International Journal of Computer and Electrical Engineering, Vol. 2, No. 5, October, 2010
1793-8163

Fingerprint Identification in Biometric Security Systems

Mary Lourde R* and Dushyant Khosla**

***Abstract*—Perhaps the most important application of accurate personal identification is securing limited access systems from malicious attacks. Among all the presently employed biometric techniques, fingerprint identification systems have received the most attention due to the long history of fingerprints and their extensive use in forensics. This paper deals with the issue of selection of an optimal algorithm for fingerprint matching in order to design a system that matches required specifications in performance and accuracy. Two competing algorithms were compared against a common database using MATLAB simulations.**

***Index Terms*—Fingerprint Matching, Biometric System Design, MATLAB**

I. Introduction

Conventional security systems used either *knowledge-based methods* (passwords or PIN), and *token-based methods* (passport, driver license, ID card) and were prone to fraud because PIN numbers could be forgotten or hacked and the tokens could be lost, duplicated or stolen. To address the need for robust, reliable, and foolproof personal identification, authentication systems will necessarily require a biometric component.

The word "*biometrics*"[1] comes from the Greek language and is derived from the words *bio* (life) and *metric* (to measure). Biometric systems use a person's *physical characteristics* (like fingerprints, irises or veins), or *behavioral characteristics* (like voice, handwriting or typing rhythm) to determine their identity or to confirm that they are who they claim to be. Biometric data are highly unique to each individual, easily obtainable non-intrusively, time-invariant (no significant changes over a period of time) and distinguishable by humans without much special training. [2]

Enrollment and *authentication* are the two primary processes involved in a biometric security system. During *enrollment*, biometric measurements are captured from a subject and related information from the raw measurements is gleaned by the feature extractor, and this information is stored on the database. During *authentication*, biometric information is detected and compared against the database through pattern recognition techniques that involve a feature extractor and a biometric matcher working in cascade

A typical automated biometrics-based identification system consists of the six major components depicted in Fig. 1.

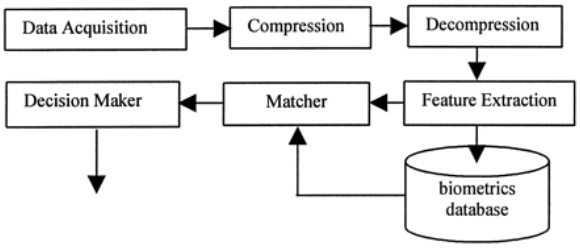

Figure 1 - A generic biometrics-based system.[1]

The data acquisition component acquires the biometric data in digital format by using a sensor. The second and third components of the system are optional, based on the system's storage requirements. The fourth component employs a feature extraction algorithm to produce a *feature vector* whose components are numerical characterizations of the underlying biometrics. The fifth component of the system is the *matcher* which compares feature vectors to produce a score which indicates the degree of similarity between the pair of biometrics data under consideration. The sixth component of the system is a decision-maker that can be programmed to accommodate system specifications.

System performance and accuracy is primarily determined by two parameters – FAR and FRR[7] . A genuine individual could be mistakenly recognized as an imposter. This scenario is referred to as "*false reject*" and the corresponding error rate is called the false reject rate (FRR); an imposter could be also mistakenly recognized as genuine. This scenario is referred to as "*false accept*" and the corresponding error rate is called the false accept rate (FAR). FAR and FRR are widely used measurements in today's commercial environment.

II. Fingerprint Identification

Fingerprints are made of a series of *ridges* and *furrows* on the surface of the finger and have a core around which patterns like swirls, loops, or arches are curved to ensure that each print is unique[3] . An *arch* is a pattern where the ridges enter from one side of the finger, rise in the center forming an arc, and then exit the other side of the finger. The *loop* is a pattern where the ridges enter from one side of a finger, form a curve, and tend to exit from the same side they enter. In the *whorl* pattern, ridges form circularly around a central point on the finger. The ridges and furrows are characterized by irregularities known as *minutiae*, the distinctive feature upon which finger scanning technologies are based. Minutiae points are local ridge characteristics that occur at either a ridge bifurcation or a ridge ending. The ridge ending is the point at which a ridge terminates. Bifurcations are points at which a single ridge splits into two ridges. Minutiae and patterns are very important in the analysis of fingerprints since no two fingers have been

* Associate Professor, EEE, marylrd@yahoo.com
** B.E. (Hons.), EIE, dushyantkhosla320@gmail.com
BITS-Pilani, Dubai; Dubai International Academic City, U.A.E.

shown to be identical.

