ONLINE FINGERPRINT RECOGNITION

KHAIRUL BARIYAH BINTI ABD RAHIM

This thesis is submitted as partial fulfillment of the requirements for the award of the Bachelor of Electrical Engineering (Hons.) (Electronic)

Faculty of Electrical & Electronics Engineering
Universiti Malaysia Pahang

NOVEMBER, 2010

“All the trademark and copyrights use herein are property of their respective owner. References of information from other sources are quoted accordingly; otherwise the information presented in this report is solely work of the author.”

Signature	: ______________________________
Author	: KHAIRUL BARIYAH BT ABD RAHIM
Date	: 29 NOVEMBER 2010

"I hereby acknowledge that the scope and quality of this thesis is qualified for the award of the Bachelor Degree of Electrical Engineering (Electronic)"

Signature	: ______________________________
Name	: NURUL WAHIDAH BINTI ARSHAD
Date	: 29 NOVEMBER 2010

Specially dedicated to my beloved parents, sister, brother and friends.

Also to my supervisor. I’m nothing without them.

ACKNOWLEDGEMENT

First and foremost, I feel gratefull to thank Allah for giving me the chance to complete this project. Alhamdullillah.

I would also like to express my sincere appreciation to my supervisor Madam Nurul Wahidah Binti Arshad, lecture from Faculty of Electric and Electronic for her kindness on delivering me her knowledge on writhing the thesis, and lead me for support, guidance, advices and patience. Without her motivation, I could not finish this thesis as present here.

Lastly my utmost thanks go to my parent, Mr Abd RAhim Ali and Mdm Rashidah Kamaliah Hussin, my family member for their support and the most important of all, their love. Special thanks to my fellow friends that have support me a lot, help and advice me to be strongly and confident for doing this thesis; Siti Nurmaisarah, Aimi Nadia and Myzatul Diana. for their support. Not to forget, all my friends, and whoever involve directly or indirectly, in making this project a success. Thank you very much.

ABSTRACT

Fingerprints are the most popularly used in biometric identification and recognition systems, because they can be easily used and their features are highly reliable. Because of their uniqueness and consistency over time, fingerprint has been used for identification for over a century, more recently becoming automated due to advancements in computing capabilities. The systems are increasingly employed into business, trading and living fields for automatic personal identification. Besides that, fingerprint recognition beyond criminal identification applications to several civilian applications such as access control, time and attendance, and computer user login. This project introduces and implementation of an online fingerprint recognition system which is capable of verifying identities of people so fast, accurate and suitable for the real time. Such a system has great utility in a variety of personal identification and access control applications by operating in minutiae extraction and minutiae matching. Minutiae extraction algorithm is implemented for extracting features from an input fingerprint image captured with an online inkless scanner. For minutiae matching, the matching algorithm has been developed. This algorithm is capable of finding the correspondences between minutiae in the input image and store template. The system will be tested on set of fingerprint images captured with inkless scanner. The recognition accuracy is found to be acceptable. This result shows that our systems meet the response requirement of online recognition with high accuracy. All the systems will be built using MATLAB software.

ABSTRAK

Cap jari adalah paling popular digunakan dalam pengenalan biometrik dan sistem pengenalan, kerana ia dapat digunakan dengan mudah dan ciri-ciri cap jari juga sangat dipercayai . Ini kerana keunikannya dan keselarian dari masa ke masa, cap jari telah digunakan dalam pengenalan untuk lebih dari satu abad, dan kini menjadi automatik seiring dengan kemajuan dalam teknologi kekomputeran. Sistem ini semakin banyak digunakan dalam bidang perniagaan, perdagangan dan bidang-bidang lain untuk pengenalan peribadi automatik. Selain itu, pengesahan cap jari di luar aplikasi pengenalan jenayah untuk aplikasi beberapa kegunaan awam seperti kawalan akses, masa dan kehadiran, dan *login* (daftar masuk) bagi pengguna komputer. Projek ini memperkenalkan dan pelaksanaan suatu sistem pengenalan cap jari yang mampu mengesahkan identiti orang-orang begitu cepat, tepat dan sesuai untuk satu-satu masa (*real time*). Sistem seperti ini memiliki kegunaan yang besar dalam pelbagai pengenalan peribadi dan aplikasi kawalan akses oleh yang beroperasi di titik-titik pengenalan “minutiae” dan pemadanan titik-titik “minutiae”. Algoritma ekstraksi “minutiae” dilaksanakan untuk mengekstrak ciri dari citra cap jari yang diambil dengan pengimbas cap jari. Untuk hal pemadanan, algoritma pemadanan telah dibuat. Algoritma ini mampu menemukan hal-hal kecil dalam keserupaan antara citra cap jari yang dimasukkan dan cap jari yg tersimpan di dalam pencontoh. Sistem ini akan diuji pada serangkaian gambar yang diambil dengan pengimbas cap jari. Ketepatan pengakuan didapati boleh diterima. Hal ini menunjukkan bahawa sistem kami memenuhi keperluan dan pengakuan secara langsung dengan ketepatan tinggi. Semua sistem akan dibina menggunakan perisian MATLAB.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLE

LIST OF ABBREVIATIONS

PIN	-	Personal Identification Number
DNA	-	Ddeoxyribonucleic
FTIR	-	Frustrated Total Internal Reflection
MATLAB	-	Matrix laboratory software
CBFS	-	Coupled Breadth First Search
2-D	-	Two Dimensional
ADD	-	Average Absolute Deviation
dpi	-	Dots per inch
GUI	-	Graphical User Interface

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Overview

The increasing of society care to security threat has born new ways to protect software, hardware, building and even network system from outside party attacks. One of the security ways is by using biometric system. Such system use human body which always can be brought and not possible to leave it at home or loss during the trip. The technology becomes a popular identification and verification tool. Types of existing biometric are fingerprint, iris, eye retina, hand, gait, face and voice.

Biometric fingerprint recognition systems are the most common used biometric technology due to their long tradition. Fingerprint identification systems have been developed for more than hundred years and the identification of person through their unique fingerprint. Everyone's fingerprint pattern has unique character so that differ one to another. Equally, there is no human being having the same fingerprint, through both people of twin. The pattern formed when the human still in obstetric. A fingerprint is made of a series of ridges and furrows on the surface of the finger. The uniqueness of a fingerprint can be determined by the pattern of ridge and furrows as well as the minutiae points. Minutiae points are local ridge characteristics that occur at either a ridge bifurcation or ridge ending.

This project describes the design and implementation of an online fingerprint recognition system which is operating in minutiae extraction and minutiae matching. Minutiae extraction algorithm is implemented for extracting features from an input fingerprint image captured with an online inkless scanner. For minutiae matching, the matching algorithm has been developed. This algorithm is capable of finding the correspondences between minutiae in the input image and store template. The system will be tested on set of fingerprint images captured with inkless scanner. The recognition accuracy is found to be acceptable. This result shows that the systems meet the response requirement of online recognition with high accuracy.

1.2 Problem Statement

As our everyday life is getting more computerized automated security systems are getting more important. Password, smart card or personal identification number (PIN) is classical approach where there have tendency to lost or to be stolen and may be forgotten. For the example, the students in few college or university have to bring their own card, not only to define them as a student but they use the card to enter the door especially at lecture hall or the office where they must put their card to the machine and the system will recognize to enter the door. Sometime they forget to bring their own card and sometimes they are easily to change and take the card that belongs to them. That’s mean, the security are still not properly useful.

1.3 Objective

The objectives of this project are to;

i. To introduce a fingerprint matching system where is capable of verifying identities.

ii. To make analysis of the identification by using minutiae matching algorithm.

1.4 Scope of Project

i. Introduce automatic technique to extract the minutiae from the fingerprint input image and concentrates on thumb.

ii. Analysis the method that match two minutiae point-set based on minutiae matching algorithm.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Biometrics recognition is the process in which a biometrics system compares incoming information with data in its system to determine whether or not it can find a match [1]. If it does, it is said to have recognized the person it is analyzing. This technology is used in security systems all over the world.

The field of biometrics relies on the fact that many humans have distinctive and unique traits which can be used to distinguish them from other humans, acting as a form of identification. A number of traits can be used for biometrics. Fingerprints are one of the oldest examples, as everyone on Earth appears to have a unique set of fingerprints at any given time. The irises of the eye are also distinctive, as are faces [2].

With a biometrics system, sophisticated processing software can be taught to identify specific individuals, and when someone approaches the system, it can determine whether or not the person is someone familiar [1]. For example, if a laboratory decides to control access to an area with fingerprinting, a list of authorized personnel would be generated and their fingerprints would be entered into the biometrics system. Any time one of these people wanted to enter the restricted area, she or he would have to present a finger to the biometrics system so that it could run the fingerprint against its database in a biometrics recognition process.

The process of biometrics recognition is not foolproof. Sometimes, a small variation causes the system to reject someone, even when that person is in the system. For example, if facial recognition is used and someone experiences a change to the face like a poorly healed broken nose, plastic surgery, or swelling due to injury, biometrics recognition may fail. Conversely, sometimes it is possible to trick a system.

In a biometric system, a physical trait needs to be recoded. The recording is referred to as an enrollment. This enrollment is based on the creation of a template. A template is the digital representation of a physical trait. The template is normally a long string of alphanumeric characteristics that describe, based on a biometric algorithm, characteristics or features of a physical trait. The algorithm will also allow the matching of an enrolled template with a new template just created for verifying an identity, called a live template [3]. When a store template and a live template are compared, the system calculates how closely they match. If the match is close enough, a person will be verified. If the match is not close enough, a person will not verify.

2.1.1 History of Fingerprint

Fingerprint offer an infallible means of personal identification. In civilization, branding and even maiming were used to mark the criminal for what he was. The thief was deprived of hand which committed the thievery. The modern history of fingerprint identification begins in the late 19th century with the development of identification bureaus charged with keeping accurate records about individuals indexed, not according to name but according to some physical attribute [4]. Henry Fauld, in 1880, first scientifically suggested the individually and uniqueness of fingerprint. At the same time, Herschel asserted that he had practiced fingerprint identification for about 20 years old [5]. This discovery established the foundation of modern fingerprint identification. It formally accepted as a valid personal identification method by law enforcement agencies and became a standard procedure in forensic. With the advent of lives can fingerprinting and availability of cheap fingerprint sensor, fingerprint are increasing used in government and commercial application for positive person identification [5]. The conclusion of the history is shown in Table 2.1.1

Table 2.1 The History of Fingerprint [6][7].

Year	**Description**
1686	Marcello Malpighi, a professor of anatomy at the University of Bologna, noted in his treatise; ridges, spirals and loops in fingerprints. He made no mention of their value as a tool for individual identification. A layer of skin was named after him; "Malpighi" layer, which is approximately 1.8mm thick.
1823	In 1823, John Evangelist Purkinje, an anatomy professor at the University of Breslau, published his thesis discussing 9 fingerprint patterns, but he too made no mention of the value of fingerprints for personal identification.
1856	Sir William Hershel, Chief Administrative Office, Bengal India, first used fingerprints on native contracts.
1880	Dr. Henry Faulds, who was working in Tokyo, Japan, published an article in the Scientific Journal, "Nautre" (nature). He discussed fingerprints as a means of personal identification, and the use of printers ink as a method for obtaining such fingerprints. He is also credited with the first fingerprint identification of a greasy fingerprint left on an alcohol bottle.

1882	Gilbert Thompson of the U.S. Geological Survey in New Mexico, used his own fingerprints on a document to prevent forgery. This is the first known use of fingerprints in the United States.
1883	In Mark Twain's book, "Life on the Mississippi", a murderer was identified by the use of fingerprint identification. In a later book by Mark Twain, "Pudd'n Head Wilson", there was a dramatic court trial on fingerprint identification. A more recent movie as made from this book.
1891-1895	In 1891,the introduction of fingerprints for criminal identification in England and Wales, using Galton's observations and revised by Sir Edward Richard Henry. Thus began the Henry Classification System, used even today in all English speaking countries. In 1892, First systematic use of fingerprints in the U.S. with the New York Civil Service Commission for testing. Dr. Henry P. DeForrest, a pioneer in U.S. fingerprinting. In 1894, The use of fingerprints began in Leavenworth State Penitentiary in Kansas, and the St. Louis Police Department. They were assisted by a Sergeant from Scotland Yard who had been on duty at the St. Louis Exposition guarding the British Display.
1901	Introduction of fingerprints for criminal identification in England and Wales, using Galton's observations and revised by Sir Edward Richard Henry. Thus began the Henry Classification System, used even today in all English speaking countries.
1904	The use of fingerprints began in Leavenworth Federal Penitentiary in Kansas, and the St. Louis Police Department. They were assisted by a Sergeant from Scotland Yard who had been on duty at the St. Louis Exposition guarding the British Display.

