Fast Match-on-Card technique using In-Matcher Clustering with ISO Minutia Template

Tai-Pang Chen

Institute for Infocomm Research,
1 Fusionopolis Way, #21-01 Connexis (South Tower), Singapore 138632
E-mail: tpchen@i2r.a-star.edu.sg

Wei-Yun Yau

Institute for Infocomm Research,
1 Fusionopolis Way, #21-01 Connexis (South Tower), Singapore 138632
E-mail: wyyau@i2r.a-star.edu.sg

Xudong Jiang

School of Electrical and Electronic Engineering,
Nanyang Technological University,
S1-B1c-105, EEE, NTU, 50 Nanyang Avenue, Singapore 639798
E-mail: exdjiang@ntu.edu.sg

Abstract: Fingerprint match-on-card is receiving more attention from government and the IT industry as it provides higher level of security. In addition, it also has less privacy concern as the enrolled template does not leave the card. To address the interoperability needs for match-on-card implementation, the ISO standard ISO/IEC19794-2, defines the finger minutiae data format that contains only the basic information for matching. However, with such limited basic information, fingerprint matching is a challenge for match-on-card with limited processing power, especially when dealing with deformation and common correspondence for alignment. In this paper, a novel in-matcher clustering method is proposed to search for the matched clusters of minutiae with the least deformation error. The proposed clustering method solves the deformation and alignment problems. However, matched minutia clusters may contain false or missing minutiae which may cause mismatch in some regions between the genuine user and imposter. Such mismatch may result in false acceptance. To alleviate this problem, a further matching step using Mahalanobis distance to measure the inter-cluster similarity is proposed to remove the wrongly matched clusters. Finally, the overall match score is generated by combining the minutia matching (local matching) in group (cluster) and the matching of the geometrical structure between groups (global matching). The proposed algorithm achieved an average EER<=5.1979% using all FVC databases (except FVC2006 DB1a). In the NIST evaluation, the achieved False Match Rate (FMR) = 0.001 and False Non-Match Rate (FNMR) = 0.08 (average across minutiae detectors of different vendors) and the average

on-card verification time is 1.01s using 8-bit native smartcard running at 25MHz with 5K RAM.

Keywords: Fingerprint; biometric comparison; minutiae matching; in-matcher clustering; inter-cluster similarity; smartcard.

Reference to this paper should be made as follows: Chen, T.P., Yau, W.Y. and Jiang, X.D. (2011) 'Fast Match-on-Card technique using In-Matcher Clustering with ISO Minutiae Template', *Int. J. Biometrics*, Vol. X, No. X, pp.XX–XX.

Biographical notes: Tai-Pang Chen received the BEng (1996) and MEng by research (1998) in Computer System Engineering, both from the Royal Melbourne Institute of Technology University, Australia. Currently, he is working at the Institute for Infocomm Research as Senior Research Engineer for the Robotics Department. From 2008 to 2010, he was the Project Editor of ISO/IEC 24787 — Information technology — Identification cards — On-card biometric comparison in ISO/IEC JTC1 SC17/WG11. In 2014, he was appointed as Chairman, Biometric Technical Committee under IT Standard Committee, Singapore. Tai Pang is the recipient of the Standard Council Merit Award 2009. He is also pursuing a part-time Ph.D at the School of Electrical and Electronic Engineering, Nanyang Technological University. His research interests are in the area of fingerprint authentication, minutiae matching on resources-constrained platform, pattern recognition and parallel processing algorithm for computer vision and secured biometrics.

Wei-Yun Yau received the BEng (Electrical) from the National University of Singapore (1992), MEng degree (1995) and PhD degree (1999) from the Nanyang Technological University, Singapore. From 1997 to 2002, he was with the Centre for Signal Processing, Singapore leading the research and development effort in biometrics signal processing. His team won the top 3 positions in both speed and accuracy at the international Fingerprint Verification Competition 2000. Currently, he is a Programme Manager with the Institute for Infocomm Research, leading the research in Robotics Department. Wei-Yun also serves as a member of the IAPR's Technical Committee on Biometrics (TC4), Exco member of the Asian Biometric Consortium, Chairman of the IPTV Working Group, Singapore and Chairman of Biometrics Technical Committee, Singapore. He was also the project editor of ISO/IEC JTC1 SC37 29794-4 on fingerprint quality score normalization and co-editor for ISO/IEC JTC1 SC37 29794-1. Wei-Yun is the recipient of the TEC Innovator Award 2002, the Tan Kah Kee Young Inventors' Award 2003 (Merit), IES Prestigious Engineering Achievement Awards 2006 and the Standards Council Distinguished Award 2007. His research interest includes biometrics, active vision system, personalized media and interactive TV and has published widely, with 8 patents granted and over 100 publications in these areas. A paper he co-published in 2008 received the Pattern Recognition Journal Honorable Mention 2010.

Xudong Jiang received the B.Eng. and M.Eng. degrees from the University of Electronic Science and Technology of China in 1983 and 1986, respectively, and the PhD degree from Helmut Schmidt University Hamburg, Germany, in 1997, all in electrical engineering. Currently, he is a tenured associate professor at Nanyang Technological University, Singapore. Dr. Jiang has published over 100 papers and holds 7 patents. His research interest includes signal/image processing, pattern recognition, computer vision, machine learning and biometrics.

1 Introduction

Fingerprint authentication has been extensively deployed especially for border control because of its good accuracy in verifying a person's identity. Moreover Jain et al. (2002) showed that fingerprint can be used to distinguish identical twins, though with a slightly lower accuracy than non-twins. The advancement in semi-conductor technology has produced small, low cost and low power consumption fingerprint sensor and the associated security token. With the increasing use of fingerprint recognition in many civilian, commercial and financial applications (Jain and Pankanti, 2009), research on realizing fingerprint authentication on low-cost, energy efficient devices and security tokens such as smartcards remain crucial.

Schouten et al. (2009) described the use of biometrics with e-passport using the International Civil Aviation Organization (ICAO) standard. The proposed approach requires that the enrolled biometric template stored in the e-passport to be sent to the biometric terminal to verify the person's identity. This is an off-card approach. The sending out of biometric information poses a security threat as it allows an attacker to steal the passport holder's biometric information for illegitimate purpose. To overcome this, the communication channel is required to be encrypted. However, it is a great challenge to properly secure the communication when there are many nations involved with unknown security track record. It is crucial to secure the biometric information, especially the fingerprint minutia record. The stolen information can be used for injection or replay attack [Maltoni et al. (2003)]. Worse, Galbally et al. (2010) and Feng and Jain (2011) showed that it is possible to use the stolen minutiae template to recover the fingerprint image, which can then be used to make gummy finger for spoofing attack. Their experiments showed that over 75% of the attempts are successful, even under a high-security operating point. Hence, minutiae information should not be disclosed without strong protection. To solve this, Match-on-Card (MoC) is proposed instead.

Unlike off-card matching which uses the smartcard only as a secure storage media for the fingerprint template, MoC uses the on-card processor to perform fingerprint matching to determine whether the query template is similar to the template stored in it. Therefore, for MoC implementation, the enrolled template does not have to be sent out to the biometric terminal for verification. In fact, to comply with the ISO/IEC 24787 (2010) on-card biometric comparison standard, it is required that there is no software mechanism to download the template(s) stored in the secure storage of the smartcard. Hence, even if the user loses the smartcard, it is very difficult to extract the enrolled template as all data in the secure storage are encrypted. To further enhance the protection of match-on-card operation, Martinez-Diaz et al. (2006) suggested checking the minutiae distribution of the query template, e.g., to check whether the minutiae are only concentrated in certain area, such as the centre. The number of retries is also limited to counter brute-force attack and continuous matching score should not be provided to avoid hill-climbing attack. To further ensure that the data stored inside the smartcard is protected against other hacking means, the smartcard manufacturer adds lots of extra protection mechanisms, both at the circuit and functional block design level to make the reverse engineering process as difficult as possible. As such, Price (2011) mentioned that the current most popular

smartcard is still the 8-bit card. 16-bit and 32-bit cards are available but those cards are much more expensive due to the need for such protections.

MoC using ISO template is still a very challenging operation especially for low-cost smartcard. In the FVC2006 report, it was highlighted that even all the participating algorithms in the light category had computational requirement that was "still far higher for the typical capability of a smartcard or of a low-priced stand-alone device to execute". The National Institute of Standards and Technology (NIST) conducted a MINEX II Interoperability test from 2008 to 2010 and published a report by Grother et al. (2011). There are 12 fingerprint technology companies using 10 card types participating in the interoperability test. However, as mentioned by the report, participants were not required to disclose the type of smartcard (8/16/32-bit) used in the test. In addition, except MoC, there is no other application being executed inside the card. As such, the cost-to-performance of the smartcard cannot be determined. Hence, incorporating fingerprint authentication in the popular low-cost smartcard is still a research challenge. We need to develop an algorithm that is robust to deformation, missing or spurious minutiae, using the ISO template with basic minutiae information, and capable to be executed in the low-cost smartcard within reasonable time.

In this paper, we propose a novel framework of fingerprint matching using in-matcher clustering technique to compare the minutiae template. Deformation is a commonly known problem which affects the accuracy of the matcher. Our focus is on minutiae matching algorithm with ability to handle deformation and to provide orientation invariant matching. The conventional clustering method groups the nearest neighbours of a single set of points together. Our proposed in-matcher clustering scheme groups the matched points with reference to an anchor point between the enrolled template and the query template (2 sets of points). Hence, computation of our in-matcher clustering scheme is different from the conventional clustering schemes. The organization of this paper is as follow: Literature review is presented in Section 2. A novel in-matcher clustering technique and an inter-cluster similarity are introduced in Section 3 while Section 4 details the experiment conducted. Finally, Section 5 concludes the paper together with discussion and future work.

2 Literature Review

Fingerprint authentication can be classified into three categories: image-based, minutiae (point)-based and ridge pattern-based. Among these three categories, minutiae-based matching is the most popular method because of the good accuracy and the compact template. Minutiae matcher is essentially a point-based matching technique with the aid of additional features such as ridge direction and type of minutia to be considered during the matching process. The performance of the minutiae matcher depends on the accuracy of the minutiae detection and the ability of handling deformation due to elastic deformation of the finger. Some past works on MoC will be discussed in Section 4.2.

