Implementation and Experiments on Fingerprint Based Authentication System (FBAS) Using Delaunay Triangulation and Voronoi Diagram

Ridwan Boya Marqas

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Eastern Mediterranean University
February 2017
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Mustafa Tümer
Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Computer Engineering.

Prof. Dr. Işık Aybay
Chair, Department of Computer Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Computer Engineering.

Assoc. Prof. Dr. Alexander Chefranov
Supervisor

Examining Committee

1. Assoc. Prof. Dr. Alexander Chefranov _____

2. Assoc. Prof. Dr. Gürcü Öz _____

3. Assoc. Prof. Dr. Önsen Toygar _____

ABSTRACT

The fingerprint identification system is of great importance nowadays. This thesis is study about the survey of Fingerprint Based Authentication System (FBAS). The FBAS is implemented as several steps such as pre-processing steps, minutia extraction, Delaunay triangulation and Voronoi diagram. The pre-processing steps such as Fourier Transformation, Histogram equalization, Binarization, Region of Interest (ROI), Sobel filter are also described in this thesis. The minutiae extraction are applied on 3x3 block according to the number of neighbours around the centre value. It is obtained using the irreversible template of each fingerprint input image by Delaunay triangulation and Voronoi diagram to enhance the encryption level of the system. In this thesis, we used similarity method to check the number of dissimilar characters between lookup strings and querying lookup string of fingerprint impressions.

We evaluate the FBAS by measuring the Genuine Acceptance Rate (GAR), False Acceptance Rate (FAR) and False Rejection Rate (FRR). In this thesis, we evaluate the system by using the following databases: FVC2000DB1_B, FVC2002DB1_B and FVC2004DB1_B. The results are satisfactory for all the databases. Additionally, we conduct real-time application of our algorithm on several users' fingerprints by using the optical fingerprint scanner that shows our algorithm's accuracy which is reliable for real-time application.

Keywords: Fingerprint Based Authentication System (FBAS), Delaunay Triangulation, Voronoi Diagram.

ÖZ

Günümüzde parmakizi tanıma sistemleri büyük önem taşımaktadır. Bu tezde, parmakizi tanıma sistemleriyle ilgili bir çalışma yapılmıştır. Bu çalışma, parmakizine Dayalı Tanıma Sistemi'nin (FBAS), önişlem, parmakizi üzerindeki küçük ayrıntıların (minutia) çıkartılması, Delaunay Üçgenselleştirme ve Voronoi Diyagramı gibi adımlar kullanılarak yapılmasıyla oluşmuştur. Önişlem yöntemlerinden Fourier Dönüşümü, Histogram Eşitleme, İkiye Ayırma (Binarize), İlgi Bölgesi (ROI), Sobel Süzgeci kullanılmıştır. Minutia çıkartılması işlemi 3x3 boyuntundaki görüntü bölütleri üzerinde uygulanmıştır. Sıkıştırma seviyesinin artırılması için herbir parmakizi üzerinde Delaunay Üçgenselleştirme ve Voronoi Diyagramı uygulanmıştır. Bu tezde, parmakizi karşılaştırma işlemleri için benzerlik yöntemleri kullanılmıştır. FBAS sisteminin başarımı GAR, FAR ve FRR ölçüm birimleri ile gösterilmiştir. Bu tezde kullanılan parmakizi veritabanları ise FVC2000DB1_B, FVC2002DB1_B ve FVC2004DB1_B'dir. Deney sonuçları bütün veritabanları üzerinde memnuniyet vericidir. Bu çalışmalara ek olarak, birkaç kullanıcının parmakizi üzerinde optik parmak okuyucusu kullanılarak, algoritmamızın gerçek zamanlı uygulamasıyla yürüttüğümüz çalışma, algoritmamızın doğruluğunun güvenilir olduğunu göstermektedir.

Anahtar Kelimeler: Parmak İzi Tabanlı Kimlik Doğrulama Sistemi (FBAS), Delaunay Üçgenselleştirme, Voronoi Diyagramı.

DEDICATION

First of all, I would like to make it as a gift for the greatest ever my family to their love, support and sacrifice for me even and believe me even with obstacles availability and war situation in my country.

Additionally, for great friends, colleagues and relatives support me and push me forward to achieve the aim.

Believe in yourself is the first step to achieving the goal.

ACKNOWLEDGMENT

I might want to express my exceptional gratefulness and because of my supervisor Assoc. Prof. Dr. Alexander Chefranov, you have been a huge guide for me. I might want to thank you for empowering my exploration and for permitting me to develop as an examination researcher.

I would like to thank Prof. Dr. Işık Aybay, Chairman of Computer Engineering department, who provided us varies of research facilities. I shall be thankful to Assoc. Prof. Dr. Önsen Toygar, Assoc. Prof. Dr. Gürcü Öz and all those who have shared time with me for their understanding and companionship through the years, and special thanks to those who reviewed my thesis and provide me constructive criticism and feedback on my work over the years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CPU	Central Processing Unit
DBMS	Database Management System
DFT	Discrete Fourier Transform
FAR	False Acceptance Rate
FBAS	Fingerprint Based Authentication System
FFT	Fast Fourier Transform
FRR	False Rejection Rate
GAR	Genuine Acceptance Rate
HD	High Definition

Chapter 1

INTRODUCTION

1.1 Introduction to Fingerprint Based Authentication System (FBAS)

Within the past four decades, identification and recognition techniques are continuously moving from hard copy database to soft copy or computerized database. The Identity cards and many others personal identification documents are now saved in digital copy. A biometric system refers to an identification system which is based on human trait.

Moreover, the strength or the effectiveness of a human physical or behavioural trait used as biometric characteristic is approved only after a system based on it has been tested on a large sample size. The physical and behavioural human characteristics used often for the biometrics purpose is shown in Table 1.1. The fingerprint appears to be the most used trait for building a biometry system. Nowadays, the study of fingerprint and their nature and their application in biometry is widely explored by researchers. The step through which one can move the thumb (finger) are roughly the following, image acquisition and processing, feature extraction and generation of a bio key for a further use during the matching process [1-19]. The matching process is the important step in setting the biometric system.

Table 1.1: Biometrics Characteristics [1]

Physicals traits (characteristics)	***Behavioural traits (characteristics)***
Geometry of the hand	Gait
Fingerprints	Gesture
Skin pores	Handwritten signature
Hand veins or wrist	Keystrokes on keyboard
Palm print	Voice
Chemical structure of body odor	
Thermal emissions & facial features	
Eyes features (iris & retina)	

The matching algorithm aims is to compare an impostor fingerprint and a stored fingerprint in a database. This means a matching algorithm should be smart and accurate enough in order to correctly match a given fingerprint (impostor) with the query fingerprint, this is called verification. It should also be able to search if a given fingerprint (impostor) exists in a database, this is called identification.

The rest of the thesis is organized as follows. Chapter 2 contains survey of FBAS and problem definition. Chapter 3 describes implementation and testing of FBAS. Chapter 4 presents experiments results with FBAS in comparison to [2]. Chapter 5 concludes the thesis and discusses future work.

1.2 Conclusion

Based on our study of the [1-19], we come up with the conclusion that a fingerprint-based authentication is very important, and in this thesis we shall work on its implantation and analysis. We need also developing a matching algorithm since it is not described in details in [2].

Chapter 2

REVIEW OF FBAS AND PROBLEM DEFINITION

2.1 Review of FBAS

FBAS is abbreviation of Fingerprint Based Authentication System. The biometric system implementation is a science which is increasingly improved over years. Researchers focus on the various aspects of the FBAS. These aspects concern the design of the system, the hardware implementation of the system and the software implementation of the system. Below are listed a non-exhaustive list on researches done in the fields of biometric.

Extracting the feature of a fingerprint image is a key step for digital processing. The indexing of the extracted data is also needed into the next step of the process. In [3], an indexing approach based on the barycentre of the Delaunay triangulation of the minutiae extracted is discussed.

In [4], the authors discussed about the security side of a biometric system. To do so, the disasters that can be caused by the leaks of personal biometric data to tiers were defined as very high.

The fingerprint data are used nowadays in banking sector for secured transactions. In [4] the authors took advantages on the advent of mobile computing and proposed a bio-cryptographic protocol, which can enable secured online money transactions.

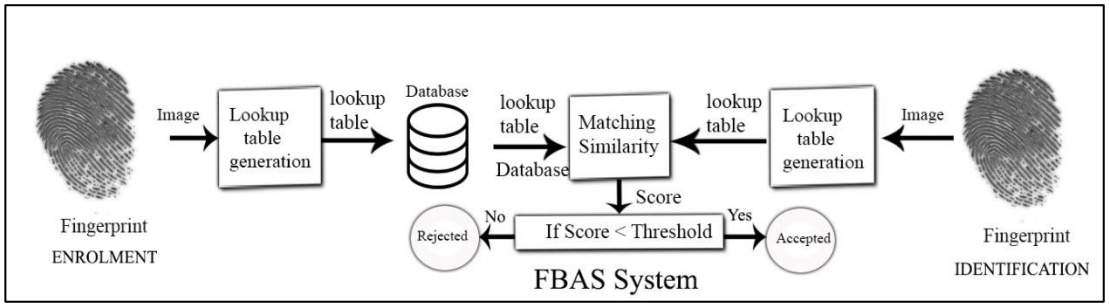
The fingerprint alignment is a very difficult task to achieve. The authors of [5] proposed a cancellable template based fingerprint design.

The Delaunay triangulation is widely used in the fields of digital image processing. In [6], for instance, the authors proposed a fingerprint, template protection method based on the triangulation computation.

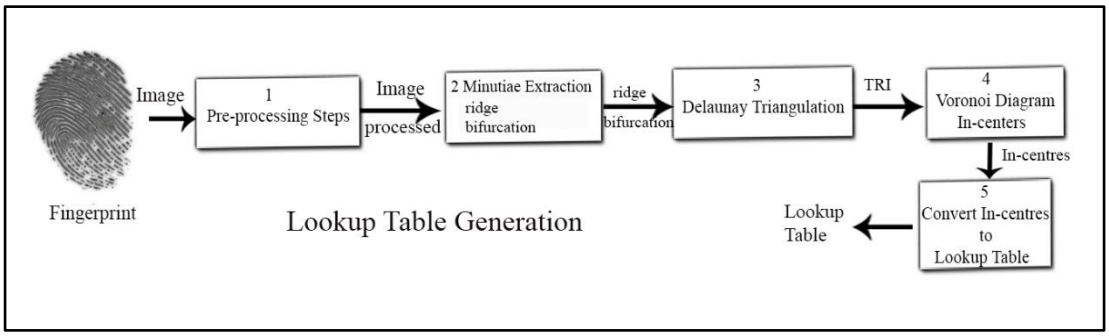
Similar work done in [7] is also done [8]. Both articles proposed fingerprint security based system that could enhance the security of online banking transactions.

In [9], a fingerprint based system for online banking is proposed as in [7, 8]. The authors also proposed an implementation of the proposed system. A fingerprint matching method based on the alignment of the coordinates of the extracted fingerprint minutiae is proposed in [10]. Similar to [10], a matching algorithm is proposed in [11]. The method is centred on alignment method.

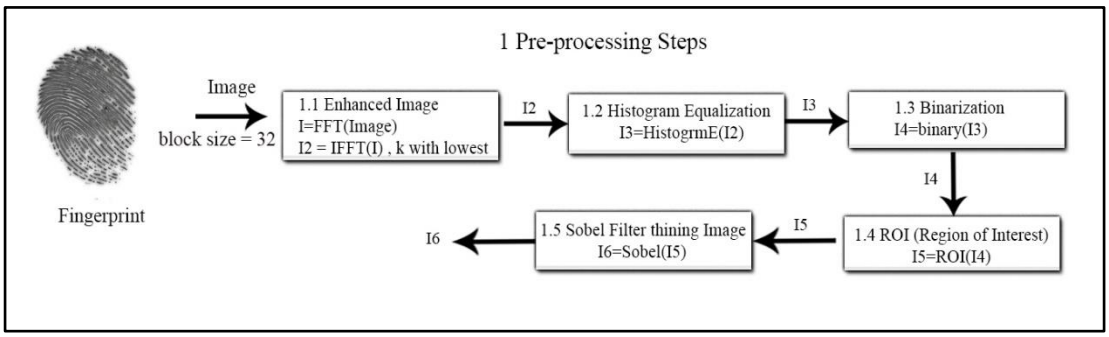
The structure of FBAS is shown as in Figure 2.1. Meanwhile, Figure 2.1 (a) shows FBAS basic structure. While Figure 2.1(b) indicates “Lookup table generation” which is first step in Figure 2.1 (a). Similarly, Figure 2.1 (c) represents the “pre-processing step” which is the first step of Figure 2.1(b).



