IRIS RECOGNITION SYSTEM

Project Report submitted in partial fulfillment of the requirement for the

degree of

Bachelor of Technology

in

Electronics and Communication Engineering

under the Supervision of

Mr. PARDEEP GARG

By

ANKIT VERMA 101062
AMIT KUMAR 101128
JATIN GARG 101132

to

Jaypee University of Information and Technology

Waknaghat, Solan – 173234, Himachal Pradesh

<u>Certificate</u>

This is to certify that project report entitled **"IRIS RECOGNITION SYSTEM"**, submitted by **Ankit Verma, Amit Kumar, Jatin Garg** in partial fulfillment for the award of degree of Bachelor of Technology in Electronics and Communication Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Date: **Supervisor's Name:** Mr.Pardeep Garg

Assistant Professor

Acknowledgement

We take this opportunity to express our profound gratitude and deep regards to our mentor **Mr. Pardeep Garg**, for his guidance, monitoring and constant encouragement throughout the course of this project. The blessings, help and guidance given by him shall carry us a long way in the journey of our life. Also, we would like to express our thanks to Laboratory Assistant Dept. of ECE, **Mr. Mohan Sharma** who has supported and encouraged our efforts which have been instrumental in giving the project its shape.

Along with that we would like to express our thanks to Indian Institute of Technology, Delhi for providing us the iris database for the project.

Date:

Name of the students
Ankit Verma
Amit Kumar
Jatin Garg

Table of Contents

List of Figures

List of Tables

<u>ABSTRACT</u>

The objective of the project is to implement an open-source iris recognition system in order to verify the claimed performance of the technology.

Iris recognition analyzes the features that exist in the colored tissue surrounding the pupil, which has 250 points used for comparison, including rings, furrows, and freckles. Iris recognition uses a regular video camera system and can be done from further away than a retinal scan. It has the ability to create an accurate enough measurement that can be used for Identification purposes, not just verification.

The probability of finding two people with identical iris patterns is considered to be approximately 1 in 10^{52}(population of the earth is of the order 10^{10}). Not even one-egged twins or a future clone of a person will have the same iris patterns. The iris is considered to be an internal organ because it is so well protected by the eyelid and the cornea from environmental damage. It is stable over time even though the person ages. Iris recognition is the most precise and fastest of the biometric authentication methods.

The development tool used is MATLAB, and emphasis is on the software for performing recognition, and not hardware for capturing an eye image. The iris images were taken from IIT Delhi database, which was solely taken for research purpose.

CHAPTER - 1

BIOMETRIC TECHNOLOGY

1.1 Introduction

A biometric system provides automatic recognition of an individual based on some sort of unique feature or characteristic possessed by the individual. Biometric systems have been developed based on fingerprints, facial features, voice, hand geometry, handwriting, the retina and the one presented in this thesis, the iris.

Biometric systems work by first capturing a sample of the feature, such as recording a digital sound signal for voice recognition, or taking a digital color image for face recognition. The sample is then transformed using some sort of mathematical function into a biometric template. The biometric template will provide a normalized, efficient and highly discriminating representation of the feature, which can then be objectively compared with other templates in order to determine identity. Most biometric systems allow two modes of operation. An enrolment mode for adding templates to a database, and an identification mode, where a template is created for an individual and then a match is searched for in the database of pre-enrolled templates.

A good biometric is characterized by use of a feature that is; highly unique, stable and be easily captured and the chance of any two people having the same characteristic will be minimal. Also the feature does not change over time .In order to provide convenience to the user, and prevent misrepresentation of the feature.

1.2 Advantages of using biometrics

- Easier fraud detection
- Better than password/PIN or smart cards
- No need to memorize passwords
- Requires physical presence of the person to be identified
- Unique physical or behavioral characteristic
- Cannot be borrowed, stolen, or forgotten
- Cannot leave it at home

1.3 Types of biometrics

Physical Biometrics

- Finger print Recognition
- Facial Recognition
- Hand Geometry
- IRIS Recognition
- DNA

Behavioral Biometrics

- Speaker Recognition
- Signature
- Keystroke
- Walking style

1.4 Iris recognition system

The purpose of 'Iris Recognition', a biometrical based technology for personal identification and verification, is to recognize a person from his/her iris prints. In fact, iris patterns are characterized by high level of stability and distinctiveness. Each individual has a unique iris.

First the iris recognition system has to localize the inner and outer boundaries of the iris (pupil and limbus) in an image of an eye. Further subroutines detect and exclude eyelids, eyelashes, and specular reflections that often occlude parts of the iris. The set of pixels containing only the iris, normalized by a rubber sheet model to compensate for pupil dilation or constriction, is then analyzed to extract a bit pattern encoding the information needed to compare two iris images.

In the case of Daugman's algorithms, a Gabor_wavelet transform is used. The result is a set of complex numbers that carry local amplitude and phase information about the iris pattern. In Daugman's algorithms, most amplitude information is discarded, and the 2048 bits representing an iris pattern consist of phase information (complex sign bits of the Gabor wavelet projections). Discarding the amplitude information ensures that the template remains largely unaffected by changes in illumination or camera gain (contrast), and contributes to the long-term usability of the biometric template.

For identification (one-to-many template matching) or verification (one-to-one template matching) a template created by imaging an iris is compared to stored template(s) in a database. If the Hamming_distance is below the decision threshold, a positive identification has effectively been made because of the statistical extreme improbability that two different persons could agree by chance ("collide") in so many bits, given the high entropy of iris templates.

