FINGERPRINT RECOGNITION SYSTEM

by

Ng E-Laine

Dissertation submitted in partial fulfilment of
the requirements for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

JUNE 2008

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

FINGERPRINT RECOGNITION SYSTEM

by

Ng E-Laine

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
BACHELOR OF ENGINEERING (Hons)
(ELECTRICAL & ELECTRONICS ENGINEERING)

Approved by,

[illegible]

(Dr. Yap Vooi Voon)

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

JUNE 2008

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

[illegible]

Ng E-Laine

ABSTRACT

This project is to design a fingerprint recognition system for security purposes. It will also explore and suggest some solution to the improvement to the existing fingerprint system. Security system that uses a pin code or access card can be easily misused or mishandled. A pin code can be cracked using some hacker software while an access card can easily be stolen or misplaced. Thus, these security methods are very vulnerable to hackers and criminals. Instead, a fingerprint is unique to every person and due to the fact that no two people have the same fingerprint pattern, it makes the fingerprint a very good resource in a security system.

The aim of this project is to focus on the concept and methodology of the fingerprint recognition system. By grasping the concept and method of the fingerprint recognition flow, a prototype is developed that will compare an input fingerprint with its predefined template. The system should be able to compare and decide if the input fingerprint is the same as the predefined template.

The output of the first stage is a preprocessing stage. There are two stages involved in preprocessing which is the image enhancement and image skeletonization. Fourier transform and histogram equalization is utilized to enhance the low quality image to a better image so that the feature extraction process will run smoothly.

The second stage of the project is to define the orientation, ROI extraction and minutia extraction. The matching sequence and the angle orientation problem were resolved.

ACKNOWLEDGEMENT

First and foremost, I would like to thank my supervisor, Dr. Yap Vooi Voon for taking up my project and guided me throughout the semester. Also, I would like to express my gratitude to Mr. Patrick for enlightening me on the project background and also offering help from time to time.

Next, I would like to thank the ELID team members that have cleared up my confusion on the basis of the project. Such experienced advice would be very valuable for a novice like myself to be able to come up with a good working project.

I would also like to thank the Final Year Project committee of UTP that has provided a good guideline on how the project should be.

I am also grateful to my parents that have contributed not only financially, but also the support that they have given me on this project.

Last but not least, I would like to thank all those that have contributed directly and indirectly to the success of this project.

THANK YOU

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1
INTRODUCTION

1.1 Background Study

Fingerprint recognition is the most common and oldest biometric authentication method used to date. This is because every person has been proven to have unique fingerprint pattern. The skin under the finger contains pattern that forms ridges and valley. Even identical twins do not have the same set of fingerprint, so a fingerprint recognition system can be developed and used as a security system. Some system will consider the basis of the actual comparison, but in this project, specific characteristics for the comparison process are extracted.

Fingerprint pattern is made up of ridges and bifurcations and these special points are known as minutiae. See Figure 1. Typically, a fingerprint contains of more than a hundred minutiae, but for identification purposes, only 12 minutiae are required.

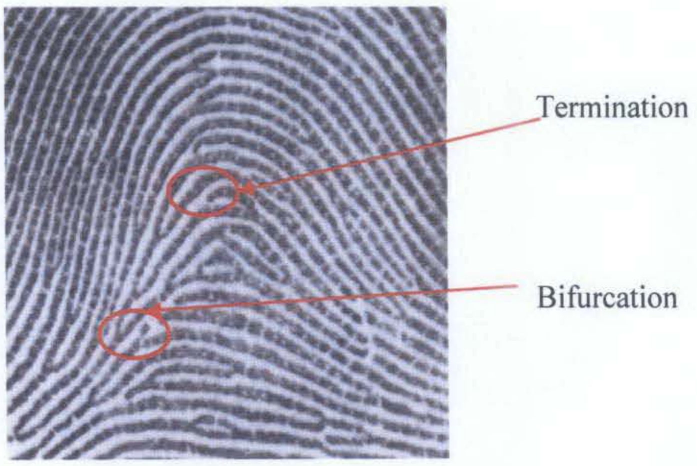


Figure 1: Ridges and bifurcations points

There are a couple of methods to conduct a fingerprint comparison and each method comes with its own pros and cons. Some of them include:

1. Point based matching
2. Correlation based matching
3. Euclidean Distance-based matching
4. Gabor filtering

Point based matching

By using this method, minutiae points on the fingerprint is located and then mapped to its predefined template during the comparison process. See Figure 2. The problem with this method is it is greatly affected by the image quality of the fingerprint. If the image is blurred, the feature points will be missed.

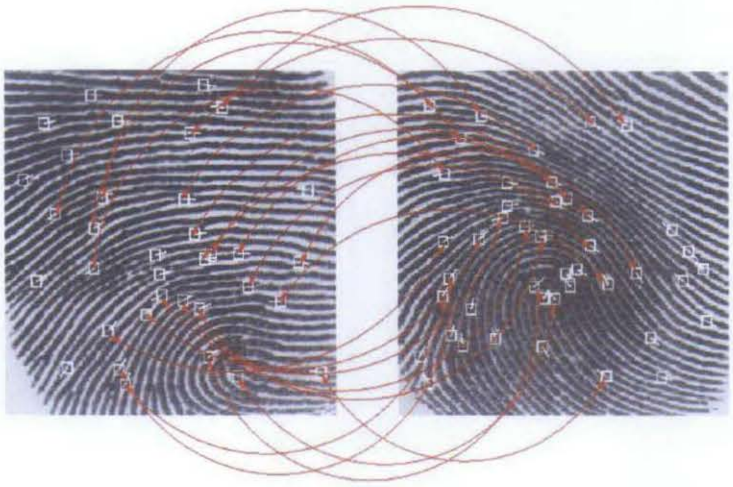


Figure 2: Minutiae based matching

Correlation based matching

Correlation based method actually overcomes some of the problem that the point based method is having. A simple correlation technique is to align an input print to a predefined one. It will subtract the input image to see if the points correspond to the match. This method in turn encounters numerous problems, for example the orientation of the images and error in the alignment. Also, it requires a higher processing time as it needs to convert the spatial images to frequency representation.

Euclidean Distance-based matching

This method computes the distance between two points and then compares it with the input print. It assumes that the matching algorithms are translation-invariant. Although this method is pretty straight forward and requires less computational time, it has difficulties in matching prints with large, nonlinear deformations.

Gabor filtering

In the Gabor filter technique, a set of Gabor filters are applied to a fingerprint image. Then the Gabor feature is obtained by convolving the filter with the image and the features will then be used in the matching algorithm. The good point about Gabor filter is the quality of the image is not considered as important as other techniques as the Gabor technique does not need to extract points like the bifurcation or termination.

1.2 Problem statement

The major problems that the fingerprint recognition system is facing include:

1. Image quality
2. Image positioning
3. Lack of accurate approach
4. Duplication

All these problems show that there are still rooms for improvement for the fingerprint recognition system. One of the aim of my project is to increase to accuracy of the fingerprint system as high as possible.

1.3 Objective

1. To create a fully working fingerprint recognition system that will boost the maximum accuracy of the system.
2. To rectify and overcome some problems that the current recognition system possesses.

