LOCALIZATION AND SEGMENTATION OF NON-IDEAL IRIS IMAGES

Prepared By

AYUSH NARAYAN SHUKLA (15101104019)
RITHIK BANERJEE (15101104063)
ANIKET PATRA (15101104016)

Under the Guidance of

Prof. CHINMOY GHOSH

Assistant Professor

Department of Computer Science & Engineering
Jalpaiguri Govt. Engineering College (Autonomous)

PROJECT REPORT

Submitted in partial fulfillment of the requirement for the degree of
Bachelor of Technology

Department of Computer Science & Engineering
Jalpaiguri Govt. Engineering College (Autonomous)

Affiliated to

Maulana Abul Kalam Azad University of Technology, West Bengal

May 2019

Jalpaiguri Govt. Engineering College (Autonomous)

CERTIFICATION

Jalpaiguri Govt. Engineering College
Faculty of Computer Science & Engineering

This is to certify that the work in preparing the project entitled "*Localization & Segmentation of Non-Ideal Iris Images*", has been carried out by Ayush Narayan Shukla,Rithik Banerjee and Aniket Patra under my guidance during the session 2018-19 and accepted in partial fulfillment of requirement for the degree of Bachelor of Technology in Computer Science & Engineering.

D Mondal
....
Prof. Dhiman Mondal
Head of Department
Computer Science & Engineering
Jalpaiguri Govt. Engineering College
HEAD
Dept. of Comp. Sc. & Engg.
Jalpaiguri Govt. Engg. College

....
Prof. Chinmoy Ghosh
Mentor
Computer Science & Engineering
Jalpaiguri Govt. Engineering College
Assistant Professor
Dept. of Computer Science & Engg.
Jalpaiguri Govt. Engineering College
Jalpaiguri-735102

Acknowledgement

We sincerely thank our project mentor **Prof. Chinmoy Ghosh** for guiding and giving us his valuable time and advice. He always enriched us with his knowledge and gave us the necessary input to carry out this work. We are grateful to him for his extra efforts and for being patient with us. We are grateful to **Prof. Dhiman Mondal**, Head of the Department of Computer Science & Engineering for permitting us to make use of the facilities available in the department to carry out the project successfully. Last but not the least we express our sincere thanks to all of our other staffs and our parents who have patiently extended all sorts of help for accomplishing this undertaking.

Ayush Narayan Shukla

Rithik Banerjee

Aniket Patra

Abstract

Biometric deals with Identification of individuals based on their biological or behavioural characteristics. Iris recognition system is regarded as the most reliable and accurate biometric identification system available. Traditional personal authentication methods have many instinctive defects. Biometrics is an effective technology to overcome these defects. Among the available biometric approaches, iris recognition is one of the most accurate techniques. In this project work, two approaches for Iris recognition system have been proposed. Simple morphological operations and two dimensional median filtering techniques are used to detect the pupil using OpenCV python. The noises such as eyelashes, reflections are removed through the linear thresholding. Finally after the circle is detected around the iris it is cropped using a 8 unit mask so that it could be used for feature extraction process.

Iris recognition is an automated method of biometric identification that uses mathematical pattern-recognition techniques on video images of the irises of an individual's eyes, whose complex random patterns are unique and can be seen from some distance. It has a very high accuracy (infinitesimally small false match rates) and is non-destructive as compared to retina-scanning. Hence, it finds a great usage in security systems today and in future.

Index

List of Figures

Lists of Tables

Chapter 1

Introduction

Security of Computer systems plays a crucial role nowadays. A term "Biometric" indicates to the identification and authentication of an individual identity based on unique features or characteristics in the individuals [1]. Biometric systems consist of physiological characteristics and behavioral characteristics. Physiological characteristics are a group of biometrics which includes the physiological and biological features as dominated by a biometric system. It specifically contains DNA, Hand, Face, Earlobe, and iris. Behavioral characteristics are a group of biometrics which is concerned by the non-physiological or non-biological features as dominated by a biometric system. It consists of four categories: Signature, Voice, Gait and Keystroke recognition [2].

To meet security requirements [3] of the current networked society, personal identifiers are becoming more important. Conventional methods that are used for personal identifiers can be either token-based methods or knowledge –based methods. Token-based methods use keys or ID cards for authentication, and knowledge-based methods use preset code or password by the user. However, conventional methods become not reliable if, for example the token is lost or the password is forgotten, therefore; the needs of new & developed reliable methods for personal identification become more and more important research area [4]. The iris is one of the most reliable methods that used to identify individuals because it is fixed and does not change throughout life. Moreover, it is impossible to find two persons have the same iris features even for the twins [5]. The iris is a circular anatomical structure which is located between the cornea and the lens of the eye as shown in Figure 1-1. The iris's task is to control the light that is entering through the pupil; this is done by the sphincter and the dilator muscles, which regulated the size of the pupil. The average of iris diameter is between 11.6 mm and 12.0 mm, and the pupil size is between 10% and 80% of the iris diameter. Human iris consists of two layers; the epithelium layer which consists of intensive pigmentation cells and the stroma layer which contains blood vessels. It is responsible for reducing the size of the pupil. This layer lies on top of the epithelium layer [3].

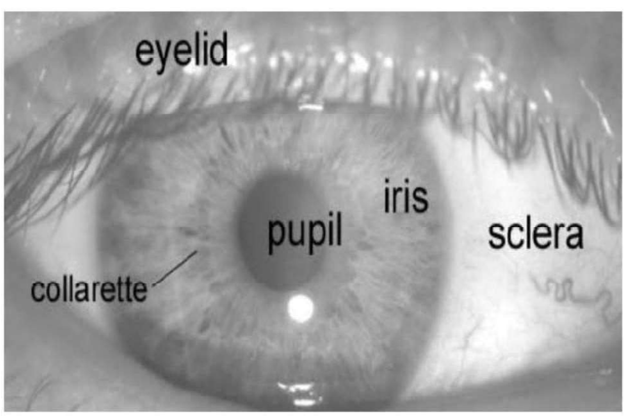


Figure 1.1: A front view of the Human Iris [3].

Iris recognition is a method that is used to identify people based on unique features within the iris. Moreover, the iris usually has a grey, blue, brown or green color. Iris recognition is considered a form of biometric verification [6]. The first concept in iris recognition was proposed in 1987 by Flom and Safir. They proposed highly controlled and non-functional conditions to change the illumination so that size of the pupil in all images remains same for suitable Iris segmentation. They outlined the basic subsystems of Iris recognition system, namely image acquisition phase, preprocessing, Iris segmentation phase, Iris analysis, feature extraction phase, classification phase along with appropriate image processing, and pattern recognition techniques. This theoretical work on Iris recognition system has considered as a basis for all practical approaches of Iris recognition system [7]. A typical iris recognition system includes six main stages[8]. The first stage is image acquisition that is done with capturing the series of images of the iris using cameras, in order to ensure acquiring the best images to increase flexibility and provide strong recognition. The second stage, image preprocessing that means the control of size, color and light of the image in order to be ready for segmentation stage.

The third stage, segmentation which includes iris and pupil boundary detection, and additionally detect eyelids and eyelashes. The fourth stage, normalization means converting the iris region into a form like a rectangle. The fifth stage, feature extraction, extracts features from the normalized iris image and encodes these features to a design that is suitable for recognition. The last stage in iris recognition system, classification that means comparison the features created by imaging the iris with stored features in the database.

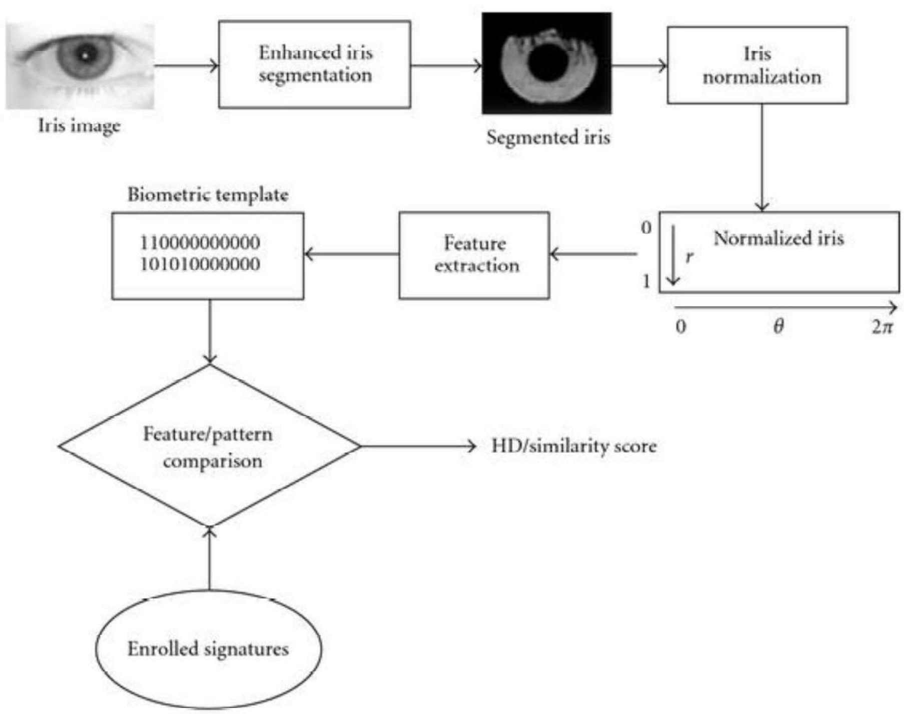


Figure 1.2: The Iris Recognition stages.