2.2 Fingerprint as a Biometric

A smoothly flowing pattern formed by alternating crests (ridges) and troughs (valleys) on the palmar aspect of hand is called a palmprint as shown in figure 2.1. Formation of a palmprint depends on the initial conditions of the embryonic mesoderm from which they develop. The pattern on pulp of each terminal phalanx is considered as an individual pattern and is commonly referred to as a fingerprint [8]. A fingerprint is believed to be unique to each person. Fingerprints of even identical twins are different.

Fingerprints are one of the most mature biometric technologies and are considered legitimate proofs of evidence in courts of law all over the world. Fingerprints are, therefore, used in forensic divisions worldwide for criminal investigations. More recently, an increasing number of civilian and commercial applications are either using or actively considering using fingerprint-based identification because of a better understanding of fingerprints as well as demonstrated matching performance than any other existing biometric technology [8].

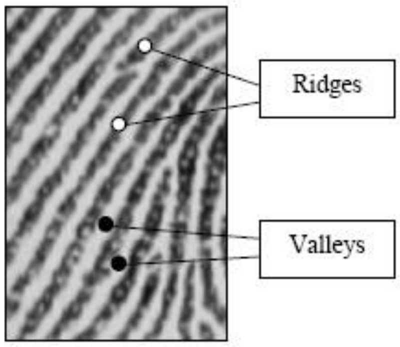


Figure 2.1: Ridges and valleys on a fingerprint image

People have tiny ridges of skin on their fingers because this particular adaptation was extremely advantageous to the ancestors of the human species. The pattern of ridges and "valleys" on fingers make it easier for the hands to grip things, in the same way a rubber tread pattern helps a tire grip the road [9].

The other function of fingerprints is a total coincidence. Like everything in the human body, these ridges form through a combination of genetic and environmental factors. The genetic code in DNA gives general orders on the way skin should form in a developing fetus, but the specific way it forms is a result of random events. The exact position of the fetus in the womb at a particular moment and the exact composition and density of surrounding amniotic fluid decides how every individual ridge will form.

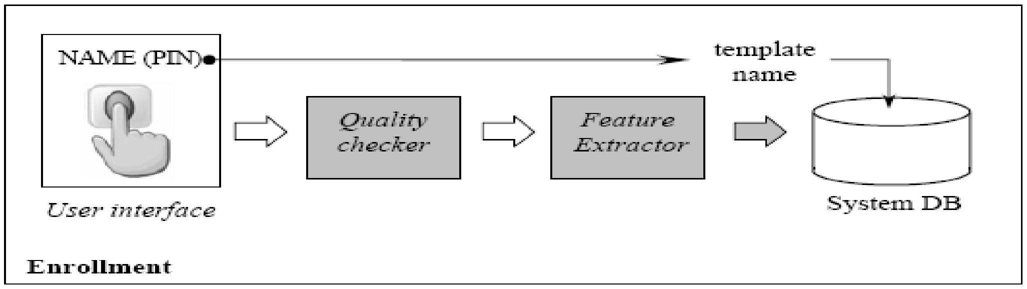
Consequently, fingerprints are a unique marker for a person, even an identical twin. And while two prints may look basically the same at a glance, a trained investigator or an advanced piece of software can pick out clear, defined differences.

2.3 Fingerprint Classification

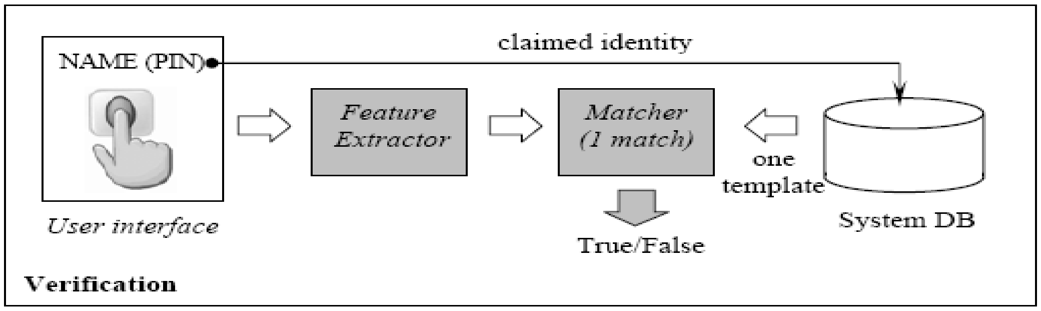
Fingerprint recognition technology is divided into two distinct processes to define a problem of resolving the identity of a person with different inherent complexities which is verification and identification [8].

In the verification process the user states who he or she is and a fingerprint is taken and compared to the user's previously registered fingerprint. If the fingerprints match, the user is "verified" as who he or she says he or she is. Since the newly acquired fingerprint is compared to only one stored fingerprint, this is called a one-to-one matching process (1:1) as shown in figure 2.2 (b). As in the enrollment process where shown in figure 2.2 (a), when fingerprint verification is done, only the fingerprint template is used in the comparison, not the actual image of the fingerprint.

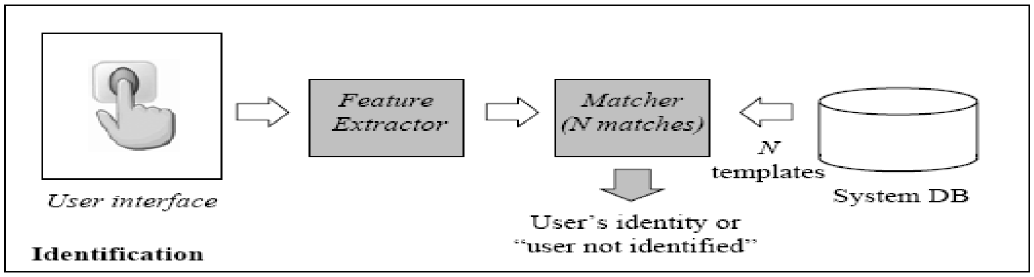
In the identification process the user doesn't need to state who he or she is. A fingerprint is taken and compared to each fingerprint in the database of registered users. When a match occurs, the user is "identified" as the existing user the system found. Since the newly acquired fingerprint is compared to many stored fingerprints, this is called a one-to-many matching process (1:N) as shown in figure 2.2 (c). As in the verification process, when fingerprint identification is done, only the fingerprint template is used in the comparison, not the actual image of the fingerprint.



a)



(b)



(c)

Figure2.2: Block diagrams of (a) enrollment, (b) verification, and (c) identification tasks. at the verification stage, the template

2.4 Type of Fingerprints

Fingerprint can be dividing into the three (3) major patterns. The patterns are arch, loop and whorls. Arches are different from loop because arches have more open curves. Arches account for approximately 5% of the ridge pattern in a given population.

Arches can also form a subgroup called tented arches. In the tented arch, the arch angle is much more obtuse than in the normal arch. For loop, loop account for approximately 60% of the ridge patterns in a given population. Loops may slant left or right, or be presented as a double loop. A double loop has both left a right loop, conforming to each other's outline. And for whorl are defined by at least one right making a complete circle. Whorls account for approximately 35% of the ridge making a complete circle. These major patterns types can be dividing future into different subgroup such as right, left or twin loops, plain or tended arches and spiral or concentric circles as whorls [10]. Figure 2.3 show the type of fingerprint with their images.

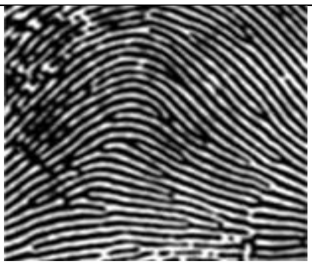
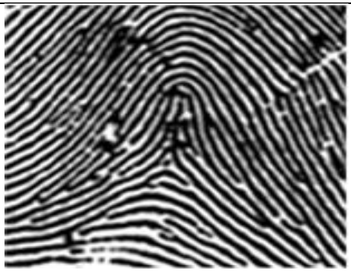
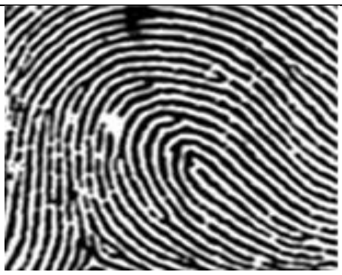
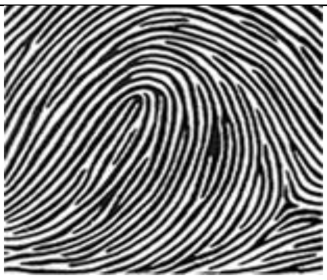
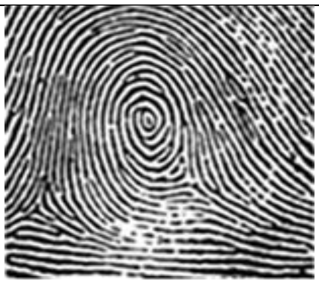
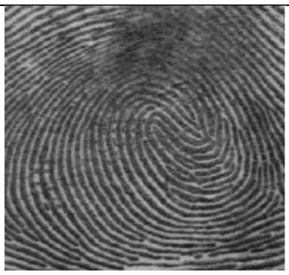
		
i) Arch	ii) Tented arch	iii) Right loop
		
iv)Left loop	v) Whorl	vi) Twin loop

Figure 2.3: General fingerprint patterns [13].

2.5 Fingerprint Sensing

There are two primary method of capturing a image [8].

i. Inked (off-line)

An inked fingerprint image is typically acquired in the following way: a trained professional obtains an impression of an inked finger on a paper and the impression is then scanned using a flat bed document scanner. It is evident that is inappropriate for fingerprint verification due to the inconvenience involved with ink and the need for subsequent digitization [19].

ii. Live scan (ink-less).

The live scan fingerprint is a collective term for a fingerprint image directly obtained from the finger without the intermediate step of getting an impression on a paper.

Image	**Description**
	An inked fingerprint image could be capture from the inked impression of finger
	A live scan fingerprint is directly image from a live finger based on optical total internal reflection principle.
	Rolled fingerprint are images depicting nail-to-nail area of a finger.

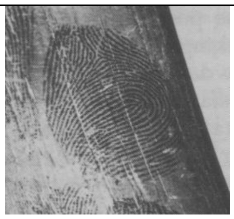
	Fingerprints capture using solid state sensors show a smaller area of finger than a typical fingerprint dab capture using optical scanners.
	A latent fingerprint refers to partial print typically lifted from a scene of crime.

Figure 2.4: Fingerprint Sensing [8].

2.6 Feature Extraction

A fingerprint is the reproduction of a fingertip epidermis, produced when a finger is pressed against a smooth surface. The most evident structural characteristic of a fingerprint is a pattern of interleaved *ridges* and *valleys*; in a fingerprint image, ridges (also called ridge lines) are dark whereas valleys are bright. Ridges and valleys often run in parallel. Sometimes they bifurcate and sometimes they terminate.

When analyze at the global level, the fingerprint pattern exhibits one or more regions where the ridge lines assume distinctive shapes (called singularities or singular regions) may be classified into three typologies: loop, delta and whorl as shown in. Figure 2.5 [10]. Singular regions belonging to loop, delta, and whorl types are typically characterized by ∩, Δ, and O shapes, respectively. Several fingerprint matching algorithms pre-align fingerprints images according to a landmark or a center point, called a core. The core point corresponds to the center of the north most loop type singularity. Unfortunately, due to the high variability of fingerprint patterns, it is difficult to reliably locate a registration (core) point in all the fingerprint images.

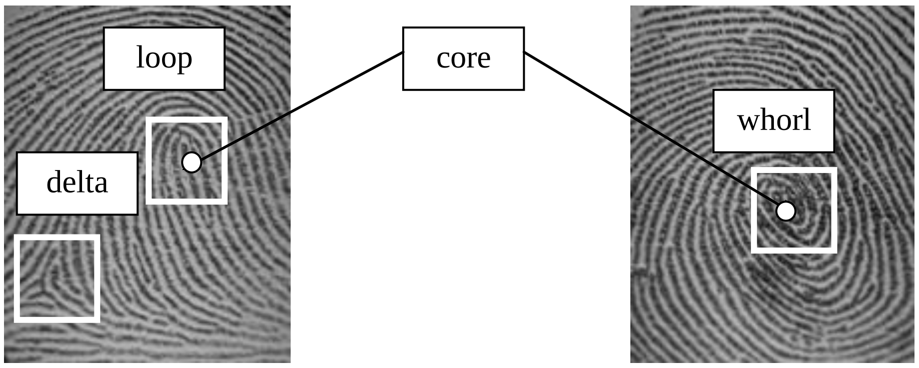
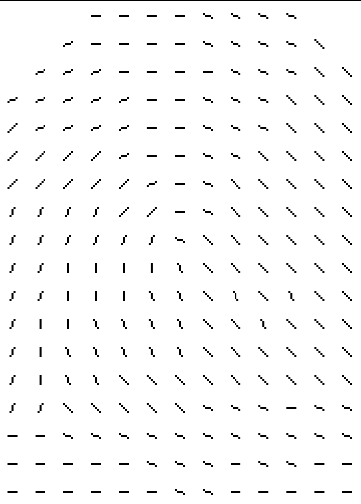


Figure 2.5 : Singular regions (white boxes) and core point (small circles) in fingerprint images.