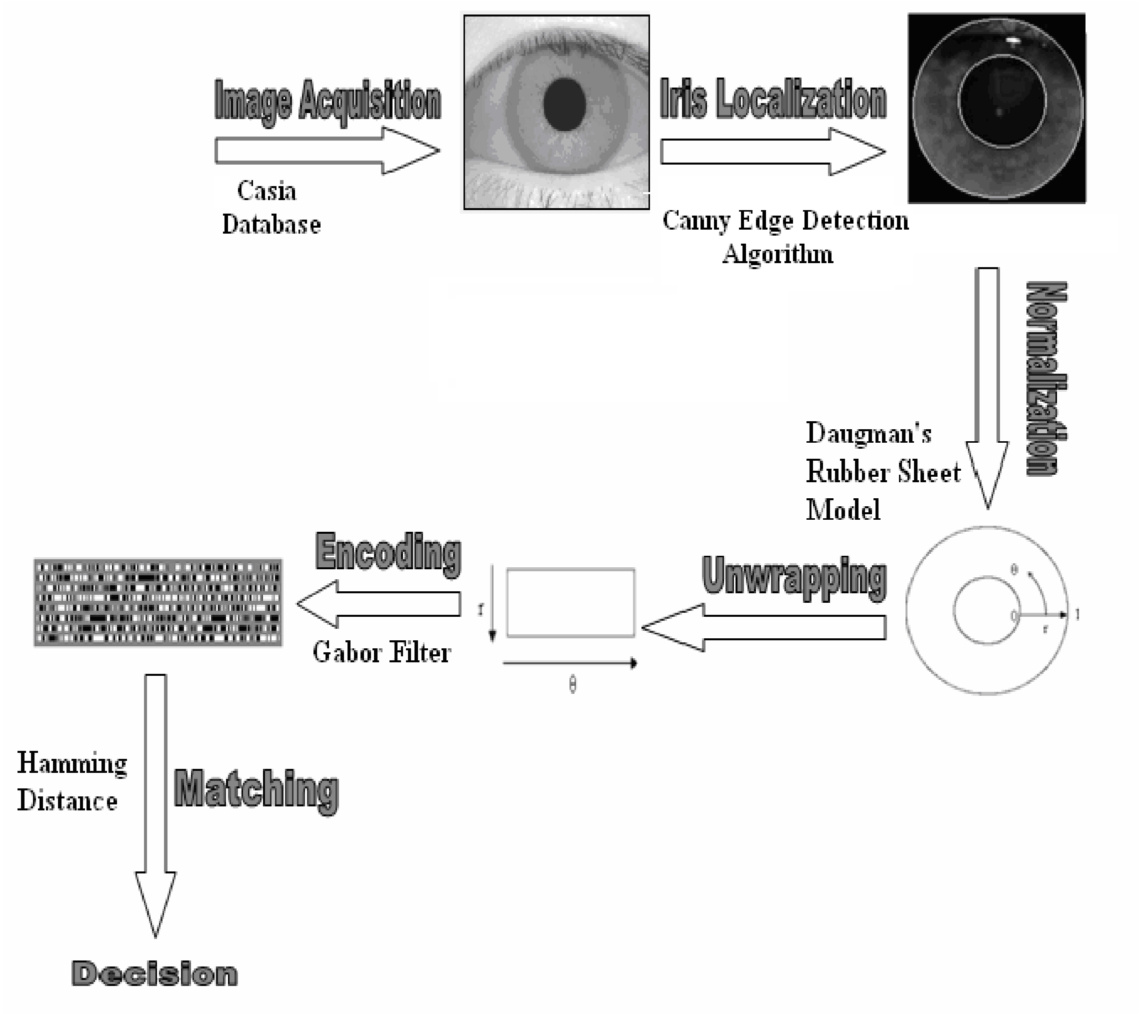

Fig. 1.1 Overview of our system [1]

CHAPTER - 2
IRIS RECOGNITION

2.1 Introduction

The iris is an externally visible, yet protected organ whose unique epigenetic pattern remains stable throughout adult life. These characteristics make it very attractive for use as a biometric for identifying individuals. The purpose of 'Iris Recognition', a biometrical based technology for personal identification and verification, is to recognize a person from his/her iris prints. In fact, iris patterns are characterized by high level of stability and distinctiveness. Each individual has a unique iris. The difference even exists between identical twins and between the left and right eye of the same person.

Iris recognition is an automated method of biometric identification that uses mathematical pattern-recognition techniques on images of the iris of an individual's eyes, whose complex random patterns are unique and can be seen from some distance.

The technology combines computer vision, pattern recognition, statistical inference, and optics. Its purpose is real-time, high confidence recognition of a person's identity by mathematical analysis of the random patterns that are visible within the iris of an eye from some distance. Because the iris is a protected internal organ whose random texture is complex, unique, and very stable throughout life, it can serve as a kind of living passport or password that one need not remember but can always present. Because the randomness of iris patterns has very high dimensionality, recognition decisions are made with confidence levels high enough to support rapid and reliable exhaustive searches through national-sized databases. The algorithms for iris recognition were developed at Cambridge University by John Daugman.

Biometrics	**Universal**	**Unique**	**Permanence**	**Collectable**	**Perfor-mance**	**Accepta-bility**	**Potential to fraud**
Face	High	Low	Medium	High	Low	Low	Low
Fingerprint	Medium	High	High	Medium	High	Medium	Low
Iris	**High**	**High**	**High**	**Medium**	**High**	**Low**	**Low**
Signature	Low	Low	Low	High	Low	High	High
Voice	Medium	Low	Low	Medium	Low	High	High
Vein	Medium	Medium	Medium	Medium	Medium	Medium	Low
DNA	High	High	High	Low	High	Low	Low

Table 2.1 Comparison of major biometrics techniques

Method	**Coded pattern**	**Misidentification rate**	**Security**	**Applications**
Iris recognition	**Iris pattern**	**1/1200000**	**High**	**High-security facilities**
Fingerprinting	Fingerprints	1/1000	Medium	Universal
Hand shape	Size, Length and thickness of hands	1/700	Low	Low securities facilities
Facial recognition	Outline, Shape and distribution of eyes and nose	1/100	Low	Low securities facilities
Signature	Shape of writing letter, writing order, pen pressure	1/100	Low	Low securities facilities
Voice printing	Voice Characteristics	1/30	Low	Telephone service

Table 2.2 Iris recognition system accuracy

The comparison of various biometrics techniques is given in tables 2.1 and 2.2. The comparison shows that IRIS recognition system is the most stable, precise and the fastest biometric authentication method.

2.2 Human iris

The iris is a thin circular diaphragm, which lies between the cornea and the lens of the human eye. The iris is perforated close to its centre by a circular aperture known as the pupil. The function of the iris is to control the amount of light entering through the pupil, and this is done by the sphincter and the dilator m333uscles, which adjust the size of the pupil. The average diameter of the iris is 12 mm, and the pupil size can vary from 10% to 80% of the iris diameter.

The iris consists of a number of layers; the lowest is the epithelium layer, which contains dense pigmentation cells. The stromal layer lies above the epithelium layer, and contains blood vessels, pigment cells and the two iris muscles. The density of stromal pigmentation determines the color of the iris.

The externally visible surface of the multi-layered iris contains two zones, which often differ in color. An outer ciliary zone and an inner pupillary zone, and these two zones are divided by the collarette – which appears as a zigzag pattern

Iris**,** in anatomy, is the pigmented muscular curtain near the front of the eye, between the cornea and the lens, that is perforated by an opening called the pupil. The iris is located in front of the lens and ciliary body and behind the cornea. It is bathed in front and behind by a fluid known as the aqueous humor. The iris consists of two sheets of smooth muscle with contrary actions: dilation (expansion) and contraction (constriction). These muscles control the size of the pupil and thus determine how much light reaches the sensory tissue of the retina. The sphincter_muscle of the iris is a circular muscle that constricts the pupil in bright light, whereas the dilator_muscle of the iris expands the opening when it contracts. The amount of pigment contained in the iris determines eye color. When there is very little pigment, the eye appears blue. With increased pigment, the shade becomes deep brown to black.

Iris patterns are formed by combined layers of pigmented epithelial cells, muscles for controlling the pupil, a stromal layer consisting of connective tissue, blood vessels and an anterior border layer. The physiological complexity of the organ results in the random patterns of the iris, which are statistically unique and suitable for biometric measurements. In addition, iris patterns are stable over time and only minor changes happen to them throughout an individual’s life. It is also an internal organ, located behind the cornea and aqueous humor and is well protected from the external environment. This characteristics such as being protected from the environment and having more reliable stability over time, compared to other popular biometrics, have well justified the ongoing research and investments on iris recognition by various researchers and industries around the world. Compared to other biometric systems, iris recognition has been in the limelight for high-security biometric applications.

2.3 Working of iris recognition system

Image processing techniques can be employed to extract the unique iris pattern from a digitized image of the eye, and encode it into a biometric template, which can be stored in a database. This biometric template contains an objective mathematical representation of the unique information stored in the iris, and allows comparisons to be made between templates. When a subject wishes to be identified by iris recognition system, their eye is first photographed, and then a template created for their iris region. This template is then compared with the other templates stored in a database until either a matching template is found and the subject is identified, or no match is found and the subject remains unidentified.