Chapter 2
LITERATURE REVIEW

Fingerprints, among all biometrics authentication method are the oldest because of their high acceptability, immutability and individuality. Immutability refers to the change of the fingerprints over time whereas each individual is related by the uniqueness of ridge details across individuals. The probability that two fingerprints are alike is 1 in 1.9 x 1015. Thus, these features make the use of fingerprints extremely effective in areas where the high security is in the question. The major steps involved in automated fingerprint recognition include

a) Fingerprint Acquisition
b) Fingerprint Segmentation
c) Fingerprint Image Enhancement
d) Feature Extraction
e) Minutiae Matching
f) Fingerprint Classification.

Fingerprint acquisition can be processed in either offline or online. Offline is done by using ink and online is conducted using a sensor or Live-scan. In the inked method, the print an inked finger is first obtained on a paper, which is then scanned. This method usually produces images of very poor quality because of the non-uniform spread of ink and is therefore not used in online Automated Fingerprint Identification System (AFIS). For online fingerprint image acquisition, optical fingerprint scanners are used and live scan scanners offer much greater image quality, usually a resolution of 512 dpi, which results in superior reliability during matching in comparison to inked fingerprints.

Segmentation refers to the separation of fingerprint area (foreground) from the image background [1]. Segmentation is useful to avoid extraction of features in the noisy areas of fingerprints or the background. Ren et al. [2] proposed an algorithm for segmentation that employs feature dots, which are then used to obtain a close segmentation curve. Bazen et al. [3] proposed a pixel wise technique for segmentation involving a linear combination of three feature vectors (i.e. gradient coherence, intensity mean and variance). A final morphological post-processing step is performed to eliminate holes in both the foreground and background. In spite of its high accuracy this algorithm has a very high computational complexity, which makes it impractical for real time processing.

Feature extraction is much easier, efficient and reliable in a good quality fingerprint compared to a relatively lower quality fingerprint. The quality of fingerprints is degraded by skin conditions (e.g. wet or dry, cuts and bruises), sensor noise, non-uniform contact with sensor surface, and inherently low quality fingerprint images (e.g. those of elderly people, laborers). A significant percentage of fingerprints are of poor quality, which must be enhanced for the recognition process to be effective. The two major objectives of fingerprint enhancement are

i) To increase the contrast between ridges and valleys and

ii) To connect broken ridges.

These can be done by using a contextual filter whose characteristics vary according to the local context to be used for fingerprint enhancement instead of conventional image filters. The filter should posses the following characteristics:

- Provide a low pass (averaging) effect along the ridge direction to link small gaps and filling impurities due to pores or noise.
- Have a band pass (differentiating) effect in the direction orthogonal to the ridges to increase the discrimination between ridges and valleys and top separate parallel linked ridges.

Sherlock et al. [4] proposed an algorithm for fingerprint image enhancement that employs position-dependent Fourier-domain-filtering-based orientation smoothing and thresholding technique. Greenberg et al. [5] proposed two schemes for fingerprint enhancement. One method uses local histogram equalization, Wiener filtering, and image binarization whereas the other method uses a unique anisotropic filter for direct grayscale enhancement. Hong et al. [6] proposed an effective method based on Gabor filters for image enhancement. Gabor filters fulfill the requirements of a good fingerprint enhancement filter mentioned earlier as they possess both the differentiating and averaging effects. Watson et al. [7] multiplied the Fourier transform of each 32x32 block by its power spectrum raised to a power k to produce an efficient technique that can find its use in online fingerprint recognition systems.

Fingerprints possess two major types of features: special type of pattern formed by the ridge and furrow structures in the central region of the fingerprints and minutiae details associated with local ridges and furrows. Minutiae are point details such as ridge endings (a point where a ridge ends abruptly) or a ridge bifurcation (where a ridge breaks up into two ridges). Minutiae are characterized by the position, direction and type. Fingerprint analysts utilize the minutiae information for fingerprint identification and it is the most established technique in the field of AFIS. The accuracy of the technique is dependent upon the precision in the extraction of minutiae. There are two major types of methods that are used for minutiae extraction:

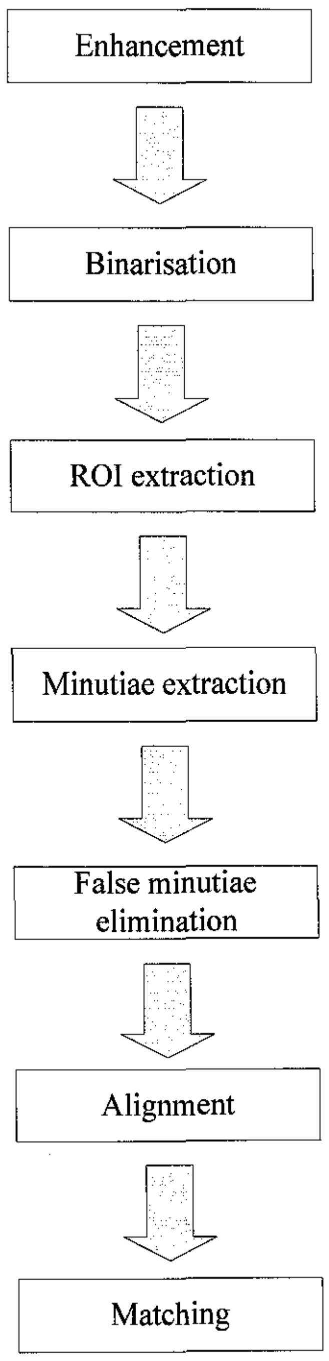
i) binarization based methods and
ii) direct gray scale methods.

In binarization-based methods, some information is lost during binarization which can affect the performance of the minutiae extractor. Direct gray scale methods overcome these problems but may be difficult to implement and time consuming to operate. The typical approach for a binarization-based method involves a priori enhancement, binarization and then thinning. Once a binary skeleton has been obtained, a simple image scan allows the pixels corresponding to the minutiae to be detected by finding the crossing number. False minutiae will then need to be removed after the minutia extraction. During enrollment the minutiae set obtained from an individual's fingerprint is stored as a template for that subject.

In the authentication module, the fingerprint given as input is compared or matched with the templates to provide the decision of authentication. This process is known as minutiae matching. For minutiae matching to be effective the input fingerprint should be registered to the template fingerprint using the minutiae information of both the fingerprints. After registration the minutiae sets are compared using the spatial distance, which must be smaller than a particular threshold for two minutiae to be declared as matched. A minutia matching is usually performed by using Hough transform [8, 9] or by pre alignment[10].

Chapter 3
METHODOLOGY

The fingerprint recognition procedure can be summarize as follows:



By using histogram equalization and FFT, we could enhance the image for better processing.

Thin the image into one pixel images for easier processing.

Extract only regions that contain information. Eliminate redundant information.

Extract minutia points for matching algorithm.

Remove minutia points that are false and might affect accuracy.

Rotate and translate 2 sets of reference points if their similarity exceeds a certain threshold.

Match translated points and calculates their matching percentage.

3.1 Image Acquisition

For this project, the images are be obtained from the Fingerprint Verification Competition 2002 (FVC2002) database, so no acquisition process is required.