Traditionally, the feature extraction system follows a staged sequential architecture which prelude effective integration of extracted information available from the measurements. Increased availability of inexpensive computing and sensing resources makes it possible to use better architectures or methods for information processing to detect the features reliably. Operating upon the thinned imaged, the minutiae are straightforward to detect. Ending are found at termination points of thin lines. Bifurcations are found at the junction of three lines. Sequences of fingerprint processing steps are shown in Figure 2.6.

		
a) Original	b) Orientation	c) Binarized

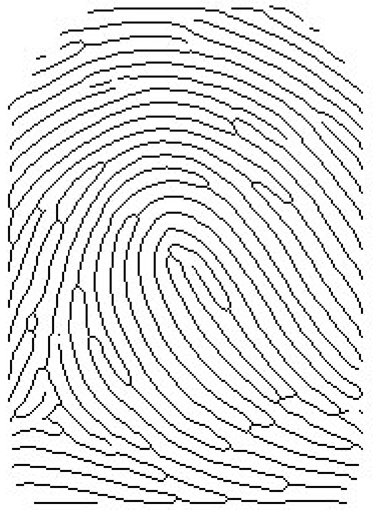
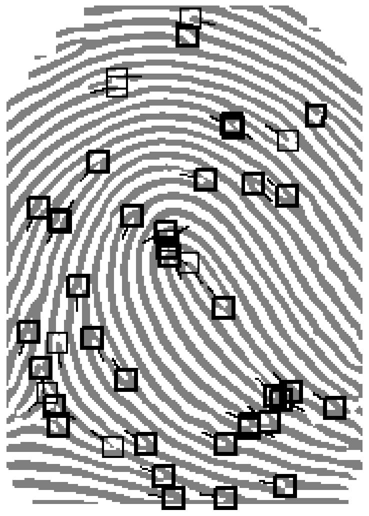
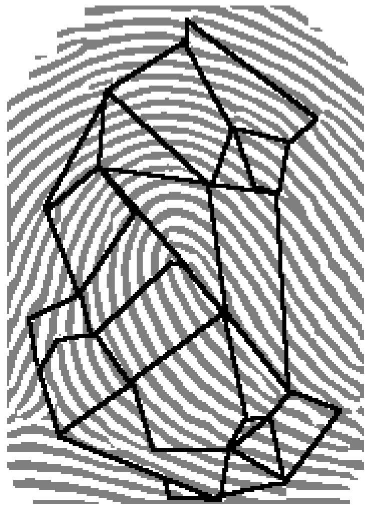
		
d) Thinned	e) Minutiae	f) Minutiae graph

Figure 2.6 : Sequence of fingerprint processing steps.

Once the features are determine, it is also a common practice to design feature extraction process in a somewhat ad hoc manner. The efficacy of such methods is limited especially when input measurements are noisy. Rigorous models of features from the input measurements, especially, in noisy situations.

There will always be extraneous minutiae found due to a noisy original image or due to artifacts introduced during matched filtering and thinning. These extraneous features are reduced by using empirically determined thresholds. For instance, a bifurcation having a branch that is much shorter than an empirically determined threshold length is eliminated because it is likely to be a spur. Two ending on a very short isolated line are eliminated this line is likely due to noise. Two endings that are closely opposing are eliminated because these are likely to be on the same ridge that has been broken due to a scar or noise or a dry finger condition that results in discontinuous ridges. Endings as the boundary of the fingerprint are eliminated because they are not true endings but rather the extent of the fingerprint in contact with the capture device.

Feature attributes are determined for each valid minutia found. These consist of ridge ending or bifurcation type, the (x, y) location, and the direction of the ending or bifurcation. Although minutia type is usually determined and stored, many fingerprint matching systems do not use this information because discrimination of one from the other is often difficult.

This result of the feature extraction is what is called a minutia template. This is a list of minutiae with accompanying attribute values. An approximate range on the number of minutiae found at this stage is from 10 to 100. If each minutia is stored with type 1 bit, location (9 bits each for x and y), and direction (8 bits), then each will require 27 bits and the template will required up to 400 bytes.

2.6 Type of Algorithm used for Interpretation

The algorithm used to match and enrol fingerprint fall under the following categories; Minutia-based, pattern-based, and hybrid algorithm.

2.7.1 Minutia-based Algorithm.

This type of algorithm would be good to use in a situation where template sizing is important. For example, doing a match on a card would be more efficient with a minutia-based algorithm. For minutia-based fingerprint algorithm, only a small part of the finger image is required for verification. As such , it is ideal to have as much minutia as required for verification, it would be ideal to use this algorithm where space restrictions impact the use and deployment of biometrics. Thus, a good imager for a minutia-based algorithm would be one that take a high-quality image and has a large enough capture window for the relative core of the fingers to be imaged and captured.

2.7.2 Pattern-based Algorithm.

Pattern-based matching algorithms used both the micro and macro features of a fingerprint. When the macro features are utilized, the size of the image required for authentication, when compared to the size of the image needed for minutia-only requirements, is larger. Since only the macro features need to be compared, these type of algorithms tent to be fast and have a larger template size. They also require more of the image area to be present during verification. A good imager for pattern-based matching algorithms is one that has a large enough scanning surface to capture the important macro details.

2.7.3 Hybrid Algorithm.

As the name implies, a hybrid algorithm uses the best feature from the minutia-based algorithms and those from pattern-based matching. This algorithm is a good all-purpose algorithm, giving a good trade-off between the accuracy of the minutia algorithm and the speed of pattern-based recognition. This algorithm would require the resulting template to be slightly larger than a minutia template, and smaller than a pattern-based matching algorithm. A high-quality optical sensor is best for this type of algorithm. It would offer a large enough image area, with very good quality for the images. The hybrid algorithm takes longer to enrol because of the use of both minutia and pattern-based recognition. Once this has occurred, the matching is actually faster than the minutia-based algorithm.

2.8 MATLAB (Matrix Laboratory)

The Matrix Laboratory program also known as MATLAB was initially written with the objective of providing it user with the numerical computation libraries [12]. Based on The Math Work Inc. (1999), MATLAB is a mathematical programming language and environment, optimized for matrix operations. Matrix operations are crucial to all kinds of signal processing for both sound and images. MATLAB is a high-level technical computing language and interactive environment for algorithm development, data visualization, data analysis, and numerical computation that enables a faster performance in computationally intensive tasks than with traditional programming languages such as C, C++, and FORTRAN.

2.9 Previous Research

In previous research, several approaches have been developed for automatic fingerprint classification. Research from Nasir Rehan and Khalid Rashid, they are present about Multi-matcher Based Fingerprint Identification System which they are using Minutiae-based as their fingerprint representation approach [16]. This study is reveals the possibility for introducing a fingerprint identification system based on different fingerprint matching systems, working in close collaboration. They said that if appropriate efforts been utilized in this direction it may prove to be the cure for deficiencies that are present in the present day fingerprint identification systems. Their feather investigations will also focus on the precedence of fingerprint matchers for decision making in a multi-matcher based fingerprint identification system relative to their individual performance results obtained when operated individually.

By Sharat S. Chikkerur and friends researches, they found a new matching algorithm based on graph matching principles which they presented a new representation (K-plet) to encode local neighbourhood for each minutiae. The K-plet representation is invariant under translation and rotation since it defines its own local co-ordinate system. They also present a dynamic programming approach for matching each local neighbourhood in an optimal fashion. And lastly they propose CBFS (Coupled Breadth First Search) algorithm, a new dual graph traversal algorithm for consolidating all the local neighbourhood matches and analyze its computational complexity [22]. A important advantage of this algorithm is that, no explicit alignment of the minutiae sets is required at any stage of the matching process. Furthermore, the proposed algorithm provides a very genetic but formal framework of consolidating the local matches during fingerprint recognition.

While the research by Ying Jie,Yuan Yi fang and their friends in topic “Fingerprint Minutiae Matching Algorithm For Real Time System”, they propose a new fingerprint minutiae matching algorithm, which is fast, accurate and suitable for the real time fingerprint identification system. They used the core point to determine the reference point and used a round bounding box for matching [23]. And using the round bounding box made the matching more accurate. They approve that this algorithm is more suitable for the real system time.

By Anil Jain and Lin Hong, they describe the design and implementation of an online fingerprint verification system which operates in two stages which is minutia extraction and minutia matching [17]. For minutia matching, they use alignment-based elastic matching algorithm. This algorithm is capable of finding the correspondences between input minutiae and the stored template without resorting to exhaustive search and has the ability to adaptively compensate for the nonlinear deformations and inexact pose transformations between fingerprints. The experiment results show that their system achieves excellent performance in a real online verification environment. It meets the response time requirements of online verification with high accuracy.

Other than that, in previous research done by Davide Maltoni, he introduces fingerprint recognition systems and their main components: sensing, feature extraction and matching. The basic technologies are surveyed and some state-of-the-art algorithms are discussed. Other than that, the research from J.V Kulkarni and friends, they discuss about the scheme for the extraction of Gabor feature and minutiae feature where they compare which is the best fingerprint feature extraction and their own advantages.

In the different paper written by Ying Hao, Tieniu Tanand Yunhong Wang, they propose an effective fingerprint matching algorithm based on error propagation. The algorithm makes use of matching pairs of ends and bifurcations, thus provides more reliable alignment of two templates. Further, the similarity of the common region is used as the similarity measure of two fingerprint templates. Finally, the concept of error propagation is applied to track nonlinear deformation adaptively. Experimental results indicate the effectiveness and robustness of our algorithm and it does meet the response time and accuracy requirements for an automatic fingerprint verification system.

CHAPTER 3

METHODOLOGY

3.1 Introduction.

The methodology for this project consists of four major stages, which are image pre-processing, image enhancement, feature extraction, that's done by the senior and lastly matching method. In this project will concentrate more in post processing (matching method). The various stages of a typical fingerprint recognition system are shown in Figure 3.1. The first stage is image pre-processing. In this part we are loading the fingerprint as the database. For the image enhancement stage, it consists of standard image processing algorithms such as noise removal and smoothening. However, it is to be noted that unlike regular images, the fingerprint image represents a system of oriented texture and has very rich structural information within the image.

Furthermore, the definition of noise and unwanted artifacts are also specific to fingerprints. The fingerprint image enhancement algorithms are specifically designed to exploit the periodic and directional nature of the ridges. Finally, the minutiae features are extracted from the image and are subsequently used for matching. The overall block diagram of the system in Figure 3.1 [17]. It operates as follows which offline phase is a digital image of one fingerprint of a person to be verified is captured. Here a feature extraction algorithm is applied. Minutiae are extracted and stored as a template for later use. In online phase, the individual to be verified indicates his or her identity and places his or her finger on the inkless scanner. A digital image of his or her fingerprint is captured. Here minutiae are extracted from the captured and fed to matching algorithm, which matches it against the stored template.

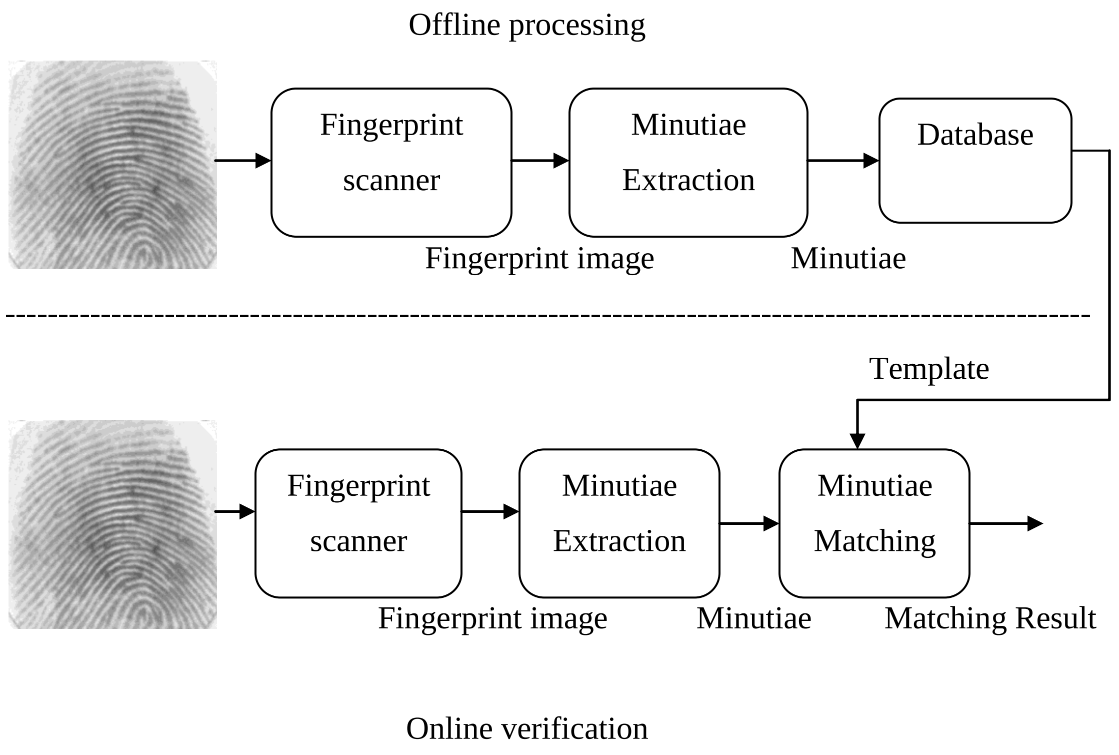


Figure 3.1 The General architecture of a fingerprint recognition system.