(a)



(b)



(c)

Figure 2.1: FBAS Structure. (a) Indicate Basic Structure of FBAS while (b) and (c) Represent Lookup Table Generation and Pre-processing Steps Respectively

First step of the FBAS is lookup table which consist of pre-processing steps, Minutiae extraction, Delaunay Triangulation and Voronoi Diagram, according to Figure 2.1 (a). Therefore, the following steps describe the sub parts of lookup table:

2.2 Fingerprint Image Pre-processing and Minutiae Extraction

We discuss the image acquisition techniques in this section. The acquisition is the first step of image processing [2]. It is therefore prior to all other step. The quality of the image and the acquisition technique do matter a lot in the image processing; namely feature extraction. When it come to the fingerprint image, parameters such sensitivity of the tools used for acquisition, status of the finger itself (dirty or clean, sweating), and the thermal environment are of great importance. The assumption make prior to theoretical studies of image acquisition are of great importance. In practice, we aim to build a biometric system such a way to combine both efficiency and flexibility. There exits various method to capture or acquire an image. There are even more techniques to get fingerprint image, because beyond all the existing technique of image acquisition, one can also get the fingerprint by using a paper and ink. In the special case of fingerprint image acquisition, we discuss old technique such acquisition via paper (solid-state fingerprint readers) and ink and modern technique such the acquisition using electronic device, sensors (optical fingerprint readers). The first mentioned technique is very rudimentary though it has existed for several decades; it is based on electrochemical reaction whereas the second is purely electronic or computerized based method [12]. It is based on an optical scanner.

2.2.1 Enhance Image

For enhancement of the image, the Fourier transform (FT) is one of the most used image processing tools. It consists of the decomposition of an image into sine and cosine components. The results is called frequency domain whereas the input image is called the spatial domain [13]. There is a direct correspondence between each point from the frequency domain with the frequency in the spatial domain. Each point represents frequency.

The discrete Fourier transform (DFT), this technique is mostly used with digital image. This is a sub-category of the Fourier transform. The DFT does not require all the frequencies of the original image in the spatial domain. Instead, it makes use only of the sample of frequencies enough to describe the spatial domain. The correspondence between the frequency domain and the spatial domain is as follows: the number of frequency from the frequency domain is equal to the number of pixels in the spatial domain. Consider an m x n squared image (32x32 block size) then the Fourier Transform is following [2]:

$$F(u,\ v) = \sum_{x=0}^{m-1} \sum_{y=0}^{n-1} f(x,\ y) \times \exp\left\{-\mathrm{j}2\pi \times \left(\frac{ux}{m} + \frac{vx}{n}\right)\right\} \tag{2.1}$$

Where m and n is the block size and x, y is the pixel position on that block [2].

The reverse transformation of a Fourier image to its spatial domain can be obtained throughout the inverse Fourier transform. To obtain the inverse Fourier Transform on each block (32x32) pixels, the following function can be used:

$$G(x,\ y) = F^{-1}\{F(u,\ v)\ |F(u,\ v)|^k\} \tag{2.2}$$

Where k value is constant in equation (2.2), that improve the appearance of ridges, the k with highest value will lead result to wrong connection between ridge endings [2]. But they didn't define the k value clearly. However, $F^{-1}(F(u, v))$ can be calculated by following equation (2.3) [2].

$$f(x,\ y) = \frac{1}{mn} \sum_{x=0}^{m-1} \sum_{y=0}^{n-1} F(u,\ v) \times \exp\left\{-\mathrm{j}2\pi \times \left(\frac{ux}{m} + \frac{vx}{n}\right)\right\} \tag{2.3}$$

Where the x=0,1,2,…,31 and y=0,1,2,…,31 for each block in equation (2.4).

2.2.2 Histogram Equalization

In block diagram (Figure 2.1 (c), step 1.2), the appearance of an image can be improved through gains control or its brightness. But an ideal question is to know how their best value can be determined automatically. The answer to the above question can be intuitively given as follows: map the darkest pixel respectively the brightest pixel of the image to the pure black respectively pure white. A second solution is to find out the average colour value of the image and expand its middle grey to fill up the displayable values. It appears that, in order to clearly visualize the solutions mentioned above, we need to plot a histogram which represents individual colour channel and their intensity of light. The histogram mentioned above helps to compute the common value such as minimum, maximum and even the average value of the intensity of light of an image [1].

Assume that variable r shows the gray levels of the image to be enhanced and r has been normalized to the interval [0, 1] where r=0 represents black and r =1 represents white [1]. They consider a discrete formulation and allow pixel values to be in interval [0, L-1], where L is intensity level of input image. For each r value the transformation of the form

$$S=T(r) \text{ for } 0 \leq r \leq 1. \tag{2.4}$$

It produce a level s for every pixel value r in the original image and transformation function T(r) satisfies the following conditions: T(r) is single valued and increasing in the interval $0 \leq r \leq 1$; and $0 \leq T(r) \leq 1$ for $0 \leq r \leq 1$, where T(r) function of particular importance in image processing has the form

$$s=T(r) = \int_0^r Pr(\omega)d\omega, \tag{2.5}$$

where ω is a dummy variable of integration. The right side of the function is called as cumulative distribution function (CDF) of the random variable r, detailed is described in [1]. Similarly, the integral of a probability density function for variable in the range [0,1] also in the range [0,1], so $0 \leq T(r) \leq 1$ condition is satisfied as well. Hence it noted that T(r) depends on $P_r(r)$. We deal with probabilities and summation during discrete values instead of probability density function and integrals. The probability of occurrence of gray level r_k in the image is considered by

$$P_r(r_k) = \frac{n_k}{n} \text{ where } k=0,1,2,3\ldots,L-1, \tag{2.6}$$

where n is total number of pixels in the image and n_k is the number of pixels that have gray level r_k, and L is total number of possible gray levels in the image. Transformation function for discrete value is shown in equation (2.7).

$$S_k = T(r_k) = \sum_{j=0}^{k} p_r(r_j), \tag{2.7}$$

$$= \sum_{j=0}^{k} \frac{n_j}{n} \quad k=0, 1, 2, \ldots, L-1.$$

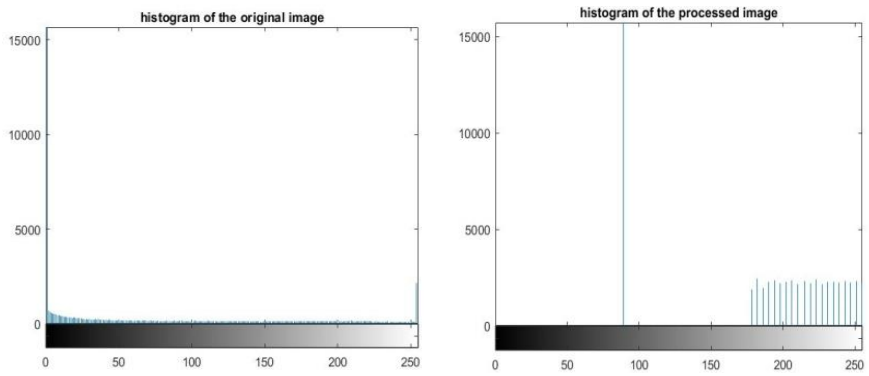
The output image of this process is obtained by mapping each pixel with level r_k in the input image into a corresponding pixel with level s_k in the output image by equation (2.7). Hence, $p_r(r_k)$ respective r_k is called histogram and the transformation in equation (2.7) is called histogram equalization[1].

One other problem which may occur is that a dark noisy region of the image can be amplifying so much to become visible enough. The global equalization is useful, nevertheless, for some image, it appears preferable to divide the image and apply a corresponding type of equalization to each part. This concerns images with wide

ranges of luminance. In this case, the image is divided into M x M blocks of pixels. The histogram equalization method is then applied on each block. The problem encountered with this technique is that the fragmentation of the image into blocks affects the results by showing blocking artifacts. The blocking artifacts are discontinuities which are observed at the blocks boundaries. To avoid blocking artifacts, we should recompute the histogram equalization of each block M x M centred at the each pixel. Such computation process is slow because it required M x $M=M^2$ operations per pixel. An efficient way to reduce the computation complexity is to use a non-overlapping blocks-based histogram equalization function with a smooth interpolation transfer function to move from one block to another. This means at the boundaries. The importance of the histogram equalization in one word is to enhance the quality of an image by expanding the distribution of the pixels value for further processing. The Figure 2.1 (a) and Figure 2.1 (b), represent a fingerprint image before equalization and after equalization respectively. On the other hand, the Figure 2.2 (a), Figure 2.2 (b) shows the difference given by the histogram of the original image and the equalized image respectively and this enhancement is computed by Matlab function's imhist(image) [14].

(a) (b)

Figure 2.2: Fingerprint Input Image. (a) is Indicating Raw Image and (b) after Equalization According to Appendix A, at Line 11 and 24 Respectively



(a) (b)

Figure 2.3: Histogram Equalization of Fingerprint Image. (a) Pixel Distribution before Figure Equalization and (B) Pixel Distribution after Equalization According to Appendix A, at Line 26 and 28 Respectively

2.2.3 Image Binarization

In block diagram (Figure 2.1 (c), step 1.3), the image binarization is a useful method applicable for the image segmentation. It helps to calculate the feature of minutiae. More usually the binarization separate the pixels of the object into two groups. One of which is the image background and the second representing the object. The binarization process used only two level of gray. The object foreground is usually represented in black (0), while the background is represented in white (255). In binary, each pixel is stored in a single bit. A threshold gray value is defined in order to build the binary image. The procedure is as follow: pixels with gray-level equal to or greater than the threshold are set to one, the remaining are set to zero. A problem that usually occurred in the binarization is the selection of the suitable threshold. The built-in Matlab function *graythresh*(*Image*) used to compute threshold value. There are two basic threshold level for binarization such as global/fixed and iterative threshold.

The fixed threshold, the value is fixed intuitively and it is used to classify every pixel based on their gray level. In which the binary image function $B(x, y)$ is computed from the original image $I(x, y)$ as follows

$$B(x,y)=\begin{cases}0 & if\ I(x,y)<T\\ 1 & if\ I(x,y)\geq T\end{cases} \tag{2.8}$$

The T is the threshold which means if the grayscale value of a pixel is greater or equal to the threshold then the pixel is set the value 1 otherwise it is set the value 0.

2.2.4 Region of Interest (ROI)

In block diagram (Figure 2.1 (c), step 1.5), ROI can be calculated for finding tightly coupled region that contains minutiae feature. ROI deducts the image from the background area. Then the ROI removes those leftmost, rightmost, uppermost and bottommost blocks out of the bound. After finding ROI, the tightly coupled region is subjected to feature extraction phase [2]. ROI can be calculated by using the following Matlab function *bwareaopen(Image)* [14].

2.2.5 Directed Image (Sobel Filter)

In block diagram (Figure 2.1 (c), step 1.5), The Sobel operator performs a 2-D spatial gradient measurement on an image and so emphasizes regions of high spatial frequency that correspond to edges [2]. For this purpose, the built-in Matlab function "*edge(Image, 'Sobel')"* has been used to detect the edges of the fingerprint and thinning it [14]. The following formula represent the Sobel filter (3x3):

$$\mathbf{G}_x=\begin{bmatrix}-1 & 0 & +1\\ -2 & 0 & +2\\ -1 & 0 & +1\end{bmatrix}*\mathrm{I} \quad \text{and} \quad \mathbf{G}_y=\begin{bmatrix}+1 & +2 & +1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{bmatrix}*\mathrm{I} \tag{2.9}$$

Where I represent the image used to apply Sobel filter on it as equation (2.9) while the G_x and G_y represent the two images with having the horizontal and vertical derivate approximations respectively [15].