3.2 Image enhancement

3.2.1 Fourier Transform

First of all, the image is divided into 8 by 8 bit per pixel and perform Fourier transform according to the following formula;

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) \times \exp\left\{-j2\pi \times \left(\frac{ux}{M} + \frac{vy}{N}\right)\right\}$$

For u= 1, 2, 3...7 and v=1, 2, 3...7

To enhance a specific block by its dominant frequencies, the FFT of the block is multiplies by its magnitude a set of times. The magnitude of the original FFT is defined by

FFT = abs(F(u,v)) = |F(u,v)|.

Then the enhanced block is obtained according to the below formula

$$g(x,y) = F^{-1}\left\{F(u,v) \times |F(u,v)|^{K}\right\}$$

where $F^{-1}(F(u,v))$ is defined by:

$$f(x,y) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} F(u,v) \times \exp\left\{j2\pi \times \left(\frac{ux}{M} + \frac{vy}{N}\right)\right\}$$

for x = 0, 1, 2, ..., 7 and y = 0, 1, 2, ..., 7.

The k in formula g(x,y) is an empirical constant, k=0.1. Having a higher constant improves the appearance of the ridges and will fill up small holes in ridges but on the other hand, if the constant is too high, it can result in false joining of ridges. Thus a termination might become a bifurcation. Refer to Figure 4 in **Results and Discussion** section for the enhance images.

3.2.2 Histogram Equalization

Histogram equalization is basically similar to histogram stretching which is to stretch the intensity of the image as flat as possible to increase the perceptional information. Although histogram equalization managed to enhance the image, it is not able to connect broken ridges. So, 2 types of enhancement method is used.

3.3 Image Thinning

Binarization of an image is to transform the gray scale image into a binary image. The reason to convert it into pixels 1 and 0 is because the image is easier to work with. To do that, the threshold level of the image is first determined. Then the pixels of the image will be converted to 1s and 0s based on the threshold limit. Then, the image will go through the thinning process. This process is required to identify the location of the ridges and furrow. Ridges are represented in black while furrows are represented in white.

3.4 Feature Extraction

3.4.1 Block direction estimation

In a fingerprint image, it not only contains information to be used in the matching stage, but it also contains redundant background information such as ineffective ridges or furrows. Thus, a ROI extraction is essential to obtain only effective and accurate information. There are two methods that can be used to perform the ROI extraction which is the block direction estimation and morphological operations.

Block direction estimation

The block direction is estimated in a W x W pixel by:

1. Calculate the gradient values along x-direction (g_x) and y-direction (g_y) for each pixel of the block. The horizontal Sobel operator can be used to calculate the gradient for x-direction while a vertical Sobel operator is used to calculate the gradient along the y-direction.
2. For each block, use formula below to get the Least Square approximation of the block direction,

$$tg2\beta = 2 \sum \sum (g_x * g_y) / \sum \sum (g_x^2 - g_y^2)$$

   for all the pixels in each block. By regarding the gradient along x and y as sine and cosine value, it can be deduced the tangent value to be nearly similar to the following formula,

$$tg2\theta = 2\sin\theta \cos\theta / (\cos^2\theta - \sin^2\theta)$$

3. After that, blocks with inaccurate ridges and furrow can be eliminated based on the following formula

$$E = \{2 \sum \sum (g_x * g_y) + \sum \sum (g_x^2 - g_y^2)\} / W * W * \sum \sum (g_x^2 + g_y^2)$$

   For each block, if its certainty level E is below a threshold, then the block is regarded as a background block.

3.4.2 Region of Interest (ROI) extraction

Two Morphological operations called 'OPEN' and 'CLOSE' are used to obtain the ROI bound. The 'OPEN' operation can expand images and remove peaks introduced by background noise .The 'CLOSE' operation can shrink images and eliminate small cavities. By subtracting the CLOSE area from the OPEN area, the center area can be binded, where the effective ridges and furrows lie.

Also the average inter-ridge width D is estimated at this stage. The average inter-ridge width refers to the average distance between two neighboring ridges. The way to approximate the D value is simple. Scan a row of the thinned ridge image and sum up all pixels in the row whose value is one. Then divide the row length with the above summation to get an inter-ridge width. For more accuracy, such kind of row scan is performed upon several other rows and column scans are also conducted, finally all the inter-ridge widths are averaged to get the D.

Together with the minutia marking, all thinned ridges in the fingerprint image are labeled with a unique ID for further operation. The labeling operation is realized by using the Morphological operation: BWLABEL.

3.5 Minutiae Extraction

In general minutiae are divided into ridges and furrows. So, in a 3 x 3 window, the pixels can be defined as:

0	1	0
0	1	0
1	0	1

Figure 3.1: Bifurcation

0	0	0
0	1	0
0	0	1

Figure 3.2: Ridge ending

If the central pixel is 1 and has exactly 3 one-value neighbors, then the central pixel is a ridge branch [Figure 8.1]. If the central pixel is 1 and has only 1 one-value neighbor, then the central pixel is a ridge ending [Figure 8.2].

3.5.1 False Minutiae

Currently, there are 7 types of false minutiae. The preprocessing stage does not totally heal the fingerprint image. For example, false ridge breaks due to insufficient amount of ink and ridge cross-connections due to over inking are not totally eliminated. These false minutiae will significantly affect the accuracy of matching if they are simply regarded as genuine minutia. So some mechanisms of removing false minutia are essential to keep the fingerprint verification system effective.

The types of false minutiae can be classified as follow:

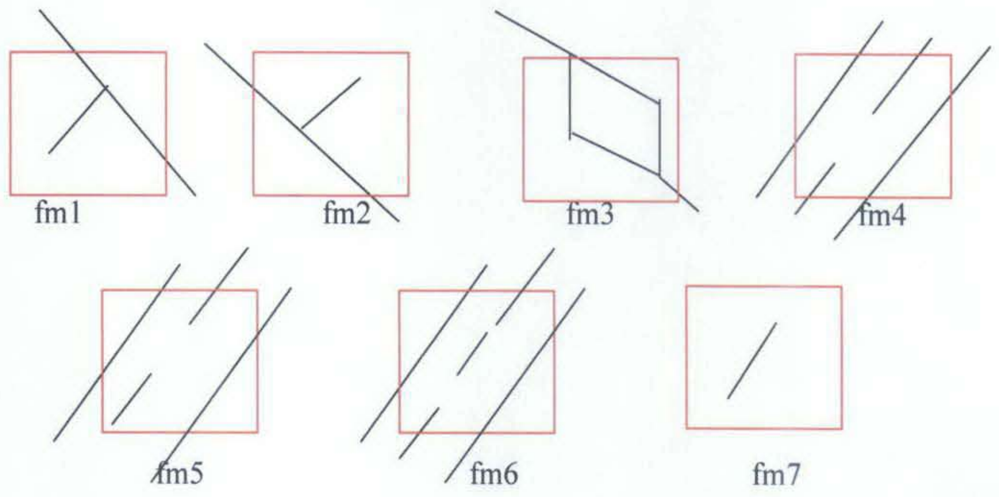


Figure 4: fm1 is a spike piercing into a valley. In the fm2 case a spike falsely connects two ridges. fm3 has two near bifurcations located in the same ridge. The two ridge broken points in the fm4 case have nearly the same orientation and a short distance. fm5 is alike the fm4 case with the exception that one part of the broken ridge is so short that another termination is generated. fm6 extends the fm4 case but with the extra property that a third ridge is found in the middle of the two parts of the broken ridge. fm7 has only one short ridge found in the threshold window.

