FINGERPRINT RECOGNITION USING NEURAL NETWORKS

submitted by

KENNIE YEOH ENG HOE
WEK 98005

A Report for a Final Year Project, submitted as a partial requirement for the degree of Bachelor of Computer Science & Information Technology, University of Malaya, Kuala Lumpur 2000/2001.

Faculty of Computer Science & Information Technology
University of Malaya
Kuala Lumpur

January 2001

Abstract

Traditional methods of fingerprint verification uses either complicated feature detection algorithms that are not specific to each fingerprint, or compare two fingerprint images directly using image processing tools. The former involves very complicated calculations and tedious algorithms, and the latter tend to work poorly. In this paper it is described a new method which takes the middle ground.

This paper studies the implementation of the Fast Fourier Transform and Artificial Neural Networks into the recognition of fingerprints. With tests conducted on the implementation of the Fourier Transform as a method of fingerprint feature extraction, the use of the Fourier Transform was proven not to work. Alternatively, patch-matching algorithm was developed in success of the Fourier Transform method when results were not favourable to it. A flow of the process goes from fingerprint acquisition using inkpads and a scanner, followed by image pre-processing steps to produce cleaner more visually acceptable images. Next, features are extracted from the fingerprint and later fed into neural networks for recognition.

This project aims at producing a system study on various factors that need to be taken into consideration for fingerprint recognition, from response time, to stringency levels and of course, accurate recognition of verified fingerprints.

Acknowledgement

I would like to extend my sincere gratitude to my supervisor En. Omar Zakaria, my moderator Dr. Mazliza Othman and lecturers, Dr. Selvanathan and En. Ibrahim Abu Bakar on their direction and benevolence in enabling me to come up with this undergraduate study. I would also like to thank my fellow course mates whom together went through tedious researching to uncover resources on the Internet and available libraries.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Fingerprints are imprints or impressions of patterns formed by friction ridges of the skin on the fingers and thumbs. The friction ridges, otherwise known as fingerprints can be positively identified through comparison of fingerprints. Fingerprints serves as an infallible means of personal identification, because the ridge arrangement on every finger of every human being is unique and does not alter with growth or age. The probability that two fingerprints are alike is approximately 1 in 1.9×10^{15}. This means, in approximately 7 million times the size of the US population (1999), there are probably 2 people of similar prints. Not surprisingly, this unique means of personal identification has been used for a very long time.

Because of immutability and individuality, the use of fingerprints for identification has always been of great interest to pattern recognition to researches and law enforcement people. Back in history, the Egyptians and Chinese have been using fingerprints to identify criminals and record business transactions. The first automatic (or semi-automatic) fingerprint recognition system was created by the FBI (U.S. Federal Bureau of Investigation) [3]. Since then volumes of fingerprint databases and amount of requests for identification increased constantly, forcing the industry to automate and classify fingerprints to improve recognition efficiency.

Although the first approaches to automatic fingerprint classification was proposed many years ago [8] [9], dating back to 1968, manual classification is still used, with great expense of time. The problem of fingerprint classification is very complex : a good classification system should be very *reliable* (must not mis-classify fingerprints), it should be *selective* (the fingerprint database has to be partitioned in a large number of non-overlapping classes with about the same cardinality) and it should be *efficient* (each fingerprint should be processed in a short amount of time).

Conventionally, fingerprint recognition has been conducted via either statistical or syntactic approaches. In the statistical approach, a fingerprint is represented by a set of *n*–dimensional feature vector and the decision making process is determined by a similarity measure such as a discriminant function. In the syntactic approach, a pattern is represented as a string, tree [1] or graph [2] of fingerprint features or pattern primitives and their relations. The decision making process is then simply a syntax analysis or parsing process. However, in this project, the work done will be focused on the implementations of neural networks as a method of fingerprint classification and recognition. Neural network works similar to the way neurons work in the human body. The topic on neural networks will be further discussed in Chapter 4 – Artificial Neural Networks.

1.2 Project Objective

Many studies on automated fingerprint recognition are based on technical mathematical framework. The ability of the human brain to classify and differentiate pictures and patterns has been much overlooked. This is until one actually takes a step into finding out how a computer recognizes two similar pictures. It is interesting to study how computers mimic nature in solving problems of this kind.

This study is with the intent of developing an Artificial Neural Network based fingerprint recognition system. A multi-layer neural network with back-propagation learning algorithm will be constructed to accomplish this task. On the parallel, research is also carried out on current fingerprint recognition methods and technologies. This thesis carries the following objectives :

i.) Implementing biometrics (fingerprints) as a form of verification (computer security).

ii.) Develop and evaluate an Artificial Neural Network based fingerprint classifier.

iii.) Analyse the feasibility of the Fourier Transform algorithms and other alternative methods in feature extraction of fingerprints.

iv.) Study and analyse factors that needs to be considered when developing systems for fingerprint recognition.

1.3 Project Scope

Fingerprints and other biometric technologies hold uniqueness and strength to fraudulent identification and verification (replacement part surgery is outside the scope of this paper). Automated fingerprint verification has received much attention from various organizations around the world. As such, successful algorithms and researches remain closely within the walls of the developer.

1.3.1 Core Point and Image Size

A common reference point is needed in when features are to be extracted from the image for classification. The boundary faced in this thesis is the automated detection of the core point (centre of fingerprint). As such, human intervention is needed in scanning the fingerprint image in an approximate size of between 250 x 250 to 500 x 500 and selecting the image with the core as much in the centre as possible. There are four methods of fingerprint sampling, being and inkpad, optical, silicon and ultrasound. Due to unavailability of electronic biometric devices for fingerprint sampling, two different inkpads were used in place of biometric devices. An ordinary inkpad and an Eazyprint™ inkpad, developed specifically for fingerprint stamping.

1.3.2 Fingerprint Irregularities

In this paper fingerprint samples are collected using the inkpad method, resulting in imperfections of the fingerprint such as ridge gaps, which are usually caused by (a) skin folds and contiguous ridges, or (b) the spreading of ink due to finger pressure, or (c) excessive/insufficient inking or by smearing during rolling of the finger [4]. Owing to the nature of fingerprints, scars are inevitable. The effect of scars are ignored and considered a part of the fingerprint. Other irregularities are fingerprints that are shifted and/or rotated. As such, certain preconditions are set when sampling fingerprints, which are:

a.) Do not apply excessive/insufficient ink

b.) Apply moderate pressure

c.) Do not rotate fingerprint too much

1.3.3 Software Dependency

The testing and implementations of methods and algorithms will be done on the Matlab platform. This results in the developed application being software dependant. All program codes and the Graphical User Interface (GUI) will be built on Matlab.

Chapter 2

Literature Review

2.1 Introduction to Biometrics

Biometrics is best defined as a measurable physiological and/or behavioural characteristic that can be utilized to verify the identity of an individual [10]. They are of interest and significance in areas where it is important to verify the true identity of an individual. These techniques were employed primarily in high security applications. Existing systems in immigration, law enforcement, physical and logical access control, time monitoring and other areas have proven the concept and use of biometrics well beyond doubt.

Before biometrics was introduced as a form of identification, PIN's (personal identification numbers) were one of the first identifiers to offer automated recognition. However, it should be understood that this means recognition of the PIN, not necessarily the person who has provided it. The same applies with cards and other tokens. We may easily recognise the token, but it could be presented by anybody. Using the two together provides a slightly higher confidence level, but this is still easily compromised if one is determined to do so. A biometric however cannot be easily transferred between individuals (replacement part surgery is outside the scope of this paper) and represents a unique identifier between individuals.

2.1.1 Biometric Background

It is tempting to think of biometrics as being a futuristic technology that we shall all be using together in the near future, as we see in movies. This popular image suggests that they are a product of the late twentieth century computer age. In fact, the basic principles of biometric verification were understood and practiced thousands of years earlier. Early Egyptians and Chinese routinely employed biometric verification in a number of everyday business situations. There are many references to individuals being formally identified via unique physiological parameters such as scars, measured physical criteria or a combination of features such as complexion, eye colour, height and so on. This would often be the case in relation to transactions with regard to legal proceedings of various descriptions. If you remember, we had to declare the eye colour, hair colour, and also visible scars on our face in the passport. Of course, they didn't have automated electronic biometric readers and computer networks, and they certainly were not dealing with the numbers of individuals that we have to accommodate today, but the basic principles were similar.

Later, in the nineteenth century there was a peak of interest as researchers into criminology attempted to relate physical features and characteristics with criminal tendencies. This resulted in a variety of measuring devices being produced and much data being collected. The results were not conclusive but the idea of measuring individual physical characteristics seemed to stick. Subsequently, development of fingerprinting became the international methodology among police forces for identity verification.

With this background, various projects were initiated to look at the potential of biometrics and one of these eventually led to a large hand geometry reader being produced. It wasn't pretty, but it worked and motivated it's designers to further refine the concept. Eventually, a small specialist company was formed and a much smaller, and considerably enhanced hand geometry reader became one of the cornerstones of the early biometric industry. This device worked well and found favour in numerous biometric projects around the world. In parallel, other biometric methodologies such as fingerprint verification were being steadily improved and refined to the point where they would become reliable, easily deployed devices. In recent years, we have also seen much interest in iris scanning and facial recognition techniques, which offer the potential of a non-contact technology, although there are additional issues involved in this respect.

2.1.2 Biometric Methodologies

A number of applicable biometric methodologies are much in use and research these days. They include fingerprints, hand geometry, voice patterns, retinal and iris scanning, signature verification, facial recognition and other techniques. You will see reference to a number of biometrics, some of which are rather impractical even if technically interesting. The popular biometrics seems to gravitate at present around the following methodologies [10]:

2.1.2.1 Fingerprint Verification

There are a variety of approaches to fingerprint verification. Some of them try to emulate the traditional police method of matching minutiae, others are straight pattern matching devices, and some adopt a unique approach all of their own. Some of them can detect when a live finger is presented, some cannot. There are greater varieties of fingerprint devices available than any other biometric at present. Potentially capable of good accuracy (low instances of false acceptance) fingerprint devices can also suffer from usage errors among insufficiently disciplined users (higher instances of false rejection) such as might be the case with large user bases. Workstation access application area can be based almost exclusively around fingerprints, due to the relatively low cost, small size (easily integrated into keyboards) and ease of integration.

2.1.2.2 Hand Geometry

As the name suggests, hand geometry is concerned with measuring the physical characteristics of the users hand and fingers, from a three-dimensional (3D) perspective in the case of the leading product. One of the most established methodologies, hand geometry offers a good balance of performance characteristics and is relatively easy to use. This methodology may be suitable where we have larger user bases or users who may access the system infrequently and may therefore be less disciplined in their approach to the system. Ease of integration into other systems and processes, coupled to ease of use makes hand geometry an obvious first step for many biometric projects.

2.1.2.3 Voice Verification

A potentially interesting technique bearing in mind how much voice communication takes place with regard to everyday business transactions. Voice verification techniques are based on the waveform from the voice. This works such that the user utters a word or sentence for verification. Some designs have concentrated on wall-mounted readers whilst others have sought to integrate voice verification into conventional telephone handsets. However, the enrolment procedure has often been more complicated than with other biometrics leading to the perception of voice verification as unfriendly in some quarters.

2.1.2.4 Retinal Scanning

The retina is the layer at the back of the eyeball, which is sensitive to light. An established technology where the unique patterns of the retina are scanned by a low intensity light source via an optical coupler. Retinal scanning has proved to be quite accurate in use but does require the user to look into a receptacle and focus on a given point. This is not particularly convenient if you are a spectacle wearer or have concerns about intimate contact with the reading device. For these reasons retinal scanning has a few user acceptance problems although the technology itself can work well. The leading product underwent a redesign in the mid nineties, providing enhanced connectivity and an improved user interface, however this is still a relatively marginal biometric technology.

2.1.2.5 Iris Scanning

The iris is the circular coloured membrane surrounding the pupil, which controls the amount of light that enters the eye. The iris also displays the colour of the eye (blue,green,etc.). Iris scanning is undoubtedly the less intrusive of the eye related biometrics. It utilises a fairly conventional CCD camera element and requires no intimate contact between user and reader. In addition it has the potential for higher than average template matching performance. It has been demonstrated to work with spectacles in place and is one of the few devices which can work well in identification mode. Traditionally, iris scanning devices were not easily used and integrated into systems.

2.1.2.6 Signature Verification

Signature verification enjoys a synergy with existing processes that other biometrics do not. People are used to signatures as a means of transaction related identity verification and would mostly see nothing unusual in extending this to encompass biometrics. Signature verification devices have proved to be reasonably accurate in operation and obviously lend themselves to applications where the signature is an accepted identifier. Curiously, there have been relatively few significant applications to date in comparison with other biometric methodologies. If your application fits, it is a technology worth considering, although signature verification vendors have tended to have a somewhat chequered history.

2.1.2.7 Facial Recognition

A technique which has attracted considerable interest and whose capabilities have often been misunderstood. Extravagant claims have sometimes been made for facial recognition devices that have been difficult if not impossible to substantiate in practice. It is one thing to match two static images (all that some systems actually do - not in fact biometrics at all), it is quite another to unobtrusively detect and verify the identity of an individual within a group (as some systems claim). Facial recognition systems have had limited success in practical applications. If technical obstacles can be overcome, we may eventually see facial recognition become a primary biometric methodology.

There are other biometric methodologies including the use of scent, ear lobes and various other parameters. Whilst these may be technically interesting, they are not considered at this stage to be workable solutions in everyday applications. Those listed above represent the majority interest and would be a good starting place for one to consider within a biometric project.

2.2 Studies on Fingerprints in Biometrics

You will see reference to a number of biometrics, some of which are rather impractical even if technically interesting. The 'popular' biometrics seems to gravitate at present around the following methodologies. For serious large-scale positive-identification applications, no other currently available biometric technology comes close to fingerprints. The strongpoint of fingerprint identification technologies are [5] :

- *Well established* : fingerprint identification has been used in law enforcement applications over the past 100 years, and has become the *de facto* international standard for positive identification of individuals.
- *Proven* : AFIS (Automated Fingerprint Identification System) technology has been developed, refined and proven in demanding law enforcement applications over the last two decades.
- *Legally accepted* : The legal precedents which have been established in the U.S. court system make fingerprints the only biometric proof of identification which is readily accepted in legal proceedings.
- *Mature* : Fingerprint identification technologies are well beyond the R&D stage, as evidenced by the fact that a number of viable manufacturers produce competing products for a wide-spread and well established

marketplace. In most other biometrics, the technology is only available from a single vendor, making any large-scale long-term application very risky.

There are a variety of approaches to fingerprint verification. Some of them try to emulate the traditional method of matching minutiae [11] [12] [13], others are straight pattern matching devices [19] [25], and some adopt a unique approach all of their own, including fringe patterns and ultrasonic. Some of them can detect when a live finger is presented, some cannot. There is a greater variety of fingerprint devices available than any other biometric at present.

Potentially capable of good accuracy (low instances of false acceptance) fingerprint devices can also suffer from usage errors among insufficiently disciplined users (higher instances of false rejection) such as might be the case with large user bases. One must also consider the transducer / user interface and how this would be affected by large scale usage in a variety of environments. Fingerprint verification may be a good choice for in house systems where adequate explanation and training can be provided to users and where the system is operated within a controlled environment. It is not surprising that the workstation access application area seems to be based almost exclusively around fingerprints, due to the relatively low cost, small size (easily integrated into keyboards) and ease of integration.

2.2.1 What are Fingerprints?

A fingerprint is the impression made by the papillary ridges on the ends of the fingers and thumbs. Unique fingerprint characteristics that are constant are shape of the fingerprint, number of lines, and the distances between specific points. Fingerprints afford an infallible means of personal identification, because the ridge arrangement on every finger of every human being is unique and does not alter with growth or age. Fingerprints serve to reveal an individual's true identity despite personal denial, assumed names, or changes in personal appearance resulting from age, disease, plastic surgery, or accident. The practice of utilizing fingerprints as a means of identification, referred to as dactyloscopy [7], is an indispensable aid to modern law enforcement.

Each ridge of the epidermis (outer skin) is dotted with sweat pores for its entire length and is anchored to the dermis (inner skin) by a double row of peg like protuberances, or papillae. Injuries such as superficial burns, abrasions, or cuts do not affect the ridge structure or alter the dermal papillae, and the original pattern is duplicated in any new skin that grows. An injury that destroys the dermal papillae, however, will permanently obliterate the ridges.

Any ridged area of the hand or foot may be used as identification. However, finger impressions are preferred to those from other parts of the body because they can be taken with a minimum of time and effort, and the ridges in such impressions form patterns (distinctive outlines or shapes) that can be readily sorted into groups for ease in filing.

The Galton-Henry system of fingerprint classification, published in June 1900, was officially introduced at Scotland Yard in 1901 and quickly became the basis for its criminal-identification records. The system was adopted immediately by law-enforcement agencies in the English-speaking countries of the world and is now the most widely used method of fingerprint classification.

2.2.2 Fingerprint Characteristics

Fingerprints are classified in a three-way process: by the shapes and contours of individual patterns, by noting the finger positions of the pattern types, and by relative size, determined by counting the ridges in loops and by tracing the ridges in whorls. The information obtained in this way is incorporated in a concise formula, which is known as the individual's fingerprint classification.

Fingerprints are first classified into their high-level features and low-level features. The high-level features are obvious and can be seen with the naked eye. Essentially, there are 4 main classes of fingerprints, being whorl, arch, loop or a mixture of the two or more of the above. Loops make up nearly 2/3 of all fingerprints, whorls are nearly 1/3, and perhaps 5-10% are arches [14]. The rest are combinations of classes. The Federal Bureau of Investigation (FBI) in the United States recognizes eight different types of patterns: Plain Arch, Tendent arch, Radial Loop, Ulnar Loop, Plain Whorl, Central Pocket, Double Loop and Accidental Loop [3].

(a) Whorl (b) Arch (c) Loop

Figure 2.1 : Three main classes of fingerprints. (a) Whorl, (b) Arch and (c) Loop.

Whorls are usually circular or spiral in shape. Arches have a moundlike contour, while tented arches have a spikelike or steeplelike appearance in the center. Loops have concentric hairpin or staple-shaped ridges and are described as "radial" or "ulnar" to denote their slopes; ulnar loops slope toward the little finger side of the hand, radial loops toward the thumb. Loops constitute about 65 percent of the total fingerprint patterns; whorls make up about 30 percent, and arches and tented arches together account for the other 5 percent. The most common pattern is the ulnar loop [7].

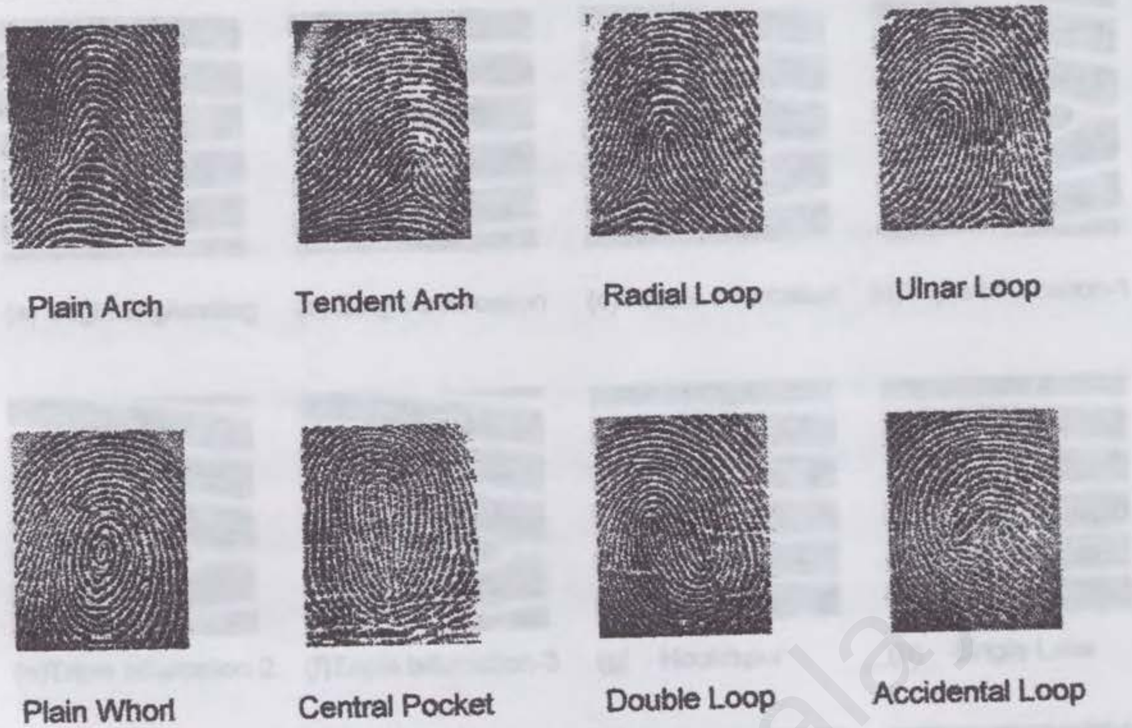


Figure 2.2 : Fingerprint subclasses. Each fingerprint in the figure belongs to different subclasses of the FBI's classification scheme; from the left to the right and from the top to the bottom: Plain Arch, Tendent arch, Radial Loop, Ulnar Loop, Plain Whorl, Central Pocket, Double Loop and Accidental Loop.

Subsequently, low-level features are the important features that are used in fingerprint recognition. They are classified from their comprehensive features, otherwise known as minutiae. Listed in Figure 2.3 are the types of minutiae [6] :

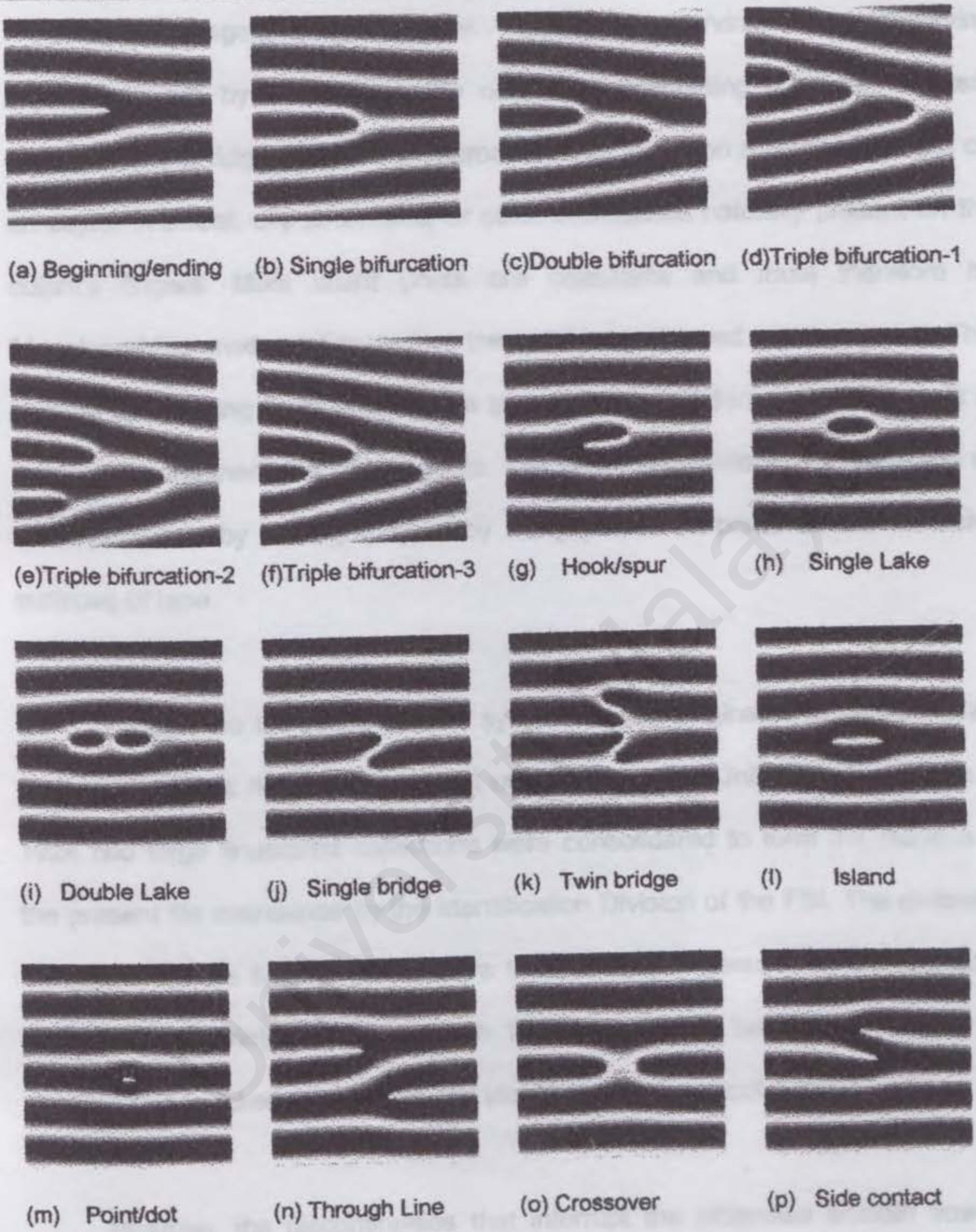


Figure 2.3 : Types of minutiae in fingerprints. The most common types of minutiae found in fingerprints are (a)ending, (b)single bifurcation, (h)single lake, (l)island, (m)dot, (g)hook and (o)crossover.

Latent fingerprinting involves locating, preserving, and identifying impressions left by a culprit in the course of committing a crime. In latent fingerprints, the ridge structure is reproduced not in ink on a record card but on an object in sweat, oily secretions, or other substances naturally present on the culprit's fingers. Most latent prints are colourless and must therefore be "developed," or made visible, before they can be preserved and compared. This is done by brushing them with various grey or black powders containing chalk or lampblack combined with other agents. The latent impressions are preserved as evidence either by photography or by lifting powdered prints on the adhesive surfaces of tape.

Though the technique and its systematic use originated in Great Britain, fingerprinting was developed to great usefulness in the United States, where in 1924 two large fingerprint collections were consolidated to form the nucleus of the present file maintained by the Identification Division of the FBI. The division's file contained the fingerprints of more than 90 million persons by the late 20th century. Fingerprint files and search techniques have been computerized to enable much quicker comparison and identification of particular prints.

Minutiae, the discontinuities that interrupt the otherwise smooth flow of ridges, are the basis for most finger-scan authentication. Codified in the late 1800's as Galton features, minutiae are at their most rudimentary ridge endings, the points at which a ridge stops, and bifurcations, the point at which one ridge divides into two. Many types of minutiae exist, including dots (very small ridges), islands (ridges slightly longer than dots, occupying a middle space between two temporarily divergent ridges), ponds or lakes (empty spaces between two temporarily divergent ridges), spurs (a notch protruding from a ridge), bridges (small ridges joining two longer adjacent ridges), and crossovers (two ridges which cross each other) [14].

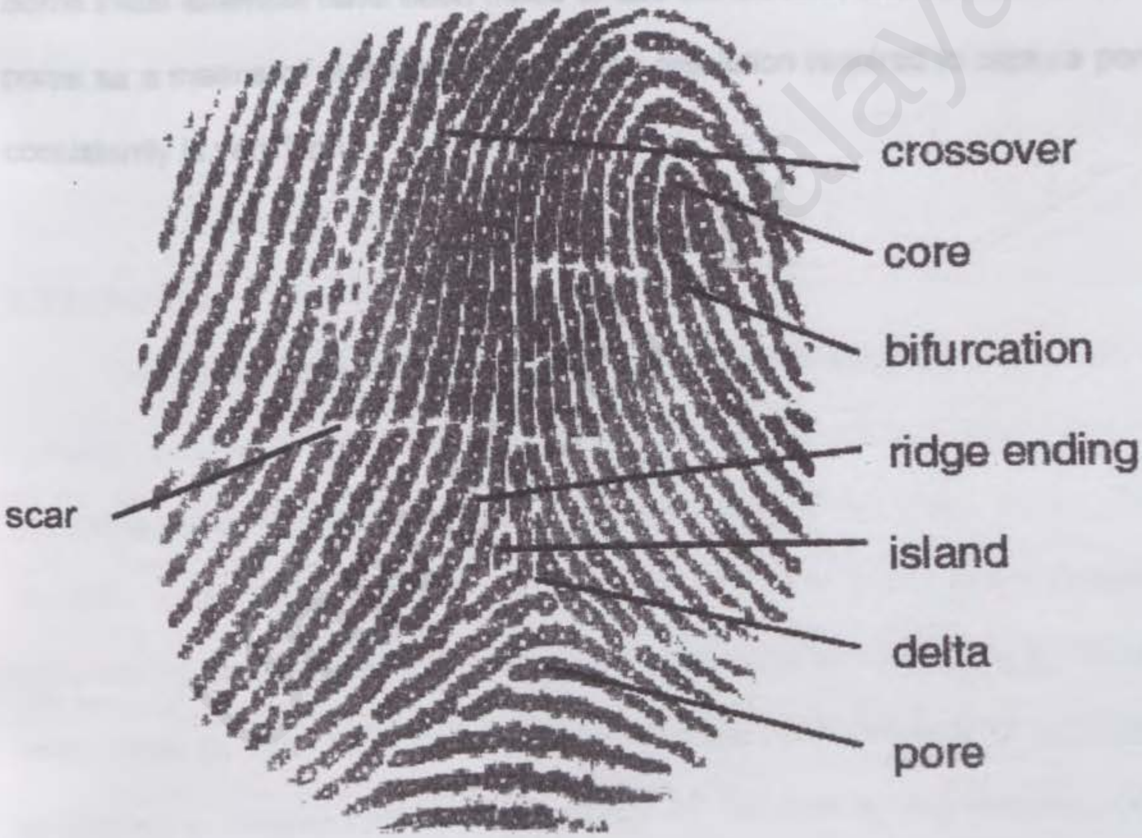


Figure 2.4 : Fingerprint minutiae characteristics (This fingerprint is a right loop.)

Other features are essential to fingerprint authentication. The core is the inner point, normally in the middle of the print, around which swirls, loops, or arches centre. It is frequently characterized by a ridge ending and several acutely curved ridges. Deltas are the points, normally at the lower left and right hand of the fingerprint, around which a triangular series of ridges centre [14].

The ridges are also marked by pores, which appear at steady intervals. Some initial attempts have been made to use the location and distribution of the pores as a means of authentication, but the resolution required to capture pores consistently is very high.

2.3 Studies on Fingerprint Sampling Methods

Getting good images of these distinctive ridges and minutiae is a complicated task. The fingerprint is a small area from which to take measurements, and the wear of daily life affects which ridge patterns show most prominently. Increasingly sophisticated mechanisms have been developed to capture the fingerprint image with sufficient detail and resolution. The technologies in use today are inkpad, optical, silicon, and ultrasound [14].