2.3 Minutiae Extraction

In block diagram (Figure 2.1 (b), step 2), the identification of a pixel point as minutiae and to identify its nature is based on the neighbourhood technique, also known as crossing number technique. This technique stipulates the following: Consider a window of size 3 by 3 pixels as it shown in the pseudo code, in Figure 2.3. If the central pixel of the window has a value 1 and that exactly three pixels of its neighbourhood also have the value 1, then there is a bifurcation type of minutiae. If the central pixel of the window has a value 1 and that exactly one pixel of its neighbourhood also has the value 1, then there is a ridge ending type of minutiae [2].

```
Begin
Input: W=window 3x3 pixels
Output: R=ridge, B=bifurcation
        If (centre pixel of W == (3 neighbour pixels surrounded))
                Then B = centre pixel of W.
        Else If (centre pixel of W == (1 neighbour pixels surrounded))
                Then R = centre pixel of W.
        End If
END
```

Figure 2.4: Pseudo Code of Minutiae Extraction

2.4 Delaunay Triangulation

In block diagram (Figure 2.1 (b), step 3), Material of this section is based on [2, 16]. A Delaunay triangulation is consists of a set P points in a plane is a triangulation such that no point in P is inside the circumcircle of any triangle in Delaunay Triangulation (P), example is shown in Figure 2.4. Delaunay triangulations maximize the minimum angle of all the angles of the triangles in the triangulation. Consider an example, the set of 40 points on the plane and computed by the Delaunay Triangulation, shown it

Figure 2.4, the Matlab code of Delaunay Triangulation for 40 points is shown in Appendix D.

Example 2.1 Delaunay Triangulation computed by Matlab Function '*Delaunay-Tri()*'

Consider the Table 2.1, representing a set of 40 points on the plane R^2 .

Table 2.1: Dataset P of 40 Points

X	**Y**	**X**	**Y**
2	9	7.5	9
3.5	10	6.5	3.5
1.5	6	5.5	4
2.5	3	4.5	3.5
4	2	2	7.5
6	2.5	5.5	9.5
7	4	5	8.5
7.5	7	5	5.5
6	9.5	5	7.5
4	9	7.5	3.5
3	8	5.5	6.5
4	6	3	5
5	7.5	3	6
6.5	8.5	6	2
4.5	4		
6	5		
5.5	3.5		
4	3		
5	6.5		
2	4.5		
2.5	6.5		
5.5	9		
5	10.5		
6.5	7.5		
4	7.5		

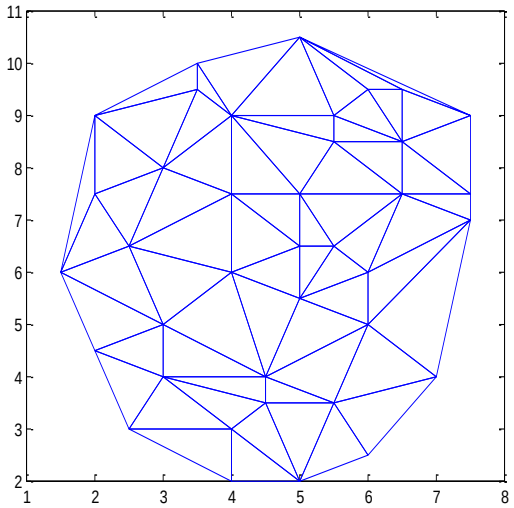


Figure 2.5: Delaunay Triangulation of 40 Points According to Appendix A, at Line 76

Given a set of points $P=\{p_1, p_2, \cdots, p_n\}$ on a plane, it is possible to compute its Delaunay Triangulation if the points are not all collinear. On the other hand, there exists another geometry shape called Voronoi Diagram that can be computed from the elements of $P=\{p_1, p_2, \cdots, p_n\}$. The Voronoi Diagram is a Dual of the Delaunay Triangulation.

2.5 Voronoi Diagram

In block diagram (Figure 2.1 (b), step 4), the materials of this section are from [17]. In what follows we will use the Delaunay triangulation to build Voronoi diagram, the In-centre are the centres of the triangles obtained from the Delaunay triangulation. After the In-centre of all the triangles are computed, their coordinates are used to generate the Voronoi diagram, example is shown in Figure 2.5 and Matlab code is described in Appendix E. The in-centre is the centre of a triangle's "in-circle", it is where the "angle bisectors" (lines that split each corner's angle in half) meet. Assuming that a set of *n* distinct points in a plane is given by $S=\{P_1, P_2, \ldots, P_n\}$. The subdivision of the plane S into n subsets each of them called a cell and such that the following property a point

$x \in P_i$ if and only if $d(x, P_i) < d(x, P_j)$, $\forall i \neq j, \quad i, j = 1 \ldots n$ is verified is called the Voronoi diagram [17].

At this point it is important to get interest in the complexity of $\upsilon(P_i)$ that is the total number of its edges and vertices.

Before the computation of the Voronoi diagram, we need to extract the In-centres of all the triangles obtained from the Delaunay triangulation.

Voronoi vertex can be exploited by considering an example, using the set of 40 points as it shown in Table 2.1.

Example 2.2 Voronoi Diagram computed by Matlab function "*Voronoi()"*

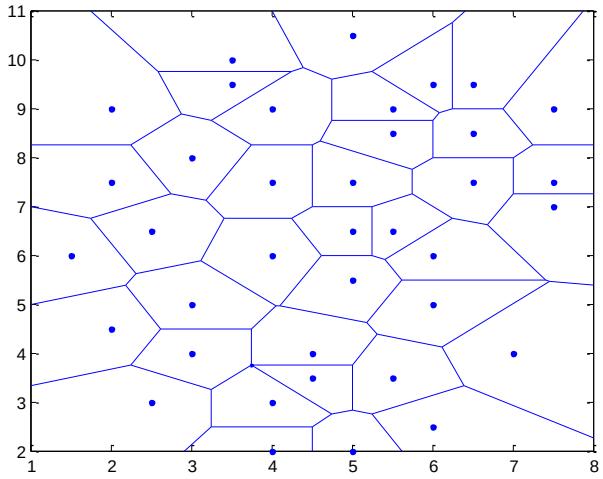


Figure 2.6: Voronoi Diagram of 40 Points According to Appendix A, at Line 86

2.6 Lookup Table [2]

In block diagram (Figure 2.1 (b), step 5), lookup table is generated after Voronoi diagram that contains of In-centres values of Voronoi vertex and coordination angle θ

of each vertex. In [2] they stored lookup values in both server and client side and to improve the security they added more number of chaff points to lookup table. However, server side lookup table contains a Boolean value for genuine and chaff points. Nevertheless, at the client side, lookup table contains combination of genuine and chaff points. However, the lookup table will be more accurate if ridge and bifurcation minutiae features are separately extracted and converting In-centres to lookup table, only the Boolean value that represents the ridge by zeros and bifurcation by ones are stored as string without storing coordinates of In-centres because the processing will be lighter and the information will be more secure because those Boolean values represented by string will be useless for retrieving the minutiae data.

2.7 Biometrics Systems Performance Evaluation [2]

The discussion in this section focus on useful metrics in biometrics sciences. We discussed the flexibility and the efficiency of a system. To do so, an evaluation framework based on the following key words or statement is defined.

False Acceptance Rate (FAR); *Genuine Acceptance Rate* (GAR); *False Rejection Rate* (FRR).

TR (True Rejection) is the number of the fake users correctly rejected, FR (False Rejection) is the number of incorrectly rejected genuine users, TA (True Acceptance) is the number of correctly accepted genuine users, FA (False Acceptance) is the number of the incorrectly accepted fake users.

G is the number of genuine and F is the number of fake users from definitions of G, F, TR, FR, TA, FA, we can write (2.9), (2.10) as follows:

$$G = TA + FR \tag{2.9}$$

$$F = TR + FA \tag{2.10}$$

False Rejection Rate (FRR) can be defined by the following way:

$$FRR = \frac{FR}{F+G} \tag{2.11}$$

False Acceptance Rate (FAR) can be represented by the following way:

$$FAR = \frac{FA}{F+G} \tag{2.12}$$

The efficiency of a biometrics system is also measured base on its ability to enable user's enrolment and authentication. The GAR is False Acceptance (FA) plus True Rejection (TR) over the genuine (G) and Fake (F).

The Genuine Acceptance Rate (GAR) can be defined as:

$$GAR = \frac{FA+TR}{G+F} \tag{2.13}$$

From (2.9)-(2.13), we get

$$\text{FAR+FRR+GAR=1} \tag{2.14}$$

That complies with experimental results from [2] that are described in section 2.8.

2.8 Experiments Results of [2]

The performance of the FBAS [2] was evaluated by using four databases in [2], which are available on internet [18] and the evaluation result is given in Figure 2.6. In this figure the experiment number 1 represents the evaluation of FVC2004DB1_A database which consists of 800 fingerprints test while the 2, 3 and 4 are representing the evaluation of FVC2004DB1_B, FVC2002DB1_B and FVC2000DB1_B databases respectively that each of them made up of 80 fingerprints.

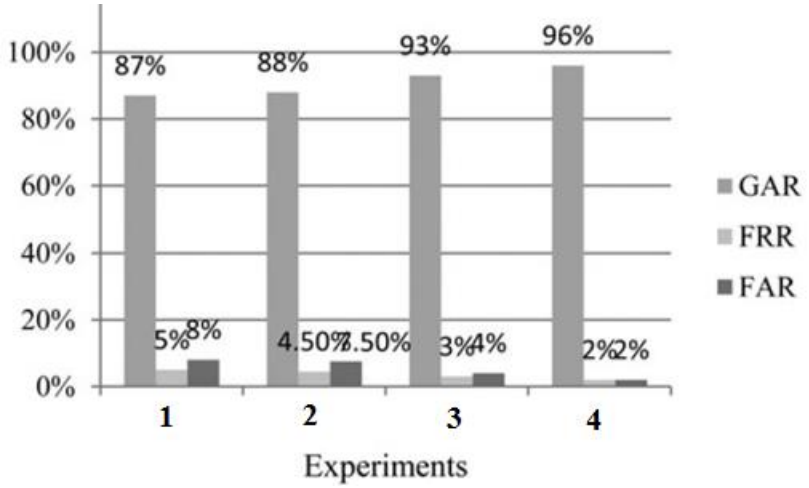


Figure 2.7: Performance Evaluation of the FBAS in [2],

We see that is all four experiment, (2.14) holds lookup table generation.

2.9 Problem Definition

We analyzed the state-of-art algorithm [2] and fundamental components including image processing, Fourier Transformation, Delaunay triangulation and Voronoi diagram. Our problem is implementation of FBAS (similar to [2]), according to Figure 2.1.

Hence, our FBAS provides well defined system properties including tools which use for system implementation, k value for the Fourier Transformation, threshold for Binarization and ROI and matching scheme (Levenshtein Distance method). Additionally, we proposed the lookup string which contains the only Boolean values (in form of string). To extract the lookup string, we implemented the pre-processing steps required to process the image from the input image. After that, the minutiae are extracted from fingerprint which has two types: ridge ending (1 pixels neighbourhood) and bifurcation (3 pixels neighbourhood). To generate lookup string, we combining the ridge and bifurcation In-centre points which consists of only Boolean values. Eventually, matching scheme is used for matching similarity between two strings for

identification of imposter images corresponding to stored Lookup strings. To make easier for understanding, we provide example in detail of lookup string and matching scheme.

2.10 Conclusion

In this literature review we discus about FBAS and its steps. The enhancement of the image by applying histogram equalization appears to be an important step of the process. The image binarization plays an important in the minutiae extraction. The minutiae extraction method based on grayscale (binary) image is given in [2]. The Delaunay Triangulation and the Voronoi Diagram which are two dual geometric pattern computation methods are described. When both methods are applied on a set of minutiae the obtaining result is irreversible [2]. This irreversibility property is used to enhance the security level of the system. The evaluation method and elements of a biometric system namely, GAR, FAR, FRR have been discussed in this chapter. The lookup table generation and its security during online transfer from the client to the server as discussed in [2] were reviewed. The performance of the method proposed in [2] was also discussed as illustrated by Figure 2.6.