3.2 Image acquisition

The first steps in developing of this system start with the image acquisition. For this project, the grey scale image is used as the input. The database is get from scanner. The fingerprint image will go through an image pre-processing process in order to extract the fingerprint local feature. In this project, eight images are used for minutiae-based extraction from one database (eight images per finger). The image is from students, which the average student’s age is 20-30 years old and 50% of them are males. A series of finger image is collected in the standard fingerprint database and after that each image will be test one by one by the next sequential process to be use as a training image.

There are a number of different ways to get an image of a fingerprint, and the most common methods today for electronic fingerprint readers are optical scanning and capacitance scanning. In this project the fingerprint scanner that used is the U.are.U 4500 fingerprint reader from Digital Persona. The resolution of the scanned fingerprint must be 500 dpi while the size is 328 x 364 (119 kpixel). This scanner used optical sensor where it is better than other type of sensor. When a fingerprint is applied to or passed over the sensor window of the fingerprint reader, the fingerprint is scanned and a gray-scale image is captured. Special computer software then identifies the key minutiae points from the image [14].

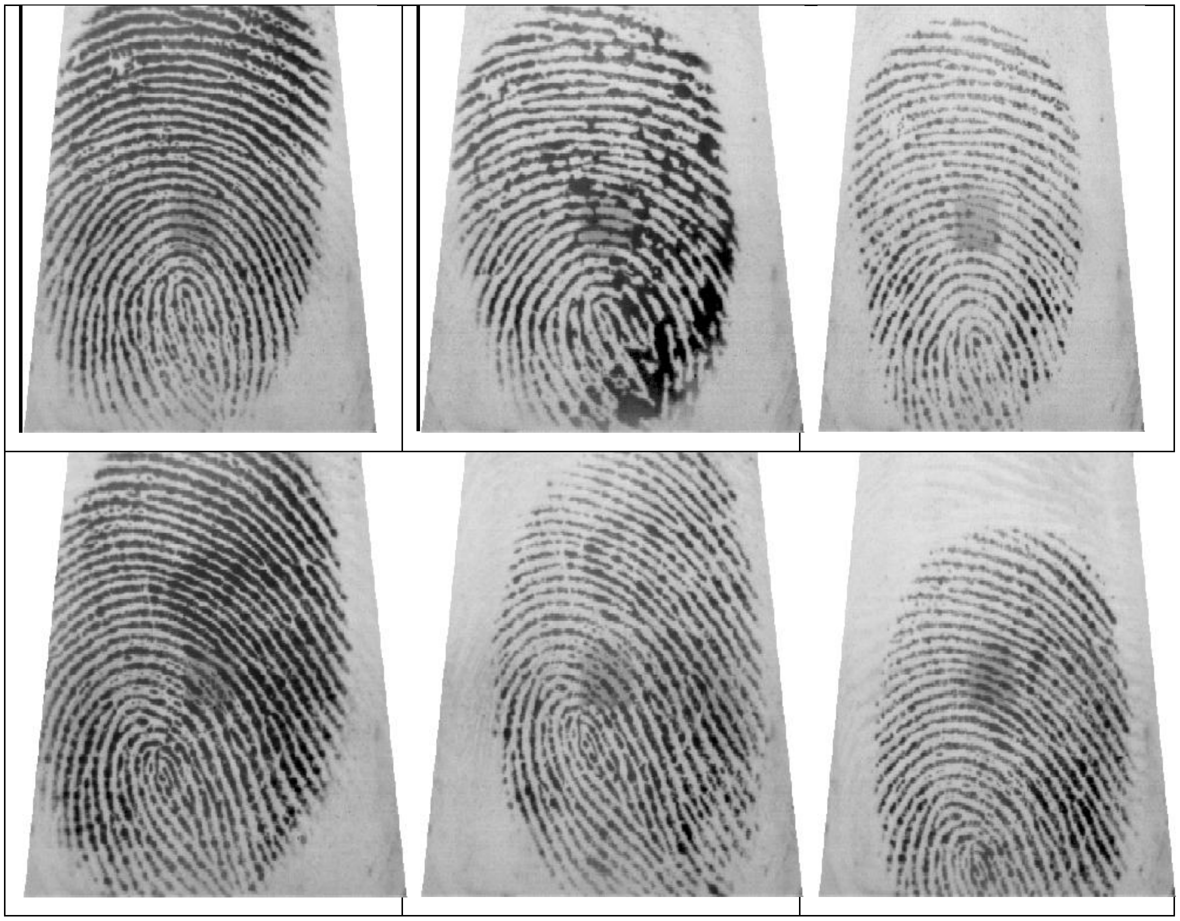


Figure 3.2 : Sample of finger image in the database.

3.3 Enrollment.

Enrollment is a process of collecting and compiling all the information in the database that going to use during the personal recognition process to compare the entire input image with the database to observe the result. This process only occurs in the enrolment system but not in the recognition system. This step usually uses multiple fingerprint images of the same fingerprint register and enrolment the fingerprint database. The purpose is to produce a representative sample that is less noise. While store the database the uniqueness of each image must be correctly compute. For this system the correctness of the data had been test for a few times in order to get the specific value. All the information is being compute in the table of database. This database can be compute at the same coding file or at other coding files. If database are store at the different files, it need to have a function to link between database and the process image. However, it is if the data computed in the same file that can directly run the process of identification.

3.4 Loading Image.

The first step is loading the fingerprint image into the program. In MATLAB M-file to load images into the program is by using “uigetfile” function. “uigetfile” displays a modal dialog box that lists files in the current directory and enables the user to select or type the name of a file to be opened. If the filename is valid and if the file exists, “uigetfile” returns the filename when the user clicks Open. Otherwise “uigetfile” displays an appropriate error message from which control returns to the dialog box. The user can then enter another filename or click Cancel. If the user clicks Cancel or closes the dialog window, “uigetfile” returns 0. “imshow” function is to display the grayscale image on the figure window.

```
%% Load image

[imagefile1]=
uigetfile('*.bmp;*.BMP;*.tif;*.TIF;*.jpg',
'Open An Fingerprint image');
if imagefile1 ~= 0
image1=imread(char(imagefile1));
end;
figure(1),imshow(image1);
```

Figure 3.3 : MATLAB coding for loading image.

3.5 Pre-processing Process.

After loading the fingerprint image or also known as the input image, came the pre-processing process. The pre-processing process consist four stages. The four stages are enhancement, binarization, thinning and minutiae extraction. Figure 3.3 shows the pre-processing stages.

3.5.1 Enhancement.

After loading process the fingerprint image will go through an image pre-processing which involve image enhancement. This process will enhances the contrast of images by transforming the values in an intensity image, or the values in the colourmap of an indexed image, so that the histogram of the output image approximately matches a specified histogram.

```
%% enhance image

J = imadjust(I);
figure (2), imshow (J);
title('imadjust');

K= histeq(J);
figure (3), imshow (K);
title('Histeq');
```

Figure 3.4: MATLAB coding for enhancement.

In order to enhance the fingerprint image, the “histeq” function is used. This function enhances the contrast of images by transforming the values in an intensity image, or the values in the colormap of an indexed image, so that the histogram of the output image approximately matches a specified histogram.

3.5.2 Binarization.

After enhance process, the fingerprint images will go through binarization process. In this step, the gray scale image is converted to a binary image through the process of simple thresholding or some form of adaptive binarization. The quality of the binarization output is improved if the gray scale image is enhanced prior to this process. Each pixel in the output image (greyscale image) is converted into one bit which has been assign the value as ‘1’ or ‘0’ depending upon the mean value of all the pixel (graythresh). If the grater then level (graythresh) then its ‘1’ for whites otherwise its ‘0’ for black. By trying a few values, the best performance value will get a useful image. This step is also referred to as segmentation in literature.

```
%% binarize image

L=K(:,:,1)>52;
L=~L;
figure (4), imshow(L)
```

Figure 3.5: MATLAB coding for binarization.

3.5.3 Thinning

The next step is the thinning stage. The resulting binary image is thinning by an iterative morphological process resulting in a single pixel wide ride map. Some algorithms and proposed approach does not require this stage. An important approach to representing the structural shape of a plane region is to reduce it to a graph. This reduction may be accomplished by obtaining the skeleton of the region via thinning (also called skeletonising) algorithm. The thinning algorithm while deleting unwanted edge point should not:

i) Remove end point
ii) Break connectedness
iii) Cause excessive erosion of the region

```
%% thinning image

M=bwmorph(L,'thin','inf');
figure (5), imshow(~M)
set(gcf,'position',[1 1 400 400]);
```

Figure 3.6 : MATLAB coding for Thinning stage.

In order to thin the fingerprint image, the “bwmorph” function is used. “bwmorph” uses the following algorithm [20].

1. Divide the image into two distinct subfields in a checkerboard pattern.
2. In the first subiteration, delete pixel p from the first subfield if and only if the conditions G1, G2, and G3 are all satisfied.
3. In the second subiteration, delete pixel p from the second subfield if and

only if the conditions G1, G2, and G3' are all satisfied.

Condition G1:

$$X_H(p) = 1 \tag{3.6}$$

where

$$X_H(p) = \sum b_i \tag{3.7}$$

$$b_i = \begin{cases} 1 \text{ if } x_{2k-1} = 0 \text{ and } (x_{2i} = 1 \text{ or } x_{2k-1} = 1) \\ 0 \text{ otherwise} \end{cases}$$

x1, x2, ..., x8 are the values of the eight neighbors of p, starting with the east neighbor and numbered in counter-clockwise order.

Condition G2:

$$2 \leq \min \{ n_1 (p), n_2 (p)\} \leq 3 \tag{3.8}$$

Where

$$n_1 (p) = \sum x_{2k-1} \vee x_{2k} \tag{3.9}$$

$$n_2 (p) = \sum x_{2k} \vee x_{2k-1} \tag{3.10}$$

Condition G3:

$$(x_2 \vee x_3 \vee x_8) \wedge x_1 = 0 \tag{3.11}$$

Condition G3':

$$(x_6 \vee x_7 \vee x_4) \wedge x_5 = 0 \tag{3.12}$$

The two sub iterations together make up one iteration of the thinning algorithm. When the user specifies an infinite number of iterations (n=Inf), the iterations are repeated until the image stops changing. The conditions are all tested using applylut with precomputed lookup tables.

3.5.4 Minutiae Extraction.

Minutiae extraction was carried out using crossing number approach [24]. Crossing number of pixel 'p' is defined as sum of the differences between pairs of adjacent pixels defining the 8-neighborhood of 'p'. Mathematically:

$$cn(p) = \frac{1}{2}\sum_{i=1}\left|val(p_{imod\,8}\ ) - val(p_{i-1}\ )\ \right| \tag{3.13}$$

Where p_0 to p_7 are the belonging to an ordered sequence of pixel defining the 8-neighborhood of p and *val(p)* is the pixel value. Figure 3.7 shown cn(p)=2, cn(p)=3 and cn(p)=1 represent a non-minutiae region, a bifurcation and ridge ending.

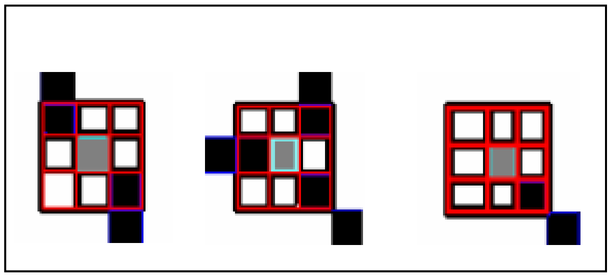


Figure 3.7: Non-minutiae, bifurcation and ridge ending [25].