1.1 Problem Statement

Localization and Segmentation of Non-ideal iris images.

1.2 Problem Description

The main problem studied in this thesis is iris recognition. Thus, our main challenge in this paper is to propose a methodology, design, and implementation of iris recognition system in order to achieve high accuracy in recognizing human iris by using two separate approaches. There are many studies focusing on iris recognition using images that are taken from eyes of individuals. However, many of the negative results occur because of errors in the method of capturing images, image dimensions, quality, shadows, image background, eyes color, in addition to contact lenses in the eyes, etc.

A non Ideal image occurs due to off-angles, noise, blurring, and occlusion by eye lashes, eyelids, glasses and hair.

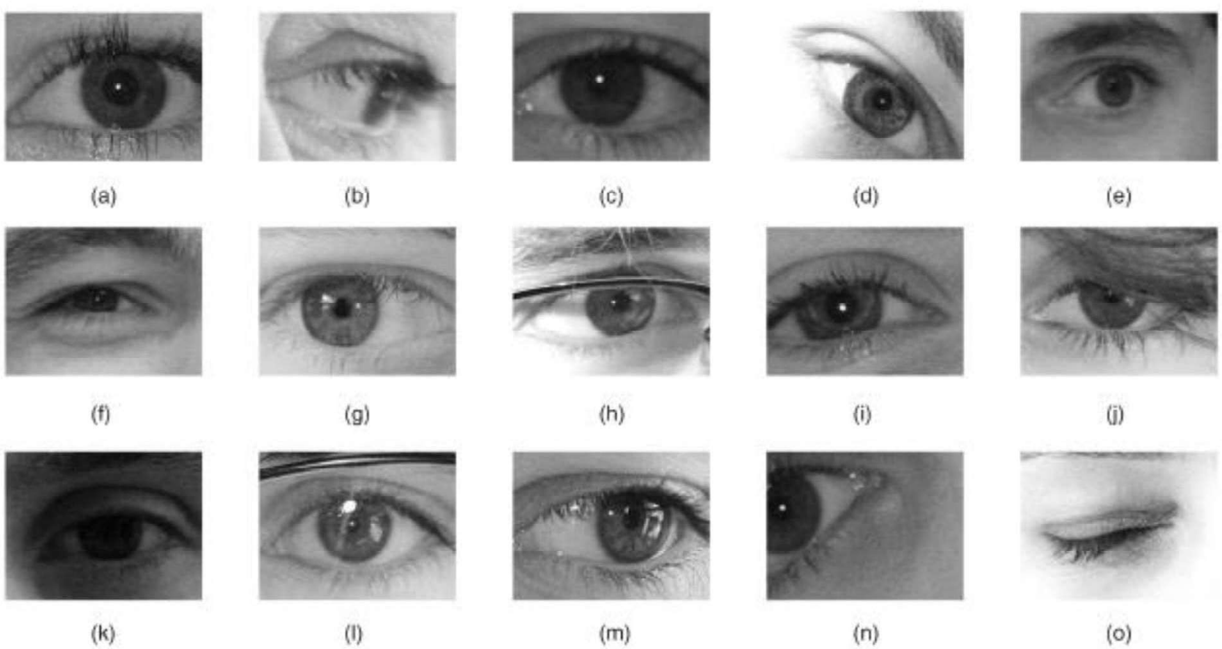


Figure 1.3 Examples of Non ideal Images.

1.3 Tools and Platforms

OS Used:	Windows 10
OS Manufacturer:	Microsoft Corporation
Processor:	Intel(R) Core™ i5-6200U CPU @ 2.20GHz
Installed Memory (RAM):	8.00 GB(7.89 GB usable)
System Type:	64-bit Operating System, x64-based processor
System Environment:	OpenCV 2.4
Language Used:	Python 3.7.0
IDE Used:	Python 3.7.0 Shell

Chapter 2

Literature Survey

This chapter provides an overview of the papers that studied the iris recognition system.

Previous studies on iris recognition

Rashad, Shams and El-Awady [9] proposed Local Binary Pattern (LBP) algorithm to extract features and combined two classifiers; Artificial Neural Network (ANN) classifier and Learning Vector Quantization (LVQ) classifier for classification. The combination produced a hybrid pattern depending on both features. On the other hand, in localization stage and segmentation stage, Hough Circular Transform and Canny edge detection are applied to separate an iris from an entire eye as well as for noise detection. Features which are extracted from iris by LBP are classified by both LVQ & ANN classifiers with different classes in order to determine the minimum agreeable performance. Three iris databases are combined and used in this work; CASIA, MMU1, MMU2. The proposed method achieved a recognition rate of 99.87%.

Savithiri and Murugan [10] made a comparison between different techniques on feature extraction for iris recognition, whole iris template and half template. They used Gabor Wavelet (GW), Histogram Oriented Gradient (HOG), and Local Binary Pattern (LBP) to extract the features from a specific portion of the iris in order to improve the performance of iris recognition. Their basic aim was to choose the half portion of the iris to recognize instead of entire extension of the iris. The proposed techniques are evaluated by using (FAR) and (FRR). Moreover, 130 images from MMU database were selected for these experiments. These images divided into 26 classes, and each class has 5 images. The first image from each class is chosen to be the template. The remaining 4 images of each class are selected to be the test set. Hamming Distance (HD) classifier is applied to classify the extracted features. GW+HD method achieved recognition rate of 99.94%, FAR 0.006%, LBP+HD method achieved 99.90%, and HOG+HD method achieved 99.95%.

Manisha and Sanjay [11] suggested different method for iris recognition by using Daugman's Integro-Differential Operator technique (IDO) to segment the iris andDaugman's Rubber Sheet technique (DRS) is used for iris normalization. On the other hand, Haar Transform (HT) algorithm is applied to extract features from iris region. Finally, Hamming Distance classifier (HD) is used to classify iris. The proposed method is applied on two databases, MMU and BATH together and the achieved recognition rate is 99.94%. Sarode and Patil [22] used Canny Edge Detection technique for segmentation in order to determine boundaries of the iris, then used local Binary Pattern(LBP) to extract the features from the iris image. Finally, they used two classifiers to classify the features extracted from the iris image; these classifiers are K_ Nearest Neighbors classifier (KNN) and Navie Bayes classifier (NB). These experiments were applied on MMU database. LBP+KNN method achieved 100% and LBP+NB method achieved 94.18% of recognition.

Kovoor, Supriya and Jacob [12] suggested a technique based on Daugman's approach. The technique is applied to examine the iris segmentation mechanisms, eyelid detection, and additionally to remove the area undesired. Haar Wavelet Transform (HWT) is used to encode the iris region to obtain the iris code that contains the best feature in iris pattern. It is concluded that Canny Edge Detector (CED) is the best approach to extract the edges to produce the iris code for the comparison. On other hand, Hamming Distance classifier (HD) is applied to classify the features. In this work, they have used UBIRIS database which includes 1877 images from 241 persons, and the achieved accuracy is 89% of recognition rate. In order to improve the efficiency of iris recognition system, a unique combination of three feature extraction techniques was proposed by Mrinalini, Pratusha, Manikantan and Ramachandran [34]. These techniques are Triangular DCT based Feature Extraction (T-DCT), Binary Particle Swarm Optimization (BPSO) based feature selection, and Radon Transform (RT) based Pre-processing. The proposed techniques were applied on three databases: MMU which consists of 225 images, IITD which includes 1120 images, and UPOL database which contains 384 images. Moreover, Euclidean Distance (ED) Classifier is used to evaluate the techniques performance which achieved accuracy of 78.04% on MMU database, 88.89% on UPOL database, and 94.04% on IITD database.

Aalaa, Israa and Ja'far [13] reviewed four feature extraction approaches, HOG, Gabor, DCT, and GLCM. These approaches are used to extract features from iris image. UBIRIS database version1 is used to test the proposed approaches. They used Logistic Model Trees (LMT) classifier to classify the extracted features. Results show that GLCM approach is better than other approaches. The approaches achieved accuracy 20% by HOG, 76% by combined (Gabor+ DCT), 96% by GLCM, and 92% by combined (HOG+ Gabor+ DCT+ GLCM).

Chapter 3

Concept and Problem Analysis

3.1 Concept

The word ‘Biometric’ is a two sections terminology, is taken from the Greek word, of which ‘Bio’ means life and ‘Metric’, mean measure. By combining these two words, ‘Biometric’ can be defined as the measure (study) of life, which includes humans, animals, and plants.“Biometric technologies” defined as automated methods of verifying or recognizing the identity of a living person based on a physiological or behavioral characteristic.