Chapter 3

IMPLEMENTATION OF FINGERPRINT-BASED AUTHENTICATION SYSTEM

In this chapter, we focus on the explanation, the design and the implementation of a fingerprint matching algorithm. The references [1-19], all discuss about fingerprint biometric system. In most of them, the matching algorithm which represents a key point of the system, is sometime inexistent [2], or described by a pseudo code without any implementation [1-19]. Nevertheless, In this chapter a matching algorithm inspired from pseudo-code and many others documents we read so far. As we mentioned in a previous chapter, the matching algorithm is a key part of fingerprint biometric, because it aims to perform identification and verification.

3.1 Tools Used for FBAS Implementation

Below is a complete list of the materials we used for the implementation of the FBAS. The computer performance has an impact on the processing data speed. Therefore, we give below the properties of the computer used to implement and test the FBAS. The computed we used was an Intel Core i7, with a unit CPU speed of 2.6GHz. A Random Access Memory of size 8 GB; HD Graphics display card of size 10 GB. The operating system installed on it was a windows10, 64 bits. We also used fingerprint tools namely Suprema SFR300-S version 2 with Neurotic Biometric 9.0 [18], to capture some of the fingerprint images used in the experiment. All what we mentioned above is considered as the hardware properties of the used equipment's. On the other hand, we also used a grape of software. The software Matlab R2016a [14] is what we used for the main part

of our job. It was used for instance for the image processing and minutiae extraction. The software XAMPP Control Panel v3.2.2 [19] was used for the DBMS and the local server.

3.2 Design of the FBAS

In this section we discuss the design and the implementation of the FBAS. As mentioned in [2], such system can be integrated in any online system for the security purpose. The Figure 3.1 shows the functional diagram of the FBAS.

Below is the overall functional diagram of the FBAS.

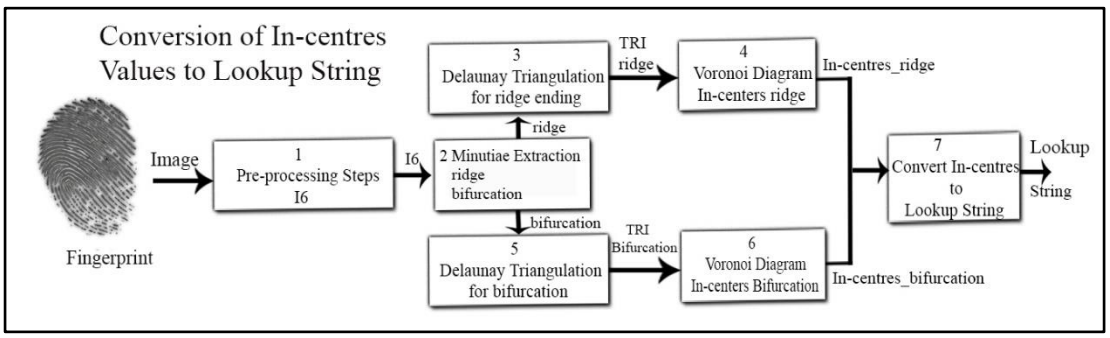


Figure 3.1: Block Diagram of Method for Conversion of In-centres Values to Lookup String

Referring to the Figure 2.1, the functionality of the system is the following. A user provides his personal data and his fingerprint image during the registration. After that, the pre-processed step are performed on input fingerprint then feature extraction process is applied as it shown in Figure 2.1(c). After that, Delaunay Triangulation and Voronoi diagram is utilized and a personal lookup table is generated, steps are shown in Figure 2.1(b). On the other hand, during identification, an impostor provides his data similarly to the registration as it describes in Figure 2.1(a). The provided fingerprint is processed then the minutiae features are extracted (illustrated in Figure 3.1) in form of ridge and bifurcation, according to pseudo code which is given in Figure

2.3. Then Delaunay Triangulation applied on both ridge and bifurcation minutiae features separately as shown in the Figure 3.1, at step 3 and 5. After that, the lookup table is extracted the In-centres value corresponding to Voronoi vertex as describe in Figure 3.1, at step 4 and 6. Then the In-centres ridge and bifurcation represented as lookup table as given in Figure 3.1, at step 7. The impostor lookup table is matched with the lookup table of the registered data, if they are similar then the matching is said to be accepted provided that the remaining data, such as name, surname matched. Otherwise, the matching is said to be rejected as it describe in Figure 2.1(a).

3.2.1 Enrolment and Identification of Fingerprint

The first stage is to enrol the fingerprint which is shown in Figure 3.1, through using SFR S300 Optical Scanner V.3.2.2 [18] and FVC2000DB1_B, FVC2002DB1_B and FVC2004DB1_B databases [18]. Then the image of fingerprint will be processed by pre-processing and feature extraction steps then the Delaunay triangulation and Voronoi vertex will be applied to obtain In-centres for lookup table to be stored in MySQL server database.

In Identification stage, the querying fingerprint will be captured and processed by the same way like the enrolment stage then it will match using similarity function as a client with the lookup table of the MySQL server database.

First step of the FBAS is lookup table which consist of pre-processing steps, Minutiae extraction, Delaunay Triangulation and Voronoi Diagram, according to Figure 2.1 (a). Therefore, the following steps describe the sub parts of lookup table:

3.2.2 Image Pre-processing and Minutiae Extraction Steps

Image pre-processing steps will be followed the block diagram of FBAS, shown in Figure 3.1.

Figure 3.2 represents the image input. This is the beginning of the process that corresponds to the acquisition on the image as it described in Figure 3.1(c) Block diagram of FBAS. First, we add path of database to connect with MySQL database, shown in Appendix A, on line 4. After that, we read image for further pre-processing steps which is shown in line 7.

Figure 3.2: The Input Image (Define by Code Appendix A, at Line 11)

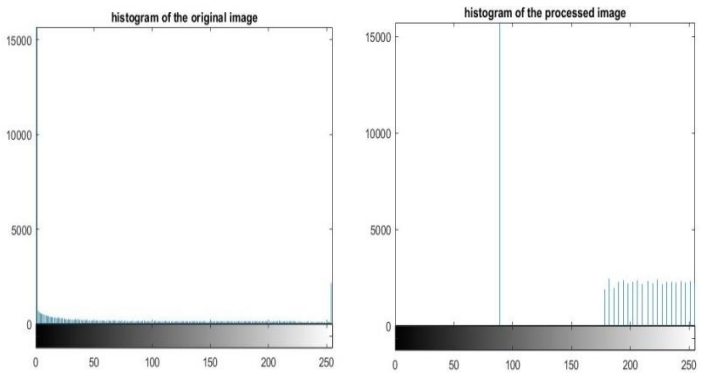
After reading input image, we use Fast Fourier transformation and Inverse Fast Fourier Transformation, formulas are described in equation (2.1 and 2.3) respectively, to enhance the image quality by specifying the k best value which is a constant, as it described in equation (2.2). However, the Fast Fourier Transform (fft2) and Inverse Fast Fourier Transform (ifft2) is applied on each block (32x32) of input image, the Matlab functions has been used, as it shown in Appendix A, on line 16 and 17 and K value is shown at line 6. The output image is shown in Figure 3.3.

Figure 3.3: Enhanced Image by Using Fast Fourier Transformation (Define by Code Appendix A, at Line 21)

After the enhancement of an image, the histogram equalization process is applied on it to enhance contrast and to obtain better feature extraction to be equally distributed by using histogram formula which shown in equation (2.7) and Matlab code of the histogram equalized image is exploit in Appendix A, at line 23-25. The output of the Histogram equalization is shown in Figure 3.4.

Figure 3.4: Histogram Equalization of Fingerprint Image (Define by Code Appendix A, at Line 24)

After that, the histogram of both original and enhanced images are plotted in Figure 3.5 and corresponding Matlab code are shown in Appendix A, at line 24 and 26 respectively.



(a) (b)

Figure 3.5: Histogram Image (a) is Original Image Histogram and (b) Enhanced Image Histogram (Define by Code Appendix A, at Line 27 and 29)

After enhancement of fingerprint image, the Binarization process is applied on it. It means, the image converted into binary image according to equation (2.8), and threshold is calculated by using *graythresh()*, the Matlab code is shown at line 30-33 in Appendix A. The output image of Binarization is described in Figure 3.6.

Figure 3.6: Binary Image of Fingerprint (Define by Code Appendix A, at Line 32)

After that, Opening Operation image or Region of Interest (ROI) is applied as it shown in Block diagram, Figure 3.1(c) step 1.4. In this process, we detect and remove the noise around the fingerprint of enhanced image by using built-in function of Matlab, described in Appendix A, at line 34-36. However, the threshold value 50 is the most

appropriate for removing the noise around the fingerprint impression as it find by conducting the several experiments. The output of ROI is shown in Figure 3.7.

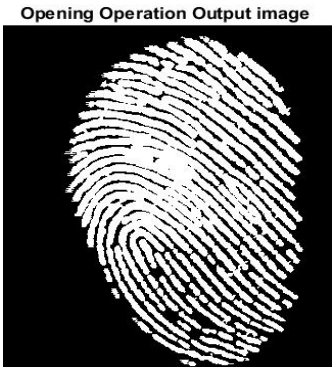


Figure 3.7: ROI Output Image (Define by Code Appendix A, at Line 35)

By following the steps of block diagram, shown in Figure 3.1 (c) step 1.5, the output of ROI is considered as input of Sobel filter. This is the edge detection filter through thinning the image to make easy detection of minutiae for further processing. The mathematical formula of Sobel filter is shown in equation (2.9) and implementation Matlab code is described at line 38-39 in Appendix A. The output is shown in Figure 3.8.

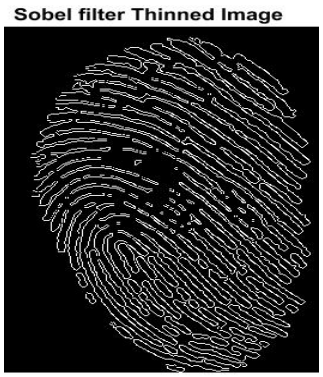


Figure 3.8: Sobel Filter Thinning Image (Define by Code Appendix A, at Line 39)

After that, by following block diagram, Figure 3.1 (b) (step 2), the extraction of minutiae from fingerprint which has two types: ridge ending (1 pixels neighbourhood)

and bifurcation (3 pixels neighbourhood), as it described Pseudo code in Figure 2.3. These minutiae are important for the lookup table generation. Where it calculate the number of neighbours if it has three adjacent neighbours it considered a bifurcation and if a pixel has just one or two adjacent neighbour/sit will be consider as ridge ending. Implementation code of this step is shown in Appendix A, at line 42-47 where this code is written according to Pseudo code shown in figure 2.3 and the output is shown in Figure 3.9.

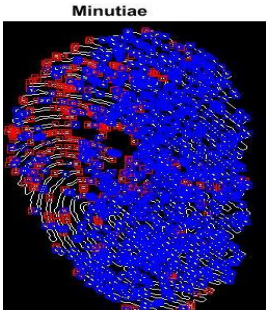
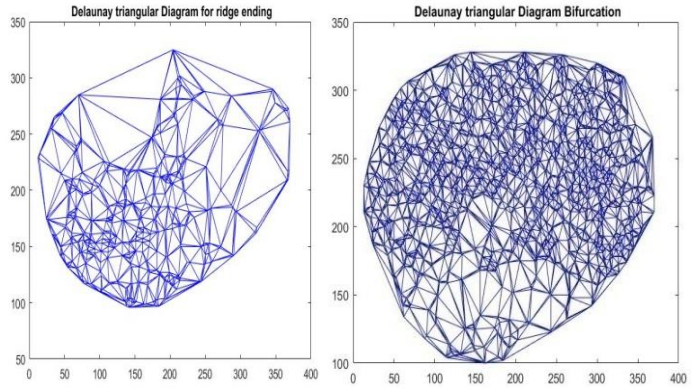


Figure 3.9: Minutiae of Fingerprint, Ridge (red) and Bifurcation (blue) (Define by Code Appendix A, at Line 74)

After that, the Delaunay Triangulation scheme is used to protect the fingerprint by making cancellable template and it applied individually on the ridge ending and bifurcation minutiae as it shown in block diagram in Figure 3.1(b) step 3 and 5. The triangulation of the ridge ending and bifurcation is computed by Matlab function, described at line 75 and 78 of Appendix A respectively. The output of Delaunay triangulation of ridge and bifurcation in Figure 3.10 (a) and (b) respectively.