2.1 Deformation problem

Deformation of fingerprint image is caused by non-orthogonal finger pressure and 3D to 2D mapping during fingerprint acquisition. Such deformation prevents the image acquired to be consistent at all instances, thus affecting the accuracy of fingerprint

verification. Conventional bounding box method used can handle distortion up to certain level by relaxing the matching criteria (Maltoni et al., 2003). However, if there is a large number of minutiae found in a small region or if the deformation is large, this method will fail. He et al. (2006) proposed a global comprehensive similarity-based fingerprint matching algorithm, in which minutia-simplex, including a pair of minutiae as well as their associated textures, were employed to achieve fingerprint matching which can deal with higher distortion level. Cao et al. (2012) introduced a method using ant colony optimization algorithm to establish minutiae correspondences in large-distorted fingerprints. However, such optimization requires extracting many groups of potential corresponding matched minutiae to compute an optimized solution and the overall performance is only average compared to the other algorithms. Nevertheless, it has good result for heterogeneous verification using fingerprint captured from different sensors. Girgis et al. (2009) proposed a hybrid structure consisting of a Genetic Algorithm (GA) and a local search algorithm. The GA requires constructing additional line information from the image for the search of the local similarity. This requires good enhancement technique. Otherwise, the performance will deteriorate. Zhu et al. (2005) proposed using multiple reference minutiae for minutiae matching to solve the deformation problem. This method requires transforming the minutiae template to polar coordinate for each minutia to test for optimal matching and applies polar match globally with additional orientation features for matching. They concluded that using multiple references with different locations can solve certain deformation problem, but the computation is very heavy. Zheng et al. (2007) presented a hybrid method of using minutiae and orientation field to perform matching to deal with distortion. This method also showed slight improvement over minutiae alone. Chen at el. (2006) used local triangle feature set as fuzzy feature set for the fuzzy feature matcher to match fingerprint with large distortion. The local triangle feature is relative differences of distance, ridge directions and slope of minutiae. However, this method requires training using sensor specific images. Chen et al. (2005) proposed an average deformation model based on minutiae locations and orientations using 2-D Thin-Plate Spline (TPS). This average deformation model is applied to the enrolment template prior to matching with the query template. One major disadvantage is that the TPS model relies on correct correspondences of point patterns during deformation estimation. Hence, this method can only enhance the accuracy marginally.

2.2 Correspondence problem

In order to compare minutiae templates, the first step is to find suitable correspondence to align the query template with the enrolled one. The enhanced graph matching approach proposed by Uz et al. (2009) used Delaunay triangulation hierarchies to perform fingerprint matching with template synthesis that combines multiple templates into a single large one. Their approach overcomes certain missing minutiae and spurious minutiae problem and the problem with enhanced graph matching. However, the overall matching complexity is very high due to the computation of Delaunay triangulation. Wei et al. (2006) combines the graph approach with ridge lines for fingerprint matching. However, this approach requires a large memory to build the graph model for matching and is not able to withstand many missing and extra minutiae. Fan et al. (2012) proposed a probabilistic graphical model for fingerprint matching. Similar to the other graph methods, the complexity of the graph model is high which is only suitable for machine with powerful processor for matching. Wen et al. (2009) proposed a fingerprint matching method based on convex hull to exclude spurious minutiae that is matched during

matching process using bipartite matching model. This method is extremely exhaustive. They only showed that their algorithm works well with FVC2002 DB1 which contains relatively cleaner fingerprint image. All the available graph matching approaches are not suitable for smartcard application due to its high computation cost, especially for poor quality image that contains large number of spurious minutiae.

For fingerprint matching, the relative transformation between fingerprints is unknown before matching. Hence, we need to find correspondence between the query template and the enrolled template. Jiang and Yau (2000) proposed a local and global matching. Local information containing information relative to each minutia's nearest neighbors is used to find the best correspondence pair for alignment. Once the query template is aligned with the enrolment template, all coordinates of respective minutiae will be transformed to polar coordinate as global information to match fingerprints. Unlike graph matching, this method only requires processing each minutia with fixed number of nearest neighbors and it is not necessary to edit the graph. Hence, this method is faster than the graph matching approach. However, the local matching still requires cross matching with all the local structures to find the best correspondence for alignment. Poor quality fingerprint affects the consistency of good local structures, and thus affects the matching accuracy. Jea et al. (2005) proposed using localized secondary features derived from relative minutiae information to perform comparison of a partial print against the complete fingerprint. A neural network-based matcher was used to compute the final score. Such method can solve certain partial print problem but the verification time is too slow to be implemented in the resource-constrained platform. Cao et al. (2010) proposed a method to remove the spurious minutiae near the crease and introduced a global feature, namely, finger placement direction as extra information to improve the performance especially with the presence of large distortion in the fingerprint image. Feng. (2008) highlighted that it is difficult to distinguish genuine matches from the imposter matches using only one or two features. The author suggested using a 17-D local feature vectors including texture-based feature vector and minutiae-based feature vector, and formulated the matching as a 2-class problem. Support Vector Machine (SVM) was used to train a classifier to compute a matching score to decide whether a query print is from the imposter or the genuine user. The authors showed that additional features with optimization using machine learning approach can improve the performance of a matcher. Tan et al. (2006) formulated a fitness function based on the local properties of each triplet of minutiae including angles, triangle handedness, triangle direction, maximum side, minutiae density and ridges counts, and then used genetic algorithms to achieve a globally optimized solution with this fitness function. Jain et al. (2006) proposed a matching method using binary tree that is constructed by Nearest Neighbor Vector (NNV) with consideration of k-nearest neighbor. This method uses the core point as the root of the binary tree. Hence, it is very important that the core point can be detected accurately. Otherwise, this method is not able to enroll or verify the fingerprint, especially for those arch and tented arch fingerprints. In general, most of the fingerprint matching methods are not able to match fingerprint in linear time. The matching time increases exponentially with the number of minutiae.

Several researchers have developed minutiae matching algorithms for MoC. Krivec et al (2003) proposed a hybrid fingerprint matcher, which combines minutiae matcher and homogeneity structure matcher to perform authentication on the smartcard. However, this hybrid approach still has limited accuracy. Bistarelli et al (2005) proposed a matching method using local relative information between nearest minutiae. This method achieves

EER at 4.5% on average on FVC2002 DB1 and DB2. Rikin et al (2005) proposed using minutia ridge shape for fingerprint matching. The ridge shape information is used during the minutiae matching to improve the matching accuracy. In their experiment, they showed that the accuracy was comparable with the conventional approaches but the proposed method has higher matching speed. Mimura et al. (2002) and Sanchez-Avila et al. (2003) proposed respective reference implementations of fingerprint MoC. These reference implementations addressed the basic components and requirements of MoC.

To summarize the above literature reviews, we have the following constraints to be solved for fingerprint MoC by various approaches (table 1):

The first two constraints also exist in fingerprint authentication on PC but they are more difficult to solve on smartcard with limited processing power. However, by examining the nature of minutiae template, it is possible to solve the problem by making the performance of matching close to the PC version of fingerprint matching. To deal with the problem of correspondence and deformation, we propose an in-matcher clustering method to find the matched clusters between minutiae templates.

3 In-matcher Clustering for Match-on-card

An effective way to reduce the computational load and increase the accuracy is to examine the fingerprint by using a smaller group (cluster) of minutiae instead of the entire template. This leads us to develop an in-matcher clustering scheme to search for matched clusters of minutiae at different locations between the enrolled template and the query template. However, the minutia cluster may contain false or missing minutiae. This could lead to false acceptance. To solve this problem, we propose another measurement method using Mahalanobis distance to measure the inter-cluster similarity. The structure of the relative clusters in the enrolled template, when compared with the relative cluster in the query template can be used to reject the wrong group. Finally, the overall match score is the combination of the minutia matching (local matching) in cluster and the matching of the geometrical structure between clusters (global matching). A complexity analysis will be given in section 3.4 of our proposed scheme.

3.1 Deformation Analysis and Clustering Schemes

Figure 1 illustrates how deformation affects the matching in a cluster of minutiae. Both fingerprint images are from the FVC2006 DB2a. This diagram is generated by the proposed algorithm to perform in-matcher clustering. Two matched clusters were found between the enrolled template and the query template for illustrating the deformation problem that affects fingerprint matching mentioned in previous section. When the minutia circled in yellow is used as the reference location to perform the matching, only the minutiae within the solid circle can be matched (see “matched impression 1). Similarly when the minutia circled in blue is used instead as the reference location, only the minutiae within the solid ellipse can be matched (see “matched impression 2”). This shows the deformation changes the location of the minutiae unevenly, causing failure to use consistent low tolerance for the whole image. This can be solved by grouping the minutiae into clusters. Another advantage of clustering is that with more minutiae in a

cluster, the likelihood of wrong cluster matching is less compared to a local minutiae structure which comprises 3-4 minutiae (Jiang & Yau, 2000).

3.2 In-Matcher Clustering

Searching for a cluster of matched minutiae between the enrolled template and the query template is a local optimization process to find a group of minutia pairs that are best matched. To reduce the search time, the minutiae list in the enrolled template and query template are sorted using X-Y sort, Y-X sort or polar sort based on the minutiae centre. The details of respective sorting methods to encode minutiae template can be found in ISO/IEC 19794-2. Based on the NIST MINEX II (2013), the performance of polar sort is slightly better than X-Y/Y-X sort. The sorting process is computed during the template generation. The ISO/IEC 19794-2 conformed minutiae template can be formulated as:

$$\mathbf{T} = \{\mathbf{m_1}, \mathbf{m_2}, \dots, \mathbf{m_n}\} \quad (1)$$

$$\mathbf{m_i} = (x_i, y_i, r_i, t_i)^{\mathbf{t}} \quad (2)$$

where **T** is the minutiae template, n is the number of minutiae, x_i and y_i represent the Cartesian coordinate, r_i is the ridge direction and t_i is the minutiae type (not use = 0, termination=1, bifurcation=2, others=3). In this paper, $\mathbf{T^e}$ and $\mathbf{m^e}$ denote the enrolled template and minutia in the enrolled template respectively. Similarly, $\mathbf{T^q}$ and $\mathbf{m^q}$ denote the query template and minutia in the query template respectively.