2.3.1 Inkpad

The inkpad method, is the earliest technique of fingerprinting, involving the optional cleaning of fingers in benzene or ether, drying them, then rolling the balls of each over a glass surface coated with printer's ink. Each finger is then carefully rolled on prepared cards or paper according to an exact technique designed to obtain a light grey impression with clear spaces showing between each ridge so that the ridges may be counted and scanned. If necessary, simultaneous impressions are also taken of all fingers and thumbs. Other "fingerprinting" techniques have also been developed. This method is the oldest technique used, and has many drawbacks to it. Effects of smearing, over inking, under inking, and stretching is highly volatile using this method.

2.3.2 Optical

Optical technology is the oldest and most widely used. The finger is placed on a coated platen, usually built of hard plastic. In most devices, a charged coupled device (CCD) converts the image of the fingerprint, with dark ridges and light valleys, into a digital signal. The brightness is either adjusted automatically (preferable) or manually (difficult), leading to a usable image.

Optical devices have several strengths: they are the most proven over time; they can withstand, to some degree, temperature fluctuations; they are fairly inexpensive; and they can provide resolutions up to 500 dpi. Drawbacks to the technology include size - the platen must be of sufficient size to achieve a quality image - and latent prints. Latent prints are leftover prints from previous users. This can cause image degradation, as severe latent prints can cause two sets of prints to be superimposed. Also, the coating and CCD arrays can wear with age, reducing accuracy. Optical is the most implemented technology by a significant margin. The majority of companies use optical technology, but the trend is toward silicon.

2.3.3 Silicon

Silicon technology has gained considerable acceptance since its introduction in the late 90's. Most silicon, or chip, technology is based on DC capacitance. The silicon sensor acts as one plate of a capacitor, and the finger is the other. The capacitance between platen and the finger is converted into a 8-bit greyscale digital image. All silicon finger-scan vendors use a variation of this type of capacitance except for one, whose technology employs AC capacitance and reads to the live layer of skin.

Silicon generally produces better image quality, with less surface area, than optical. Since the chip is comprised of discreet rows and columns - between 200-300 lines in each direction on a 1cmx1.5cm wafer - it can return exceptionally detailed data. The reduced size of the chip means that costs should drop significantly, now that much of the R&D necessary to develop the technology is bearing fruit. Silicon chips are small enough to be integrated into many devices, which cannot accommodate optical technology.

Silicon's durability, especially in sub-optimal conditions, has yet to be proven. Although manufacturers use coating devices to treat the silicon, and claim that the surface is 100x more durable than optical, this has to be proven. Also, with the reduction in sensor size, it is even more important to ensure that enrolment and verification are done carefully - a poor enrolment may not capture the centre of the fingerprint, and subsequent verifications are subject to the same type of placement.

2.3.4 Ultrasound

Ultrasound technology, though considered perhaps the most accurate of the finger-scan technologies, is not yet widely used. It transmits acoustic waves and measures the distance based on the impedance of the finger, the platen, and air. Ultrasound is capable of penetrating dirt and residue on the platen and the finger, countering a main drawback to optical technology.

Until ultrasound technology gains more widespread usage, it will be difficult to assess its long-term performance. However, preliminary usage of products from Ultra-Scan Corporation (USC) indicates that this is a technology with significant promise. It combines strength of optical technology, large platen size and ease of use, with strength of silicon technology, the ability to overcome sub-optimal reading conditions.

These include the use of a sound spectrograph--a device that depicts graphically such vocal variables as frequency, duration, and intensity--to produce voice graphs, or voiceprints, and the use of a technique known as DNA fingerprinting, an analysis of those regions of DNA that vary among individuals, to identify physical evidence (blood, semen, hair, etc.) as belonging to a suspect. The latter test has been used in paternity testing as well as in forensics.

2.4 Studies on Fingerprint Feature Extraction Algorithms

Approximately 80% of biometric vendors utilize minutiae in some fashion. Those who do not utilize minutia use pattern matching, which extrapolates data from a particular series of ridges. This series of ridges used in enrolment is the basis of comparison, and verification requires that a segment of the same area be found and compared. The use of multiple ridges reduces dependence on minutiae points, which tend to be affected by wear and tear. The templates created in pattern matching are 2-3 times larger than in minutia – usually 900-1200 bytes.

2.4.1 Extracting and Matching Minutiae

This method is the most widely used and implemented. Once a high-quality image is captured, there are a several steps required to convert its distinctive features into a compact template. This process, known as feature extraction, is at the core of fingerprint recognition technology. Each of the primary vendors has a proprietary feature extraction mechanism; the vendors guard these unique algorithms very closely. What follows is a series of steps used, in some fashion, by many vendors - the basic principles apply even to those vendors who use alternative mechanisms.

Once a quality image is captured, it must be converted to a usable format. If the image is greyscale, areas lighter than a particular threshold are discarded (made white), and those darker are made black [15]. The ridges are then thinned from 5-8 pixels in width down to one pixel, for precise location of endings and bifurcations [15]. Minutiae localization begins with this processed image. At this point, even a very precise image will have distortions and false minutiae that need to be filtered out. For example, an algorithm may search the image and eliminate one of two adjacent minutiae, as minutiae are very rarely adjacent. Anomalies caused by scars, sweat, or dirt appear as false minutiae, and algorithms locate any points or patterns that don't make sense, such as a spur on an island (probably false) or a ridge crossing perpendicular to 2-3 others (probably a scar or dirt). A large percentage of would-be minutiae are discarded in this process.

The point at which a ridge ends, and the point where a bifurcation begins, are the most rudimentary minutiae, and are used in most applications. There is variance in how exactly to situate a minutia point: whether to place it directly on the end of the ridge, one pixel away from the ending, or one pixel within the ridge ending (the same applies to bifurcation). Once the point has been situated, its location is commonly indicated by the distance from the core, with the core serving as the 0,0 on an X,Y axis [16]. Some vendors use the far left and bottom boundaries of the image as the axes, correcting for misplacement by locating and adjusting from the core. In addition to the placement of the minutia, the angle of the minutia is normally used. When a ridge ends, its direction at the point of termination establishes the angle (more complicated rules can apply to curved endings). This angle is taken from a horizontal line extending rightward from the core, and can be up to 359°.

2.4.2 Principal Component Analysis (PCA)

PCA is a way to compress a set of high dimensional vectors down to low dimensional vectors without losing much information if the vectors contain correlated components. Matching the compressed vectors between fingerprints is easier than matching full-length vectors (composed of the image pixel values). This method basically match fingerprints based on the pixel values of the image. As was found in the past, this algorithm works poorly because PCA does not seem to retain the topological information about the fingerprint, but rather an overall appearance which changes too much with distortion.

2.4.3 Optical Correlation

Optical correlation in software requires 3 FFT operations on the whole images. Doing the correlation on a normal CPU takes seconds. On the technical side, the algorithms transforms the image from the spatial domain (pixel based) to the frequency domain (cumulative value based). For the frequency domain, all pixels in the input image contribute to each value in the output image for frequency transforms. On the whole, it manipulates the grey-level image and matches prints. invariant to a limited degree of rotation. Using the FFT algorithm, the coefficients are the resultants in the feature extraction.

2.4.4 Baldi-Chauvin Algorithm

This algorithms attempts at using image processing algorithms instead of going into details and writing complex algorithms for finding features. The algorithm works such that a window is slide along the test fingerprint until they match well (using a sum of squared differences, or yet another matching criterion). Following that, feature finding filters are applied to both fingerprints and the sum of squared differences is calculated between the filter outputs for corresponding points on each print. All sums of squared differences are then fed into a neural network which decides whether or not to accept the test fingerprint based on how different the filter outputs are. Both the neural network and the feature finding filters are evolved using an algorithm similar to back propagation.

However, due to the first slow part of the algorithm, the print size is reduced, at which point a lot of information is lost. The matching algorithm used is highly sensitive to distortion and rotation. If the relative positions of features on the two prints do not match exactly, then the outputs from the feature finding filter will not match at all.

2.5 Studies on Fingerprint Classification / Recognition

There are many approaches to be taken into consideration when one is to classify or recognize a fingerprint. As mentioned in the introduction of this paper, conventionally, fingerprint recognition has been conducted via either statistical or syntactical approaches. On one hand, in the statistical approach, a fingerprint is represented by a set of *n*-dimensional feature vector. When the term statistical is used, it essentially means that numerical facts on the fingerprints are systematically collected. This may be by means of division of the image into a set of *n* x *n* vector. The image provided is split into an *n* x *n* vector, and each of the vector features and values are treated independently.

On the other hand, using the syntactical approach, an image pattern is represented as a string, tree or graph. Syntactical methods are used to describe methods relating to syntax. Literally, syntax means the grammatical arrangement of words, or grammatical rules in languages. However, in this context, a syntactical approach would mean the comparison of the value or shape of the string, tree or graph. For example, a frequency graph may be plotted for the image, and when 2 images given, the shape and values of the graph is taken in and compared, and matching is based on the similarities of the graph.

Based on this introduction, the fingerprint recognition is classified into 2. Among the various methods of fingerprint recognition/classification which fall into either of these 2 areas are:- Structural Matching [15], Directional Image Partitioning [17], Gabor Filters with Back Propagation Neural Networks and Patch Matching Algorithms [25] just to name a few. In certain methodologies, Artificial Neural Networks (ANN) is implemented as a way of mimicking the human brain to recognize patterns and images. Fingerprint features are extracted and fed into the neural network. The outputs are then compared to that of the original, and accepted if within the threshold, and rejected otherwise.

2.5.1 Structural Matching

In this study on structural matching for fingerprint recognition [15], fingerprint recognition is done using the minutiae based approach. There are 3 stages in the system, being image pre-processing, feature extraction and lastly fingerprint matching. Fingerprint pre-processing is done to complement the inefficiencies of the fingerprint sampling method and also impurities found in the sample image. The core of the fingerprint is located using contour tracing (points with maximum rate of change of tracing movement). At the feature extraction level, basic features of fingerprints are extracted, being ridge endings and bifurcations. Structural matching is able to correlate features based on their minutiae type, position, orientation (direction) and location relative to other features. This means, for each feature that is detected at each segment of the image, the mentioned features are simultaneously extracted from the image.

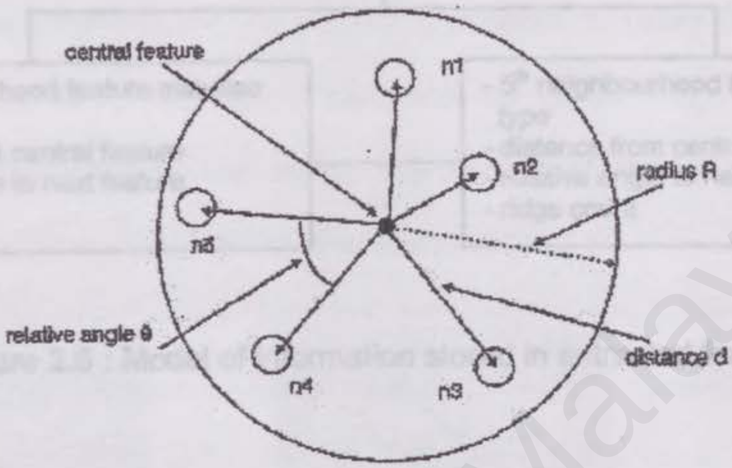


Figure 2.5 : Local features of a structured model.

The structural model of the local features is shown in Figure 2.5. For each extracted feature on the fingerprint, a neighbourhood of radius R is defined. Next, five features within the radius R that are nearest to the central feature are selected for matching. The elements found in the local features which are useful for matching purposes are the minutiae type, X,Y coordinate of the central feature and its minutiae type, distance, relative angle and the ridge count of the neighbourhood features.

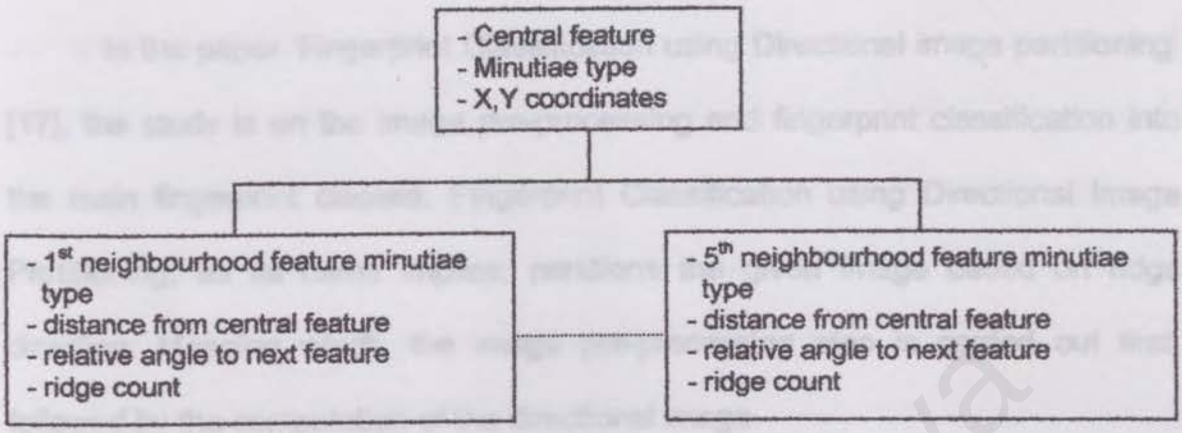


Figure 2.6 : Model of information stored in extracted feature file.

Fingerprints are then matched to determine if they are identical based on the local features that are found in the earlier step. Therefore, using this method, the fingerprint recognition is robust towards the 3 major complications of fingerprint sampling, being rotation, translation and distortion (e.g. stretching). In this system, of the 50 fingerprints tested, the system was able to match all of then even when they are geometrically distorted (stretched, rotated and translated). The false rejection ratio is less than 1% and the false acceptance ratio (FAR) is approximately 0%. However, the database may be too small to deduce the correct ratios. The time taken for a fingerprint to be identified from enrolment to verification is approximately 10 seconds.

2.5.2 Directional Image Partitioning

In the paper 'Fingerprint Classification using Directional image partitioning' [17], the study is on the image pre-processing and fingerprint classification into the main fingerprint classes. Fingerprint Classification using Directional Image Partitioning, as its name implies, partitions the given image based on ridge direction. Meaning which, the image pre-processing step is carried out first, followed by the computation of the directional image.

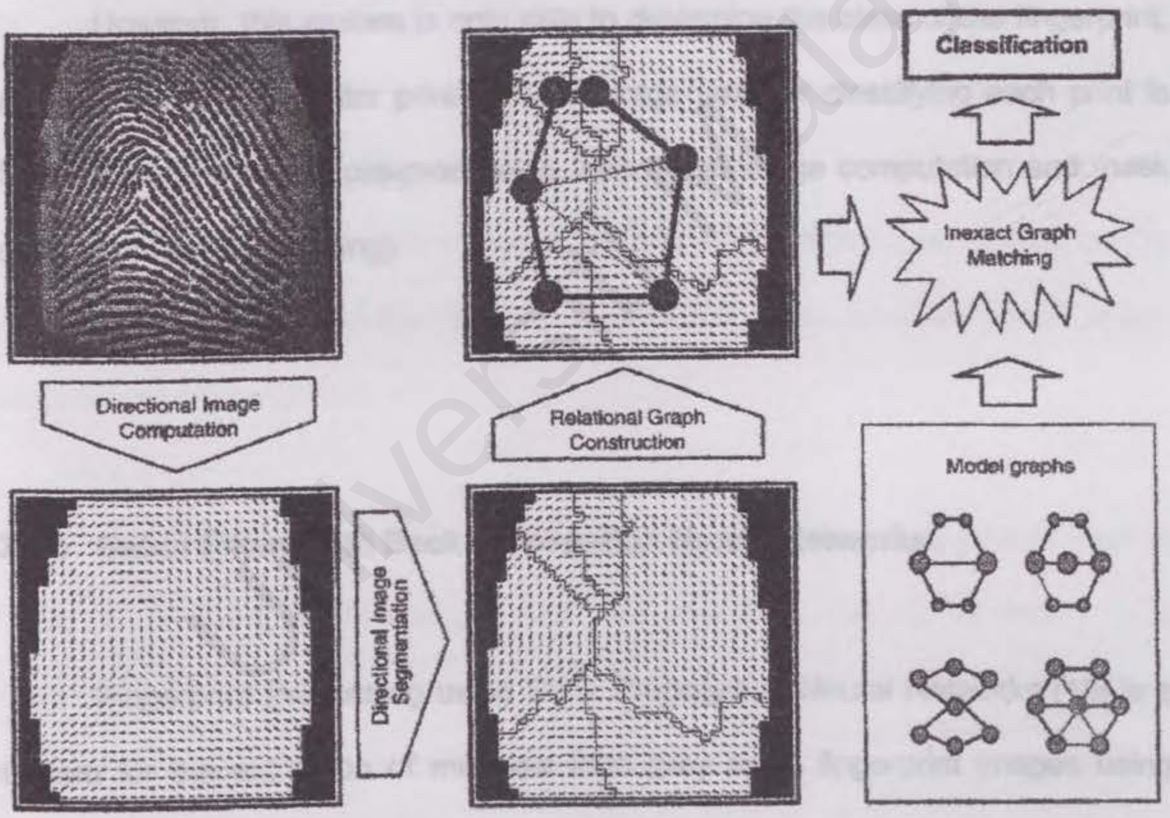


Figure 2.7 : The main sub-steps of the directional image partitioning. The figure shows the intermediate results during the classification of a Tendent arc fingerprint.

A fingerprint sample is first pre-processed, and the result will be a directional image of the fingerprint. Following that, the image is segmented based on its directional ridges. The average value of the 'centre' of the segmented image is the calculated and plotted. Based on the centre points, a relational graph is constructed and inexact graph matching is performed with reference to previously stored database values. If there is a positive match, the fingerprint is classified; in this case a Tendent arch fingerprint.

However, this system is only able to determine the class of the fingerprint, and not match two similar prints. The average time for classifying each print is 3.5 seconds, for image pre-processing, directional image computation and mask application(graph matching).

2.5.3 Gabor Filters with Back Propagation Neural Networks

Fingerprint Processing using Back Propagation Neural Networks [18] is a system for the extraction of minutiae from grey scale fingerprint images using back propagation neural networks and Gabor filters.

An artificial neural network (ANN) is an architecture that mimics the way a human neuron in performs. All inputs into the neural network carry weights and have a part to play in determining the result. For example, in deciding to buy a car, the main concern may be the price, with capacity and transmission coming in second. In this sequence, the colour of the car may be last, carrying much less weight than the transmission. Similarly in neural networks, all data inputs are weighed and represent the output. Each layer may be treated as a neuron in relation to the human body (refer to Chapter 4 for more details).

Just like any other layer in the in the network, the representation layer may be considered a remapping and filtering of the pre-processed data. It could undergo training, or not. In this paper Gabor filters are used in the representation-processing layer. Here, Gabor transform is used to provide the orientation information of features, and also localizing the features. Based on the paper, good detection ratios and low false alarm rates were achieved (exact values were not available).

In another similar approach, the proposed filter-based algorithm uses a bank of Gabor filters to capture both the local and the global details in a fingerprint. The fingerprint matching is based on the Euclidean distance between two features [19]. The Euclidean distance is the straight-line distance between 2 points in a two-dimensional plane. Initial results show identification accuracies comparable to the best results of minutiae-based algorithms. It is shown that matching performance can be improved by combining the decisions of the matchers based on complementary (corresponding) fingerprint information.

2.6 Existing Problems in Fingerprint Recognition

2.6.1 Distortion

The fingerprint sample obtained is tainted with dirt, effects of stretching, excess pressure of finger, scars, incomplete image and other such irregularities.

In this paper, the effects of scars and are ignored and treated as a part of the fingerprint.

2.6.2 Translation

The location of the fingerprint image is 'moved' and no longer has a common reference point. This will cause the image to look different to the system and thus, has to find a reference point for recognition purposes.

(a) training image (b) test image

Figure 2.8 : Fingerprint samples showing translation

2.6.3 Rotation

The image is rotated a certain angle from the original image used. This results in problems in pattern recognition. This problem can be solved when a reference point is found, and also the direction of it.

(a) training image (b) test image

Figure 2.9 : Fingerprint samples showing rotation

Chapter 3

Digital Image Processing

3.1 Introduction to Digital Image Processing

An image is a picture, photograph, display or other form giving a visual representation of an object or scene. In digital image processing, an image is a two dimensional array of numbers.

Images are rectangular arrays. Typical image sizes are 256 x 256, 512 x 512, and 1024 x 1024. The minimum value of a pixel is 0 (represents black) and the maximum number usually 255 (represents white) and the number in between representing different shades of grey. A colour image can be represented by a two-dimensional array of Red, Green and Blue triples. Typically, each number in the triple also ranges from 0 to 255, where 0 indicates that none of that primary colour is present in that pixel and 255 indicates a maximum amount of that primary colour. The computer requires more memory to this data and approximately 3 times the data storage.

There are two ways of storing pixels. One way is to store each pixel as a single bit which means that the computer can only take the values 0 and 1 or just black and white. Another common way is to store each pixel as a byte that is 8 bits (one byte), In this form the maximum pixel value is 255. In byte format only integers are used.

3.1 Introduction to Digital Image Processing

An image is a picture, photograph, display or other form giving a visual representation of an object or scene. In digital image processing, an image is a two dimensional array of numbers.

Images are rectangular arrays. Typical image sizes are 256 x 256, 512 x 512, and 1024 x 1024. The minimum value of a pixel is 0 (represents black) and the maximum number usually 255 (represents white) and the number in between representing different shades of grey. A colour image can be represented by a two-dimensional array of Red, Green and Blue triples. Typically, each number in the triple also ranges from 0 to 255, where 0 indicates that none of that primary colour is present in that pixel and 255 indicates a maximum amount of that primary colour. The computer requires more memory to this data and approximately 3 times the data storage.

There are two ways of storing pixels. One way is to store each pixel as a single bit which means that the computer can only take the values 0 and 1 or just black and white. Another common way is to store each pixel as a byte that is 8 bits (one byte), In this form the maximum pixel value is 255. In byte format only integers are used.

An image is digitised to convert it to a form that can be stored in a computer's memory or in some form of storage media such as a hard disk or CD -ROM. A scanner or other devices such as a digital camera or a digital video recorder can do the digitisation of the images. Once an image is digitised, it can be operated upon by various image-processing techniques.

3.1.1 Digital image definition

A digital image I[m,n] described in a 2D discrete space is derived from an analog image I(x,y) in a 2D continuous space through a sampling process that is frequently referred to as digitisation. The effect of digitisation is shown in Figure 3.1.

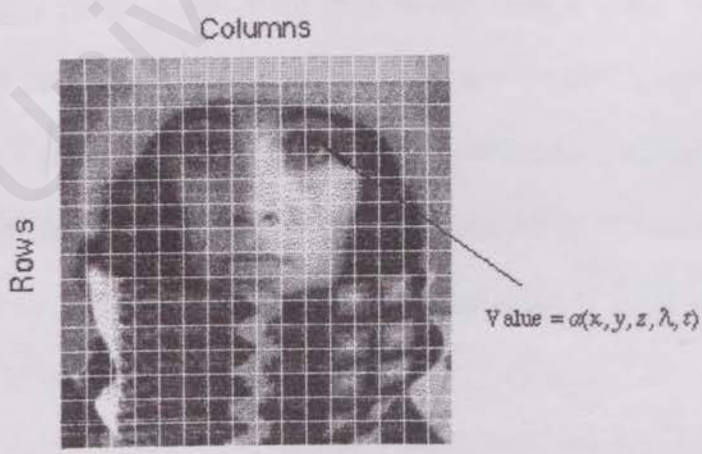


Figure 3.1 : Digitisation of a continuous image.

The pixel at coordinates [m=10, n=3] has the integer brightness value 110. The 2D continuous image a(x,y) is divided into N rows and M columns. The intersection of a row and a column is termed a pixel. The value assigned to the integer coordinates [m,n] with {m=0,1,2,...,M-1} and {n=0,1,2,...,N-1} is a[m,n]. In fact, in most cases I(x,y)--which we might consider to be the physical signal that impinges on the face of a 2D sensor—is actually a function of many variables including depth, colour, and time.

The image shown in Figure 3.1 has been divided into N = 16 rows and M = 16 columns. The value assigned to every pixel is the average brightness in the pixel rounded to the nearest integer value. The process of representing the amplitude of the 2D signal at a given coordinate as an integer value with L different grey levels is usually referred to as amplitude quantization or simply quantization.

3.2 Image Representation

The human visual system receives an input image as a collection of spatially distributed light energy; this form is called an optical image. Optical Images are the types we deal with everyday – cameras and scanners capture them, monitors and pictures display them, and we see them. Optical images are represented as video information in the form of analog electrical signals and have seen how these are sampled to generate digital image f(x,y).

The digital image I(r,c) is represented as a two-dimensional (2D) array of data, where each pixel value corresponds to the brightness of the image at the point (r,c). Two-dimensional array is what we refer to as a matrix, and a one row (or column) a vector. This image model is for monochrome (one colour, this is what we normally refer to as black and white) image data, but other types of image data that requires extensions or modifications are represented in different ways. Similar to data types in programming languages (integer, float, char, etc), these show that different types have different meaning. Similarly for images, the image type indicates that the way an image can appear or is stored. The image types considered are binary, grey-scale, colour images.

3.2.1 Binary Images

In a binary image, each pixel assumes one of only two discrete values. Essentially, these two values correspond to on and off. A binary image is stored as a two-dimensional matrix of '0' (off pixel) and '1' (on pixels). A binary image is referred to as a 1 bit per pixel image, because it only takes 1 binary digit to represent each pixel. A binary image can be considered a special kind of intensity image, containing only black and white. Binary images can also be looked at as an indexed image with only two colours. Binary images are often created from grey-scale images via a threshold operation, where every pixel above the threshold value is turned white ('1'), and below it turned to black ('0').

3.2.2 Grey-Scale Images

Grey-scale images, otherwise known as intensity images, are referred to as monochrome, or one-colour images. Grey-scale images are what we usually refer to as black-and-white pictures. They contain brightness information only, no colour information. The number of bits used for each pixel determines the number of different brightness levels available. The typical image contains 8 bits per pixel data, which allows us to have 256 different brightness (grey) levels. The higher the level, the more white there is in the image, making it lighter. An image that is at level 0 is black, and of level 255 relating to pure white.

3.2.3 Colour Images

Colour images can be modelled as three-band monochrome image data, where each band of data corresponds to a different colour. The actual information stored in the digital image data is the brightness information in each spectral band. In colour transforms, the hue/saturation/lightness (HSL) allows a more detailed description of colours. The lightness is the brightness of the colour, and the hue is what we normally think of as 'colour' (for example green, blue or orange). The saturation is a measure of how much white is in the colour (for example, pink is actually red with more white).

Some coloured images are stored as indexed images, which consists of two arrays, an image matrix and a colourmap. The colourmap is an ordered set of values that represent the colours in the image. For each image pixel, the image matrix contains a value that is an index into the colourmap. The colourmap is an m-by-3 matrix of class double. Each row of the colourmap matrix specifies the red, green, and blue (RGB) values for a single colour:

Row n Colour = [R G B]

Where n is the row number and R, G, and B are real scalars that range from 0 (black) to 1.0 (full intensity). The pixels in the image are represented by integers, which are pointer (indices) to colour values stored in the colourmap. For example, if the value of the pointer is 5 that means the value of the colour is in row 5 of the colourmap.

Typical colour images are represented directly as red, green and blue, or RGB images, where we can refer to a single pixel's red, green and blue values as a colour pixel vector – [R,G,B]. Like an indexed image, it represents each pixel colour as a set of three values, representing the red, green and blue intensities that make up the colour. Unlike the indexed image, however, these intensity values are stored directly in the image array, not indirectly in a colourmap. The red, green and blue components of an RGB image reside in a single m-by-n-3 array. m and n are the numbers of rows and columns of pixels in the image, and the third dimension consists of three planes, containing red, green and blue intensity values. For each pixel in the image, the red, green and blue elements combine to create the pixel's actual colour.

3.3 Objective of Digital Image Processing

3.3.1 Main Objectives for Digital Image Processing

1. To enhance the quality of images so that they can be viewed clearly without any strain on the eyes. Since important images are extracted from images, it is essential that the images produced are clear and therefore digital image processing is utilized to produces the best images. The main aim of digital image processing is to enhance the pictorial information in the images for human interpretation.
2. To process acquired data for autonomous machine perception. That is procedures are used for extracting information from an image, information in a form suitable for computer processing. Examples of this type of information that is used in machine perception are statistical moments, Fourier coefficients, and multi-dimensional distance problems. Common tasks in machine perception that often utilize image processing techniques are automatic character recognition, industrial robots for product assembly and inspection, screening of x-rays and blood samples and automatic processing of fingerprints.

Here are a few examples on the uses of digital image processing :

- To remove a blur from an image
- Smooth out the graininess, speckle, or noise in an image
- Improve the contrast or other visual properties of an image prior to displaying it
- Segment an image into regions such as object and background
- Magnify, minify, or rotate an image
- Remove warps or distortions from an image
- Code the image in some efficient way for storage or transmission

3.3.2 Advantages Digital Image Processing

1. *Precision.* In each generation of photographic process, there is a loss of image quality and electrical signals are degraded by the physical limitations of the electrical components, whereas digital image processing can maintain exact precision.
2. *Flexibility.* Using an enlarger, an image may be magnified, reduced or rotated. The contract, brightness and other parameters of the picture can be altered.

3.3.3 Disadvantages of Digital Image Processing

1. *Speed and Expense.* Many of the operations that is utilized by digital image processing are slower and more expensive than the corresponding optical or electrical operations and the computing resources may be expensive. These advantages have been reduced considerably through computer technology.

3.4 Noise in Image Operations

3.4.1 Types of Noise

Noise is any undesired information that contaminates an image. Noise appears in images from a variety of sources. The digital image acquisition process, which converts an optical image into a continuous electrical signal that is then sampled, is the primary process by which noise appears in digital images. Normally, the noise comes from :

1. If the image is scanned from a photograph made on a film, the firm grain is the source of the noise. Noise can also be the result of damage of the firm, or to be introduced by the scanner itself.
2. If the image is acquired directly in the digital format, the mechanism for gathering the data can introduce noise.
3. Electric transmission of image data can introduce noise.

