PALM RECOGNITION FOR BIOMETRIC IDENTIFICATION AND SECURITY

AYUSH MADAN (071010)
JASDEEP SINGH BHATIA (071043)

Under the supervision of
Dr. VINAY KUMAR

विद्या तत्त्व ज्योतिस्तम्
JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

May 2011

Submitted in partial fulfilment of the Degree of

Bachelor of Technology

Department of Electronics and Communication Engineering

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY
WAKNAGHAT (DISTT. SOLAN), HIMACHAL PRADESH

TABLE OF CONTENTS

S. No.		Topic	Page No.
		Certificate from the supervisor	iv
		Acknowledgement	v
		Summary	vi
		List of Figures	vii
1.		**Introduction**	**1**
	1.1	Objective	1
	1.2	Conception	2
	1.3	Advantages	2
	1.4	Few Definitions	2
	1.5	Block Diagram	4
	1.6	Work/Methodology	5
	1.7	Software used	5
2.		**Palm Extraction**	**6**
	2.1	Algorithm	9
	2.2	Code	9
	2.3	Results	12
3.		**Edge Detection**	**14**
	3.1	Algorithm	18
	3.2	Code	18
	3.3	Results	22

4. Pattern Formation 24

4.1 Number of Lines 24
4.1.1 Algorithm 25
4.2 Distance between the head and the heart line 25
4.2.1 Algorithm 25
4.3 Angle between the head and the heart line 26
4.3.1 Algorithm 27
4.3.2 Results 27
4.4 Area of the palm extracted 27
4.5 Code 28

5. Database Formation and Pattern Matching 32

5.1 Database Formation 32
5.1.1 Code 32
5.1.2 Explanation/Algorithm 33
5.2 Pattern Matching 35
5.2.1 Code 36
5.2.2 Results 39

6. Final Result 42

7. Conclusion 45

List of References 46

CERTIFICATE

This is to certify that the work titled **"PALM RECOGNITION FOR BIOMETRIC IDENTIFICATION AND SECURITY"**, submitted by **"AYUSH MADAN (071010)"** and **"JASDEEP SINGH BHATIA (071043)"** in partial fulfilment for the award of degree of Bachelor of Technology in Electronics and Communication Engineering of Jaypee University of Information Technology, Waknaghat has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor: [illegible]

Name of Supervisor: Dr. Vinay Kumar

Designation: Assistant Professor

Date: May 9, 2011

ACKNOWLEDGEMENT

"Nothing In This World Will Take Place Of Persistence. Talent Will Not; Nothing Is Uncommon Then Unsuccessful Man With Talent, Genius Will Not; Unrecorded Genius Is Almost A Proverb, Education Alone Will Not ; The World Is Full Of Derelicts, Persistence And Determination Are Omnipotent"

We wish to express our gratitude and indebtness to our project guide Dr. Vinay Kumar for his encouragement, guidance and valuable assistance which helped us to complete this project successfully.

We are at loss of words to express our deep gratitude towards our guide who made us realize the fact that stumbling blocks were in fact stepping stones to success. He motivated us to take this project as a challenge and come out with flying colours.

We would also like to thank Prof. Dr. Sunil Bhooshan and Mr. Mohammad Wajid for their valuable suggestions which helped us to complete the project work in time.

[illegible]

AYUSH MADAN

[illegible]

JASDEEP SINGH BHATIA

Date:- May 9, 2011

SUMMARY

In the contemporary world, we need to have efficient security systems in order to protect our valuables and assets. Also, in situations where an organisation may not want intruders in their premises; an office may need an automatic system to record the attendance of their employees, etc. We require systems that can distinguish an individual on the basis of some unique characteristics (such as a palm, a finger-print, eye retina, etc.) and hence decide whether to grant an access to the person. Biometrics is one of the emerging branch of study that deals in such kind of systems.

This project deals with the biometric identification and security and the features that we exploited for this purpose are the principal lines present on the palm of an individual.

We have developed a software which extracts the palm region form a human hand, calculates some parameters mathematically using principal lines that differentiate one palm from another and then verifies it against a database which contains the palms of the registered users. Then, based on the results obtained, it decides whether the individual should be allowed access to the system or not.

The code is in two parts. The first part is meant for Database Formation and the second for Pattern Matching . With the help of Database formation, one can register a user for an access to the system whenever he needs to do so and the code for pattern matching will be needed for comparing an input palm against all those present in the database.

AYUSH MADAN Dr. VINAY KUMAR
JASDEEP SINGH BHATIA

Date:- May 9, 2011

LIST OF FIGURES

S. No.	Title	Page No.
1.1	Principal lines	1
1.2	Block diagram to carry out palm recognition using principal lines	4
2.1	ROI	6
2.2	Results using Harris Detector	6
2.3	Approach for ROI	7
2.4	Palm extraction results	12
3.1	Edge detection results	15
3.2	Edge detection results after thresholding and thinning	22
3.3	Some other results	23
4.1	Number of lines	24
4.2	Distance calculation	25
4.3	Angle calculation	26
5.1	A view of the database	34
5.2	Match found(matrix present in database)	39
5.3	Match found(matrix present in database)	40
5.4	Match found(matrix present in database)	40
5.5	Match not found(matrix not present in database)	41
6.1	Process	42

CHAPTER 1: INTRODUCTION

1.1 OBJECTIVE

The objective of our project was to detect the palm pattern and then verify it against a pre registered pattern to authenticate an individual for access to a system thereby ensuring security of the system.

The design methods proposed till now are based on vein and ridge patterns. The problem with these techniques is that they produce anomalous results because of the change in vein and ridge patterns with time. In addition to that, they are very time consuming, difficult and costly to design[3].

We have developed and designed a software system which can be used to recognize palms for various purposes. The design we propose is based on the mathematical features present on the palm. The design methodology is based only on the principal lines of the hand. We have formed some definite patterns through the principal lines which are unique for each palm. Figure 1 shows the principal lines of a palm.

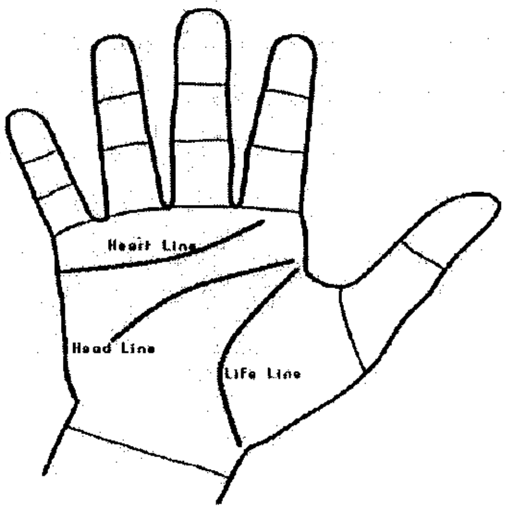


Figure 1.1: Principal lines

1.2 CONCEPTION

Palm recognition has become increasingly prevalent in modern biometric identification and verification systems. It is an approach which uses the biological features inherent in the palm of each individual. The basic idea behind authentication through palm recognition is that no two palms are alike[10]. As palms have both uniqueness and permanence, they can be used as a trusted form of identification .

Of all the biometrics studied, palm-print has an advantage over other biometrics such as voice and face recognition where uniqueness between people is doubtful or fingerprint and iris pattern where high-resolution images are required[16]. Palm-prints are unique between people and relatively low-resolution images will suffice. Moreover, palm is less prone to injuries. It is easy to use as it has fewer intrusions as compared to other scans. It is also resistant to spoofing.

Palm recognition technique can be used for security enhancement, authentication purposes and in some other large scale management systems.

1.3 ADVANTAGES

Iris and fingerprint recognition are the other two most commonly used biometric identification systems. Palm pattern has an edge over them as it is less prone to injuries. Also, palm pattern is more consistent as compared to change in facial features. It is more easy to use. Palm has less intrusions as compared to other scans especially when the detection is done only using the principal lines present on the palm[4]. It is important to note that the sensor cost for capturing palm features is very less as compared to other biometric systems.

1.4 FEW DEFINITONS

Image:

An image is an array, or a matrix, of square pixels (picture elements) arranged in columns and rows.[17]