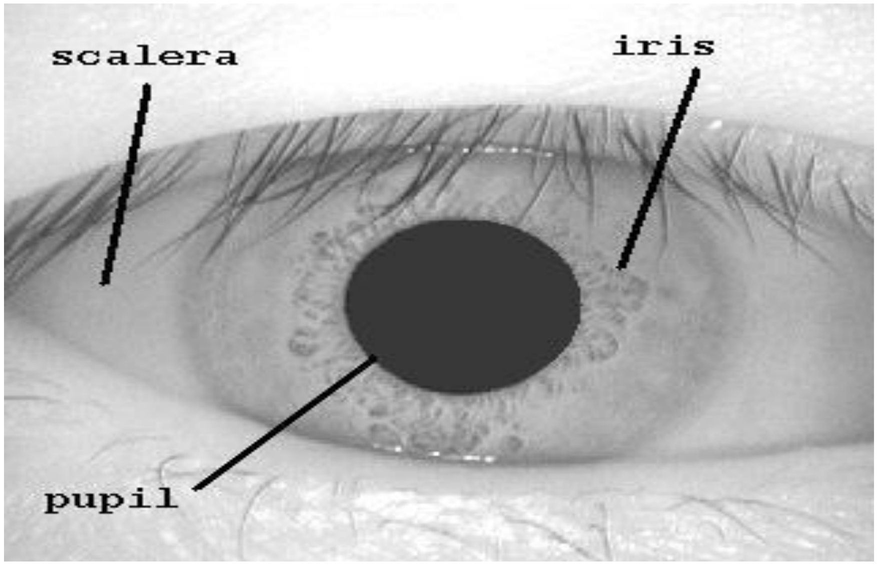

Fig 2.1 Human eye[2]

2.4 Features of iris recognition

- **Measurable Physical Features:** 250 degrees of freedom,250 non-related unique features of a person's iris

- **Unique:** Every iris is absolutely unique. No two iris are the same

- **Stable:** iris remains stable from 1styear till death.

- **Accurate:** Iris recognition is the most accurate of the commonly used biometric technologies.

- **Fast:** Iris recognition takes less than 2 seconds. 20 times more matches per minute than its closest competitor.

- **Non-Invasive:** No bright lights or lasers are used in the imaging and iris authentication process**.**

2.5 Iris recognition system strengths

- **Highly Protected:** Internal organ of the eye. Externally visible; patterns imaged from a distance

- **Measurable Features of iris pattern:** Iris patterns possess a high degree of randomness Variability: 244 degrees-of-freedom Entropy: 3.2 bits per square-millimeter

- **Uniqueness:** Set by combinatorial complexity

- **Stable:** Patterns apparently stable throughout life

- **Quick and accurate**: Image analysis and encoding time: 1 second

2.6 Iris recognition system weakness

- Some difficulty in usage as individuals does not know exactly where they should position themselves.

- And if the person to be identified is not cooperating by holding the head still and looking into the camera.

CHAPTER-3

IRIS SYSTEM IMPLEMENTATION

3.1 Image acquisition

Image acquisition is considered the most critical step in the project since all subsequent stages depend highly on the image quality. In order to accomplish this, we use a CCD camera. We set the resolution to 640x480, the type of the image to jpeg, and the mode to white and black for greater details. The camera is situated normally between half a meter to one meter from the subject. (3 to 10 inches)

The CCD-cameras job is to take the image from the optical system and convert it into electronic data. Find the iris image by a monochrome CCD (Charged couple Device) camera transfer the value of the different pixels out of the CCD chip. Read out the voltages from the CCD-chip. Thereafter the signals of each data are amplified and sent to an ADC (Analog to Digital Converter).

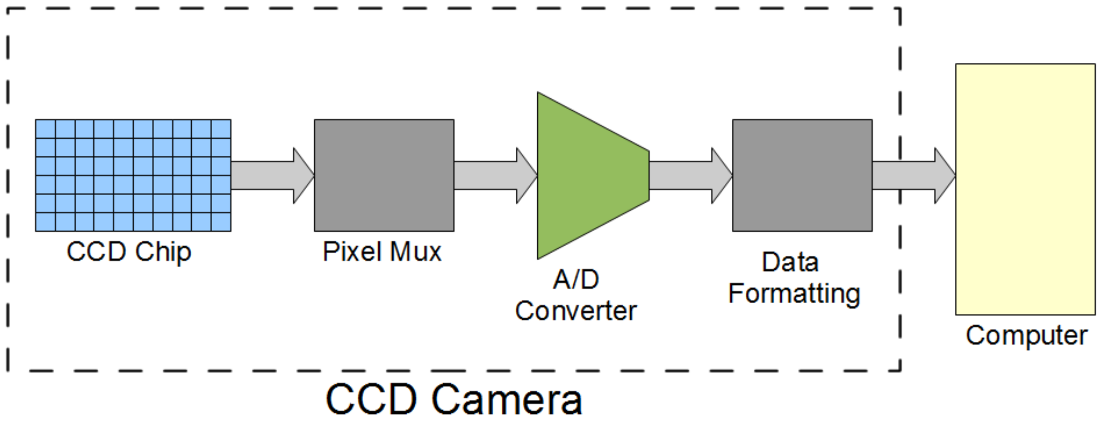

Fig. 3.1 BLOCK DIAGRAM OF IMAGE ACQUISITION [3]

3.2 Image manipulation

In the preprocessing stage, we transformed the images from RGB to gray level and from eight-bit to double precision thus facilitating the manipulation of the images in subsequent steps.

3.3 Iris localization

Before performing iris pattern matching, the boundaries of the iris should be located. In other words, we are supposed to detect the part of the image that extends from inside the limbus (the border between the sclera and the iris) to the outside of the pupil. We start by determining the outer edge by first down sampling the images by a factor of 4 then use the canny operator with the default threshold value given by Matlab, to obtain the gradient image.

Since the picture was acquired using an infrared camera the pupil is a very distinct black circle. The pupil is in fact so black relative to everything else in the picture simple edge detection should be able to find its outside edge very easily. Furthermore, the thresholding on the edge detection can be set very high as to ignore smaller less contrasting edges while still being able to retrieve the entire perimeter of the pupil. The best edge detection algorithm for outlining the pupil is canny edge detection. This algorithm uses horizontal and vertical gradients in order to deduce edges in the image. After running the canny edge detection on the image a circle is clearly present along the pupil boundary.

Canny Edge Detector finds edges by looking for the local maxima of the gradient of the input image. It calculates the gradient using the derivative of the Gaussian filter. The Canny method uses two thresholds to detect strong and weak edges. It includes the weak edges in the output only if they are connected to strong edges. As a result, the method is more robust to noise, and more likely to detect true weak edges.

3.4 Edge detection

Edges often occur at points where there is a large variation in the luminance values in an image, and consequently they often indicate the edges, or occluding boundaries, of the objects in a scene. However, large luminance changes can also correspond to surface markings on objects. Points of tangent discontinuity in the luminance signal can also signal an object boundary in the scene.

So the first problem encountered with modeling this biological process is that of defining, precisely, what an edge might be. The usual approach is to simply define edges as step discontinuities in the image signal. The method of localizing these discontinuities often then becomes one of finding local maxima in the derivative of the signal, or zero-crossings in the second derivative of the signal.

In computer vision, edge detection is traditionally implemented by convolving the signal with some form of linear filter, usually a filter that approximates a first or second derivative operator. An odd symmetric filter will approximate a first derivative, and peaks in the convolution output will correspond to edges (luminance discontinuities) in the image.

An even symmetric filter will approximate a second derivative operator. Zero-crossings in the output of convolution with an even symmetric filter will correspond to edges; maxima in the output of this operator will correspond to tangent discontinuities, often referred to as bars or lines.

3.5 Canny edge detector

Edges characterize boundaries and are therefore a problem of fundamental importance in image processing. Edges in images are areas with strong intensity contrasts a jump in intensity from one pixel to the next. Edge detecting an image significantly reduces the amount of data and filters out useless information, while preserving the important structural properties in an image.

The Canny edge detection algorithm is known to many as the optimal edge detector. A list of criteria to improve current methods of edge detection is the first and most obvious is low error rate. It is important that edges occurring in images should not be missed and that there be no responses to non-edges. The second criterion is that the edge points be well localized. In other words, the distance between the edge pixels as found by the detector and the actual edge is to be at a minimum. A third criterion is to have only one response to a single edge. This was implemented because the first 2 were not substantial enough to completely eliminate the possibility of multiple responses to an edge.

Canny's edge detection is the optimal edge detection algorithm. In this situation, an "optimal" edge detector means:

- Good detection – the algorithm should mark as many real edges in the image as possible.
- Good localization – edges marked should be as close as possible to the edge in the real image.
- Minimal response – a given edge in the image should only be marked once, and where possible, image noise should not create false edges.

