**An Undergraduate Approach to Iris Recogntion**

**Abstract**

Biometrics refers to the identification and verification of human identity based on certain physiological traits of a person. Biometric identifiers have become increasingly popular in security for access control. Commonly used features include speech, fingerprint, face, handwriting, gait, hand geometry etc. The face and speech techniques have been used for over 25 years, while iris analysis is more recent. The iris recognition outperforms other methods of biometric identification in terms of accuracy. Iris recognition systems use iris textures as a unique identifier. Iris recognition uses object segmentation based on Hough transform. In this paper, the algorithm and solution for developing an iris recognition program will be discussed. An image of an eye is acquired and converted to grayscale. The image then has the iris and pupil localised, including any light reflections, or eyelashes. To normalize the image, we perform some preprocessing, and we use Daugman’s rubber sheet model, which turns the iris from a donut shape to a rectangle. And then encoding is performed on the iris using a gabor filter to get the features. A code is then made from the features and compared with ones in the database using hamming distance to determine matches.

**Introduction**

The iris is the only externally visible organ, and its pattern remains stable throughout the adult life. It is also estimated that the probability of any two people having the same iris pattern is close to zero. These characteristics are very beneficial for the use in biometrics in identifying individuals. Through the use of image processing, the iris patterns are extracted from an image and encoded in a way that can be easily stored and compared in a database. This provides a mathematical representation of the code in the iris. If a person would like their iris to be detected, a template is made of their iris region. This template is compared with other templates stored in the database and the program informs the user of a match if one is found.

**Algorithm**

Overview

The iris recognition process includes localization, normalization, encoding and comparison. The localization of the image is attempting to locate the iris and pupil centers along with their radii. The normalization of the image is an attempt at making the iris rectangular for encoding. The encoding is an attempt at generating a unique code for a specific iris. The comparison is an attempt at comparing two iris’ unique code to determine how similar they are, and from that, determine whether it is a good enough match or not.

Image preprocessing

The image is first converted to grayscale to have a uniform dataset, as well as make future processing easier.

1. Iris localization: In order to provide iris recognition, the iris must be found first. To do this the image is processed to have the iris and pupil localised, through edge detection, and the Hough transform to find the circles. The Hough transform works by looking at every edge pixel in the edge detection image, and computes every circle that can be made with it and the other pixels, and then plotting the result in the Hough space. The result of this graph will give the variables that give the most likely circle. Since the pupil and iris will show the biggest peaks in the transform since they would have the most complete circle, the pupil and iris can be found, along with their radius, and centers. The iris and pupil are not expected to have the same center since they are not concentric, but it is possible.
   1. Edge Detection:
      1. Utilize the canny operator by applying it to the image to obtain the edges of the image we are obtaining the iris code from. Figure 1 shows the result of applying the canny operator to the image.
   2. Obtaining Pupil and Iris Boundaries
      1. Use the built in function imfindcircles on the edge detected image to find all the circles in the edge detected image within the specified radius range. This function is implementation of Hough Circle Transform application. To find the all the circle that could be the pupil the radius range of 20 to 40 will be used and for the iris the radius range of 50 to 80. This function will return two variables the coordinate for the center of the circles it found and the its radii. From both results found by the two radii ranges select the top result for each which will result in both the circles that define the pupil and iris. Figure 2 shows the two detected circles overlaying the edge detect image to represent the locations of the pupil and iris.

