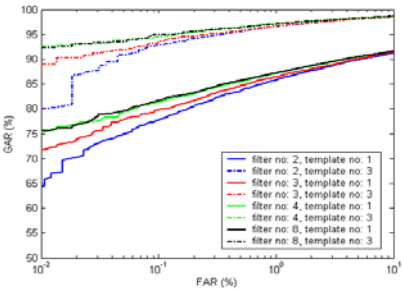

Fig. 4. Effect of number of filters used

Effect of Feature Bit Number: Figure 5 shows the ROC curves for different bit numbers per feature, with other parameters fixed (block size is 8x8, filter number is 4). We observe a performance increase when the number of bits is increased from 4 to 8, especially for small FAR (False Accept Rate) values. On the other hand, increasing the bit number to 16 does not result in any improvements.

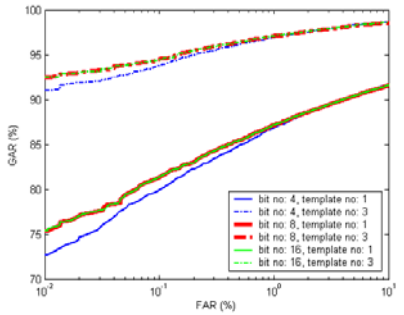

Fig. 5. Effect of bits used on matching

Effect of Multiple Template Number: Figure 6 shows the ROC curves for different template numbers per user, with other parameters fixed (block size is 8x8, filter number is 4, bits per feature is 8). As expected, increasing the number of templates results in increased performance, especially for small FAR values.

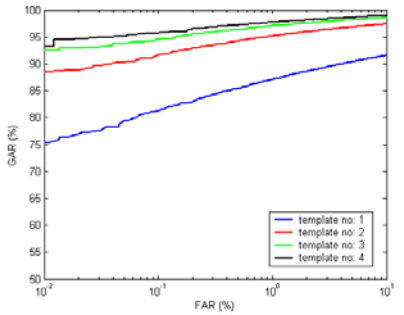

Fig. 6. Effect of number of templates

Accuracy vs. Template Size: Figure 7(a) shows the EER values against associated template sizes (in KBytes) for several different system configurations. Similarly, Figure 7(b) shows the FRR (False Reject Rate) values against associated template sizes, at a FAR equal to 0.1%. Note that in these figures, the configurations that result in values closer to the lower left corner of the graphs can be thought of as *optimal* ones (low error rate and small template size).

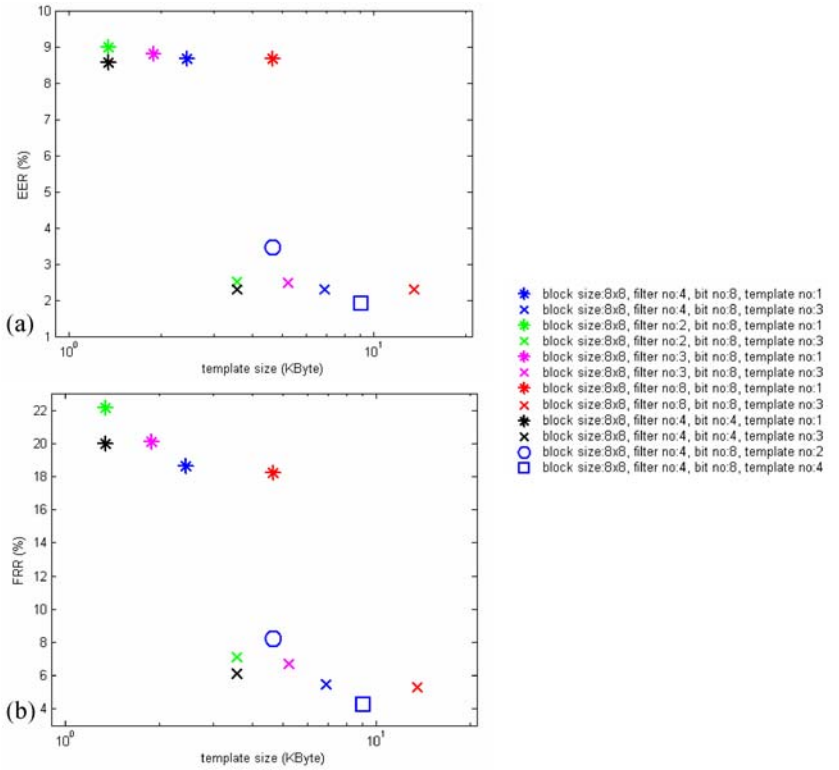

Fig. 7. (a) EER vs. template sizes for different configurations, (b) FRR vs. template sizes for different configurations (FAR = 0.1%)