To satisfy these requirements Canny used the calculus of variations – a technique which finds the function which optimizes a given functional. The optimal function in Canny's detector is described by the sum of four exponential terms, but it can be approximated by the first derivative of a Gaussian.

3.6 Canny edge detection algorithm

The algorithm runs in 5 separate steps:

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

Each step is described in the following subsections.

3.6.1 Smoothing:

It is inevitable that all images taken from a camera will contain some amount of noise. To prevent that noise is mistaken for edges, noise must be reduced. Therefore the image is first smoothed by applying a Gaussian filter.

3.6.2 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. First step is to approximate the gradient in the x- and y-direction.

The gradient magnitudes (also known as the edge strengths) can then be determined as an Euclidean distance measure by applying the law of Pythagoras. It is sometimes simplified by applying Manhattan distance measure to reduce the computational complexity.

Euclidean distance: |G| =sqrt(|Gy|^2+|Gx|^2)

Manhattan distance: |G|=|Gx|+|Gy|

Where Gx and Gy are the gradients in the x- and y-directions respectively. It is obvious that an image of the gradient magnitudes often indicate the edges quite clearly. However, the edges are typically broad and thus do not indicate exactly where the edges are.

To make it possible, the direction of the edges must be determined.

$$\theta = \arctan(|Gy|/|Gx|)$$

3.6.3 Non-maximum suppression:

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

1. Round the gradient direction θ to nearest 45◦, corresponding to the use of an 8-connected neighbourhood.
2. Compare the edge strength of the current pixel with the edge strength of the pixel in the positive and negative gradient direction. I.e. if the gradient direction is north (theta =90◦), compare with the pixels to the north and south.
3. If the edge strength of the current pixel is largest; preserve the value of the edge strength. If not, suppress (i.e. remove) the value.

3.6.4 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.

3.6.5 Edge tracking by hysteresis:

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 edge.

3.7 Hough transform

The Hough transform is a technique which can be used to isolate features of a particular shape within an image. Because it requires that the desired features be specified in some parametric form, the classical Hough transform is most commonly used for the detection of regular curves such as lines, circles, ellipses, etc. A generalized Hough transform can be employed in applications where a simple analytic description of a feature(s) is not possible. Due to the computational complexity of the generalized Hough algorithm, we restrict the main focus of this discussion to the classical Hough transform.

Despite its domain restrictions, the classical Hough transform retains many applications; as most manufactured parts contain feature boundaries which can be described by regular curves. The main advantage of the Hough transform technique is that it is tolerant of gaps in feature boundary descriptions and is relatively unaffected by image noise.

The Hough transform is computed by taking the gradient of the original image and accumulating each non-zero point from the gradient image into every point that is one radius distance away from it.

That way, edge points that lie along the outline of a circle of the given radius all contribute to the transform at the center of the circle, and so peaks in the transformed image correspond to the centers of circular features of the given size in the original image. Once a peak is detected and a circle 'found' at a particular point, nearby points (within one-half of the original radius) are excluded as possible circle centers to avoid detecting the same circular feature repeatedly.

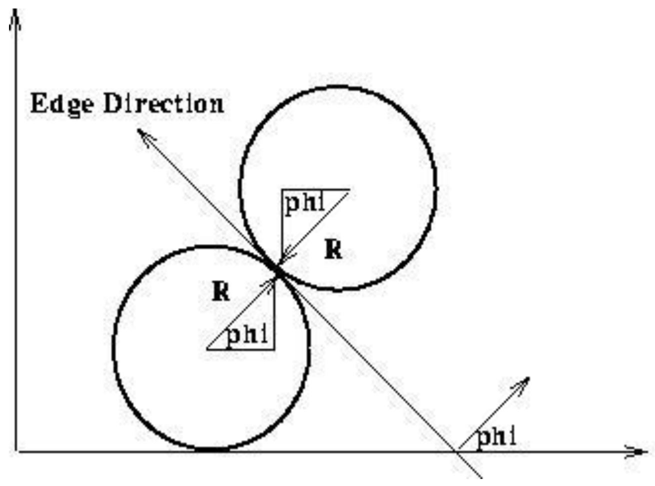

Fig 3.2: Hough circle detection with gradient information [4]

One way of reducing the computation required to perform the Hough transform is to make use of gradient information which is often available as output from an edge detector. In the case of the Hough circle detector, the edge gradient tells us in which direction a circle must lie from a given edge coordinate point.

The Hough transform can be seen as an efficient implementation of a generalized matched filter strategy. In other words, if we created a template composed of a circle of 1's (at a fixed radius) and 0's everywhere else in the image, then we could convolve it with the gradient image to yield an accumulator array-like description of all the circles of radius r in the image. It is shown formally that the basic Hough transforms (*i.e.* the algorithm with no use of gradient direction information) is equivalent to template matching.

CHAPTER-4
NORMALISATION

4.1 Introduction

Once the iris region is successfully segmented from an eye image, the next stage is to transform the iris region so that it has fixed dimensions in order to allow comparisons. The dimensional inconsistencies between eye images are mainly due to the stretching of the iris caused by pupil dilation from varying levels of illumination. Other sources of inconsistency include, varying imaging distance, rotation of the camera, head tilt, and rotation of the eye within the eye socket. The normalization process will produce iris regions, which have the same constant dimensions, so that two photographs of the same iris under different conditions will have characteristic features at the same spatial location. Another point of note is that the pupil region is not always concentric within the iris region, and is usually slightly nasal. This must be taken into account if trying to normalize the ‘doughnut’ shaped iris region to have constant radius.

4.2 Asymmetry of the eye

Although the pupil and iris circles appear to be perfectly concentric, they rarely are. In fact, the pupil and iris regions each have their own bounding circle radius and center location. This means that the unwrapped region between the pupil and iris bounding does not map perfectly to a rectangle. This is easily taken care of with a little trigonometry. There is also the matter of the pupil, which grows and contracts its area to control the amount of light entering the eye. Between any two images of the same person's eye, the pupil will likely have a different radius. When the pupil radius changes, the iris stretches with it like a rubber sheet. Luckily, this stretching is almost linear, and can be compensated back to a standard dimension before further processing.

4.3 Unwrapping

Image processing of the iris region is computationally expensive. In addition the area of interest in the image is a 'donut' shape, and grabbing pixels in this region requires repeated rectangular-to-polar conversions. To make things easier, the iris region is first unwrapped into a rectangular region using simple trigonometry. This allows the iris decoding algorithm to address pixels in simple (row, column) format.

The homogenous rubber sheet model devised by Daugman remaps each point within the iris region to a pair of polar coordinates (r, θ) where r is on the interval [0, 1] and θ is angle $[0,2\pi]$.