(a) (b)

Figure 3.10: Delaunay Triangulation of Ridge (a) and Bifurcation (b) (Define by Code Appendix A, at Line 76 and 79)

Eventually, the Voronoi Diagram is to make indexing for Delaunay Triangulation and make irreversible template. The Voronoi Diagram of all the In-centres obtained by Delaunay triangulation of the ridge and bifurcation as it shown in block diagram in Figure 3.1(b) step 4 and 6, which the In-centres for each of ridge and bifurcation is computed separately by Matlab function *incenters(), voronoi()* that shown in Appendix A, on line 81-87 respectively. The output image is shown in Figure 3.11.

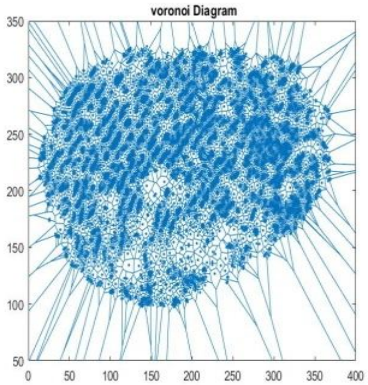


Figure 3.11: Voronoi Diagram of In-centres of Bifurcation and Ridge (Define by Code Appendix A, at Line 86)

3.2.3 Method for Conversion of In-centres to Lookup String

After getting In-centres values, the lookup table is generated as it shown in block diagram Figure 3.1, on step 7. The FBAS lookup string represent by Boolean string values of the In-centres values corresponding to Voronoi vertex which are stored in MySQL database. Where the In-centres ridge represented by zeros and In-centres bifurcation represented by ones then In-centres values are concatenated as it in Appendix A, on lines 98-123. Then In-centres are sequenced according to Lexicographical order (illustrated in Example 3.1). The Lexicographical order are values from lower x coordinate (Appendix A, at line 105) value to higher x coordinate value if the x coordinates value of In-centres are equal (Appendix A, at line 109-110) then it will be order according to y coordinate value. Then it will be stored in MySQL database in form of string corresponding to username.

Example 3.1: Lexicographical order description according to Matlab code in Appendix A, on lines 98-123.

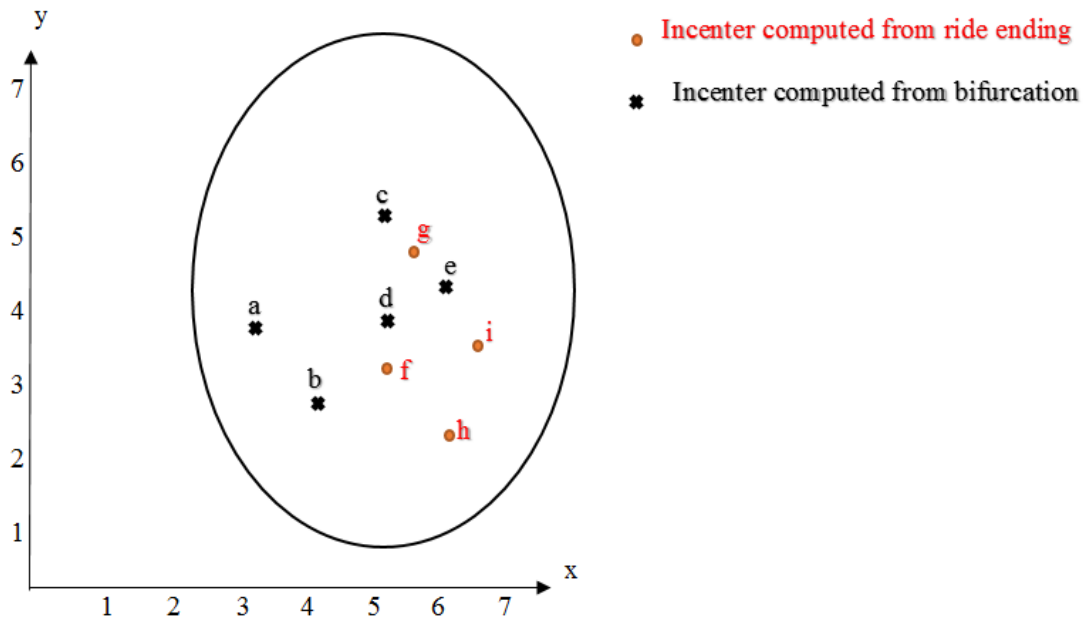


Figure 3.12: Example of the Lexicographical Order

Consider the image above representing the In-centres computed from the both types of minutiae, which are bifurcation and ridge ending.

Definition: Lexicographical order is done as follows

The first point or minimum is the point with the smallest x-value and the next is the point which x-value follows the previous smallest x-value and so forth. If two points have the same x-value, then the smallest among them is the one with the smallest y-value.

Using the definition above, the lexicographical order of the In-centres computed from ridge ending is as **f, g, h, I** and the lexicographical order of the In-centres computed from bifurcation is **a, b, c, d, and e**.

If we consider their coordinates value, we have the following for bifurcation

Table 3.1: Bifurcation Values According to Figure 3.12

point	f	g	h	i
x-value	5	5.5	6	6.5
y-value	3.5	5	2.5	3.5
key	**1**	**1**	**1**	**1**

We identify with the value '**1**' all the bifurcation in-centres. If we consider their coordinates value, we have the following for ridge ending

Table 3.2: Ridge Values According to Figure 3.12

point	a	b	c	d	e
x-value	3	4	5	5	6
y-value	4	3	4	5.5	4.5
key	**0**	**0**	**0**	**0**	**0**

At this points both types of minutiae are ordered following lexicographical order. It is clear that the distribution of the In-centres is random. We can perform the general lexicographical order on both types of minutiae merged together. While doing this we have to keep their respective key values. The new ordered table is then

Table 3.3: Lexicographical Order of Ridge and Bifurcation Values According to Figure 3.12

point	a	b	f	c	d	g	h	e	i
x-value	3	4	5	5	5	5.5	6	6	6.5
y-value	4	3	3.5	4	5.5	5	2.5	4.5	3.5
key	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**0**	**1**

Now the key string **001001101**, represents our lookup table that is what is stored on the database.

3.2.4 Matching Method

After storing the Boolean value inform of string in database, the next step is identification of imposter users corresponding to register users as it shown in Figure 2.1(a). Hence, we define the matching scheme for identification of our system as follow: the similarity function (defined in Figure 3.13) check similarity of strings between the stored lookup string (S1) in MySQL Database and the lookup string (S2) of querying fingerprint. The similarity function will find the level of similarity between the two lookup strings stored in form of string S1 and S2. Three operation are used to evaluate the similarity. There are insertion, deletion and substitution of characters in either of the strings to make them fully similar. The similarity of the two strings is therefore evaluated based on the number or percentage of characters that make them to be not similar. The percentage of similarity among the strings can evaluated at this stage as a ratio of the length of the similar substring by the length of either of these strings. The output will return the number of dissimilar string which is describes in

example 3.2. After that, the score will be the key value for decision that the input fingerprint of imposter user is genuine or fake. For this purpose, this algorithm describes the threshold value which based on the database to make decision that the input image is genuine or fake. The implementation of matching scheme is shown in Appendix C and the example of threshold is based conducting experiments on different fingerprints image resolution which are considered and discussed in next Chapter.

Algorithm Levenshtein Distance Steps

1 Set n to be the length of S1.
Set m to be the length of S2.
If n = 0, return m and exit.
If m = 0, return n and exit.
Construct a matrix containing 0..m rows and 0..n columns.
2 Initialize the first row to 0..n.
Initialize the first column to 0..m.
3 Examine each character of s (i from 1 to n).
4 Examine each character of t (j from 1 to m).
5 If s[i] equals t[j], the cost is 0.
If s[i] doesn't equal t[j], the cost is 1.
6 Set cell d[i,j] of the matrix equal to the minimum of:
The cell immediately above plus 1: d[i-1,j] + 1.
The cell immediately to the left plus 1: d[i,j-1] + 1.
The cell diagonally above and to the left plus the cost: d[i-1,j-1] + cost.
7 After the iteration steps (3, 4, 5, 6) are complete, the distance is found in cell d[n,m].

Figure 3.13: Steps of Levenshtein Distance Method [20] and Implemented in Appendix C

Example 3.2

To make easy for understanding, we describe the matching algorithm “Levenshtein Distance” in this example by following steps of Figure 3.13. This example shows how the Levenshtein distance [20] is computed when the source string is "GUMBO" and the target string is "GAMBOL". Steps of matching similarities between two strings are following:

Steps 1 and 2:

		G	U	M	B	O
	0	1	2	3	4	5
G	1					
A	2					
M	3					
B	4					
O	5					
L	6					

Steps 3 to 6 when i = 1:

		G	U	M	B	O
	0	1	2	3	4	5
G	1	0				
A	2	1				
M	3	2				
B	4	3				
O	5	4				
L	6	5				

Steps 3 to 6 when i = 2:

		G	U	M	B	O
	0	1	2	3	4	5
G	1	0	1			
A	2	1	1			
M	3	2	2			
B	4	3	3			
O	5	4	4			
L	6	5	5			

Steps 3 to 6 when i = 3:

		G	U	M	B	O
	0	1	2	3	4	5
G	1	0	1	2		
A	2	1	1	2		
M	3	2	2	1		
B	4	3	3	2		
O	5	4	4	3		
L	6	5	5	4		

Steps 3 to 6 when i = 4:

		G	U	M	B	O
	0	1	2	3	4	5
G	1	0	1	2	3	
A	2	1	1	2	3	
M	3	2	2	1	2	
B	4	3	3	2	1	
O	5	4	4	3	2	
L	6	5	5	4	3	

Steps 3 to 6 when i = 5:

		G	U	M	B	O
	0	1	2	3	4	5
G	1	0	1	2	3	4
A	2	1	1	2	3	4
M	3	2	2	1	2	3
B	4	3	3	2	1	2
O	5	4	4	3	2	1
L	6	5	5	4	3	2

Step 7:

Hence, the distance between two strings (GUMBO and GAMOL), shown in lower right corner which is 2 because U is substituted with A and adding L is new insertion in GAMOL. So this is 2 changes (one substitution and one Insertion).

3.3 Conclusion

In this chapter, we described in details the structure of FBAS and our proposed lookup string and example of the lookup string is shown in an example 3.1. To obtain the Lookup string, we implement the pre-processing steps required to process the image from the input image. After that, the minutiae are extracted from fingerprint which has two types: ridge ending (1 pixels neighbourhood) and bifurcation (3 pixels neighbourhood). Then lookup string is generated by combining the ridge ending and bifurcation in-centres. It means, lookup string contains only Boolean value. We implemented the matching scheme using Levenshtein Distance Method to measure the

similarity between two lookup string (one is stored in database and another is querying lookup string) for identification of imposter fingerprint images. Additionally, we explain the Levenshtein Distance Method in detail and for making easier to understand, we described it in the example 3.2.

Chapter 4

EXPERIMENTS ON FBAS

First we explain experiment setup for the FBAS designed and implemented in Chapter 3. Section 4.2 presents experiments results on FBAS.

4.1 Experiment Setup for FBAS

The experiment was conducted on fingerprint of 3 databases. We also used some fingerprint image provided to us by some volunteer and the results were satisfactory. The experiment result is consigned below. Let us also remind how the string similarity method works. Given two strings with more or less same size, the string similarity method, compute the number of character change (Add or deletion of characters), required in order for the two string to become similar. This means the closer are the string length, the better is the similarity result. The dimension and resolution for each sample of 3 databases and real time application shown as below:

- FVC2000DB1_B: 300 x 300 pixels dimension, 96 PPI resolution.
- FVC2002DB1_B: 388 x 374 pixels dimension, 96 PPI resolution.
- FVC2000DB1_B: 640x 480 pixels dimension, 96 PPI resolution.
- Real time application: 288 x 320 pixels dimension, 500 PPI resolution
    - PPI: (Pixels per inch)

For these databases, we specified the threshold values for acceptance and rejection of input fingerprint image by conducting experiments, variation of image resolution and

dimension. Hence, the threshold value is varied for each database because of it resolution and dimension and results according to these thresholds are shown in Figure 4.1, 4.2, and 4.3.