Figure 2 illustrates a matched pair of minutiae ($\mathbf{m^e_i}$ and $\mathbf{m^q_j}$) and their nearest surrounding neighbors, where i and j are minutia location in the enrolled and query template respectively. Only 5 minutiae in each template are shown to simplify the illustration. Equation 3 and 4 are the relative measurement of minutiae $\mathbf{m_n}$ (which is the neighbor of $\mathbf{m_b}$) with respect to $\mathbf{m_b}$ (which is the base minutiae). The relative information consists of relative minutiae distance (3) and relative absolute ridge direction (4). Equation 5 is used to normalize the angle. Only relative information between the minutiae can be matched directly without alignment, as such information is translation and orientation invariance.

$$\Delta_{Dist}(\mathbf{m_b}, \mathbf{m_n}) = \sqrt{(x_b - x_n)^2 + (y_b - y_n)^2} \quad (3)$$

$$\Delta_{RidgeAng}(\mathbf{m_b}, \mathbf{m_n}) = norm(|r_b - r_n|) \quad (4)$$

$$norm(\theta) = \begin{cases} \theta, if\ 0 \le \theta \le 180^\circ \\ 360^\circ - \theta, if \theta > 180^\circ \end{cases} \quad (5)$$

The orientation invariant minutiae matching using Euclidean distance for a given correspondence index i and index j of enrolled template and query template respectively can be formulated using:

$$localmatchscore(i,j) = \sum_{k=1}^{\omega_1}\sum_{l=1}^{\omega_2}\left(\alpha.compare_{Dist}\left(\Delta_{Dist}(\mathbf{m}_i^{\mathbf{e}},\mathbf{m}_{i+k}^{\mathbf{e}}),\Delta_{Dist}\left(\mathbf{m}_j^{\mathbf{q}},\mathbf{m}_{j+l}^{\mathbf{q}}\right)\right) + \beta.compare_{RidgeAng}\left(\Delta_{RidgeAng}(\mathbf{m}_i^{\mathbf{e}},\mathbf{m}_{i+k}^{\mathbf{e}}),\Delta_{RidgeAng}\left(\mathbf{m}_j^{\mathbf{q}},\mathbf{m}_{j+l}^{\mathbf{q}}\right)\right)\right) \quad (6)$$

$$compare_{Dist}(\delta_1,\delta_2) = \begin{cases} \frac{\delta_1}{\delta_2}, if\left(\delta_1 \le \delta_2 \text{ and } \frac{\delta_1}{\delta_2} > Threshold_{dist}\right) \\ \frac{\delta_2}{\delta_1}, if\left(\delta_1 > \delta_2 \text{ and } \frac{\delta_2}{\delta_1} > Threshold_{dist}\right) \\ 0, \text{otherwise} \end{cases} \quad (7)$$

$$compare_{RidgeAng}(\varphi_1,\varphi_2) = \begin{cases} \frac{\varphi_1}{\varphi_2}, if\left(\varphi_1 \le \varphi_2 \text{ and } \frac{\varphi_1}{\varphi_2} > Threshold_{RidgeAng}\right) \\ \frac{\varphi_2}{\varphi_1}, if\left(\varphi_1 > \varphi_2 \text{ and } \frac{\varphi_2}{\varphi_1} > Threshold_{RidgeAng}\right) \\ 0, \text{otherwise} \end{cases} \quad (8)$$

where $\mathbf{m}_i^{\mathbf{e}}$ and $\mathbf{m}_j^{\mathbf{q}}$ are minutiae in enrolled template and query template respectively, i and j are minutiae index of enrolled template and query template respectively. The index k and l are used to search both templates within the window size of ω_1 and ω_2. Equation (6) computes the local similarity between the enrolled template and query template at minutiae index i and j respectively using the linear weighted sum to fuse the $compare_{Dist}(\cdot)$ function and the $compare_{RidgeAng}(\cdot)$ together to obtain the local similarity score. $\mathbf{m}_i^{\mathbf{e}}$ and $\mathbf{m}_j^{\mathbf{q}}$ are used as reference bases to compute the relative minutiae distance and ridge direction with the respective nearest neighbors within the window size of ω. Equation (7) is used to compare the relative minutiae distance between the enrolled template and the query template. A ratio of relative distances is being used to measure the similarity quantitatively within the range of (0,1], where close to 0 means dissimilar and 1 means exact similarity. Similarly, equation (8) compares the relative ridge angles between the enrolled template and the query template. The $Threshold_{dist}$ and $Threshold_{RidgeAng}$ with the range (0,1] are used to filter out those obvious dissimilar minutiae pairs. For example, both $Threshold_{dist}$ and $Threshold_{RidgeAng}$ can be set to 0.85 to accept pairs with similarity over 85% as potential candidates, otherwise, a zero will be returned. Moreover, our objective is to find a cluster of matched minutiae with low matching error as described in figure 1. Therefore, both $Threshold_{dist}$ and $Threshold_{RidgeAng}$ can be set to be more restrictive and a smaller window ω_1 and ω_1 can be used. The easiest way to solve equation (6) is to find the maximum local match score, which is the biggest cluster of matched minutiae pairs, by cross matching all the minutiae in both templates. If m and n are the number of minutiae in enrolled template and query template respectively and ω_1 and ω_2 are set to m and n respectively, the total number of iterations to completely cross match both enrolled template and query template using equation (6) is $m \cdot n \cdot (m\text{-}1) \cdot (n\text{-}1)$. For example, if both enrolled and query template contains 40 minutiae, the number of iterations is 2433600. The search dimension of this

method is really large which is only suitable to be used to match fingerprint with small database even using a PC. As such, it is not possible to be used on a smartcard. As all minutiae are sorted in both enrolled template and query template, the nearest minutiae neighbors are approximately known. To simplify the computation of equation (6), a smaller window ω_1 and ω_2 can be used to speed up the search. Instead of summing the similarity of minutiae matched pair to compute the overall local similarity score, a similarity matrix can be defined by re-writing equation (6) as equation (9) to compute the respective elements inside the similarity matrix $\boldsymbol{\xi}_{i,j}$:

$$\boldsymbol{\xi}_{i,j}(k,l) = \alpha \cdot compare_{Dist}\left(\Delta_{Dist}(\mathbf{m}_i^{\mathbf{e}}, \mathbf{m}_{i+k}^{\mathbf{e}}), \Delta_{Dist}\left(\mathbf{m}_j^{\mathbf{q}}, \mathbf{m}_{j+l}^{\mathbf{q}}\right)\right) + \beta \cdot compare_{RidgeAng}\left(\Delta_{RidgeAng}(\mathbf{m}_i^{\mathbf{e}}, \mathbf{m}_{i+k}^{\mathbf{e}}), \Delta_{RidgeAng}\left(\mathbf{m}_j^{\mathbf{q}}, \mathbf{m}_{j+l}^{\mathbf{q}}\right)\right) \quad (9)$$

where k =1..ω_1 and l =1..ω_2. The window parameters ω_1 and ω_2 are set to be the same size W_s which also controls the size of matrix $\boldsymbol{\xi}_{i,j}$. Equation (9) computes the elements inside the similarity matrix $\boldsymbol{\xi}_{i,j}$ to describe the local similarity of particular minutiae location at index i in enrolled template and index j in query template. Each column inside the similarity matrix $\boldsymbol{\xi}_{i,j}$ can be treated as a vector. For each vector, it should contain at most 1 matched minutiae, as minutiae matching is always 1-to-1 matching. All the matched minutiae appear in the similarity matrix $\boldsymbol{\xi}_{i,j}$ as local maxima along the row and column. The thresholds act as the constraint for clustering. All the local maxima within the similarity matrix $\boldsymbol{\xi}_{i,j}$ are cluster of matched minutiae. To get all the matched minutiae, a maximum filter is applied to rows and columns of the matrix $\boldsymbol{\xi}_{i,j}$. A filtered similarity matrix $\boldsymbol{\xi}'_{i,j}$ is computed using the following steps:

1) For each row, find the element with the maximum value.
2) Except the element with the maximum value, all the other elements along the row are set to zeros.
3) Repeat for each row until the end of matrix
4) For each column, find the element with maximum value.
5) Except the element with the maximum value, all the other values along the column are set to zeros.
6) Repeat for each column until the end of matrix.

Once the filtering process is completed, all the non-zero elements inside the filtered similarity matrix $\boldsymbol{\xi}'_{i,j}$ represent the matched minutiae within the cluster. For each non-zero element, the corresponding row index and column index represent the index of enrolled template and the index of query template respectively, which is the matched-pair of minutiae within the cluster.

Hence, for each column vector, the maximum value will be selected to form a group similarity vector $\mathbf{v}$ as the following equation (10).

$$\mathbf{v}(l) = \max\left(\boldsymbol{\xi}'_{i,j}(k,l)\right), k = 1..\omega_1 \quad (10)$$

Once the group similarity vector $\mathbf{v}$ is computed, decision is made using this vector to decide whether the location is a matched location or not. For matched cluster (i.e. the query is from the genuine user), the vector $\mathbf{v}$ should contain some elements with the

magnitude larger than $Threshold_{match}$ (e.g. $\mathbf{v}$ should contains at least 1/3 of ω_l to be considered as matched cluster). Thus, the local similarity score ς is computed using the following equation:

$$ls(l) = \begin{cases} \mathbf{v}(l), \text{if } \mathbf{v}(l) > Threshold_{match} \\ 0, otherwise \end{cases} \quad (11)$$

$$\varsigma_p = \sum_{l=1}^{\omega_1} ls(l)/\omega_1 \quad (12)$$

During the computation of equation (11), we count the number of non-zero elements which represent the number of matched minutiae within the cluster. Equation (12) computes the score ς that can be used as a confidence level on whether the two clusters match or not. The major advantage of this method is that the windows size ω is small.

The equations that compute the similarity of given minutiae in enrolled template and query template are used to scan the two templates using a window size W_s. The following pseudo code describes the scanning process to search for potential location for matching.

The pseudo code in figure 3 describes the in-matcher clustering process. As all minutiae in both templates are sorted, the nearby minutiae may be the nearest neighbors of each minutia at particular location. Initially, the search process starts from the beginning of both minutiae templates where both *Init_i* and *Init_j* are set to 1, p=1 and $\eta_{matched}$ = 0, where $\eta_{matched}$ is the number of matched clusters. For every combination of i and j, the matrix $\boldsymbol{\xi}_{i,j}$**,** the group similarity vector $\mathbf{v}$**,** and the local similarity score ς_p will be computed. If the local similarity score ς_p is less than the $Threshold_{cluster,}$ it implies the current location is unlikely to be a good location for matching, the process will continue to search for the next pair of i and j by advancing the indexes in the for loops. Once a matched cluster is found, the indexes of non-zero elements in $\boldsymbol{\xi}'_{i,j}$ will be stored into the cluster of matched minutiae list $\mathbf{C}_p$ and $\mathbf{C'}_p$ from enrolled template and query template respectively and both p and $\eta_{matched}$ will be increased by 1. This function will be called multiple times until either *Init_i* or *Init_j* reach the maximum number of minutiae in enrolled template and query template respectively.

Figure 4 illustrates how *Init_i* and *Init_j* are updated. At the beginning, both *Init_i* and *Init_j* are set to 1. Once the first cluster of matched minutiae $\mathbf{C}_1$ is found, the subsequent search for matched clusters can be accelerated by selecting those matched minutiae near the boundary as the next initial condition (*Init_i* and *Init_j*) for the next in-matcher clustering search. In figure 4, *Init_i*=7 and *Init_j*=7 are updated in the 2nd iteration of the cluster search for $\mathbf{C}_2$. Hence, those matched minutiae in the first iteration can be skipped. In order to further speed up the search for the next cluster, within the cluster of matched minutiae list $\mathbf{C}_p$ and $\mathbf{C'}_p$, we select the matched minutiae pair with maximum x-coordinate (far right hand side in the cluster) or maximum y-coordinate (bottom of the cluster) in both the enrolled and query template as the initial condition (*Init_i* and *Init_j*) for the next cluster search using the same function. Once all minutiae has been matched in both templates (*Init_i* or *Init_j* reach the maximum number of enrolled template or query template respectively), the matching process is terminated. If the number of matched clusters is non-zero, the next inter-cluster matching process will be executed. Otherwise, a zero score will be returned as there is no matched cluster. Note that some

matched minutiae pairs may be included in multiple clusters. In addition, the clusters may overlap.