At every step in the process there are fluctuations caused by natural phenomena that add a random value to the exact brightness value for a given pixel. In typical images the noise can be modelled with either a Gaussian ("normal"), uniform, or salt-and-pepper ("impulse") distribution. The shape of the distribution of these noise types as a function of grey level can be modelled as a histogram and can be seen in Figure 3.6. In Figure 3.6-a we see the bell-shaped curve of the Gaussian noise distribution, which can be analytically described by

$$\mathrm{HISTOGRAM}_{\mathrm{Gaussian}} = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(g-m)^2/2\sigma^2}$$

..... **(equation 3.1)**

where g = grey level

m = mean (average)

o = standard deviation (a^2 = variance)

Theoretically, this equation defines values from -∞ to +∞, but because the actual grey levels are only defined over a finite range, the number of pixels at the lower and upper values will be higher than this equation predicts. This is a result of the fact that all the noise values below the minimum will be clipped to the minimum, and those above the maximum will be clipped at the maximum value. This is a factor that must be considered with all theoretical noise models, when applied to a fixed, discrete range such as with digital images (e.g., 0 to 255). In Figure 3.6-a is the uniform distribution :

$$\text{HISTOGRAM}_{\text{Uniform}} = \begin{cases} \dfrac{1}{b-a} & \text{for } a \le g \le b \\ 0 & \text{elsewhere} \end{cases}$$

$$\text{mean} = \frac{a+b}{2}$$

$$\text{variance} = \frac{(b-a)^2}{12}$$

..... **(equation 3.2)**

With the uniform distribution, the grey-level values of the noise are evenly distributed across a specific range, which may be the entire range (0 to 255 for 8-bits), or a smaller portion of the entire range. In Figure 3.2-b is the salt-and-pepper distribution:

$$\text{HISTOGRAM}_{\text{Salt \& Pepper}} = \begin{cases} A & \text{for } g = a \text{ ("pepper")} \\ B & \text{for } g = b \text{ ("salt")} \end{cases}$$

..... **(equation 3.3)**

In the salt-and-pepper noise model there are only two possible values, a and b, and the probability of each is typically less than 0.1 with numbers greater than this, the noise will dominate the image. For an 8-bit image, the typical value for pepper noise is 0 and for salt-noise, 255.

Figure 3.2 shows the example figure of noise:

(a) Uniform Noise

(b) Salt and pepper noise

Figure 3.2 : Noise effects in images

3.5 Image Enhancement

3.5.1 Spatial Domain Filters

The spatial domain is refers to the space of the image itself. Approach to this method is based on direct manipulation of pixels in an image. Spatial filters can be effectively used to remove various types of noise in digital images. These spatial filters typically operate on small neighbourhoods, 3 x 3 to 11 x 11, and some can be implemented as convolution masks.

3.5.1.1 Mean Filter

The mean filter function by finding some form of an average within the N x N image. The most basic of these filters is the arithmetic mean filter, which finds the arithmetic average of the pixel values in the matrix [27]. For example, consider a 3 x 3 matrix with values 3,7,15,9,11,16,5,13,2 as shown in Figure 3.3. The average sum for this matrix is the total number of pixel values divided by the number of pixels. In this case, the pixel with original value 11 will be replaced by value 9.

3	7	15
9	11	16
5	13	2

Figure 3.3 : 3 x 3 matrix

In mean filtering, the grey-level of each pixel is replaced by the mean of the grey levels in the neighbourhood of that pixel. Mean filters work best with Gaussian or uniform noise

3.5.1.2 Median Filter

The median *m* of a set of values is such that half the values in the set are less than *m* and half are greater than *m* [27]. In order to perform median filtering in a neighbourhood of a pixel, the values in its neighbourhood are first rearranged in ascending order, and the median determined. This value is the assigned to the pixel. For example, consider matrix M in Figure 3.3 with set of numbers, 3,7,15,9,11,16,5,13,2. These numbers, rearranged in an ascending order will result in 2,3,5,7,8,9,11,13,15 and 16, of which the median value is 8.

In median filtering, the grey-level of each pixel is replaced by the median of the grey levels in a neighbourhood of that pixel, instead of the average. Median filters are used to achieve noise reduction rather than blurring. Median filters work best with salt-and-pepper noise. The logic for median filtering is as follows:

For each pixel in the original image (except border pixels)

i. Assign a 3 x 3 neighbourhood

ii. Rearrange the pixel value in the neighbourhood in an ascending order.

iii. Replace centre element by median value, which is the middle (fifth) element after rearranging the output images corresponding pixels.

3.5.1.3 Gaussian Filter

A Gaussian filter is a smoothing filter (low pass filter), which reduces noise in an image by lowering the proportion of its high frequency components. It allows low frequencies to pass unchanged but attenuate high frequencies. In Gaussian filtering a smoothing convolution mask is used in a two-dimensional Gaussian curve. This weights some neighbouring pixels more than others.

Figure 3.4 shows a mask of the 3 x 3 Gaussian filter. The Gaussian filtering algorithm is as follows:

For each pixel in the image (except border pixels)

i. Assign a 3 x 3 neighbourhood

ii. Perform convolution following Equation 3.1. Divide the values calculated by the sum of the filter used.

$$O_{i,j} = \sum_{r=0}^{I+1} \sum_{c=0}^{N-1} I_{k,l}\, h_{i-k,\, j-l} \quad \text{..... (equation 3.4)}$$

iii. Store result in the corresponding pixel in the output image.

After the whole image has been processed, the pixel value of the output image is restored into the original image

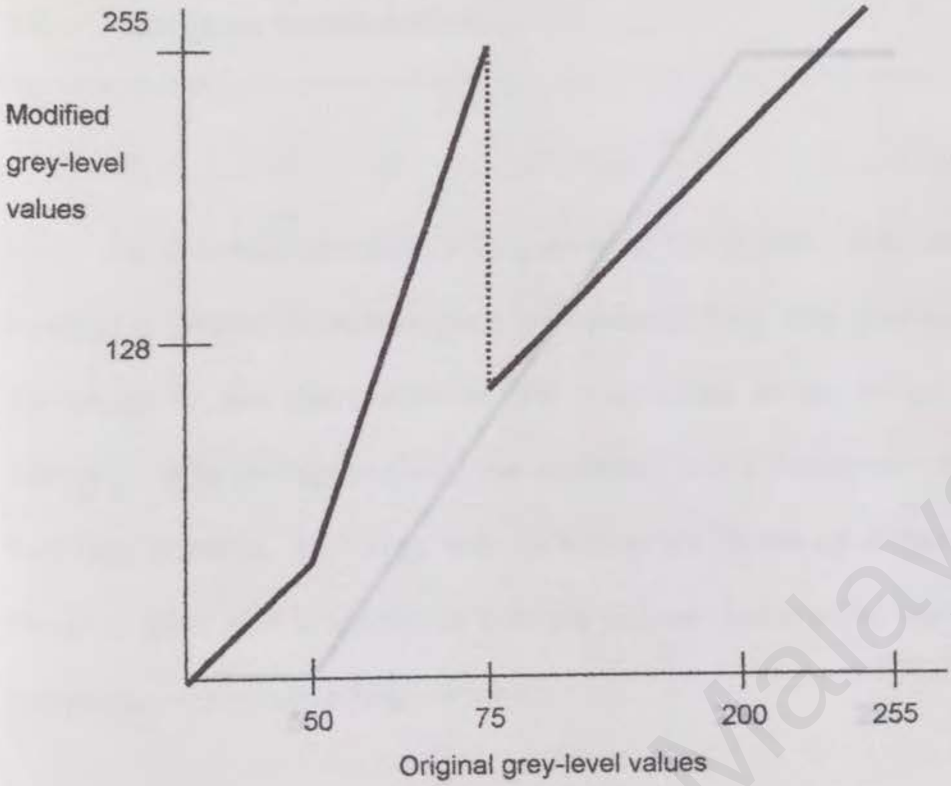
1	2	1
2	4	2
1	2	1

Figure 3.4: Gaussian filter mask

3.5.2 Grey-Scale Modification

Grey-scale modification (also called grey-level scaling) methods belong int the category of point operations and function by changing the pixel's (grey-level) values by a mapping equation. The mapping equation is typically linear (nonlinear equations can be modelled by piecewise linear models) and maps the original grey-level values to other specified values [26]. Typical applications include contrast enhancement and feature enhancement.

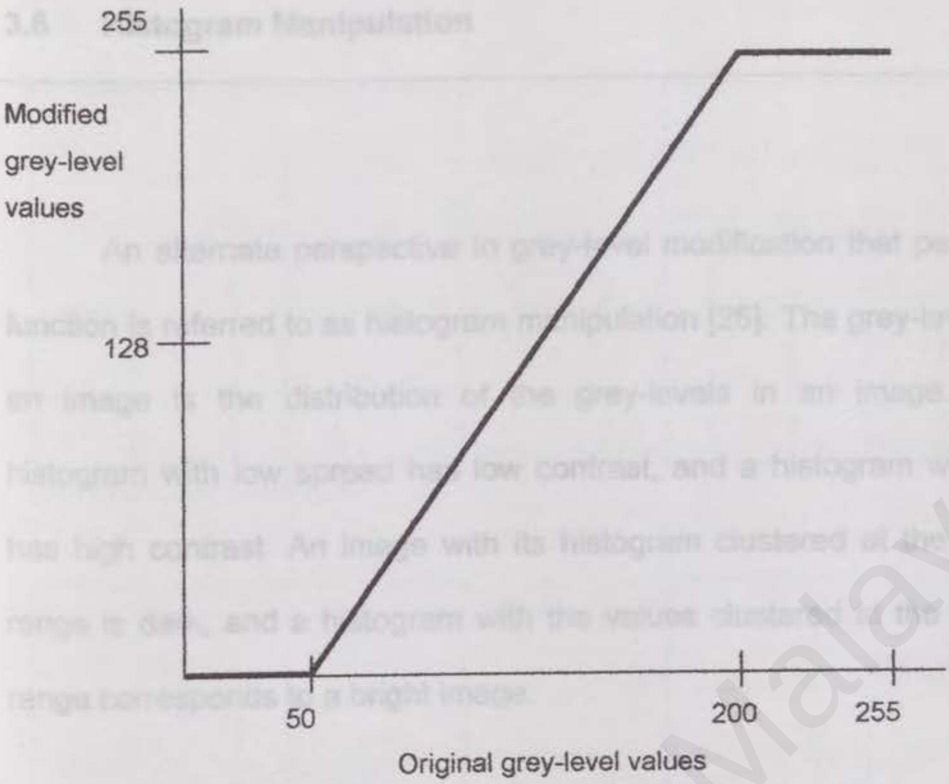
The primary operations applied to the grey-scale of an image are to compress or stretch it. We typically compress grey-level ranges that are of little interest to us and stretch the grey-level ranges where we desire more information. This is illustrated in Figure 3.5, where the original image data are shown on the horizontal axis and the modified values are shown on the vertical axis. The linear equations corresponding to modified values are shown on the vertical axis. The linear equations corresponding to the lines shown on the graph represent the mapping equations. If the slope of the line is between zero and one, this is called *grey-level compression*, whereas if the slope is greater than one, it is called *grey-level stretching*. In Figure 3.5-a, the range of grey-level values from 28 to 75 is stretched, while the other grey-level values are left alone. The original and modified images are shown in Figure 3.5-b and 3.5-c, where we can see that stretching this range exposed previously hidden visual information.



(a) Grey-level stretching

(b) Original image

(c) Image after modification



(d) Grey-level stretching with clipping at ends

Figure 3.5 : Grey-scale modification

In some cases we may want to stretch a specific range of grey-levels, while clipping the values at the low and high ends. Figure 3.5-d illustrates a linear function to stretch the grey levels between 50 and 200, while clipping any values below 50 to zero (black), and any value above 200 to 255 (white). Another type of mapping equation used for feature extraction is called intensity-level slicing, where specific grey-level values of interest and mapping them to specified (typically bright) values. This is similar to binarization using thresholding. The only difference is that a specified range is specified (e.g. 150 to 200), and mapped to white, and the background either left in its original grey-level values, or mapped to zero (black).

3.6 Histogram Manipulation

An alternate perspective to grey-level modification that performs a similar function is referred to as histogram manipulation [26]. The grey-level histogram of an image is the distribution of the grey-levels in an image. In general, a histogram with low spread has low contrast, and a histogram with wide spread has high contrast. An image with its histogram clustered at the low end of the range is dark, and a histogram with the values clustered at the high end of the range corresponds to a bright image.

3.6.1 Histogram Modification

Histograms can also be modified by a mapping function, which will stretch, shrink (compress), or slide the histogram. Histogram stretching and histogram shrinking are forms of grey-scale modification, sometimes referred to as histogram scaling. Figure 3.6 shows images of histograms stretching, shrinking and sliding.

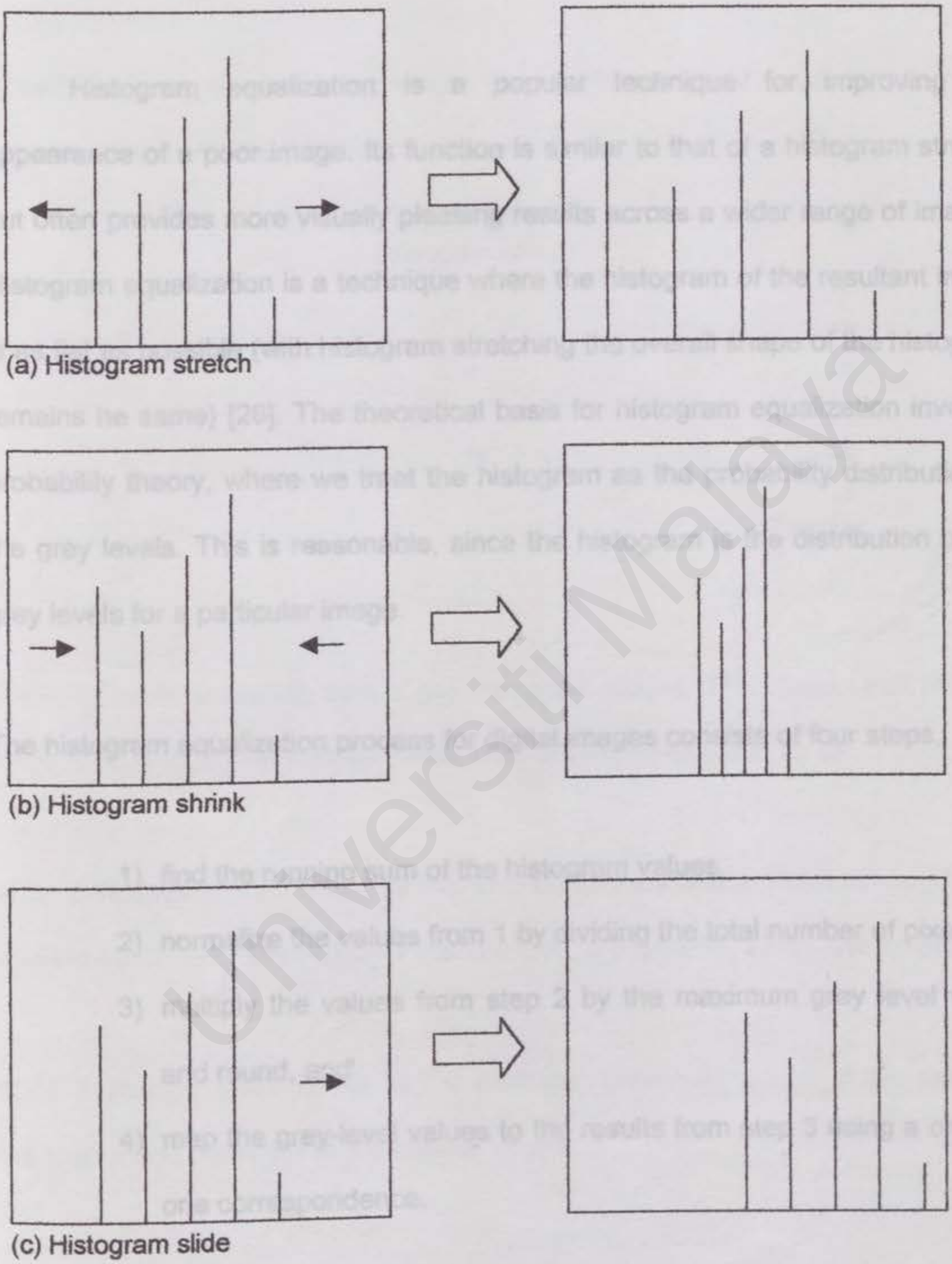


Figure 3.6 : Histogram modification

3.6.2 Histogram Equalization

Histogram equalization is a popular technique for improving the appearance of a poor image. Its function is similar to that of a histogram stretch, but often provides more visually pleasing results across a wider range of images. Histogram equalization is a technique where the histogram of the resultant image is as flat as possible (with histogram stretching the overall shape of the histogram remains he same) [26]. The theoretical basis for histogram equalization involves probability theory, where we treat the histogram as the probability distribution of the grey levels. This is reasonable, since the histogram is the distribution of the grey levels for a particular image.

The histogram equalization process for digital images consists of four steps,

1) find the running sum of the histogram values.
2) normalize the values from 1 by dividing the total number of pixels.
3) multiply the values from step 2 by the maximum grey level value and round, and
4) map the grey-level values to the results from step 3 using a one-to-one correspondence.

The following examples will help to clarify the process:

Consider an image with 3 bits/pixel, so the possible range of values is 0 to 7. We have an image with the following histogram:

Grey-Level Value	*Number of Pixels (Histogram values)*	*Running Sum*
0	10	10
1	8	18
2	9	27
3	2	29
4	14	43
5	1	44
6	5	49
7	2	51

STEP 1: Create a running sum of the histogram values. This means that the first value is 10, second value is 10+8=18, next is 10+9+9=27, and so on.

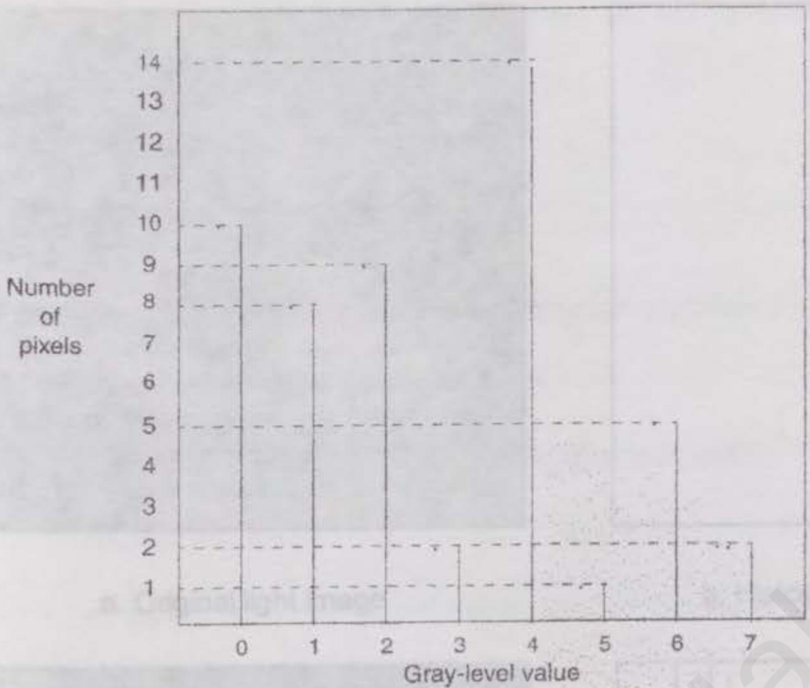
STEP 2: Normalize by dividing by the total number of pixels. The total number of pixels is 10+8+9+2+14+1+5+2=51. So we get:10/51, 18/51, 27/51 and so on.

STEP 3: Multiply these values by the maximum grey-level values, in this case, 7, and then round the result to the closest integer.

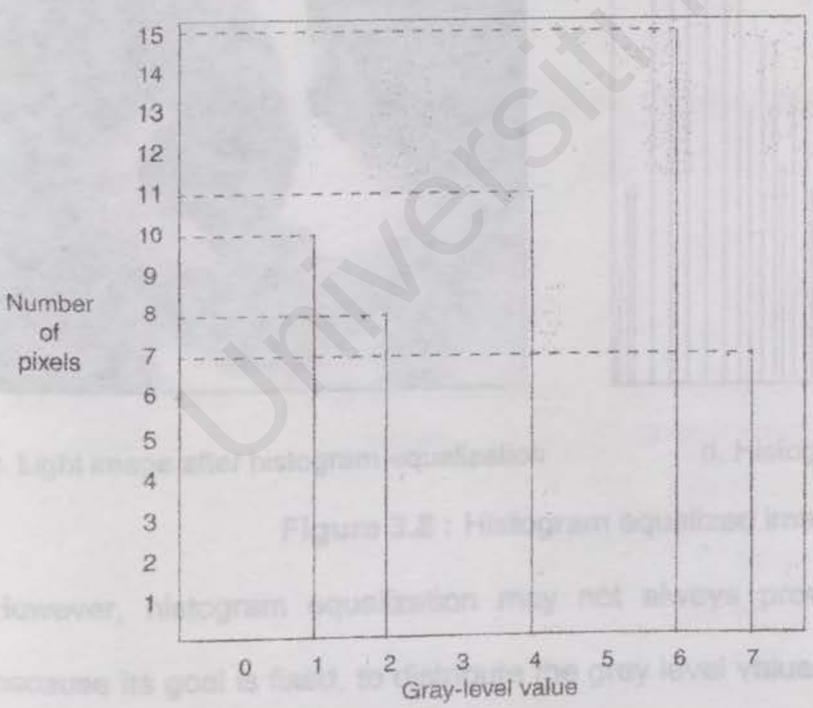
STEP 4: Map the original values to the results from step 3 by a one-to-one correspondence. This is done as follows:

Grey-Level Value	*Normalized Values*	*Histogram Equalized Values*	*Number of Pixels*	
0	10/51 * 7 = 1.37	1	10	
1	18/51 * 7 = 2.47	2	8	
2	27/51 * 7 = 3.71	4	9	11
3	29/51 * 7 = 3.98	4	2	
4	43/51 * 7 = 5.90	6	14	15
5	44/51 * 7 = 6.04	6	1	
6	49/51 * 7 = 6.72	7	5	
7	51/51 * 7 = 7.00	7	2	

All pixels in the original image with grey level 0 are set to 1, values of 1 set to 2, 2 set to 4, 3 set to 4, and so on. In Figure 4.7 we see the original histogram and the resulting histogram-equalization histogram. As seen, there are less grey-level values in the histogram-equalized histogram because of the normalized values, resulting in similar histogram-equalized values, thus a flatter result.

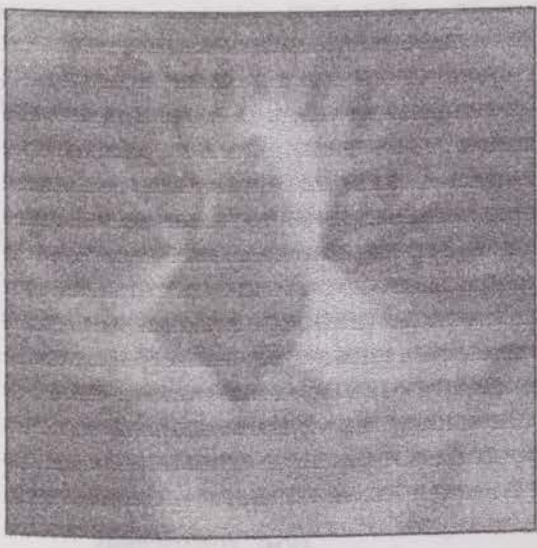


(a) Original Histogram

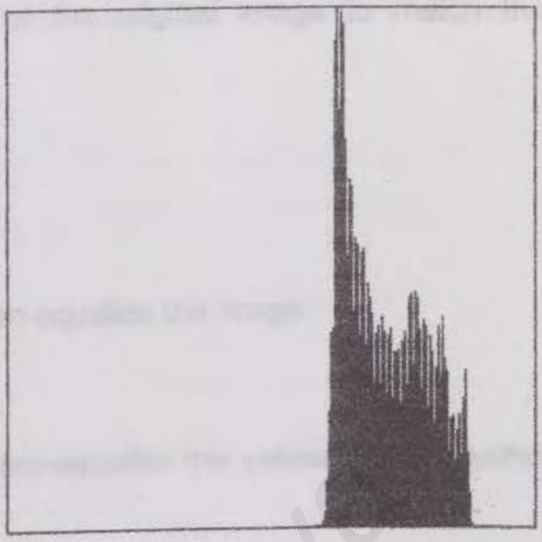


(b) After histogram equalization

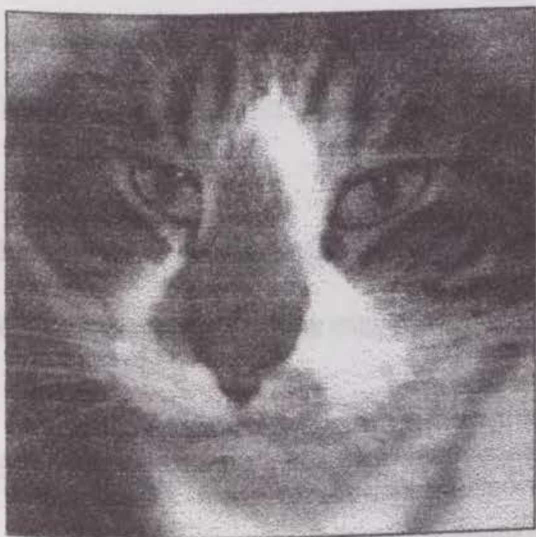
Figure 3.7 : Histogram equalization



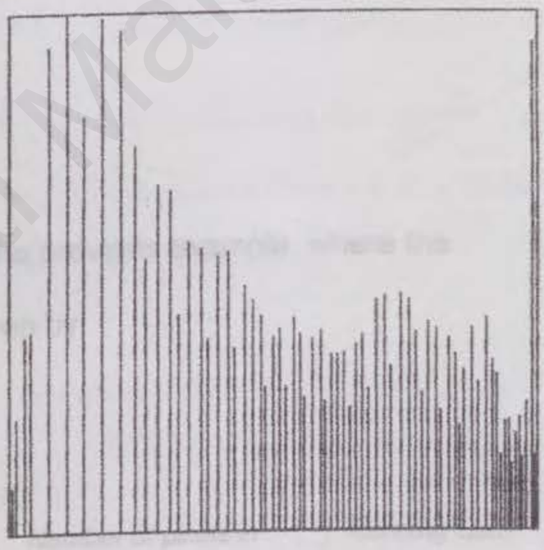
a. Original light image



b. Histogram of image (a)



c. Light image after histogram equalization



d. Histogram of image (c)

Figure 3.8 : Histogram equalized images

However, histogram equalization may not always provide the desired effect because its goal is fixed, to distribute the grey level values as evenly as possible. To allow for interactive histogram manipulation, the ability to specify the histogram is necessary. Histogram specification is the process of defining a histogram and modifying the histogram of the original image to match the histogram as specified.

This process can be implemented by:

1. finding the mapping table to histogram-equalize the image.
2. specifying the desired histogram
3. finding the mapping table to histogram-equalize the values of the desired histogram, and
4. mapping the original values to the values from step 3, by using the table from step 1.

This process is best illustrated by example.

STEP 1: For this we will use the data from the previous example, where the histogram-equalization mapping table is given by:

STEP 2: Specify the desired histogram:

Original Grey-Level Value – 0	*Histogram Equalized Values – H*	*Number of pixels in Desired Histogram*	*Running Sum (desired histogram)*
0	1	1	1
1	2	5	6
2	4	10	16
3	4	15	31
4	6	20	51
5	6	0	51
6	7	0	51
7	7	0	51

STEP 3: Find the histogram-equalization mapping table for the desired histogram:

Original Grey-Level Value – 0	*Histogram equalized values – S (rounded to closest integer)*
0	1/51 * 7 = 0
1	6/51 * 7 = 1
2	16/51 * 7 = 2
3	31/51 * 7 = 4
4	51/51 * 7 = 7
5	51/51 * 7 = 7
6	51/51 * 7 = 7
7	51/51 * 7 = 7

STEP 4: Map the original values to values from step 3 using the table from step 1. This is done by setting up a table created by combining the tables from steps 1 and 3. Obtain the final mapping by mapping the value in H to the closest value in S.

Original Grey-Level Value – 0	*Histogram Equalized Values – H*	*Histogram equalized Values – S*	*Final Mapping Value - M*
0	1	0	1
1	2	1	2
2	4	2	3
3	4	4	3
4	6	7	4
5	6	7	4
6	7	7	4
7	7	7	4

The M column is obtained by mapping the value in H to the closest value in S and then using the corresponding row in 0 for the entry in M. For example, the first entry in H is 1, and the closest value in S is also 1. The value 1 in S appears in row 1, so the entry for M is written as 1. The third entry in H is 4, and the closest value in S is 4. This 4 appears in row 3, so we write 3 for that entry in M. If we consider the fifth entry in H, we see that 6 must map to 7(closest value), but 7 appears on rows 4, 5 ,6 ,7. Which one is the appropriate to select? It depends on what we want; picking the largest value will provide maximum contrast, but picking the smallest will produce a smoother (gradually changing) image. Typically, the smallest is chosen because a histogram stretch or equalization on the output image if maximum contrast is needed.

3.7 Image Segmentation

Segmentation means division the image into regions corresponding to objects, which have the same characteristic. The goal of image segmentation is to find regions that represent meaningful parts of objects. The two most common use of segmentation technique is thresholding and edge detection.

3.7.1 Thresholding

Image thresholding classifies the pixels of an image into the foreground and the background. In the case of grey-scale images, the pixels initially have a value from 0-255, and the general idea of thresholding is to convert pixels above a certain level of grey into foreground and to convert pixels below the level into background. For example, we can use a linear function to threshold the image into a binary image which is only contain black and white colour where black is foreground, white is background or vice versa. From this technique, we can separate or extract out the useful foreground image.

There is no universally applicable segmentation technique that will work for all images and no segmentation technique is perfect. Lets say we have an input vector A, which is performed on by the threshold function, resulting in B, a binary image (based on the threshold function).

$$\underset{A}{\begin{pmatrix} 34 & 55 & 6 \\ 75 & 23 & 4 \\ 56 & 99 & 98 \end{pmatrix}} \xrightarrow{\text{Thresholding}} \underset{B}{\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}}$$

This is known as binarization, using a threshold function as stated :

X=0 if y < 50

X=1 if y >= 50

3.7.2 Edge Detection

Finding the boundaries between objects performs edge detection. Marking points that may be part of an edge usually begins this method. Edge points represent local discontinuities in specific features, such as brightness or texture.

Most of the edge detection operators are implemented with convolution masks, and most are based on discrete approximations to the rate of change in a function (e.g. image brightness function). A large change in brightness over a short spatial distance indicates the presence of an edge. Some edge detection operators return information about the direction of the edge, whereas some only the existence of an edge at each point [26].