This project proposed the following in removing false minutia:

1. If the distance between one bifurcation and one termination is less than average inter-ridge width, D and the two minutiae are in the same ridge (fm1 case). Remove both of them.
2. If the distance between two bifurcations is less than D and they are in the same ridge, remove the two bifurcations. (fm2, fm3 cases).
3. If two terminations are within a distance D and their directions are coincident with a small angle variation. And they suffice the condition that no any other termination is located between the two terminations. Then the two terminations are regarded as false minutia derived from a broken ridge and are removed. (case fm4, fm5, fm6).

4. If two terminations are located in a short ridge with length less than D, remove the two terminations (fm7).

Given two set of minutia of two fingerprint images, the minutia match algorithm determines whether the two minutia sets are from the same finger or not.

3.6 Alignment and Matching

An alignment-based match algorithm is used in this project. It includes two consecutive stages: which is the alignment stage and the match stage.

1. **Alignment stage**. One minutia is chosen from two images that are to be matched as reference points. Then the similarity of the two ridges associated with the two referenced minutia points is calculated. If the similarity is larger than a threshold, transform each set of minutia to a new coordination system whose origin is at the referenced point and whose x-axis is coincident with the direction of the referenced point.
2. **Match stage:** The elastic match algorithm is used to count the matched minutia pairs by assuming two minutiae having nearly the same position and directions are identical.

3.6.1 Alignment Stage

1. The ridge associated with each minutia is represented as a series of x-coordinates (x_1, $x_2 \ldots x_n$) of the points on the ridge. A point is sampled per ridge length L starting from the minutia point, where the L is the average inter-ridge length.
   So the similarity of correlating the two ridges is derived from:

$$S = \Sigma^{m}_{i=0} x_i X_i / [\Sigma^{m}_{i=0} x_i^2 X_i^2]^\wedge 0.5,$$

where ($x_{i\sim} x_n$) and ($X_{i\sim} X_N$) are the set of minutia for each fingerprint image respectively. And m is minimal one of the n and N value. If the similarity score is larger than 0.8 (determined empirically), then go to step 2, otherwise continue to match the next pair of ridges.

2. For each fingerprint, translate and rotate all other minutia with respect to the reference minutia according to the following formula:

$$\begin{pmatrix} xi\_new \\ yi\_new \\ \theta i\_new \end{pmatrix} = TM * \begin{bmatrix} (xi - x) \\ (yi - y) \\ (\theta i - \theta) \end{bmatrix}$$

,

where (x,y,θ) is the parameters of the reference minutia, and TM is

$$TM = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3.6.2 Match Stage

The matching algorithm for the aligned minutia patterns needs to be elastic since the strict match requiring that all parameters (x, y, θ) are the same for two identical minutia is impossible due to the slight deformations and inexact quantizations of minutia.

The approach that was taken in this project to elastically match minutia is achieved by placing a bounding box around each template minutia. If the minutia to be matched is within the rectangle box and the direction discrepancy between them is very small, then the two minutiae are regarded as a matched minutia pair. Each minutia in the template image either has no matched minutia or has only one corresponding minutia.

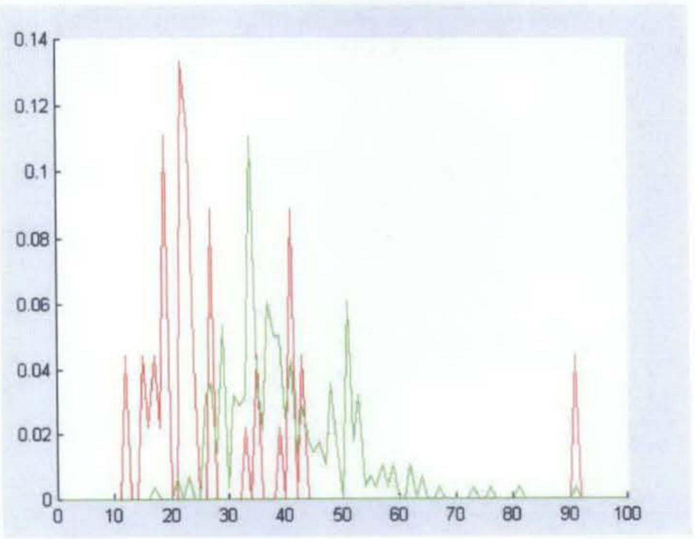
However, the elastic match algorithm has large computation complexity and is vulnerable to spurious minutia.

Chapter 4
RESULTS AND DISCUSSION

Two of the widely accepted indexes to determine the performance of a fingerprint recognition system are the False Accept Rate (FAR) and False Reject Rate (FRR). Each sample is matched against samples from the same finger to compute the False Rejection Rate. All the scores for matches are then composed into a series of Correct Score. Also the first sample of each finger in the database is matched against the first sample of the remaining fingers to compute the False Acceptance Rate. All the scores from such matches are composed into a series of Incorrect Score.

4.1 Experimental Results

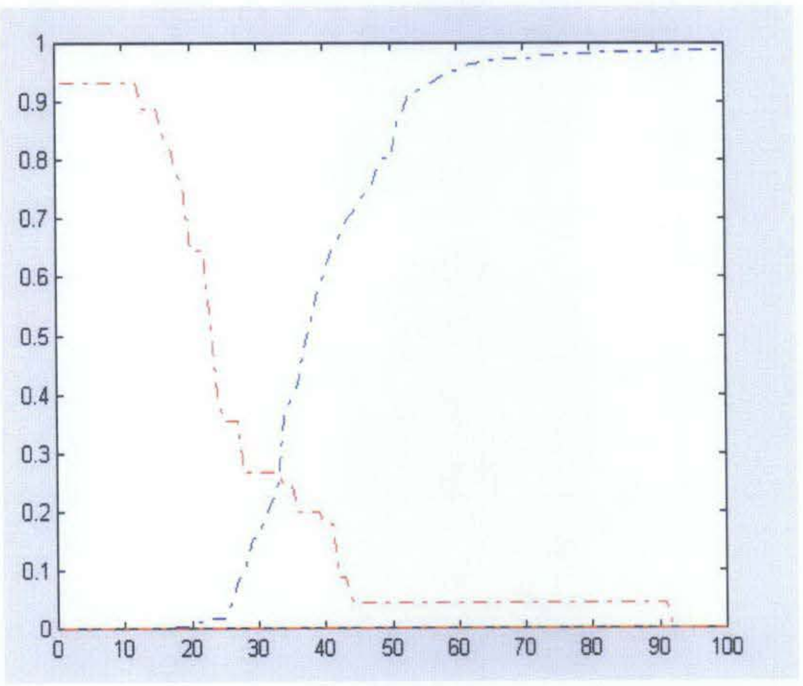
A fingerprint database from the FVC2000 (Fingerprint Verification Competition 2000) is used to test the performance of the fingerprint system. The experiments show that the program can differentiate imposturous minutia pairs from genuine minutia pairs in a certain confidence level. The diagram for Correct Score and Incorrect Score distribution is shown in Figure 4.1.



Red line: Incorrect Score
Green line: Correct Scores

Figure 5.1: Score distribution results

From the above figure, it can be seen that there exist two partially overlapped distributions. The Red curve whose peaks are mainly located at the left part means the average incorrect match score is 25. The green curve whose peaks are mainly located on the right side of red curve means the average correct match score is 35. This indicates the algorithm is capable of differentiate fingerprints at a good correct rate by setting an appropriate threshold value.