In light of these experiments, we can conclude that it is better to use multiple templates per user (e.g., 3) compared to using only one template and trying to extract more information from each template (by increasing filter number, decreasing the block size, or increasing the bit numbers per feature). Note that the computational and storage costs are linearly dependent on the number of templates (e.g., for 3 templates, the required storage space is tripled, along with the computational time).

4 Conclusions

A novel fingerprint matching technique suited for execution in resource-constrained devices (e.g., smart cards) is proposed. The registration of fingerprints, which is computationally expensive, takes place in the host computer (i.e., scanning unit),

whereas the matching takes place within the resource-constrained device, without revealing the user template to the outside environment. This accomplishes both template security (since the matching takes place within the smart card, which can be protected against malicious attacks via software or hardware) and high accuracy (since the registration benefits from the availability of abundant computational resources in the host) at the same time. Experimental results showing the effects of several system parameters on the matching accuracy have been provided. These experiments help in evaluating the feasibility of candidate smart-card based fingerprint matching techniques.

References

1. M. Bazen and S. H. Gerez, “An Intrinsic Coordinate System for Fingerprint Matching”, Proc. 3rd International Conference on Audio- and Video-Based Biometric Person Authentication, pp. 223-228, Sweden, June 6-8, 2001.
2. N. Ratha, J. H. Connell, and R. M. Bolle, "An Analysis of Minutiae Matching Strength", Proc. 3rd International Conference on Audio- and Video-Based Biometric Person Authentication, pp. 223-228, Sweden, June 6-8, 2001.
3. S. Yang and I.M. Verbauwhede, “A secure fingerprint matching technique”, *Proc. ACM SIGMM Workshop on Biometrics Methods and Applications*, pp. 89-94, 2003.
4. S.B. Pan, D. Moon, Y. Gil, D. Ahn, and Y. Chung, “An ultra low memory fingerprint matching algorithm and its implementation on a 32-bit smart card”, *IEEE Trans. Consumer Electronics*, vol. 49, no. 2, pp. 453-459, May 2003.
5. Y.S. Moon, H.C. Ho, K.L. Ng, S.F. Wan, and S.T. Wong, “Collaborative fingerprint authentication by smart card and a trusted host”, *Proc. Canadian Conf. Electrical & Computer Engineering*, pp. 108-112, 2000.
6. S. Ishida, M. Mimura, and Y. Seto, “Development of personal authentication techniques using fingerprint matching embedded in smart cards”, *IEICE Trans. Inf. & Syst*., vol. E84-D, no. 7, pp. 812-818, 2001.
7. T.Y. Yang, Y.S. Moon, and K.C. Chan, “Efficient implementation of fingerprint-verification for mobile embedded systems using fixed-point arithmetic”, *Proc. ACM Symp. Applied Computing*, pp. 821-825, 2004.
8. D. Maio and D. Maltoni, “Direct gray-scale minutiae detection in fingerprints”, *IEEE Trans. PAMI*, vol. 19, no. 1, pp. 27-40, 1997.
9. A.K. Jain, L. Hong, S. Pankanti, and R. Bolle, “An identity authentication system using fingerprints”, *Proc. IEEE*, vol. 85, no. 9, pp. 1365-1388, 1997.
10. U. Uludag and A.K. Jain, “Attacks on biometric systems: a case study in fingerprints”, *Proc. SPIE-EI 2004, Security, Steganography and Watermarking of Multimedia Contents VI*, vol. 5306, pp. 622-633, 2004.
11. A. Ross, A. Jain, and J. Reisman, “A hybrid fingerprint matcher”, *Pattern Recognition*, vol. 36, pp. 1661-1673, 2003.