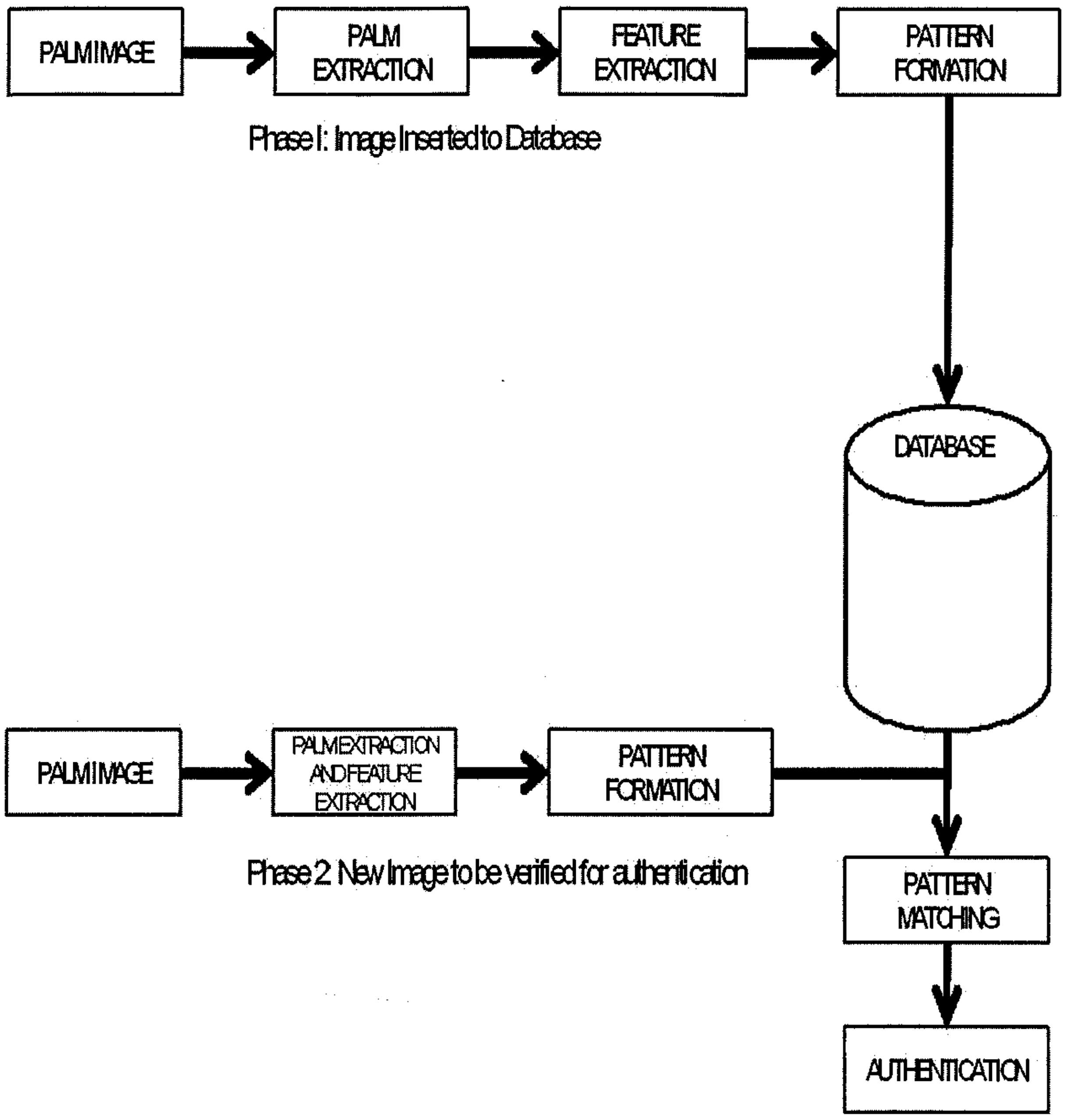
Point operations:

These operations map the pixel values without changing the size, geometry, or local structure of image. Every new pixel is dependent only on its previous pixel value, and not on any other neighbour value. [17]

Filter operations:

Filters use more than one pixel to calculate new pixel value. The new pixel value is at the same location as the reference point.[17]

1.5 BLOCK DIAGRAM



PALM IMAGE
PALM EXTRACTION
FEATURE EXTRACTION
PATTERN FORMATION
Phase I: Image Inserted to Database
DATABASE
PALM IMAGE
PALM EXTRACTION AND FEATURE EXTRACTION
PATTERN FORMATION
Phase 2: New Image to be verified for authentication
PATTERN MATCHING
AUTHENTICATION

Figure 1.2: Block diagram to carry out palm recognition using principal lines

1.6 WORK/METHODOLOGY

The first step was to get a pool of palm images. For uniformity, we obtained images of 448*336 pixels under similar light conditions.

The project has been divided into 4 separate modules.

1) Palm extraction
2) Edge detection
3) Pattern formation
4) Database formation and Pattern matching

All the modules of the project have been implemented and are discussed below.

1.7 SOFTWARE USED:-

- We have used MATLAB 7.1[21] in this project.

  MATLAB is a high-performance language for technical computing. It provides an easy-to-use and interactive environment. Coding becomes simpler. Inbuilt image processing functions.

- For database handling, we have used Microsoft MS-Excel 2003.

CHAPTER 2:- PALM EXTRACTION

The actual palm area is called the region of interest (ROI) as depicted in figure 2.1. We use the ROI to analyze a palm image and further obtain all the parameters characteristic to that palm.

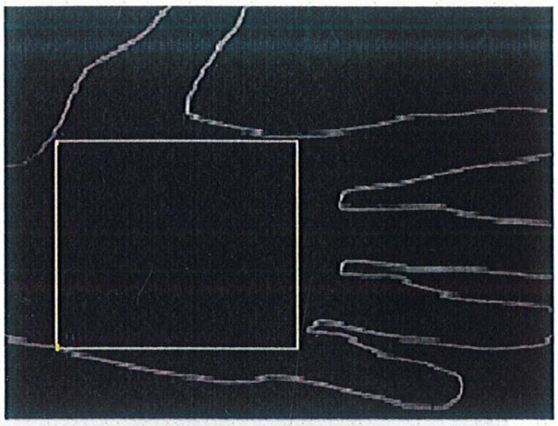


Figure 2.1: ROI

Initially, we tried to extract the palm from the scanned image itself. We tried out finding the discontinuities in the hand image using Harris corner detection[6]. From these discontinuities, we found out the ROI. The results obtained were neither consistent nor appropriate (Figure 2.2). The problems faced were that the detector did not produce uniform results and gave unwanted points of discontinuities.

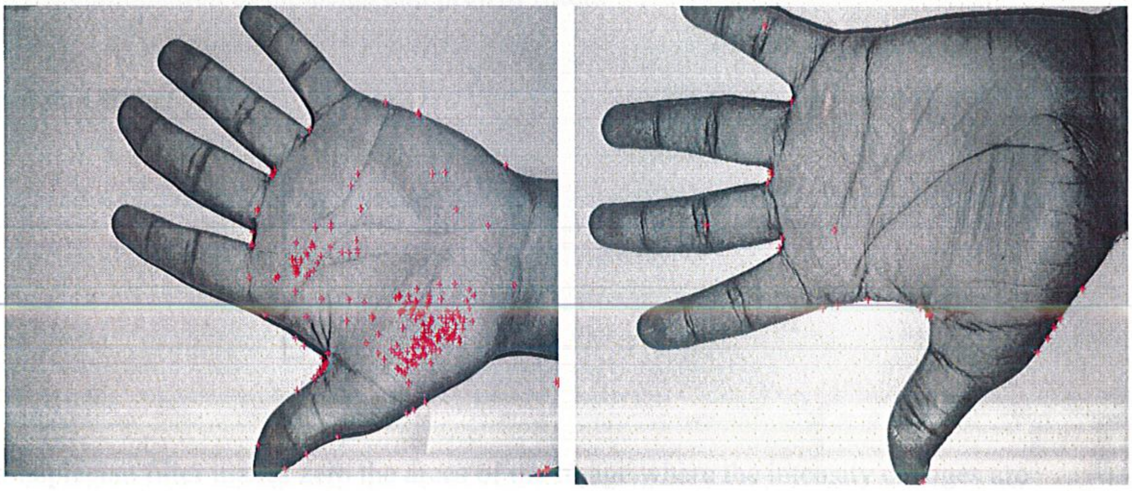


Figure 2.2: Results using Harris detector

After the failure of initial efforts, we tried an iterative approach on a binary image of the palm by dynamically increasing square matrix to find the largest black portion in binary form of the original image. Figure 2.3 illustrates this technique.

In this approach, we found the location of the pixel with co-ordinates (am, bm). These co-ordinates were used to extract the required black portion; which is essentially a square, consequently giving us the extracted palm. Since it is a dynamic approach, the ROI generated varies from sample to sample.

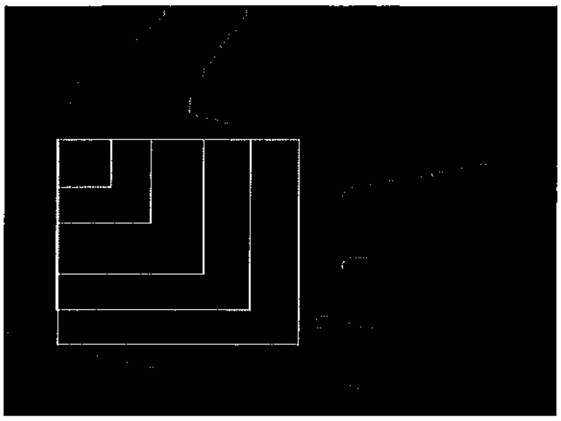


Figure 2.3: Approach for ROI

The foremost step in this approach was to get an outline of the hand for which we used the combination of laplacian and prewitt filters[11].

Laplacian filter:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

The Laplacian filter makes zero the areas of the image where the intensity changes are smooth. When the variation of the intensities is not smooth, the Laplacian returns non-zero values. The less smooth the variation of the intensities, the higher in absolute value the results of the Laplacian filter. The Laplacian filter also detects the edges of the image.

Consider a 1-dimensional image with three pixels with intensities a, b, c.

$$[\,a \quad b \quad c\,]$$

The intensity changes between a, b, c are given by the differences

b-a, c-b

The variation of the intensity changes is given by the difference

$$(c-b)-(b-a)=c-2b+a$$

This difference can be written as the convolution of the image

$$[\,a \quad b \quad c\,]$$

by the mask

$$[\,1 \quad -2 \quad 1\,]$$

In 2D images, the variation of the intensity changes is given by the the mask

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

for the horizontal direction, and the mask for the vertical direction.

The $\begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix}$ sum of these two masks, corresponding to the sum of the variations of intensity changes in horizontal and vertical direction, is the Laplacian mask[18].

2.1 ALGORITHM:-

a. Read Image.
b. Convert the RGB image to Grayscale image.
c. Apply a combination of operators (laplacian, prewitt) to get the binary palm region.
d. Define and initialize the variables **am, bm** equal to 0.
e. Define and initialize **sqrside, sqrsidem** equal to 0.
f. **Sidelimit** is the variable which is assigned a value of maximum size of a square that can be formed from any particular pixel.
g. **Sqrsidem** variable stores the value of the size of the maximum square formed till now.
h. The **sqrside** variable is changed dynamically in a pixel by pixel operation and the final result is stored in the **sqrsidem** variable.
i. A square with side equal to **sqrsidem** is constructed with starting pixel co-ordinates as **(am,bm)**.
j. These co-ordinates along with **sqrsidem** are passed as arguments to the 'imcrop'
   function which is applied on the original image.
k. The resulting image is the 'Extracted palm' which is the desired result.