Edge detection operators are fairly sensitive to the effects of noise. Therefore, pre-processing steps are taken to remove or minimize noise. On the whole, the parameters set will determine the accuracy of the edge detector. If it is made sensitive, it may attribute noise effects to possible edges, whereas if less sensitive, it may miss valid edges. These operators are based on the idea that edge information in an image is found by looking at a point's neighbouring pixels. If a pixel's grey level is similar to those around it, chances are it is probably not an edge. However, if a pixel has neighbours with widely varying grey levels, it may represent an edge point. In other words, ad edge is defined by a discontinuity in grey-level values.

3.8 Characteristics of Image Operations

There is a variety of ways to classify and characterize image operations. The reason for doing so is to understand what type of results we might expect to achieve with a given type of operation or what might be the computational burden associated with a given operation.

The types of operations that can be applied to digital images to transform an input image a[m,n] into an output image b[m,n] (or another representation) can be classified into three categories as shown in Table 3.1.

Operation	Characterization	Generic Complexity/Pixel
* Point	the output value at a specific coordinate is dependent only on the input value at that same coordinate.	Constant
* Local	the output value at a specific coordinate is dependent on the input values in the neighbourhood of that same coordinate.	P^2
* Global	the output value at a specific coordinate is dependent on all the values in the input image.	N^2

(Image size = N x N; neighbourhood size = P x P)

Table 3.1 : Types of image operations.

Note that the complexity is specified in operations per pixel.

This is shown graphically in Figure 3.9

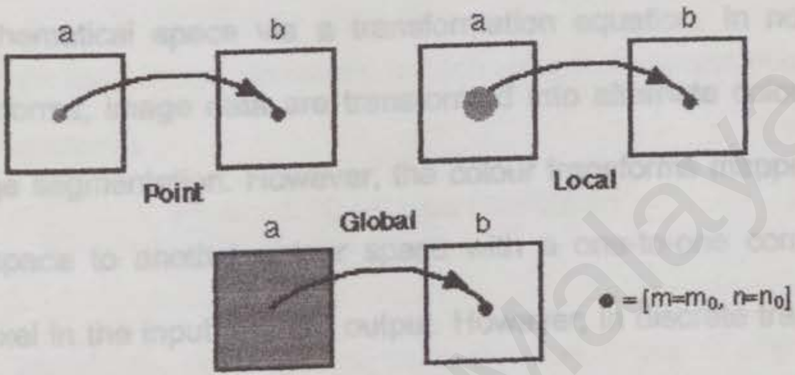
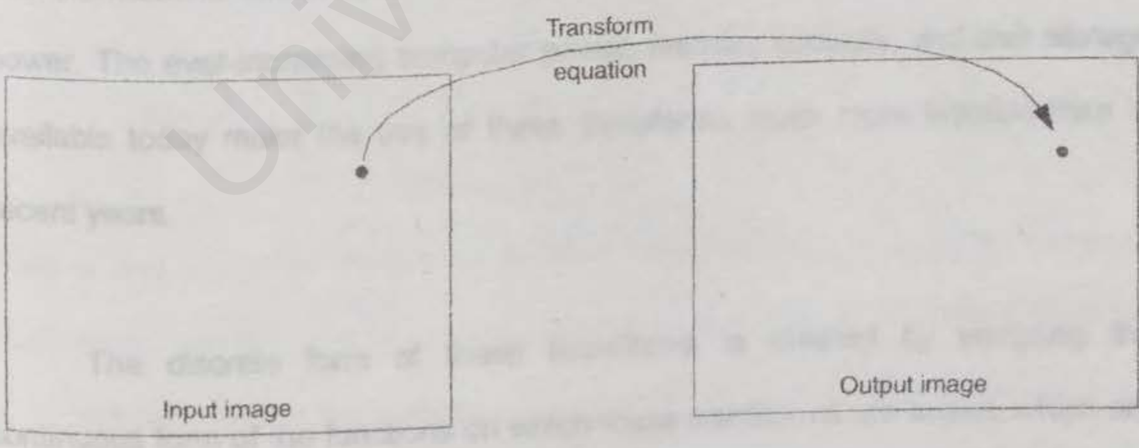


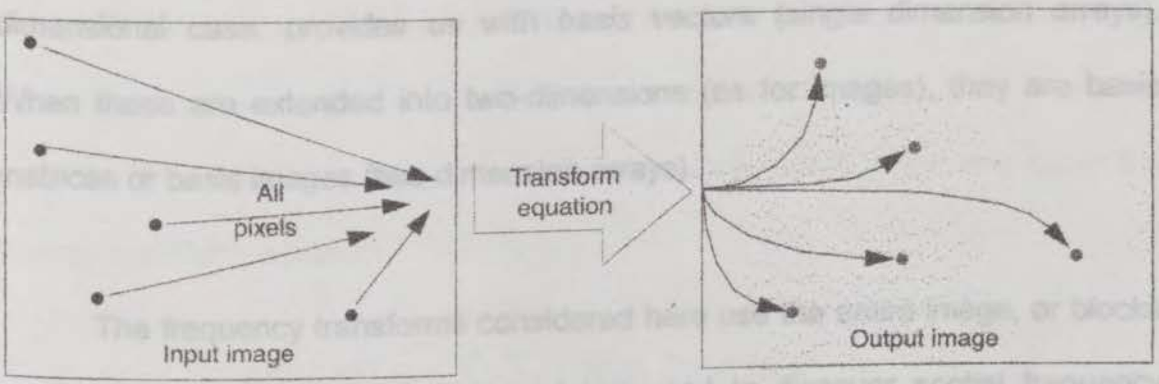
Figure 3.9 : Illustration of various types of image operations

3.9 Discrete Transforms

The transforms considered here provide information regarding the spatial frequency content of an image. In general, a transform maps image data into a different mathematical space via a transformation equation. In normal spatial domain transforms, image data are transformed into alternate colour spaces to achieve image segmentation. However, the colour transforms mapped data from one colour space to another colour space with a one-to-one correspondence between a pixel in the input and the output. However, in discrete transforms, the image data is mapped from the spatial domain to the frequency domain (also called the spectral domain), where all the pixels in the input (spatial domain) contribute to each value in the output (frequency domain) [26].



(a) Colour transforms use a single-pixel to single-pixel mapping



(b) All pixels in the input image correspond to each value in the output image for frequency transforms

Figure 3.10 : Image transforms

These transforms are used as tools in many areas of engineering and science, including computer imaging. Originally defined in their continuous forms, they are commonly used today in their discrete (sampled) forms. The large number of arithmetic operations required for the discrete transforms, combine with the massive amounts of data in an image, requires a great deal of computer power. The ever-increasing computer power, memory capacity, and disk storage available today make the use of these transforms much more feasible than in recent years.

The discrete form of these transforms is created by sampling the continuous form of the functions on which these transforms are based, which are the basis functions. Two-dimensional arrays are what we refer to as a matrix, and a single row (or column) a vector. The functions used for these transforms are typically sinusoidal or rectangular, and the sampling process, for the one-dimensional case, provides us with *basis vectors* (single dimension arrays). When these are extended into two-dimensions (as for images), they are basis matrices or basis images (two-dimension arrays).

The frequency transforms considered here use the entire image, or blocks that are typically at least 8 x 8, and are used to discover spatial frequency information. The ways in which the image brightness levels change, in space defines the spatial frequency. For example, rapidly changing brightness corresponds to high spatial frequency, whereas slowly changing brightness levels relate to low frequency information. The lowest spatial frequency, called the zero frequency term corresponds to an image with a constant value.

The general form is the transformation equation, assuming an N x N image, is given by:

$$T(u,v) = \sum_{r=0}^{N-1} \sum_{c=0}^{N-1} I(r,c)\, B(r,c;\, u,v) \qquad \text{..... (equation 3.5)}$$

Here u and v are the frequency domain variables, T(u, v) are the transform coefficients, and B(r, c; u, v) correspond to the basis images. The notation B(r, c; u, v) defines a set of basis images, corresponding to each different value for u and v, and the size of each is r by c (Figure 2.5-4). The transform coefficients T(u, v) are the projections of I(r, c) onto each B(u, u). These coefficients tell us how similar the image is to the basis image; the more alike they are, the bigger the coefficient. This transformation princess amounts to decomposing the image into a weighted sum of the basis images, where the coefficients T(u, v) are the weights. To obtain the image from the transform coefficients, we apply the inverse transform equation:

$$I(r,c) = T^{-1}[T(u,v)] = \sum_{r=0}^{N-1}\sum_{c=0}^{N-1} '(r,c)\, B(r,c;u,v)$$

..... **(equation 3.6)**

Here the T^{-1} [T(u, v)] represents the inverse transform, and the B^{-1} (r, c; u, v) represents the inverse basis images. In many cases the inverse basis images are the same as the forward ones, but possibly weighted by a constant.

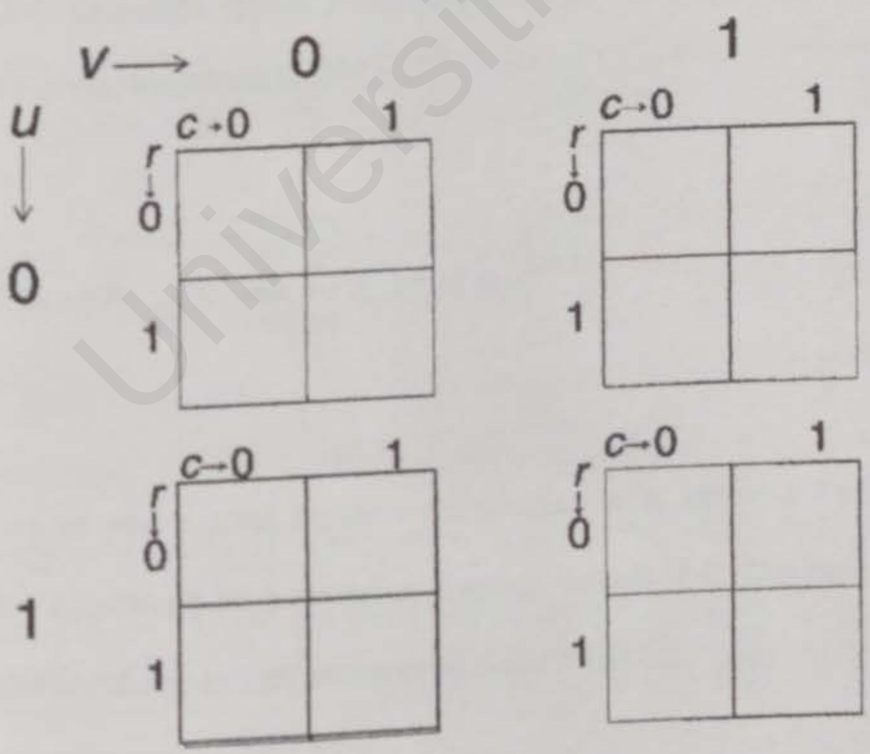


Figure 3.11 : Size of generic basis vectors for a 2 x 2 transform

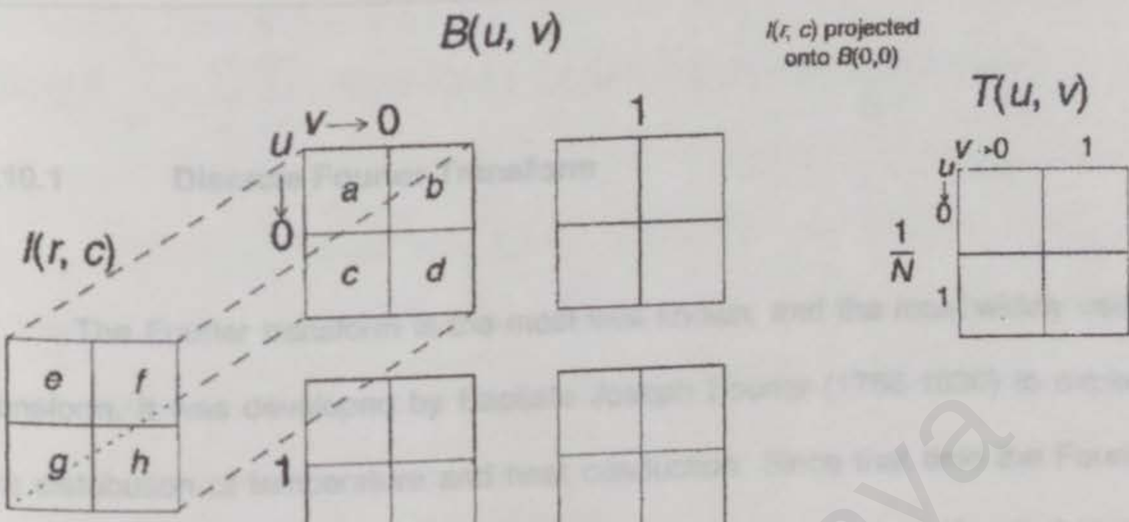


Figure 3.12 : Transform Coefficients

3.10 Fourier Transform

3.10.1 Discrete Fourier Transform

The Fourier transform is the most well known, and the most widely used, transform. It was developed by Baptiste Joseph Fourier (1768-1830) to explain the distribution of temperature and heat conduction. Since that time the Fourier transform has found numerous uses including vibration analysis in mechanical engineering, circuit analysis in electrical engineering, and here in computer imaging. This transform allows for the decomposition of an image into a weighted sum of 2-D sinusoidal terms. Assuming an N x N image, the equation for the 2-D discrete Fourier transform is [26] :

$$F(u,v) = \frac{1}{N} \sum_{r=0}^{N-1} \sum_{c=0}^{N-1} I(r,c)\, e^{-j2\pi (ur+vc)/N}$$

..... **(equation 3.7)**

The base of the natural logarithmic function *e* is about 2.71828; and j, the imaginary coordinate for a complex number, equals √-1. The basis functions are sinusoidal in nature, as can be seen by Euler's identity:

..... **(equation 3.8)**

$$e^{jx} = \cos x + j \sin x$$

The Fourier transform equation can also be written as

$$F(u,v) = \frac{1}{N} \sum_{r=0}^{N-1} \sum_{c=0}^{N-1} I(r,c) \, [\cos(\frac{2\pi}{N}(ur+vc)) + j \sin(\frac{2\pi}{N}(ur+vc))]$$

..... **(equation 3.9)**

In this case, F(u, v) is also complex, with the real part corresponding to the cosine terms and the imaginary part corresponding to the sine terms. If we represent a complex spectral component by F(u, u) = R(u, v) + j1(u v), where R(u, u) is the real part and 1(u, u) is the imaginary part, then we can define the magnitude and phase of a complex spectral component as

$$\text{MAGNITUDE} = | F(u, v) | = \sqrt{[R(u, v)]^2 + [I(u, v)]^2}$$

..... **(equation 3.10)**

and

$$\text{PHASE} = \phi(u, v) = \tan^{-1}\left(\frac{I(u,v)}{R(u,v)}\right)$$

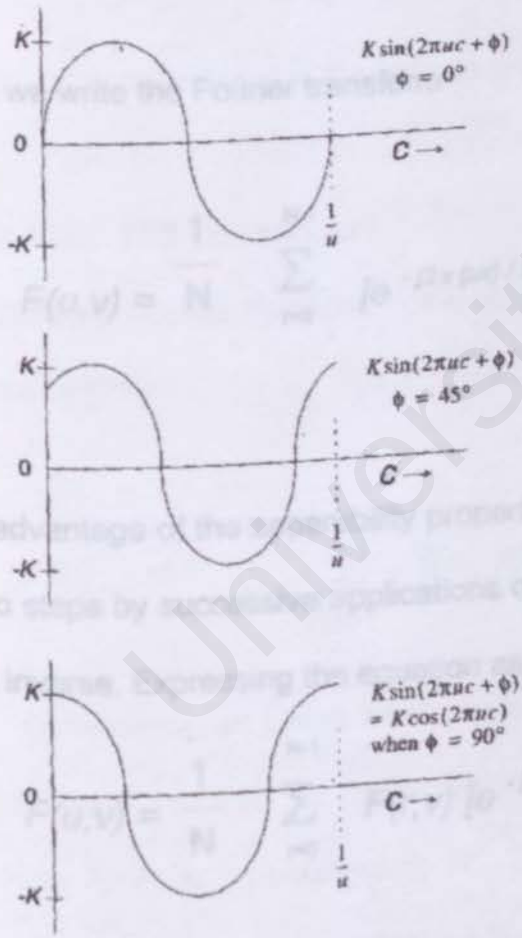
..... **(equation 3.11)**

The magnitude of a sinusoid is simply its peak value and the phase determines where the origin is or where the sinusoid starts (Figure 3.13). After we perform the transform, if we want to get our original image back, we need to apply the inverse transform. The inverse Fourier transform is given by

$$F^{-1}[F(u,v)] = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u,v) \, e^{-j2\Pi(ur+vc)/N}$$

..... **(equation 3.12)**

The F^{-1} notation represents the inverse transform. This equation illustrates that the function I(r, c) is represented by a weighted sum of the basis functions and that the transform coefficients F(u,v) are the weights. With the inverse Fourier transform, the sign on the basis functions' exponent is changed from -1 to +1. However this corresponds only to the phase and not the frequency and magnitude of the basis functions.



a. Magnitude and phase of sinusoidal waves.

Figure 3.13 : Magnitude and Phase of the sinusoidal waves

One important property of the Fourier transform is called separability. If a two-dimensional transform is separable, then the result can found by successive application of two one-dimensional transforms. This is illustrated by first separating the basis image term (also called the transform kernel) into a product, as follows :

$$e\left[^{-j2\pi(ur+vc)/N}\right] = e\left[^{-j2\pi(ur)/N}\right] e\left[^{-j2\pi(vc)/N}\right]$$

..... (equation 3.13)

Next, we write the Fourier transform

$$F(u,v) = \frac{1}{N} \sum_{r=0}^{N-1} [e^{-j2\pi(ur)/N}] \sum_{r=0}^{N-1} I(r,c)\, e^{-j2\pi(vc)/N}$$

..... (equation 3.14)

The advantage of the separability property is that F(u,v) or I(r,c) can be obtained in two steps by successive applications of the one-dimensional Fourier transform or its inverse. Expressing the equation as :

$$F(u,v) = \frac{1}{N} \sum_{r=0}^{N-1} F(r,v)\, [e^{-j2\pi\ ur/N}]$$

..... (equation 3.15)

where

$$F(r,v) = (N) \frac{1}{N} \sum_{r=0}^{N-1} I(r,c)\, [e^{-j2\pi\ vc/N}]$$

..... (equation 3.16)

For each value of r, the expression inside the brackets is a one-dimensional transform with frequency values v=0,1,2,...,N-1. Hence the two-dimensional function F(r,v) is obtained by taking a transform along each row of I(r,c) and multiplying the result by N. The desired result F(u,v) is obtained by taking a transform along each column of F(r,v).

3.10.2 Fast Fourier Transform (FFT)

3.10.2.1 FFT Description

For standard Fourier Transform of a single array, the discrete Fourier transform is given by [28]:

$$F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)\ exp\,[-j2\pi / N] \qquad \text{..... (equation 3.17)}$$

Considering an image of N pixels, the number of complex multiplications and additions required to implement the normal discrete Fourier Transform (Equation 3.17) is proportional to N^2. That is, for each of the N values of u, expansion of the summation requires N complex multiplications of f(x) by *exp [-j2πux/N]* and N – 1 additions to the results. The terms of *exp [-j2πux/N]* can be computed once and stored in a table for all subsequent applications. For this reason the multiplication of u by x in these terms is usually not considered a direct part of the implementation.

Proper decomposition of Equation 3.17 can make the number of multiplication and addition operations proportional to N $\log_2 N$. The decomposition procedure is called the *Fast Fourier Transform (FFT) algorithm* [28]. The reduction in proportionality from N^2 to N $\log_2 N$ represents a significant saving in computational efforts and costs. Obviously, the FFT approach offers a considerable computational advantage over the direct implementation of the Fourier Transform, particularly when N is relatively large. For example, suppose that the FFT of an 8192-point array requires 5 seconds of computation time on a general-purpose computer. The same machine would take about 600 times longer (50 minutes) to compute the Fourier transform on the same array using Eq. 3.17.

3.10.2.2 FFT Algorithm

The FFT algorithm developed in this section is based on the so-called successive doubling method [28]. For convenience we express Eq. 3.17 in the form

$$F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)\, W_N^{ux} \qquad \text{..... (equation 3.18)}$$

where

$$W^{ux}_{N} = exp\ [-j2\pi / N] \qquad \text{..... (equation 3.19)}$$

and N is assumed to be of the form

$$N = 2^{N} \qquad \text{..... (equation 3.20)}$$

where n is a positive integer. Hence N can be expressed as

$$N = 2M \qquad \text{..... (equation 3.21)}$$

where M is also a positive integer. Substitution of Eq. 3.19 into Eq. 3.18 yields

$$F(u) = \frac{1}{2M} \sum_{x=0}^{2M-1} f(x)\, W^{ux}_{2M} \qquad \text{..... (equation 3.22)}$$

$$= \left(\frac{1}{M} \sum_{x=0}^{M-1} f(2x)\, W^{2ux}_{2M} + \frac{1}{M} \sum_{x=0}^{M-1} f(2x+1)\ W^{2(x+1)}_{2M} \right)$$

From Eq. (3.19). $W^{2ux}_{2M} = W^{ux}_{M}$, so Eq. (3.22) may be expressed in the form

$$F(u) = \left(\frac{1}{M} \sum_{x=0}^{M-1} f(2x)\, W^{2ux}_{2M} + \frac{1}{M} \sum_{x=0}^{M-1} f(2x)\, W^{ux}_{M}\, W^{u}_{2M} \right) \qquad \text{..... (equation 3.23)}$$

Defining

$$F_{even}(u) = \frac{1}{2M} \sum_{x=0}^{M-1} f(2x)\, W_{M}^{ux}$$

..... (equation 3.24)

for u = 0, 1, 2, ..., M-1, and

$$F_{odd}(u) = \frac{1}{2M} \sum_{x=0}^{M-1} f(2x+1)\, W_{M}^{ux}$$

..... (equation 3.25)

for u = 0, 1, 2, ..., M-1, reduces Eq. (3.23) to

$$F(u) = \frac{1}{2M} \left[F_{even}(u) + F_{odd}(u)\, W_{M}^{ux} \right]$$

..... (equation 3.26)

Also, since W_{M}^{u+M} and $W_{2M}^{u+M} = -W_{2M}^{u}$, Eqs (3.24) – (3.26) give

$$F(u) = \frac{1}{2} \left[F_{even}(u) - F_{odd}(u)\, W_{2M}^{u} \right]$$

..... (equation 3.27)

Continuing the argument for any positive integer value of n leads to recursive expressions for the number of multiplications and additions required to implement the FFT:

$$M(n) = 2M(n-1) + 2^{n+1} \qquad n \geq 1$$

..... (equation 3.28)

and

$$a(n) = 2a(n-1) + 2^{n} \qquad n \geq 1$$

..... (equation 3.29)

where m(0) = 0 and a(0) = 0 because the transform of a single point does not require any additions or multiplications.

Implementation of Eqs. (3.24) – (3.27) constitutes the successive doubling FFT algorithm. This name comes from the method computing of two-point transform from two one-point transforms, a four-point transform from two two-point transforms, and so on, for any N that is equal to an integer power of 2.

The same fast Fourier transform algorithm can be used – applying the separability property of the 2D transform.

The 2D Discrete Fourier Transform (DFT) can be written as :

$$F(u,v) = \frac{1}{N} \sum_{r=0}^{N-1} [e^{-j2\pi (ur)/N}] \sum_{r=0}^{N-1} I(r,c)\, e^{-j2\pi (vc)/N}$$

. . . . . **(repeat of equation 3.14)**

The right hand sum is basically just a one-dimensional DFT if x is held constant. The left hand sum is then another one-dimensional DFT performed with the numbers that come out of the first set of sums.

So we can compute a two-dimensional DFT by

- performing a one-dimensional Discrete Fourier Transform (DFT) for each value of x (*i.e.* for each column of $f(x,y)$), then
- performing a one-dimensional DFT in the opposite direction (for each row) on the resulting values.

This requires a total of 2 N one dimensional transforms, so the overall process takes time $\boldsymbol{O (N^2 \log_2 N)}$.

Chapter 4

Artificial Neural Networks

4.1 Introduction to Artificial Neural Networks

Artificial Neural Networks, or as it is commonly referred to as, "neural networks" has been motivated right from its inception by the recognition that the brain computes in an entirely different way from the conventional digital computer.

Artificial neural networks (simplified as ANN's) are simplified method of the central neural system of a human being. They are networks of highly interconnected neural computing elements that have the ability to response to input stimuli and to learn to adapt to the environment. To enable one to understand ANN's better, here's a simple illustration. Ever wondered how you identify a car? Well, when you were young, your parents would have told you that the combination of four wheels in the shape of what you now call a car is in fact a car. How this work? When you were taught to recognise a car, you learnt it, and after some conditioning, it became a part of your long-term memory. Following that, the next time you came across a car of any shape and size (input of 4 wheels, shape of a car, etc) your nervous system will process it (processed by the neuron) and recall it from your memory and tell you that this it is a car. This is what we refer to also as knowledge.

Similarly, trained neural networks extract information from their knowledge base. Neural networks are similar to our nervous system. In neural networks, computers are programmed to mimic our nervous system. To be specific, the brain routinely accomplishes perceptual recognition tasks (e.g. recognizing a familiar face embedded in an unfamiliar scene) in something of the order of 100 – 200ms, whereas tasks of much lesser complexity will take days on a huge conventional computer [22].

A vague description is as follows:

"An ANN is a network of many very simple processors ("units"), each possibly having a (small amount of) local memory. The units are connected by unidirectional communication channels ("connections"), which carry numeric (as opposed to symbolic) data. The units operate only on their local data and on the inputs they receive via the connections."

Another definition of a neural network is viewed as an adaptive machine [20] (adapted from Aleksander and Morton (1990)) is as follows:

"A neural network is a massively parallel distributed processor that has a natural propensity for storing experimental knowledge and making it available for use. It resembles the brain in two respects:

1. *Knowledge is acquired by the network through a learning process.*
2. *Interneuron connection strengths known as synaptic weights are used to store the knowledge".*

The design motivation is what distinguishes neural networks from other mathematical techniques:

"A neural network is a processing device, either an algorithm, or actual hardware, whose design was motivated by the design and functioning of human brains and components thereof."

Most neural networks have some sort of "training" rule whereby the weights of connections are adjusted on the basis of presented patterns. In other words, neural networks "learn" from examples, just like children learn to recognize dogs from examples of dogs, and exhibit some structural capability for generalization. Neural networks normally have great potential for parallelism, since the computations of the components are independent of each other.

4.2 Biological Neural Networks and Artificial Neural Networks

Much of the research work in ANN's has been inspired and influenced by our knowledge of biological nervous systems. Our knowledge of the mammalian nervous system is far more complete. The basic computing element in biological systems is the neuron. A neuron is a small cell that receives electrochemical stimuli from multiple sources and responds by generating electrical impulses that are transmitted to other neurons or affecter cells. Each neuron can store information just like the transistor and capacitor in our electronic circuit board. The neuron is analogue to a capacitor and also a node in our neural network. Neurons are complex cell that response to electrochemical signals. They are composed of a nucleus, a cell body, numerous dendrites links providing input "connection" from other neurons through synapses and an axon trunk that carries an action potential output to other neuron through terminal links and synapses.

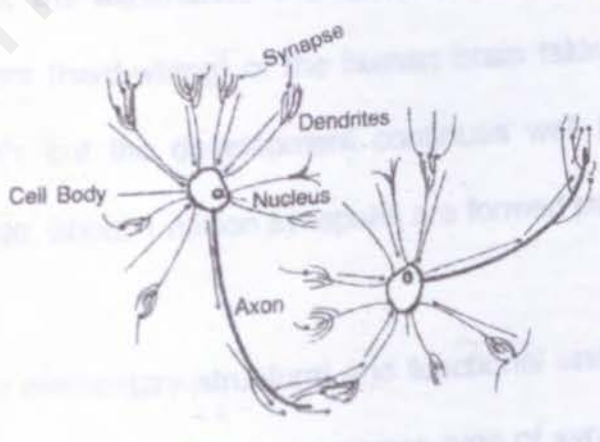


Figure 4.1 : Structure of a neuron

Typically, neurons are magnitudes slower than silicone logic gates; events in the silicon chip happen in the nanosecond (10^{-9}) range, whereas neural events happen in the millisecond (10^{-3}) range. However, the brain makes up for the relatively slow rate of operation of a neuron by having a truly staggering number of neurons (nerve cells) with massive interconnection between them. It is estimated that there must be on the order of 10 billion neurons in the human cortex, and 60 trillion synapses or connections [23].

The net result is an enormously efficient structure called the brain. Specifically, the energetic efficiency of the brain is approximately 10^{-16} Joules (J) per operation per second, whereas the corresponding value for the best computers in use today(1994) is about 10^{-6} Joules per operation per second [21].

How then does the human brain do it? At birth, a brain has great structure and the ability to build up its own rules to what we usually refer to as 'experience'. Indeed, the experience is built up over the years, with the most dramatic development (hard-wiring) of the human brain taking place in the first two years from birth; but the development continues well beyond that stage. During this early stage, about 1 million synapses are formed per second.

Synapses are elementary structural and functional units that mediate the interactions between neurons. The most common type of synapse is a chemical synapse, which operates as follows. A presynaptic process liberates a transmitter substance that diffuses across the synaptic junction between neurons and then acts on a postsynaptic process. Thus, a synapse converts a presynaptic electrical signal into a chemical signal and then back into a postsynaptic electrical signal [23]. It is assumed that a synapse is a simple connection that can impose *excitation* or *inhibition*, but not both on the receptive neuron.

In an adult brain, plasticity (flexibility) may be accounted for by two mechanisms: the creation of new synaptic connections between neurons, and the modification of existing synapses. Axons, the transmission lines and dendrites, the receptive zones, constitutes two types of cell filaments that are distinguished on morphological grounds; an axon has a smoother surface, fewer branches and greater length, whereas a dendrite has an irregular surface and more branches [20]. Neurons come in a wide variety of shapes and sizes in different parts of the brain. Figure 2.1a illustrates the shape of the pyramidal cell, which is one of the most common types of cortical neurons. Like many other types of neurons, it receives most of its inputs through dendritic spines. The pyramidal cell can receive 10,000 or more synaptic contacts and it can project onto thousands of target cells.