Extraction of appropriate feature is one of the most important tasks for a recognition system. A multilayer perceptron (MLP) of tree layer is trained to detect the minutiae in the thinned fingerprint image of size 328x364. The first layer of the network has nine neurons

The next step is ridge detection. The most salient property corresponding to ridges in a fingerprint image is that grey level values on ridge attain their local maxima along the normal direction of local ridges. In minutia detection algorithm, a fingerprint image is first convolved with the following two masks, $h_t\ (x, y, u, v\ )$ and $h_b\,(x, y, u, v\ )$ of size W x H, which are capable of adaptively accentuating the local maximum grey level values along the normal direction of the local ridge direction:

$$h_t(x,y,u,v) = \begin{cases} \frac{-1e^{\frac{-u}{\delta^2}}}{\sqrt{2\pi\delta}}, & \text{if } u=l(v) - d,\ v \in \Omega \\ \frac{1e^{\frac{-u}{\delta^2}}}{\sqrt{2\pi\delta}}, & \text{if } u=l(v),\ v \in \Omega \\ 0 & \text{Otherwise,} \end{cases} \tag{3.1}$$

$$h_b(x,y,u,v) = \begin{cases} \frac{-1e^{\frac{-u}{\delta^2}}}{\sqrt{2\pi\delta}}, & \text{if } u=l(v) + d,\ v \in \Omega \\ \frac{1e^{\frac{-u}{\delta^2}}}{\sqrt{2\pi\delta}}, & \text{if } u=l(v),\ v \in \Omega \\ 0 & \text{Otherwise,} \end{cases} \tag{3.2}$$

$$l(v) = v \tan(\theta(x,y)), \tag{3.3}$$

$$d = \frac{H}{2\cos(\theta(x,y))}, \tag{3.4}$$

$$\Omega = H\left[\left|\frac{\sin(\theta(x,y))}{-2}\right|\right], \left[\left|\frac{\sin(\theta(x,y))}{2}\right|\right], \tag{3.5}$$

Where θ(x, y) represent the local ridge orientation at pixel (x, y). If both h the grey level values at pixel (x, y) of the convolved images are larger than a threshold T_r, then the pixel (x, y) is labelled as a ridge [17]. By the adapting the mask width to the local ridge, this algorithm can efficiently locate the ridge in a fingerprint image. In MATLAB, “bwlabel” command is used to connected component in binary image. In the coding, NEndLab=bwlabel(NEnd) return a matrix NEndLab, of the same size as BW, containing labels for the connected object and 8 specifies 8-connected object. If the argument is omitted, it defaulys to 8. The elements of L are integer values greater than or equal to 0. The pixels labeled 0 are the background. The pixels labeled 1 make up one object, the pixels labeled 2 make up a second object, and so on.

```
%% RIDGE ENDING

NEnd=(N==1);
%imshow(NEnd)
NEndLab=bwlabel(NEnd);
End=regionprops(NEndLab,'Centroid');
dEnd=round(cat(1,End(:).Centroid));
figure (8), imshow(~N)
set(gcf,'position',[1 1 400 400]);
hold on
plot(dEnd(:,1),dEnd(:,2),'ro')
P=size(dEnd,1)

%% Ridge Bifurcation

NBif=(N==3);
NBifLab=bwlabel(NBif);
Bif=regionprops(NBifLab,'Centroid','Image');
dBif=round(cat(1,Bif(:).Centroid));
plot(dBif(:,1),dBif(:,2),'go')
Q=size(dBif,1)
```

Figure 3.8: MATLAB coding for Ridge Detection.

The third step is the Minutiae Detection. Without o loss generally, assume that if a pixel is on a thinned ridge, then it has a value 1, and 0 otherwise. However, the presence of undesired spike and break present in a thinned ridge map may lead to many spurious minutiae being detected. Therefore, the following heuristics are used in pre-processing. If a branch in a ridge map is orthogonal to the local ridge direction and its length is less than a specified threshold, then it will be removed. If a break in a ridge is short enough and no other ridge pass through it, then it will be connected. Here is the command of minutiae detection.

```
%% Find minutiae


fun = @minutie
N = nlfilter(M,[3 3],fun);
figure (7), imshow (N)
set(gcf,'position',[1 1 400 400]);
```

Figure 3.9 : MATLAB coding for Minutiae Detection

And create the formula of minutiae detection.

```
function y=minutie(x)
i=ceil(size(x)/2);
if x(i,i)==0;
    y=0;
else
    y=sum(x(:)) - 1;
end
```

Figure 3.10 : MATLAB formula for Minutiae Detection

3.6 False Minutia Removal

False ridge breaks due to insufficient amount of ink and ridge cross-connections due to over inking are not totally eliminated [26]. Actually all the earlier stages themselves occasionally introduce some artifacts which later lead to spurious minutia. Twelve types of false minutia are specified in Figure 3.11 [27].

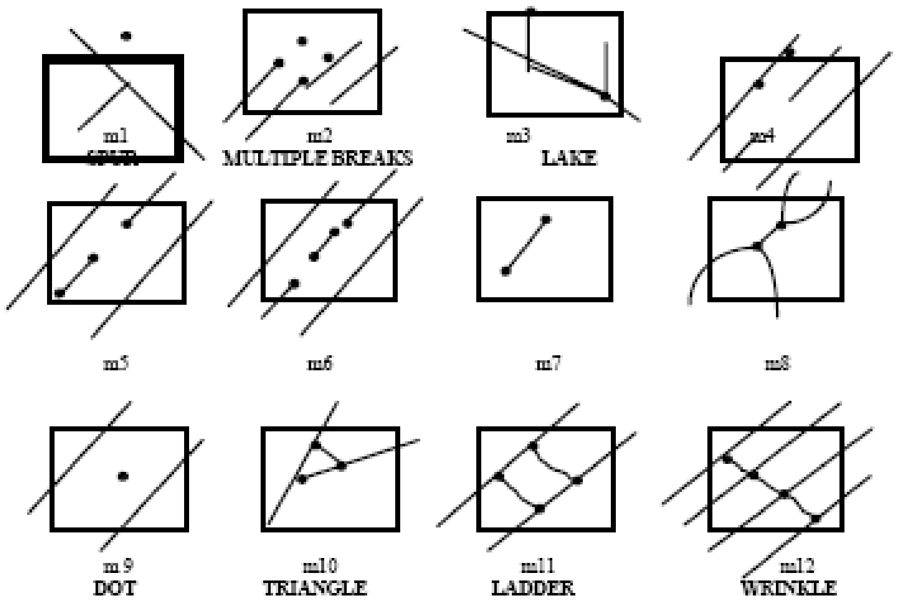


Figure 3.11 : False Minutia Structure

Procedures to remove false minutia are first, the distance between one bifurcation and one termination is less than D and the two minutia’s are in the same ridge for m1 case. Remove both of them while D is the average inter-ridge width representing the average distance between two parallel neighbouring ridge. If the distance between two bifurcations is less then D and they are in the same ridge, remove the two bifurcations (for m2, m3, m10, and m11 cases). If two terminations are within a distance D and their directions are coincident with a small angle variation. And they suffice the condition that no any other termination is located between the two terminations. Then the two terminations are regarded as false minutia derived from a broken ridge and are removed (for case m4, m5, and m6). If two terminations are located in a short ridge with length less than D, remove two terminations (m7). And lastly, if a branch point has at least two neighbouring branch point, which are each no further away than maximum distance threshold value and these branch point are closely connected on common line segment than remove the branch point (m12).

3.7 Matching Method

Clearly the most important stage of a fingerprint recognition system is the matching process. In this part, the canter of the fingerprint image used to calculate the Euclidean distance between the canter and the feature point. Usually, the canter or reference point of the fingerprint image is what called the “core” point. A core point is located at the approximate centre, which defined as the topmost point on the innermost upwardly curving ridgeline. After extracting the location of the minutiae for the prototype fingerprint images, the calculated distances was stored in the database along with the name of the person to worm each fingerprint belongs. The last phase is the recognition phase where testing fingerprint image that input to the system and minutiae matching comparing the distance extracted minutiae to the many stored in the database. There are two set of minutia of two fingerprint images, the minutia match algorithm determines whether the two minutia sets are from the same finger or not. It includes two consecutive stages: one is alignment stage and the second is match stage. For alignment stage, two fingerprint images to be matched, any one minutia from each image was chosen. The similarity of the two ridges then calculate associated with the two referenced minutia point. If the similarity is larger than a threshold, each set of minutia was transform to a new coordination system whose origin is at the referenced point and whose x-axis is coincident with the direction of the referenced point. For the match stage, after the set of transformed minutia points is derived, the elastic match algorithm is used to count the matched minutia pairs by assuming two minutia having nearly the same position and direction are identical. Then, identify the person result [15]. The function that being use for this step is:

```
matched1=double(e).*double(a);
matched=uint8(matched1);
```

Figure 3.11 : MATLAB formula for Minutiae matching

3.8 GUI Simulation

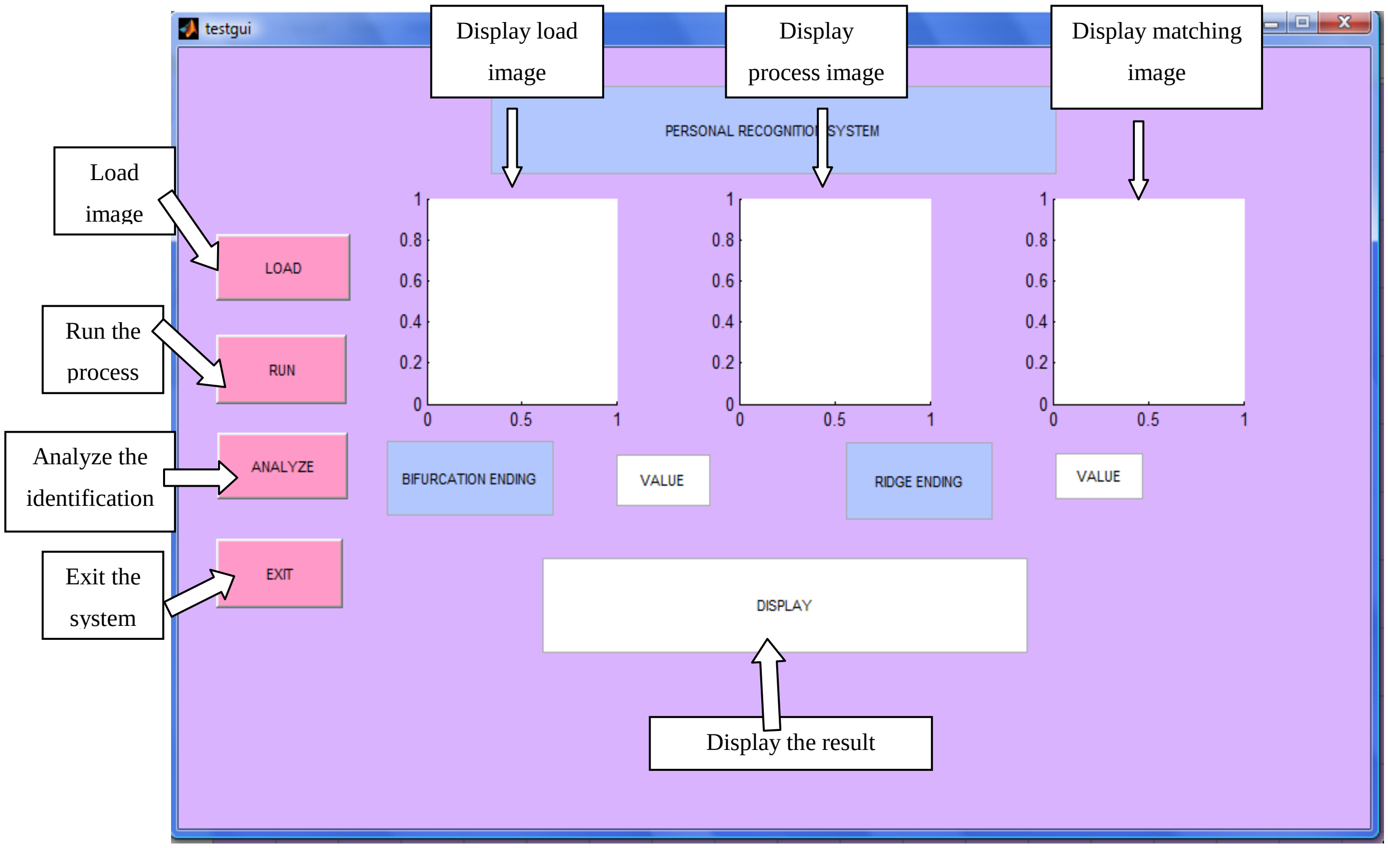


Figure 3.12: Running GUI

During the evaluation of the system, when the LOAD button are push, the system will required to select the file that had placed all the input image. After the selection, the image will appear at the axis 1 as an input image that going to be process. The image that been display is the original image with the same colour, pixel and size without any changing.