3.3 Inter-cluster matching and overall similarity

The above in-matcher clustering obtains the matched clusters of minutiae between the enrolled template and the query template. Ideally speaking, no matched cluster should be found between the genuine user and the imposter. However, the minutia cluster may have false or missing minutiae which cause some similarity between templates from different persons. Such similarity may generate false acceptance. In order to solve this problem, we propose another measurement method using Mahalanobis distance to measure the inter-cluster similarity. For genuine user, if multiple clusters can be found (usually 3~6 clusters, depends on the size of the fingerprint), both enrolled template and query template should have similar inter-cluster structure. For imposter, even though multiple matched clusters can be found (usually less than 3 but occasionally more than 5 for noisy fingerprints), the structure between the clusters of the query template is likely to be different from that of the enrolled template. The above numbers are based on the experiments mentioned in section 4. Therefore, we propose to use Mahalanobis distances between clusters to measure the inter-cluster structures of the both templates.

Let $\mathbf{C}_p$ and $\mathbf{C}_q$ contain the Cartesian coordinates of matched minutiae within the clusters p and q in the enrolled template respectively, and $\mathbf{C'}_p$ and $\mathbf{C'}_q$ contains those in the query template, respectively. To measure the inter-cluster Mahalanobis distance γ_{pq} between $\mathbf{C}_p$ and $\mathbf{C}_q$ in the enrolled template, the following equation is used:

$$\gamma_{pq} = \sqrt{(\overline{\mathbf{C}_p} - \overline{\mathbf{C}_q})^T \mathbf{S}^{-1} (\overline{\mathbf{C}_p} - \overline{\mathbf{C}_q})} \tag{13}$$

where $\mathbf{S}$ is the pooled covariance matrix of $\mathbf{C}_p$ and $\mathbf{C}_q$, $\overline{\mathbf{C}_p}$ and $\overline{\mathbf{C}_q}$ are the means of the minutia locations in cluster $\mathbf{C}_p$ and cluster $\mathbf{C}_q$, respectively. Similarly, the inter-cluster Mahalanobis distance γ'_{pq} between clusters $\mathbf{C'}_p$ and $\mathbf{C'}_q$ in query template can be calculated using equation (13) as well. The inter-cluster similarity ψ_{pq} is given by:

$$\psi_{pq} = \begin{cases} 1 - \frac{|\gamma_{pq} - \gamma'_{pq}|}{\max(\gamma_{pq}, \gamma'_{pq})}, p \neq q \\ 0, p = q \end{cases} \tag{14}$$

The inter-cluster similarity is the measurement of similarity of the above distances between the enrolled template and the query template. If both p and q are the same (i.e. same cluster), the value of ψ_{pq} will be zero to ignore the measurement. The range of inter-cluster similarity ψ_{pq} is from zero (dissimilar) to 1 (identical). In practice, more than 2 clusters can be found using the in-matcher clustering technique introduced in the previous section. Once the inter-cluster similarity is computed, the following equation (15) can be used to compute the relationship between clusters.

$$\beta_p = \frac{\sum_{q=1}^{\eta_{matched}} \psi_{pq}}{\eta_{matched} - 1} \tag{15}$$

where $\eta_{matched}$ is the number of matched clusters computed in the previous section. The summation process in equation (15) sums from 1 to the total number of clusters found. The sum $\beta_p$indicates the similarity of clusters *p* between the enrolled template and the query template in terms of the position of the cluster *p* to other clusters. If the value of $\beta_p$is too small (e.g. smaller than 0.5), the particular cluster may be wrongly matched which will be removed from the cluster list. If the processing power of MCU is not sufficient for certain low-cost smartcard, it is possible to use Euclidian distance between cluster centers to replace with the Mahalanobis distance with lower accuracy.

As deformation may affect the distance between clusters, a more lenient tolerance compared to those thresholds can be used by the in-matcher clustering. We can use the inter-cluster similarity to reject those wrongly matched clusters by using the following pseudo code (figure 5):

α_p is the parameter to decide whether the particular cluster is accepted or not. Within the process, if $\beta_p > Threshold_{Inter\text{-}cluster}$, the number of overall matched minutiae $N_{matched}$ can be counted. As the matched minutiae among clusters may be overlapped, those counted minutiae in the previous cluster are excluded in the subsequence clusters to ensure that the overlapped minutiae among clusters are counted only once. Once the process is completed and the $N_{matched}$ is known, the final decision score is obtained by:

$$S = \left(\mu \cdot \frac{N_{matched}}{(N_{erollment} + N_{query})/2} + \nu \cdot \frac{\sum_{p=1}^{\eta_{matched}} \alpha_p \beta_p \zeta_p}{\eta_{matched}} \right) \cdot 100 \qquad (16)$$

where S ($0 \le S \le 100$) is the overall matching score between enrolled template and query template, μ and ν are weights to compute the weighted sum, and $N_{enrollment}$ and N_{query} are the number of minutiae in enrolled template and query template, respectively. Equation (16) can be divided into 2 portions. The first portion in the equation indicates the confidence level of matching using the number of matched minutiae between templates, whereas the second portion of the equation represents the pattern-based similarity using inter-cluster similarity.

3.4 Complexity Analysis

For our in-matcher clustering scheme, the heaviest part is to compute equation (9) with the pseudo-code listed in figure 3. The lengthiest part is to search for the first matched cluster. Let n_e and n_q be the number of minutiae of the enrolled template and the query template respectively. The windows size ω_1 *and* ω_2 are set to the same size W_s. The number of iterations to perform a full search of equation (9) is $n_e \cdot n_q \cdot (W_s^2)$. However, if the query is from the genuine user, the search space is usually less than 1/3 of the minutiae based on the experiments conducted in section 4 using various databases. Hence, the number of searches for the first cluster is approximately $O((n_e/3) \cdot (n_q/3) \cdot (W_s^2))$. For subsequent cluster search, the speed is much higher as the approximate correspondence is already known. Hence, the search space is approximately 1/8 of minutiae based on the experiments conducted in section 4. Thus, the complexity of in-matcher clustering for genuine user is approximately $O((n_e/3) \cdot (n_q/3) \cdot (W_s^2) + (\eta_{matched}$-

1)·(n_e/8)·(n_q/8)·(W_s^2)) = ($\eta_{matched}$+6) · (W_s^2/64) ·O(n_e·n_q), where $\eta_{matched}$ is the number of matched cluster found during the in-matcher clustering process, usually 2~3 clusters can be found. The window size W_s is often set to 5~7 points.

For imposter matching, we assume that no matched cluster or just few small clusters $\eta'_{matched}$ are found. In each round of scanning, approximately 2/3 of the total minutiae are involved. Thus on average, the complexity can be estimated as O(($\eta'_{matched}$)·(2/3)·n_e·(2/3)·n_q·(W_s^2)). Hence, the searching process of imposter match is relatively longer than genuine user match. In general, the complexity of our in-matcher cluster scheme is acceptable for 8-bit smartcard MCU.

Comparing with the method of Jiang and Yau (2000) which is the fastest algorithm reported in FVC2000, if the number of nearest neighbors is set to 3, their local matching requires O(n_e·n_q·3·3). Once the correspondence is found, the matching in polar coordinates is O(n_e·n_q). The overall complexity is around O(n_e·n_q·3·3) + O(n_e·n_q) = 10·O(n_e·n_q). The ratio of the proposed algorithm divided by their method is ($\eta_{matched}$+6)·(W_s^2/640). The key advantages of the new algorithm are:

1) Does not need to perform a full search for each round in order to compute a matched cluster.
2) Able to skip the matched minutiae during matching especially in the case that the query is from genuine user.
3) Additional global matching in polar coordinate is not necessary.

3.5 Implementation on Smartcard

The advantages of our proposed algorithm are orientation invariant and able to better withstand deformation. In terms of computation, the major advantages of our proposed algorithm are that the only transcendental function used is square root . To speed up the search, we use dual windows for the search as given by the pseudo code in figure 3. In figure 3, the longest part is to calculate the matrix $\boldsymbol{\xi}_{i,j}$. For example, we can set a smaller window W_s (e.g. 6 neighbors) to do a preliminary check. If the number of matched minutiae is more than 2, we can expand the size of the windows at the same location for further search to get a bigger matched cluster. Moreover, once a matched cluster is found, the approximate correspondence is known. Hence, the subsequent cluster matching will be much faster. As all the equations use only simple arithmetic operations, all the computations required can be done using 8/16-bit integer computation with scaling factor. For computation of equation (9), (13), (14), (15) & (16) in smartcard, it is necessary to do a scaling before arithmetic operations to avoid using floating point computation. As a result, the proposed algorithm can be implemented using integer arithmetic. For the square root computation, as the amount of flash memory of smartcard is usually large, look-up table approach is used instead.

4 Experiment

We have implemented a PC version and a smartcard version of the proposed algorithm in C language. The smartcard consists of an 8-bit MCU (using 8051 instruction set) running at 25 MHz, 6k bytes static RAM and 78k bytes flash memory/EEPROM. The compiled code size is around 16k bytes, occupying only about 20% of the available flash memory.

As such the card still has sufficient free flash memory to install other applications. The smartcard supports both contact and contactless connection to the smartcard reader. The on-card implementation with 8-bit MCU and the PC version using the proposed algorithm was submitted to NIST MINEX II Interoperability Test (only participated in Phase IV). We will present and discuss the results in the later part of this section.

In this paper, Fingerprint Verification Competition (FVC) databases are used for benchmarking the proposed algorithm. All databases, including FVC2000 (2000), FVC2002 (2002), FVC2004 (2004) and FVC2006 (2006) can be obtained from the respective FVC websites. We follow the protocol from FVC2000 for all databases for benchmarking. Hence, for FVC2006, we choose the first 100 users and the first 8 impressions for each user to perform benchmarking. Moreover, in FVC2006, DB1 is not used as the image is too small, such that the NIST minutiae detector failed to detect the minutiae for quite a number of fingerprint images.