There are five stages involved in finger-scan verification and identification:

1. Fingerprint Image Acquisition
2. Image Processing
3. Locating Distinctive Characteristics
4. Template Creation
5. Template Matching

A sensor takes a mathematical snapshot of the user's unique pattern, which is then saved in a fingerprint database. A fingerprint *enhancement* algorithm (that uses *Gabor filters* as band-pass filters to remove the noise and preserve true ridge/valley structures) is included in the minutiae extraction module to ensure that the performance of the system is not affected by variations in quality of fingerprint images.

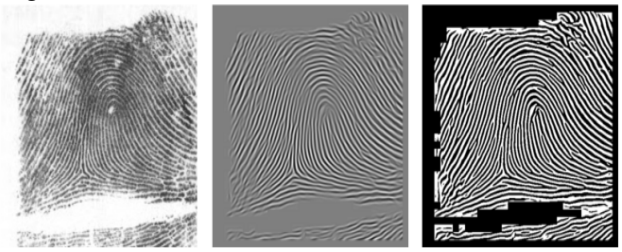

Figure 2 - The noisy fingerprint image, output of the enhancement module and the final binary image.

The continuously changing directions of the ridges constitute an *oriented texture* possessing different spatial frequency, orientation, or phase; and hence, by decomposing the image in several spatial frequency and orientation channels fingerprints can be discriminated or matched.

A. Feature Extraction

Most Feature extraction algorithms function on the following four steps

- Determine a reference point for the fingerprint image,
- Tessellate the region around the reference point,
- Filter the region of interest in different directions, and,
- Define the feature vector.

B. Fingerprint Matching

Fingerprint matching refers to finding the similarity between two given fingerprint images. Due to noise and distortion introduced during fingerprint capture and the inexact nature of feature extraction, the fingerprint representation often has missing, spurious, or noisy features. Therefore, the matching algorithm should be immune to these errors. The matching algorithm outputs a similarity value that indicates its confidence in the decision that the two images come from the same finger.

The existing popular fingerprint matching techniques can be broadly classified into three categories depending on the types of features used:[8]

- Minutiae-based
- Correlation-based
- Euclidean distance-based

One of the main difficulties in the minutiae-based approach is that it is very difficult to reliably extract minutiae in a poor quality fingerprint image. The simplest correlation-based technique is to align the two fingerprint images and subtract the input image from the template image to see if the ridges correspond. For the third approach, matching is based on a simple computation of the Euclidean distance between the two corresponding feature vectors, and hence is extremely fast.

III. Comparison of Competing Fingerprint Matching Algorithms

From an extensive research of available literature, it was found that software based on competing matching algorithms had been developed and were available as freeware. Most of them were built upon a common MATLAB based platform by picking up m-files from an open source and integrating them according to the algorithm that they desired to implement.

Two such software – one based on the traditional Minutiae-Matching Algorithm developed at the Hong Kong Baptist University, and the other a hybrid of the former and a novel Gabor-filter bank technique developed at the Michigan State University in the USA – were downloaded and compared against a common database.

A. The Database.

The publicly available *NIST Special Database 4*, which contains 8-bit gray scale images of randomly selected fingerprints distributed for use in the development and testing of automated fingerprint classification systems was used. In addition, a small personal database of 20 prints (10 pairs) was created from my friends, by the inked method. The images were scanned using a standard Epson scanner and saved in the JPEG format at 500dpi according to the accepted standard.

B. A Minutiae-Based Matcher (developed by the Hong Kong Baptist University)

To implement a minutia extractor, a three-stage approach is used, as shown in Fig 3.

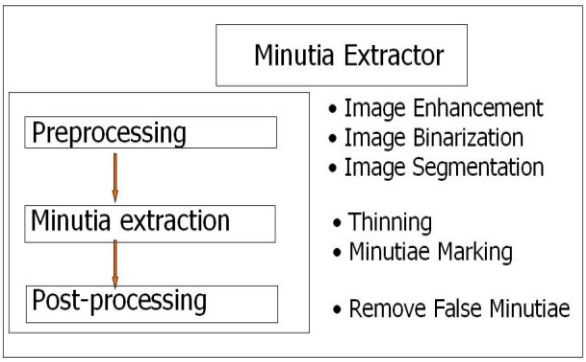

Figure 3 - Minutiae Extractor Block Diagram, HKBU

For the fingerprint image preprocessing stage, they used Histogram Equalization and Fourier Transform for image

enhancement. Binarization is done using the locally adaptive threshold method. For the post-processing stage, a more rigorous algorithm is developed to remove false minutia.