2.2 CODE:-

The MATLAB code for the process of palm extraction is as follows:-

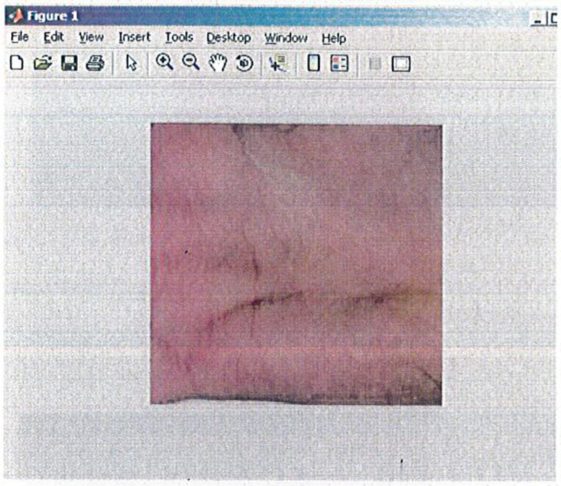
```
i=imread(image);
sizei=size(i);
i=i(1:sizei(1),1:sizei(2));
i=i(1:end,1:end);
```

```
k1=fspecial('laplacian');
l1=imfilter(i,k1);
t1=edge(l1,'prewitt',0.05,'vertical');
figure;
imshow(t1);
am=0;
bm=0;
sqrsidem=0;
sqrside=0;
x=0;
y=0;
brk=0;
a=1;
sidelimit=0;        %%The max size of a square that can be formed from a
%%particular pixel
while a<=336        %%it will take the lower value of the two factors (336- a),(448-b)
  b=1;
  while b<=448
    a
    b;
    brk=0;
    sqrside=1;
    if (336-a)<(448-b)     %%Which side would determine maximum size of  square
      sidelimit=336-a;
    else
      sidelimit=448-b;
    end
    while sqrside<sidelimit
      for x=a:(a+sqrside)
        for y=b:(b+sqrside)
          if t1(x,y)==1           %%Condition if white pixel encountered
            brk=1;
            if sqrside>sqrsidem     %% Check if the obtained square is the largest
                                    %%square till this iteration
```

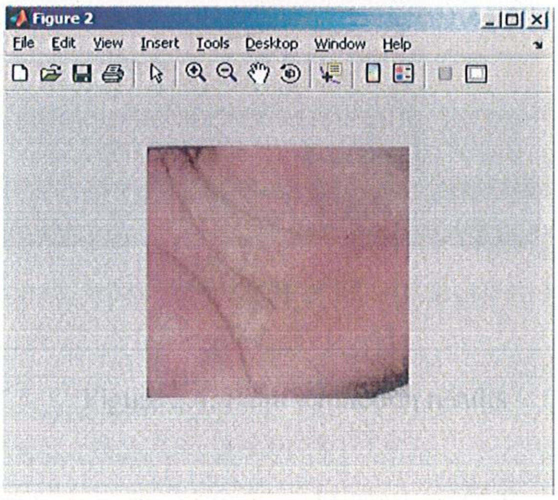
```
                    sqrsidem=sqrside;
                    am=a;
                    bm=b;
                  end
                end
                if brk==1
                  break
                end
              end
              if brk==1
                break
              end
            end
            if brk==1
              break
            end
            sqrside=sqrside+1;
          end
            b=b+1;
        end
        a=a+1;
      end
      sqrsidem
      am
      bm
      cwd = pwd;
      cd(tempdir);
      pack
      cd(cwd)
      I2 = imcrop(i,[bm am sqrsidem sqrsidem]);
       figure;
       imshow(I2);
```

('I2' image gives us the region of interest)

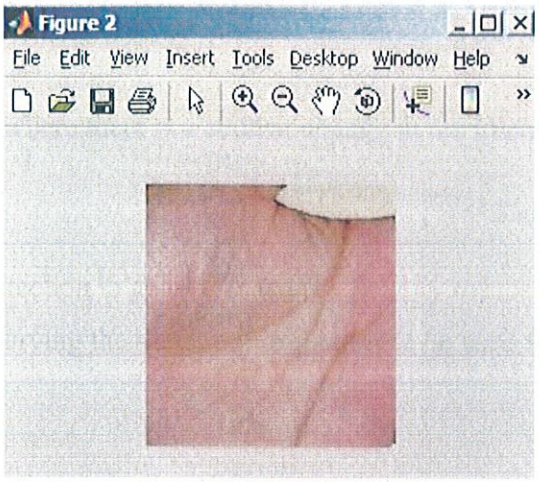
2.3 RESULTS:-



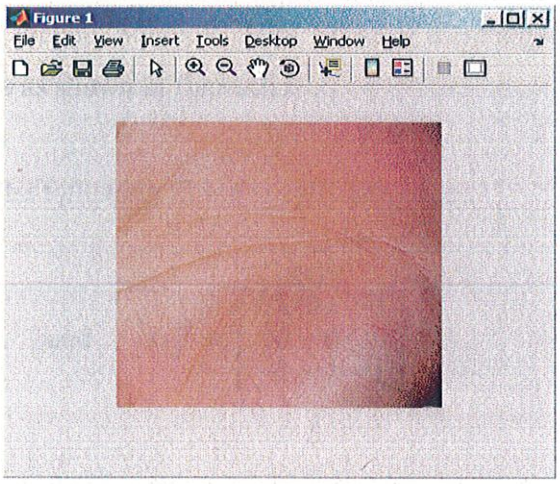
(a)



(b)



(c)



(d)

Figure 2.4: Palm extraction results

These results were highly accurate and we could move on to the next stage.

CHAPTER 3:- EDGE DETECTION

Edges are those points where there is a sudden change in the intensity value of the pixel[17] [19].

The purpose of edge detection in general is to significantly reduce the amount of data in an image, while preserving the structural properties to be used for further image processing.

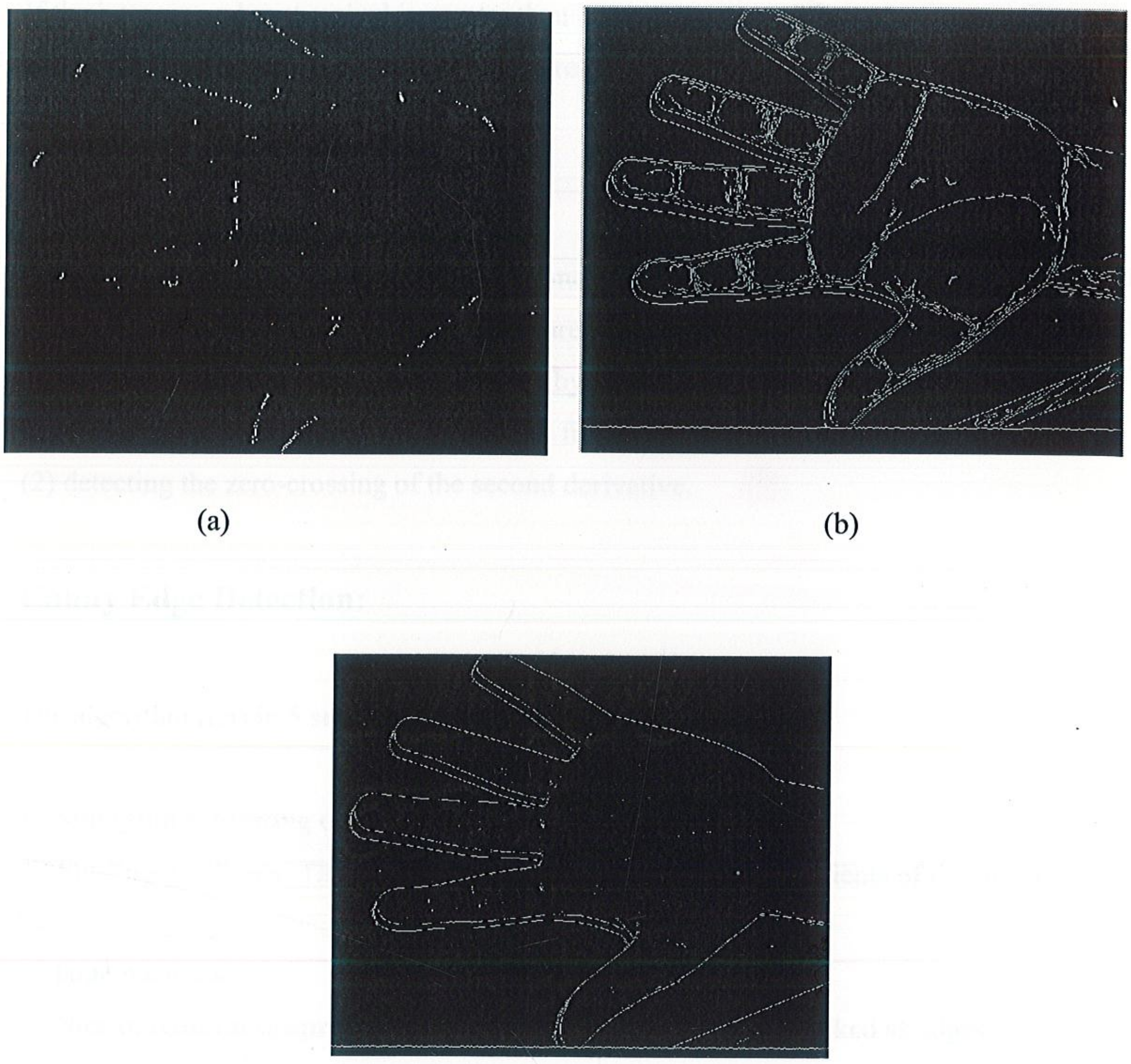
The **Edge Detection** process is based on the concept of detecting and extracting the principal lines. After extracting the palm (the region of interest) from a scanned image, we found out the edges from the palm which gave us the principal lines in most of the cases. We tried out many algorithms using a combination of various edge detectors; viz., Canny, Prewitt, Sobel, Roberts. Some of these algorithms provided us with anomalous results as shown in figure 3.1.