4.1 Benchmark using NIST Minutiae Detector

We used the open source Fingerprint SDK from National Institute of Standards Technology (NIST) to generate the minutiae template. MINDTCT is used to detect minutiae to generate minutiae template for all databases in our experiment. The reason for using the open source NIST detector is because the generated template files can be easily converted to ISO/IEC 19794-2 (2005) compact size or normal format. Although the performance of this extractor is not very well with quite a number of false minutiae, it can serve as a basis for performance benchmarking of the matcher. As MINDTCT can only accept the image in wavelet format or RAW format, the fingerprint images in other formats are converted to wavelet format with compression ratio of 5:1. Then, the template which conformed to ISO/IEC 19794-2 (2005) is constructed from the output of MINDTCT. All minutiae with quality less than or equal to 10 were rejected. X-Y sort (sort the order using the x-axis first in ascending order, and then sort the order using the y-axis in ascending order) is used to re-arrange the sequence of minutiae as per specification in ISO/IEC19794-2 (2005). No pre-processing is applied for all images.

All minutiae templates are generated using PC, whereas the proposed minutiae matching using in-matcher clustering algorithm is performed by both smartcard and PC. It is very slow to conduct the entire experiment using smartcard. For each database, we selectively chose the first 100 matches for genuine users and 100 matches for imposters to verify both smartcard score and PC score to ensure that both scores are identical. NIST also conducted similar test to check the scores between PC and smartcard to see whether both scores are identical, as we submitted both executable for PC and smartcard to NIST for benchmarking. More details about the NIST benchmarking will be described in section 4.2. The following figure (Figure 6 to 9) shows the Receiver Operating Curves (ROC) generated during the experiment.

We compare our performance obtained in Table 2 with participants in the respective FVC competitions. For FVC2000, our proposed algorithm can achieve accuracy similar to the top two on average, except for DB3 where we can only achieve 5^{th} place. DB3 is an

optical fingerprint sensor with large capturing area. The generator falsely detects quite a number of minutiae in the blank area and dirty area, thus causing degradation in the performance. For FVC2002, we can achieve rank 14, 9, 14 and 12 of DB1, DB2, DB3 and DB4 respectively, out of 31 participants. For FVC2004, we can achieve rank 9, 12, 9 and 13 of DB1, DB2, DB3 and DB4 respectively out of 26 participants in the light category. For FVC2006, we can achieve 15, 17 and 20 out of 24 participants in the light category. Even in the open category, our light computation algorithm can also achieve around the mid position in FVC2004 and FVC2006 compared with those matching algorithms requiring much larger computational power. The overall average EER of all databases listed in Table 2 is 5.1979%. For the above result, the proposed algorithm is comparable to the average performance of the fingerprint matcher for PC but using much lesser resources and template size.

For FVC2000 DB3, FVC2004 DB1 and FVC 2006 DB4, the performance is not satisfactory. We found that the reason is mainly because of the large number of false minutiae detected by MINDTCT, especially for those noisy fingerprint images. Figure 10 shows some samples of the minutiae detected by MINDTCT. The engine sometimes detects minutiae outside the fingerprint and at the edge of fingerprint. Some of those false minutiae have good minutiae quality score, thus they cannot be removed using quality score alone. We tried to adjust the quality threshold generated by MINDTCT to reject bad minutiae for benchmarking. However, we found that the threshold of minutiae quality of 10 is the optimal value to reject bad minutiae without causing too many false rejections of valid minutiae. Any higher value will cause even higher performance reduction.

Please note that while we only use the ISO 19794-2 compliant template information with a common minutiae extractor, the other FVC participants are using other more complex extraction methods which will provide more accurate detection. The participants are also not limited to ISO 19794-2 compliant template, allowing them to use additional information to improve the matching speed and accuracy. This can be seen by the model size and matching memory size in FVC2004 and FVC2006. For example, in FVC2004 DB1, the average model size ranges from 0.5k bytes to 2k bytes for the light category and 0.5k bytes to 64k bytes (excluding those very large size (>256k bytes) and where EER>20%) for open category. Do note that the maximum match memory size is from 1M bytes to 4M bytes of light category and 1M to 42M for open category. For our implementation, the memory usage of the proposed module on smartcard is approximately 2k bytes at runtime by checking the memory map output from the compiler and the size of the executable is around 7k bytes.

4.2 Submission to NIST MINEX II Phase IV Interoperability Test

We submitted a prototype 8-bit smartcard (8051 instruction) with 6k bytes RAM and 78k EEPROM running at 25 MHz to the NIST MINEX IV Performance of Fingerprint MoC Algorithms test in 2010 and a performance test report written by Grother et al. (2011)

was published on 31st March 2011 by NIST. This was our only submission as Phase IV was the last round of MINEX II. Due to the constraints of the physical limitation of the smartcard from our card provider in 2010, our submitted MoC engine can only support maximum 60 minutiae. Based on the MINEX II testing protocol, if the number of minutiae, *n*, is more than the supported number of minutiae (in our case, *n*=60) , the minutiae quality score from respective detector's engine will be used to remove those low quality minutiae in order to reduce the size to *n* minutiae for enrolment or matching. We evaluated our algorithm with other minutiae detectors submitted from different vendors for interoperability. The number of genuine and imposter comparisons was 247924 and 2499880 respectively for single user test. For two-finger matching, the number of genuine and impostor comparisons was 123962 and 1249940. The details of protocol for single finger verification and two-finger verification can be found in the report by Grother et al. (2011).

Figure 11 to 14 and Table 2 show the performance (using FVC databases) of the MoC algorithm submitted to NIST MINEX II Phase IV where there is restriction of maximum 60 minutiae due to the memory and processing limitation of smartcard from our supplier. The average EER of the submitted algorithm using FVC databases is 6.12% which is ~1% higher compared to the proposed algorithm. Thus the proposed method provides an improvement over the earlier algorithm with clustering, but without inter-cluster analysis.

Based on the Phase IV report, our earlier submitted algorithm (labeled as MX2-IV-Q) achieved False Match Rate (FMR) = 0.001, False Non-Match Rate (FNMR) = 0.08 (average across minutiae detectors of different vendors) and average on-card verification time of 1.01s. According to the report, our submitted algorithm (and couple of other participants) comes reasonable close to attaining the Personal Identity Verification (PIV) compliance specified by NIST. When FMR=0.01, the FNMR exceeded the required 0.01 limit with small margin for some difficult minutiae detectors. The report explained that such behavior is partly because of the differing relative sizes of the templates from different detectors, which is a known interoperability issue. As some minutiae extractors are more verbose (more false minutiae) than the others, the minutiae quality indicator alone may not be a sufficient criteria to correctly reduce the template size, thus affecting the accuracy, especially for those participants who cannot support large number of minutiae to be verified on-card. The Phase IV report also specifies that the number of minutiae supported by our submitted card is the lowest among the other participants in Phase IV.

We have conducted a full scale matching test using FVC2000 DB2 with our algorithm running on an 8-bit 25MHz smartcard. For the algorithm submitted to NIST, the average matching time for genuine user is around 1.27s and the average matching time for imposter is around 3.49s. For the proposed algorithm mentioned in this paper with inter-cluster analysis, the average matching time for genuine is around 1.33s and the average matching time for imposter is around 3.6s. The proposed algorithm is around 0.1~0.2s longer than without inter-cluster analysis, but the new algorithm can provide ~1% lower EER.

As such, the proposed algorithm performs better compared to the earlier submission to NIST for interoperability test. In the past, there are also some MoC implementations done

by various researchers, such as Krivec et al. (2003), Rikin et al. (2005), Allah (2005), Mimura et al. (2002), Sanchez-Reillo et al. (2003), Sanchez-Reillo and Sanchez-Avila (2002) and Bistarelli et al. (2005). Most of their implementations are using their own small database for benchmarking and the performances are only moderate. Bistarelli et al used FVC2002 (DB1 and DB2) for benchmarking but they achieved EER 4.6% on average, whereas our latest engine can achieve average EER=1.58% using FVC2002 DB1 and DB2. In the NIST interoperability test, there are quite a number of commercial submissions with good result but they did not reveal their actual implementation about their algorithm. Moreover, as mentioned in the report, NIST did not request the information of the card's capabilities from participants and they assumed that the participants will generally use the latest or most capable cards (e.g. 32-bit processor or co-processor) in their submission to the NIST evaluation.

Based on the FVC test, the result shows that the accuracy is comparable with those fingerprint verification for PC despite the limited information used in the template as well as the limited number of minutiae used for matching and the low computational complexity. However, the performance is still affected by the false minutiae detected, especially if the minutiae extractor did not perform well in low quality images. We have conducted a separate test using our own proprietary minutiae detector to test FVC2000 DB3, FVC2004 DB1 and FVC2006 DB4 with the following results:

Based on the above table, we can get 1.3~5.73% improvement compared to our submitted engine with the NIST MINDTCT detector. Hence, although our implementation can handle orientation and deformation, it is still necessary to have good minutiae detector or good fingerprint sensor to avoid too many false minutiae in order to achieve good accuracy. For example, as FVC2006 DB2 contains clear images in general, our implementation with MINDTCT can achieve EER of 0.879893% (note that the settings of the MINDTCT remain the same throughout the experiment for all databases).

5 Conclusion

A fingerprint minutiae matching algorithm working on the low-cost smartcard using basic minutiae information has been presented in this paper which is able to solve the constraints mentioned in section 2: limited minutiae information, inconsistent minutiae detection, and limited resources of smartcard. As the elasticity of skin causes deformation of captured fingerprint image during the image acquisition process, the proposed algorithm uses the novel in-matcher clustering technique to find matched clusters of minutiae with the least deformation error. Once the first matched cluster is found, the subsequent search will be fast as the approximate correspondence is known, providing an initial condition to limit the subsequent search. Moreover, this new method can exploit the sorted format of minutiae arrangement as proposed in the ISO/IEC 24787 to speed up the initial search to find the first matched cluster. Once all matched clusters are found, the Mahalanobis distances between a cluster to all the other clusters are used to compute the inter-cluster distances to generate the inter-cluster similarity between the query template and the enrolled template. They are used to reject the false clusters and are incorporated into the overall matching score to enhance the matching accuracy. The overall matching score is computed by combining the confidence level of the cluster matching and that of the inter-cluster similarity score. The proposed algorithm was evaluated on FVC2000, FVC2002, FVC2004 and FVC2006 databases with the NIST's MINDTCT minutiae

detector for benchmarking. The overall EER of all FVC databases is 5.1979%. Although the accuracy performance is only above average, our proposed matching framework requires much lower processing load, allowing it to be easily implemented on a low-end smartcard. In addition, the matching algorithm uses only the standard compliant template information while the false minutiae detected by MINDTCT adversely affects the matching accuracy as our smartcard implementation can only accept 60 minutiae due to the physical constraint of our smartcard. The initial smartcard version of the proposed algorithm has been submitted to NIST MINEX II Phase IV to benchmark the interoperability. Our submitted algorithm achieves FMR = 0.001, FNMR = 0.08 (average) and average on-card verification time ~1.01s with maximum of 60 minutiae. Based on the benchmarking on the FVC databases, our proposed algorithm presented in this paper achieves at least an improvement of 1% EER with lesser computing resources over that submitted to NIST MINEX II Phase IV.