The science of using humans for the purpose of identification dates back to the 1875 when the measurement system of Alphonse Bertillon was proposed. Bertillon’s system of body measurements, including skull diameter and arm and foot length, was used in the USA to identify prisoners until the 1925. Before that William Herschel and Sir Francis Galton proposed quantitative identification through fingerprint and facial measurements in the 1880s. The development of digital signal processing techniques in the 1960s led immediately to work in automating human identification. Speaker and fingerprint recognition systems were among the first to be applied. The potential for application of this technology to high-security access control, personal locks and financial transactions was recognized in the early 1960s.
The 1970s saw development and deployment of hand geometry systems, the start of a large-scale testing and increasing interest in government use of these automated personal identification technologies. Then, Retinal and signature verification systems came in the 1980s, followed by face systems. Lastly, Iris recognition systems were developed in the 1990s.

Technologies	**Accuracy**	**Cost**	**Social Acceptability**	**Devices**
DNA	High	High	High	Lab Test
Iris	High	High	Medium-Low	Camera
Face	Medium-Low	Medium	Low	Camera
Voice	Medium	Medium	Medium	Microphone
Finger Print	High	Medium	High	Scanner
Hand Gesture	Medium-Low	Low	Medium	Scanner
Signature	Low	Medium	High	Optical pen

Table3.1: Comparison Of Different Biometric Technique

The idea of using the iris as a biometric is over 100 years old. However, the idea of automating iris recognition is more recent. In 1987, Flom and Sa obtained a patent for an unimplemented conceptual design of an automated iris biometrics system. Image processing techniques can be used 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 later. This biometric template contains an objective mathematical representation of the unique information stored in the iris, and allows comparisons to be made between templates. Our project is up till segmentation of the iris image.

One of the major challenges of automated iris recognition systems is to capture a high quality image of iris while remaining noninvasive to the human operator. Moreover, capturing the rich details of iris patterns, an imaging system should resolve a minimum of 70 pixels in iris radius. In the field trials to date, a resolved iris radius of 80–130 pixels has been more typical. Monochrome CCD cameras (480×640) have been widely used because Near Infrared (NIR) illumination in the 700–900-nm band was required for imaging to be non-intrusive to humans. Some imaging platforms deployed a wide-angle camera for coarse localization of eyes in faces; to steer the optics of a narrow- angle pan/tilt camera that acquired higher resolution images of eyes. Moreover, we have worked with non ideal iris images which include off-angle iris images, eyelid & eyelash blocking the iris.

The WVU off-angle and off-axis data set was collected from about 100 people. For each eye of a person, two images were acquired from a front angle and two from an off-angle. The off-angles were preset to be 15° and 30° , however, those angles are not specifically used in locating the iris due to varying gaze and head positions. Each image is a grayscale image, which helps to reduce the processing time and complexity, as opposed to operating on an RGB image.

3.2 Image Processing

3.2.1 Thresholding

1. **Simple Thresholding**

If pixel value is greater than a threshold value, it is assigned one value (may be white), else it is assigned another value (may be black). First argument is the source image, which should be a grayscale image. Second argument is the threshold value which is used to classify the pixel values. Third argument is the maxVal which represents the value to be given if pixel value is more than (sometimes less than) the threshold value. Different types are:

1. Binary Threshold
2. Binary Threshold Inverse
3. Tozero Threshold
4. Turnc Threshold
5. Tozero Threshold Inverse

Two outputs are obtained. First one is a retVal which will be explained later. Second output is our thresholded image.

2. **Adaptive Thresholding**

In the previous section, we used a global value as threshold value. But it may not be good in all the conditions where image has different lighting conditions in different areas. In that case, we go for adaptive thresholding. In this, the algorithm calculate the threshold for a small regions of the image. So we get different thresholds for different regions of the same image and it gives us better results for images with varying illumination.

Adaptive Method - It decides how thresholding value is calculated.

1. Adaptive Mean Threshold: threshold value is the mean of neighbourhood area.
2. Adaptive Gaussian Threshold: threshold value is the weighted sum of neighbourhood values where weights are a gaussian window.

Block Size - It decides the size of neighbourhood area.

C - It is just a constant which is subtracted from the mean or weighted mean calculated.

Otsu's Binarization

Here is a second parameter retVal. Its use comes when we go for Otsu's Binarization.
In global thresholding, we used an arbitrary value for threshold value, right? So, how can we know a value we selected is good or not? Answer is trial and error method. But consider a bimodal image (In simple words, bimodal image is an image whose histogram has two peaks). For that image, we can approximately take a value in the middle of those peaks as threshold value, right ? That is what Otsu binarization does. So in simple words, it automatically calculates a threshold value from image histogram for a bimodal image. (For images which are not bimodal, binarization won't be accurate.)
Since we are working with bimodal images, Otsu's algorithm tries to find a threshold value (t) which minimizes the weighted within-class variance given by the relation :

$$\sigma_w^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t)$$

$$q_1(t) = \sum_{i=1}^{t} P(i) \quad \& \quad q_1(t) = \sum_{i=t+1}^{I} P(i)$$

$$\mu_1(t) = \sum_{i=1}^{t} \frac{iP(i)}{q_1(t)} \quad \& \quad \mu_2(t) = \sum_{i=t+1}^{I} \frac{iP(i)}{q_2(t)}$$

$$\sigma_1^2(t) = \sum_{i=1}^{t} [i - \mu_1(t)]^2 \frac{P(i)}{q_1(t)} \quad \& \quad \sigma_2^2(t) = \sum_{i=t+1}^{I} [i - \mu_1(t)]^2 \frac{P(i)}{q_2(t)}$$

It actually finds a value of t which lies in between two peaks such that variances to both classes are minimum.

3.2.2 Smoothing Images

1. **2D Convolution (Image Filtering)**

As for one-dimensional signals, images also can be filtered with various low-pass filters (LPF), high-pass filters (HPF),etc. A LPF helps in removing noise, or blurring the image. A HPF filters helps in finding edges in an image.

$$K = \frac{1}{25} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Filtering with the above kernel results in the following being performed: for each pixel, a 5x5 window is centered on this pixel, all pixels falling within this window are summed up, and the result is then divided by 25. This equates to computing the average of the pixel values inside that window. This operation is performed for all the pixels in the image to produce the output filtered image.

2. **Image Blurring (Image Smoothing)**

Image blurring is achieved by convolving the image with a low-pass filter kernel. It is useful for removing noise. It actually removes high frequency content (e.g: noise, edges) from the image

resulting in edges being blurred when this is filter is applied. (Well, there are blurring techniques which do not blur edges).
Blurring techniques:

(a) Averaging

This is done by convolving the image with a normalized box filter. It simply takes the average of all the pixels under kernel area and replaces the central element with this average. We should specify the width and height of kernel.

(b) Gaussian Filtering

In this approach, instead of a box filter consisting of equal filter coefficients, a Gaussian kernel is used. We should specify the width and height of the kernel which should be positive and odd. We also should specify the standard deviation in the X and Y directions, sigmaX and sigmaY respectively. If only sigmaX is specified, sigmaY is taken as equal to sigmaX. If both are given as zeros, they are calculated from the kernel size. Gaussian filtering is highly effective in removing Gaussian noise from the image.

(c) Median Filtering

Here, the operation computes the median of all the pixels under the kernel window and the central pixel is replaced with this median value. This is highly effective in removing salt-and-pepper noise. One interesting thing to note is that, in the Gaussian and box filters, the filtered value for the central element can be a value which may not exist in the original image. However this is not the case in median filtering, since the central element is always replaced by some pixel value in the image. This reduces the noise effectively. The kernel size must be a positive odd integer.

(d) Bilateral Filtering

As we noted, the filters we presented earlier tend to blur edges. This is not the case for the bilateral filter which was defined for, and is highly effective at noise removal while preserving edges. Butthe operation is slower compared to other filters. We already saw that a Gaussian filter takes the a neighborhood around the pixel and finds its Gaussian weighted average. This Gaussian filter is a function of space alone with nearby pixels are considered while filtering. It does not consider whether pixels have almost the same intensity value and does not consider whether the pixel lies on an edge or not. The resulting effect is that Gaussian filters tend to blur edges, which is undesirable. The bilateral filter also uses a Gaussian filter in the space domain, but it also uses one more (multiplicative) Gaussian filter component which is a function of pixel intensity differences. The Gaussian function of space makes sure that only pixels are 'spatial neighbors' are considered for filtering, while the Gaussian component applied in the intensity domain (a Gaussian function of intensity differences) ensures that only those pixels with intensities similar to that of the central pixel ('intensity neighbors') are included to compute the blurred intensity value. As a result, this method preserves edges, since for pixels lying near edges, neighboring pixels placed on the other side of the edge, and therefore exhibiting large intensity variations when compared to the central pixel, will not be included for blurring.