Prewitt operator uses the following filters to calculate edges:-

$$H_x^P = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad H_y^P = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Sobel operator uses the following filters:-

$$H_x^S = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad H_y^S = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$



(a) (b)

(c)

Figure 3.1: Edge detection results (unwanted)

But we were successful in devising an algorithm which gave us the desired results. In order to get the result which is independent of orientation, we used the concept of gradients[11] [13]. Gradients are used for calculating the strength of palm edges. After that, we used thresholding and thinning algorithms. With application of the above mentioned procedures, we were able to achieve positive results.

$$D_x(u,v) = H_x * I \quad \text{and} \quad D_y(u,v) = H_y * I$$

The local edge strength is defined as

$$E(u,v) = \sqrt{\left(D_x(u,v)\right)^2 + \left(D_y(u,v)\right)^2}$$

If the intensity value at a pixel is greater than edge strength, then that pixel is an edge and if intensity value is less than the edge strength, then that pixel is not an edge.

Edge detection using derivatives:

Calculus describes changes of continuous functions using derivatives. An image is a 2D function, so operators describing edges are expressed using partial derivatives. Points which lie on an edge can be detected by:
(1) detecting local maxima or minima of the first derivative.
(2) detecting the zero-crossing of the second derivative.

Canny Edge Detection:

The algorithm runs in 5 separate steps:

1. **Smoothing**: Blurring of the image to remove noise.
2. **Finding gradients**: The edges should be marked where the gradients of the image has
   large magnitudes.
3. **Non-maximum suppression**: Only local maxima should be marked as edges.
4. **Double thresholding**: Potential edges are determined by thresholding.
5. **Edge tracking by hysteresis**: Final edges are determined by suppressing all edges that are not connected to a very certain (strong) edge.

1. Smoothing:

To prevent that noise is mistaken for edges, noise must be reduced as it is inevitable that all images taken from a camera will contain no amount of noise. Therefore the image is first smoothed by applying a Gaussian filter.

2. Finding Gradients:

The Canny algorithm basically finds edges where the grayscale intensity of the image changes the most. These areas are found by determining gradients of the image. Gradients at each pixel in the smoothed image are determined by applying what is known as the Sobel-operator.

$$H_x^S = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad H_y^S = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

First step is to approximate the gradient in the x- and y-direction respectively by applying the kernels.

3. Non-maximum suppression:

The purpose of this step is to convert the "blurred" edges in the image of the gradient magnitudes to "sharp" edges. Basically this is done by preserving all local maxima in the gradient image, and deleting everything else.

4. Double thresholding:

The edge-pixels remaining after the non-maximum suppression step are (still) marked with their strength pixel-by-pixel. Many of these will probably be true edges in the image, but some may be caused by noise or color variations for instance due to rough surfaces. The simplest way to discern between these would be to use a threshold, so that only edges stronger that a certain value would be preserved. The Canny edge detection algorithm uses double thresholding. Edge pixels stronger than the high threshold are marked as strong; edge pixels weaker than the low threshold are suppressed and edge pixels between the two thresholds are marked as weak.

5. **Edge tracking by hysteresis (localization):**

Strong edges are interpreted as "certain edges", and can immediately be included in the final edge image. Weak edges are included if and only if they are connected to strong edges. The logic is of course that noise and other small variations are unlikely to result in a strong edge (with proper adjustment of the threshold levels).
Thus strong edges will (almost) only be due to true edges in the original image. The

weak edges can either be due to true edges or noise/color variations. The latter type will probably be distributed independently of edges on the entire image, and thus only a small amount will be located adjacent to strong edges. Weak edges due to true edges are much more likely to be connected directly to strong edges.

3.1 ALGORITHM:-

a. Read image.
b. Convert the RGB image into a grayscale image and set the colormap to gray.
c. Use $\mathbf{H_x}$ and $\mathbf{H_y}$ to calculate $\mathbf{D_x}$ and $\mathbf{D_y}$.
d. Calculate edge strength.
e. Calculate the level for thresholding.
f. Store the maximum of edge strength or threshold level in a new matrix **ibw**.
g. If **ibw(I,j) >th**
    - If the elements in $\mathbf{D_x}$ and $\mathbf{D_y}$ matrices have value greater than **ibw(I,j)**, then **(i,j)** position of matrix **itemp** is assigned value **imax**; i.e., maximum edge strength.
    - If the elements in $\mathbf{D_x}$ and $\mathbf{D_y}$ matrices have value less than **ibw(I,j)**, then **(i,j)** position of matrix **itemp** is assigned value **imin**; i.e., minimum edge strength.

h. If **ibw(I,j) <=th**
    - **(i,j)** position of matrix **itemp** is assigned value **imin.**

i. Thus **itemp** gives us the extracted edges.

3.2 CODE:

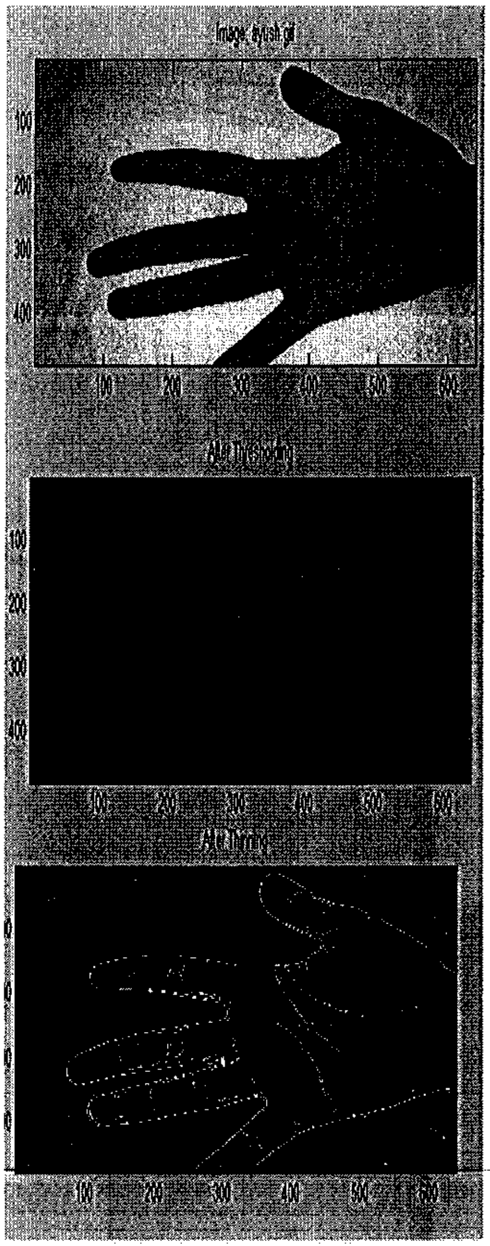
```
Clear all;
i=imread(image);
```

```
i=i(1:end,1:end);
[x,map]=imread(image);
w=ind2gray(x,map);
figure(1);colormap(gray);
subplot(3,2,1);
imagesc(w);
title('Image: image');

x1=10;sdx1=1;x2=10;sdx2=1;Theta1=pi/2;
y1=10;sdy1=1;y2=10;sdy2=1;Theta2=0;
beta=0.08;

subplot(3,2,2);
hx=d2fun(x1,sdx1,x2,sdx2,Theta1);
dx= conv2(w,hx,'same');
imagesc(dx);
title('dx');
subplot(3,2,3)
hy=d2fun(y1,sdy1,y2,sdy2,Theta2);
dy=conv2(w,hy,'same');
imagesc(dy);
title('dy');

subplot(3,2,4);
es=sqrt(dx.*dx+dy.*dy);
imagesc(es);
title('Norm of Gradient');

imax=max(max(es));
imin=min(min(es));
th=beta*(imax-imin)+imin;
subplot(3,2,5);
ibw=max(es,th.*ones(size(es)));
imagesc(ibw);
```

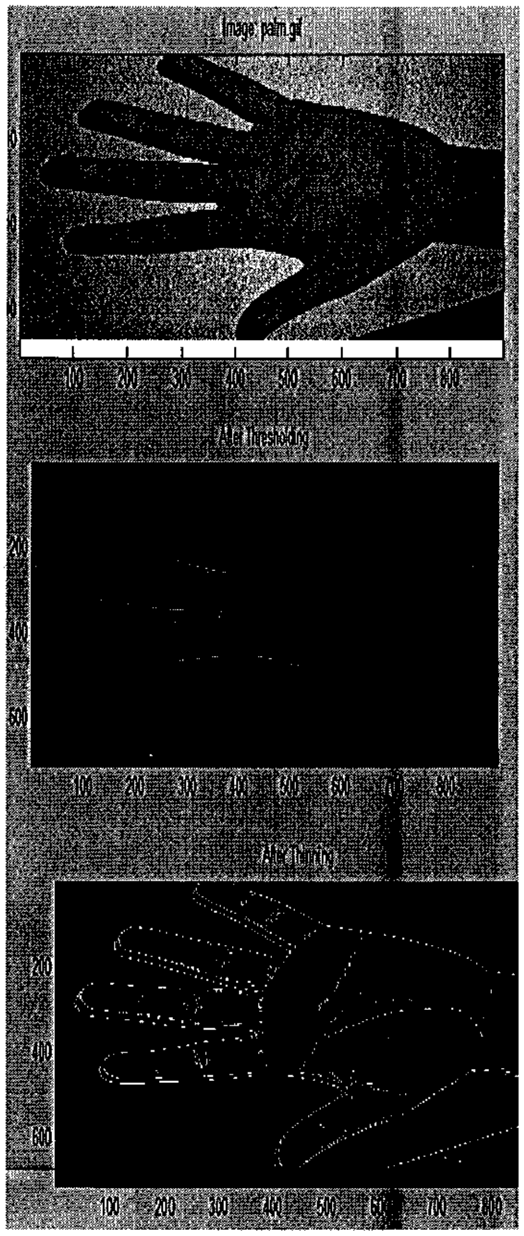
```
title('After Thresholding');

subplot(3,2,6);
[n,m]=size(ibw);
for i=2:n-1,
  for j=2:m-1,
     if ibw(i,j) > th
           fx=[-1,0,+1;-1,0,+1;-1,0,+1];
           fy=[-1,-1,-1;0,0,0;+1,+1,+1];
           Z=[ibw(i-1,j-1),ibw(i-1,j),ibw(i-1,j+1);
          ibw(i,j-1),ibw(i,j),ibw(i,j+1);
          ibw(i+1,j-1),ibw(i+1,j),ibw(i+1,j+1)];
          XI=[dx(i,j)/es(i,j), -dx(i,j)/es(i,j)];
          YI=[dy(i,j)/es(i,j), -dy(i,j)/es(i,j)];
          ZI=interp2(fx,fy,Z,XI,YI);
        if ibw(i,j) >= ZI(1) & ibw(i,j) >= ZI(2)
itemp(i,j)=imax;

          else
               itemp(i,j)=imin;
          end
      else
          itemp(i,j)=imin;
      end
  end
end

imagesc(itemp);
title('Extracted edges');
colormap(gray);

%%%end of main file
%%Functions
```

```
function l = d2fun(n1,sd1,n2,sd2,theta)
r=[cos(theta)-sin(theta);
  sin(theta)  cos(theta)];
for i = 1 : n2
   for j = 1 : n1
      u = r * [j-(n1+1)/2 i-(n2+1)/2]';
      l(i,j) = fun(u(1),sd1)*dfun(u(2),sd2);
   end
end

l = l / sqrt(sum(sum(abs(l).*abs(l))));

function y = fun(x,std)
y = exp(-x^2/(2*std^2)) / (std*sqrt(2*pi));

function y = dfun(x,std)
y = -x * fun(x,std) / std^2;
```