Blue dot line: FRR curve
Red dot line: FAR curve

Figure 5.2: FAR and FRR curve

The above diagram shows the FRR and FAR curves. At the equal error rate 25%, the separating score 33 will falsely reject 25% genuine minutia pairs and falsely accept 25% imposturous minutia pairs and has 75% verification rate.

4.2 Fourier transform enhancement

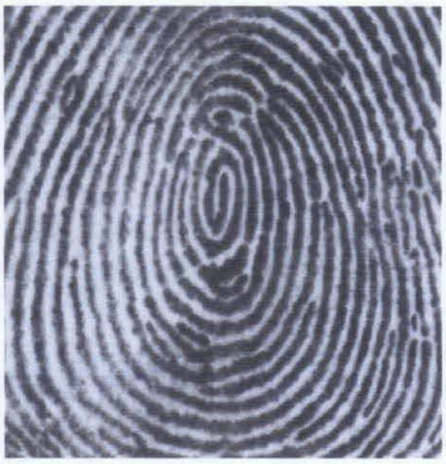


Image before FFT

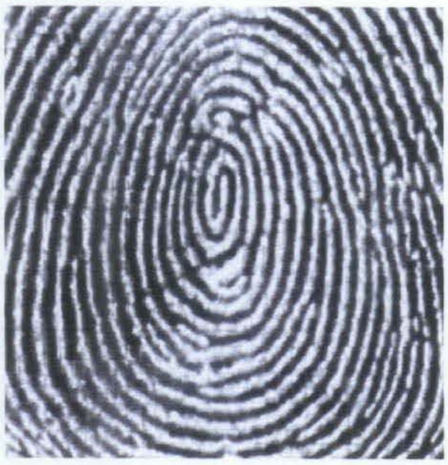


Image after FFT

Figure 6.1

In the above enhanced image, it can be seen that quality of the image has been significantly enhanced.

4.3 Histogram equalization enhancement

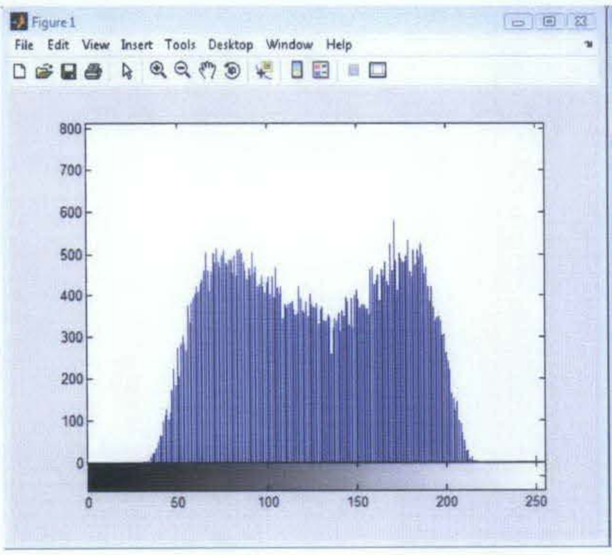


Image histogram before enhancement

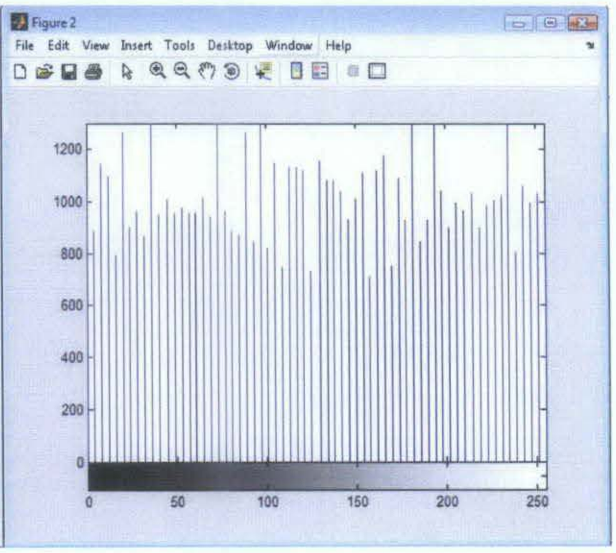


Image histogram after enhancement

Figure 6.2

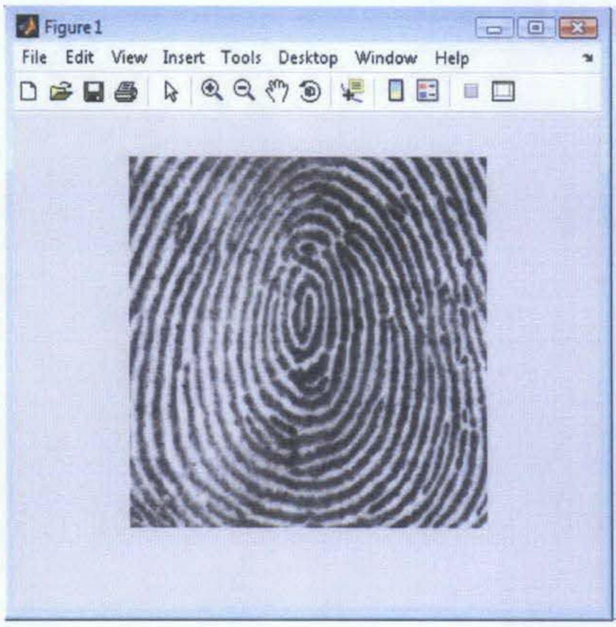


Image before histogram equalization

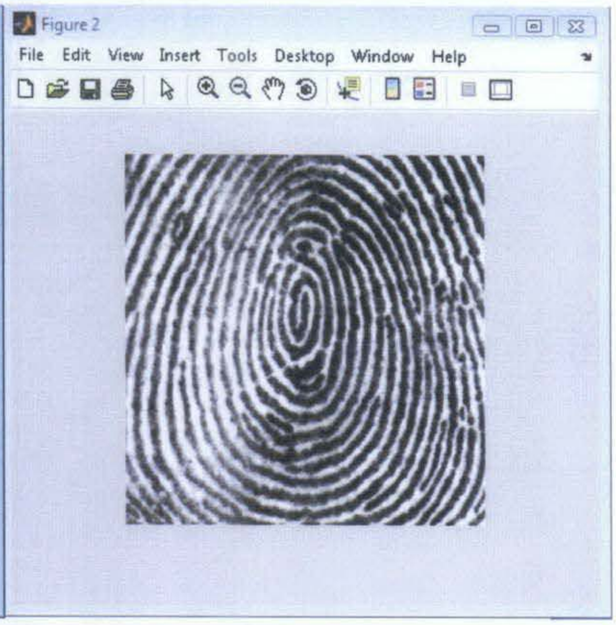
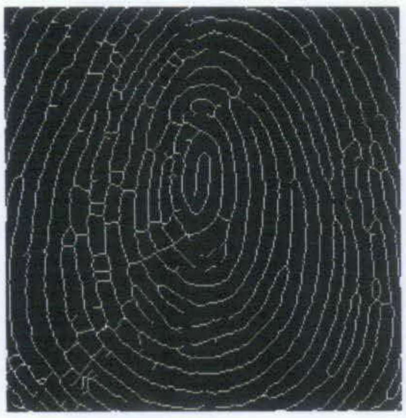


Image after histogram equalization

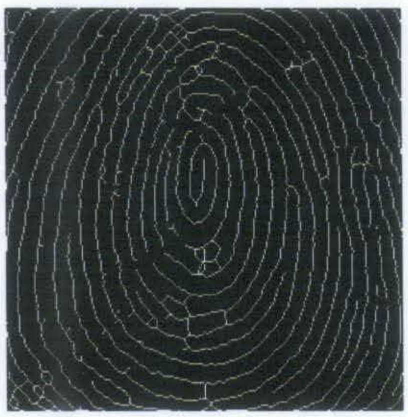
Figure 6.3

From Figure 6.2, it can be seen that the histogram after the enhancement method is more distributed compared to the before enhancement. An enhanced image in Figure 6.3 can be obtained using the histogram equalization technique.