In order to achieve better performance and interoperability, performance of the minutiae extractor is also crucial as the presence of large number of false minutiae will degrade the matching performance. A useful way to reduce the template size is to only consider high quality minutiae. This requires a reliable quality measurement of the minutiae. Our future research direction is to find a quantitative descriptor to represent the quality of the minutiae and to incorporate the quality of the detected minutiae as part of the matching parameter to allow the matcher to select the minutiae for matching or to properly generate the matching score based on the quality-weighted minutiae.

References

Jain, A.K., Prabhakarb, S. and Pankantic, S. (2002) 'On the similarity of identical twin fingerprints', Pattern Recognition, Vol. 35, pp. 2653–2663.

Jain, A.K. and Pankanti S. (2009) 'Fingerprint Recognition', The Essential Guide to Image Processing, in Bovik, A.C. (Eds.), Academic Press, p. 649-676.

Schouten, B. and Jacobs B., 'Biometrics and their use in e-passports', (2009) Image and Vision Computing, Vol. 27, pp. 305–312.

Maltoni, D., Maio, D., Jain, A.K. and Prabhakar, S. (2003) Handbook of Fingerprint Recognition, Springer, New York, USA.

Galbally, J., Cappelli, R., Lumini , A., Gonzalez-de-Rivera , G., Maltoni , D., Fierrez , J., Ortega-Garcia, J. and Maio , D. (2010) 'An evaluation of direct attacks using fake fingers generated from ISO templates, Pattern Recognition Letters, Vol. 31, pp725–732.

Feng . J. and Jain, A.K. (2011) 'Fingerprint Reconstruction: From Minutiae to Phase', IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 33, Iss. 2, pp.209-223.

International Organization for Standardization and International Electrotechnical Commission (2010) ISO/IEC 24787:2010: Information technology — Identification cards — On-card biometric comparison. Geneva, ISO/IEC.

Martinez-Diaz , M., Fierrez-Aguilar, J., Alonso-Fernandez, F., Ortega-Garcia, J. and Siguenza , J.A. (2006) 'Hill-Climbing and Brute-Force Attacks on Biometric Systems: A Case Study in Match-on-Card Fingerprint Verification', Proc. IEEE of International Carnahan Conf. on Security Technology, pp151-159.

NIST MINEXII (2013)- An assessment of match-on-card technology. [online] http://www.nist.gov/itl/iad/ig/minexii.cfm (Accessed 1 August 2013).

NIST Biometric Image Software. [online] http://www.nist.gov/itl/iad/ig/nbis.cfm (Accessed 1 August 2013).

He, Y., Tian, J., Li, L., Chen , H. and Yang, X. (2006) 'Fingerprint matching based on global comprehensive similarity', IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 28, Iss. 6, pp. 850–862.

Cao, K., Yang, X., Chen, X., Zang , Y., Liang , J. and Tian , J. (2012) 'A novel ant colony optimization algorithm for large-distorted fingerprint matching', Pattern Recognition, Vol. 45, pp. 151–161.

Girgis , M.R., Sewisy , A.A. and Mansour , R.F. (2009) 'A robust method for partial deformed fingerprints verification using genetic algorithm', Expert Systems with Applications, Vol. 36, pp. 2008–2016.

Zhu, E., Yin, J. and Zhang, G. (2005) 'Fingerprint matching based on global alignment of multiple reference minutiae', Pattern Recognition, Vol.38, pp. 1685 – 1694.

Zheng, X., Wang, Y. and Zhao, X. (2007) 'A robust matching method for distorted fingerprints', Int. Conf. Image Processing, pp. 377–380.

Chen, X., Tian , J., Yang, X. and Zhang , Y. (2006) 'An algorithm for distorted fingerprint matching based on local triangle feature set, IEEE Transactions on Information Forensics and Security, Vol. 1, Iss. 2, pp. 169-177.

Chen, Y., Dass, S., Ross, A. and Jain, A.K. (2005) 'Fingerprint Deformation Models Using Minutiae Locations and Orientations', Proc. IEEE Workshop on Applications of Computer Vision, pp 1-6.

Uz. T., Bebis , G., Erol , A. and Prabhakar , S. (2009) 'Minutiae-based template synthesis and matching for fingerprint authentication', Computer Vision and Image Understanding, Vol. 113, pp. 979–992.

Wei, H., Ou, Z. and Zhang , J. (2006) 'Fingerprint Identification Based on Ridge Lines and Graph Matching', Proc. the 6th World Congress on Intelligent Control and Automation, pp. 9965 – 9968.

Fan, D., Yu , P., Du , P., Li, W. and Cao, X. (2012) 'A Novel Probabilistic Model Based Fingerprint Recognition Algorithm', International Workshop on Information and Electronics Engineering, pp 201 – 206.

Wen, C. and Guo, T. (2009) 'An efficient algorithm for fingerprint matching based on convex hulls', International Conference on Computational Intelligence and Natural Computing, pp 66-69.

Jiang, X. and Yau, W.Y. (2000) 'Fingerprint minutiae matching based on the local and global structures', Proc. Int. Conf. Pattern Recog., Vol. 2, pp. 1042-1045.

Jea , T.Y. and Govindaraju , V. (2005) 'A minutia-based partial fingerprint recognition system', Pattern Recognition, Vol. 38, pp. 1672 – 1684.

Cao K., Yang , X., Tao , X., Li, P., Zang, Y. and Tian, J. (2010) 'Combining features for distorted fingerprint matching', Journal of Network and Computer Applications, Vol. 33, pp. 258–267.

Feng J. (2008) 'Combining minutiae descriptors for fingerprint matching', Pattern Recognition, Vol. 41, pp. 342 – 352.

Tan. X. and Bhanu , B. (2006) 'Fingerprint matching by genetic algorithms', Pattern Recognition, Vol. 39, pp. 465 – 477.

Wei, H., Ou, Z. and Zhang, J. (2006) 'Fingerprint Identification Based on Ridge Lines and Graph Matching', Proceedings of the 6th World Congress on Intelligent Control and Automation, pp. 9965-9968.

Jain , M.D., Pradeep , S.N., Prakash , C. and Raman , B. (2006) 'Binary Tree Based Linear Time Fingerprint Matching', Proc. IEEE Int. Conf. Image Processing, pp309 – 312.

Pal. N.R. and Bezdek , J.C. (1995) 'On cluster validity for the fuzzy c-means model', IEEE Transaction on Fuzzy Systems, Vol. 3, Iss. 3, pp. 370-379.

International Organization for Standardization and International Electrotechnical Commission (2005) ISO/IEC 19794-2:2005: Information technology – Biometric data interchange format – Finger minutiae data. Geneva, ISO/IEC.

FVC2000 (2000). [online] http://bias.csr.unibo.it/fvc2000/ (Accessed 1 August 2013).

FVC2002 (2002). [online] http://bias.csr.unibo.it/fvc2002/ (Accessed 1 August 2013).

FVC2004 (2004). [online] http://bias.csr.unibo.it/fvc2004/ (Accessed 1 August 2013).

FVC2006 (2006). [online] http://bias.csr.unibo.it/fvc2006/ (Accessed 1 August 2013).

Price, A. (2011) German researchers hack, clone common smart card model. http://www.dw.de/german-researchers-hack-clone-common-smart-card-model/a-15511213 (Accessed 1 August 2013).

Grother, P., Salamon, W., Watson, C., Indovina, M. and Flanagan, P. (2011) 'Performance of Fingerprint Match-on-Card Algorithms', National Institute of Standards and Technology,

http://www.nist.gov/customcf/get_pdf.cfm?pub_id=908096 (Accessed 1 August 2013).

Krivec, V., Birchhauer, J.A., Marius, W. and Bischof, H. (2003) 'A Hybrid Fingerprint Matcher in Memory Constrained Environments', Proceedings of the 3rd International Symposium on Image and Signal Processing and Analysis, pp. 617-620.

Rikin, A.S., Li, D., Isshiki, T. and Kunied, A.H. (2005) 'A Fingerprint Matching Using Minutia Ridge Shape for Low Cost Match-on-Card Systems', IEICE Trans. Fundamentals, Vol. E88–A, No. 5, pp.1305-1312.

Allah, M.M.A. (2005) 'A Fast and Memory Efficient Approach for Fingerprint Authentication System', IEEE Conf. on Advanced Video and Signal Based Surveillance, pp.259-263.

Mimura, M., Ishida, S. and Seto, Y. (2002) 'Fingerprint Verification System on Smart Card', International Conference on Consumer Electronics, pp182-183.

Sanchez-Reillo, R., Mengibar-Pozo, L. and Sanchez-Avila, C. (2003) 'Microprocessor smart cards with fingerprint user authentication', IEEE Aerospace and Electronics Systems, Magazine, Vol. 18, Iss. 3, pp. 22 -24.

Sanchez-Reillo, R. and Sanchez-Avila,C. (2002) 'Fingerprint verification using smart cards for access control systems', IEEE Aerospace and Electronics Systems Magazine, Vol. 17, Iss. 9, pp.12-17.

Bistarelli, S., Santini, F. and Vaccarelli, A. (2005), 'An Asymmetric Fingerprint Matching Algorithm for Java Card™', Audio- and Video-Based Biometric Person Authentication LNCS, Vol. 3546, pp. 279-288.