3.2.3 Morphological Transformations

Morphological transformations are some simple operations based on the image shape. It is normally performed on binary images. It needs two inputs, one is our original image, second one is called structuring element or kernel which decides the nature of operation. Two basic morphological operators are Erosion and Dilation. Then its variant forms like Opening, Closing, Gradient etc also comes into play. We will see them one-by-one with help of following image:

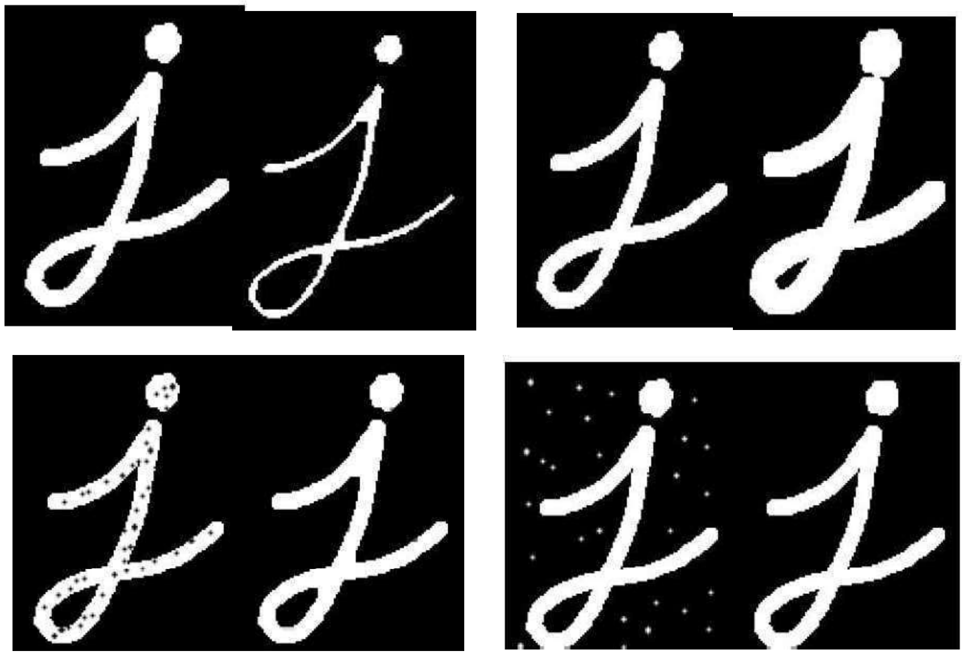


Figure3.1: Morphological Operations: (a)Erosion, (b)Dilation, (c)Opening and,(d) Closing.

Erosion

The basic idea of erosion is just like soil erosion only, it erodes away the boundaries of foreground object (Always try to keep foreground in white). So what does it do? The kernel slides through the image (as in 2D convolution). A pixel in the original image (either 1 or 0) will be considered 1 only if all the pixels under the kernel is 1, otherwise it is eroded (made to zero). So what happens is that, all the pixels near boundary will be discarded depending upon the size of kernel. So the thickness or size of the foreground object decreases or simply white region decreases in the image. It is useful for removing small white noises (as we have seen in colorspace chapter), detach two connected objects etc.

Dilation

It is just opposite of erosion. Here, a pixel element is '1' if atleast one pixel under the kernel is '1'. So it increases the white region in the image or size of foreground object increases. Normally, in cases like noise removal, erosion is followed by dilation. Because, erosion removes white noises, but it also shrinks our object. So we dilate it. Since noise is gone, they won't come back, but our object area increases. It is also useful in joining broken parts of an object.

Opening

Opening is just another name of erosion followed by dilation. It is useful in removing noise, as we explained above.

Closing

Closing is reverse of Opening, Dilation followed by Erosion. It is useful in closing small holes inside the foreground objects, or small black points on the object.

Morphological Gradient

It is the difference between dilation and erosion of an image.

Top Hat

It is the difference between input image and Opening of the image.

Black Hat

It is the difference between the closing of the input image and input image.

3.2.4 Canny Edge Detection

Canny Edge Detection is a popular edge detection algorithm. It was developed by John F. Canny in 1986. It is a multi-stage algorithm and we will go through each stages.

1. **Noise Reduction**

Since edge detection is susceptible to noise in the image, first step is to remove the noise in the image with a 5x5 Gaussian filter.

2. **Finding Intensity Gradient of the Image**

Smoothened image is then filtered with a Sobel kernel in both horizontal and vertical direction to get first derivative in horizontal direction (Gx) and vertical direction (Gy). From these two images, we can find edge gradient and direction
for each pixel as follows:

$$Edge\_Gradient\ (G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle\ (\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

Gradient direction is always perpendicular to edges. It is rounded to one of four angles representing vertical, horizontal and two diagonal directions.

3. **Non-maximum Suppression**

After getting gradient magnitude and direction, a full scan of image is done to remove any unwanted pixels which may not constitute the edge. For this, at every pixel, pixel is checked if it is a local maximum in its neighborhood in the direction of gradient. Check the image below:

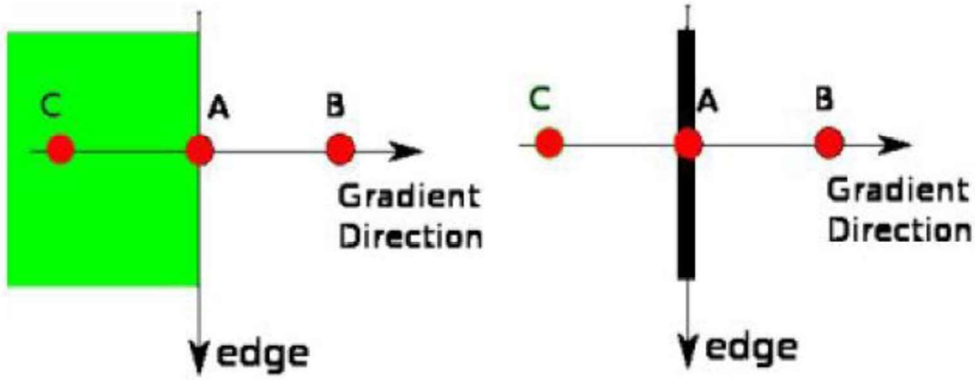


Figure3.2: CED Steps.

Point A is on the edge (in vertical direction). Gradient direction is normal to the edge. Point B and C are in gradient directions. So point A is checked with point B and C to see if it forms a local maximum. If so, it is considered for next stage, otherwise, it is suppressed (put to zero).
In short, the result you get is a binary image with "thin edges".

4. **Hysteresis Thresholding**

This stage decides which are all edges are really edges and which are not. For this, we need two threshold values, minVal and maxVal. Any edges with intensity gradient more than maxVal are sure to be edges and those below minVal are sure to be non-edges, so discarded. Those who lie between these two thresholds are classified edges or non-edges based on their connectivity. If they are connected to "sure-edge" pixels, they are considered to be part of edges. Otherwise, they are also discarded.
The edge A is above the maxVal, so considered as "sure-edge". Although edge C is below maxVal, it is connected to edge A, so that also considered as valid edge and we get that full curve. But edge B, although it is above minVal and is in same region as that of edge C, it is not connected to any "sure-edge", so that is discarded. So it is very important that we have to select minVal and maxVal accordingly to get the correct result. This stage also removes small pixels noises on the assumption that edges are long lines. So what we finally get is strong edges in the image.

3.2.5 Hough Circle Transform

The Hough transform may be used to detect circular shapes in images, after binarisation, for example by an edge detector. Often, functions to do this operation require the radius of the circle to be specified. The function circle_hough allows a range of radii to be specified, so that the radius does not need to be known exactly in advance. It is likely to be faster than calling a standard function repeatedly for different radii.
A circle is represented mathematically as

$$(x-a)^2 + (y-b)^2 = r^2$$

From equation, we can see we have 3 parameters, so we need a 3D accumulator for hough transform, which would be highly ineffective.

3.2.6 Image Segmentation with Watershed Algorithm

Any grayscale image can be viewed as a topographic surface where high intensity denotes peaks and hills while low intensity denotes valleys. You start filling every isolated valleys (local minima) with different colored water (labels). As the water rises, depending on the peaks (gradients) nearby, water from different valleys, obviously with different colors will start to merge. To avoid that, you build barriers in the locations where water merges. You continue the work of filling water and building barriers until all the peaks are under water. Then the barriers you created give you the segmentation result. This is the "philosophy" behind the watershed.

But this approach gives you over segmented result due to noise or any other irregularities in the image. So OpenCV implemented a marker-based watershed algorithm where you specify which are all valley points are to be merged and which are not. It is interactive image segmentation. What we do is to give different labels for our object we know. Label the region which we are sure of being the foreground or object with one color (or intensity), label the region which we are sure of being background or non-object with another color and finally the region which we are not sure of anything, label it with 0. That is our marker. Then apply watershed algorithm. Then our marker will be updated with the labels we gave, and the boundaries of objects will have a value of -1.