4.4 Image Thinning



Skeleton image before enhancement



Skeleton image after enhancement

Figure 7

From the two images in Figure 7, it can see that the ridges and furrows are better enhanced.

4.5 Block Direction Estimation

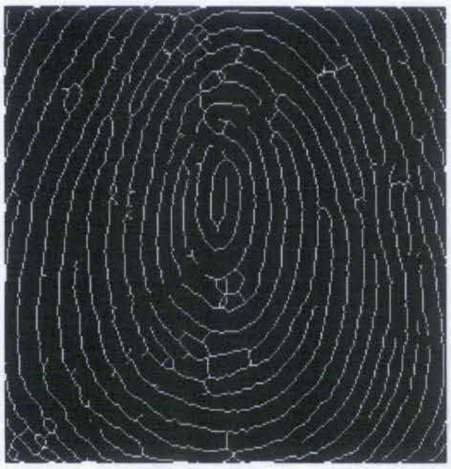


Figure 8.1: Original image

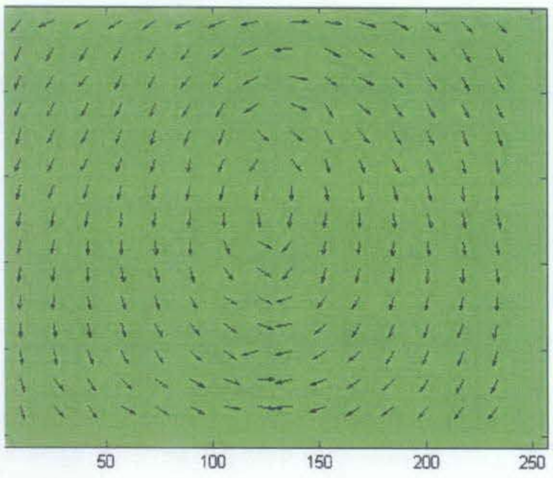


Figure 8.2: Direction map

4.6 ROI EXTRACTION

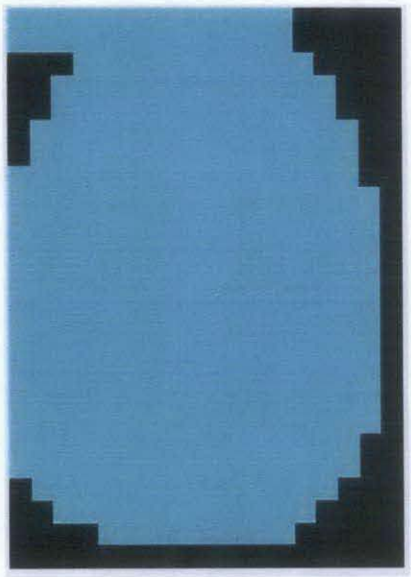


Figure 9.1: Original Image Area

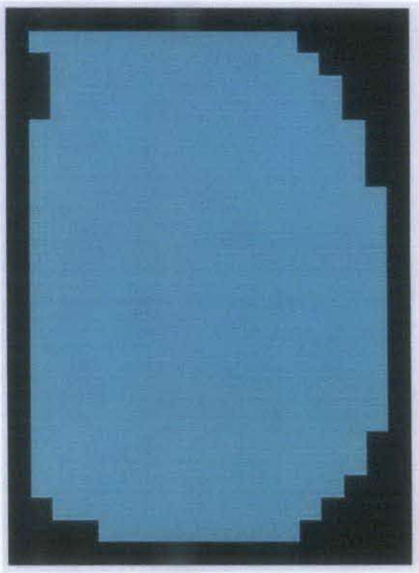


Figure 9.2: Area after 'OPEN'

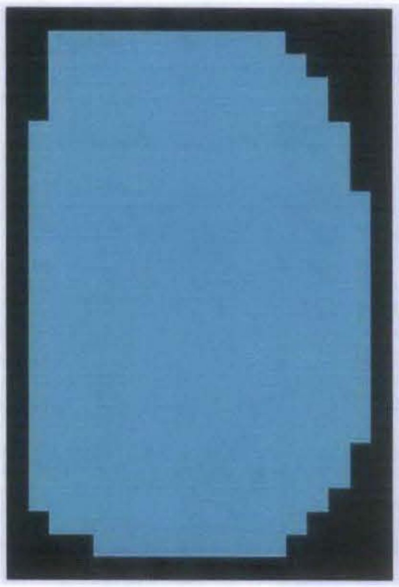


Figure 8.3: Area after 'CLOSE'

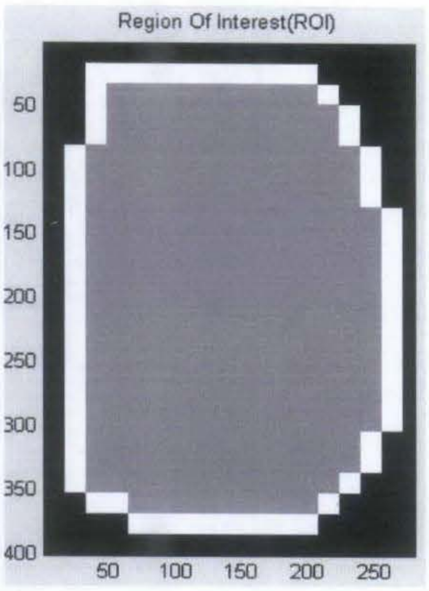


Figure 8.4: ROI

4.7 False Minutiae elimination

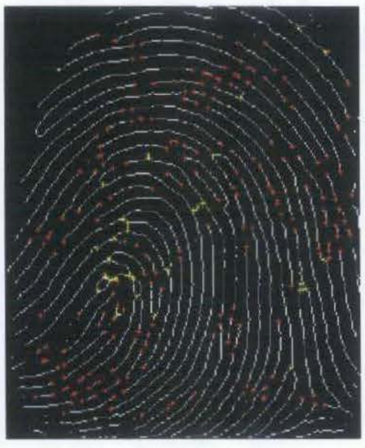


Figure 10.1: Before false minutiae removal

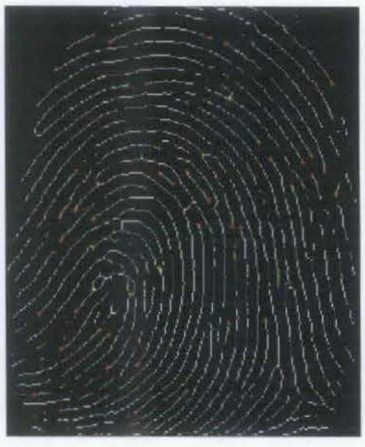


Figure 10.2: After false minutiae removal

Chapter 5
CONCLUSION AND RECOMMENDATION

As a conclusion, this project has successfully completed a fingerprint recognition method. The results demonstrated that it is possible to enhance the input image for a better extraction session. In addition, this project also shows that it is possible to conduct the block direction estimation and extract the ROI. And finally, this project demonstrated that it is possible to eliminate the false minutiae points from the real ones.