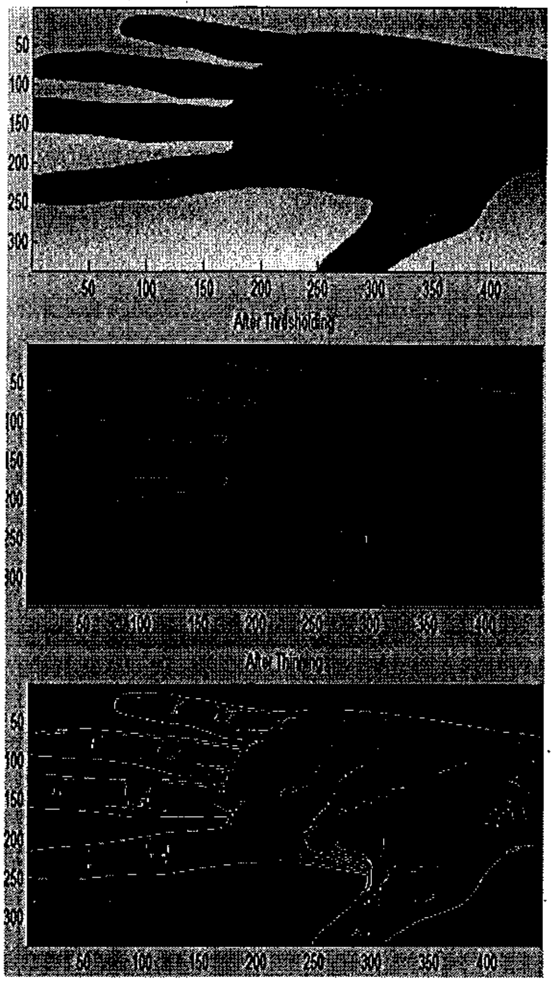
3.3 RESULTS:-



(a)

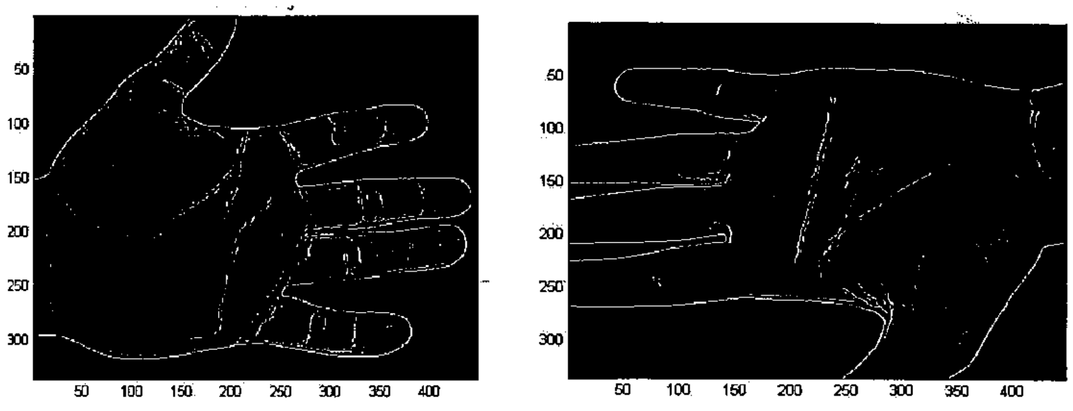


(b)



(c)

Figure 3.2: Edge detection results after thresholding and thinning



(a) (b)

Figure 3.3: Some other results

CHAPTER 4:-PATTERN FORMATION

The next phase of project was to decide and hence calculate the parameters[8] which would uniquely identify a palm.

The criteria which have been used to recognize the uniqueness of the palms are as follows:-

1. The number of lines extracted during edge extraction.
2. The perpendicular distance between the head line and the heart line.
3. The angle between lines formed by head line and heart line.
4. The area of the palm extracted.

All the four criteria will be used in pattern matching and therefore it is important for each of these parameter to match with the required value, else there will be a negative match.

4.1 Number of Lines :-

The number of lines extracted after the edge detection (chapter 3) process are calculated. This value is generally between 1 and 5. It has been observed that palm samples may have different number of prominent lines. This can be used as a distinguishing feature.

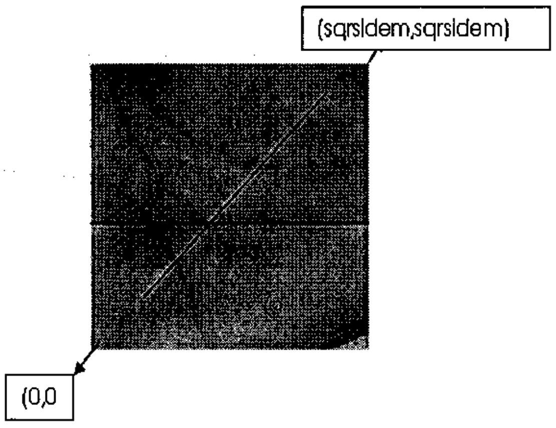


Figure 4.1: Number of lines

4.1.1 ALGORITHM to calculate number of lines:

1. Perform palm extraction and edge detection.
2. Draw a line between (0,0) and (sqrsidem, sqrsidem) as shown is figure 4.1.
3. Check for the points where this line intersects with black pixels.
4. Store these points in a separate array.
5. The number of entries in this array gives the number of lines on the palm.

4.2 Distance between the Head line and the Heart line:-

The next step was to calculate the distance between the head line and the heart line. This is also a very useful measure as the distance between the head and the heart line is also one of the unique feature of every palm. This distance is calculated in pixels which gives us a fairly sufficient amount of sample space.

The numeric value of distance obtained generally stood between 20 to 70 pixels.

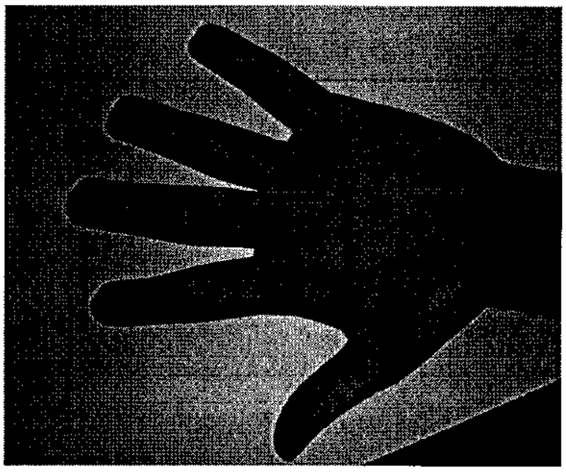


Figure 4.2: Distance Calculation

4.2.1 ALGORITHM to calculate distance:

1. Extract ROI.
2. Traverse to half the length in vertical direction.
3. Draw a line perpendicular to vertical axis from that point.

4. This line intersects the heart line and head line at two points, say A and B respectively.
5. We calculate the distance, d between points A(x1,y1) and B(x2,y2) using distance formula.

d=sqrt[(x1-x2)2+(y1-y2)2]

This distance is measured in pixels.

4.3 Angle between the head line and the heart line:-

If the head line and the heart line are extrapolated, they intersect at some point and the angle at the point of intersection can be used as a distinguishing feature.

This value generally between 0 to 10 degrees.

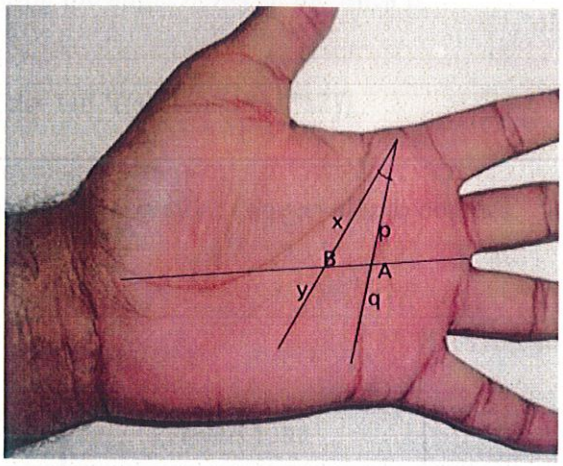


Figure 4.3: Angle calculation

Referring to figure 4.3:

a. The principal axis intersects the heart line and the head line, say at points A and B respectively.
b. Points P and Q lie on the heart line whereas X and Y lie on the head line. P & Q are equidistant from point A and X & Y are equidistant from point B.
c. We form a line using points P and Q and another line using points X and Y.

d. Then, finally we find the angle between 2 lines obtained. This angle shall work as the distinguishing factor between palms.