3.2.7 Interactive Foreground Extraction using GrabCut Algorithm

GrabCut algorithm was designed by Carsten Rother, Vladimir Kolmogorov & Andrew Blake from Microsoft Research Cambridge, UK. in their paper, "GrabCut": interactive foreground extraction using iterated graph cuts . An algorithm was needed for foreground extraction with minimal user interaction, and the result was GrabCut. How it works from user point of view? Initially user draws a rectangle around the foreground region (foreground region shoule be completely inside the rectangle). Then algorithm segments it iteratively to get the best result.

So what happens in background?

- User inputs the rectangle. Everything outside this rectangle will be taken as sure background (That is the reason it is mentioned before that your rectangle should include all the objects). Everything inside rectangle is unknown. Similarly any user input specifying foreground and background are considered as hard-labeling which means they won't change in the process.
- Computer does an initial labeling depending on the data we gave. It labels the foreground and background pixels (or it hard-labels)
- Now a Gaussian Mixture Model(GMM) is used to model the foreground and background.
- Depending on the data we gave, GMM learns and create new pixel distribution. That is, the unknown pixels are labeled either probable foreground or probable background depending on its relation with the other hard labeled pixels in terms of color statistics (It is just like clustering).
- A graph is built from this pixel distribution. Nodes in the graphs are pixels. Additional two nodes are added,Source node and Sink node. Every foreground pixel is connected to Source node and every background pixel is connected to sink node.
- The weights of edges connecting pixels to source node/end node are defined by the probability of a pixel being foreground/background. The weights between the pixels are defined by the edge information or pixel similarity.

If there is a large difference in pixel color, the edge between them will get a low weight.

- Then a mincut algorithm is used to segment the graph. It cuts the graph into two separating source node and sink node with minimum cost function. The cost function is the sum of all weights of the edges that are cut. After the cut, all the pixels connected to Source node become foreground and those connected to sink node become background.

• The process is continued until the classification converges.

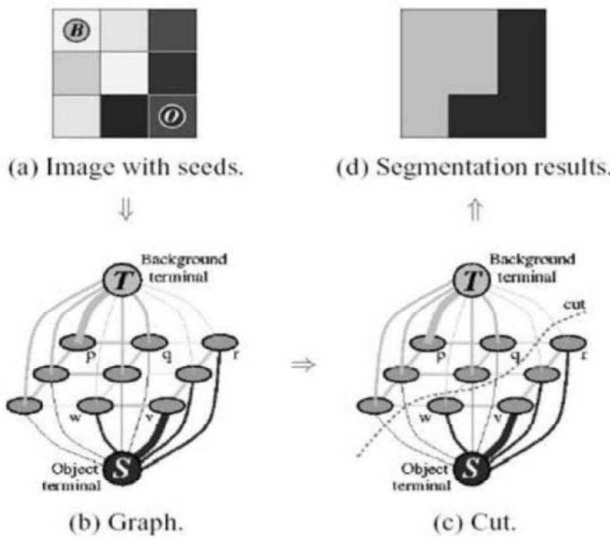


Figure3.3: Grab-cut Algorithm

Chapter 4

Methodology

In this paper, we have used OpenCV2 library module in python3.7.0.

4.1 Databases

The major purpose of the UBIRIS.v2 database is to constitute a new tool to evaluate the feasibility of visible wavelength iris recognition under far from ideal imaging conditions. In this scope, the various types of non-ideal images, imaging distances, subject perspectives and lighting conditions existent on this database could be of strong utility in the specification of the visible wavelength iris recognition feasibility and constraints.

Image Acquisition Framework and Set-Up	
Camera = Canon EOS 5D	Color Representation = sRGB
Shutter Speed = 1/197 sec.	Lens Aperture = F/6.4 - F/7
Focal Length = 400 mm	F-Number = F/6.3 - F/7.1
Exposure Time = 1/200 sec.	ISO Speed = ISO-1600
Metering Mode = Pattern	
Details of the Manually Cropped Resultant Images	
Width = 400 pixels	Height = 300 pixels
Format = tiff	Horizontal Resolution = 72 dpi
Vertical Resolution = 72 dpi	Bit Depth = 24 bit
Volunteers	
Totals = Subjects 261; Irises 522; Images 11 102	Gender = Male: 54.4%; Female: 45.6%
Age = [0,20]: 6.6% [21,25]: 32.9% [26,30]: 23.8% [31,35]: 21.0% [36,99]: 15.7%	Iris Pigmentation = Light : 18.3% Medium : 42.6% Heavy : 39.1%

Table4.1: UBIRIS .v2

4.2 Image Preprocessing

Achieving high performance of iris recognition system requires overcoming some of the major difficulties, such as choosing the appropriate database and unifies dimensions of the image, and recruits a sufficient number of images in each experiment. In this section, the technique used in Grab cut, and unify dimensions for images in each database will be explained.

4.2.1 Grab Cut Algorithm

GrabCut [Rother et al. 2004] is an iterative image segmentation technique based upon the Graph Cut algorithm [Boykov and Jolly 2001]. GrabCut extends Graph Cut to color images and to incomplete trimaps. These developments greatly increase the usefulness of Graph Cut. User interaction is simplified to drawing a rectangle around the desired foreground, followed by a small amount of corrective editing. The inclusion of color information in the Graph Cut algorithm and the iterative learning approach increases its robustness. Thus, GrabCut is a very promising image editing tool for foreground extraction.

GrabCut requires four different pieces of information for each pixel. In our implementation, each piece is stored in its own array. Each array is the same size as the original image. Variable names that occur in the original paper appear in parenthesis after the description.

- Color – an RGB value (z)
- Trimap – either TrimapUnknown, TrimapBackground, or TrimapForeground
- Matte – in the initial hard segmentation step, either MatteBackground or MatteForeground (α)
- Component Index – a number between 1 and K, where K is the number of Gaussian components in a GMM (k)

In addition to these, GrabCut also requires K Gaussian components for each of the foreground and background GMMs. For each component, we store:

- μ – the mean (an RGB triple)
- Σ^{-1} – the inverse of the covariance matrix (a 3x3 matrix)
- $\det\Sigma$ – the determinant of the covariance matrix (a real)
- π – a component weight (a real)

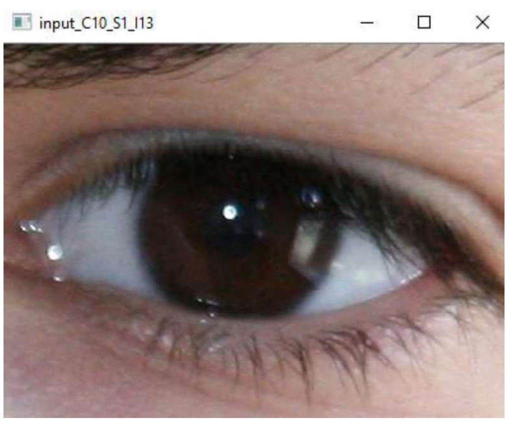


Figure4.1: Input Image

Figure4.2: Grab-cut Image

The initialization process includes steps 1-3. In step 1, the user initializes the trimap by selecting a rectangular region around the object of interest. Pixels inside the rectangle are marked as TrimapUnknown. Pixels outside are marked as TrimapBackground. This is the initial information given to the algorithm. In step 2, the matte is initialized to MatteBackground in the TrimapBackground set and to MatteForeground in the TrimapUnknown set. This distinction between the trimap and the matte formalizes the separation between the user input, which is regarded as correct, and the segmentation derived by the GrabCut algorithm, which may be incorrect. In step 3, given the initialized matte, we create the K components of the Gaussian Mixture Models (GMM) for the MatteForeground and MatteBackground regions. To clarify, this means that we must create a total of 2K components. We first divide both regions into K pixel clusters. The Gaussian components are then initialized from the colors in each cluster. For good separation between foreground and background, it is necessary that we generate low variance Gaussian components. Thus, we seek to find tight, well-separated clusters. There are a wide variety of clustering algorithms that could be used for this step. Guided by Ruzon and Tomasi [2000] and Chuang et al. [2001] we use the color quantization technique described by Orchard and Bouman [1991]. This technique uses the eigenvector of the color covariance matrix to determine good cluster splits.

4.2.2 Noise Reduction

4.2.2.1 Black Hat

Black hat is a two stepped approach to enhance those black pixels in the white background. The first step is making the closing of the image and then subtracting it with the original image. The closing function has two morphological operators, namely, erosion and dilation. In order to simplify the image structure while avoiding the expansion effects of dilation, one can perform erosion after the dilation. The resulting operation is called closing:

$f \bullet B = (f \oplus B) B$

In easy terms, closing is dilation followed by erosion. And hence closing removes "dark" details. The top-hat filter (or white top-hat filter) is used to enhance bright objects of interest in a dark background. The black-hat operation (or bottom-hat filter or black top-hat filter) is used to do the opposite, i.e., enhance dark objects of interest in a bright background. Essentially, the filters suppress large regions while keeping the small ones, as specified by the size of the structuring element. This could be used simply for visual purposes or as a pre-processing step in an automated image-analysis algorithm.