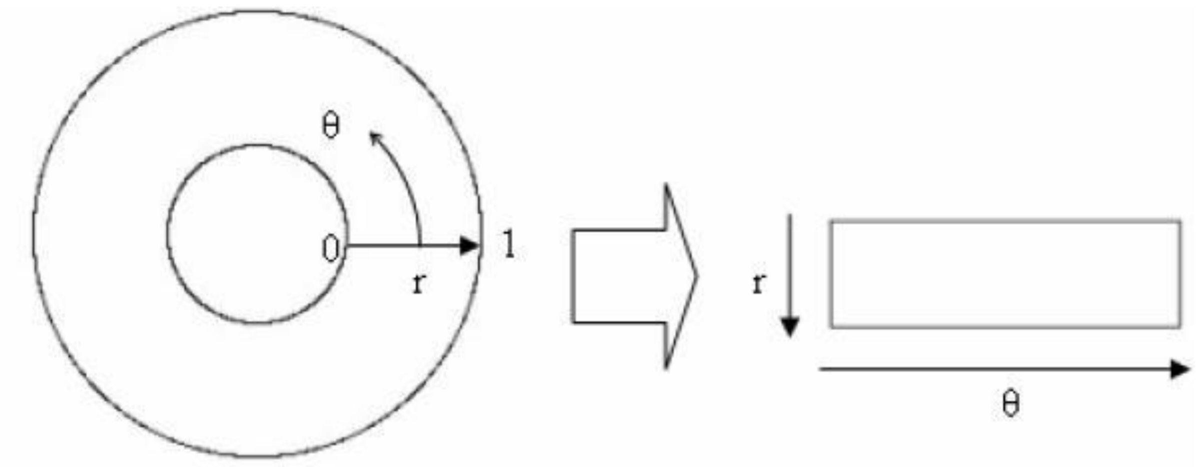

Fig 4.1 : Rubber sheet model [5]

The remapping of the iris region from *(x,y)* Cartesian coordinates to the normalized non concentric polar representation is modeled as

$$I(x(r,\theta), y(r,\theta)) \rightarrow I(r,\theta)$$

with

$$x(r,\theta)=(1-r)x_{p}(\theta) + rx_{i}(\theta)$$

$$y(r,\theta)=(1-r)y_{p}(\theta) + ry_{i}(\theta)$$

Where *I(x, y)* is the iris region image, *(x, y)* are the original Cartesian coordinates, (r, θ) are the corresponding normalized polar coordinates, and x_p, y_p, and x_i,

y_i are the coordinates of the pupil and iris boundaries along the θ direction. The rubber sheet model takes into account pupil dilation and size inconsistencies in order to produce a normalized representation with constant dimensions. In this way the iris region is modeled as a flexible rubber sheet anchored at the iris boundary with the pupil centre as the reference point.

Even though the homogenous rubber sheet model accounts for pupil dilation, imaging distance and non concentric pupil displacement, it does not compensate for rotational inconsistencies. In the Daugman system, rotation is accounted for during matching by shifting the iris templates in the θ direction until two iris templates are aligned.

However, the normalization process was not able to perfectly reconstruct the same pattern from images with varying amounts of pupil dilation, since deformation of the iris results in small changes of its surface patterns. The rectangular representation is constructed from data points in each iris region. Rotational inconsistencies have not been accounted for by the normalization process, and the two normalized patterns are slightly misaligned in the horizontal (angular) direction.

CHAPTER-5

FEATURE EXTRACTION

5.1 Introduction

The iris has an interesting structure and presents plentiful Texture information. So, it is attractive to search representation methods which can capture local crucial information in an Iris. The distinctive spatial characteristics of the human iris are manifest at a variety of scales. For example, distinguishing structures range from the overall shape of the iris to the distribution of tiny crypts and detailed texture. To capture this range of spatial detail, it is advantageous to make use of a multi-scale representation. Some works have used multi-resolution techniques for iris feature extraction and have proven a high recognition accuracy.

At the same time, however, it has been observed that each multi-resolution technique has its specification and situation in which it is suitable; for example, a Gabor filter bank has been shown to be most known multi-resolution method used for iris feature extraction and Daugman in his proposed iris recognition system demonstrated the highest accuracy by using Gabor filters. However, from the point of view of texture analysis one can observe that Gabor filter based methods analysis pretty well the texture orientations. In this paper, we have investigated the use of wavelet maxima components as a multi-resolution technique alternative for iris feature extraction.

“One of the most interesting aspects of the world is that it can be considered to be made up of patterns. A pattern is essentially an arrangement. It is characterized by the order of the elements of which it is made, rather than by the intrinsic nature of these elements” (Nobert Wiener). This definition summarizes our purpose in this part. In fact, this step is responsible of extracting the patterns of the iris taking into account the correlation between adjacent pixels. For this we use wavelets transform, and more specifically the “Gabor Filtering”.

5.2 Gabor filtering

To understand the concept of Gabor filtering, we must first start with Gabor wavelets. Gabor wavelets are formed from two components, a complex sinusoidal carrier and a Gaussian envelope.

$$g\ (x, y) = s\ (x, y)\ wr\ (x,y)$$

The complex carrier takes the form:

$$s\ (x, y) = e^{\ j(2\pi(u_0x+v_0y)+P)}$$

We can visualize the real and imaginary parts of this function separately as shown in this figure. The real part of the function is given by:

$$Re\ (s\ (x, y)) = \cos\ (2\pi(u0x + v0y) + P)$$

and the imaginary:

$$Im\ (s\ (x, y)) = \sin\ (2\pi(u0x + v0y) + P)$$

The parameters u0 and v0 represent the frequency of the horizontal and vertical sinusoids respectively. P represents an arbitrary phase shift. The second component of a Gabor wavelet is its envelope. The resulting wavelet is the product of the sinusoidal carrier and this envelope. The envelope has a Gaussian profile and is described by the following equation:

$$Ke^{\ -\pi(a2(x-x_0)_r2+b2(y-y_0)_r2)}$$

Where:

$$(x - x0)_r = (x - x0)\cos(\theta) + (y - y0)\sin(\theta)$$
$$(y - y0)_r = -(x - x0)\sin(\theta) + (y - y0)\cos(\theta)$$

The parameters used above are:

K – a scaling constant

(a, b) - Envelope axis scaling constants,

θ - envelope rotation constant,

(x0, y0) - Gaussian envelope peak

5.3 Generating iris code

After performing unwrapping algorithm it maps the image to Cartesian coordinates. What we want to do is somehow extract a set of unique features from this iris and then store them. That way if we are presented with an unknown iris, we can compare the stored features to the features in the unknown iris to see if they are the same. We'll call this set of features an "Iris Code."

Any given iris has a unique texture that is generated through a random process before birth. Filters based on Gabor wavelets turn out to be very good at detecting patterns in images. We'll use a fixed frequency 1D Gabor filter to look for patterns in our unrolled image.

First, we'll take a one pixel wide column from our unrolled image and convolve it with a 1D Gabor wavelet. Because the Gabor filter is complex, the result will have real and imaginary parts which are treated separately. We only want to store a small number of bits for each iris code, so the real and imaginary parts are each quantized. If a given value in the result vector is greater than zero, a one is stored; otherwise zero is stored. Once all the columns of the image have been filtered and quantized, we can form a new black and white image by putting all of the columns side by side.

5.4 Matching using hamming distance

For matching, the Hamming distance was chosen as a metric for recognition, since bit-wise comparisons were necessary. The Hamming distance will be calculated using only the bits generated from the true iris region. Ratio of Hamming distance to total bits is given as

$$\textbf{HD/Total no. of bits=(1/N)}\sum_{j=1}^{N} \boldsymbol{Xj(XOR)\ Yj}$$

where *Xj* and *Yj* are the two bit-wise templates to compare.

Although, in theory, two iris templates generated from the same iris will have a Hamming distance of 0.0, in practice this will not occur. Normalization is not perfect, and also there will be some noise that goes undetected, so some variation will be present when comparing two intra-class iris templates.

In order to account for rotational inconsistencies, when the Hamming distance of two templates is calculated, one template is shifted left and right bit-wise and a number of Hamming distance values are calculated from successive shifts. This bit-wise shifting in the horizontal direction corresponds to rotation of the original iris region by an angle given by the angular resolution used. If an angular resolution of 180 is used, each shift will correspond to a rotation of 2 degrees in the iris region. This method is suggested by Daugman, and corrects for misalignments in the normalized iris pattern caused by rotational differences during imaging. From the calculated Hamming distance values, only the lowest is taken, since this corresponds to the best match between two templates. The number of bits moved during each shift is given by two times the number of filters used, since each filter will generate two bits of information from one pixel of the normalized region. The actual number of shifts required to normalize rotational inconsistencies will be determined by the maximum angle difference between two images of the same eye, and one shift is defined as one shift to the left, followed by one shift to the right.