After the image was loaded, the processes continue by running the process. When the button RUN is push, the system will continue process the input image with pre-processing method and also feature extraction method until matched image between the original image and the process image. Axis 2 will display the pre-processing and feature extraction from input image and axis 3 will display the matched image. The button EXIT is to close the system automatically and exit from the system. Then all the data will be reset to the original condition.

CHAPTER 4

RESULT AND DISCUSSION

4.1 Introduction.

This discussions and results chapter will discuss the result that obtained from MATLAB programming. The MATLAB programming consists of six main parts which are loading image, enhancement, ridge detection (binarization), thinning, minutiae extraction and matching part.

4.2 Loading Image.

In Figure 4.1 shows the input fingerprint image which has been loaded into the pre-processing system. The fingerprint image that used as an input image is in greyscale. The fingerprint images used in this program are from U.are.U 4500 digital Persona USB fingerprint scanner is an online database .The fingerprints were acquired by using a low-cost optical sensor and the image size of pixel is 512 dpi (average x, y over the scan area [21]. Besides that, the fingerprints are mainly from 20 to 30 year-old students and about most of them are male. The image file format is in Tagged Image File Format (tiff). Figure 4.1 shows the example of the image from the scanner.

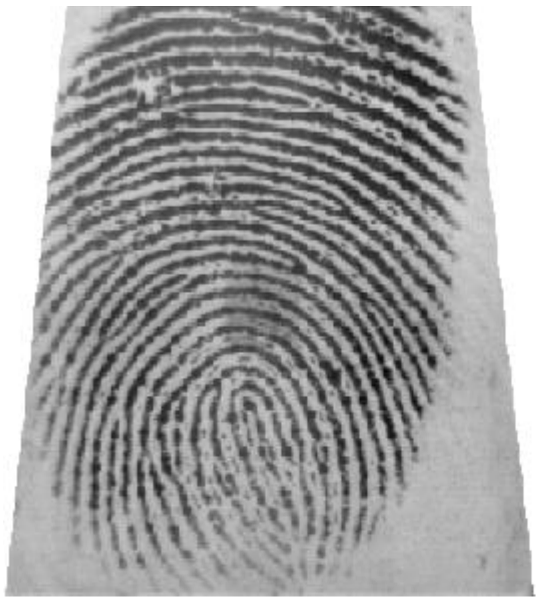
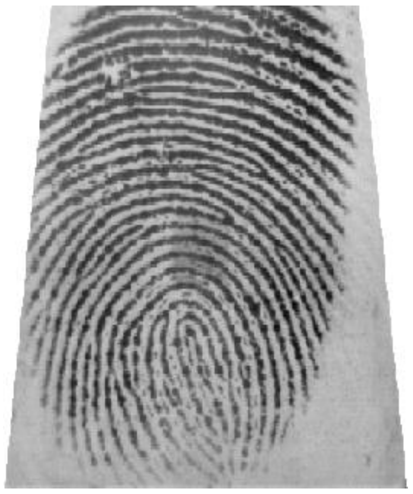
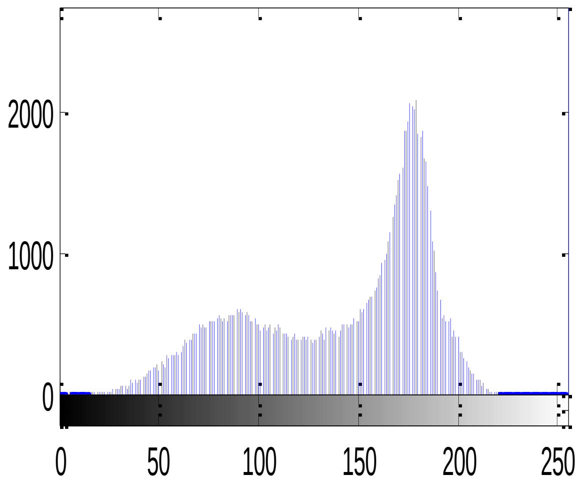


Figure 4.1: Example of the image from fingerprint scanner.

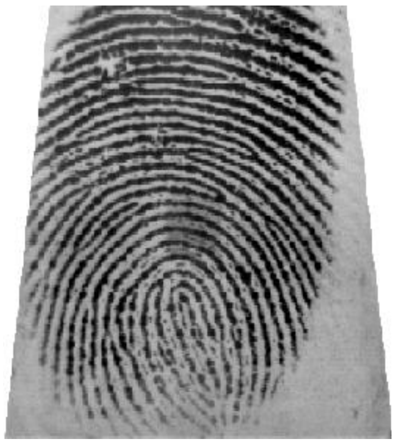
4.3 Enhancement.

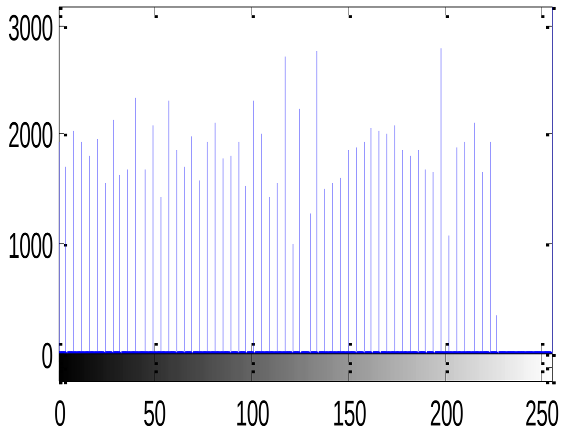
Enhancement is the next step after loading images with the aim of obtaining a better contrast of the fingerprint image. Enhancement using histogram method enhances the contrast of images by transforming the values in an intensity image, or the values in the colourmap of an indexed image, so that the histogram of the output image approximately matches a specified histogram. Figure 4.2 (c) shows the fingerprint image from Figure 4.2 (a) after histogram enhancement. It is easy to see that the fingerprint has now a much better contrast. As visible in the histogram, the gray level value start at 50 (gray) and end at 255 (white).





(a) (b)



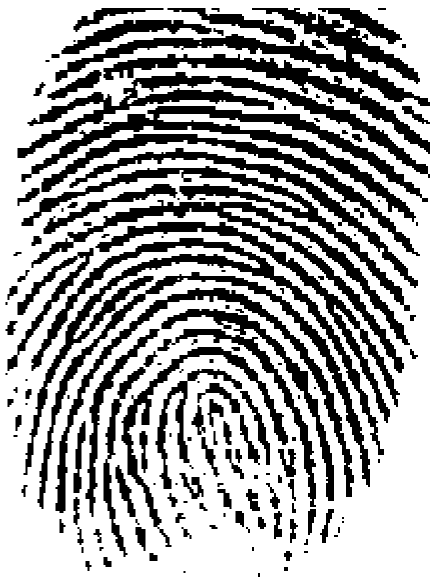


(c) (d)

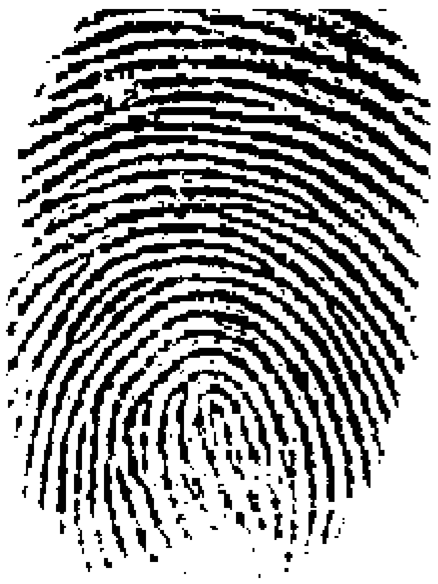
Figure 4.2: Result for : (a) Original image and (b) Histogram (c)image after enhancement process (b) Histogram for (c).

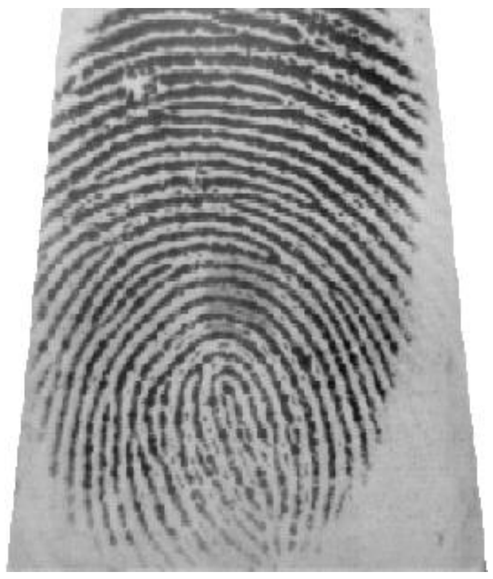
4.4 Binarization

In binarization process, each pixel in the output image (greyscale image) is converted into one bit which has been assign the value as ‘1’ or ‘0’ depending upon the mean value of all the pixel (graythresh). If the grater then level (graythresh) then its ‘1’ for whites otherwise its ‘0’ for black. The result is shown in Figure 4.3 (a) and the comparison binarization and the original image is shown in Figure 4.3 (b) and 4.3 (c)



(a)



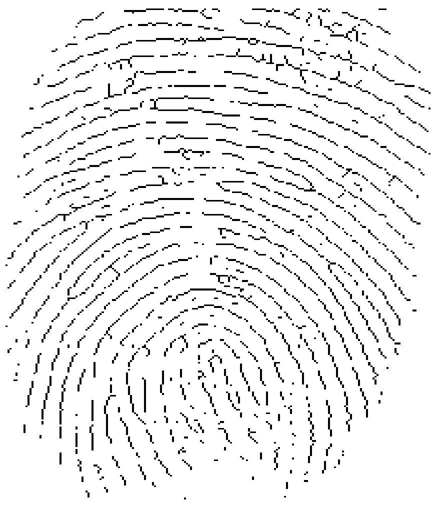


(b) (c)

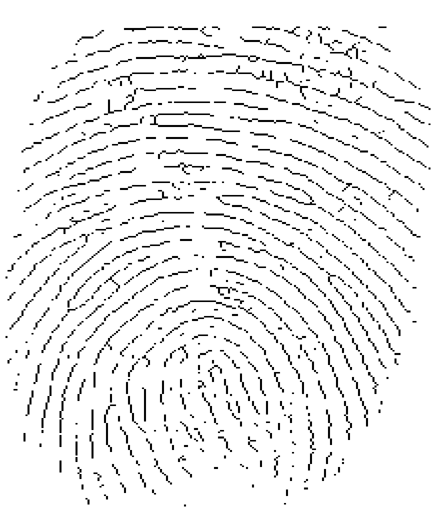
Figure 4.3: Result for : (a) Binarized image (b) and (c) comparison binarization and the original image

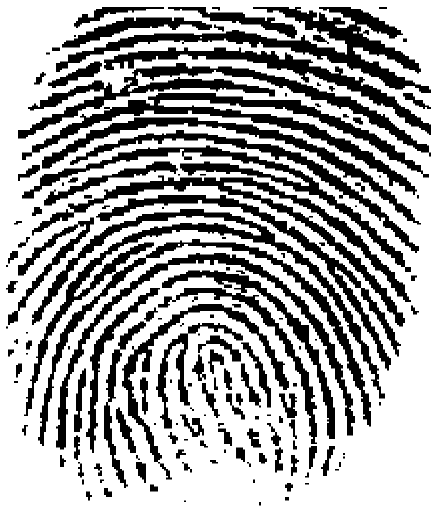
4.5 Thinning

In thinning process, the binarized input image is thinned to one-pixel width after it been morphologically thinned using “bwmorph” function in MATLAB. The result is shown in Figure 4.4 (a) and the comparison between binarized and thinned image is shown in Figure 4.4 (b) and Figure 4.4 (c).



(a)





(b) (c)

Figure 4.4: Result for (a) Thinned image (b) and (c) comparison thinned and the binarized image

4.6 Found minutiae

The thinned input image will be the input of the minutiae extraction system. End point and bifurcation point of the minutiae will be extracted. End points are the red circle while bifurcation points are the green circle. The end point is shown in Figure 4.5 (a) and bifurcation point of the fingerprint minutiae is shown in Figure 4.5 (b). The result of the minutiae point found on the thinned input image is shown in Figure 4.5 (c).