Figure 1 Illustration of deformation using different location for alignment

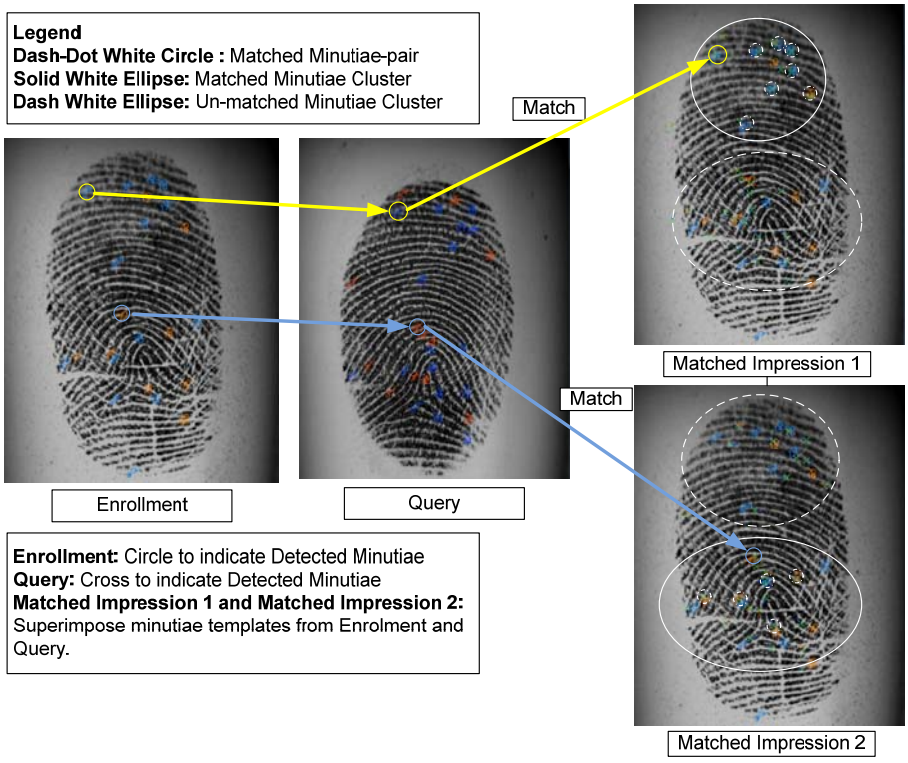

Figure 2 Illustration of matched pair of minutiae with their nearest neighbors

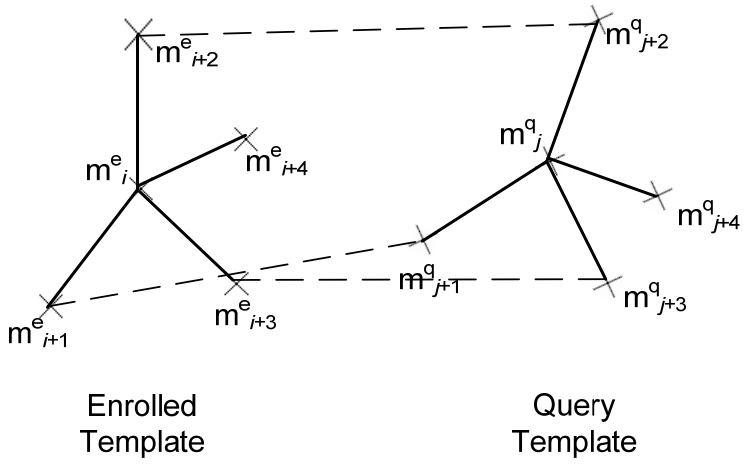

Figure 3 Pseudo code to calculate the in matcher-clustering process

```
function MinutiaeClustering(Init_i,Init_j)
      for i=Init_i..m // enrolled template
            for j=Init_j..n // query template
              Compute the matrix ξ_{i,j} using Equation (9)
              Compute ξ'_{i,j} using maximum filter
              Compute the group similarity vector ν
              using Equation (10)
              Compute ls from ν using Equation (11)
              Compute ς_p from ls using Equation (12)
                  if(ς_p < Threshold_cluster)
                    continue;
                  else
                    Store the matched minutiae into Cp
                    and C'p;
                    p=p+1;
                    return;
                  endif
            endfor % j
       endfor % i
return;
```

Figure 4 Diagram illustrates the update of Init_i and Init_j

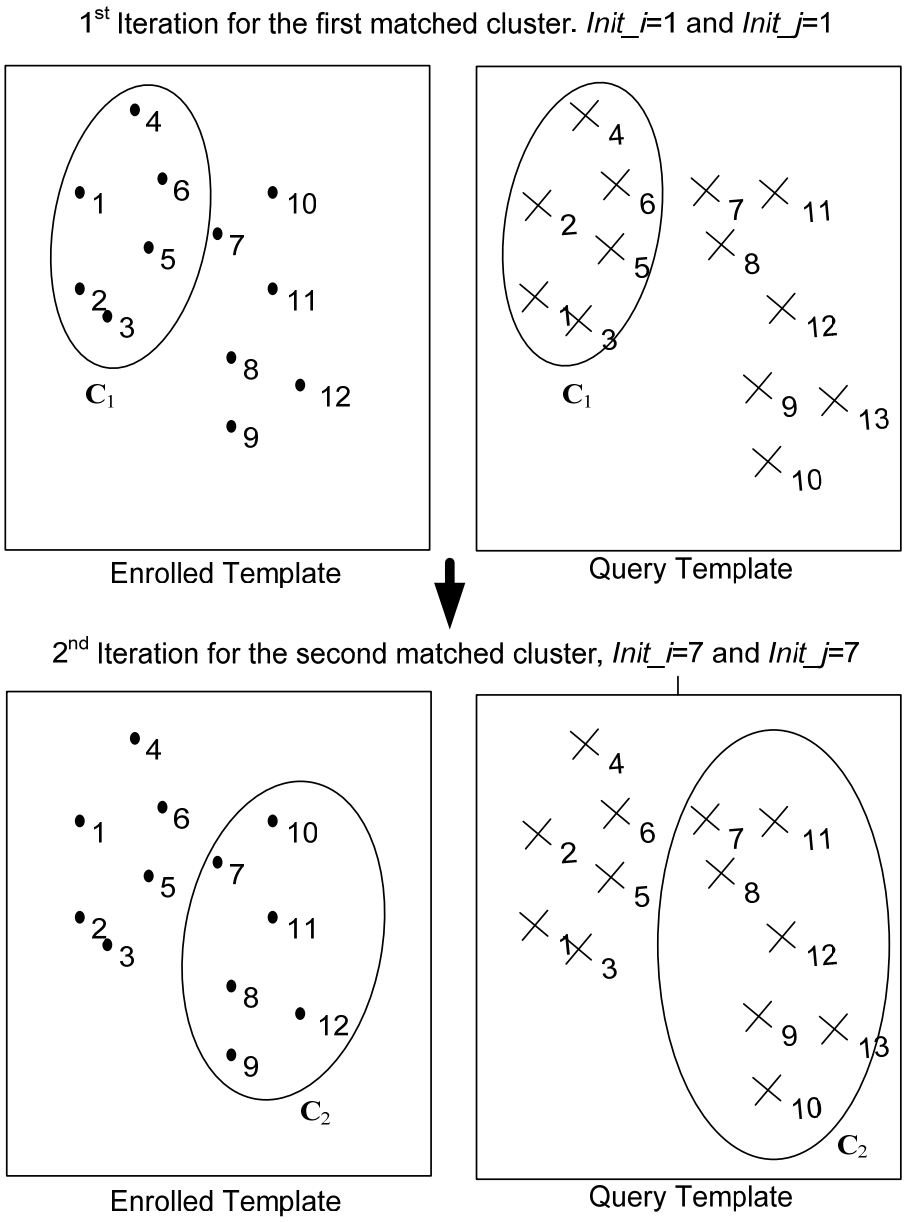

Figure 5 Psuedo-code to reject those bad clusters

$N_{matched}$ = 0;
for p = 1.. $\eta_{matched}$
    To calculate β_p using equation (14) and (15)
        if β_p > $Threshold_{Inter\text{-}cluster}$
            α_p = 1; % Correct cluster
            Count number of matched minutiae, $N_{matched}$
        else
            α_p = 0; % Wrong cluster, remove the cluster.
endfor