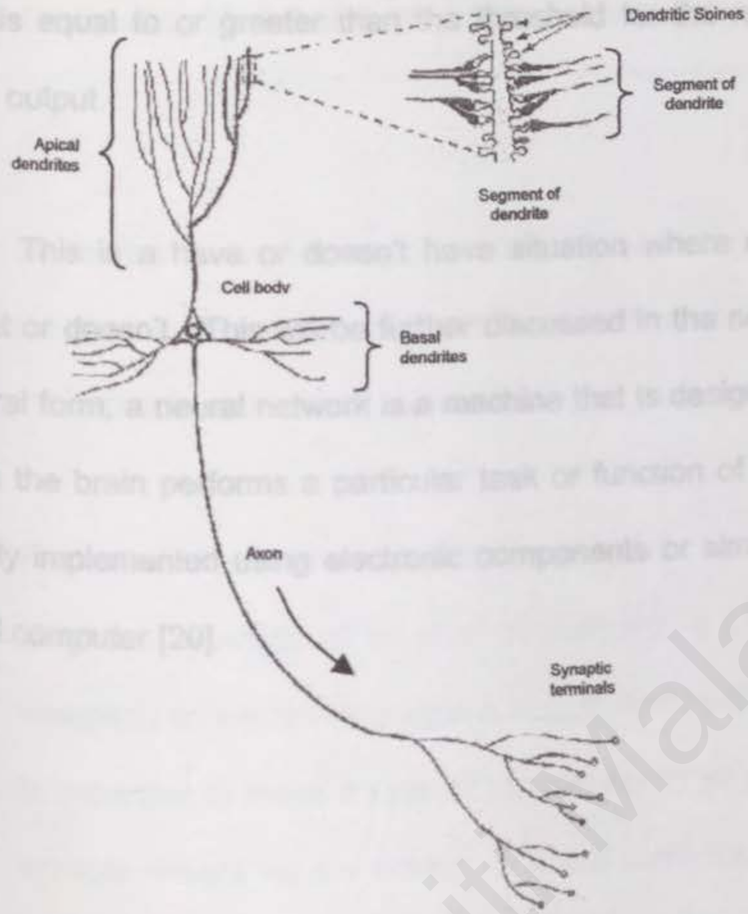


Figure 4.2 : The pyramidal biological cell

In a simplified scenario, nerves conduct impulses from receptor organs such as eyes to affecter organs. The point between two neurons in a neural pathway, where the termination of the axon of one neuron comes into close proximity with the cell body or dendrites of another is called synapses.

In breaking the synaptic activity down, the point's show in figure 4.2 are important for our neural network analogy. Signals come into the synapses. These are inputs. They are "weighted" (in ANN's). Some signals excite (are positive) and others inhibit (are negative). The effects of all weighted inputs are summed. If the sum is equal to or greater than the threshold for the neuron, then the neuron gives output.

This is a have or doesn't have situation where either the neuron gives output or doesn't. (This will be further discussed in the next chapter). In its most general form, a neural network is a machine that is designed to model the way in which the brain performs a particular task or function of interest; the network is usually implemented using electronic components or simulated in software on a digital computer [20].

4.3 Computing Model of Artificial Neural Networks

A neuron is an information-processing unit that is fundamental to the operation of a neural network. Figure 3.3 shows the model for a neuron. We may identify three basic elements of the neural model, as described here [20]:

1. A set of synapses or connecting links, each of which is characterized by a weight or strength of its own. Specifically, a signal x_j at the input of synapse j connected to neuron k is multiplied by the synaptic weight w_{kj}. It is important to make a note of the manner in which the subscripts of the synaptic weight w_{kj} are written. The first subscript refers to the neuron in question and the second subscript refers to the input end of the synapse to which the weight refers; the reverse of this notation is also used in the literature. The weight w_{kj} is positive if the associated synapse is excitatory; it is negative if the synapse is inhibitory.
2. An adder for summing the input signals, weighted by the respective synapses of the neuron; the operations described here constitute a linear combiner.
3. An actuation function for limiting the amplitude of the output of a neuron. The activation function is also referred to in the literature as a squashing function in that it squashes (limits) the permissible amplitude range of the output signal to some finite value. Typically, the normalized amplitude

range of the output of a neuron is written as the closed unit interval [0,1] or alternatively [-1,1].

The model of a neuron shown in Figure 3.3 also includes an externally applied threshold θ_k that has the effect of lowering the net input of the activation function. On the other hand, the net input of the activation function may be increased by employing a bias term rather than a threshold; the bias is the negative of the threshold.

In mathematical terms, we may describe a neuron k by writing the following pair of equations:

$$u_k = \sum_{i=1}^{p} w_{kj} x_j \qquad \text{.............. (equation 4.1)}$$

and

$$y_k = \varphi (u_k - \theta_k) \qquad \text{.............. (equation 4.2)}$$

where x_1, x_2, ... x_p are the input signals; w_{k1}, w_{k2}, ... w_{kp} are the synaptic weights of neuron *k*; u_k is the linear combiner output; θ_k is the threshold; $\varphi(.)$ is the activation function; and y_k is the output signal of the neuron.

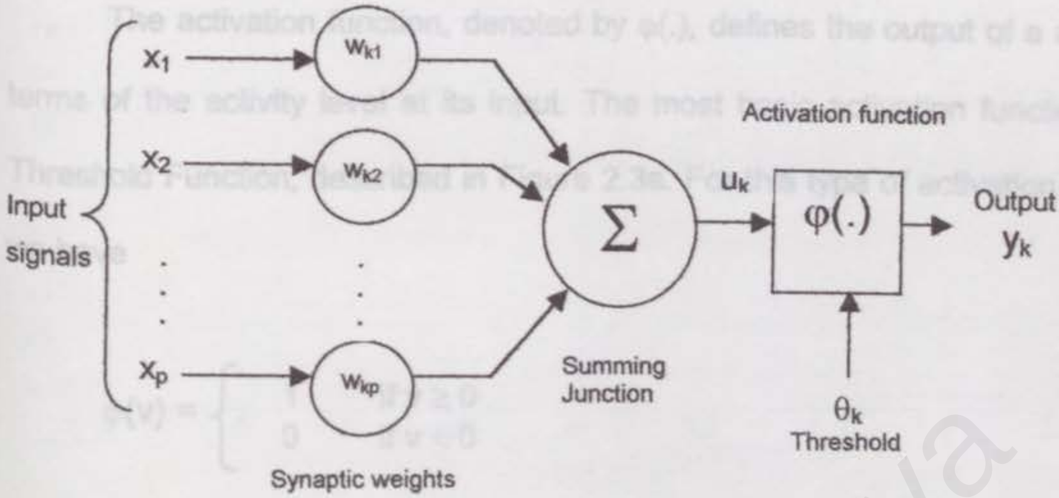


Figure 3.3 : Non-linear model of a neuron

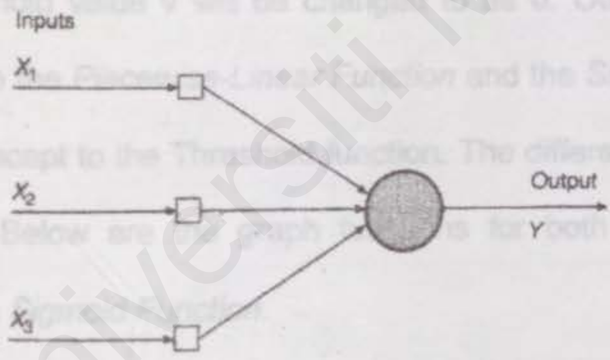


Figure 3.4 : A Simple Artificial Neural Network

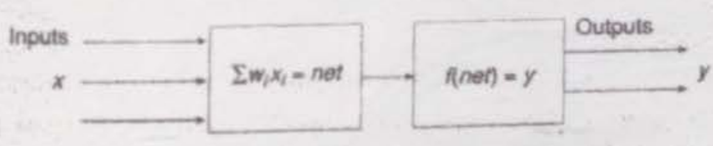


Figure 3.5 : Schematic Diagram of a Single Neuron

The activation function, denoted by $\varphi(.)$, defines the output of a neuron in terms of the activity level at its input. The most basic activation function is the Threshold Function, described in Figure 2.3a. For this type of activation function, we have

$$\varphi(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{if } v < 0 \end{cases}$$

This means that any output which is equals to or more than the threshold value v will automatically be referred to as 1. All at once, any output with values below the threshold value v will be changed to as 0. Other types of activation functions are like the *Piecewise-Linear Function* and the *Sigmoid Function*, which are similar in concept to the Threshold function. The differences are based on the graph function. Below are the graph functions for both the *Piecewise-Linear Function* and the *Sigmoid Function*.

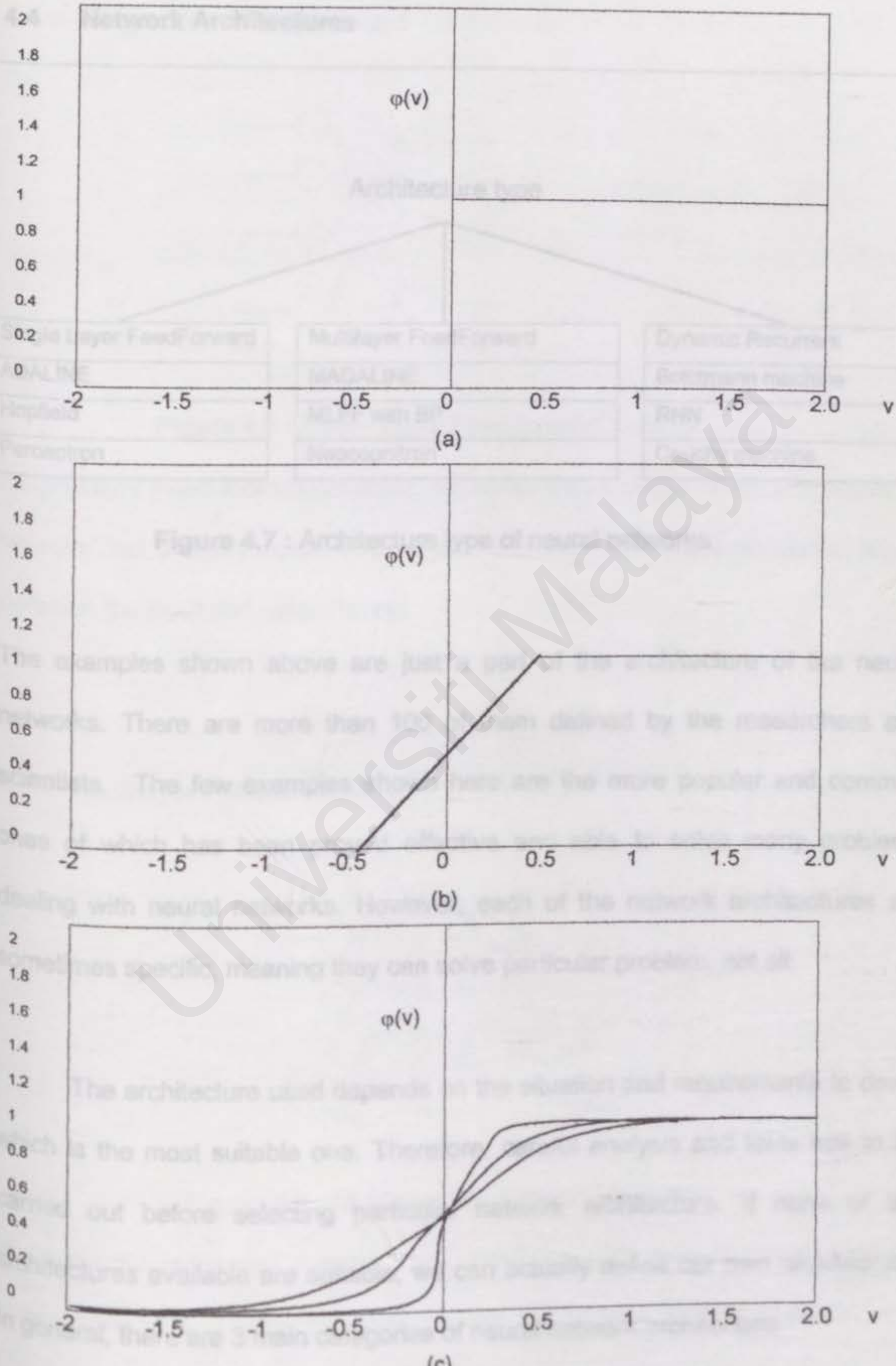


Figure 4.6 :(a)Threshold function (b)Piecewise-Linear function (c)Sigmoid function

4.4 Network Architectures

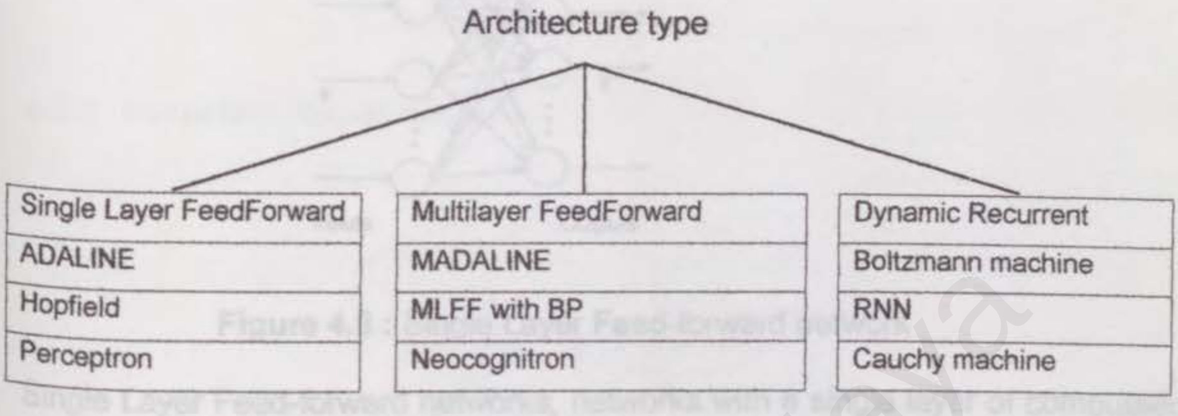


Figure 4.7 : Architecture type of neural networks

The examples shown above are just a part of the architecture of the neural networks. There are more than 100 of them defined by the researchers and scientists. The few examples shown here are the more popular and common ones of which has been proved effective and able to solve many problems dealing with neural networks. However, each of the network architectures are sometimes specific, meaning they can solve particular problem, not all.

The architecture used depends on the situation and requirements to deem which is the most suitable one. Therefore, careful analysis and tests has to be carried out before selecting particular network architecture. If none of the architectures available are suitable, we can actually define our own architecture. In general, there are 3 main categories of neural network architecture.

4.4.1 Single Layer Feed-forward

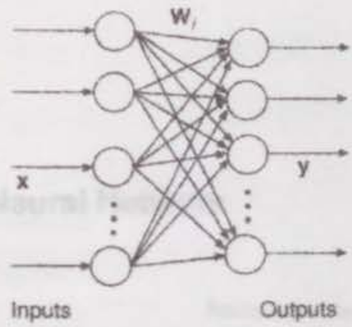


Figure 4.8 : Single Layer Feed-forward network

Single Layer Feed-forward networks, networks with a single layer of computation neurons that process inputs in a forward direction. There are no hidden layers between the input and output layers.

4.4.2 Multilayer Feed-forward

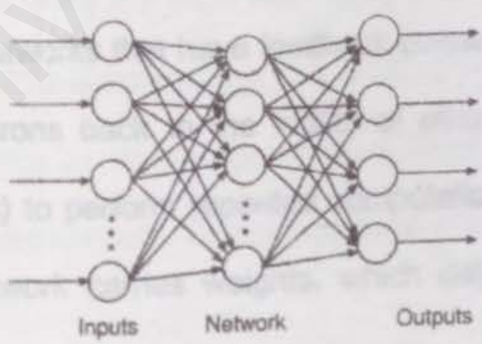


Figure 4.9 : Multilayer Feed-forward Network

Multilayer Feed-forward network, are networks with two or more layer of connections with weights that process the inputs in a forward direction. The hidden layers may compose of 1 or more layers of which the weights help determine the output.

4.4.3 Recurrent Neural Network

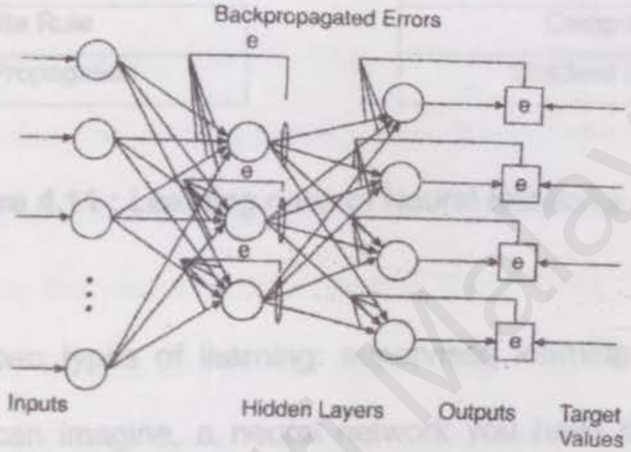


Figure 4.10 : Recurrent network with back propagation

Recurrent network, networks that have feedback connection which propagate the outputs of some neurons back to the inputs of other neurons (including self-feedback connections) to perform repeated computation on the signals. Each of the nodes in the network carries weights, which determine the output of the network.

4.5 Learning Types / Methods

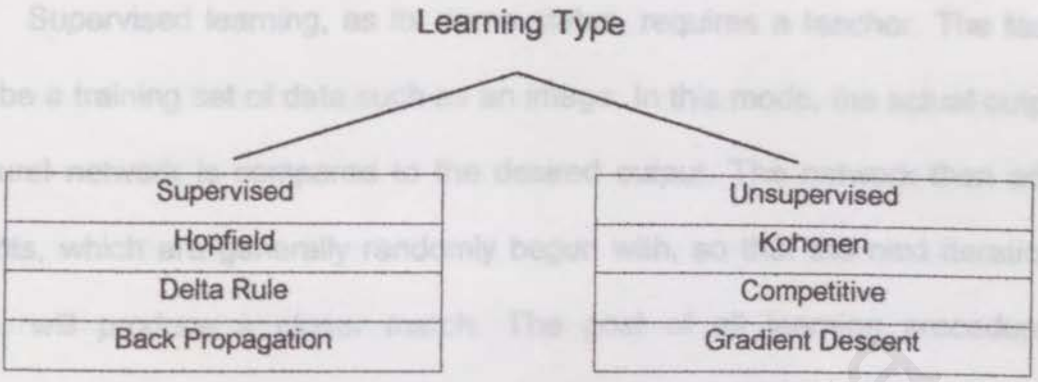


Figure 4.11 : Learning rules of Neural Networks

There are two types of learning: supervised learning and unsupervised learning. As you can imagine, a neural network you have designed (from any type of architecture), is analogous to a newborn baby, it knows nothing. You have to teach and train them so that they learn. This 'learning' is what us humans refer to as experience. Why are wise men usually old? Have you ever heard of the young wise man? The older you get, the wiser. This is because the older one is, the more experience or knowledge (training/teaching) one gains, thus the wiser one gets. The meaning of knowledge here is given by the following generic definition [20]:

" *Knowledge refers to stored information or models used by a person or machine to interpret, predict, and appropriately respond to the outside world.* "

4.5.1 Supervised Learning

Supervised learning, as its name states, requires a teacher. The teacher may be a training set of data such as an image. In this mode, the actual output of a neural network is compared to the desired output. The network then adjusts weights, which are generally randomly begun with, so that the next iteration, or cycle, will produce a closer match. The goal of all learning procedures is ultimately to minimize the error between the desired output and the current output sample by continuously modifying the weights. With supervised learning, it is necessary to train the neural network before it becomes operational. Training consists of presenting input and output data to the network. This data is often referred to as training set. That is, for each input presented, the corresponding desired output is presented as well. When there is no further learning, meaning there are no error feedback, the weights are fixed.

A simple analogy to enable a clearer picture (of learn by example): Imagine a teacher teaching a student the pronunciation of a new word, for example the word "technology". The teacher will first pronounce the word, at which the student will listen and try to learn. Generally, the student will not get it completely fault free the first time. Now comes the coaching part. The teacher now asks the student to pronounce the word, and assesses if the pronunciation is correct (result checking). If so, the training stops and the student is ready to learn new things. This is when the student is said to have knowledge. Otherwise, learning continues until pronunciation is correct. The student then recognizes the word the next time he sees it, and is able to pronounce it correctly.

In neural networks, the part where the teacher coaches the student is known as training; the teacher's words, as the training set (input) and desired output, and after the student learns it (gains knowledge), the knowledge base. The knowledge base functions as the reference for the network to recognize future situations/inputs.

4.5.2 Unsupervised Learning

Unsupervised learning sometimes called self-supervised learning. Here, networks use no external influences to adjust their weights. Instead there is an internal monitoring of performance. The network looks for regularities or trends in the input signals, and makes adaptations according to the function of the network. Even without being told whether it's right or wrong, the network still must have some information about how to organize itself. An unsupervised learning algorithm might emphasize cooperation among cluster of processing elements. In such a scheme, the clusters would work together and try to stimulate each other. If external input activated any node in the cluster, the cluster activity as a whole could be increased. Likewise, if external input to nodes in the cluster was decreased, that could have an inhibitory effect on the entire cluster.

At this moment, unsupervised learning is not well understood by many and is still the subject of much research because many situations do not have a data set available until a conflict arises. In certain situations, the neural network is able to produce good results on unlearnt conditions and events. Supervised learning, on the other hand, has achieved a reputation for producing good results in practical applications and is gaining in popularity.

4.6 Learning Laws

Many learning laws are in common use. Most of the common ones are some sort of variation of the best known and oldest learning laws, Hebb's Rule, or the Hebb synapse. Research has continued, however, and new ideas are being tried. Some researchers have the modelling of biological learning as their main objective; others are experimenting with adaptations of their perceptions of how nature handles learning. Unfortunately, there is still a great deal we don't know about how learning happens, and experimental evidence is not easy to obtain. Learning is certainly more complex than the simplifications represented by the laws we have developed.

4.6.1 Hebb's Rule

The first, and undoubtedly the best known learning rule was introduced by Donald Hebb. His basic rule, simply stated, is this: If a processing element receives an input from another processing element, and if both are highly active (mathematically have the same sign), the weight between the processing elements should be strengthened.

4.6.2 Delta Rule

This commonly used rule is based on the simple idea on continuously modifying the strengths of the connections to reduce the difference between the desired output value and the current output value of a processing element. This rule is also referred to as the Widrow-Hoff Learning Rule (Widrow and Hoff used in it in their ADALINE model) and the Least Mean squared (LMS) Learning Rule, because it minimizes the mean squared error.

4.6.3 Gradient Descent Rule

Here we have a mathematical approach to minimizing the error between actual and desired outputs. The weights are modified by an amount proportional to the first derivative of the error with respect to the weight. This rule is commonly used, even though it converges to a point of stability very slowly. You could think of this procedure as descending along the curve to the bottom of a basin. Once you reach the bottom, the error is at the minimum; you would be at the lowest point on the basin. The Delta Rule is one example of a gradient decent rule.

4.6.4 Kohenen's Learning Law

This procedure, developed by Teuvo Kohenen was inspired by learning in biological systems. It is employed only in unsupervised learning network application. In this procedure, the processing elements compete for the opportunity of learning. The processing element with the largest output is declared the winner and has the capability of inhibiting its competitors as well as exciting its neighbours. Only the winner is permitted an output, and only the winner plus its neighbours are permitted to adjust their weights.

Further, the size of the neighbourhood can vary during the training period. The usual pattern is to start with a larger definition of the neighbourhood, and narrow in as the training proceeds. Because the winning element is defined as the one that has the closest match to the input pattern, Kohonen networks model the distribution of the inputs.

4.6.5 Back Propagation Learning law

The back propagation of errors technique is the most commonly used generalization of the Delta Rule. This procedure involves two phases. The first phase, called the forward phase, occurs where the input is presented and propagated forward through the network to compute an output value for each processing element. For each processing element, all current outputs are compared with the desired output, and the difference, or error, is computed. In the second phase, called the backward phase, the recurring difference computation (from the first phase) is now performed in a backward direction. Only when these two phases are complete can new inputs be presented.

Generally this technique is applied to hierarchical networks of three (or more) layers. At the output layer, there's no problem you know the actual output from each node and also the expected output. The trick comes in adjusting the weights on hidden layers. What is the "desired" output for a hidden-layer processing element? How do you tell how much each individual node from a proceeding layer contributed to the total error at a given node? The information from the forward phase tells you how much each input influenced the error at that node. With two phases, all the weights can be appropriately adjusted.

So far there doesn't appear to be any evidence that this method is used in biological systems. Additionally, there are several important drawbacks: back propagation is very slow, requires much off-line training, exhibits temporal instability (can oscillate), and has a tendency to become stuck at local minima. This last item occurs when the system finds an error that is lower than the surrounding possibilities but does not finally get to the smallest possible error.

Chapter 5

Proposed Methodology

5.1 Project Description

Because of the cumulative look of the Fourier Transform and the ability of neural networks, it is therefore combined and used in parallel in this approach. The Fast Fourier Transform (FFT) algorithm is a modification of the Fourier Transform, which results from a cumulative analysis of the image for pattern recognition. This method however may be sensitive to differences in two samples of a same fingerprint as it looks at the image collectively. After the FFT algorithm is performed on the image, the coefficients at each point of segmentation will be fed into the neural networks for processing and recognition. Neural networks are used for their ability for parallel processing and learning abilities. This project consists of 3 major modules, being:

1. *Fingerprint Image Initialisation*

   This module involves the step of obtaining the fingerprint samples, and removing noise and image irregularities in the acquired fingerprint image. All images used are represented in the format below:

   - Grey-scaled image
   - Image size of approximately 250x250 to 500x500 pixels
   - In JPEG, BMP or TIFF
   - Avoid excessive inking and pressure, which will cause smearing.

2. *Feature Extraction*

   This is the most crucial module in the system, of which the features of each fingerprint, which gives them the individuality, are extracted. The path of the initial fingerprint image is saved in the database. The identity of the next fingerprint image used is to be validated against the initial fingerprint image.

3. *Recognition / Training*

   All of the features extracted from the feature extraction module are to be fed into the Neural Network, either for training or recognition. The results of the training will be saved as a neural network object and user's details stored in the database. If a validated image is found in the database, it is then responsible to display the user's name and acceptance after successful verification. Verification is based on a standard preset threshold value.

5.2 Approach

5.2.1 Prototyping Model Approach

The prototyping approach is chosen to develop the project in this paper. Prototyping is the art of building a small scale, representative or working model of the entire system/subsystem based on user requirements for purposes of discovering or verifying those requirements [27]. This is an approach where a simple running program will be developed first, and modified and changed to suit the objectives and target.

This model provides communication basis for discussions among all the groups involved in the development process, especially between users and developers. Furthermore, prototyping also enables the developer to adopt an approach to software construction based on experiment and experience.

Steps in prototyping are as follows:

1. *Analysis and Specification* : The first step to prototyping requires investigation, analysis and problem definition of existing systems. A comparison is made on advantages and disadvantages of various approaches and methods.

2. *Design and Development* : The prototype is developed in this step. The final product will be based on development of this prototyped system.

3. *Utilisation and Evaluation* : The feasibility and functionality of the system will be evaluated based upon the results and response to the system.

4. *Revision and Enhancement* : The prototype is revised, refined and enhanced according to results and feedback on the system.

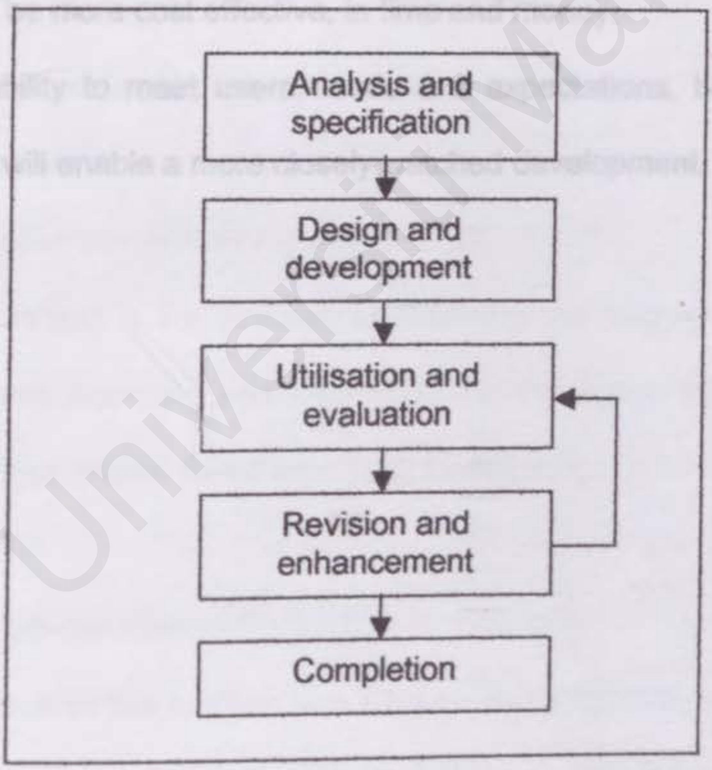


Figure 5.1 : Prototyping model

5.2.2 Justification

The reason for taking this approach is for the following reasons:

1. All the 3 main modules are very closely linked, and the results of each of the modules will carry weight in the next module.
2. The ability for modification or changes to the system while its still in its early development stages.
3. The possibility of scrapping a system that is not what was hoped to be would be more cost effective, in time and money.
4. The ability to meet users needs and expectations, based on feedback, which will enable a more closely watched development.

5.3 Development Strategy

The development strategy covers the area of functional and non-functional requirements of the proposed Fingerprint Recognition System. The functional requirements can be divided into five main modules and the non-functional requirements covering aspects from the user-interface to response time.

5.3.1 Functional Requirements

1. *Fingerprint acquisition*

   This function is the process of obtaining the fingerprint image, through fingerprint stamping and then scanning the image in 300dpi resolution. The image is then saved as a TIFF format image.

2. *Image pre-processing*

   The result of this function is a cleaner, more normalized image. This step involves filtering out noise effects that result from fingerprint irregularities. Running median and Gaussian filters, together with histogram equalization operators are used to obtain a visually clearer image for a clearer image.

3. *Feature extraction*

   Fourier filters are used in this step, implemented using Fast Fourier Transform, where pattern recognition processes take place here. However, further feature extraction filters may be needed to supplement more accurate feature extraction. The result from this function is to be used in the next process.

4. *Neural Networks*

   The inputs of the Artificial Neural Networks are obtained from the feature extraction process after the features extraction filters has been performed on the pre-processed image. These inputs will all carry weight in the output (result) of the recognition process.