Initially, in order to obtain points P, Q, X and Y, we tried to form a circle at points A and B which would intersect the principal lines at equidistant points from the axis[1]. We couldn't proceed on this logic. Alternatively, we drew two lines parallel and close to the principal axis that gave us the required points.

4.3.1 ALGORITHM:

- Using the 'Mid+10' and 'Mid-10' coordinates, we found the slope of the lines which cut the heart and head line respectively.
- Then using the slopes obtained, we found the angle at which the lines cut each other.

The formula for calculating the angle between the lines is

$$th = \tan^{-1}((s1-s2)/(1+s1*s2))$$

The th value is in radians and is therefore converted into degrees.

4.3.2 RESULTS:

Point	Heart Line		Head Line	
	X axis	Y axis	X axis	Y axis
Mid	27	76	53	76
Mid + 10	36	86	66	86
Mid-10	15	66	46	66

4.4 Area of the palm extracted[9]:-

This parameter is to be used in worst cases wherein all the previous parameters give same results (highly unlikely). The area of palm region is calculated by taking square of variable sqrsidem[1]. Before saving this value in the matrix, it is divided by thousand for simplicity of the matrix. The unit of area is pixel*pixel.

Final matrix:

All the four parameters obtained for a sample palm are stored in a matrix format which is further saved in a database. The matrix is unique for each palm. The format of this matrix is shown in the DATABASE FORMATION which is the next part of the project.

4.5 CODE:

The MATLAB code for the process of pattern formation is as follows:-

```
clear all;
clc;
i=imread(image2);           %%obtained after edge detection
i=i(1:end,1:end,1);          %%converts image to 2-d
am;                          %%value obtained from previous code
bm;                          %%value obtained from previous code
sqrsidem;                                        %%value obtained from
previous code
 BW = imcrop(i,[bm am sqrsidem sqrsidem]);        %%extracted palm
 %imshow(I2)
%figure;
'area='
area=sqrsidem*sqrsidem;  %%area of extracted palm
area=area/1000
imshow(~BW)

%figure;
```

[1] sqrsidem is a variable used to find the length of the square during palm extraction module.

```
%imshow(I2);
if mod(sqrsidem,2)==0
x=(sqrsidem/2)-0;
else
x=((sqrsidem+1)/2);
end
b=1;


p=1;
% i3=(1:10);

while (b<=sqrsidem)  %%from mid

if BW(x,b)==1


     i3(p)=b;
     p=p+1;
  end
b=b+1;


end
p=p-1;
'main points'
 while(p>0)              %%prints points
   i3(p)
   p=p-1;
 end
'distance='
l=i3(2)-i3(1);           %%distance between head line and heart line
l
y=x+10;
d=1;
 k=1;
% i3=(1:10);
```

```
 [v,r]=size(i3);
 disp('Number of lines = ');
 r
 while (k<=sqrsidem)    %%from mid+10
   if BW(y,d)==1
      i4(k)=d;
       k=k+1;
   end

if d==153
         break;
else
   d=d+1;
end
 end
 k=k-1;

'main+10 points'
 while(k>0)        %%prints points
   i4(k)
 k=k-1;
 end
z=x-10;
e=1;
 f=1;
% i3=(1:10);

 while (f<=sqrsidem)           %%for mid-10
   if BW(z,e)==1
            i5(f)=e;
      f=f+1;
end
if e==153
         break;
```

```
else
        e=e+1;
end
 end
 f=f-1;

 'main-10 points'
 while(f>0)        %%prints points
   i5(f)

   f=f-1;
 end
slope1=((i4(2)-i5(2))/(y-z));
slope2=((i4(1)-i5(1))/(y-z));

th=abs((slope1-slope2)/(1+(slope1*slope2)));        %%calculation of slope
theta=atan(th);
deg=theta*180/pi                          %%conversion from radians to degrees
final=[];

'Final matrix is as follows:'
 final=[r,l,deg,area]              %%final matrix
```

CHAPTER 5:- DATABASE FORMATION AND PATTERN MATCHING

Having completed the first three phases of our project successfully, we started working on the database formation and pattern matching.

5.1 DATABASE FORMATION

In the process of database formation, the objective was to create a database wherein we could store the parameters (calculated in the pattern formation process) of a particular palm. The database could be an MS Excel sheet, a text file, etc. We used an Excel Sheet in our project.

We wrote a MATLAB code for storing all the parameters obtained for any palm in an excel sheet where data for each palm occupied one row and four columns in the sheet. The process was repeated for every palm that had to be stored in the database.

Number of Lines	Distance Between the head and the heart line (in Pixels)	Angle (in degrees)	Area (in thousands)

We put the parameters obtained for an image after the calculations that were done in the pattern formation process, in a matrix (1*4) form, in a format shown above and this matrix is saved in a database in the same format.

5.1.1 CODE:-

The snippet of MATLAB code for the process of database formation is as follows:-

```
'Final matrix is as follows:'
final=[r,l,deg,area]

p=final
a=xlsread('try3.xls','Sheet6')
[v,x]=size(a);
v=v+1;
q=num2str(v);
kul=strcat('A',q);
d=xlswrite('try3.xls',p,'Sheet6',kul)
```

The code of database formation[20][21] is preceded by the codes for palm extraction, feature extraction and pattern formation. This was done because when this project will be realised, the algorithm for database formation will be burnt on the device that will be used by the administrator and for the database formation this device will need the codes for prior stages of palm extraction, feature extraction and pattern formation. (Any new user who wishes to access to the system, has to first go to the administrator and get his palm registered. All the other devices will have only the algorithm for matching and authentication).

5.1.2 EXPLANATION:-

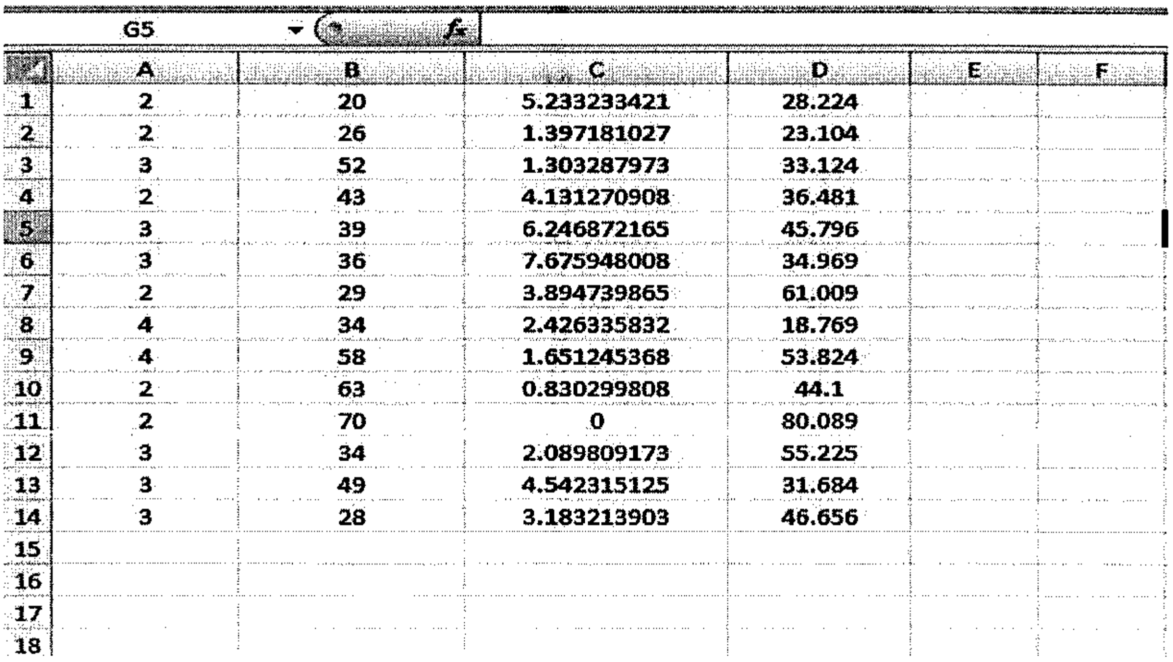
We put the results obtained after the pattern formation into a matrix named final. This matrix is stored in another matrix p so that the original results are not tampered with during the process.
We opened an MS Excel file and saved it with a name 'try3.xls'. The next thing we did was to read a particular sheet from this excel file with the help of a MATLAB instruction – xlsread.

```
a=xlsread('try3.xls','Sheet6')
```

We required the size of the data stored in the sheet and the same was obtained as [v,x]

where v and x represented the number of rows and columns respectively.
Everytime the code is run for storing the data for an image, the row index has to be incremented so that the new data goes to the next row. Hence v was incremented in the next step.
In order to write the data onto the sheet, we used a MATLAB instruction – xlswrite. The syntax of this instruction has the name of the excel file, the sheet number on which the data has to be written and the index for the location from which the process of writing starts. This location had to be variable since everytime the new data comes, it had to go into the next row. So, we needed the locations such as A1,A2,A3, etc. , to indicate the next row. Now, while implementing this, we observed that the xlswrite didn't read the numbers in the index part. So we had to first convert the row number v into a string and then merge it with 'A' to obtain the complete address for a row in string format. 'kul' is the variable in which this address was stored, and it was ultimately put in the xlswrite statement to achieve our objective.

Hence, we obtained a database for the palms that must be granted an access to the system. Now, in the pattern matching process, the input palm will be verified against the data stored in this database and will be granted (or rejected) an access to the system accordingly.
Here is a view of database formed:-



G5

	A	B	C	D	E	F
1	2	20	5.233233421	28.224		
2	2	26	1.397181027	23.104		
3	3	52	1.303287973	33.124		
4	2	43	4.131270908	36.481		
5	3	39	6.246872165	45.796		
6	3	36	7.675948008	34.969		
7	2	29	3.894739865	61.009		
8	4	34	2.426335832	18.769		
9	4	58	1.651245368	53.824		
10	2	63	0.830299808	44.1		
11	2	70	0	80.089		
12	3	34	2.089809173	55.225		
13	3	49	4.542315125	31.684		
14	3	28	3.183213903	46.656		
15						
16						
17						
18						

Figure 5.1: A view of the database

Now that we had obtained a database, the next step in this last phase of our project was to develop an algorithm for pattern matching.