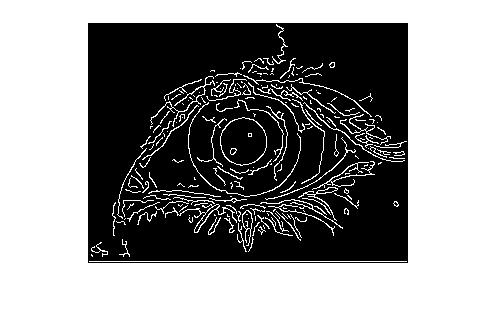


Figure 1

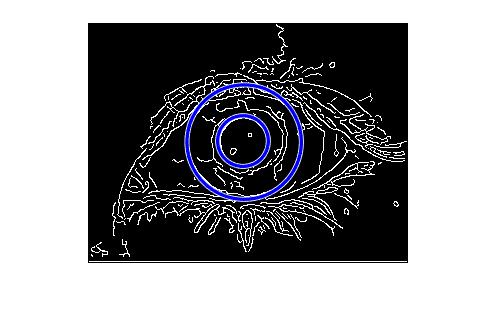


Figure 2

1. Normalization: In order to compare the iris, constant calculations have to be made, and computing the polar to cartesian conversion each time would be costly, so the iris needs to be preprocessed and then unwrapped. To unwrap the iris, Daugman’s rubber sheet model is used. This will convert the iris from a donut shape into a rectangular shape. Since every pixel in the iris has an equivalent position that can be found on the polar axes.
   1. Preprocessing
      1. Preprocessed the original iris image by applying a median filter to refine the features of the iris.
   2. Mapping
      1. The iris is mapped to polar coordinates using the equation in figure 3, which is visualized in the same figure, with the effect being that the iris is now unrolled (see figure 4), making the computation of the iris code much easier. The entire circle of the iris is unrolled, by first determining the offset of the pupil and iris center. After doing so, some cases are checked that would determine the sign of the equation in figure 3. Then using the x and y offsets, as well as all the angles in a circle converted to radians, and both the iris and pupil radius, the distance from the edge of the pupil to the edge of the iris is determined. Then a matrix is created using the distances, and the desired radius (which will basically be how much of the iris is mapped), which can then be used to map the polar coordinates of the unrolled iris to the corresponding Cartesian coordinates in the original image. Then using a meshgrid, and the interp2 function, we map the intensity values of the original coordinates to the polar coordinates in the unrolled iris matrix, so the iris code can then be calculated using a gabor filter. The final result can be seen in figure 5.