5. *Fingerprint Matching*

   After all of the functions above have been performed on the fingerprint, if the verification is successful the result will be displayed to the user, together with information on the fingerprint used and the identification of it.

5.3.2 Non-Functional Requirements

Non-functional requirements are an essential definition of system properties and constraints under which a system must operate. Although the non-requirements are very subjective, they are as important as the functional requirements. There are five aspects of which are looked into:

1. *User-friendly Interface*

   Simple but attractive design and bonded with straightforward interfaces will lend users to control and manage the system easily.

2. Response Time

   This looks at the ability of the system to produce results within an acceptable interval. In this system, two different response times are taken into consideration. The first one, being the Neural Network training, and the second the recognition of fingerprints.

3. Expandability

   Maintainability is the degree to which architectural and procedural design can be extended.

4. Reliability

   A system is considered reliable if it doesn't produce costly failures when used in a reasonable manner.

5. Robustness

   Robustness refers to the quality of results a system is able to produce, and the ability to handle or avoid disaster in the series of unexpected or faulty data inputs.

5.4 Proposed Tools

5.4.1 Hardware Requirements

In this paper, fingerprints were sampled using the inkpad method, at which a finger is rolled over an inkpad, and stamped onto a piece of paper. The paper will eventually be scanned into the computer using an Epson Perfection 610 scanner in a 300dpi resolution and saved in TIFF image. The following hardware requirements are necessary :

1. Inkpad and paper for fingerprint stamping
2. Epson Perfection 610 scanner
3. Minimum 64MB RAM
4. Minimum processor clock speed of Pentium 166MHz

5.4.2 Software Requirements

- *Operating system*

The operating system used in this study is Microsoft Windows 98. This is because Windows 98 performs in a stable and user-friendly manner and is also convenient to the developer and users for its widely used platform.

- *System implementation*

In this paper, the application will be developed on Matlab v5.3.1 (for Windows 95/98). The reason for choosing this software is for the reasons as follows:

1. has Neural Network toolbox for implementation into application.
2. supports use of basic image processing tools for working on images.
3. has Graphical User Interface (GUI) capabilities for developing applications.

- *Database*

Microsoft Access 97 is used in this application as the database selection. Microsoft Access 97 is a relational database management system used to create and manage the relational databases. The database is used to store fingerprint features and outputs and information regarding the user such as the name, sex, address and so on. Access is used for the following reasons :

i. provides a powerful and intuitive development environment.

ii. a common application which is also user friendly.

5.5 System Design

5.5.1 Image Acquisition

Image acquisition, is the process where an image of the fingerprint is obtained. This is where a fingerprint is stamped onto a piece of paper, and then scanned. The image will be scanned using a 300dpi resolution and saved in TIFF format of a size varying between 250 – 500 pixels, depending on the size of the fingerprint.

5.5.2 Image Pre-processing

This step of the system is for removing any irregularities and noise effects in the image saved. Impurities such as dust/dirt or excess pressure relating to smearing are considered to be noise. The image pre-processing step plays a vital part in the recognition, as any unresolved noise or irregularities will cause the system to treat these as features in the feature extraction process, which will result in false acceptance/rejection. Images obtained through the fingerprint acquisition step are pre-processed through 3 steps, which are: 1) median filter, 2) mean filter, and 3) histogram equalization.

5.5.2.1 Median Filter

The median filter works such that, for a pixel with a neighbourhood of N x N, the pixel value is changed to the median value of the rearranged grey-level values within the neighbourhood (refer Chapter 3.5). This filter works best with salt and pepper noise. This will remove noise effects that result from scanning impurities.

5.5.2.2 Gaussian Filter

A Gaussian filter is a smoothing filter (low pass filter), which reduces noise in an image by lowering the proportion of its high frequency components. It allows low frequencies to pass unchanged but attenuate high frequencies. The Gaussian filter is used with the general purpose of obtaining an average value of the image, which solves Gaussian noise, or uniform noise that blurs the image. This step results in a smoother and less distorted image as compared to the original.

5.5.2.3 Histogram Equalization

Histogram equalization intends to work at improving the image from effects such as over-inking and under-inking. It will produce a much more visually acceptable image, which makes the features clearer and also more uniformed grey levels in the image.

5.5.3 Feature Extraction

The feature extraction process involves the implementation of the Fast Fourier Transform (FFT) algorithm on the fingerprint image. The FFT algorithm is a transform of the spatial domain into the frequency domain, which basically is a cumulative look into the spatial domain values, and transformed into the frequency domain.

Simplified, the Fourier Transform transforms the spatial domain of the image into the frequency domain, based on grey-level values. All pixels in the spatial domain image contribute to each value in the output image for frequency transforms. Thus, after the transform, the values on the new frequency domain are treated as the features of the fingerprint image, and used for recognition. These coefficients are used as inputs into the neural network for recognition purposes.

When an image is transformed into the frequency domain (from the spatial domain), values of the image are based on the grey-level of the image. All of the pixels correspond to the output in the frequency domain. In the frequency domain, the image is represented in low frequencies, and filter out high frequencies. This enhances the features of the image and attenuates the unwanted features.

The product of a Fast Fourier Transform (refer to Section 3.10) is a coefficient F(u,v) which will serve as feature vectors in the output image. These vectors are transformed features of the fingerprint image. Each of these feature coefficients produced are considered to be an input vector in the Neural Network. However, as the outcome of the Fourier Transform is dependant on the grey levels of the fingerprint, it may be possible that this method is sensitive to varying depths of colour, and also to degrees of rotation. Therefore, it may be necessary to use other feature extraction methods for a more robust comparison of prints.

5.5.4 Neural Networks

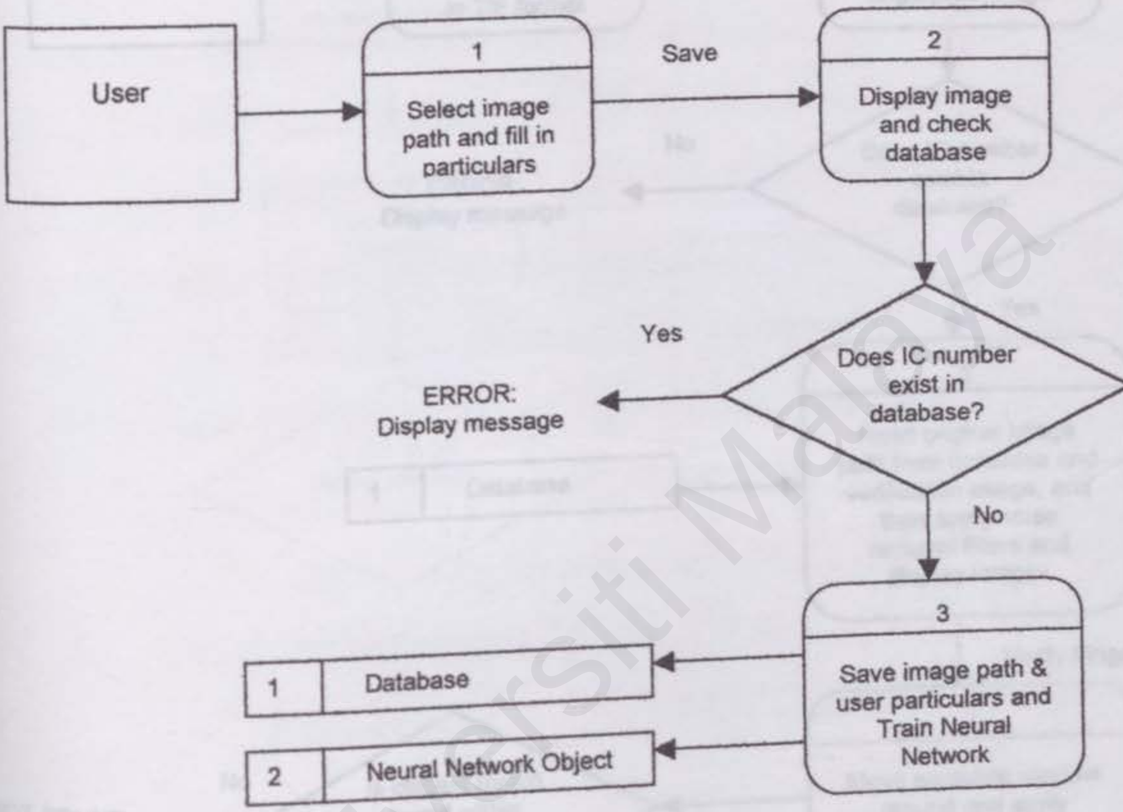
Neural networks are used to look at the features as a whole, rather than single feature. A back propagation neural network is used for the training process, where weights are modified based on the desired output. Neural networks are used for their ability in parallel processing, which enables a faster more efficient use of resources. The ability of the network to learn is also considered a plus point as it makes the system expandable and more robust. The strongpoint for choosing neural networks as opposed to other approaches is discussed in Section 5.6.

5.5.5 Matching / Recognition

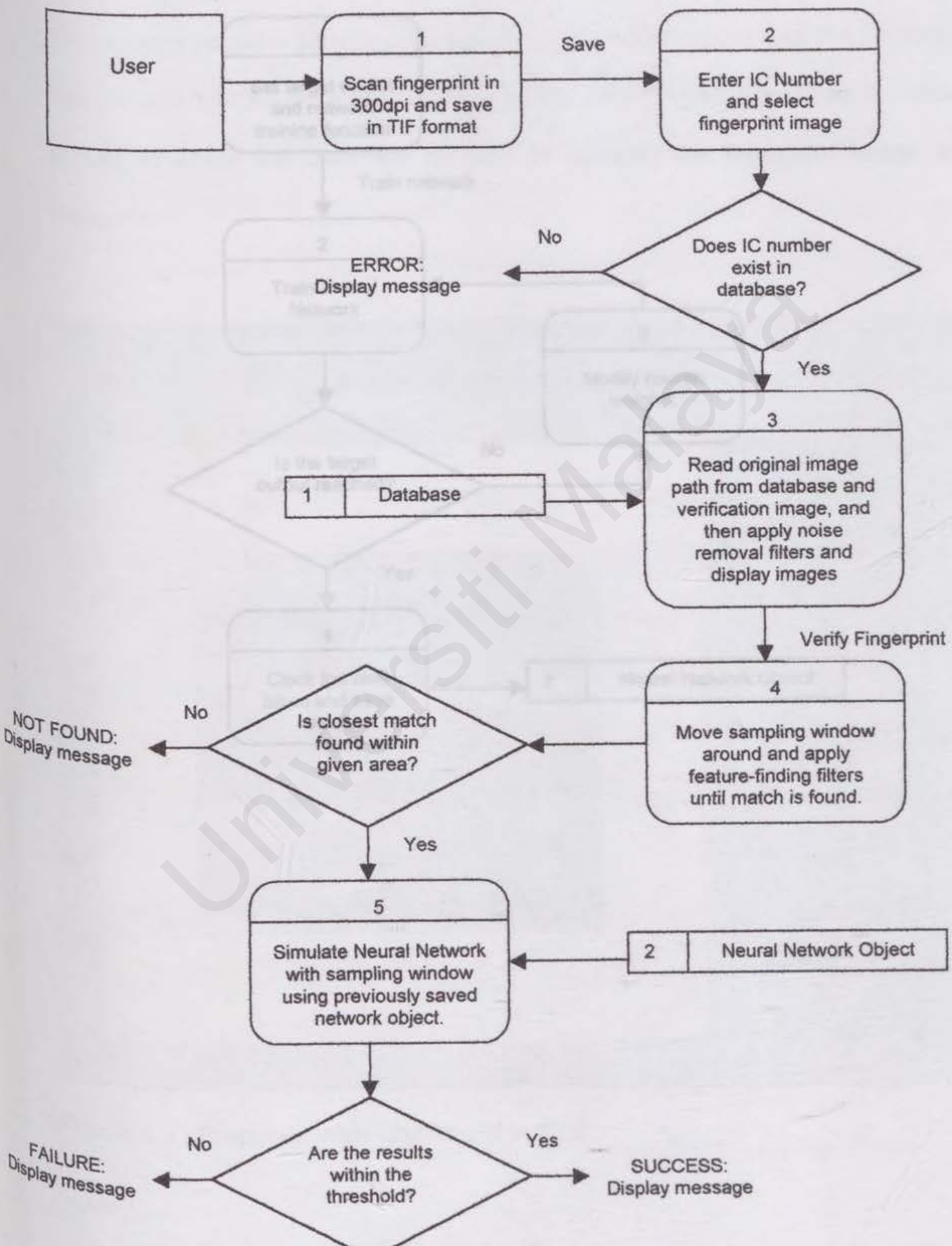
After all of the functions above have been performed on the fingerprint, the results are used for comparisons. The results obtained in the final stage from the neural networks output will determine if the fingerprint is in fact a valid fingerprint in the database. If the result is within the preset threshold value, it is accepted, and rejected otherwise. The threshold value is determined based on the false rejection and false acceptance numbers. On the whole, a successful and working system should have less than 1% of false rejection or acceptance cases. After the process of validating it, information on the fingerprint identity is presented to the user. Therefore, the path of the original fingerprint image is stored in the database.

5.5.6 System Flow Diagram

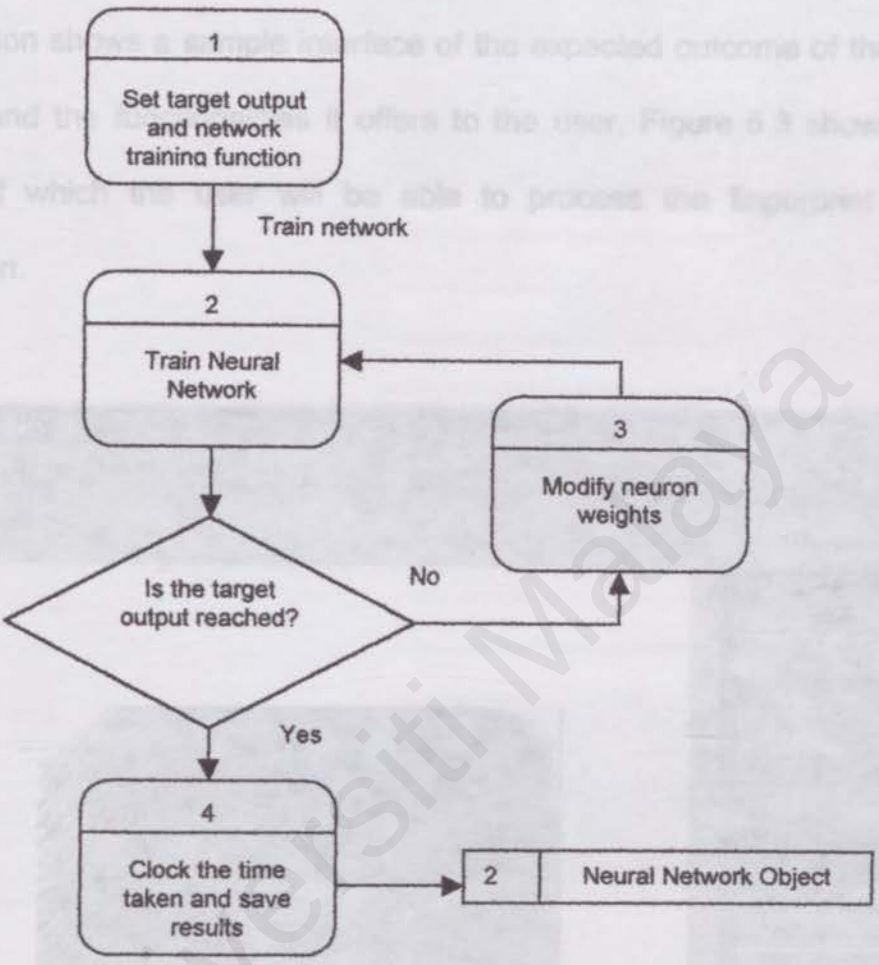
5.5.6.1 Adding New Users



5.5.6.2 Verifying Fingerprints



5.5.6.3 Training Neural Networks



5.5.7 System Mock-up

This section shows a sample interface of the expected outcome of the proposed system, and the functionalities it offers to the user. Figure 5.3 shows the main screen of which the user will be able to process the fingerprint image for recognition.

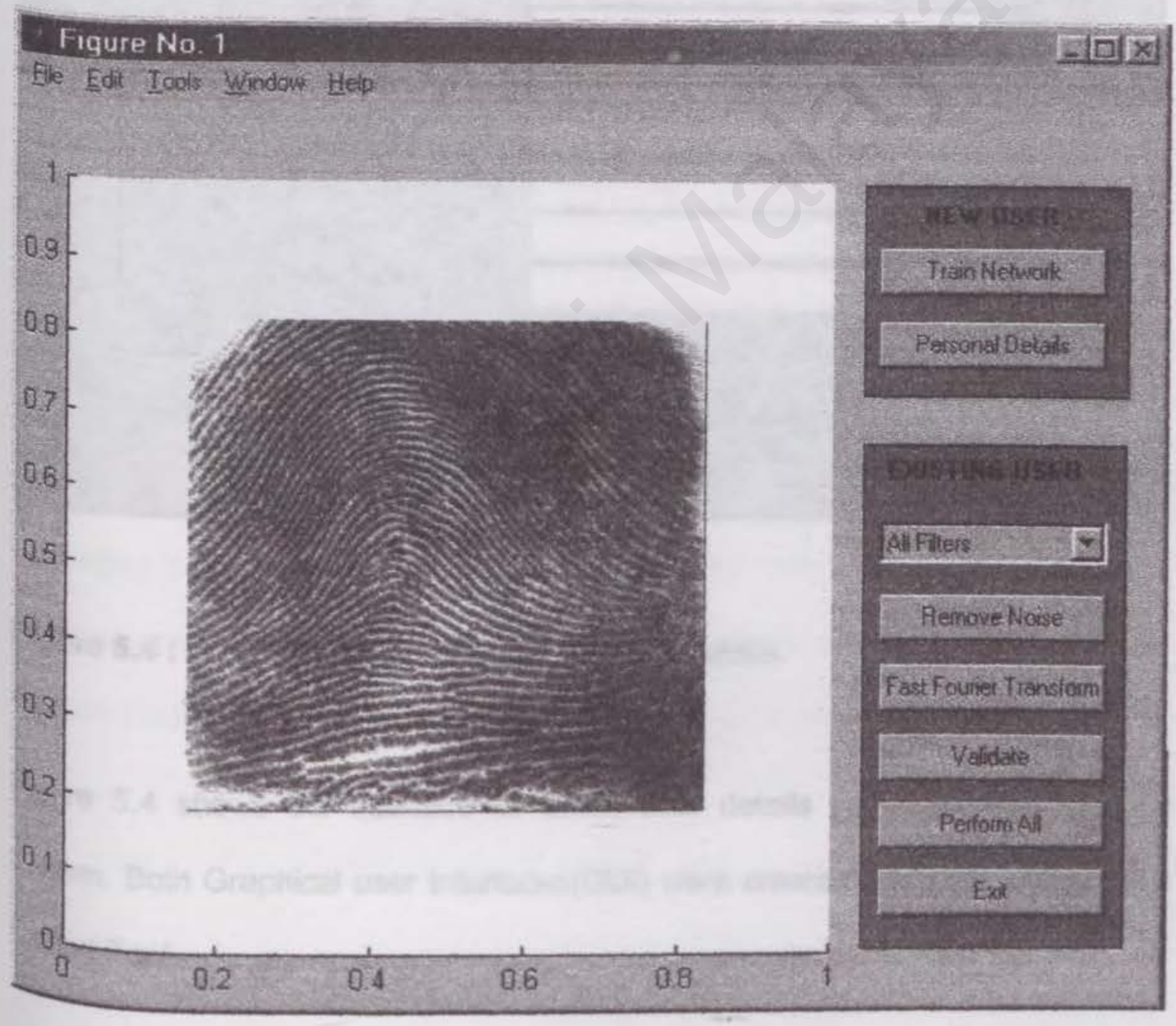


Figure 5.3 : Proposed main interface of system

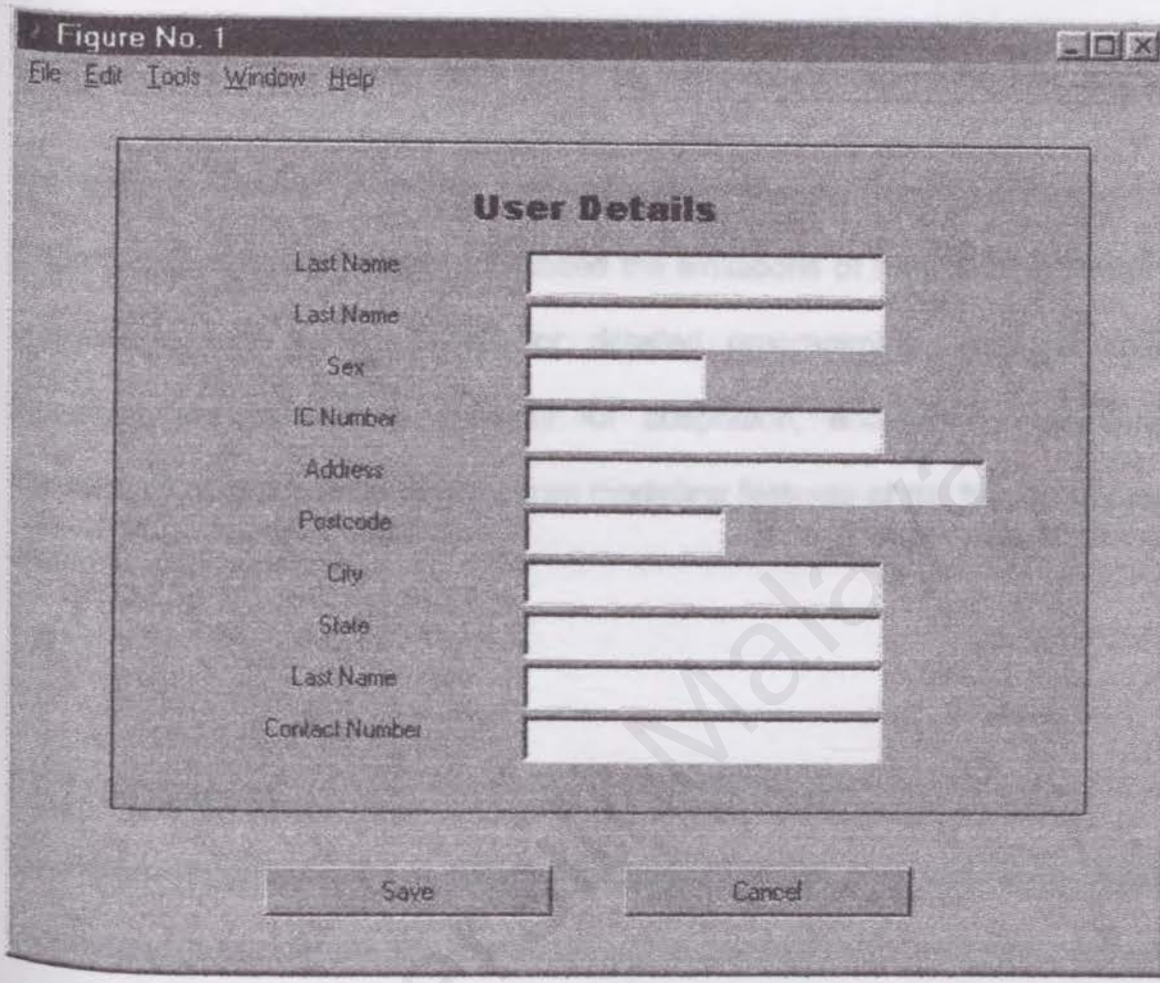


Figure 5.4 : Proposed screen for entering user details.

Figure 5.4 shows the interface of which user details are registered into the system. Both Graphical user Interfaces(GUI) were created using the Matlab GUI Layout Tool.

5.6 Reasons for using Neural Networks

The general interest in neural networks today arises from their fascinating properties, which enable them to exceed the limitations of traditional information processing, such as the need for detailed programming. Amongst these properties are parallelism, capacity for adaptation, and use of distributed memory, all of which arise directly from modelling features of the human nervous system.

5.6.1 Parallelism

Parallelism is fundamental in the architecture of neural networks when sets of elementary units operate simultaneously. This parallelism in data processing is interesting because of the limitations of sequential methods of processing large problems needing a large quantity of data. Parallel processing greatly increased speed of calculation and processing.

5.6.2 Capacity for Adaptation / Learning

This property of neural networks is their ability to learn, which allows networks to take account of new constraints or new data as they arise. Furthermore, it appears in certain networks by their capacity to self-organize, ensuring their stability as dynamic systems. This capacity for adaptation is particularly relevant for problems, which change from time to time. This needs to take account of situations which are not yet known in order to resolve problems. This may mean that the network is able to take account of a change in the problem that it is solving, or that it may learn to resolve a new problem.

5.6.3 Memory

In neural networks 'memory' corresponds to an activation map of the neurons. This map is in some ways a coding of facts that are stored. Memory is thus distributed over many units, giving it resistance to noise. In the first place, the loss of one individual component does not necessarily cause the loss of a stored data item. This is different from the case of a traditional computer, in which individual data is stored in individual memory units, and in which the loss of one memory unit causes its data to be lost permanently. In a neural network the destruction of one memory unit only marginally changes the activation map of the neurons. Secondly, when a piece of knowledge corresponds to one piece of data stored in a particular place, the problem of managing the full set of knowledge arises. In order to find or to use one particular fact, it is necessary to know precisely its address or its contents. This technique cannot therefore take account of noisy data and pre-processing of data must therefore be used to eliminate the noise. This limitation is overcome in distributed memories such as neural networks, in which it is possible to start with noisy data and to make the correct data appear from the network's activation map without noise.

5.6.4 Capacity for Generalization

This capability is crucial. Its importance has been shown in recent years by the difficulty of acquiring rules for expert systems. Many problems are solved by experts in a more or less intuitive manner, making it very difficult to state explicitly the knowledge base and the rules which are necessary for its exploitation. It is therefore highly significant to consider a system which may learn the rules simply from a set of examples, or which may learn to mimic behaviour, allowing the problem itself to be solved.

5.6.5 Ease of Construction

Computer simulation of a neural network for a small example is simple and requires only a short development time. For more complex applications, simulators or accelerator cards have proved useful. Some simulators are now extremely easy to use.

5.6.6 Other Solutions

There are many other approaches to fingerprint recognition that uses direct pattern recognition and feature recognition, but these traditional approaches faces many limitations. You can develop using traditional programming algorithm, however, it will include a lot of if , else, then, logic condition. While the program grow larger, it will difficult to control. Another problem is, it is not efficient. See table below for comparison:

Traditional algorithm	*Neural networks*
Distributive memory only	Both associated and parallel distributive memory
Computer is rendered useless by even a small amount of damage to memory	Neural networks are fault tolerant because information is distributed throughout the neural network system
Incomplete output produces no output	Incomplete input produces reasonable output results
Formalized structured programming is required	Neural networks are self organized

Table 5.1 : Comparison of traditional algorithms and neural networks

Chapter 6

System Implementation and Testing

6.1 Objective

The objectives of implementation and testing are :-

- To test the feasibility of feature extraction filters applied in fingerprints.
- To locate errors, especially programming errors for correction.
- To assure that the system meets the specifications and requirements for its intended use and performance against response time.
- To review the system for correctness, completeness, reliability and maintainability.

6.2 Implementation and Strategy

The entire system was divided into three main modules. Each of the modules was separately developed and tested. After successful testing and the verification of the accuracy and correctness of each module's functionalities, the three modules were brought together and integrated to form the entire system.

During the integration phase, the system was tested as a whole and this was the time when errors, due to system integration, would be discovered and amended. Errors were mainly due to the inefficiency of feature extraction filters or common programming faults such as typing errors and incorrect passing of variables from file to file. However, the integration is carried out phase by phase to ensure that any error that appeared were easily traced to its source and corrected. System integration testing also tested the conformance of the entire system to its non-functional requirements.

The implementation stage includes the setting up of the environment and tools needed for system development, program coding, fingerprint sampling and modification, testing and debugging for errors for each unit, and system integration and testing.

6.3 Actual Methodology

In this discussion, it intends to comply with the need of a successful recognition algorithm. As the test have been conducted based on the Fast Fourier Transform, the results were not as expected in the proposed methodology. Instead, the results from the tests proved to be negative. Image samples that were slightly rotated or translated differed very far from the original. As such, an alternative methodology was derived and used instead of the originally proposed method.

6.3.1 Image Acquisition

Image acquisition, is the process where an image of the fingerprint is obtained. This is where a fingerprint is stamped onto a piece of paper, and then scanned. Fingerprints that are stamped onto an inkpad (for fingerprints), which is evidently drier than normal inkpads used for stamping. From there, the fingerprints are pressed onto a white piece of paper. A few prints are needed in case of smearing or when clarity is compromised.

The fingerprint image is scanned on 300dpi resolution and saved in TIFF format of a size varying between 250 – 500 pixels, depending on the size of the fingerprint. It is important that when the fingerprint image is scanned, it is scanned in black and white format, and saved only in TIFF, JPEG or BITMAP format. This is for the reason that Matlab only recognizes these three formats and other not so common file types. Therefore, this is a criterion in the scanning of fingerprint images.

6.3.2 Image Pre-processing

In the proposed methodology, the initial image pre-processing steps included three steps, being the median filter (salt and pepper noise), mean filter (averaging filter) and histogram equalization. Applying the filters helps in removing irregularities and noise effects on the image. Impurities such as dust/dirt or excess pressure relating to smearing are considered to be noise. However, these filters have its limitations and removes noise to a certain limit only.

Based on test results, of the three filters considered here, only the median and Gaussian filter proved to be helpful in noise removal, whereas the histogram equalization did not make a positive difference in the removal of fingerprint impurities. Instead, the histogram equalization method made images extreme in lightness or darkness.

The median filter proved visibly helpful in removal of salt and pepper noise effects, as it could be seen that images were cleaner and clearer. It removed noise effects that resulted from scanning impurities. The Gaussian/mean filter, helped in ironing out uneven grounds in the image, as to how the filter is named, it takes the average value of a certain neighbourhood and filters out uniform noise that blurs the image. This resulted in a smoother and less distorted image as compared to the original.

6.3.3 Feature Extraction

In the proposed methodology, the feature extraction process involves the implementation of the Fast Fourier Transform (FFT) algorithm on the fingerprint image. However, as mentioned earlier the results proved negative and the Fourier Transform method was deserted.

In alternative to the Fourier Transform method, a patch-matching algorithm was derived and tested on the fingerprints. Initial results proved rather positive in the recognition of artificially rotated and translated fingerprint samples. These artificially modified samples are that of a same image, which are modified using an image-processing tool.

The windowing method works such that for each new user that is added to the database, an initial fingerprint sample is taken and its path saved. This sample will be the basis of recognition of all other fingerprints that are to be verified later.