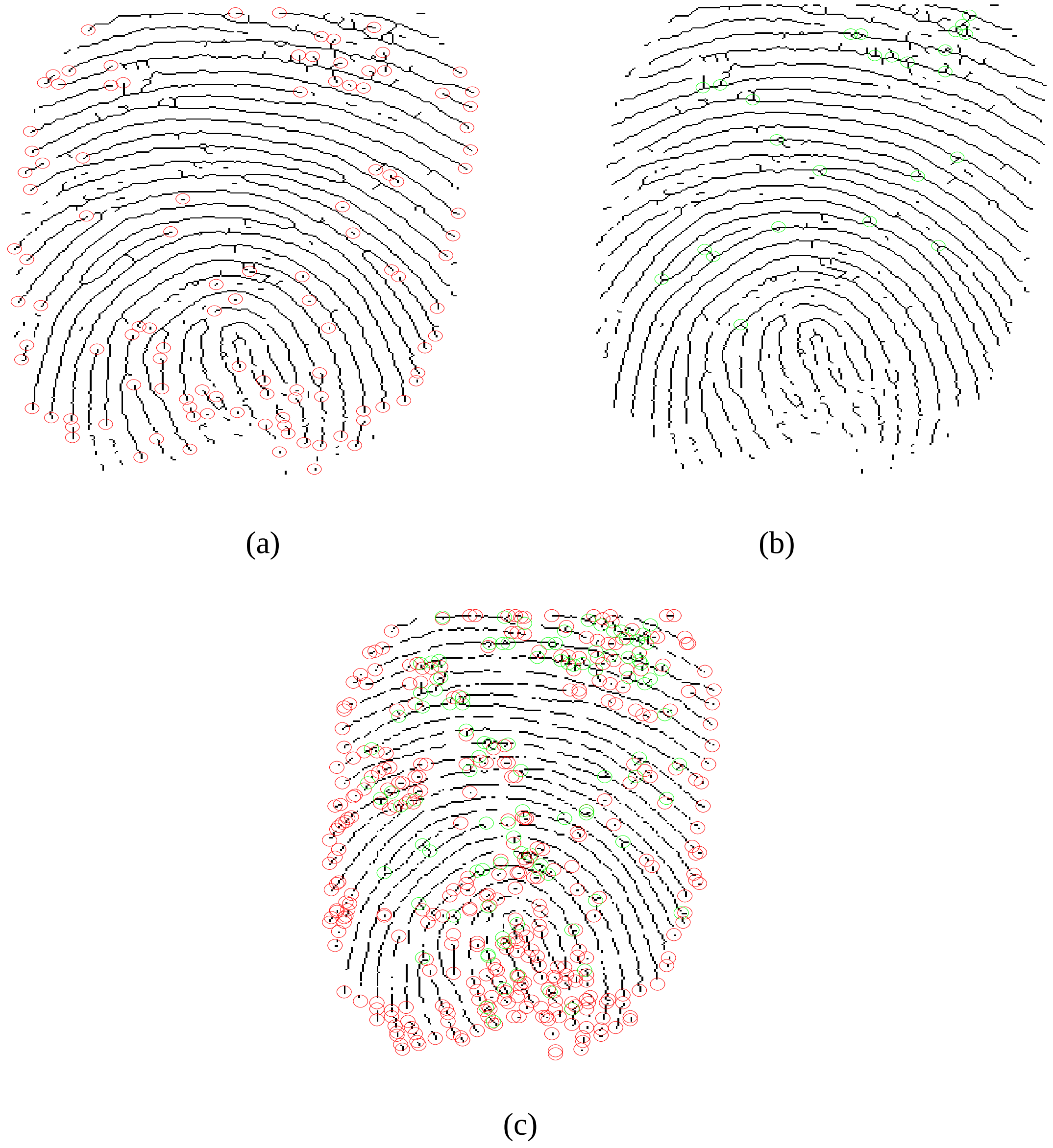
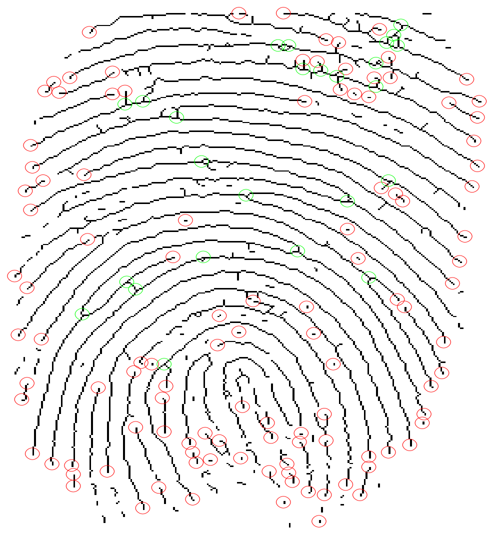

(a) (b) (c)

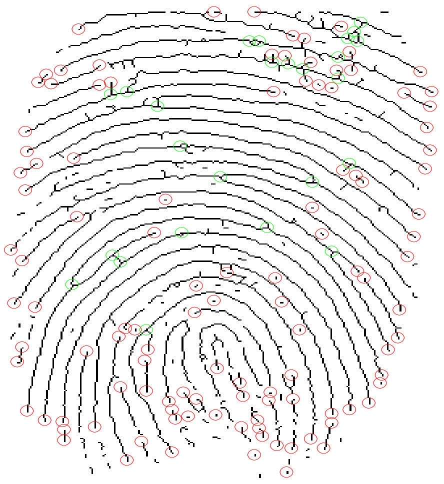
Figure 4.5: Result for (a) end point (b) bifurcation point(c) result of the minutiae point found

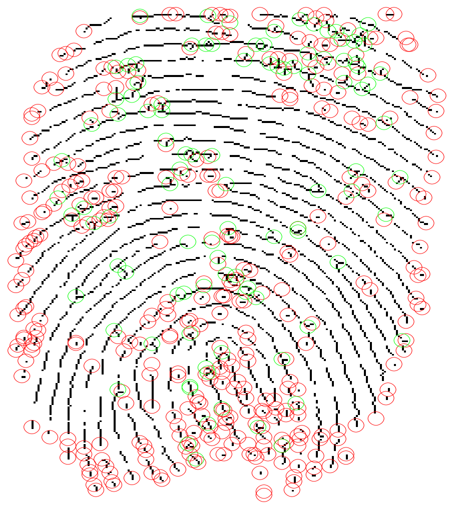
4.7 Reduces false minutiae.

After the minutiae have been extracted, there are still lots of false minutiae in the input fingerprint image. Figure shows the result of reduced false minutiae and the comparison between before and after false minutiae has been reduced will be shown in figure.



(a)





(b) (c)

Figure 4.6: Result for (a) false minutiae (b) and(c) comparison between before and after false minutiae.

4.8 GUI Analysis

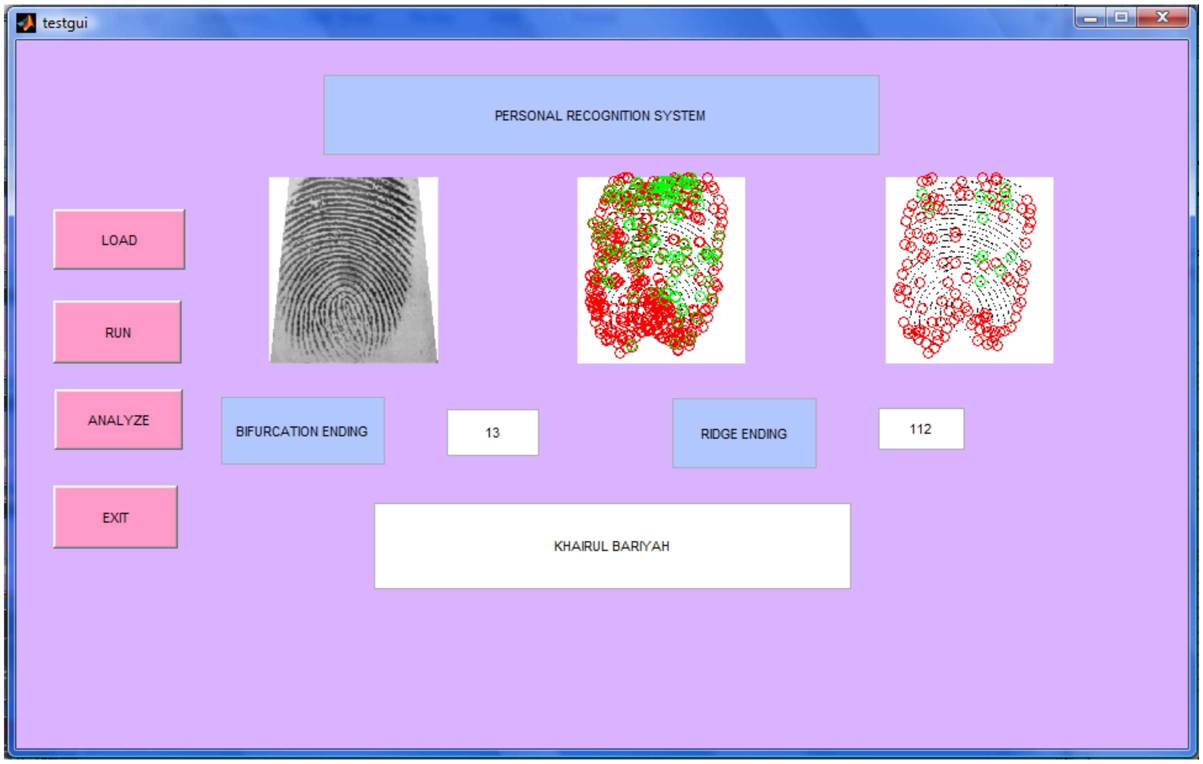
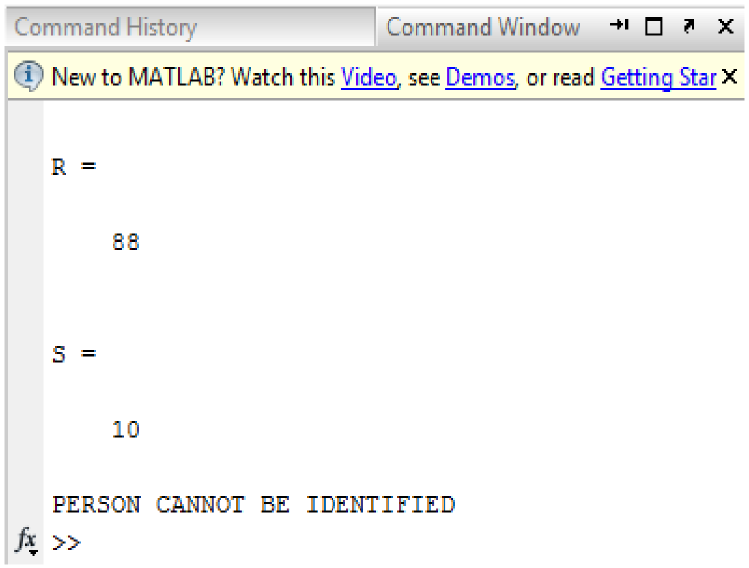


Figure 4.7: GUI Simulation system

For the purpose of recognition the button ANALYZE will make our system to display the information to recognize the owner of the fingerprint. It will display the information like name of the owner such as in Figure 4.6. If the image is being able to recognized, this had form a successful system in term of recognition. When the image which is does not have the data in the database, the display board will display 'PERSON NOT IDENTIFIED'. Both of this condition is important to determine the effectiveness of this system in term of accuracy.


Command History
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Star

```
R =

    88


S =

    10

PERSON CANNOT BE IDENTIFIED
>>
```

Figure 4.8: Command window for unidentified image.

The observation from the figure above show the identification for 'PERSON CANNOT BE IDENTIFIED' when the image cannot match up with the database. If there is 0 related data that have the same and exact value it won not display any information. If the image used is not computes data process at the recognition step cannot be match up with the information. This strongly shows that the system had a good capability for identification. This case of mismatch between input and database had not been occurred that had cause the percentage (%) of FAR is low.

4.9 Effectiveness

Each fingerprint in the test set was matches with the fingerprint in the set (one-to-one matching process). A matching was labelled correct if the matched fingerprint was among the fingerprints of same individuals, and incorrect otherwise. For every fingerprint in database, that consider it is an nput fingerprint and match with other 10 coming from the same finger. Denote reject_num as the number of rejected times. If the matching score is not in a range, then increase reject_num by one. Denote the match_num as the number of total matching times. Then the FRR (False Reject Rate) will get by equation:

$$FRR(\%) = \frac{reject_{num}}{total\ iamge}\ x\ 100 \tag{4.1}$$

$$FRR(\%) = \frac{0}{24}\ x\ 100 = 0\%$$

For every fingerprint in database, match it with other fingerprint coming from other fingers. Denote accept_num as the number of accepted times. If the matching score is higher than a threshold, then increase accept_num by one. Denote the match_num as the number of total matching times. Then, the FAR (false accept rate) will get by equation:

$$Accuracy(\%) = \frac{no.of\ image\ correctly\ identified}{total\ image}\ x\ 100\% \tag{4.2}$$

$$Accuracy(\%) = \frac{24}{24}\ x\ 100\% = 100\%$$

At the end of the process, it finally calculates the value of accuracy of system. By analysis and calculation shows that the system achieve a good performance where by the accuracy value is 100% of correctness with 0% of FFR. When the system received the correct image it will compute and match up with the correct data. Only after that it will display information. If there are differences in during matching session even 1% this system will display “PERSON CANNOT BE IDENTIFIED”. Not even one image can be accepted and be recognized if there are not listed in database. From this analysis, the objective was achieve to produce a system with high accuracy in doing recognition of a person.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Introduction

This system had achieved a good accuracy in personal identification. This chapter summarizes the important finding from the analysis carried out in this project. It also includes some recommendation and suggestion for future work to improve the system and use that system.