The main reason for the high incorrect acceptance rate is due to the quality of the images and also the vulnerable match algorithm. If a method that will not depend significantly on the image quality is used, the fingerprint matching rate will increase dramatically.

REFERENCES

[1] "A Handbook Of Fingerprint Recognition", Maltino, Maio, Jain And Prabhakar. Springer Press.

[2] Ren Qun, Tian Jie, Zhang Xiaopeng, Automatic Segmentation Of Fingerprint Images, Chinese Academy Of Sciences, P.R. China

[3] Bazen and Garez., "Segmentation Of Fingerprint Images", In Proc. Workshop On Circuits Systems And Signal Processing, 2001

[4] Sherlock, Monro and Millard., Algo For Enhancing Fingerprint Images, Electronic Letters, 1992

[5] Shlomo Greenberg, Mayer Aladjem, Daniel Kogan, "Fingerprint image enhancement using filtering techniques" Real-Time Imaging Volume 8, Issue 3 (June 2002) Pages: 227 – 23, 2002

[6] Hong, Wan and Jain, "Fingerprint Image Enhancement: Algorithm And Performance Evaluation", IEEE Transaction On Pattern Analysis And Machine Intelligence, 1998

[7] Watson, Candela and Grother, "Comparison of FFT Fingerprint Filtering Methods For Neural Network Classification, Tech Report: NIST IR 5493, 1994

[8] Ratha Et Al., "A Real Time Matching System For Large Fingerprint Databases", IEEE Transaction On Pattern Analysis And Machine Intelligence, 1996

[9] Bhanu, Boshra and Tan, "Logical Templates For Feature Extraction In Fingerprint Images", In Proc. Int. Conference On Pattern Recognition", 2000]

[10] Jain, "An Identity Authentication System Using Fingerprints", Proc. Of IEEE, 1997

APPENDIX

MATLAB CODE

Image enhancement

```
f=0.45;
I = 255-double(image);

[w,h] = size(I);

w1=w;
h1=h;

inner = zeros(w1,h1);

for i=1:8:w1
  for j=1:8:h1
    a=i+7;
    b=j+7;
    F=fft2( I(i:a,j:b) );
    factor=abs(F).^f;
    block = abs(ifft2(F.*factor));

    larv=max(block(:));
    if larv==0
      larv=1;
    end;

    block= block./larv;
    inner(i:a,j:b) = block;
  end;
end;


fft=inner*255;
fft=histeq(uint8(fft));

bin=im2bw(fft,graythresh(fft));
image=bwmorph(bin,'thin',Inf);
```

Block direction estimation and ROI extraction

```
[w,h] = size(image);
direction = zeros(w,h);
gradient_times_value = zeros(w,h);
gradient_sq_minus_value = zeros(w,h);
gradient_for_bg_under = zeros(w,h);

W = blocksize;
theta = 0;
sum_value = 1;
bg_certainty = 0;

blockindex = zeros(ceil(w/W),ceil(h/W));
```

```
times_value = 0;
minus_value = 0;

center = [];
filter_gradient = fspecial('sobel');
I_horizon = filter2(filter_gradient,image);
filter_gradient = transpose(filter_gradient);
I_vertical = filter2(filter_gradient,image);


gradient_times_value=I_horizon.*I_vertical;
gradient_sq_minus_value=(I_vertical-I_horizon).*(I_vertical+I_horizon);
gradient_for_bg_under = (I_horizon.*I_horizon) + (I_vertical.*I_vertical);


for i=1:W:w
   for j=1:W:h
    if j+W-1 < h & i+W-1 < w
        times_value = sum(sum(gradient_times_value(i:i+W-1, j:j+W-1)));
       minus_value = sum(sum(gradient_sq_minus_value(i:i+W-1, j:j+W-1)));
       sum_value = sum(sum(gradient_for_bg_under(i:i+W-1, j:j+W-1)));

       bg_certainty = 0;
       theta = 0;

       if sum_value ~= 0 & times_value ~=0
         bg_certainty = (times_value*times_value + minus_value*minus_value)/(W*W*sum_value);

        if bg_certainty > 0.05
         blockindex(ceil(i/W),ceil(j/W)) = 1;

         tan_value = atan2(2*times_value,minus_value);


           theta = (tan_value)/2 ;
           theta = theta+pi/2;

         center = [center;[round(i + (W-1)/2),round(j + (W-1)/2),theta]];
           end;
      end;
   end;
         times_value = 0;
       minus_value = 0;
       sum_value = 0;

  end;
end;

x = bwlabel(blockindex,4);
close = bwmorph(x,'close');
open= bwmorph(close,'open');
region = bwperim(open);
```

Direction of image

```
[w,h] = size(image);

a=sum(inROI);

b=find(a>0);

c=min(b);
d=max(b);
i=round(w/5);
c=0;

for k=1:4
  c=c+sum(image(k*i,16*c:16*d));
end;
e=(64*(d-c))/c;

a=sum(inROI,2);
b=find(a>0);

c=min(b);
d=max(b);

i=round(h/5);
c=0;

for k=1:4
  c=c+sum(image(16*c:16*d,k*i));
end;
c=(64*(d-c))/c;

D=round((c+e)/2);
```

Marking the minutiae

```
[w,h] = size(image);

[ridgeOrderMap,totalRidgeNum] = bwlabel(image);

imageBound = bound;
imageArea = area;
blkSize = block;

edgeWidth = interRidgeWidth(in,inArea,blkSize);

end_list    = [];
branch_list = [];


for n=1:totalRidgeNum
   [m,n] = find(ridgeOrderMap==n);
   b = [m,n];
   ridgeW = size(b,1);
   for x = 1:ridgeW
     i = b(x,1);
     j = b(x,2);

if area(ceil(i/blkSize),ceil(j/blkSize)) == 1
     neighbor = 0;
     neighbor = sum(sum(image(i-1:i+1,j-1:j+1)));
     neighbor = neighbor -1;


   if neighbor == 1
       end_list =[end_list; [i,j]];

   elseif neighbor == 3
     tempo=image(i-1:i+1,j-1:j+1);

     tempo(2,2)=0;
     [abr,bbr]=find(tmp==1);
     t=[abr,bbr];

     if isempty(branch_list)
       branch_list = [branch_list;[i,j]];
     else

     for p=1:3
       cbr=find(branch_list(:,1)==(abr(p)-2+i) & branch_list(:,2)==(bbr(p)-2+j) );
       if ~isempty(cbr)
         p=4;
         break;
       end;
     end;

     if p==3
       branch_list = [branch_list;[i,j]];
     end;

    end;

    end;

    end;

end;
end;
```