- FVC2000DB1_B: Threshold value=2000, 1800 and 1100 characters
- FVC2002DB1_B: Threshold value=1500, 2000 and 1700 characters
- FVC2000DB1_B: Threshold value=1500, 1400 and 1350 characters
- Real time application: Threshold value=1300 characters

The code for the connection of databases is shown in Appendix A.

Databases FVC2000DB1_B, FVC2002DB1_B and FVC2000DB1_B are made of 80 fingerprint templates captured from 10 individuals. This means each individual provided 8 different impression of the same fingerprint. On fingerprint sample is stored in the database and the 79 remaining fingerprint query the database for the matching process. The system will said to be 100% efficient, if out 79 templates, 72 are rejected during matching process and the remaining 7 are accepted as genuine. In either case, there might some false acceptance or / and false rejection. The system performance is thus defined base on these elements. The result is obtained for FVC2000DB1_B from the simulation is given in the Table 4.1.

Table 4.1: Result of Simulation Using Database FVC2000DB1_B

Experiments	Dataset	Threshold	TA	TR	FA	FR	GAR	FAR	FRR
1	101_8	<2000	3	4	10	62	82.29%	12.65%	5.06%
2	101_8	<1800	2	5	9	63	82.29%	11.39%	6.32%
3	101_8	<1100	1	6	0	72	92.41%	0%	7.59%

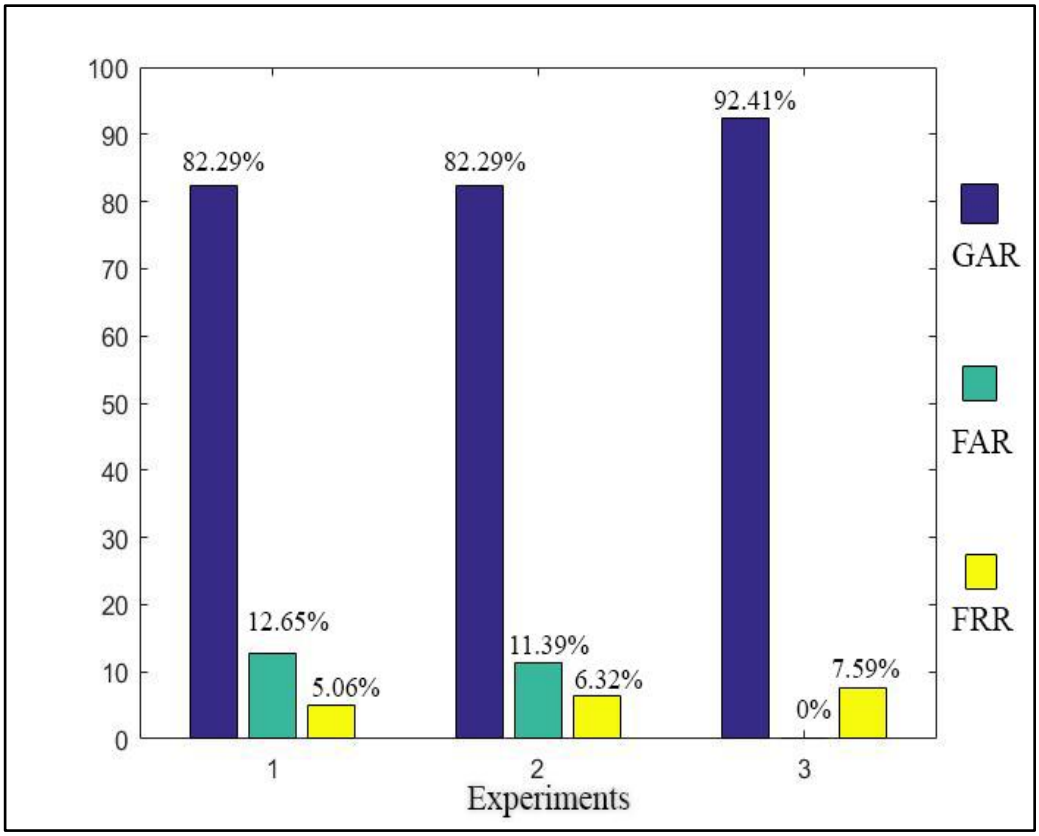


Figure 4.1: Result of FVC2000DB1_B with a Threshold Value of 2000 in (1), 1800 in (2) and 1100 Characters in (3)

The Figure 4.1 represents the performance of the system based on various threshold values. The bar chart group 1, 2 and 3 represent the results for the threshold values 2000, 1800 and 1100 characters respectively, for the database's FVC2000DB1_B. While GAR is Genuine Acceptance Rate, FAR is False Acceptance Rate and FRR is False Rejection Rate as they are described in Table 4.1.

Database FVC2002DB1_B: Similarly as the first database, the result is obtained from the simulation is given in the Table 4.2.

Table 4.2: Result of Simulation Using Database FVC2002DB1_B

Experiments	Dataset	Threshold	TA	TR	FA	FR	GAR	FAR	FRR
1	101_1	<1500	2	5	0	72	93.68%	0%	6.32%
2	101_1	<2000	5	2	5	67	91.15%	6.32%	2.53%
3	101_1	<1700	4	3	1	71	94.95%	1.26%	3.79%

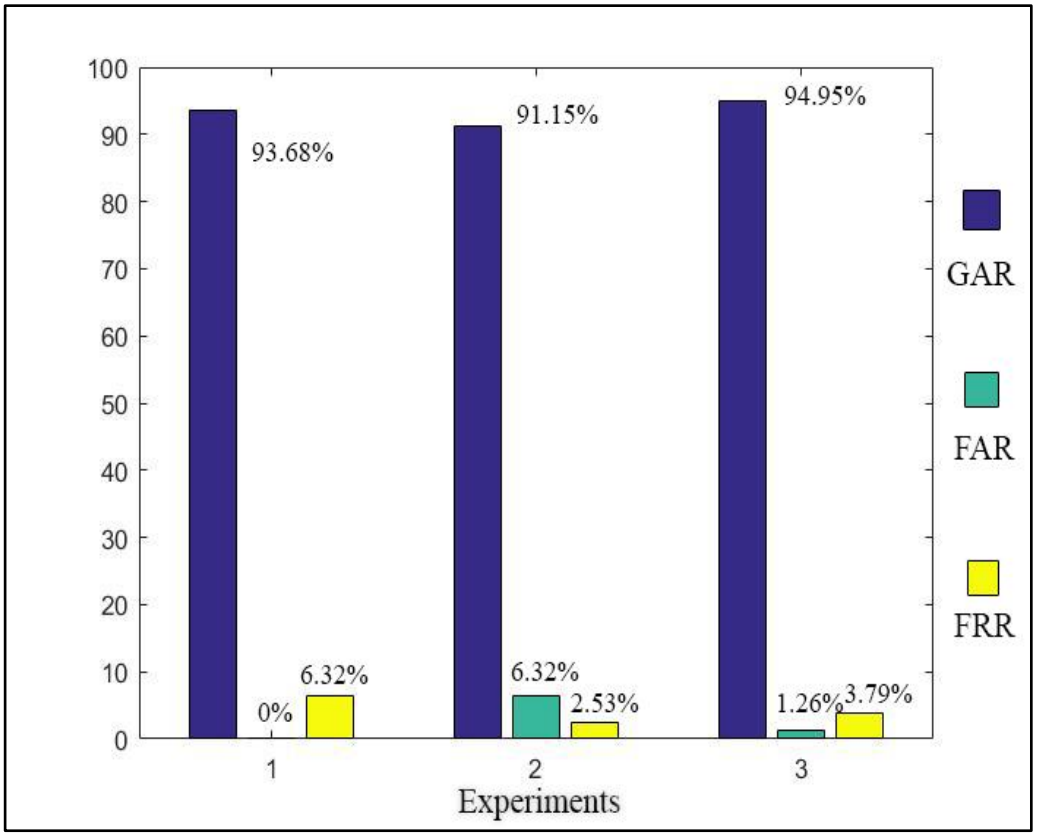


Figure 4.2: Result of FVC2002DB1_B with a Threshold Value of 1500 in (1), 2000 in (2) and 1700 Characters in (3)

The Figure 4.2 represents the performance of the system based on various threshold values. The bar chart group 1, 2 and 3 represent the results for the threshold values of 1500, 2000and 1700 characters respectively for the database's FVC2002DB1_B and corresponding GAR, FAR and FRR is shown as well.

Database FVC2004DB1_B: The result is obtained from the simulation is given in the Table 4.3.

Table 4.3: Result of Simulation Using Database FVC2004DB1_B

Experiments	Dataset	Threshold	TA	TR	FA	FR	GAR	FAR	FRR
1	101_2	<1500	4	3	15	57	77.23%	18.98%	3.79%
2	101_2	<1400	2	5	8	64	83.56%	10.12%	6.32%
3	101_2	<1350	1	6	6	66	84.82%	7.59%	7.59%

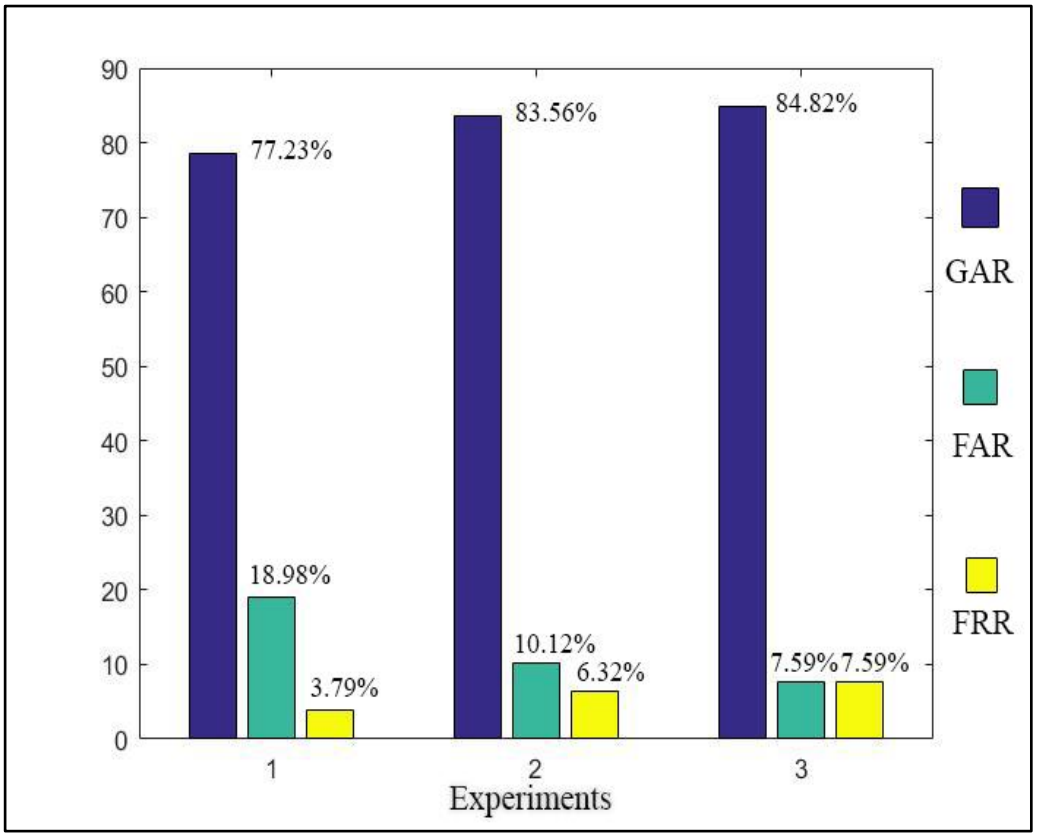


Figure 4.3: Result of FVC2004DB1_B with a Threshold Value of 1500 in (1), 1400 in (2) and 1350 Characters in (3)

The Figure 4.3 represents the performance of the system based on various threshold values. The bar chart group 1, 2 and 3 represent the results for the threshold values of 1500, 1400 and 1350 characters respectively for the database's FVC2002DB1_B and their corresponding GAR, FAR and FRR is shown as well.