How the algorithm works is simplified in steps as follows :

1. Original fingerprint window is slid across the test fingerprint until the fingerprint window matched well (using squared differences in grey levels).
2. The window images grey levels are fed into the neural network for training and save results based on each of the individual fingerprints.

(a) Original print (b) Test print from the same person (c) Print from another person

Figure 6.1 : Example of feature matching. A feature is just a small square taken from a fingerprint image (shaded). Using the matching algorithm we can locate the same feature from an original print(a) on another print from the same person(b). If we try to locate the feature on a print from another person(c), the match won't be as good.

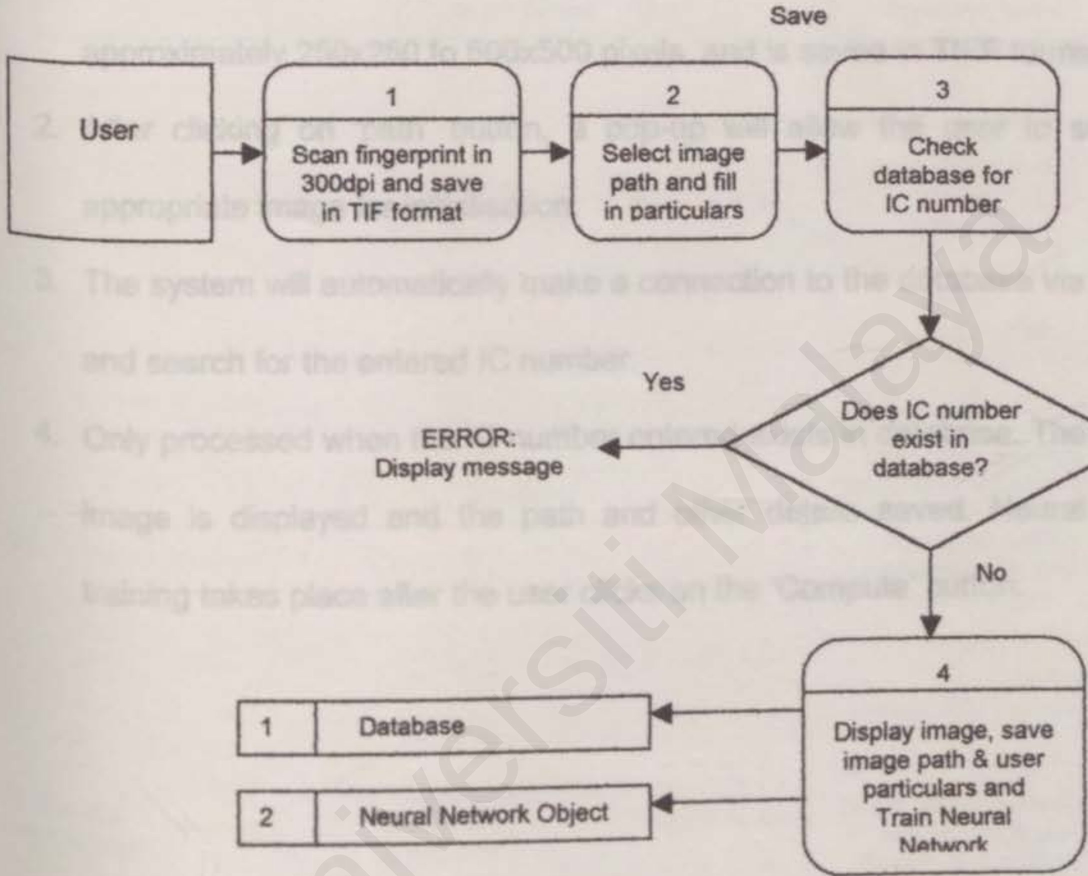
Two filters are used to accelerate the recognition algorithm. The first filter checks a 10x10 window for a minimum percentage of similarity in the image. If it doesn't pass the first filter, it will then proceed to the next column in the specified search window. In any case along the way where the first filter fulfils the minimum threshold in the 10x10 filter window, it will proceed to check the larger 64x64 window, which is the second level filter.

6.3.4 Neural Networks

In the actual methodology the use of neural networks does not differ from the proposed. Neural networks are used to look at the fingerprint features as a whole, rather than separately. In this study, a back propagation training function is used for the training process, where weights are modified based on the targeted output. The inputs of the neural network are that of a binary form of the image, which makes the neurons easier to train and still work efficiently in the recognition of fingerprints. However, the reliability and function of neural networks depends entirely on its inputs, which come from the feature extraction algorithm used. The strongpoint for choosing neural networks as opposed to other approaches is discussed in Section 5.6.

6.3.5 System Flow Diagram

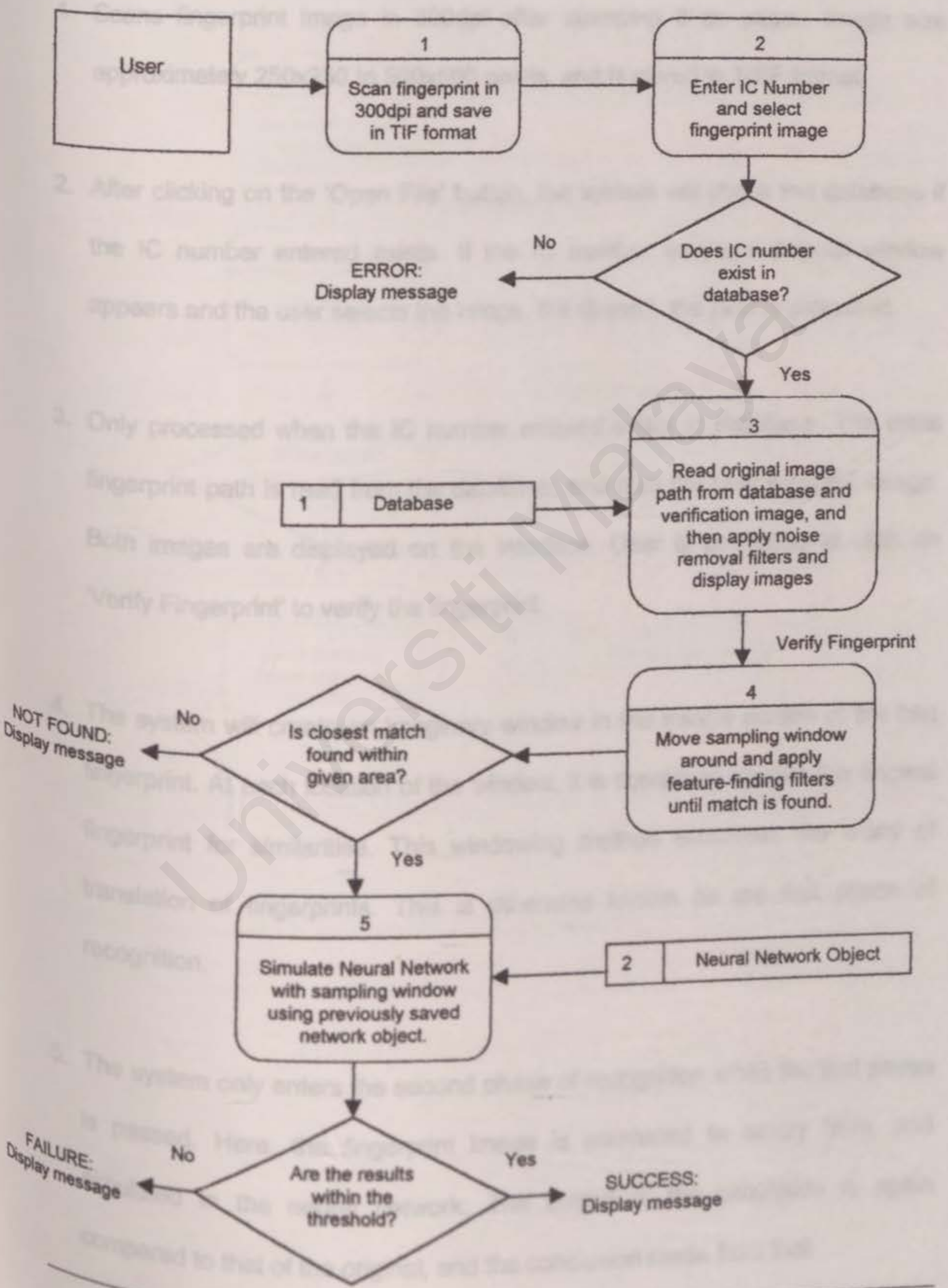
6.3.5.1 Adding New Users



Process : Adding New Users

1. Scans fingerprint image in 300dpi after stamping it on paper. Image size approximately 250x250 to 500x500 pixels, and is saved in TIFF format.
2. After clicking on 'path' button, a pop-up will allow the user to select the appropriate image for initialisation.
3. The system will automatically make a connection to the database via the DSN and search for the entered IC number.
4. Only processed when the IC number entered exists in database. The selected image is displayed and the path and other details saved. Neural network training takes place after the user clicks on the 'Compute' button.

6.3.5.2 Verifying Fingerprints



Process : Verifying Fingerprints

1. Scans fingerprint image in 300dpi after stamping it on paper. Image size approximately 250x250 to 500x500 pixels, and is saved in TIFF format.

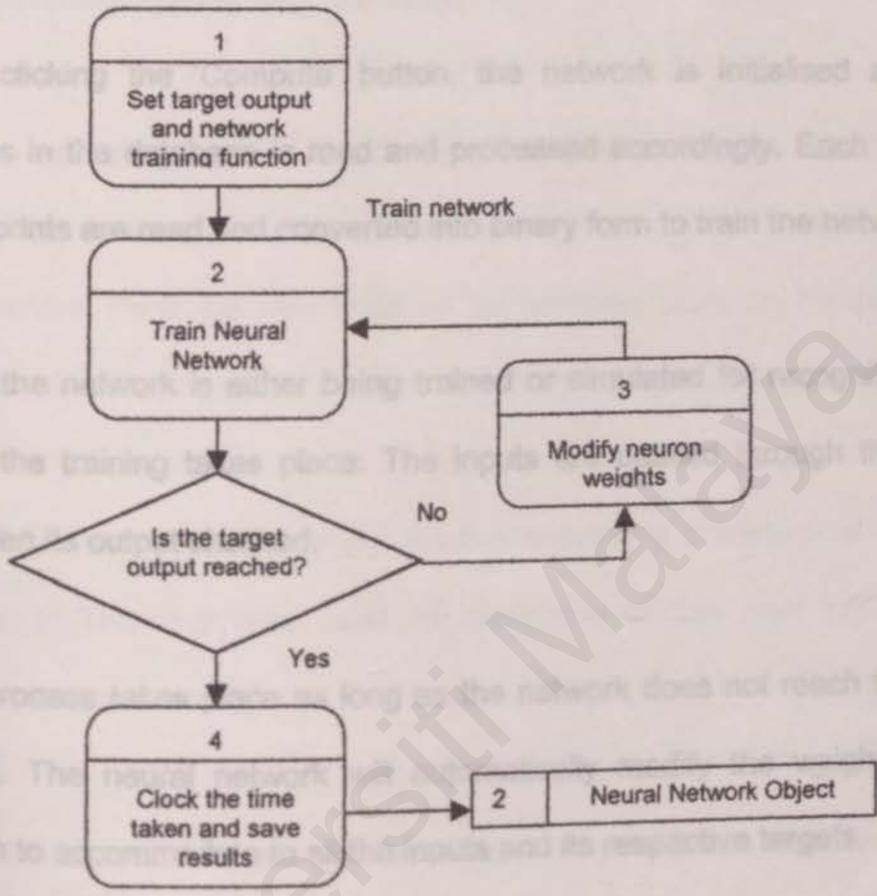
2. After clicking on the 'Open File' button, the system will check the database if the IC number entered exists. If the IC number exists, a pop-up window appears and the user selects the image. If it doesn't, the user is prompted.

3. Only processed when the IC number entered exists in database. The initial fingerprint path is read from the database, and also the user selected image. Both images are displayed on the interface. User is prompted to click on 'Verify Fingerprint' to verify the fingerprint.

4. The system will create an imaginary window in the middle portion of the test fingerprint. At each location of the window, it is compared against the original fingerprint for similarities. This windowing method eliminates the worry of translation of fingerprints. This is otherwise known as the first phase of recognition.

5. The system only enters the second phase of recognition when the first phase is passed. Here, the fingerprint image is converted to binary form, and simulated in the neural network. The output of the simulation is again compared to that of the original, and the conclusion made from that.

6.3.5.3 Training Neural Networks



Process : Training Neural Networks

1. After clicking the 'Compute' button, the network is initialised and all the images in the database is read and processed accordingly. Each individual's fingerprints are read and converted into binary form to train the network.

2. Here, the network is either being trained or simulated for recognition. In this DFD, the training takes place. The inputs are passed through the neurons and then its output checked.

3. This process takes place as long as the network does not reach the desired output. The neural network will automatically modify the weights of each neuron to accommodate to all the inputs and its respective targets.

4. This is the system-programmed response once the target is reached. The network object is saved for simulation of future inputs. Simulation outputs from the network object will help determine which individual the print is from.

6.4 System Setup and Specifications

6.4.1 Operating System and Software

The operating system Windows NT 4.0 was installed onto the server machine. This process involved formatting the disk to NT File System (NTFS) format. Service Pack 4.0 was installed for patching bugs on the previous NT version, and enabling optimum performance from the operating system.

Matlab v5.3.1 was installed to set up the necessary environment for system development. This tool was used for designing of the user interfaces and program coding and testing.

Microsoft's Access 2000 was set up for data storing. With the database server up and running, the overall database structure for the system, called FPRINT, was created as the first step to begin implementation of the database into the system.

In order for the application to recognize the database, a connection has to be established. The mapping of the database to the application is established by creating a Data Source Name (DSN) through the ODBC (Open Database Connectivity). For the development of the system, the DSN used is called "FINGER". For every access by the application to the database, this DSN would have to be called.

Finally, an image-processing tool was needed. Adobe Photoshop 5.5 was installed for the modification and image manipulation for testing the robustness of the system in recognizing similar images with distortion added to them.

6.4.2 Program Platform and Coding

The coding phase involved transforming the system design into machine-readable and executed form. The coding methodology used in the development of the 'Fingerprint Verification System' began with the testing of the neural network functions and also the Fourier Transform. After the Fourier Transform function proved to be unsuitable for fingerprint feature extraction, other methods and feature extraction filters had to be exhausted in search for a more suitable and robust way of recognizing fingerprints. During the testing phase, a method of direct comparison of fingerprint grey levels was tested and results were somewhat positive. This method was later improvised and tests were conducted over and over again, resulting in a more accurate recognition, but still sensitive to the common irregularities that comes with fingerprints.

After the initial analyses and tests conducted, the designs of the user interfaces and also integration of database functions into the system was carried out. Program scripts for user interfaces were developed using Matlab's inbuilt

GUI Layout Tool, a flexible tool for building, testing and deploying Graphical User Interfaces within Matlab codes. Database functions and connections were integrated into the system using predefined functions in Matlab, connected to Microsoft Access. Matlab has a many uses, including:

- A platform for testing mathematical, graphical and sound based functions.
- Neural Network toolbox for implementation and simulation.
- Graphical User Interfaces (GUI) layout tool for creating user interface.
- Database functions for database connection and integration.

6.5 System Testing

6.5.1 General

Testing of specific programs and subsystems is essential to quality assurance. In this project, the testing done differs slightly from the conventional testing methods. For the norm, the methodology used is bound to be successful with minor changes or upgrades to it. Here, however, testing is done in research of a more basic alternative method of implementation.

In general, the proposed methodology is using the Fast Fourier Transform for feature extraction of fingerprints, which is eventually substituted with a patch-matching algorithm. This is for the reason that it the Fourier Transform used was too sensitive to minor changes in fingerprint samples, such as rotation and over/under-inking. Being that, an alternative to the Fourier Transform, a patch-matching algorithm is used for feature extraction. The only setback is the success rate of the patch-matching algorithm, which is approximately 50%, based on 10 different test samples.

The *Fingerprint Verification System* has been developed successfully developed after implementing and testing two different algorithms and feature extraction methods. The effectiveness and performance was tested under different conditions in order to meet the requirements of recognition. The minimum requirements that the system must achieve is as stated below :

- Train set

The training set or the initial fingerprint image, which is kept in the database, is very crucial and critical for the system. The first fingerprint image's path is kept in the database together with corresponding details like name, gender and address. Each person is represented by only one picture whereby the most suitable image is selected for the training set by the system user.

- Specification of the images

All images used for this system are scanned using a scanner in 300dpi resolution. As the accuracy of the system is concerned, the images are best selected in these criteria's such as the core position of the fingerprint, the size of the image and format of fingerprint images. A limited degree of rotation and translation cannot be avoided, and is therefore taken into considerations. Effects of scars on fingerprints are also inevitable, and are treated as a part of the fingerprint. The allowable degree of rotation would be ±5° and a translation of maximum 40 pixels from the original location. The threshold for minimum acceptance is a maximum of 20 in grey-level values.

a)initial sample b)slightly over-inked sample c)slightly rotated sample

Figure 6.2 : Variations of fingerprint images recognized by the system

a)initial sample b)over-inked sample c)over-rotated sample

Figure 6.3 : Variations of fingerprint images not recognized by the system

- Accuracy

Testing the accuracy during recognition is the most crucial task for this undergraduate study. All input fingerprint images are used in the recognition process. The user is prompted with the name and contact number, and also the time taken if it is successfully verified. Fingerprints used for testing the system are of two important scenarios, being the known print and the unknown print.

a) A known fingerprint image

A known fingerprint image is that of the same persons, but taken at a different time and scanned into computer readable format. Whatsoever different images from the same person is classified as known images. Figure 6.2 and 6.3 shows two sets of different images from the same person.

b) An unknown fingerprint image

An unknown fingerprint image would the fingerprint of a different person used when the valid IC number is entered, but not the valid fingerprint image. This means the fingerprint image does not match the IC number entered, and is therefore classified as an unknown image. In cases of false rejection or false acceptance, users need to determine with his or her own vision to whether or not it is the same fingerprint.

6.5.2 Image Sample Types

In the experiments carried out, three types of images were used to test the matching algorithms. All three of the sample types were obtained using the inkpad method described. The three sampling types were used in the testing of the aforementioned algorithms.

- Sample A

The test samples used are artificially modified using an image-processing tool (Adobe Photoshop). The tool is used to manipulate the images to instil fingerprint irregularities such as rotation and translation.

- Sample B

In these samples the fingerprint images are obtained using a normal inkpad of which its ink still appears wet and sticks to the finger after stamping. This ink, as tested is not waterproof and the stamp pad is rather wet. For that reason, problems of over-inking and smearing lies in the fingerprint samples taken.

- Sample C

This sampling method utilizes the Eazyprint™, an American produced device that is similar to the device used in Sample B. Three fingerprint samples were taken using this device. Of the flaws mentioned in using method B above, this device still needs the user to stamp the finger on the pad and transfer it to paper. However, the surface is dry and its ink waterproof. Due to its dry surface, the inking is more even and over-inking problems are less existent. An added feature for the user is that the ink is miraculously permanent on paper, but wipes off the finger with just rubbing it off with another finger. Samples are as seen in Figure 6.4.

Figure 6.4 : Fingerprint sample taken using the Eazyprint™ stamp pad

6.5.3 Experiments

In order to reach the conclusion of this study, two generally separate studies and tests were conducted. The first was on the algorithm for fingerprint feature extraction and the other for neural networks testing. The system was tested throughout the development stage. Three experiments have been conducted in this undergraduate study. The purpose of these experiments is to ensure the system works as expected. The experiments uses different fingerprint images for the training set and verification images. Three experiments, Experiment A (Ex-A), Experiment B (Ex-B) and Experiment C (Ex-C) are discussed in the coming topic.

- Experiment A

In this experiment, the basic recognition of fingerprints is tested based on the feature extraction method using the Fourier Transform. An important thing here is that the experiment is carried out using only the feature extraction method without the use of neural networks. Tests are done based on the irregularities such as rotation, translation and also over-inking. The inbuilt Fourier Transform function in Matlab is used and tested on the above mentioned sample types.

- Experiment B

This experiment is rather similar to Experiment A, but uses the patch-matching algorithm instead of the Fourier Transform. In the patch-matching algorithm, there three levels of threshold used. The first and second level filters are used for feature comparison, whereas the third level threshold is in the allowed percentage of minimum acceptance. It also tests the squared differences and normal threshold differences. Similar to Experiment A, the fingerprints are tested against all the normal irregularities which exist in fingerprint sampling.

- Experiment C

This experiment was tested after the first stage of development had been passed. As the Fourier Transform method did not work, the Patch-matching algorithm is used together with neural networks. The ability of the neural network is put to test here. Neural network are built on layers. Basically, there are 2 layers, the input and the output. Additional layers come in the middle of these two, and are known as hidden layers. The difference in the performance and function of using different number of layers in the neural networks are tested.

6.5.4 Test Results

The results vary largely from the types of samples used and also in which algorithm is used. Below is a table that consists of information regarding the accuracy and performance of the algorithms against the samples used.

Experiment / Criteria	Experiment A (Fourier Transform)	Experiment B (Patch-matching algorithm)	Experiment C (Neural Network layers)
Sample A (artificially modified 14 samples)	Recognizes only its own image with translation problems	100% recognition (max ±5° rotation)	Results > with 3 compared to 2 layers.
Sample B (Ordinary inkpad 2 prints)	Recognizes only its own image with translation problems	40% recognition	Results > with 3 compared to 2 layers.
Sample C (Eazyprint™ inkpad 2 prints)	Recognizes only its own image with translation problems	80% recognition	Results > with 3 compared to 2 layers.
Unknown images	0% acceptance	0% acceptance	0% acceptance

Table 6.1 : Recognition results for experiments against the sample types.

Description of the Table 6.1

Experiment A

In this experiment, the results showed very sensitive results to minor changes such as fingerprint rotation and inking effects. However, due to the algorithms window-search nature, translation problem were overcome. However, in all the strictness of the system, unknown images were also rejected, meaning there were no false acceptances, but incidentally, there are many false rejections in using the Fourier Transform. When used against the artificially modified prints, it could match all extents of translation as long as it was using the same image. However, when the image was rotated even as little as 1°, the results were not favorable to the algorithm. Results were showed more sensitivity to inking differences when tested against Sample B and C. What would be noticeably similar to the humans would be different using the Fourier Transform. Test results showed that the Fourier Transform is in fact too sensitive to be used against fingerprints due to the many irregularities it contains.

Experiment B

Experiment B was exhaustedly tested on only after vigorous test had been conducted on the Fourier Transform. Using the patch-matching algorithm, the recognition proved to be noticeably more robust to effects of rotation and inking.

When tests were initially conducted on Sample A, the results were very encouraging. It showed 100% accuracy up to a maximum of ±5° rotation. However, the threshold levels were tested out in many ways, to the extent of using 3 levels of threshold to accelerate the recognition process and also to enable higher accuracy. Each fingerprint was tested against 5 different individuals fingerprints. Based on the test results, the system showed no false acceptances of fingerprints. Using this stamping method, the threshold could be increased by 10% and also show a more significant improvement in the recognition. Using Sample C (Eazyprint™), which produces more visibly similar samples, the recognition seemed to have doubled in percentage due to the inking irregularities in using the ordinary inkpad for samples.

Experiment C

Experiment C is rather different in nature as compared to Experiment A and B. The nature of experiment C was to test the functionality of the number of layers in neural networks. As the tests were conducted based on the number of layers used in the training process, it generally showed that the more layers and neurons used, the longer it took to train the network, but gave it the advantage of training more fingerprint samples more effectively as compared to less neurons.

Layer combination (input : hidden : output) layers		Training time (seconds)
2 layers	(15 : n)	70.80
	(30 : n)	98.75
3 layers	(30 : 90 : n)	78.76
	(30 : 120 : n)	119.24
4 layers	(30 : 60 : 60 : n)	157.31
	(30 : 90 : 90 : n)	199.60

*tested on a P3 550, 128MB RAM computer, (n based on training sets. In this test, n used is 4)

Table 6.2 : Neural network training time

Based on the test results shown, between 2 and 3 layers, the 3-layer network showed not only a shorter training time but also an advantage in more accurate results. Although a 2-layer network takes a shorter time to train, the effectiveness of the network is compromised. This is however solved by using 3-layer networks, where the training time is slightly longer, but it has an advantage in more accurate simulation results. Between 3 and 4 layer networks, it took longer to train a 4-layer network, and the difference in results is minimal. This is a losing trade-off in terms of performance. The number of neurons in the hidden layer showed that training time increased along with the increase in the number of neurons in the hidden layer. Once again, this additional layer in a 4-layer network does not show significant difference in simulation results.

Chapter 7

System Evaluation and Conclusion

7.1 System Evaluation

During the system evaluation phase, the evaluation on the system developed has been carried out. A series of problems occurs when developing the system. Optimistically, these problems do not impact the transaction of the system development. Time and technique is involved to the solution of the problems.

In this study, the Fast Fourier Transform was initially applied and thought to work well with fingerprints. On the contrary, it proved to be too sensitive for the irregularities and changes that fingerprint samples are subjected to. In the Fast Fourier Transform, it takes the specified image sample, and performs cumulative calculations on it, and results in a Fourier coefficient. Due to irregularities of over/under-inking and rotation, the Fourier coefficients change with each different sample. Viewing these results after vigorous tests, the Fourier Transform method was aborted for a different approach, the patch-matching algorithm. Using the patch-matching algorithm, the fingerprint recognition proved to be more robust and efficient as compared to the Fourier Transform. Using the patch-matching algorithm, the algorithm is translation invariant and relatively rotation and grey-level invariant to a certain extent.

Based on tests conducted on the artificial samples, results were 100% positive, up to ±5° in angle and also translation invariant with help from the windowing method used. This was tested on 5 different fingerprint samples and their respective modified samples in 'Experiment A' discussed in the earlier chapter. All tests were positive and showed no false acceptance or rejects.

The patch-matching algorithm was later tested on real fingerprint samples taken using the inkpad method and scanned into computer readable format. A total of 10 samples of each fingerprint were taken. A total of 4 samples were taken of different people. Of all the 4 samples, both two were taken using an ordinary stamp pad, which showed 40% acceptance and the other 2 fingerprints taken using the Eazyprint™ stamp pad showed 80% acceptance. Tests also showed varying results based on the initial fingerprint used when registering a new user. Using the Eazyprint™ pad, the threshold level could be increased by 10%, which also showed a significant increase in the recognition. This goes to show that the clearer the input images are, the better the recognition process.

There were an acceptable percentage of correctly verified prints using the patch-matching algorithm derived (depending on the grey level). In the test results, the thumbprints were less consistent in the inking, where as the prints of the index and middle finger was more evenly spread and showed less effects of over/under-inking. Thus, there was an average of 60% false rejects (reject legitimate fingerprints) when thumbprints were verified, and approximately 20% false rejects when index and middle finger prints were used. This is for the reason that the thumb is harder to control in terms of applying pressure when rolling the finger on paper.

This project studies the alternative methods of fingerprint feature extraction as opposed to the much-used minutiae based feature extraction methods. The Fourier Transform and other image processing methods were examined their suitability and applicability in feature extraction for fingerprints. From the results of this project study, it is concluded that the Fourier Transform method is not suitable for fingerprint feature extraction as it is too sensitive to the irregularities of over-inking, rotation and translation. The other feature extraction method used, which is the squared differences of grey levels between two fingerprints. However, further research and testing need to be done in addition to this method. Due to the time limit imposed, only two methods were experimented, with only the latter giving positive results.

Due to time pressure, the algorithm could only be tested on 5 fingerprint samples that were collected (tested against each of the fingerprint samples of the basic 5 fingerprints). There was no false acceptance.

7.2 Problems and Solutions

Many problems were faced throughout the research and development of the Fingerprint Verification System. Most of the problems were solved, leaving the exceptional few, which remains under tedious research and study. The experiences and knowledge gained in this study while finding solutions to problems and time management skills were invaluable.

7.2.1 Inappropriate and inaccurate feature finding filters/methods

The feature finding filters used are as important as a password is to a username as it is in this context. The feature finding filters are basically the way that each fingerprint's unique identity is captured and eventually used for recognition. Firstly, the Fourier Transform function was studied and applied. Later, when results were negative, a patch-matching algorithm was derived and applied onto the fingerprints. Vigorous testing was conducted on the prints and results compared accordingly.

7.2.2 Lack of knowledge about Matlab logic and GUI programming

Matlab (short for Matrix Laboratory) is a programming language largely used for mathematical, graphical/imaging and sound processing purposes. Matlab was chosen for the reason that it offers a large collection of in-built functions which proved to e very useful and practical in this simulation of fingerprint recognition integrating the use of neural networks. As Matlab includes a neural network toolbox, the implementation of neural networks coupled with image processing functions was made relatively easier during the simulation process.

7.2.3 Acquiring information on various methods of fingerprint recognition through IEEE libraries and resources

The Institute of Electrical and Electronic Engineers (IEEE) is an international organization of standards in electronic and electrical technology. The IEEE electronic library was accessed through the Engineering Faculty of University Malaya via its CD-ROM collection of references. Of the references and related material found, most of them were more of an introduction to the study, rather than in depth details of it. This was in a way a setback to the study of the algorithms used in fingerprint recognition as most of them mentioned only the technology used, but not how it works. However, the resources were helpful in assisting the undergraduate study.

7.2.4 Lack of resources and time for testing feature extraction methods

As technology moves forward and improves with time, things seem to work better and also more automated. In this study, the technology refers to the existence of biometric devices for fingerprint scanning, and also advanced algorithms in fingerprint matching. This however remains a patent of the company or costly affair to obtain. The fingerprint-scanning device was not available in the faculty and thus the conventional method of fingerprint scanning via an inkpad and paper had to be used to its maximum ability.

7.2.5 Insufficient reference in utilization of neural network training functions and simulation.

Neural networks are still in its infancy in Malaysia and relatively fresh to the study. Therefore, extra time had to be taken in revising the ability of neural networks and also its functionality. On the whole, neural networks are complicated and are still vastly growing. The subjectivity of neural networks proved a complex study in testing and implementation.

7.2.6 Lengthy run time for feature extraction filters and neural networks

The time taken for testing each of the feature extraction filters and the appropriate neural network training function which doesn't take too long to train or reach its training goal. Each of the training would take different times to train and simulate, depending on the CPU used and the size of the training input. The larger the input, the longer it takes. Therefore, a suitable training function coupled with the most effective number of neurons and layers had to be analysed.