False minutiae elimination

```
[w,h] = size(in);

final_end = [];
final_branch =[];
direct = [];
pathMap = [];

end_list(:,3) = 0;
branch_list(:,3) = 1;

minutiae = [end_list;branch_list];
final = minutiae;
[noofmin,dummy] = size(minutiae);
suspectMinList = [];

for i= 1:noofmin-1
  for j = i+1:noofmin
    d =( (minutiae(i,1) - minutiae(j,1))^2 + (minutiae(i,2)-minutiae(j,2))^2)^0.5;

    if d < edgeWidth
      suspectMinList =[suspectMinList;[i,j]];
    end;
  end;
end;

[totalSuspectMin,dummy] = size(suspectMinList);
%totalSuspectMin

for k = 1:totalSuspectMin
  typesum = minutiae(suspectMinList(k,1),3) + minutiae(suspectMinList(k,2),3);

  if typesum == 1

    if ridgeOrderMap(minutiae(suspectMinList(k,1),1),minutiae(suspectMinList(k,1),2) ) ==
ridgeOrderMap(minutiae(suspectMinList(k,2),1),minutiae(suspectMinList(k,2),2) )
      final(suspectMinList(k,1),1:2) = [-1,-1];
       final(suspectMinList(k,2),1:2) = [-1,-1];
    end;

  elseif typesum == 2
    if ridgeOrderMap(minutiae(suspectMinList(k,1),1),minutiae(suspectMinList(k,1),2) ) ==
ridgeOrderMap(minutiae (suspectMinList(k,2),1),minutiae (suspectMinList(k,2),2) )
      final (suspectMinList(k,1),1:2) = [-1,-1];
       final (suspectMinList(k,2),1:2) = [-1,-1];
    end;

  elseif typesum == 0
    a = minutiae(suspectMinList(k,1),1:3);
    b = minutiae(suspectMinList(k,2),1:3);
    if ridgeOrderMap(a(1),a(2)) ~=  ridgeOrderMap(b(1),b(2))

      [thetaA,pathA,dd,mm] = getLocalTheta(in,a,edgeWidth);
      [thetaB,pathB,dd,mm] = getLocalTheta(in,b,edgeWidth);


      thetaC = atan2( (pathA(1,1)-pathB(1,1)), (pathA(1,2) - pathB(1,2)) );


      angleAB = abs(thetaA-thetaB);
      angleAC = abs(thetaA-thetaC);


      if ( (or(angleAB < pi/3, abs(angleAB -pi)<pi/3 )) & (or(angleAC < pi/3, abs(angleAC - pi) < pi/3)) )
        finalList(suspectMinList(k,1),1:2) = [-1,-1];
        finalList(suspectMinList(k,2),1:2) = [-1,-1];
      end;

    elseif  ridgeOrderMap(a(1),a(2)) ==  ridgeOrderMap(b(1),b(2))
      finalList(suspectMinList(k,1),1:2) = [-1,-1];
      finalList(suspectMinList(k,2),1:2) = [-1,-1];

    end;
  end;
end;

  for k =1:numberOfMinutia
    if finalList(k,1:2) ~= [-1,-1]
      if finalList(k,3) == 0
        [thetak,pathk,dd,mm] = getLocalTheta(in,finalList(k,:),edgeWidth);
        if size(pathk,1) >= edgeWidth
           final_end=[final_end;[finalList(k,1:2),thetak]];
           [id,dummy] = size(final_end);
           pathk(:,3) = id;
           pathMap = [pathMap;pathk];
        end;
      else

        final_branch=[final_branch;finalList(k,1:2)];

        [thetak,path1,path2,path3] = getLocalTheta(in,finalList(k,:),edgeWidth);

        if size(path1,1)>=edgeWidth & size(path2,1)>=edgeWidth & size(path3,1)>=edgeWidth

        final_end=[final_end;[path1(1,1:2),thetak(1)]];
        [id,dummy] = size(final_end);
        path1(:,3) = id;
        pathMap = [pathMap;path1];

        final_end=[final_end;[path2(1,1:2),thetak(2)]];
        path2(:,3) = id+1;
        pathMap = [pathMap;path2];
        final_end=[final_end;[path3(1,1:2),thetak(3)]];
           path3(:,3) = id+2;
        pathMap = [pathMap;path3];

        end;


      end;
    end;
  end;
```

Alignment

```
    theta = real_end(k,3);
    if theta <0
      theta1=2*pi+theta;
      end;

      theta1=pi/2-theta;

    rotate_mat=[cos(theta1),-sin(theta1);sin(theta1),cos(theta1)];

    pathPointForK = find(ridgeMap(:,3)== k);
    toBeTransformedPointSet = ridgeMap(min(pathPointForK):max(pathPointForK),1:2)';

    tonyTrickLength = size(toBeTransformedPointSet,2);
    pathStart = real_end(k,1:2)';
    translatedPointSet = toBeTransformedPointSet - pathStart(:,ones(1,tonyTrickLength));

    newXY = rotate_mat*translatedPointSet;
```

Match

```
if or(edgeWidth == 0,isempty(edgeWidth))
  edgeWidth=10;
end;

if or(isempty(template1), isempty(template2))
  percent_match = -1;
else
length1 = size(template1,1);
minu1 = template1(length1,3);
real_end1 = template1(1:minu1,:);
ridgeMap1= template1(minu1+1:length1,:);

length2 = size(template2,1);
minu2 = template2(length2,3);
real_end2 = template2(1:minu2,:);
ridgeMap2= template2(minu2+1:length2,:);
```

```
ridgeNum1 = minu1;
minuNum1 = minu1;
ridgeNum2 = minu2;
minuNum2 = minu2;


max_percent=zeros(1,3);

for k1 = 1:minuNum1
    newXY1 = MinuOriginTransRidge(real_end1,k1,ridgeMap1);
  for k2 = 1:minuNum2

    newXY2 = MinuOriginTransRidge(real_end2,k2,ridgeMap2);

    compareL = min(size(newXY1,2),size(newXY2,2));
    eachPairP = newXY1(1,1:compareL).*newXY2(1,1:compareL);
    pairPSquare = eachPairP.*eachPairP;
    temp = sum(pairPSquare);

    ridgeSimCoef = 0;

    if temp > 0
    ridgeSimCoef = sum(eachPairP)/( temp^.5 );
    end;

if ridgeSimCoef > 0.8
      fullXY1=MinuOrigin_TransAll(real_end1,k1);
      fullXY2=MinuOrigin_TransAll(real_end2,k2);

      minuN1 = size(fullXY1,2);
      minuN2 = size(fullXY2,2);
      xyrange=edgeWidth;
      num_match = 0;

for i=1:minuN1
  for j=1:minuN2
    if (abs(fullXY1(1,i)-fullXY2(1,j))<xyrange & abs(fullXY1(2,i)-fullXY2(2,j))<xyrange)
      angle = abs(fullXY1(3,i) - fullXY2(3,j) );
      if or (angle < pi/3, abs(angle-pi)<pi/6)
      num_match=num_match+1;
      break;
      end;
    end;
  end;
end;

current_match_percent=num_match;
if current_match_percent > max_percent(1,1);
  max_percent(1,1) = current_match_percent;
  max_percent(1,2) = k1;
  max_percent(1,3) = k2;
end;
num_match = 0;
```

```
end;
end;
end;

percent_match = max_percent(1,1)*100/minuNum1;
end;
```