Figure 4.3: Black Hat Image

4.2.2.2 Median Blur

The best known order-statistics filter is the median filter, which replaces the value of a pixel by the median of the gray levels in the neighborhood of that pixel. The kernel size must be a positive odd integer.

$$\hat{f}(x, y) = \frac{1}{mn} \sum_{(s,t)\in S_{xy}} g(s, t)$$

The original value of the pixel is included in the computation of the median. Median filters are quite popular because, for certain types of random noise they provide excellent noise reduction capabilities, with considerably less blurring than linear smoothing filters of similar size [14][15]. The filter is mainly used to remove the following noises:

A. *Gaussian Noise*

The PDF of a Gaussian random variable, z is given by

$$p(z) = \frac{1}{2\pi\sigma} e^{-(z-\mu)^2/2\sigma^2}$$

Where z represents gray level, μ is the mean of average value of z, and σ is its standard deviation. The standard deviation squared, σ* σ is called the variance of z. Because of its mathematical tractability in both the spatial and frequency domains, Gaussian (also called normal) noise models are frequently used in practice. In fact, this tractability is so convenient that it often results in Gaussian models being used in situations in which they are marginally applicable at best.

B. Impulse (salt-and-pepper) noise

The PDF of impulse noise is given by

$$p(z) = \begin{cases} P_a & \text{for } z = a \\ P_b & \text{for } z = b \\ 0 & \text{otherwise} \end{cases}$$

If b > a, gray-level b will appear as a light dot in the image. Conversely the dot will appear like a dark dot. If either Pa or Pb is zero, the impulse noise is called uni-polar. If neither probability is zero and especially if they are approximately equal impulse noise values will resemble salt-and-pepper granules randomly distributed over the image. For this reason bipolar impulse noise is also called salt-and-pepper noise. Shot and spike noise terms are also used to refer this type of noise. Noise impulses can be negative or positive. Scaling usually is part of the image digitizing process. Because impulse corruption usually is large compared with the strength of the image signal, impulse noise generally is digitized as extreme (pure white or black) values in an image. Thus the assumption usually is that a and b are “saturated” values in the sense that they are equal to the minimum and maximum allowed values in the digitized image. As a result, negative impulses appear as black (pepper) points in an image. For the same reason, positive impulses appear white (salt) noise. For an 8-bit image this means that a = 0 (black) and b = 255 (white).

4.2.3 Canny Edge Detection

For detection of the edge pixels in the images canny edge detection process is employed. The Canny edge detector is an edge detection operator that uses a multistage algorithm to detect a wide range of edges in images.

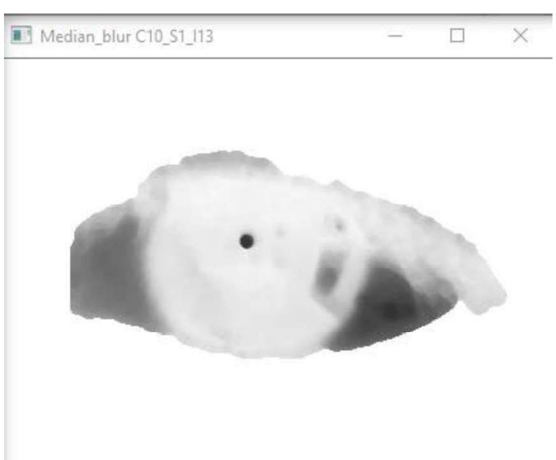


Figure4.4:Median Blur

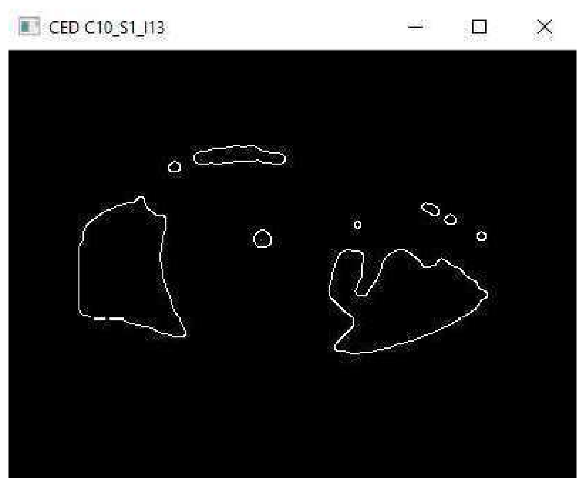


Figure4.5:CED image

4.3 Localization iris region

After all the image enhancement, noise reduction and edge detection, we move on to find out the location of the iris by the edges provided by the canny image from last section. The circle detection of an image is a common problem in machine vision. In industrial manufacturing, the circular targets are needed to be identified precisely such as identifying the circuit welding holes. Circular Hough gradient transform is the most common method of circle detection, and a rapid algorithm has been implemented in Opencv2.

4.3.1 Circular Hough Transformation

In a two-dimensional space, a circle can be described by:

$$(x-a)^2 + (y-b)^2 = r^2$$

where (a,b) is the center of the circle, and r is the radius. If a 2D point (x,y) is fixed, then the parameters can be found according to (1). The parameter space would be three dimensional, (a, b, r). And all the parameters that satisfy (x, y) would lie on the surface of an inverted right-angled cone whose apex is at (x, y, 0). In the 3D space, the circle parameters can be identified by the intersection of many conic surfaces that are defined by points on the 2D circle. This process can be divided into two stages. The first stage is fixing radius then find the optimal center of circles in a 2D parameter space. The second stage is to find the optimal radius in a one dimensional parameter space. If the radius is fixed, then the parameter space would be reduced to 2D (the position of the circle center). For each point (x, y) on the original circle, it can define a circle centered at (x, y) with radius R according to the equation. The intersection point of all such circles in the parameter space would be corresponding to the center point of the original circle.

Since the parameter space of the CHT is three dimensional, it may require lots of storage and computation. Choosing a bigger grid size can ameliorate this problem. However, choosing an appropriate grid size is difficult. Since too coarse a grid can lead to large values of the vote being obtained falsely because many quite different structures correspond to a single bucket. Also, the CHT is not very robust to noise.

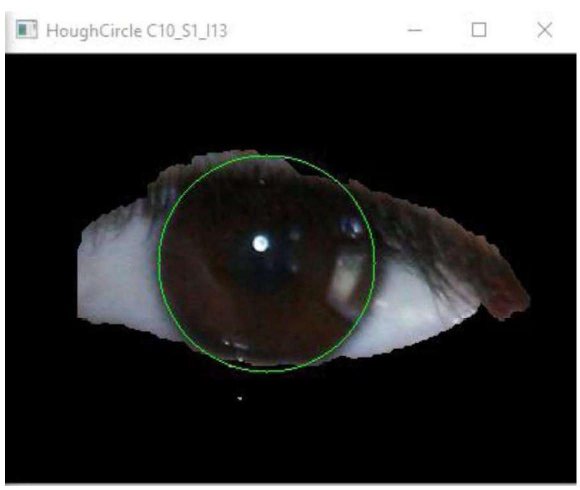


Figure4.6: Localized Image

4.4 Segmentation of Iris

Iris segmentation [16] refers to the next step after Automatic detection of the boundaries of iris and the boundaries of pupil of an iris in eye image in order to exclude the surrounding regions. This process helps in extracting the features from the iris for personal identification in a correct and clear manner. In other words, the major aim of segmentation is removing non useful regions such as the parts outside the iris (eyelids, eyelashes and skin) [17]. The success of the segmentation process depends on the quality of eye image. The segmentation process determines the iris boundaries and pupil boundaries and then converts this part to a suitable template in normalization stage.

Many studies have reviewed the iris segmentation, these studies have focused on developing the techniques which are used in iris segmentation in order to increase the performance.

We have a different approach towards cropping CHT section which is being localized in the previous section. Before explaining the approach, we already use the help of the localized iris from the last section. And so, We use simple concept of co-ordinate geometry just by importing one extra function of hypot included in the math module which is an inbuilt math function in Python that return the Euclidean norm,

$$\sqrt{(x * x + y * y)}$$

We use two loops for travelling from the very first pixel of the image[0,0] till the end pixel of the image[400,300] just to check the Euclidean distance between the centre of the found circle and the pixel is smaller than the radius of that circle. Then, the pixel intensity of that very pixel will be changed to zero or in other words, that pixel is turned black. This method of cropping can be implemented in order to get the image crop by the iris boundary and pupil boundary. In case of non-ideal iris images, in UBIRIS its very difficult to find the gamma value because the dataset has got a lot of off-angle, noisy and off-focused images which can be only segmented till the iris boundary and hence, the pupil boundary or inner boundary is not need to be cropped.