7.3 Strengths of System

7.3.1 Fingerprint Class Invariant

Unlike certain fingerprint recognition system, their system is limited to certain types of fingerprint classes only. Referring to certain minutiae based fingerprint recognition algorithms; their system is restricted to whorl, loop and certain mixtures of fingerprint classes only, leaving out the arch type fingerprint. This is for the reason that the arch type fingerprints does not have a curve/loop in the fingerprint pattern (refer 2.2.2 : Fingerprint Characteristics). The absence of this feature makes the reference point of the fingerprint hard to determine. In this Fingerprint Verification System however, the fingerprints are recognised based on a patch-matching algorithm, in which the images patterns are compared. This gives the system more leniency and openness to all types of fingerprints, and can be used also as a basis in the application to other pattern recognition algorithms.

7.3.2 Translation and Rotation Invariant

Fingerprint samples would not be the same each time they are stamped. They are bound to be a small degree of rotation and translation from which the first fingerprint sample was collected. Therefore, these two irregularities has to be taken into consideration when the system was developed. Therefore, the algorithm uses a window searching method where a fixed sized 'window' is moved around and at each location. The window-sample is compared to the original fingerprint, where the grey levels are compared for similarities. This method gives the system the efficiency of functioning as well as leaving out the problem of translated fingerprints. Similarly, it is also rotation invariant up to ±5 degrees, based on test results.

7.3.3 User-friendly Interface

The Fingerprint Verification System offers a user-friendly interface, which is direct and provides the necessary functions needed. The interface developed provides a status bar and also a help button at which the user can click for more information on how to utilize the system better. The colour used is also a standard grey colour for more professional look.

7.3.4 Status Bar as System Guideline

The status bar serves as a tour guide, at which the user is prompted on the next step in case of any unfamiliarity in using the system. The status bar appears in both the interfaces of the system, the main menu and also the new user menu. The words in the status bar are highlighted in red and appear at the bottom left part of the window.

7.4 Limitations of System

7.4.1 Dependency on Matlab Software

The software is built on a Matlab platform, and thus dependant on Matlab to function. This is however a study into an alternative feature extraction method using image processing tools, and Matlab provides the convenience of these tools. Matlab also provides the simulation of neural networks, which is another reason why it was chosen over other programming languages. Despite all the advantages, the platform dependency and the slow runtime of Matlab is a slight disadvantage to the simulation exercise.

7.4.2 Sensitive Threshold Levels

This is where the threshold level determines how 'strict' the system is in accepting fingerprint images as a verified fingerprint. If the threshold level is low (strict), the acceptability and false acceptance decreases, but security increases. However, if the threshold level is high (less strict), false acceptance ratio increases and the reliability and security of the system decreases. The sensitivity of the threshold called for more tests to be conducted in order to satisfy a general collection of various fingerprints.

7.4.3 Sensitivity to Inking

When fingerprint samples are taken, they are best sampled using biometric fingerprint devices, which are not available due to the high cost of it. Thus, the conventional method of an inkpad is used. This limitation opens the system to flaws such as over-inking/under-inking and smearing of ink. The issue of inking was largely the reason for the inaccuracy and sensitivity of recognition, and not so much the problem of translation and rotation.

7.4.4 Slow Processing Time

The Matlab platform showed decreased performance in the simulation of the feature extraction algorithm and mainly in the training of neural networks. As time is one of the issues to be taken into consideration when developing the system, the holdback of a faster platform was inevitable due to the many dependencies on Matlab such as the image processing and neural network tools. However, a system built on C/C++ would be significantly faster although it is much more complicated to develop. As the system is only a simulation, it would be wiser to simulate the algorithms first on Matlab, and later developed on a more suitable platform once found to be successful.

7.5 Future Enhancements

7.5.1 Find Most Effective Initial Print

Currently all features come from the very first fingerprint in the database for the particular person. Given a feature, it could be matched against all prints from the same person, and then take the matching parts and average them out to create a feature which should match each print better. This procedure could be repeated recursively to obtain an even better, and more even feature.

7.5.2 Different feature searching window for optimal performance

All fingerprints in the database use the same feature-searching window. The size of the feature window could vary according to fingerprints, and tests can be conducted to find out the optimal. Perhaps the optimal feature size differs from person to person, and also perhaps a larger window would make recognition more accurate with less percentages of false acceptances or rejection.

7.5.3 Adding more feature finding filters

The results of the patch-matching algorithm were not 100% successful but showed rather positive results when tested on a small number of fingerprints. This could be due to the use of only one feature extraction filter. However, adding more feature extraction filters and also improvising on the current algorithm could lead to a positive response. Additional filters which filter other aspects of the individual fingerprints, which cannot be concluded before conducting tests, once added, may very well accelerate the recognition process and also reduce the recognition time.

7.5.4 Testing on a larger database of fingerprints

The testing done in this study was carried out based on a database of 5 fingerprints, one of which was artificially modified for testing the maximum degree of rotation the algorithm can accept. Further analysis can be conducted on the algorithm if the system was tested on a larger database of fingerprints. Results could be different from the current test results on 5 fingerprints.

7.5.5 Extensive testing using biometric devices

Due to the irregularities of inking and its problems such as smearing and over/under-inking, the use of electronic biometric devices such as a fingerprint scanner could help much with the testing of simpler and faster algorithms. The use of a biometric scanner would lead to more even fingerprint images and also remove the smearing effects now existent using the inking method.

7.6 Conclusion

The algorithm implemented using the Fourier Transform proved not feasible for fingerprint feature extraction purposes based on tests conducted on the fingerprint samples. It is proved that the Fourier Transform method changed too much with minimum degrees of rotation and minor changes to the fingerprint. It is therefore not recommended that the Fourier Transform method be used for fingerprint feature extraction for recognition purposes.

The system that was designed implemented a very simple algorithm for fingerprint verification. The algorithm is fast enough to be used as a verification system (2 – 10 seconds per match, depending on CPU used), and produces zero false acceptances on the fingerprint samples used. However, the algorithm alone showed false rejection when certain fingerprint samples were compared, even though they were valid fingerprints.

The academic knowledge gained during my course of study did indeed help while analysing and developing the system. Skills such as project planning, programming practices and database was of much assistance. Much experience and knowledge was gained during the study and development of this project. The project enabled me to learn how to plan and schedule between important and less important tasks, according to my capabilities. This project study has impacted me with more mature thinking and tactical means of problem solving.

Appendix A

User Manual

1.1 Introduction

The Fingerprint Verification System can be used for verifying electronic forms that are sent in via e-mail, of which its fingerprint image is attached to. This image is then selected out from the form and placed on the system for verification. The system works as a simulator for testing feature extraction filters to be used with fingerprints. There are total of 12 files for the Fingerprint Recognition using Neural Networks, 10 program code files (.m) and 1 neural network object file (.mat) for the implementation of neural networks (.mat), and 1 Microsoft Access Database file (.mdb).

1.2 Installation

1.2.1 System Requirements

The program is run on Windows 95/98 platform. In order for the program to work correctly, there are minimum requirement that must be satisfied first. The software and tools below must be installed.

- Windows 95/98/ME, Windows NT/2000 Operating System.

  All the source codes are in Matlab format and is only compatible on the Windows platform. Windows is chosen for its wide base of compatibility with many tools and software.

- Matlab 5.3.x for Windows.

  This software is needed to convert the program files into machine code for running. All the source files are written in Matlab format without converting it to an executable file. The simulation and testing has to be done in a Matlab environment.

- Microsoft Access 97/2000 database.

  Used for data saving and retrieval. This tool is used to keep all the necessary information of users. The corresponding image path is kept along with information like name, identity card number, address and contact number.

1.2.2 Installation Procedure

The entire *Fingerprint Verification System* is separated into three portions as stated below :

- The functional files

  These are the Matlab files that run on the Matlab environment. All these files have to put into a directory called "fingerprint" under the Matlab ".../Matlab/work/." directory for the simulation to take part.

- Input images

  These are the fingerprint images in TIF format. These files will be used by the *Fingerprint Verification System* to simulate and test the system for results. The image files are kept in a directory called "prints" which located under the same directory "fingerprint". There are three subdirectories here, being "artificial", "eazyprint", "known" and "unknown". The purpose of these file organization is to reduce the confusion throughout the simulation and testing process. Subdirectory "artificial" keeps the images that were modified using Photoshop for irregularities of rotation and translation. Subdirectory "eazyprint" keeps sample images that were taken using the Eazyprint™ stamp pad. Subdirectory "known" holds fingerprint images from the same person which shows positive recognition in the simulation. Subdirectory "unknown" will keep fingerprint images of the same person and also other persons who are not recognized by the system.

- Database file

  This is a single file in MDB format for Microsoft Access Database. In this undergraduate project, the database file is named "fprintDB.mdb". It is advisable to keep this file under directory "fingerprint".

1.3 Step One (Environment Setup)

Extract zip file and create Data Source Name

Ensure that all the necessary software is installed into the system. This will be the software like Matlab version 5.3.1 and Microsoft Access 97/2000. The first step is to unzip the file called "fingerprint.zip" into the Matlab directory. The Matlab directory is usually referred to as "...\Matlab\work\". The zip file will extract all the necessary components like subdirectories, matlab code files (.m and .mat), fingerprint images and the database file (.mdb).

Once the zip file is extracted, it is necessary to create a Data Source Name (DSN) through the ODBC (Open Database Connectivity) under the Windows Control Panel. The DSN used is called "FINGER", and the reference is made to the database file "fprintDB.mdb" located in the working directory "fingerprint". After this is completed, simulation of the program can be done as expected.

1.4 Step Two (Main Menu)

Running the program for the first time

All the functional files, database file and fingerprint images are located in the working directory "fingerprint". For the first time the program is running, we need to execute the Matlab environment. Following that, we must be in the working directory "fingerprint" to access the files.

>> cd fingerprint

To execute matlab program files, the name of the file needs to be typed at the command line prompt in Matlab. The main file can be executed by typing "fpmain" at the command prompt as seen below.

>> fpmain

This file functions as the main menu of the *Fingerprint Verification System*. Here, the user can opt to verify a fingerprint, recomputed the neural network, or add a new user to the system. Figure A-1 depicted below shows the main system window. It consists of the mentioned three functions and also at the bottom, a help button and an exit/close button.

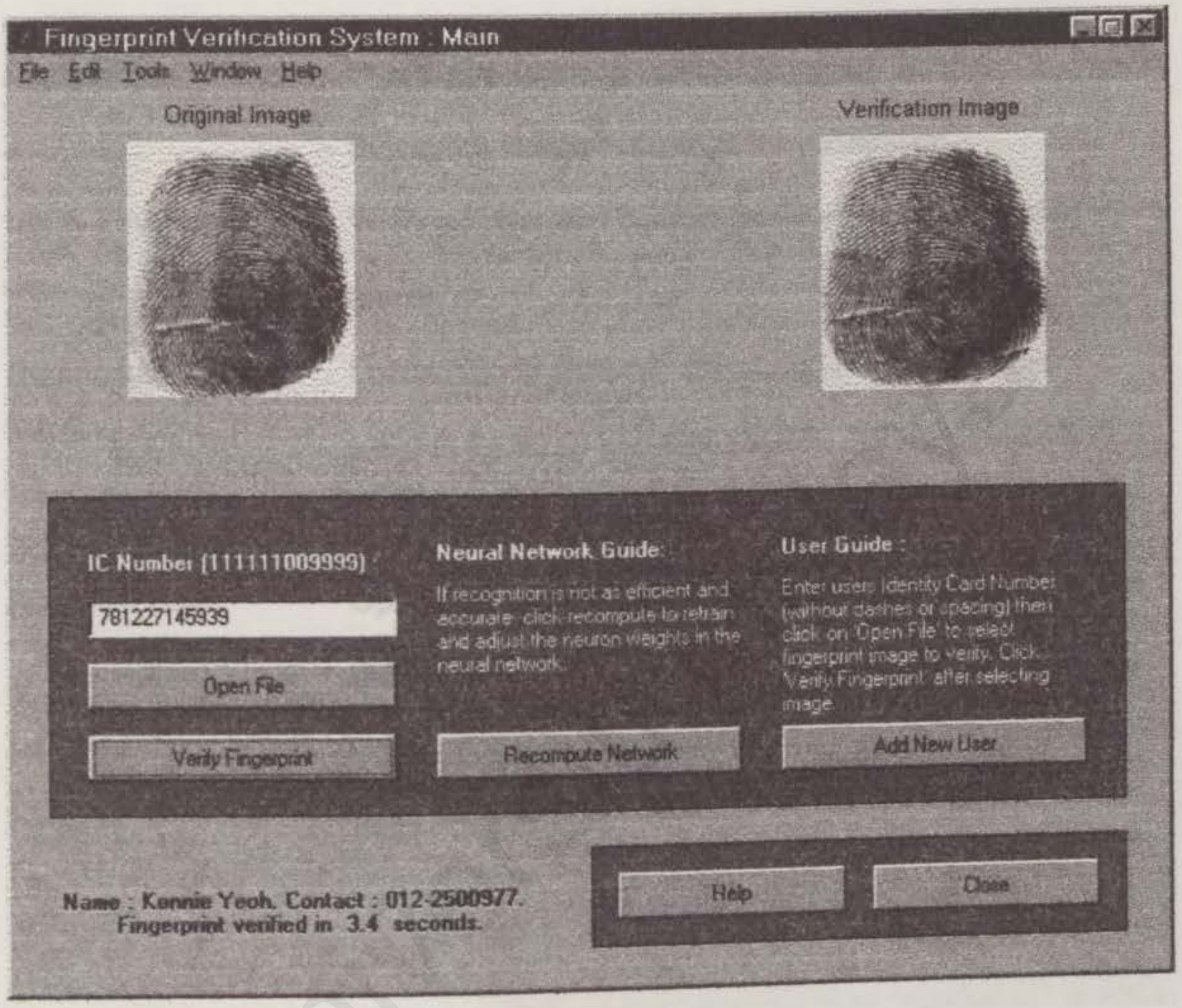


Figure A.1 : Main system interface

<u>Main Menu Buttons</u>

Open File	-	Press after entering valid IC Number. Select the fingerprint image for verification.
Verify Fingerprint	-	Verifies fingerprint against original fingerprint added earlier.
Recompute Network	-	Loads Neural Network object and readjusts neuron weights for more accurate results. Training time depends on the amount of users in the system.
Add New User	-	Opens new window to add new users to system database.
Help	-	Opens this help window.
Close	-	Exits main window.

1.5 Step Three (Adding New Users)

Insert the corresponding fingerprint image to train

Click on the "Add New User" to add new fingerprints to the database. This will lead to the module shown in figure A-2 (New User Menu). New users need to insert a fingerprint image path by clicking on the button named "path". The fingerprint images will be selected from the directory named "train" for the training set. Besides that, other information like name, identity card number, address and others are very crucial during the insertion process. The information will be saved into the database by clicking on the button "Save" as shown as figure A-2. Users will need to repeat the task until there are satisfied with the number of fingerprint images in the training set, being a minimum of two. After adding each user, the user needs to exit the user menu and click on "Add New User" again. When the "Compute" button is activated, it means that the minimum number of users in database is satisfied, and the network can now be trained with adding the new user. Following that, users can return to the Main Menu to verify user fingerprints from directories "known" and "unknown".

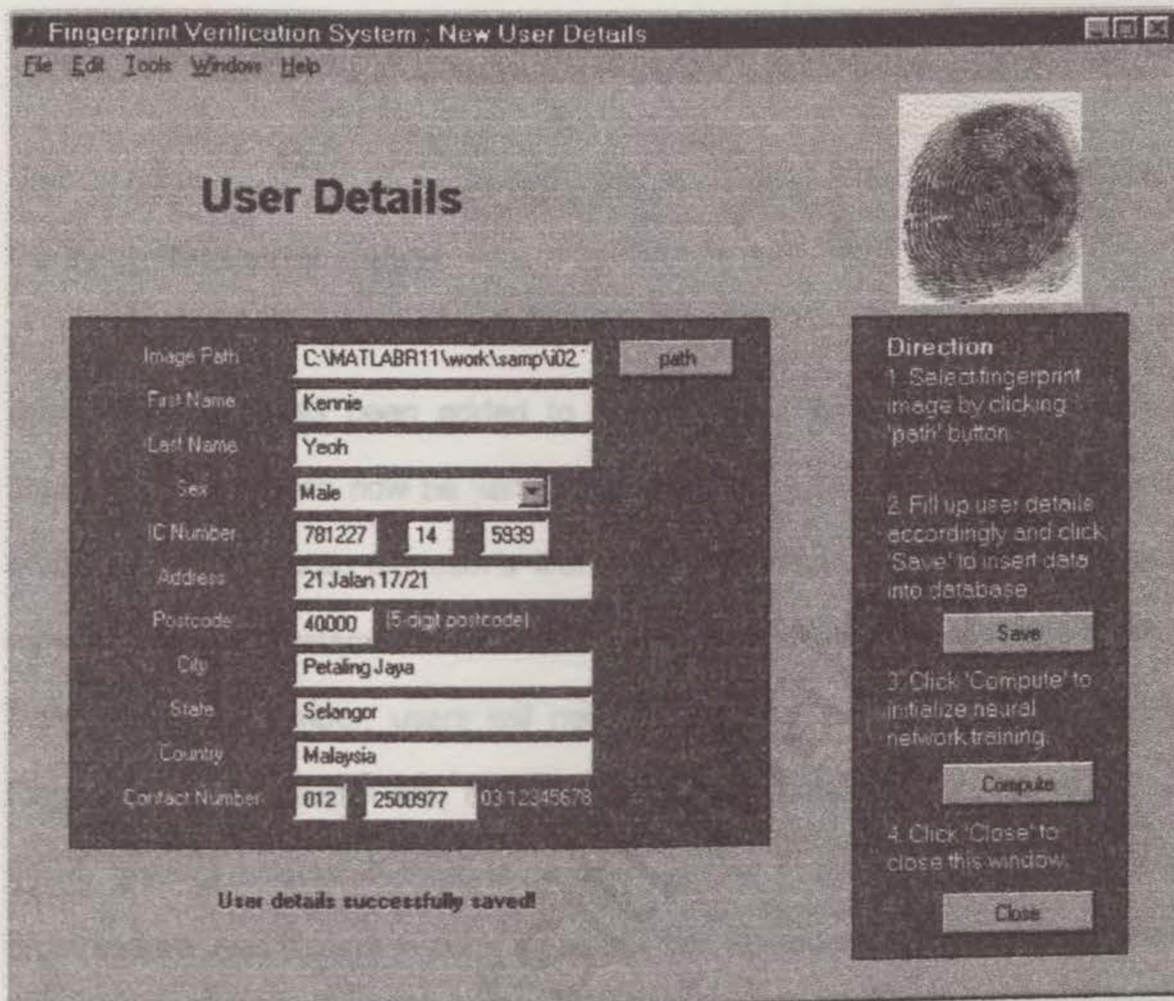


Figure A.2 : Adding New Users Interface

<u>New User Menu Buttons</u>

Path - Select user's fingerprint image, which will be used for verification.

Save - Press after entering ALL fields on the form.

Compute - Only activated when there are 2 or more users existing in the database. It computes and trains neural network for recognition. Training time depends on the amount of users in the system.

Back - Exits new user menu window.

1.6 Step Four (Recognition)

Verifying fingerprint images

After new users has been added to the database and the network trained, fingerprint images can now be verified from the main menu. For the verification process to take place, the user's identity card number has to be known and entered into the system before selecting the image file for recognition. After entering the IC number, users will need to click on the button named "verify Fingerprint" to proceed. Here, a database check is performed on the validity of the IC number entered. If there is no such number in the database, the user is prompted so. And if there is such an IC number, both the original image and the verification image just selected will be displayed on the system interface. Now, the user will need to click on "Verify Fingerprint" to perform the recognition algorithm on the two fingerprints. If the fingerprint is accepted, the user's name, contact, and the time taken to recognize the fingerprint is displayed, and an invalid message otherwise. The threshold level of the recognition is hard-coded in the systems program codes. On the whole, the higher the threshold, the more strict the system is and inevitably higher false rejects, and with a low threshold, a more lenient system with higher false rejects.

Appendix B

Project Schedule

2.1 Project schedule (Gantt Chart)

2.2 Project schedule (Table)

2.1 Project Schedule (Gantt Chart)

A Gantt Chart is used to summarize the transactions and development of the system.

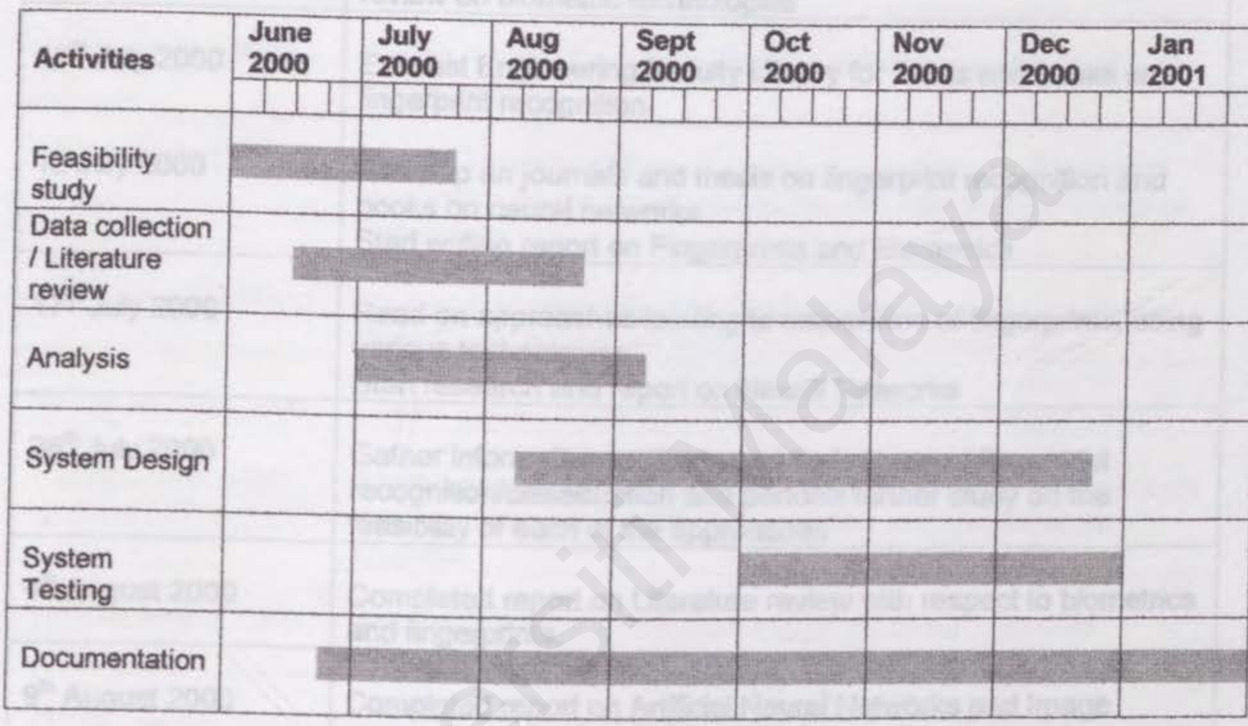


Table A-1 : Project schedule (Gantt chart)

2.2 Project Schedule (Table)

Date	*Task*
7TH July 2000	Searched internet for resources using search topics like "Fingerprint Recognition" combined with "Neural Networks". Also review on biometric technologies
10th July 2000	Exhaust Engineering Faculty Library for thesis and books on fingerprint recognition
12 July 2000	Read up on journals and thesis on fingerprint recognition and books on neural networks. Start writing report on Fingerprints and Biometrics
17th July 2000	Read on approaches leading to recognition of fingerprints, using various technologies. Start research and report on Neural Networks
26th July 2000	Gather information methods and techniques of fingerprint recognition/classification and perform further study on the feasibility of each of the approaches
4th August 2000	Completed report on Literature review with respect to biometrics and fingerprints.
9th August 2000	Completed report on Artificial Neural Networks and Image processing. Hand in reports on Literature Review and Neural Networks for supervisors evaluation
11th August 2000	Narrow down to a few selected methodologies in feature extraction methods, and further study them.
20th August 2000	Completed final reviews on methodologies on fingerprints feature extraction.
23rd August 2000	Viva (presentation) with project supervisor and moderator
24th August 2000	Make necessary changes and additions to approach taken for Literature Review, Neural Networks, and also Feature extraction
1st September 2000	Hand in reports for supervisors evaluation
4th September 2000	Make necessary modifications and modifications for proposal
15th September 2000	Print and hand up final copy of project proposal

3rd October 2000	Researched on suitable types of Neural network to use for recognition of fingerprints
14th October 2000	Applied neural networks to entire fingerprint as method of recognition
24th October 2000	Confirmed that using entire images takes a very long time for recognition
1st November 2000	Tested Fourier Transform filters as feature extraction methods for fingerprints
9th November 2000	Fourier Transform filters were found unsuitable for fingerprint recognition as the results varied far with minimal differences of fingerprints
11th November 2000	Studied and researched other more suitable image processing filters to use for feature extraction and noise removal
28th November 2000	Tested feature extraction filters for recognition of fingerprints
7th December 2000	Positive results after application of squared differences filters on images, which were rotation, translation and stretch/scale invariant.
10th December 2000	Refined threshold and feature extraction filters to enable recognition to work faster and more efficiently
13th December 2000	Included and tested database functions and interface for adding new users for database to perform efficiently
17th December 2000	Successfully completed testing for database functions and integrated database into system
26th December 2000	Refined Graphical User Interface, feature extraction and threshold functions and systems overall runtime
3rd January 2001	Started compiling work details and material in preparation for writing for final report.
6th January 2001	Completed report for systems user guide.
15th January 2001	Completed report for system Evaluation and conclusions.
21st January 2001	Completed final report and presentation material for Fingerprint Verification System

Table A-2 : Project schedule (Table)

References

[1] Bowen, J.D., "The Home Office Automatic Fingerprint Pattern Classification Project", *IEEE in proc. Coll. On neural network for image processing applications*, (1992)

[2] Bunke, H. "Structural and Syntactic Pattern Recognition", in *Handbook of Pattern Recognition & Computer Vision*, C.H. Chen et. Al. eds., World Scientific, (1993)

[3] Maio, D. and Maltoni, D., "A Structural Approach in Fingerprint Classification", *IEEE in proc. ICPR* (1996)

[4] Luk, A., Leung, S.H., Lee, C.K. and Lau, W.H., "A Two-Level Classifier for Fingerprint Recognition", *IEEE* (1991)

[5] Dechman, G.H. "Fingerprint Identification Standards for Emerging Applications", *FBI / CJIS Electronic Fingerprint Transmission Specification* (CJIS-RS-0010 (V4)), Appendix F, (August 1995)

[6] Maio, D. and Maltoni, D., "Neural Network Based Minutiae Filtering in Fingerprints", *IEEE* (1998)

[7] Earl Jackson, Jr., "Fingerprint Classification", *University of California, Santa Cruz,* (18 July 2000). http://wwwcatsic.ucsc.edu/~ltmo64d/fprnt.htm

[8] Graselli, A., "On the Automatic Classification of Fingerprint – Some consideration of the Linguistic Interpretation of Pictures" in *Methodologies of Pattern Recognition*, S. Watanabe eds., pp. 253-273, Academic Press, (1969)

[9] Hankley, W.J. and Tou, J.T., "Automatic Fingerprint Interpretation and Classification via Contextual Analysis and Topological Coding", *in Pictorial Pattern Recognition, G.C. Cheng et. Al. eds.*, pp. 411-456, Thompson Book Co., (1968)

[10] Ashbourn, J., "The Biometric White Paper" (27 July 2000) http://www.biometric.freeserve.co.uk/whitepaper.htm

[11] Mehtre, B.M. and Murthy, N.N., "A Minutiae Based Fingerprint Identification System", *proc. 2nd Int. Conf. Advances in Pattern Recognition and Digital Techniques*, Calcutta (1986)

[12] Pernus, F., Kovacic, S. and Grergyek, L., "Minutiae based fingerprint recognition", *proc. 5th Int. Conf. On Pattern Recognition*. Pp. 1380-1382, (1980)

[13] Wegstein, J.H. "An Automated Fingerprint Identification System", USA Government Publication, Washington (1982)

[14] Finger-scan Technology, (18 July 2000) http://www.finger-scan.com/finger-scan_technology.htm

[15] Wahab, A., Chin, S.H. and Tan, E.C., "Novel Approach to Automated Fingerprint Recognition", *IEEE in proc. online no. 19981809* (1998)

[16] Hrechak, A.K. and Mchugh, J.A., "Automated Fingerprint Recognition Using Structural Matching", *Pattern Recognition, 23 (8)* pp. 893-904 (1990)

[17] Cappelli, R., Lumini, A., Maio, D. and Maltoni, D., "Fingerprint Classification by Directional Image Partitioning", *IEEE Trans PAMI – 10 (5)*, (1999)

[18] Leung, M.T., Engeler, W.E. and Frank, P., "Fingerprint Processing Using Back Propagation Neural Networks", *Electronic Systems Library, IEEE, IJCNN* (1990)

[19] Jain, A.K., Prabhakar, S., Hong, L. and Pankanti, S., "FingerCode: A Filterbank for Fingerprint Representation and Matching", *IEEE in proc. CVPR* (1998)

[20] Haykin, S., "Neural Networks – A Comprehensive Foundation", *Macmillan College Publishing Company Inc.* USA (1994)

[21] Faggin, F., "VLSI implementation of neural networks", *Tutorial Notes, International Joint Conference on Neural Networks*, Seattle, WA (1991)

[22] Churchland, P.S., Neurophilisophy: Toward a Unified Science of the Mind/Brain. *Cambridge, MA: MIT Press.* (1986)

[23] Shepherd, G.M. and C. Koch, "Introduction to synaptic circuits", *In The Synaptic Organization of the Brain. pp 3-31*. New York: Oxford University Press (1990)

[24] Fingerprint Recognition Projects (5th August 2000) http://www.vision.caltech.edu/CNS248/Fingerprint/fingerprint.html

[25] Baldi, P. and Y. Chauvin, "Neural Networks for Fingerprint Recognition", *Neural Computation, Vol 5, No 3* (1993)

[26] Umbaugh, Scott E. Dr., "Computer Vision and Image Processing", *Prentice-Hall International, Inc.* (1998)

[27] Kendall, Kenneth E. and Kendall, Julie E., "Systems Analysis and Design 2nd Edition", *Prentice-Hall International, Inc.* (1992)

[28] Gonzales, R.C. and R.E. Woods, "Digital Image Processing", *Addison-Wesley Publishing Company* (1993)

[29] MathWorks Inc., "Image Processing Toolbox : for use with MATLAB – User's Guide" (1997)

[30] Demuth, H. and Beale, M., "Neural Network Toolbox : for use with MATLAB – User's Guide, Version 3", (July 1998)