An illustration of Hamming distance calculation:

Template 1: 10110110101011

Template 2: 01100100100101

Total Bits: 12

HD=Template1 XOR Template2

HD=7

HD/Total bits=7/12=0.583

Ideally Hamming distance between templates of same iris should be zero.

CHAPTER-6

EXPERIMENTAL RESULTS

Following results were obtained after the stepwise implementation of MATLAB code:

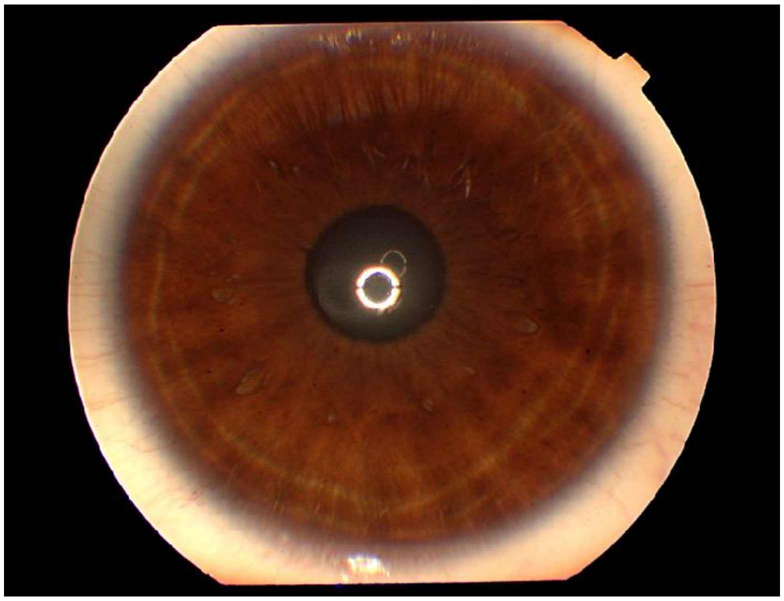

Fig. 6.1 Original iris image

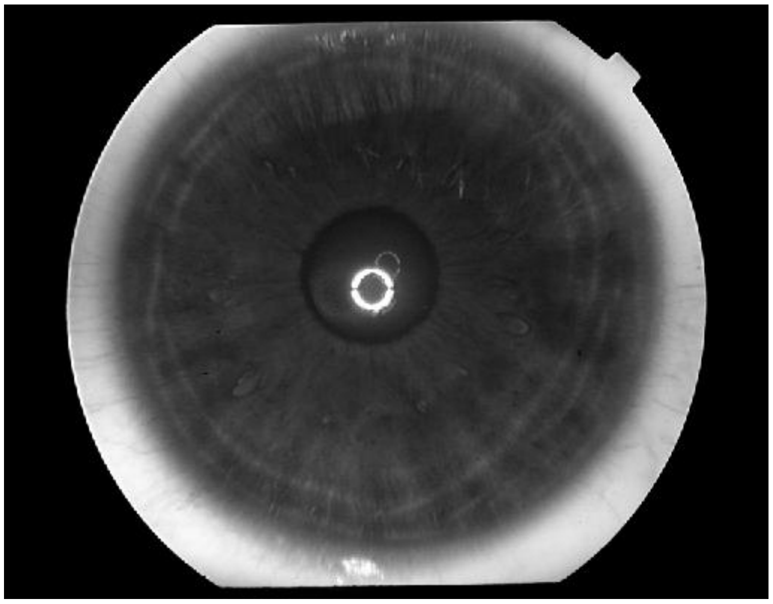

Fig. 6.2 Gray scaled image

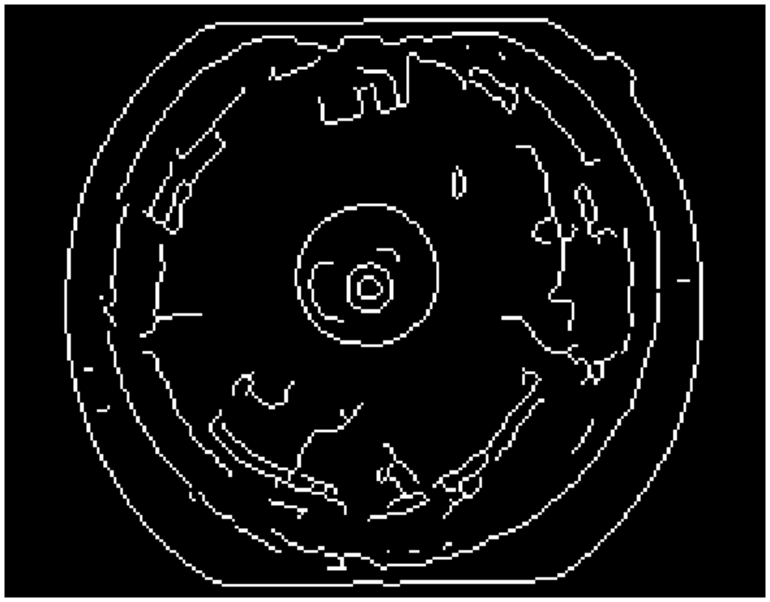

Fig. 6.3 Edge map obtained after applying Canny edge detection

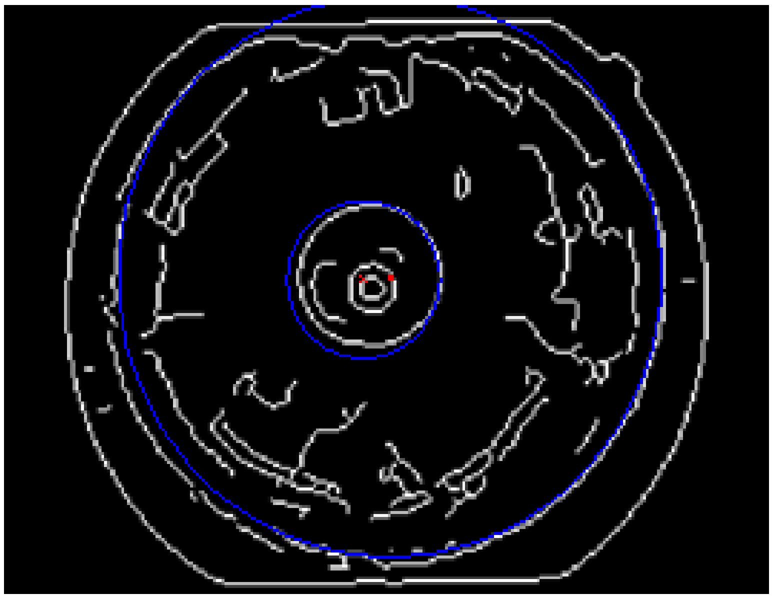

Fig 6.4 Boundary detection using Hough transform

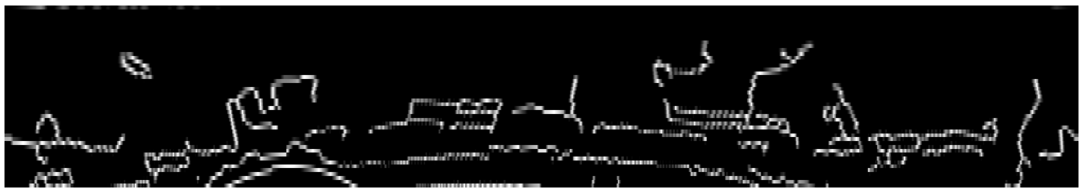

Fig 6.5 Normalized iris image

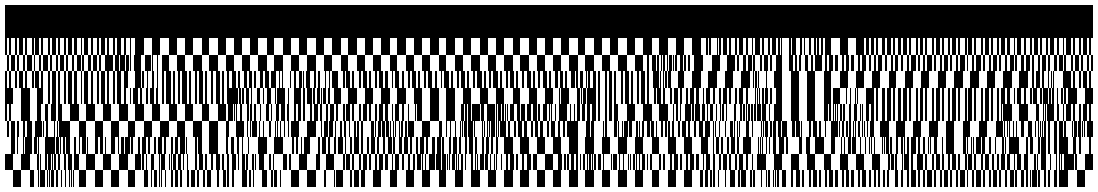

Fig 6.6 Iris template

Source Code

```
clear all
clc


%STEP 1
%Image Input

I_input=imread('D:\Users\comp\Desktop\OriginalIris.tiff');
subplot(3,2,1),imshow(I_input);
TITLE('Original Iris Image');
eyeimage=rgb2gray(I_input);
subplot(3,2,2),imshow(eyeimage);
TITLE('Gray Scaled Iris Image');



%STEP 2
%Image Scaling
scaling=0.3;
eyeimage = imresize(eyeimage, scaling);
subplot(3,2,3),imshow(eyeimage);
TITLE('IMAGE(AFTER SCALING)');


%STEP 3
%Canny Edge Detection
BW2 = EDGE(eyeimage,'canny');



%STEP 4

%Hough Transform Implementation for boundary detection(IRIS REGION)
[y,x]=find(BW2);
[sy,sx]=size(BW2);

% Find all the require information for the transformation. The 'totalpix' is the numbers of '1' in
the image.

totalpix = length(x);

%3. Preallocate memory for the Hough Matrix.
```

```
HM = zeros(sy*sx,1);
% Enter the radius for boundary detection. Radius was computed using imdistline

R = 82;
R2=R^2;


% Preparing all the matrices for the computation

b = 1:sy;
a = zeros(sy,totalpix);
y = repmat(y',[sy,1]);
x = repmat(x',[sy,1]);
b = repmat(b',[1,totalpix]);


% The equation for  circle

a = (round(x - sqrt(R2 - (y - b).^2)));


% Removing all the invalid value in matrices a and b

b = b(imag(a)==0 & a>0);

a = a(imag(a)==0 & a>0);

ind = sub2ind([sy,sx],b,a);


% Reconstruct the Hough Matrix

val = ones(length(ind),1);
data=accumarray(ind,val);
HM(1:length(data)) = data;
HM2 = reshape(HM,[sy,sx]);



% Showing the Hough Matrix
cla
imshow(HM2,[]);
```

```
% Finding the location of the circle with radius of R

[maxval, maxind] = max(max(HM2));

[B,A] = find(HM2==maxval);


plot(mean(A),mean(B),'xr')

t= 0:pi/50:2*pi;
X=A+R*cos(t);
Y=B+R*sin(t);


%Hough Transform Implementation for boundary detection(PUPIL REGION)
[yp,xp]=find(BW2);
[syp,sxp]=size(BW2);

% Find all the require information for the transformation. The 'totalpix' is the number of '1's' in
the image.

totalpix = length(xp);

%3. Preallocate memory for the Hough Matrix.

HMp = zeros(syp*sxp,1);
Rp = 23;
Rp2=Rp^2;



% Performing Hough Transform



bp = 1:syp;

ap = zeros(syp,totalpix);

yp = repmat(yp',[syp,1]);
xp = repmat(xp',[syp,1]);

bp = repmat(bp',[1,totalpix]);
```

```
% The equation for  circle

ap = (round(xp - sqrt(Rp2 - (yp - bp).^2)));


% Removing all the invalid values in matrices a and b

bp = bp(imag(ap)==0 & ap>0);

ap = ap(imag(ap)==0 & ap>0);

ind = sub2ind([syp,sxp],bp,ap);


% Reconstruct the Hough Matrix

valp = ones(length(ind),1);
datap=accumarray(ind,valp);
HMp(1:length(datap)) = datap;
HM2p = reshape(HMp,[syp,sxp]);



% Showing the Hough Matrix
cla
imshow(HM2p,[]);


% Finding the location of the circle with radius of R

[maxvalp, maxindp] = max(max(HM2p));

[Bp,Ap] = find(HM2p==maxvalp);

imshow(BW2); hold on;