5.2 PATTERN MATCHING

Pattern matching is about comparing the palm of the user trying to access the system with those stored in the database and decide whether the user should be authenticated or not (or in simple words whether he should be allowed to access the system or not).

Initially we made an algorithm wherein we performed a row-wise comparison[12]; i.e., we compared the parameters of input image with every row. There was a problem with this approach - the worst case in such an algorithm would have been a situation in which the required row exists towards the end of the database and in such a case number of calculations increased drastically, delaying the results unnecessarily.

In order to overcome this inefficiency, we devised an algorithm in which we performed a column-wise comparison. In this approach, the first element of the input matrix is compared with the first column of the database and where there is a match, we compare the second element with the corresponding second column and so on.

For example, let us say input matrix is [1 2 3 4]. This first element of the input matrix i.e. 1 is compared with the first element of all the rows in the database. Now, we kept all those rows in which the result for the matching was positive for further calculations and the rest of the entries were dropped. Then we picked up the second element i.e. 2 and matched with the second element of all those rows which were left after the first matching process. Again, we did the same thing and hence the process went on.

The advantage this approach has over the earlier one is that the number of calculations is reduced substantially.

5.2.1 CODE:-

The code snippet that will be executed at the time of pattern matching is as follows:-

```
'Final matrix is as follows:'

final=[r,l,deg,area]



% the pattern matching process starts here

final
matr=[];
matr=final;
matr
class(matr)
ja=0;
flag=1;
la=[];
ma=[];
na=[];
ba=[];
ca=[];
sa=[];
wa=[];
counta=0;
count1a=0;
count2a=0;
numa=xlsread('try3.xls','Sheet6');
class(numa)

[va,xa]=size(numa);
```

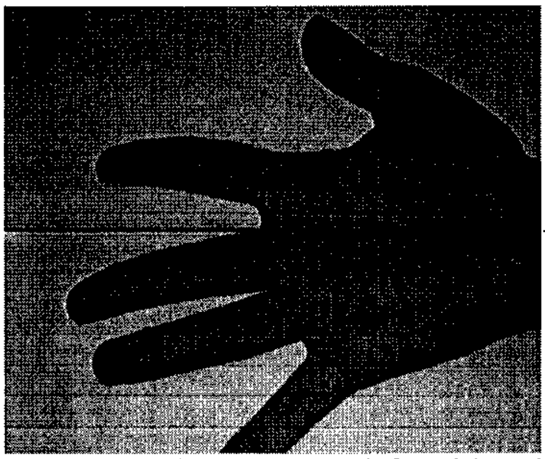
```
ja=numa(:,1);
for qa=1:va
  if matr(1)==ja(qa)
     flag=1;
     counta=counta+1;
     ja(qa)
     la(counta,:)=numa(qa,:);
  else
     flag=0;
  end
end

[va,xa]=size(la);

if va~=0
  ma= la(:,2);

  for qa=1:counta
     if matr(2)==ma(qa)
       flag=1;
       count1a=count1a+1;
       na(count1a,:)=la(qa,:);
       ma(qa)
     else
       flag=0;
     end
  end

end
count1a

[va,xa]=size(na);
va
```

```
if va~=0
  ba=na(:,3);
  for qa=1:count1a
     'matr3'
     matr(3)
     class(matr)
     'baqa'
     ba(qa)
     class(ba)
     if matr(3)==ba(qa)
        flag=1;
        count2a=count2a+1;
        'count2a'
        count2a
        ca(count2a,:)=na(qa,:);
        ba(qa)
     else
        flag=0;
     end
  end
end

[va,xa]=size(ca);

if va~=0
  sa=ca(:,4);
  for qa=1:count2a
    if matr(4)==sa(qa)
       flag=1;
       wa(1,:)=ca(qa,:);
       sa(qa)
    else
       flag=0;
    end
  end
end
wa
[va,xa]=size(wa);
if va~=0
   disp('Match Found');
   disp('You are authenticated');
   msgbox('Match Found!!! You are authenticated','RESULT')
else
   disp('Sorry, No Match Found');
   msgbox('Sorry, No Match Found','RESULT')
end
```

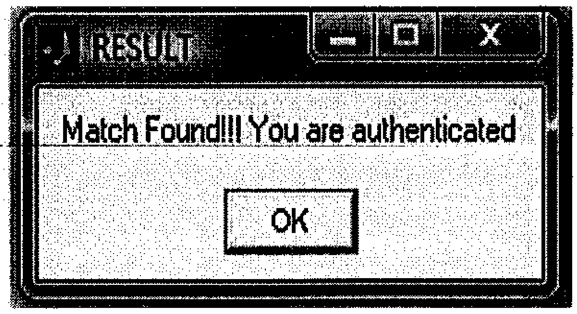
The device which will perform pattern matching will have the code for palm extraction, feature extraction and pattern formation alongwith the code for pattern matching. This is because any such device will need to calculate the parameters for the input palm first and only then it will compare the obtained parameters with the database and hence decide whether the user must be granted an access or not.

5.2.2 RESULTS:-

The Results obtained for some palms after the process of pattern matching were as :-



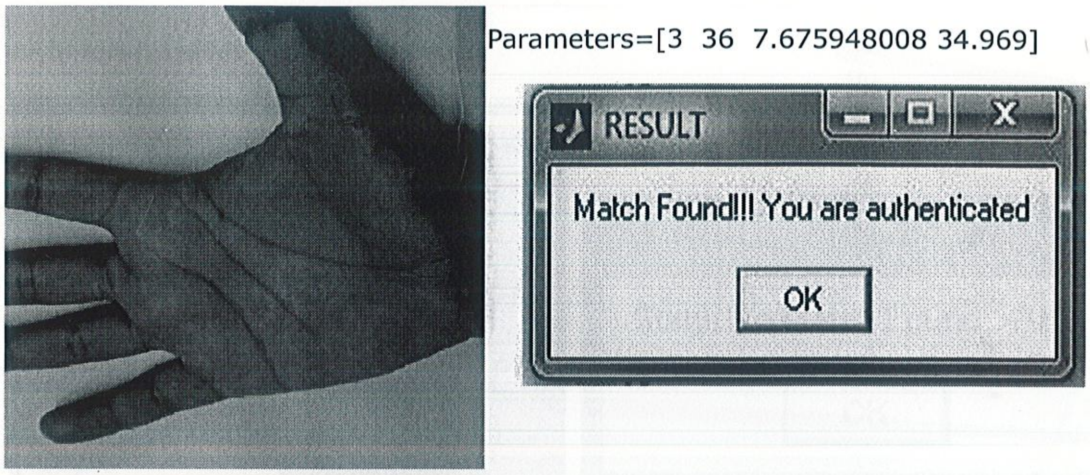
Parameters = [2 26 1.397181027 23.104]



Match Found!!! You are authenticated

Figure 5.2: Match found (matrix present in database)

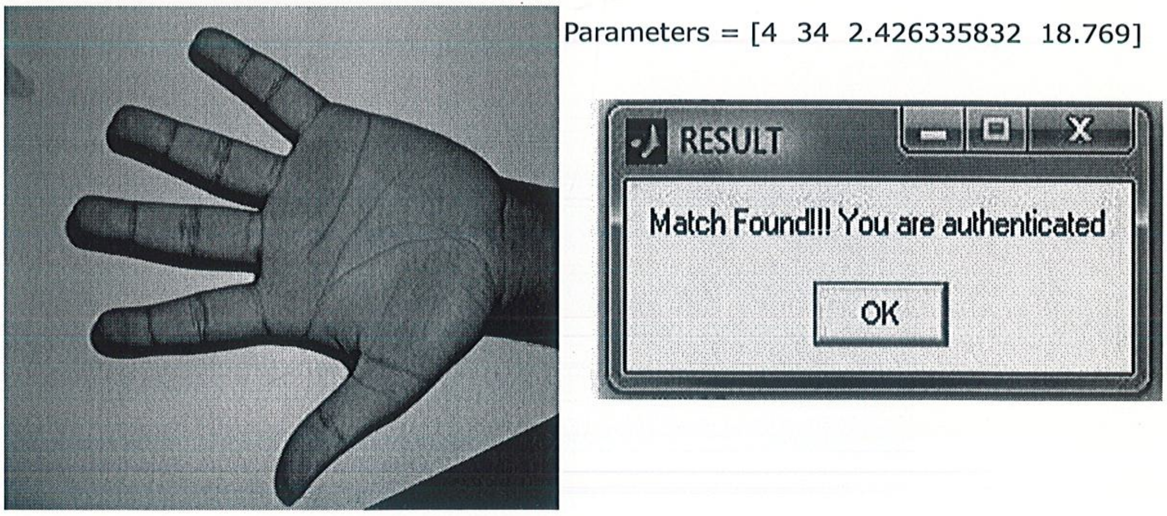
Parameters=[3 36 7.675948008 34.969]



Match Found!!! You are authenticated

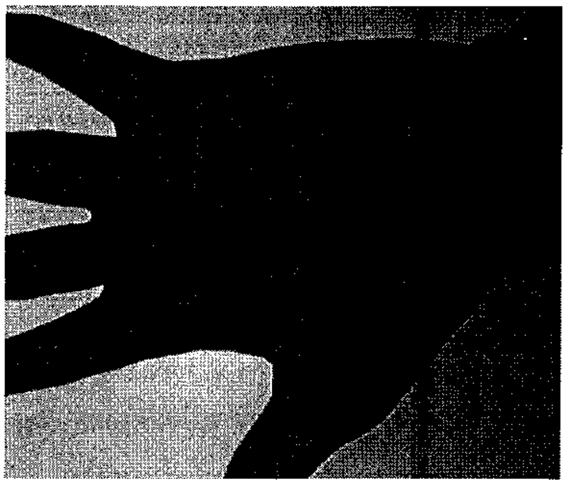
Figure 5.3:Match found (matrix present in database)