C. Fingerprint Recognition System 5.1

It is developed by S. Prabhaker and A. Jain at Michigan State University, and published as free-ware by Luigi Rosa [3]. The proposed filter-based algorithm uses a bank of Gabor filters to capture both local and global details in a fingerprint as a compact fixed length feature vector called a FingerCode [5] . The fingerprint matching is based on the Euclidean distance between the two corresponding FingerCode vectors.

D. The Method of Comparison

It was noted that, on the whole, the prints obtained from the NIST database-4 were of superior quality to the inked ones. 50 (25 pairs) fingerprint images of quality better than the rest (based on observation) were selected out of the total sum and fed first as input into the minutiae based matcher [4], and then into the filter-bank based matcher [3]. Both the softwares enable the user to first enter one of the two fingerprints in a matched pair in a database that the programs save locally on the disk. Then they asks for a second print to be matched against the database in the search for a match. The output values from each measurement were recorded.

IV. Results

The minutiae based approach [2] discussed in IIIB is used, at low FARs it captured a good amount of global information and was able to distinguish between fingerprints that have a very similar global structure.

When 25 pairs of fingerprints (of superior quality) were fed into the software using filter based algorithm discussed in section IIIC, the results were as follows: (Threshold Value = 35)

- No. of False Accepts = 2 (8 %)
- No. of False Rejects = 1 (4 %)

Now, here, we have a sort of an anomaly. Since the false accept rate is greater than the false reject rate, this would seem to suggest that the algorithm offers very little security, and is almost not effective at all. The cause of this sort of deviation may be attributed to the fact that the database that was used was small, and not *representative* of the minimum decorum needed for the proper functionality of the software. Possible, this could be remedied by using a large number of prints over which this error might gradually recede to the acceptable limits.

From the data provided by the vendor, it can be seen that these errors lie within acceptable ranges when the software was tested against a standard 10,000 print strong database.

One more issue worth addressing here is that incase we have to proceed with such a situation where number of FAR is greater than FRR, what we could do is create a log of every query made to the system and incase of every FAR we could use human intervention to clarify the claim to access till the system is fixed.

We noticed that the imposter distribution was *wider* than the corresponding imposter distribution obtained for the other algorithm. The reason for this is that the FingerCodes are capable of capturing more global and local information. The genuine distribution for this approach was quite *narrow* since the Euclidean-distance based algorithm uses Gabor-filters to enhance the noisy image whereas the former algorithm uses a Histogram Equalization technique.

V. Inference

The most important outcome of this study was the fact that none of the two approaches was a clear cut winner in terms of performance, and hence none of them can be preferred over the other in a general sense. To improve overall performance, perhaps a combination of two or more known algorithms is necessary since all algorithms have their advantages and disadvantages.

Perhaps the most important fact to be understood here is that the most efficient and effective method to improve the verification for any given system is to *combine* known algorithms in a way that we can capitalize on the advantages of each and use them to overcome the shortcomings of the complementing techniques.

VI. System Design Issues

Many different fingerprint biometric technologies are available today. A highly secure fingerprint biometrics may be difficult and time-consuming to use. On the other hand, a convenient fingerprint sensor may enhance the ease and speed of use at the expense of security. It is important to understand the security requirements of an application and the level of convenience needed by the users of the biometric system.

First, we define 'Security' and 'Convenience' in terms of known variables FAR and FRR:

- Convenience = 1 – FRR (1)
- Security = 1 – FAR (2)

The higher the FRR, the less convenient the application is because more subjects are incorrectly recognized and therefore subject to denial of service or exception handling process. The higher the FAR, the less secure the application, since it will grant access to malicious imposters. Hence, it is important to realize the 'Security/Convenience Trade-off'[7] as shown in Fig. 4

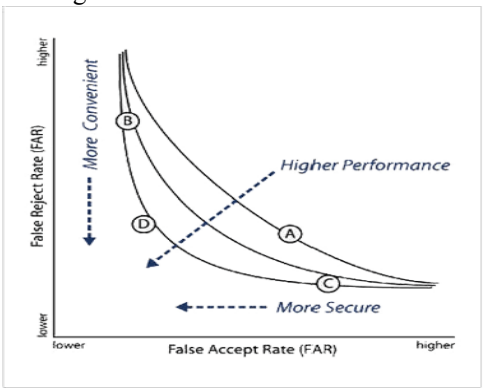

Figure 4 - The Security/Convenience Trade-Off

Depending upon the security or convenience needs of a

particular application, the designer can estimate the FAR and FRR thresholds at which the system would operate.