Figure4.7: Segmented Image

Chapter 5

Sample Code

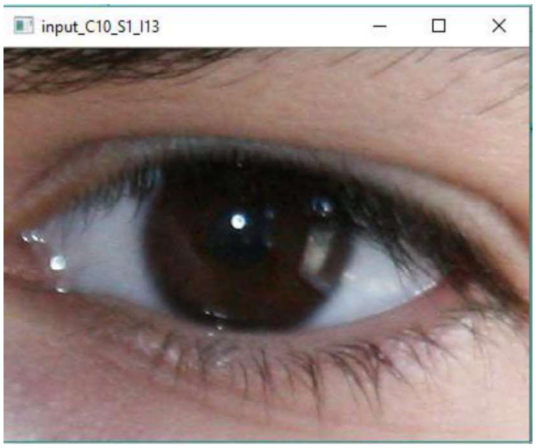
In this chapter, we provide only one sample code which can be used on any python IDE and one can easily include this file for a GUI identification system for iris recognition.

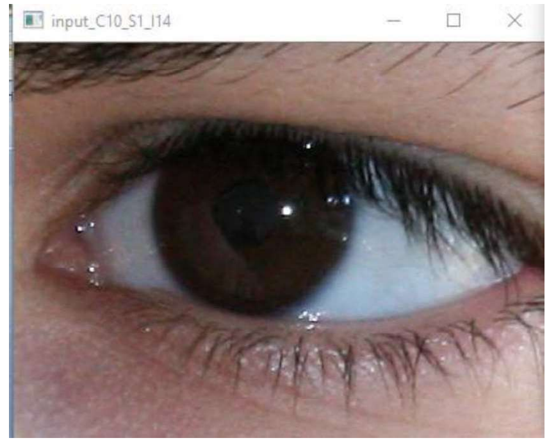
```
import cv2
import numpy as np
import os
import math
from math import hypot
gamma = -48
#gamma is -48 for UBIRIS database

def Grabcut(image):                          #function to differentiat foreground and background
    mask = np.zeros((image.shape[0],image.shape[1]),np.uint8)
    bgdModel = np.zeros((1,65),np.float64)
    fgdModel = np.zeros((1,65),np.float64)
    rect = (50,50,450,290)
    cv2.grabCut(image,mask,rect,bgdModel,fgdModel,5,cv2.GC_INIT_WITH_RECT)
    mask2 = np.where((mask==2)|(mask==0),0,1).astype('uint8')
    img = image*mask2[:,:,np.newaxis]
    return img

def noise_reduction(image):                  #all noise cancelling process for CHT
    inverted_gray = cv2.bitwise_not(image)
    kernel = np.ones((5,5),np.uint8)
    black_hat = cv2.morphologyEx(inverted_gray,cv2.MORPH_BLACKHAT,kernel)
    cv2.imshow("Black_hat"+name,black_hat)
    cv2.waitKey(0)
    no_reflec = cv2.add(inverted_gray,black_hat)
    median_blur = cv2.medianBlur(no_reflec,5)
    cv2.equalizeHist(median_blur)
    cv2.imshow("Median_blur "+name,median_blur)
    cv2.waitKey(0)
    retval,thres_image          =          cv2.threshold(cv2.bitwise_not(median_blur),50,255,
cv2.THRESH_BINARY_INV)
    canny = cv2.Canny(thres_image,200,100)
    return canny

images = []
names = []
 #running the complete set of image database from the folder "data"
for filename in os.listdir("data"):
    if filename is not None:
        image = cv2.imread(os.path.join("data",filename),1)
        images.append(image)
        names.append(filename.split('.')[0])
```

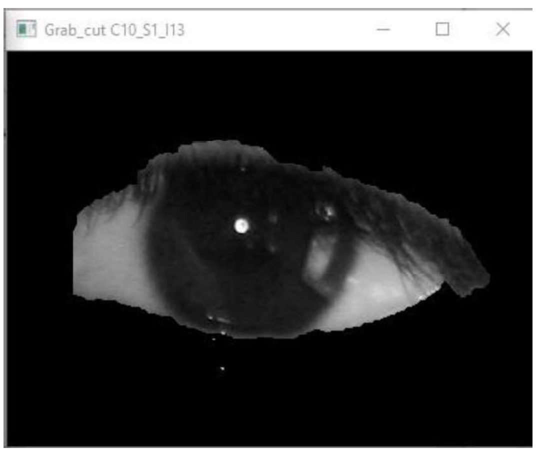
```
for i in range(len(images)):
    name = names[i]
    image = images[i]

    cv2.imshow("input_"+name,image)
    cv2.waitKey(0)
    x,y,z = image.shape
    image = Grabcut(image)
    gray_image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
    cv2.imshow("Grab_cut "+name,gray_image)
    cv2.waitKey(0)

    ipfilter = noise_reduction(gray_image)
    cv2.imshow("CED "+name,ipfilter)
    cv2.waitKey(0)
    circles = cv2.HoughCircles(ipfilter,cv2.HOUGH_GRADIENT,1,20,param1 = 200,param2 =
20,minRadius =0)

    if circles is not None:
        inner_circle = np.uint16(np.around(circles[0][0])).tolist()
    cv2.circle(image,(inner_circle[0],inner_circle[1]),inner_circle[2],(0,255,0),1)
    cv2.imshow("HoughCircle "+name,image)
    cv2.waitKey(0)

#crop image by comparing each pixel dsitance from center with radius
    for j in range(x):
        for k in range(y):
            if hypot(k-inner_circle[0], j-inner_circle[1]) >= inner_circle[2]:
                gray_image[j,k] = 0
    cv2.imshow("output2_"+name,gray_image)
    cv2.waitKey(0)

    cv2.destroyAllWindows()
```