Parameters = [4 34 2.426335832 18.769]

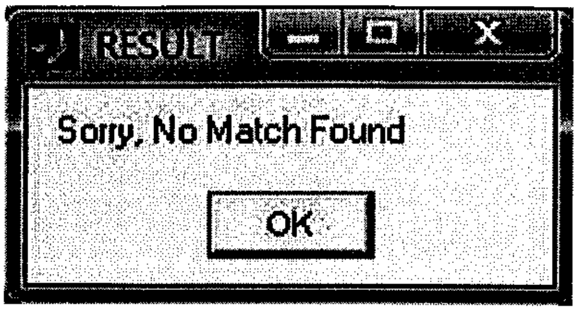


Match Found!!! You are authenticated

Figure 5.4: Match found (matrix present in database)



```
Parameters = [2  22  2.471216795  29.241]
```



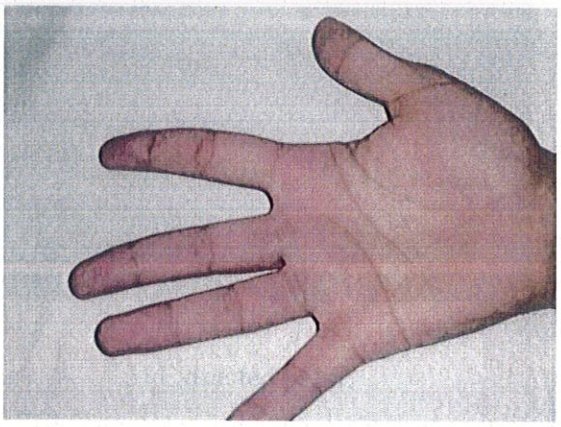
RESULT

Sorry, No Match Found

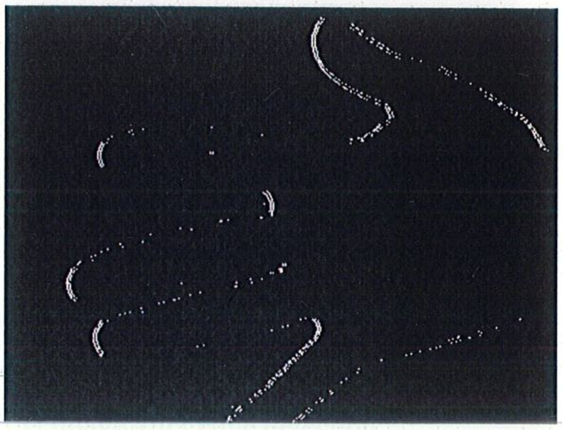
Figure 5.5: Match not found (matrix not present in database)

CHAPTER 6:- FINAL RESULT

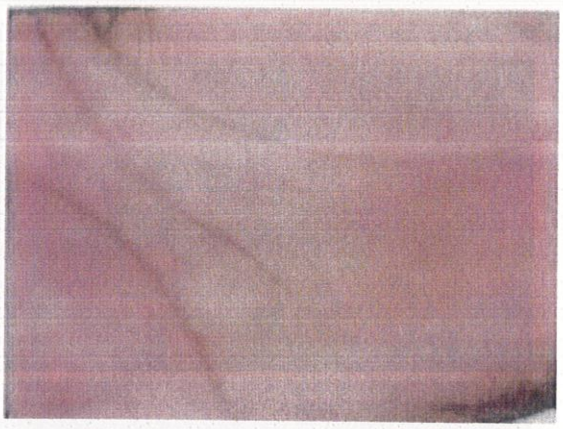
Below are all the results obtained during the execution of this project for one particular palm.



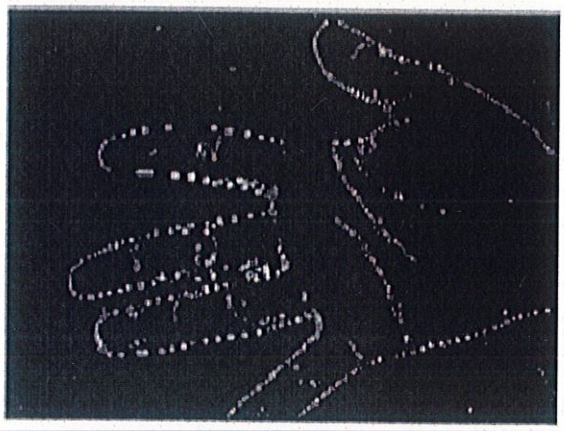
1) Input image



2) Contour of image



3) Extracted palm



4) Image after edge detection

5) Extracted palm after edge detection

Figure 6.1: Process

FINAL MATRIX:

Number of lines	Distance between head line and heart line (pixels)	Angle between head line and heart line (degrees)	Area of palm (pixel*pixel) (in thousands)
2	26	1.397181027	23.104

CHAPTER 7:- CONCLUSION

All the criteria have been implemented and checked upon 40 sample palms. They cumulatively have produced different results on all the samples checked till now.

Minimum number of samples that can be worked out through our project has been estimated below:-

Number of lines : 1-4	-	4
Distance : 20-70	-	50
Angle 0-15.0000	-	1,50,000
Minimum number of samples	=	1,50,000*50*4
	=	3,00,00,000

So, a biometric identification system using our project can cater to approximately 3,00,00,000 users (palms).

Future Work:

The code can be optimized to reduce the execution time for each sample.
More parameters can be designed for more efficient results.

Application:

The possession of access control is very important when confidential, or sensitive information and equipment is to be secured. This project is meant to be used primarily for identity access management to authenticate users and grant or deny access rights to data or services provided by an organization. It can be used as an authorization system for front doors, banks, airports, hospital wards, storage areas and schools. With the help of such softwares, one can protect a system from unauthorised individuals (intruders, hackers) and objects. It can also be used for recording attendance in schools, colleges, offices, etc.

LIST OF REFERENCES

1) Amin, and S. Fischer. A document skew Detection Method Using the Hough Transform. Springer-Verlag London Limited. Pattern Analysis and Applications, 3:243-253, 2000.

2) Anil K. Jain, Arun Ross, and Salil Prabhakar. An introduction to Biometric Recognition. IEEE Trans. on circuits and Systems for Video Technology, 14:4-20, 2004.

3) Chris Roberts. Biometric technologies -palm and hand. 2006.

4) D. C. M. Shen, H. C. Jain, A. K. Kumar, A. Wong. Personal verification using palmprint and hand geometry biometric. Lecture Notes in Computer Science, pages 668-678, 1999.

5) Erderm Yoruk, Helin Dutagaci, and Bulent Sankur. Hand Biometrics. Image Vision Comput, 24:483-497, May 2006.

6) C. Harris and M.J. Stephens. A combined corner and edge detector. Alvey Vision Conference, pages 147–152, 1988

7) Zhenan Sun, Tieniu Tan, Yunhong Wang, and Stan Z. Li. Ordinal palmprint representation for personal identification. In Proc. IEEE Computer Vision and Pattern Recognition CVPR, pages 279-284, 2005.

8) R. Zunkel, Hand Geometry Based Verification, in □2□, pp. 87-101.

9) C. Poon, D. C. M. Wong, H. C. Shen, A New Method in Locating and Segmenting Palmprint into Region-of-Interest, IEEE, ICPR Volume-04, 2004, pp. 533-536.

10) C-C. Han, H-L. Cheng, C-L. Lin, K-C. Fan, Personal Authentication Using Palmprint Features, Pattern Recognition 36, 2003, pp. 371-381.

11) D. Maio, D. Maltoni, Minutiae Extraction and Filtering from Grey-Scale Images, in L.C. Jain, U. Halici, I. Hayashi, S. B. Lee, S. Tsutsui (eds.), Intelligent Biometric Techniques in Fingerprint and Face Recognition, CRC Press, 1999, pp. 155-192.

12) N. Duta, A. K. Jain, K. V. Mardia, Matching of Palmprints, Pattern Recognition Letters, Vol. 23, No. 4, February 2002, pp. 477-485.

13) T. Pavlidis, "Algorithms for Graphics and Image Processing", Springer Verlag, Berlin, 1982.

14) J. You, W. Li, and D. Zhang, "Hierarchical palmprint identification via multiple feature extraction," *Pattern Recognition.*, vol. 35, pp. 847-859, 2002.

15) A. K. Jain, S. Prabhakar, S. Chen, Combining Multiple Matchers for a High Security Fingerprint Verification System, Pattern Recognition Letters 20 (1999), 1371-1379.

16) Ajay Kumar, David C.M. Wong, Helen C. Shen, Anil K. Jain, Personal Verification using Palmprint and Hand Geometry Biometric, AVBPA, 2003, pp. 668-678.

17) Rafael C. Gonzalez and Richard E. Woods, Digital Image Processing, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1992.

18) Digital Image Processing, William K. Pratt, John Wiley and sons, 4th edition.

19) Fundamentals of Digital Image Processing, Anil K. Jain, Prentice Hall, 1st edition.

20) Digital Image Processing - An Algorithmic Approach, Madhuri A. Joshi, PHI Learning, 3rd edition.

21) Practical MATLAB Applications for Engineers, Misza Kalechman, Taylor & Francis, Inc., 1st edition.

BRIEF BIO DATA

AYUSH MADAN- Department of Electronics and Communication Engineering
Jaypee University of Information Technology,
Waknaghat, Solan (H.P)

Pursuing	**Name of School/University**	**Year**	**CGPA**
B.Tech (ECE)	Jaypee University Of Information Technology, Solan (H.P.)	2011	8.8 (88%) (Up Till 7th sem)

Currently working on the **Palm Recognition for Biometric Identification and Security.**

JASDEEP SINGH BHATIA- Department of Electronics and Communication Engineering
Jaypee University of Information Technology,
Waknaghat, Solan (H.P)

Pursuing	**Name of School/University**	**Year**	**CGPA**
B.Tech (ECE)	Jaypee University Of Information Technology, Solan (H.P.)	2011	8.8(88%) (Up Till 7th sem)

Currently working on the **Palm Recognition for Biometric Identification and Security.**