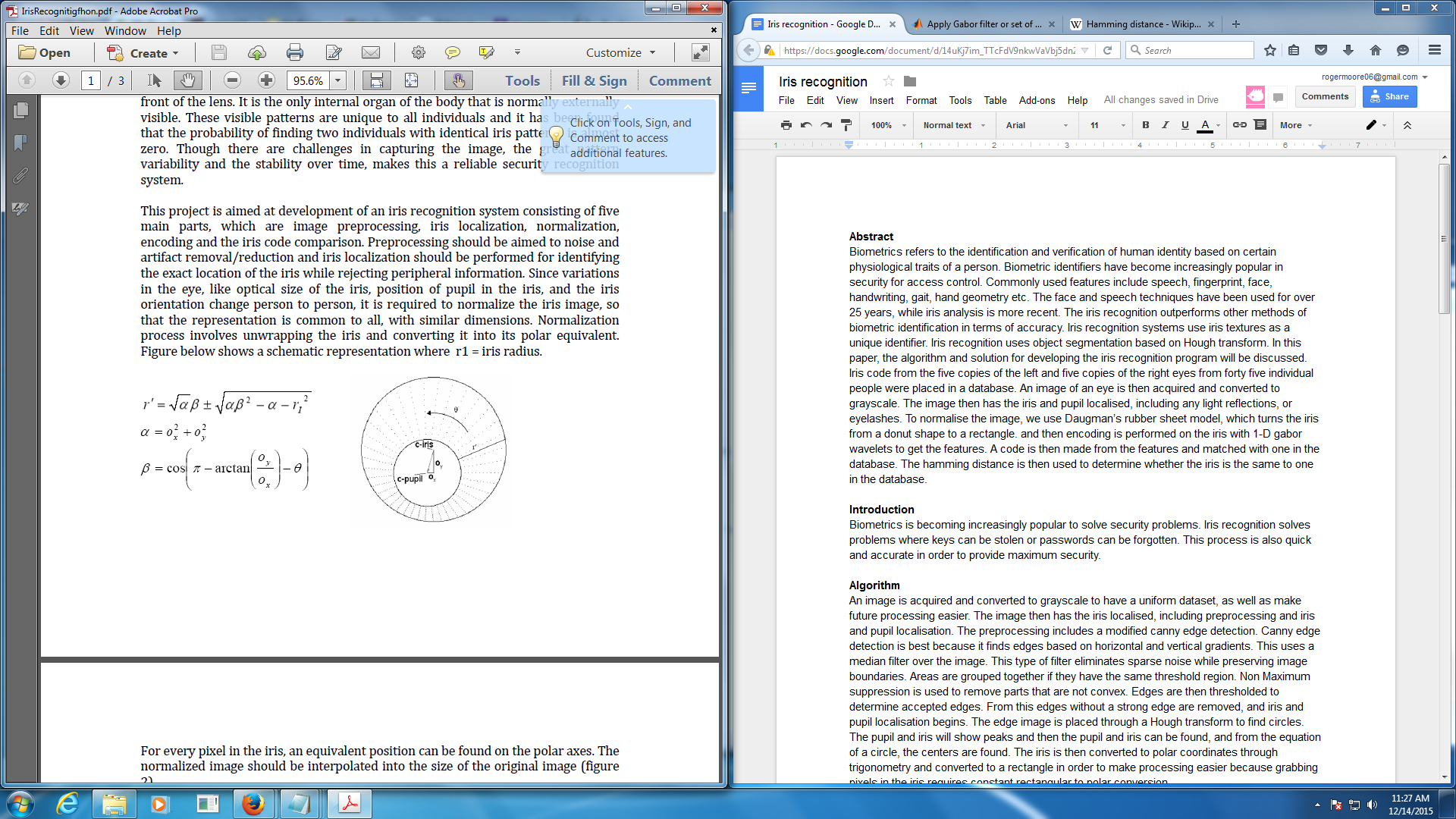


Figure 3

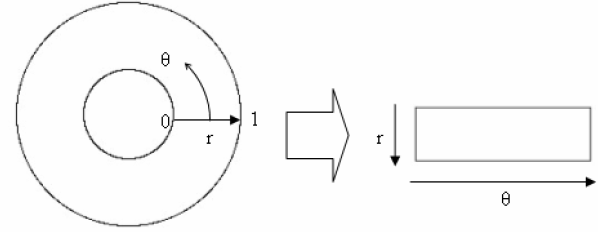


Figure 4



Figure 5

1. Encoding: In order to provide recognition of an individual, the iris pattern needs to be extracted. A code is then made by applying a Gabor filter on the iris to extract localised frequency information. The information we get back from the gabor filter is a magnitude and phase plot, from which we just use the phase plot for comparison. This is because the phase information in the pattern is assigned regardless of image contrast. The magnitude is not used since it depends on extraneous factors.
   1. Using the built in function imgaborfilt on the normalized matrix created in the Normalized step we get the features of the iris. This function gives two results, the magnitude and the phase. The phase is what we are interested in. The parameters entered besides the normalized matrix is the wavelength of 20 and the orientation of 90 degrees. After getting the phase the absolute value is taken of the phase to remove the negative results in the image. The next step is to normalize the phase by making any value below the threshold of 0.001, as that gabor wavelet is not significant enough, to zero and any value above that threshold to one. Figure 6 shows an image of how the iris code would look like.