5.2 Conclusion

The main focus of this project is to introduce fingerprint matching system which is capable of verifying identities in real time. This is done by a pre-processing process which extracting the minutia from the fingerprint input image which is used is an inkless image fingerprint as database. Besides, based on the analysis and finding that had been done a few conclusions can be made with regard to the system in term of complexity, accuracy, reliability, application and many more.

As a conclusion, the implemented minutia extraction algorithm is accurate and fast in minutiae extraction. The alignment based elastic matching algorithm is capable of finding the correspondences between minutiae without resorting to exhaustive search. It provides a good performance, because it has the ability to adaptively compensate for the nonlinear deformations and inexact pose transformations between different fingerprints. Experimental results show that our system achieves excellent performance in a real on-line verification environment. It meets the response time requirements of on-line verification with high accuracy. Using the core to determine the reference point can speed up the matching process. This is very importance for the performance of the real time system. This algorithm is more suitable for the real time system. The reliability of the fingerprint core detection needs to be improved, especially for the poor fingerprint images. The system are working on more reliable method to find the reference minutiae points.

5.3 Future Recommendations

For this system, it still can make some future study for improvement and upgrading the effectiveness and application. First, there are a number of limitations of the initial implementation described in this project. This project can be improved by using the feature extraction of Gabor filter-based method in real time verification system. Other than that, by using more Gabor filter a more improve translation and rotation invariance will be obtain.

Another improvement that can add in the recognition technique is by train data in the neural network system. Know that neural network is a very powerful method whereby the recognition is based on the knowledge that gave to the system. Because of that the system will automatically learn about the situation and decision. Besides that, other matching technique can be applied such as Hamming Distance, and also Hough Transformation. This technique might form a better calculation of uniqueness value of each image.

Another idea for future recommendation is to improve the reliability of the fingerprint core detection, especially for the poor fingerprint images. Therefore, the system will be much widely use by increasing the number of sample image.

REFERENCES

[1] Unknown, “*What Is Biometrics Recognition*” [Online]. Available at: http://www.wisegeek.com/what-is-biometrics-recognition.htm [Accessed: 17 August 2010]

[2] Manhua Liu, “Fingerprint Classification Based on Adaboost Learning From Singularity Features,” Shanghai Jiao Tong University, Shanghai, vol. 43, pp. 1062-1070, 2010.

[3] Paul Reid, “Biometric for Network Security”, Prentice Hall, vol. 1, pp.6, 2004

[4] Nalini Kanta and Ruud Bolle, “*Handbook of Automatic Fingerprint Recognition Systems.*Hawthorne New York, 2004

[5] D. Maltoni, D. Maio, A.K. Jain, and S. Prabhakar.” Handbook of Fingerprint Recognition”. Springer, New York, 2003.

[6] Unknown, “*Brief History of Fingerprint Identification*” [Online]. Available at: http://members.cox.net/bnminalpine/history.htm [Accessed: 24 August 2010]

[7] Unknown, “*The History of Fingerprint*” [Online]. Available at: http://onin.com/fp/fphistory.html [Accessed: 24 August 2010]

[8] Anil Jain, 2004, “*Fingerprint Classification And Matching*”, Michigan State University Of East Lansing.

[9] Unknown, “*How Scanner Work*” [Online]. Available at: http://computer.howstuffworks.com/fingerprint-scanner.htm [Accessed: 17 January 2010]

[10] Raymond Thai (2003), *"Fingerprint Image Enhancement and Minutiae Extraction"*, PhD Thesis, University of Western Australia.

[11] Davide Maltoni "*A Tutorial on Fingerprint Recognition*" Biometric Systems Laboratory, University of Bolognavia Sacchi 3, Italy

[12] *TheMathworks*.[Online] Available: http://www.mathworks.com/products/matlab/. [Accessed February 23, 2009].

[13] Sharat S. Chikkerur (June,2007)"*Online Fingerprint Verification System*", Master Thesis, State University of New York at Buffalo.

[14] Unknown, "*How Fingerprint Recognition Works*" [Online]. Available at: http://software.ivertech.com/_ivertechArticle5169_HowFingerprintRecognition Works htm [Accessed: 17 January 2010]

[15] Markus Huppman, *" Fingerprint Recognition by Matching of Gabor Filter-Based Patterns,"* University Munchen, Munich, Germany, 2007.

[16] Nasir Rehan and Khalid Rashid (2004),*Multi-matcher and Based Fingerprint Identification System.* Journal of Applied Sciences 4 (4). Pp. 611-618

[17] A. Jain, Lin Hong, R. Bolle, "*Online Fingerprint Verification",*IEEE Trans. Pattern Anal. Mach. Intell. 19 (4)(1997) pp. 303-314.

[18] N. Ratha, S.Chen, and A. K. Jain, Adaptive Flow Orientation Based Feature Extraction in Fingerprint Images, *Pattern Recognition*, Vol. 27, No. 11, pp. 1657-1672, 1995

[19] Mohamed K. Shahin, Ahmed M. Badawi, and Mohamed S. Kamel," *On-Line,*

Low-Cost and Pc-Based Fingerprint Verification System Based on Solid-State Capacitance Sensor", [Online]. Available at: http://www.eng.cu.edu.eg/users/ambadawi/finger1.pdf

[20] Lam, L., Seong-Whan Lee, and Ching Y. Suen, "*Thinning Methodologies-A Comprehensive Survey*," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 14, No. 9, September 1992, page 879, bottom of first column through top of second column.

[21] J Edgar Hoover, Federal Bureau of Investigation,, "*Guide to Fingerprint Recognition*", Department of Justice, Classification of Fingerprints, US Government

[22] Sharat Chikkerur∗, Alexander N. Cartwright, Venu Govindaraju," *Fingerprint enhancement using STFT analysis*", Pattern Recognition 40 (2007) 198–211.

[23] Ying Jiea,∗,YuanYi fanga, Zhang Renjiea, Song Qifab "*Fingerprint minutiae matching algorithm for real time system*", Pattern Recognition 39 (2006) pp 143 – 146

[24] Arceli and Baja., "A Width Independent Fast Thinning Algorithm", IEEE Transactions On Pattern Analysis And Machine Intelligence, 1984

[25] F.A. Afsar, M.Arif and M. Hussin," *Fingerprint Identifivation and Verification System using Minutiae Matching*", Department Of Computer & Information Science, Pakistan Institude of Engineering & Applied Sciences,Islamabad, Pakistan

[26] L.C. Jain, U. Halici, I. Hayashi, S.B. Lee, and S. Tsutsui, “Intelligent biometric techniques in fingerprint and face recognition”, The CRC Press, 1999.

[27] Manvjeet Kaur, Mukhwinder Singh, Akshay Girdhar, and Parvinder S. Sandhu “Fingerprint Verification System using Minutiae Extraction Technique”, World Academy of Science, Engineering and Technology 46 (2008)

APPENDIX A

```
function varargout = testgui(varargin)
% TESTGUI M-file for testgui.fig
%      TESTGUI, by itself, creates a new TESTGUI or raises the existing
%      singleton*.
%
%      H = TESTGUI returns the handle to a new TESTGUI or the handle to
%      the existing singleton*.
%
%      TESTGUI('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in TESTGUI.M with the given input
arguments.
%
%      TESTGUI('Property','Value',...) creates a new TESTGUI or raises
the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before testgui_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to testgui_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help testgui

% Last Modified by GUIDE v2.5 21-Nov-2010 02:47:01

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @testgui_OpeningFcn, ...
                   'gui_OutputFcn',  @testgui_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before testgui is made visible.
function testgui_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to testgui (see VARARGIN)

% Choose default command line output for testgui
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes testgui wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = testgui_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[f,rep]=uigetfile('*.bmp;*.BMP;*.tif;*.TIF;*.jpg','Open An Fingerprint
image');
OriginalImage=imread([rep,f]);
axes(handles.axes1);
imshow(OriginalImage);
handles.S = OriginalImage;
guidata(hObject, handles);

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=handles.S;
J = imadjust(I);
K= histeq(J);

%% binarize image

L=K(:,:,1)>52;
```

```
L=~L;

%% thinning image

M=bwmorph(L,'thin','inf');

%% Find minutiae

fun = @minutie;
N = nlfilter(M,[3 3],fun);

%% RIDGE ENDING

NEnd=(N==1);
%imshow(NEnd)
NEndLab=bwlabel(NEnd);
End=regionprops(NEndLab,'Centroid');
dEnd=round(cat(1,End(:).Centroid));

axes(handles.axes2),imshow(~N);
hold on
plot(dEnd(:,1),dEnd(:,2),'ro');
P=size(dEnd,1);

%% Ridge Bifurcation

NBif=(N==3);
NBifLab=bwlabel(NBif);
Bif=regionprops(NBifLab,'Centroid','Image');
dBif=round(cat(1,Bif(:).Centroid));
plot(dBif(:,1),dBif(:,2),'go');
Q=size(dBif,1);

%% remove spurios
%removing spurious using distance

D=6;
%% removing spurious minutiae between bifurcation and ending

Di=DistEuclidian(dBif,dEnd);
s=Di<D;
[i,j]=find(s);
dBif(i,:)=[];
dEnd(j,:)=[];

%% removing spurious minutiae between two bifurcations

Di=DistEuclidian(dBif);
s=Di<D;
[i,j]=find(s);
dBif(i,:)=[];


%% removing spurious minutiae between two endings
```

```
Di=DistEuclidian(dEnd);
s=Di<D;
[i,j]=find(s);
dEnd(i,:)=[];

%% display figure after remove false minutiae

hold off

hold on
%plot(dEnd(:,1),dEnd(:,2),'ro');
%plot(dBif(:,1),dBif(:,2),'go');
hold off
R=size(dEnd,1)
s=size(dBif,1)


matched1=double(s).*double(N);
matched=uint8(matched1);

axes(handles.axes3),imshow(~N);
hold on
plot(dEnd(:,1),dEnd(:,2),'ro')
plot(dBif(:,1),dBif(:,2),'go')
hold off
set(handles.edit2,'String',s)
set(handles.edit4,'String',R)
handles.S=s;
guidata(hObject, handles);

% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
s=handles.S;

switch (s)
    case 18
         set(handles.edit3,'String','KHAIRUNISA')

    case 13
          set(handles.edit3,'String','KHAIRUNNISA')

    case 23
          set(handles.edit3,'String','KHAIRUNNISA')

    case 32
          set(handles.edit3,'String','SITI SARAH')

    case 35
         set(handles.edit3,'String','SITI SARAH')
    case 37
         set(handles.edit3,'String','SITI SARAH')

    case 22
         set(handles.edit3,'String','NURUL NADIAH')

    case 30
         set(handles.edit3,'String','NURUL NADIAH')

    case 33
         set(handles.edit3,'String','NURUL NADIAH')

    case 36
         set(handles.edit3,'String','ALIF AZIZ')

    case 31
         set(handles.edit3,'String','ALIF AZIZ')

    case 21
         set(handles.edit3,'String','ALIF AZIZ')

    case 34
         set(handles.edit3,'String','NUR HAFIZAH')

    case 33
         set(handles.edit3,'String','NUR HAFIZAH')

    case 26
         set(handles.edit3,'String','NUR HAFIZAH')

    case 49
         set(handles.edit3,'String','NORAINI')

    case 51
         set(handles.edit3,'String','NORAINI')

    case 34
         set(handles.edit3,'String','NORAINI')

    case 39
         set(handles.edit3,'String','NUR IZYAN')

    case 22
         set(handles.edit3,'String','NUR IZYAN')

    case 31
         set(handles.edit3,'String','NUR IZYAN')

    otherwise
        set(handles.edit3,'String','PERSON CANNOT BE IDENTIFIED')

end
% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
pos_size=get(handles.figure1,'Position');
delete(handles.figure1)


function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as
a double


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as
a double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as
a double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4 as
a double


% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as
a double


% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%        str2double(get(hObject,'String')) returns contents of edit6 as
a double


% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```