When it comes to personal electronic devices such as laptops or mobile telephones, cost and user convenience will be important considerations. Since this application has a low number of people using each device, a moderate FAR is an acceptable security risk. Because the sensor can be quickly re-swiped in case of a rejection, a moderate FRR is acceptable.

In a limited access facility, the overriding concern will be security, and not the convenience of the people using the system or the cost of the sensor. Technically, this type of application requires a very low FAR, to ensure that security is very high. This means that the sensor and matching system must be extremely sensitive to variations. They, however, could deny access to authorized users (higher FRR) from time to time which is the price to pay for enhanced security. (Convenience is compromised).

Systems at immigration departments form a typical case. Security must be quite high so that criminals and terrorists or other malicious entities do not cross the border into a country. Additionally, the application must be very convenient so that a large number of people can be processed relatively quickly to keep the lines moving steadily. Technically, the security requirements of this application call for a low FAR, but must also have a moderately low FRR to keep the lines short and moving. In the case of FRR situations, a person will be pulled out of line and reviewed manually by a border control agent.

A. Comparing Security Systems

The ROC curves of two different systems plotted on the same axes enable us to view their comparative performance. Based on this curve we can decide which would be better suited to a particular application, considering we have the relevant data describing the specifications of both the systems. As evident from Fig. 5, matcher 'a' is superior to matcher 'b' since for every possible FRR, its FAR is lower.

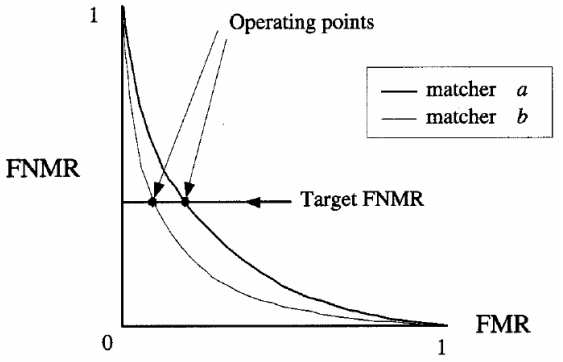

Figure 5 - Comparison of systems based on ROC

A DET curve is a modified ROC curve[9] which is sometimes preferred for its ease of interpretation. It plots FRR vs. FAR using logarithmic axes. This helps to spread out the plot and helps in identifying superior system performance more clearly.

VII. Conclusion and Future Work

The issue of selection of an optimal algorithm for fingerprint matching in order to design a system that matches the expectations in performance and accuracy is of great concern to designers. It is essential to first understand the basic architecture of a biometric based security system and then proceed onto finding out how a typical fingerprint authentication system works.

In order to achieve desired accuracy and system performance, it is essential to fully understand all specifications and then implement a combination of existing algorithms (or a modification of them).

From freeware available on the internet, image processing toolboxes and other such resources can be downloaded and integrated on a MATLAB platform to create a personal small scale authentication system.

Acknowledgment

Dushyant Khosla would like to acknowledge Dr. Mary Lourde, his project guide for her immense support, genuine concern and invaluable advice that helped and guided him whenever the need arose.

References

[1] Bolle R, Connell J, et al. Guide to Biometrics, Springer, 2003.
[2] Jain LC, Intelligent Biometric Techniques in Fingerprint and Face Recognition, CRC Press, 1999
[3] Maltoni D, Jain AK, Maio D, Prabhakar S, Handbook of Fingerprint Recognition, Springer, 2004
[4] Vacca JR, Biometric Technologies and Verification Systems, Butterworth-Heinenmann, 2007
[5] Munir MU, Javed MY; "Fingerprint Matching using Gabor Filters"; 2005
[6] NTSC Subcommittee on Biometrics, "Fingerprint Recognition", 2000
[7] http://www.isc365.com/Biometrics_Security_Vs_Convenience.aspx
[8] http://www.csse.uwa.edu.au/~pk/Research/MatlabFns
[9] http://biometrics.cse.msu.edu/fingerprint.html