Real time application: In order to test the accuracy of the method on a real life data, we used a fingerprint device of type, SUPREMA SFR300-S. The implementation is

tested on thirteen Individuals where 12 are university students and one is employee. Each individual participate with three fingerprint impressions where two of those fingerprints impressions are index fingerprints and another is middle fingerprint of right hand. The experiment results are obtained by matching against one impression of Ridwan genuine which is index fingerprint of left hand. According to experiment on real time application, the 3 genuine fingerprint impressions are true accepted as genuine individual is shown in Table 4.4 in experiments (7, 8, and 9) and 32 fingerprint impressions of imposter true rejected, since the GAR obtained is 89.75% and none of genuine is false rejected so the FRR is 0%. The remaining 4 fingerprint impressions of imposter are false accepted which obtained the FAR is 10.25%. The GAR, FAR and FRR are calculated according to equation from (2.11), (2.12) and (2.13) respectively.

Table 4.4: Result of Simulation Using Real Time Fingerprints with a Threshold Value of 1300

Experiments	Datasets	Similarity Result (dissimilar values)	Acceptance Rate <1300
1	Altaf1	4565	Rejected
2	Altaf2	4089	Rejected
3	Altaf3	4428	Rejected
4	Baran1	2325	Rejected
5	Baran2	3191	Rejected
6	Baran3	4749	Rejected
7	Ridwan1	1040	Accepted
8	Ridwan2	1254	Accepted
9	Ridwan3	1128	Accepted
10	Dler1	3921	Rejected
11	Dler2	4776	Rejected
12	Dler3	4498	Rejected
13	Faheem1	4909	Rejected
14	Faheem2	3863	Rejected
15	Faheem3	4586	Rejected
16	John1	4368	Rejected
17	John2	1271	Accepted
18	John3	4239	Rejected
19	Nasir1	3583	Rejected
20	Nasir2	4085	Rejected
21	Nasir3	1351	Rejected
22	Pawan1	3911	Rejected
23	Pawan2	1432	Rejected
24	Pawan3	1238	Accepted
25	Rasheed1	4246	Rejected
26	Rasheed2	4113	Rejected
27	Rasheed3	3757	Rejected
28	Sajjad1	1238	Accepted
29	Sajjad2	4263	Rejected
30	Sajjad3	4506	Rejected
31	Sakib1	2431	Rejected
32	Sakib2	4766	Rejected
33	Sakib3	4623	Rejected
34	Zarak1	4278	Rejected
35	Zarak2	3825	Rejected
36	Zarak3	4540	Rejected
37	Yalkin1	1509	Rejected
38	Yalkin2	4196	Rejected
39	Yalkin3	1114	Accepted

4.2 Comparison Results with [2]

We evaluated the performance of the FBAS with scheme and compared the results with a state-of-art approach [2]. For instance using the Database FVC2002DB1_B, with a threshold value 1700 characters lead us to a GAR=94.93%, FRR=3.79% and

FAR=1.26%. Whereas using the same database, [2] provided a GAR=93%, FRR=3% and FAR=4%.

Database FVC2000DB1_B**,** with a threshold value 1100 characters lead us to a GAR=92.40%, FRR=7.59% and FAR=0%. Whereas using the same database, [2] provided a GAR=96%, FRR=2% and FAR=2%.

Database FVC2004DB1_B with a threshold value of 1350 characters lead us to a GAR=84.81%, FRR=7.59% and FAR=7.59%. Whereas using the database, [2] provided a GAR=88%, FRR=4.50% and FAR=7.5%.

Moreover, we applied a real Implementation to find out the performance of our system for FBAS. The results of 13 individuals fingerprint impressions with threshold less than 1300 characters shows GAR=89.75%, FAR=10.25% and FRR=0%.

4.3 Conclusion

Hence, the comparative results of our approach with start-of-art scheme [2] are following: In database FVC2002DB1_B, our approach performance overcomes with GAR=1.93% and in database FVC2000DB1_B their scheme overcomes by GAR=3.6% and for database FVC2004DB1_B their result also overcomes by GAR=3.19%. Additionally our real time application shows significant performance such as GAR=89.75%, FAR=10.25% and FRR=0%.

Chapter 5

CONCLUSION AND FUTURE WORK

We study and analyzed the FBAS and come up with the conclusion that a fingerprint-based authentication is very important. FBAS is implemented as including pre-processing steps such as the image enhancement Fourier Transform, histogram equalization, binarization, ROI, and Sobel Filter. The image binarization plays an important role in the minutiae extraction. The Delaunay Triangulation and the Voronoi Diagram which are two dual geometric pattern computation methods are implemented for protecting the biometrics template which consists of In-centres points for each minutiae separately. We assigned ridge minutiae by zeros and bifurcation minutiae by ones then order them by lexicographical order then concatenate them into one long string to save it in lookup string that stored in MySQL database. For identification, we used similarity matching function to check number of dissimilar characters between lookup strings and querying lookup string of fingerprint impressions. Hence, this process is applied on FVC2000DB1_B, FVC2002DB1_B and FVC2004DB1_B databases which is also used by existing method. Additionally, we applied our FBAS on real time application which show reliable results. In database FVC2002DB1_B, our approach overcomes on existing method with GAR=1.95% and in database FVC2000DB1_B and FVC2004DB1_B their scheme overcomes by GAR=3.59% and GAR=3.18% respectively. Additionally, our real time scenario shows the significant performance such as GAR=89.75%, FAR=10.25% and FRR=0%. However, above

results have been obtained by considering several thresholds values, while it is not in the referred study.

My future work is to extend the performance of multiple fingerprint impression and mutual authentication by using additional human characteristic such as iris and face recognition.

REFERENCES

[1] Rafael C. G., Richard E. W. (1992). "*Digital Image Processing",* 2nd edition, Prentice Hall.

[2] Vigila, S. A. M. C., Muneeswaran, K., & Antony, W. T. B. A. (2015). Biometric security system over a finite field for mobile applications. *IET Information Security*, *9*(2), (pp. 119-126).

[3] Snelick R., Uludag U., Mink A., Indovina M., & JainS A. (2005). Large-scale evaluation of multimodal biometric authentication using state-of-the-art systems. *I IEEE Transactions on Pattern Analysis and Machine Intelligence* Computer Society Washington, DC, USA, IEEE, Volume: 27, Issue: 3, (pp. 450-455).

[4] Martiri E., Gomez-B., M. Yang, B. & Busch, C. (2016). Biometric template protection based on Bloom filters and honey templates. *IET Biometrics,* 6(1), (pp. 19-26).

[5] Wang S., & Hu J. (2012). Alignment-free cancelable fingerprint template design: A densely infinite-to-one mapping (DITOM) approach. Pattern Recognition, 45(12), (pp. 4129-4137).

[6] Sandhya M., Prasad M. V., & Chillarige R. R. (2016). Generating cancellable fingerprint templates based on Delaunay triangle feature set construction. *IET Biometrics,* 5(2), (pp. 131-139).

[7] Xi, K., Ahmad T., Han F., & Hu J. (2011). A fingerprint based bio-cryptographic security protocol designed for client/server authentication in mobile computing environment. *Security and Communication Networks,* 4(5), (pp. 487-499).

[8] Lupu C., Găitan, V. G., & Lupu V. (2015). Fingerprints used for security enhancement of online banking authentication process. *In Electronics, Computers and Artificial Intelligence* (ECAI), 2015 7th International Conference on, IEEE, (pp. 217-220).

[9] Mahajan, Priyanka (2016). Secured Internet Banking Using Fingerprint Authentication, *The International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE),* Vol. 4. ISSN (Online): 2320-9801, ISSN (Print): 2320-9798.

[10] Jin Z., Teoh, A. B. J., Ong T. S., & Tee C. (2010). A Revocable Fingerprint Template for Security and Privacy Preserving. *Interactive Intelligent Systems (TIIS),* 4(6), (pp. 1327-1342).

[11] Luo X., Tian J., & Wu Y. (2000). A minutiae matching algorithm in fingerprint verification. In Pattern Recognition, 2000. Proceedings. 15th International Conference on Barcelona, Spain, *IEEE,* Vol. 4, (pp. 833-836).

[12] India semniconductor forum. [Online] (2017, Jamuary 15) Retrieved from http://www.indiasemiconductorforum.com/physical-access/12582-isf-apps-identification-physical-access-biometrics-ti-block-diagram.html.

[13] Sneddon I. N. (1995). *Fourier transforms*. Courier Corporation.

[14] MATLAB Central - MathWorks. [Online] MathWorks, (2017, January 2). Retrieved from https://mathworks.com/matlabcentral/.

[15] Sobel Edge Detector, [Online] (2017, February 25) Retrieved from http://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm.

[16] Amira M. A. M. S., (2011). Enhanced Secure Algorithm for Fingerprint Recognition, PhD thesis, Ain Shams University, Cairo, Egypt.

[17] Berg, Mark de (2008). Computational Geometry Algorithms and Applications. Springer-Verlag Berlin Heidelberg. ISBN 978-3-540-77973-5.

[18] Suprema**.** Neuro Technology. *Neuro Technology.* [Online] (2017, January 18). Retrieved from http://www.neurotechnology.com/fingerprint-scanner-suprema-sfr300-s.html.

[19] Apache**.** XAMPP Apache + MariaDB + PHP + Perl. XAMPP Apache + MariaDB + PHP + Perl. [En ligne] Apachefriends, (2017 February 01). Retrieved from https:// www.apachefriends.org/index.html.

[20] Vladimir Levenshtein, (1965), Levenshtein Distance algorithm, [Online] (2017, February 18). Retrieved from http://people.cs.pitt.edu/kirk/cs1501/Pruhs/ Fall2006/ Assignments/editdistance/Levenshtein%20Distance.htm.