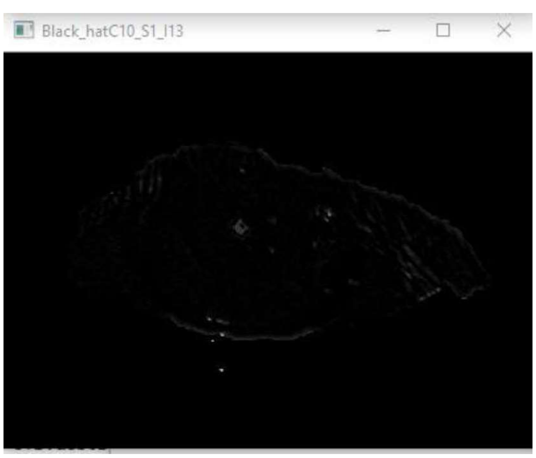
Chapter 6

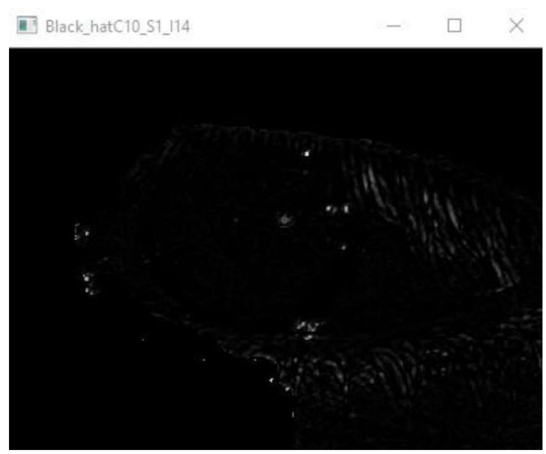
Output Images

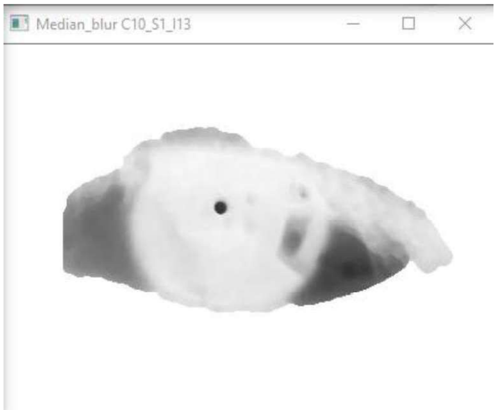


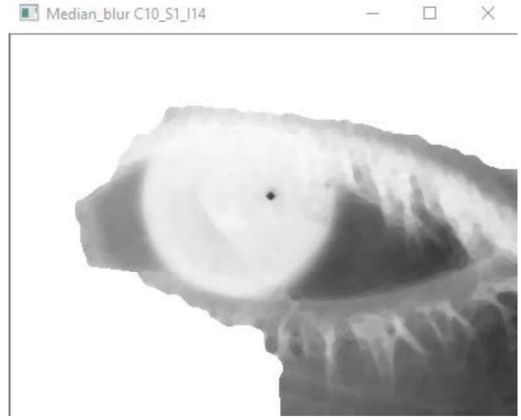


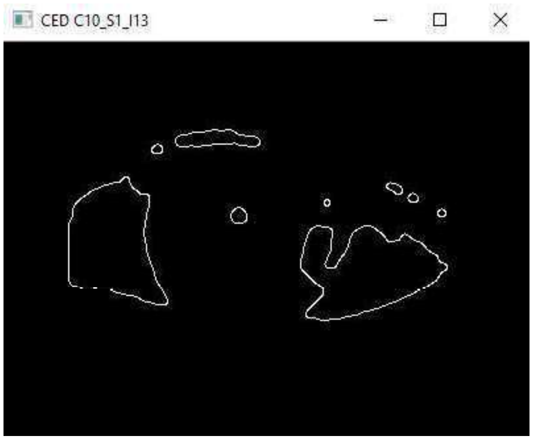


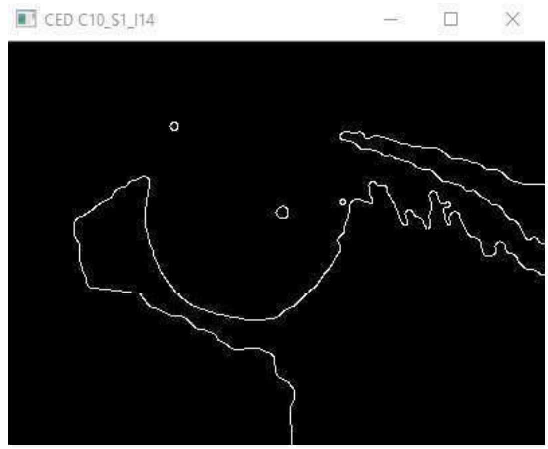


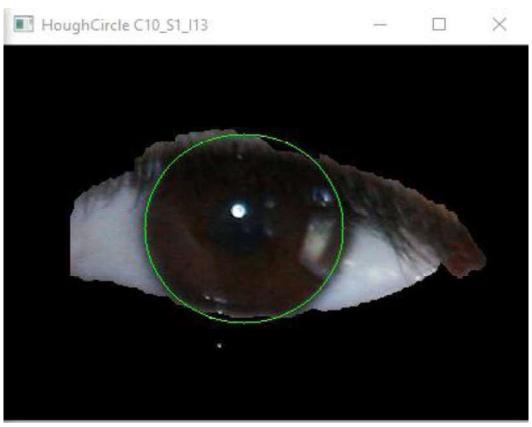


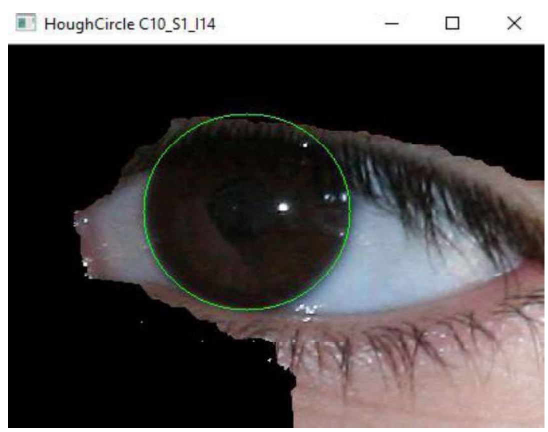

Median_blur C10_S1_I13

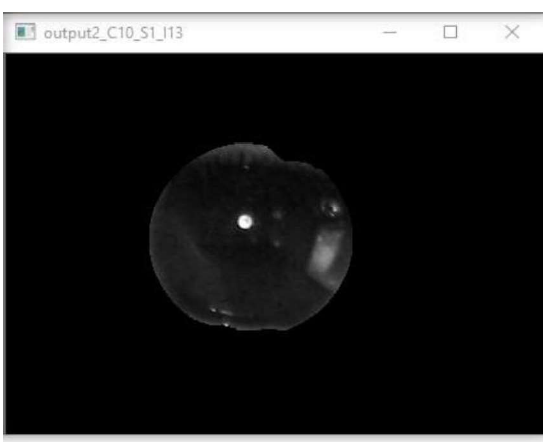

Median_blur C10_S1_I14

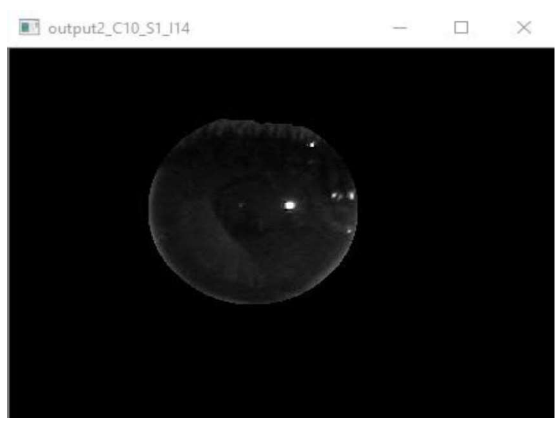

CED C10_S1_I13


CED C10_S1_I14


HoughCircle C10_S1_I13


HoughCircle C10_S1_I14


output2_C10_S1_I13


output2_C10_S1_I14

Chapter 7

Conclusion and Future Scopes

7.1 Conclusion

The proposed system recognizes the iris of the persons in the dataset based on the distance metrics. The proposed system gives accuracy which is higher than the existing algorithms which identifies that the misclassifications are reduced to a greater extend. The input iris images were taken from UBIRIS database. Basic preprocessing steps like resizing and noise removal process were employed. For noise removal process morphological operators and median filter is employed. Canny edge detection process is employed to the enhanced iris image. Image gradient is employed for the enhanced iris image. Hysteresis thresholding is employed for the iris and the pupil regions. Circular Hough transform is applied for the iris images in order to identify the iris and pupil regions. And then segment the image as the final results.

In this report, a part of iris recognition system is being implemented and can be used for the next step, i.e. Normalization, feature extraction and template matching. The resultant image or the segmented image can be easily stored by normalizing it and hence used for the feature extraction by using deep learning approaches such as K-means.

7.2 Future Scopes

The proposed methods are able to achieve high performance, but there are some issues that still hinder the achievement of better performance such as type of databases and quality of images in these databases. Also, the techniques that are used in preprocessing, segmentation and normalization stages. However, several suggestions are introduced below in order to improve the performance of the proposed system such as:

(a) Applying the proposed system on different databases such as UBIRIS database and compare the obtained results with the results in this thesis.

(b) Using feature extraction algorithms such as K-means,etc. compares the results with the results obtained in this report.

(c)Implementing the proposed system on a small organization; where the iris images are taken from the employees in this organization, and stored in the database, then apply the proposed system to identify the employees by their iris.

Chapter 8

References

1. Kharat, P. and M. Deshmukh, Iris Recognition: a Review. International Journal of Advanced Trends in Computer Science and Engineering. pp, 2013: p. 93-97.

2. Dhamala, P., Multibiometric systems. 2012, Ph. D. thesis,Norwegian University of Science and Technology.

3. Das, A., Recognition of Human Iris Patterns. 2012.

4. Cui, J., et al. A fast and robust iris localization method based on texture segmentation. In Defense and Security. 2004. International Society for Optics and Photonics.

5. Chirchi, V.R.E. and L. Waghmare, Feature Extraction and Pupil Detection Algorithm Used for Iris Biometric Authentication System. International Journal of Signal Processing, Image Processing and Pattern Recognition, 2013. 6(6): p. 141-160.

6. Patil, M.R.A., M.A. Karode, and M.S. Suralkar, Steps of Human Iris Detection for Biometric Application. Vol. 4, Issue 12, December 2015.

7. Bodade, R.M. and S.N. Talbar, Iris Analysis for Biometric Recognition Systems. 2014:Springer.

8. Sheela, S. and P. Vijaya, Iris recognition methods-survey. International Journal of Computer Applications, 2010. 3(5): p. 19-25.

9. Shams, M., et al., Iris recognition based on LBP and combined LVQ classifier. arXiv preprint arXiv:1111.1562, 2011.

10. Savithiri, G. and A. Murugan, Performance analysis on half iris feature extraction using GW, LBP and HOG. International journal of computer applications, 2011. 22(2): p. 27-32.

11. Khaladkar, M.M. and S.R. Ganorkar, A Novel Approach for Iris Recognition. vol. I, 2012(4).

12. Kovoor, B.C., M. Supriya, and K.P. Jacob, IRIS BIOMETRIC RECOGNITION SYSTEM EMPLOYING CANNY OPERATOR. 2013.

13. Albadarneh, A. and I. Albadarneh. Iris recognition system for secure authentication based on texture and shape features. in Applied Electrical Engineering and Computing Technologies (AEECT), 2015 IEEE Jordan Conference on. 2015. IEEE.

14. R. Yang, L. Lin, M. Gabbouj, J. Astola, and Y. Neuvo, "Optimal Weighted Median Filters Under Structural Constraints," IEEE Trans. Signal Processing, Vol. 43, PP. 591-604, Mar 1995

15. Pei-Eng Ng and Kai-Kuang Ma, "A Switching Median Filter with BDND for Extremely Corrupted Images", IEEE Trans Image Processing, Vol. 15, No. 6, PP. 1506-1516, June 2006.

16. Hu, X., V.P. Pauca, and R. Plemmons. Iterative directional ray-based iris segmentation for challenging periocular images. in Chinese Conference on Biometric Recognition. 2011. Springer.

17. Gupta, R. and A. Kumar, An Effective Segmentation Technique for Noisy Iris Images. International Journal of Application or Innovation in Engineering & Management, 2013. 2(12): p. 118-125.