t= 0:0.1:2*pi;
Xp=Ap+Rp*cos(t);
Yp=Bp+Rp*sin(t);
plot(X,Y);
hold on;
plot(Xp,Yp);hold on;
plot(mean(A),mean(B),'xr')
hold on;
```

```
plot(mean(Ap),mean(Bp),'xr')

%STEP 5
%Normalisation
[polar,xo,yo] = normaliseiris(BW2, A, B,R, Ap, Bp, Rp,eyeimage);
plot(polar)
polar_array=polar;

nscales=2;
minWaveLength=1;
mult=3;
sigmaOnf=2;
im=polar_array;

%STEP 6
%Gabor Filetring
[EO, filtersum] = gaborconvolve(im, nscales, minWaveLength, mult,sigmaOnf);
Ea=EO;
fs=filtersum;

%STEP 7
%Creation of Templates
[template1] = encoder(polar_array, nscales, Ea);


%STEP 8
%Getting The Hamming Distance for matching
[hd] = gethammingdistance1(template1,template2,nscales);
```

Functions used:

Normalisation Function

```
% normaliseiris - performs normalisation of the iris region by
% unwrapping the circular region into a rectangular block of
% constant dimensions.
%
% Usage:
% [polar_array] = normaliseiris(image, x_iris, y_iris, r_iris,...
% x_pupil, y_pupil, r_pupil,eyeimage_filename)
%
% Arguments:
% image           - the input eye image to extract iris data from
% x_iris          - the x coordinate of the circle defining the iris
%                   boundary
% y_iris          - the y coordinate of the circle defining the iris
%                   boundary
% r_iris             - the radius of the circle defining the iris
%                    boundary
% x_pupil              - the x coordinate of the circle defining the pupil
%                    boundary
% y_pupil              - the y coordinate of the circle defining the pupil
%                    boundary
% r_pupil             - the radius of the circle defining the pupil
%                    boundary
% eyeimage_filename     - original filename of the input eye image
% radpixels            - radial resolution, defines vertical dimension of
%                    normalised representation
% angulardiv            - angular resolution, defines horizontal dimension
%                    of normalised representation
%
% Output:
% polar_array



function [polar_array,xo,yo] = normaliseiris(BW2, X, Y,R, Xp, Yp, Rp,eyeimage)
radpixels=100;
angulardiv=2400;

global  DIAGPATH
radiuspixels = radpixels + 2;
angledivisions = angulardiv;

r = 0:(radiuspixels-1);

theta = 0:2*pi/angledivisions:2*pi;

x_iris = double(X);
y_iris = double(Y);
r_iris = double(R);

x_pupil = double(Xp);
y_pupil = double(Yp);
r_pupil = double(Rp);

% calculate displacement of pupil center from the iris center
ox = x_iris - x_pupil;
oy = y_iris - y_pupil;
```

```
r = double(r);
theta = double(theta);

a = ones(1,angledivisions+1)*(ox^2 + oy^2);



b = cos(pi - theta);

% calculate radius around the iris as a function of the angle
r = (sqrt(a).*b) + ( sqrt( a.*(b.^2) - (a - (r_iris^2))));

r = r - r_pupil;

rmat = ones(1,radiuspixels)'*r;

rmat = rmat.* (ones(angledivisions+1,1)*[0:1/(radiuspixels-1):1])';
rmat = rmat + r_pupil;


% exclude values at the boundary of the pupil iris border, and the iris scelra border
% as these may not correspond to areas in the iris region and will introduce noise.
%
% ie don't take the outside rings as iris data.
rmat  = rmat(2:(radiuspixels-1), :);

% calculate cartesian location of each data point around the circular iris
% region
xcosmat = ones(radiuspixels-2,1)*cos(theta);
xsinmat = ones(radiuspixels-2,1)*sin(theta);

xo = rmat.*xcosmat;

yo = rmat.*xsinmat;


xo = x_pupil+xo;

yo = y_pupil-yo;


% extract intensity values into the normalised polar representation through
% interpolation
[x,y] = meshgrid(1:size(BW2,2),1:size(BW2,1));
polar_array = interp2(x,y,BW2,xo,yo);
```