Figure 6

1. Comparison: The iris code is then compared to the database, using hamming distance to determine error percentage. The hamming distance tells how different the two iris’ are from each other. It does this by summing all the bits in the iris code that are different. Statistically, the hamming distance from two patterns is expected to be 0.5, because two iris’ are independent and independence implies complete randomness, so there would be a 50% chance to set a bit to 1. This means two completely different iris’ would most likely have half matching and half not matching. If two patterns are the same, the distance is expected to be 0.0 since the two iris’ are close matches. The threshold for a match is predetermined and matched with one in the database. When one is a match, a notification is sent.
   1. The image is now compared to the iris codes in the database of iris codes to look for a match. The hamming distance is applied in this step to compare two iris codes. One of the iris codes goes through a series of shifting left and shifting right to account for any tilting. After being shifted for a certain number of bits, the shifted iris code and the second iris code have XOR applied to them to get the number of bits that they differ. Those bits in they are differing is divided by the total number of bits in the image to provide the error. The smallest error percentage is the one taken and if it’s below the threshold of 0.2 then it’s a match.

**Dataset, Experiments, and Results**

Dataset

The dataset consists of four hundred and fifty eyes, five left and five right eyes from forty five people provided by Professor Shah from the University of Houston. The iris’ are pigmented with eyelashes, with good contrast between the pupil and iris. We experiment on a few eyes during testing to work out problems. The result of this process is a match of an iris if an error percentage below the threshold is found.

Experiments and Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Image 001.bmp Matching Results | | | | |
| Image | 002.bmp | 003.bmp | 004.bmp | 005.bmp |
| Error | 0.0834 | 0.1067 | 0.0947 | 0.2167 |
| Result | Match | Match | Match | Intrusion |

Table 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Image 009.bmp Matching Results | | | | |
| Image | 002.bmp | 003.bmp | 004.bmp | 005.bmp |
| Error | 0.1929 | 0.1401 | 0.1369 | 0.1152 |
| Result | Match | Match | Match | Match |

Table 2

Using the algorithm described in the earlier section we encoded 90 different eyes from 45 different people into a database. We then compared the first and second image in the database to four similar images to see how well of a match we obtained. Table 1 and Table 2 show the results from doing these comparisons. One of the images failed to be recognized as a match as it was slightly above the threshold of 0.2. The reason that threshold was set was as we took two images and compared them to the entire database 0.2 was as low of an error match we could set. The reason being that for images that didn’t come from the same eye had an error between approximately 0.2 and 0.6. We also performed tests that compared one image to the entire database to find matches, but due to the amount of space such a table would take up in this document, we left out such results though they are still performed in the program.

**Conclusion**

Iris recognition is a good way to determine the identity of a person. Iris recognition is a quick and easy way to determine identity and have a low chance of error. Iris recognition has uses in several field that require security and access control. With advancements in security, iris recognition will become increasingly common, precise, and cheaper in the future. From our results we came to the conclusion that our results aren’t ideal to a live critical situation. The first reason is that localization utilizes the Hough Circle Transform which at times fails to recognize the iris or pupil edges and produces bad iris and pupil localization as shown in figure 7 below. Another issue as shown from our results is that sometimes we don’t make a match where we know one should occur, due to our code not accounting for any scaling and because it expects the images brought in to be have very similar environments.

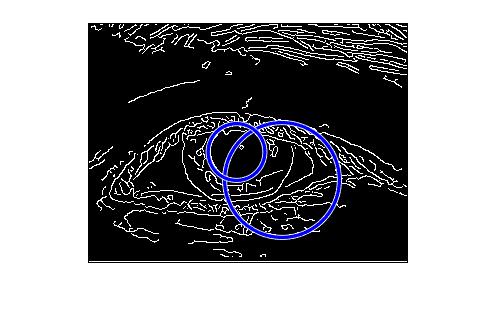


Figure 7

**Group contribution**

Mr. Cupit: program - acquire image, iris localisation, normalization- 25%

Mr. Silva: program - features encoding, features matching, iris database - 25%

Mr. Moore: paper - abstract, introduction, algorithm, dataset, experiment, and results, conclusion - 25%

Mr. Khondker: paper - abstract, introduction, algorithm, dataset, experiment, and results, conclusion - 25%