APPENDICES

Appendix A: Implementation Code of Enrolment

```
clear all;
close all;
clc;
addpath('database');
block_size = 32;
k = 0.0001;
Img = imread('102_3.tif');
[r c] = size(Img);
I = 255 - imresize( Img , [ ( r/block_size ) *  block_size , ( c/block_size ) * block_size] );
[r c] = size(I);
figure;imshow(I);
title('Input Image');

G = double( I );
for i=1:( r/block_size )
for j=1:( c/block_size )
Y = fft2( I( (i-1)*block_size + 1 : i * block_size , (j-1)*block_size + 1 : j * block_size ) );
G( (i-1)*block_size + 1 : i * block_size , (j-1)*block_size + 1 : j * block_size )  = ifft2( Y * ( abs(Y) ^ k ) );
end
end
G = uint8( real ( G ) );
figure;imshow( G );
title('Enhanced Image');

J = histeq( G );
figure;imshow( uint8( J ) );
title('Histogram equalised image');

figure; imhist(G)
title('histogram of the original image');

figure; imhist(uint8( J ))
title('histogram of the processed image');

level = graythresh(uint8( J ));
B = im2bw( uint8( J ),level);
figure;imshow( B );
title('Binary image');

B1 = bwareaopen( B, 50 );
figure;imshow( B1 );
title('Opening Operation Output image');

[Gmag,Gdir]= imgradient(B1);
```

```
thin_image = edge(B1,'Sobel',[],'both');

figure;imshow(thin_image);title('Sobel filter Thinned Image');

bifurcation_minutia=[];
ridge_ending=[];
for i=4:r-4
for j=4:c-4
if thin_image(i,j)==1
Neighbours_num = sum ( sum( thin_image(i-1:i+1 , j-1 : j+1 ) ) ) - 1;
if  Neighbours_num== 3
bifurcation_minutia = [bifurcation_minutia ; i , j ,Gdir(i,j) ];
elseif Neighbours_num== 1              ridge_ending = [ridge_ending ; i , j
,Gdir(i,j) ];
end
end
end
end


Img_disp = zeros(r,c,3);
Img_disp(:,:,1) = thin_image .* 255;
Img_disp(:,:,2) = thin_image .* 255;
Img_disp(:,:,3) = thin_image .* 255;
len=length( ridge_ending );

for i=1:len
Img_disp( (ridge_ending( i , 1 )-3):(ridge_ending( i , 1 )+3),(ridge_ending( i
, 2 )-3),2:3)=0;
Img_disp( (ridge_ending( i , 1 )-3):(ridge_ending( i , 1 )+3),(ridge_ending( i
, 2 )+3),2:3)=0;
Img_disp( (ridge_ending( i , 1 )-3),(ridge_ending( i , 2 )-3):(ridge_ending( i ,
2 )+3),2:3)=0;
Img_disp( (ridge_ending( i , 1 )+3),(ridge_ending( i , 2 )-3):(ridge_ending( i
, 2 )+3),2:3)=0;

Img_disp( (ridge_ending( i , 1 )-3):(ridge_ending( i , 1 )+3),(ridge_ending( i
, 2 )-3),1)=255;
Img_disp( (ridge_ending( i , 1 )-3):(ridge_ending( i , 1 )+3),(ridge_ending( i
, 2 )+3),1)=255;
Img_disp( (ridge_ending( i , 1 )-3),(ridge_ending( i , 2 )-3):(ridge_ending( i ,
2 )+3),1)=255;
Img_disp( (ridge_ending( i , 1 )+3),(ridge_ending( i , 2 )-3):(ridge_ending( i
, 2 )+3),1)=255;
end


len=length(bifurcation_minutia);

for i=1:len
```

```
70  Img_disp((bifurcation_minutia( i , 1 )-3):(bifurcation_minutia( i , 1
    )+3),(bifurcation_minutia( i , 2 )-3),1:2)=0;
71  Img_disp((bifurcation_minutia( i , 1 )-3):(bifurcation_minutia( i , 1
    )+3),(bifurcation_minutia( i , 2 )+3),1:2)=0;
72  Img_disp((bifurcation_minutia( i , 1 )-3),(bifurcation_minutia( i , 2 )-
    3):(bifurcation_minutia( i , 2 )+3),1:2)=0;
73  Img_disp((bifurcation_minutia( i , 1 )+3),(bifurcation_minutia( i , 2 )-
    3):(bifurcation_minutia( i , 2 )+3),1:2)=0;

74  Img_disp((bifurcation_minutia( i , 1 )-3):(bifurcation_minutia( i , 1
    )+3),(bifurcation_minutia( i , 2 )-3),3)=255;
75  Img_disp((bifurcation_minutia( i , 1 )-3):(bifurcation_minutia( i , 1
    )+3),(bifurcation_minutia( i , 2 )+3),3)=255;
76  Img_disp((bifurcation_minutia( i , 1 )-3),(bifurcation_minutia( i , 2 )-
    3):(bifurcation_minutia( i , 2 )+3),3)=255;
77  Img_disp((bifurcation_minutia( i , 1 )+3),(bifurcation_minutia( i , 2 )-
    3):(bifurcation_minutia( i , 2 )+3),3)=255;
78  end
79  figure;imshow(Img_disp);title('Minutiae');


80  TRI_ridge = DelaunayTri(ridge_ending( : , 1 ),ridge_ending( : , 2 ));
81  figure;triplot(TRI_ridge);
82  title(' Delaunay triangular Diagram for ridge ending');
83  TRI_bifurcation = DelaunayTri(bifurcation_minutia( : , 1
    ),bifurcation_minutia( : , 2 ));
84  figure;triplot(TRI_bifurcation);
85  title(' Delaunay triangular Diagram Bifurcation');

86  Incentres_ridge = zeros( length( TRI_ridge( :,1 ) ) , 2 );
87  Incentres_ridge = incenters(TRI_ridge);
88  Incentres_bifurcation = zeros( length( TRI_bifurcation( :,1 ) ) , 2 );
89  Incentres_bifurcation = incenters(TRI_bifurcation);
90  Incentres=[Incentres_ridge;Incentres_bifurcation];
91  figure;voronoi(Incentres(:,1) , Incentres(:,2) );
92  title(' voronoi Diagram');
93  Incentres = [];

94  Incentres_ridge(:,3)=zeros;
95  Incentres_bifurcation(:,3)=ones;
96  Incentres=[Incentres_ridge;Incentres_bifurcation];
97  I_centres_numbers=length(Incentres); %

98  IC=Incentres;
99  IC(:,4)=zeros;

100         for nm=1:size(IC(:,1))
101         swap_min=IC(nm,1:3);
102         swap_min_indx=nm;
```

```
103     for m=nm+1:size(IC(:,1))
104     if(IC(m,4)==0)
105     if(IC(m,1)<swap_min(1))
106     swap_value=swap_min;
107     swap_min=IC(m,1:3);
108     IC(m,1:3)=swap_value;
109     elseif(IC(m,1)==swap_min(1))
110     if(IC(m,2)<swap_min(2))
111     swap_value=swap_min;
112     swap_min=IC(m,1:3);
113     IC(m,1:3)=swap_value;
114     End
115     End
116     End
117     End
118     IC(nm,4)=1;
119     IC(nm,1:3)=swap_min;
120     swap_value=[];
121     swap_min=[];
122     End
123     IC(:,4)=[];

124     I_centres_DB=IC(:,3);
125     I_centres_DB=(num2str(I_centres_DB))'
126     total_points=length(I_centres_DB);
127     user_name='fp2102_3';

128     connection=connect_dbclient();

129     colnames={'id' 'username' 'I_centres_DB' 'total_points'};
130     data={'' user_name I_centres_DB total_points};
131     datainsert(connection,'register',colnames,data);
```

Appendix B: Implementation Code of Identification

```
clear all;
close all;
clc;
warning('off','all')
c1=clock;
addpath('Real Expriment\');
block_size = 32;
k = 0.0001;
connection=connect_dbclient();
user_name='"redwan"';
sqlquery = ['SELECT I_centres_DB FROM register WHERE username = ' user_name];
curs = exec(connection,sqlquery);
curs = fetch(curs);
I_centres_DB=cell2mat(curs.Data);


tcount(10,2)=zeros;
tcc=1;
Img = imread('Yalkin3.jpg');
[r c] = size(Img);

I = 255 - imresize( Img , [ ( r/block_size ) *  block_size , ( c/block_size ) * block_size] );
[r c] = size(I);
G = double( I );
for i=1:( r/block_size )
for j=1:( c/block_size )
Y = fft2( I( (i-1)*block_size + 1 : i * block_size , (j-1)*block_size + 1 : j * block_size ) );
G( (i-1)*block_size + 1 : i * block_size , (j-1)*block_size + 1 : j * block_size )  = ifft2( Y * ( abs(Y) ^ k ) );
end
end
G = uint8( real ( G ) );
J = histeq( G );
level = graythresh(uint8( J ));
B = im2bw( uint8( J ),level);
B1 = bwareaopen( B, 50 );
[Gmag,Gdir]= imgradient(B1);
thin_image = edge(B1,'Sobel',[],'both');
bifurcation_minutia=[];
ridge_ending=[];
for i=4:r-4
for j=4:c-4
if thin_image(i,j)==1
```

```
        a. Neighbours_num = sum ( sum( thin_image(i-1:i+1 , j-1 : j+1 ) ) ) - 1;
        b. if  Neighbours_num== 3
        c. bifurcation_minutia = [bifurcation_minutia ; i , j ,Gdir(i,j) ];
        d. elseif Neighbours_num== 1% || Neighbours_num== 2
        e. ridge_ending = [ridge_ending ; i , j ,Gdir(i,j) ];
        f. end
40  end
41  end
42  end


43  TRI_ridge = DelaunayTri(ridge_ending( : , 1 ),ridge_ending( : , 2 ));
44  TRI_bifurcation = DelaunayTri(bifurcation_minutia( : , 1
    ),bifurcation_minutia( : , 2 ));
45  Incentres_ridge = zeros( length( TRI_ridge( :,1 ) ) , 2 );
46  Incentres_ridge = incenters(TRI_ridge);
47  Incentres_bifurcation = zeros( length( TRI_bifurcation( :,1 ) ) , 2 );
48  Incentres_bifurcation = incenters(TRI_bifurcation);
49  Incentres=[Incentres_ridge;Incentres_bifurcation];
50  Incentres = [];

51  Incentres_ridge(:,3)=zeros;
52  Incentres_bifurcation(:,3)=ones;
53  Incentres=[Incentres_ridge;Incentres_bifurcation];
54  I_centres_numbers=length(Incentres);
55  IC=Incentres;
56  IC(:,4)=zeros;
57  for nm=1:size(IC(:,1))

58  swap_min=IC(nm,1:3);
59  swap_min_indx=nm;

60  for m=nm+1:size(IC(:,1))
61  if(IC(m,4)==0)
62  if(IC(m,1)<swap_min(1))
63  swap_value=swap_min;
64  swap_min=IC(m,1:3);

65  IC(m,1:3)=swap_value;
66  elseif(IC(m,1)==swap_min(1))
67  if(IC(m,2)<swap_min(2))
68  swap_value=swap_min;
69  swap_min=IC(m,1:3);

70  IC(m,1:3)=swap_value;

71  end
72  end
73  end
74  end
```

```
IC(nm,4)=1;
IC(nm,1:3)=swap_min;
swap_value=[];
swap_min=[];

end

IC(:,4)=[];
I_centres_L=IC(:,3);
I_centres_L=(num2str(I_centres_L))';
Ratio=abs(length(I_centres_L)-length(I_centres_DB));
s1=I_centres_DB;
s2=I_centres_L;
score=STRING_SIMILARITY(s1,s2)
if(score<threshold)
    Yes=Accepted
else
    No=Rejected
end
```

Appendix C: Implementation Code of Matching Similarity

```
function [score] = STRING_SIMILARITY( s1,s2 )
if length(s1) < length(s2)
score = STRING_SIMILARITY(s2, s1);
elseif isempty(s2)
score = length(s1);
else
previous_row = 0:length(s2);
for i=1:length(s1)
current_row = 0*previous_row;
current_row(1) = i;
for j=1:length(s2)
insertions = previous_row(j+1) + 1;
deletions = current_row(j) + 1;
substitutions = previous_row(j) + (s1(i) ~= s2(j));
current_row(j+1) = min([insertions, deletions, substitutions]);
end
previous_row = current_row;
end
score = current_row(end);
end
end.
```

Appendix D: Implementation Code of 40 Points for Delaunay Triangulation

```
X = [ 2     3.5   1.5   2.5   4     6     7     7.5   6     4     3
      4     5     6.5   4.5   6     5.5   4     5     2     2.5   5.5
      5     6.5   4     7.5   6.5   5.5   4.5   2     5.5   5     5
      5     7.5   5.5   3     3     6]

y= [  9     10    6     3     2     2.5   4     7     9.5   9     8
      6     7.5   8.5   4     5     3.5   3     6.5   4.5   6.5   9
      10.5  7.5   7.5   9     3.5   4     3.5   7.5   9.5   8.5   5.5
      7.5   3.5   6.5   5     6     2]
TRI=DelaunayTri(x,y)
figure;triplot(TRI);
title(' Delaunay Triangulation for 40 points');
```

Appendix E: Implementation Code of 40 Points for Voronoi Diagram

```
X = [  2     3.5   1.5   2.5   4     6     7     7.5   6     4     3
       4     5     6.5   4.5   6     5.5   4     5     2     2.5   5.5
       5     6.5   4     7.5   6.5   5.5   4.5   2     5.5   5     5
       5     7.5   5.5   3     3     6]

y= [   9     10    6     3     2     2.5   4     7     9.5   9     8
       6     7.5   8.5   4     5     3.5   3     6.5   4.5   6.5   9
       10.5  7.5   7.5   9     3.5   4     3.5   7.5   9.5   8.5   5.5
       7.5   3.5   6.5   5     6     2]

TRI=DelaunayTri(x,y)
In-centres= Inceter(TRI)
VD= Voronoi(In-centres)
figure; triplot(VD);
title(' Voronoi Vertex for 40 points');
```