Gabor Filter Function

```
% gaborconvolve - function for convolving each row of an image with 1D log-Gabor filters
%
% Usage:
% [template, mask] = createiristemplate(eyeimage_filename)
%
% Arguments:
%   im              - the image to convolve
%   nscale          - number of filters to use
%   minWaveLength   - wavelength of the basis filter
%   mult            - multiplicative factor between each filter
%   sigmaOnf        - Ratio of the standard deviation of the Gaussian describing
%                     the log Gabor filter's transfer function in the frequency
%                     domain to the filter center frequency.
%
% Output:
%   E0              - a 1D cell array of complex valued convolution results



function [EO, filtersum] = gaborconvolve(im, nscale, minWaveLength, mult, ...
   sigmaOnf)

[rows cols] = size(im);


EO = cell(1, nscale);          % Pre-allocate cell array

ndata = cols;
if mod(ndata,2) == 1             % If there is an odd No of data points
   ndata = ndata-1;              % throw away the last one.
end

logGabor  = zeros(1,ndata);
result = zeros(rows,ndata);

radius =  [0:fix(ndata/2)]/fix(ndata/2)/2;  % Frequency values 0 - 0.5
radius(1) = 1;

wavelength = minWaveLength;       % Initialize filter wavelength.


for s = 1:nscale,                 % For each scale.
```

```
   % Construct the filter - first calculate the radial filter component.
   fo = 1.0/wavelength;                  % Centre frequency of filter.
   rfo = fo/0.5;                         % Normalised radius from centre of frequency plane
   % corresponding to fo.
   logGabor(1:ndata/2+1) = exp((-(log(radius/fo)).^2) / (2 * log(sigmaOnf)^2));
   logGabor(1) = 0;

   filter = logGabor;

   filtersum = filter;

   % for each row of the input image, do the convolution, back transform
   for r = 1:rows  % For each row

      signal = im(r,1:ndata);


      imagefft = fft( signal );


      result(r,:) = ifft(imagefft .* filter);

   end

   % save the ouput for each scale
   EO{s} = result;

   wavelength = wavelength * mult;       % Finally calculate Wavelength of next filter
end                                      % ... and process the next scale

filtersum = fftshift(filtersum);
```

<u>Encode Function(For creation of templates)</u>

```
% encode - generates a biometric template from the normalised iris region,
% also generates corresponding noise mask
%
% Usage:
% [template] = encode(polar_array,noise_array, nscale,Ea)

% Arguments:
```

```
% polar_array       - normalised iris region
% sigmaOnf          - bandwidth parameter
%
% Output:
% template          - the binary iris biometric template
Ea              - a 1D cell array of complex valued convolution results

function [template, mask] = encoder(polar_array, nscales,Ea)

% convolve normalised region with Gabor filters


length = size(polar_array,2)*2*nscales;

template = zeros(size(polar_array,1), length);

length2 = size(polar_array,2);
h = 1:size(polar_array,1);

%create the iris template



for k=1:nscales

   E1 = Ea{k};

   %Phase quantisation
   H1 = real(E1) > 0;
   H2 = imag(E1) > 0;

   % if amplitude is close to zero then
   % phase data is not useful, so mark off

   for i=0:(length2-2)

      ja = double(2*nscales*(i));
      %construct the biometric template
      template(h,ja+(2*k)-1) = H1(h, i+1);
      template(h,ja+(2*k)) = H2(h,i+1);
  end
  end
```

<u>Shift Bits Function</u>

```
% shiftbits - function to shift the bit-wise iris patterns in order to provide the best match
% each shift is by two bit values and left to right, since one pixel value in the
% normalised iris pattern gives two bit values in the template
% also takes into account the number of scales used
%
% Usage:
% [template, mask] = createiristemplate(eyeimage_filename)
%
% Arguments:
%   template        - the template to shift
%   noshifts        - number of shifts to perform to the right, a negative
%                   value results in shifting to the left
%   nscales         - number of filters used for encoding, needed to
%                   determine how many bits to move in a shift
%
% Output:
%   templatenew     - the shifted template



function templatenew = shiftbits(template, noshifts,nscales)

templatenew = zeros(size(template));

width = size(template,2);
s = round(2*nscales*abs(noshifts));
p = round(width-s);

if noshifts == 0
    templatenew = template;

    % if noshifts is negatite then shift towards the left
elseif noshifts < 0

    x=1:p;

    templatenew(:,x) = template(:,s+x);

    x=(p + 1):width;

    templatenew(:,x) = template(:,x-p);

else

    x=(s+1):width;

    templatenew(:,x) = template(:,x-s);
    x=1:s;

    templatenew(:,x) = template(:,p+x);

end
```

Hamming Distance Function

```
% gethammingdistance - returns the Hamming Distance between two iris templates
% incorporates noise masks, so noise bits are not used for
% calculating the HD
%
% Usage:
% [template] = createiristemplate(template1,template2,nscales)
%
% Arguments:
%   template1       - first template
%
%   template2       - second template
%   %   scales      - the number of filters used to encode the templates,
%                     needed for shifting.
%
% Output:
%   hd              - the Hamming distance as a ratio
%



function hd = gethammingdistance1(template1,template2,nscales)

template1 = logical(template1);


template2 = logical(template2);


hd = NaN;

% shift template left and right, use the lowest Hamming distance
for shifts=-8:8
```

```
    template1s = shiftbits(template1, shifts,nscales);


    totalbits = (size(template1s,1)*size(template1s,2)) ;

    C = xor(template1s,template2);


    bitsdiff = sum(sum(C==1));

    if totalbits == 0

        hd = NaN;

    else

        hd1 = bitsdiff / totalbits;


        if  hd1 < hd || isnan(hd)

            hd = hd1;

        end


    end

end
```

Conclusion

We have presented an iris recognition system, which was tested using two databases of grayscale eye images in order to verify the claimed performance of iris recognition technology. Firstly, an automatic segmentation algorithm was presented, which would localize the iris region from an eye image. Automatic segmentation was achieved through the use of the circular Hough transform for localizing the iris and pupil regions. Next, the segmented iris region was normalized to eliminate dimensional inconsistencies between iris regions. This was achieved by implementing a version of Daugman's rubber sheet model, where the iris is modeled as a flexible rubber sheet, which is unwrapped into a rectangular block with constant polar dimensions. Finally, features of the iris were encoded by convolving the normalized iris region with 1D Log-Gabor filters and phase quantizing the output in order to produce a bit-wise biometric template. The Hamming distance was chosen as a matching metric, which gave a measure of how many bits disagreed between two templates. A failure of statistical independence between two templates would result in a match, that is, the two templates were deemed to have been generated from the same iris if the Hamming distance produced was lower than a set Hamming distance.

References

[1] https://www.aub.edu.lb/fea/ece/research/Documents/Report/fyp_0506/48_Report.pdf

[2]Shankargouda M. Patil, Sarojini B. K / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 Vol. 3, Issue 1, January -February 2013, pp.1621-1626

[3] http://www.rocketroberts.com/astro/ccd_fundamentals.htm

- Human Iris Recognition for Biometric Identification
  Padma Polash Paul, Md. Maruf Monwar Ahsanullah University of Science and Technology, Dhaka , Bangladesh.

- An effective and fast iris recognition system based on a combined multiscale feature extraction technique Institute for Electronics, Communications and Information Technology (ECIT), School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Northern Ireland, BT7 1NN UK

- The Biometrics Technology“ Third edition by Guodong Guo.

- The IRIS Recognition System “ by Kshvik Douglas.

- www.ijera.com

- www.alibris.com

- www.icdri.org