Figure 6 Receiver Operating Curves for FVC2000

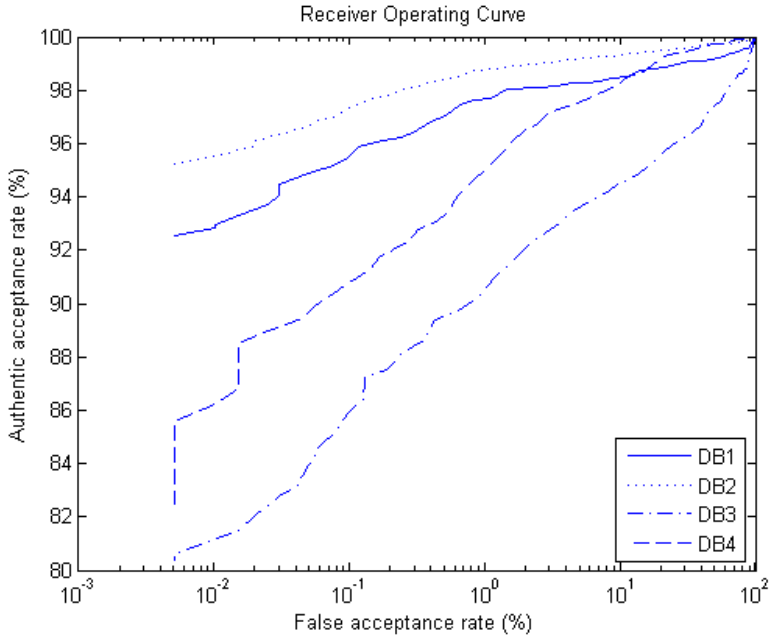

Figure 7 Receiver Operating Curves for FVC2002

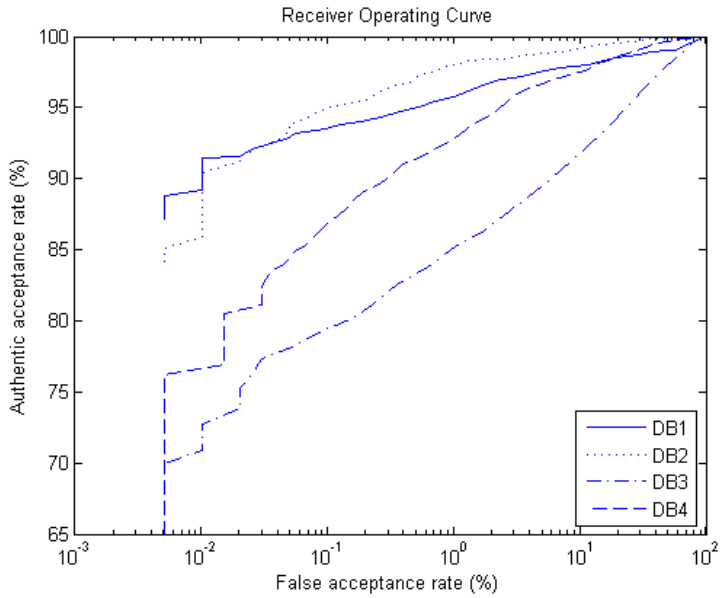

Figure 8 Receiver Operating Curves for FVC2004

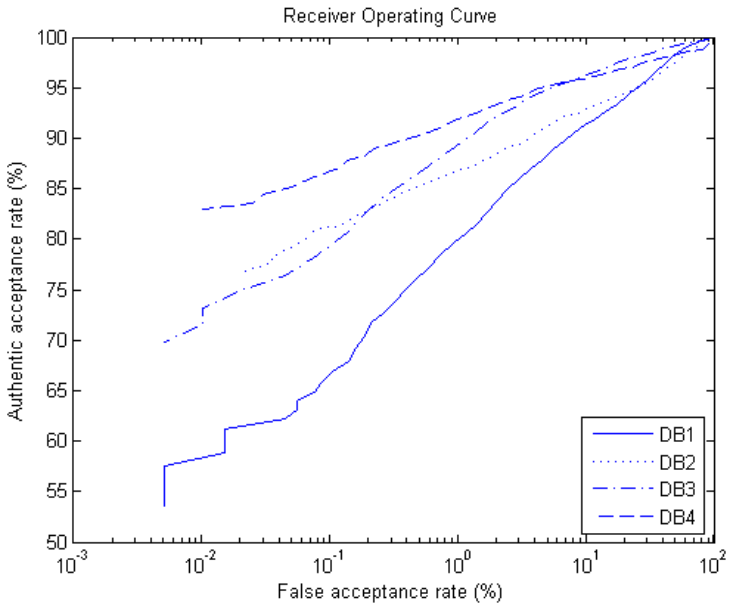

Figure 9 Receiver Operating Curves for FVC2006

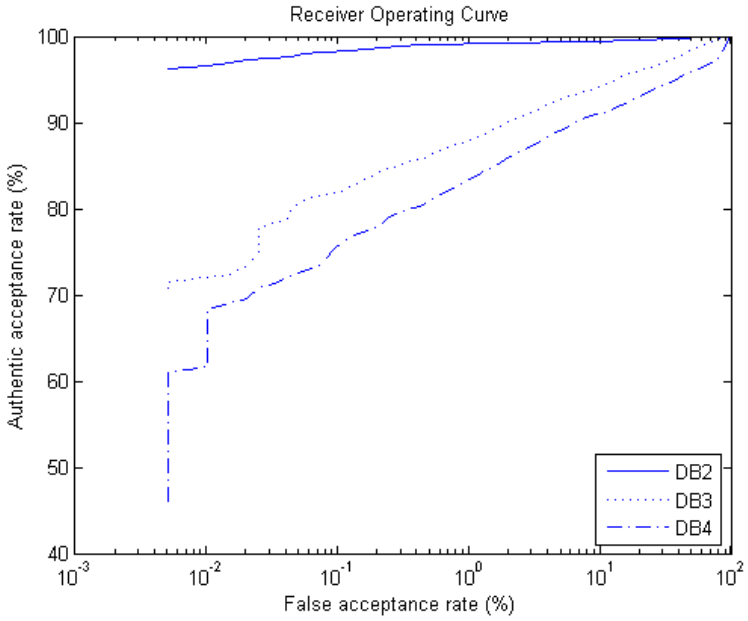

Figure 10 Samples of minutiae detected by MINDTCT using FVC database images

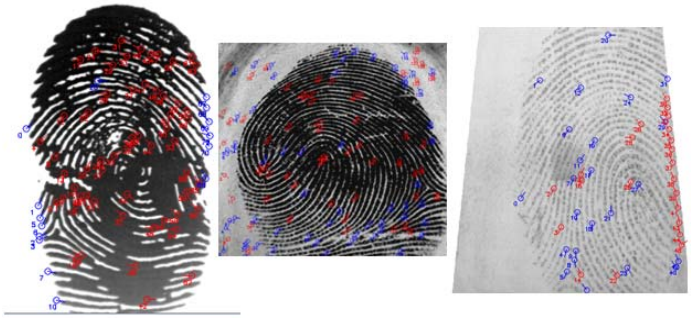

Figure 11 Receiver Operating Curves for FVC2000 (NIST Submission – 60 Minutiae)

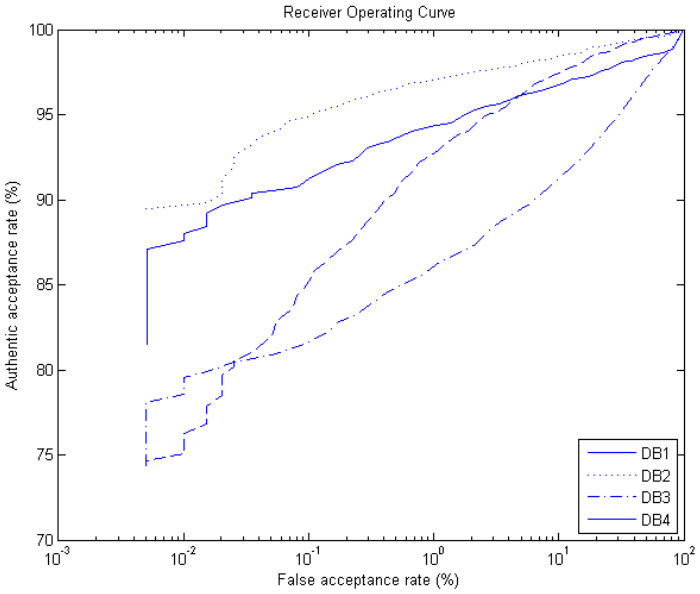

Figure 12 Receiver Operating Curves for FVC2002 (NIST Submission – 60 Minutiae)

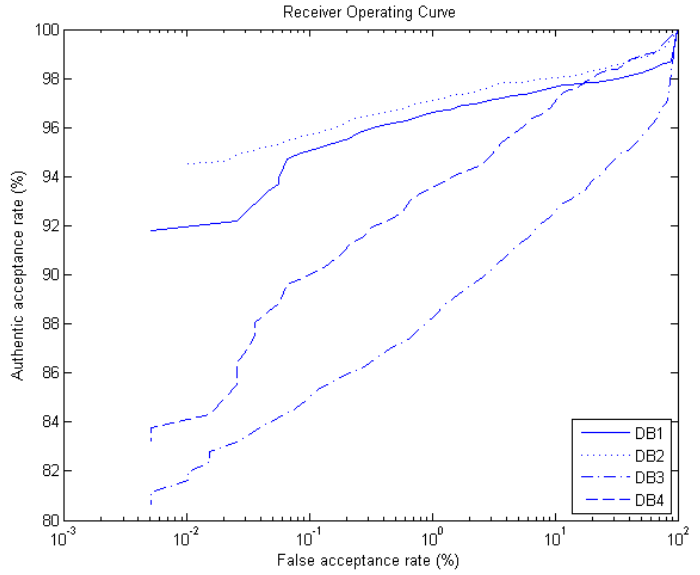

Figure 13 Receiver Operating Curves for FVC2004 (NIST Submission – 60 Minutiae)

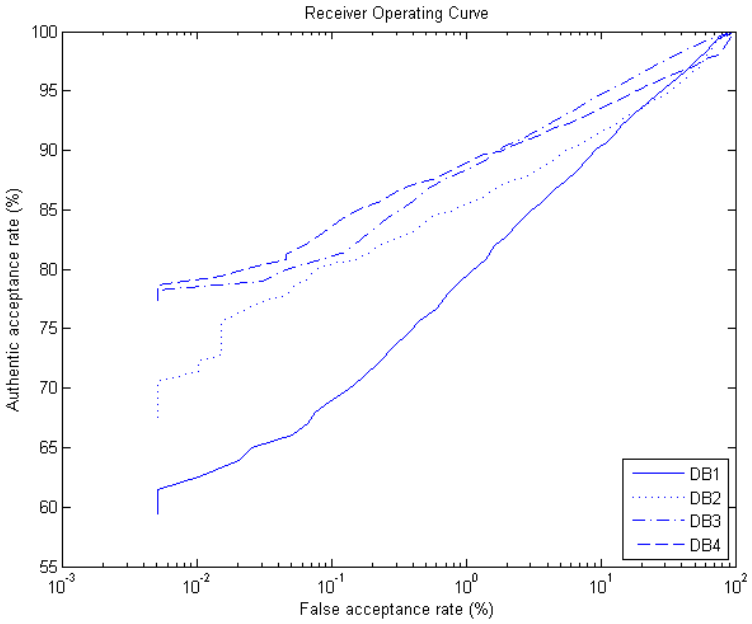

Figure 14 Receiver Operating Curves for FVC2006 (NIST Submission – 60 Minutiae)

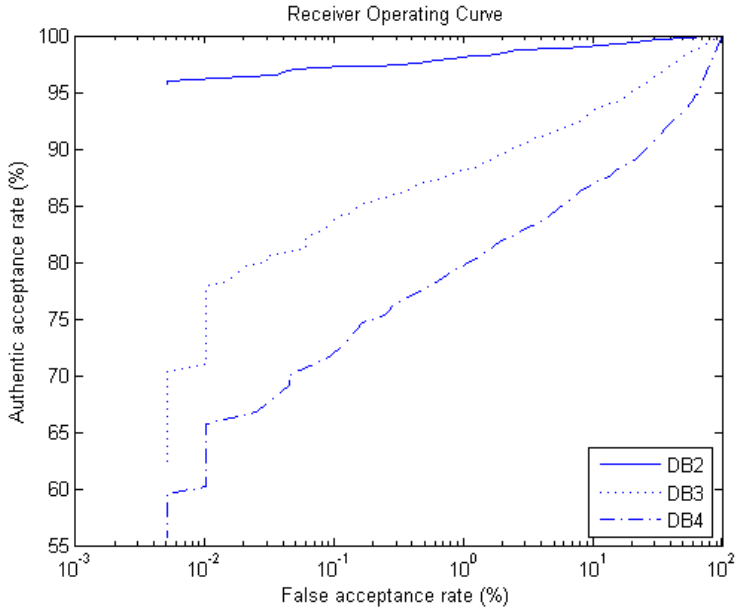

Table 1 Constraints to be solved for fingerprint MoC by various approaches

	Constraints	Approaches				
		Jain et al. (2006) and Jiang and Yau (2000)	Krivec et al. (2003)	Bistarelli et al. (2005)	Rikin et al. (2005), Allah (2005), Sanchez-Reillo et al. (2003) and Mimura et al. (2002)	Fan et al. (2012), Cao et al. (2010), Wen et al. (2009), and Tan et al. (2006)
1	Limited minutiae information for interoperability using standard template – Able to use the coordinates, ridge directions and minutiae types.	Partially	Partially	Yes	Yes	No
2	Robust to inconsistent minutiae detection.	Partially	No	Partially (limited test result)	No	Yes
3	Limited memory and processing power of smart card.	No	Yes	Yes	Yes	No

Table 2 The Equal Error Rates of the proposed algorithm on FVC2000, 2002, 2004 & 2006

EER(%)	FVC2000	FVC2002	FVC2004	FVC2006
DB1	2.891112	1.948648	9.034226	NA
DB2	1.63899	1.204097	7.690891	0.879893
DB3	8.770404	6.142982	5.187307	6.644904
DB4	3.775365	2.911489	4.925723	9.020845

Table 3 Receiver Operating Curves of FVC2000, 2002, 2004 & 2006 (NIST Submission – 60 Minutiae)

EER(%)	FVC2000	FVC2002	FVC2004	FVC2006
DB1	4.123298	2.892075	9.726841	NA
DB2	2.442774	2.469681	8.813489	1.673719
DB3	9.254912	7.878775	6.525230	7.652235
DB4	4.221829	4.357819	7.211304	12.562244

Table 4 The Equal Error Rates of FVC2000 Db3, 2004 Db1 & 2006 DB4 using our proprietary minutiae detector

	FVC2000 DB3	FVC2004 DB1	FVC2006 DB4
EER	6.208%	8.426%	6.829%