UNIVERSITY OF SOUTHAMPTON

Faculty of Physical Sciences and Engineering

Department of Electronics and Computer Science

An Iris Recognition System

A project report submitted for the award of MEng Computer Science

Author: Harry Talbot

Project Supervisor: Dr. Sasan Mahmoodi

Second Examiner: Professor George Chen

Academic Year 2017-18

Word Count: 9904

Abstract

Using biometrics for identification is an expanding field with a variety of implementations and applications. Their ease and speed of use over traditional verification methods sees them becoming more and more dominant in day to day life, such as the use of the human iris as a biometric identifier in unlocking smartphones and laptops. Many approaches have been presented for finding an iris in an image, encoding the texture information and performing fast comparisons with the encoding among huge databases, while providing accurate matching as not to compromise any security applications.

This report contains a short literature review of a range of existing methods, followed by implementation of several approaches for each stage of the recognition system - two methods for iris localisation, one for normalisation, and four for feature extraction and matching. The implemented extraction and matching approaches are compared, looking at their speed, accuracy, false accept and reject rates, ease of implementation and efficiency, when matching images from the CASIA iris database. A database structure and a graphical user interface that displays the stages of iris image processing and matching is designed. Implementation issues, localisation, encoding and matching results are discussed, and an iris recognition system that links the most successful approaches from each stage through the graphical user interface is presented with an identification accuracy of 95.74%.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my supervisor Dr. Sasan Mahmoodi for his help and guidance during this project, and for stimulating my interest in the field of biometric image processing.

I would also like to thank my close friends and family for their unending support and encouragement during the project - particularly their understanding and patience whenever I often brought the subject up in conversation.

Finally, I would like to acknowledge the Centre for Biometrics and Security Research for their development of the CASIA iris data sets, which are publicly available, free of charge and used gratefully in this project.

1 Introduction

Topic Overview

Biometric systems for human identification have proven to be just as secure as traditional password orientated methods, and while more complex they are becoming more practical. Many biometric identifiers exist, such as fingerprints, facial structure, voice, gait and even signature. In security applications, physiological biometrics are preferred over behavioural biometrics - for example, while widely used for the signing of cheques, a signature is a poor biometric as in old age or ill health writing may become more challenging (along with a signature being very easy to reproduce).

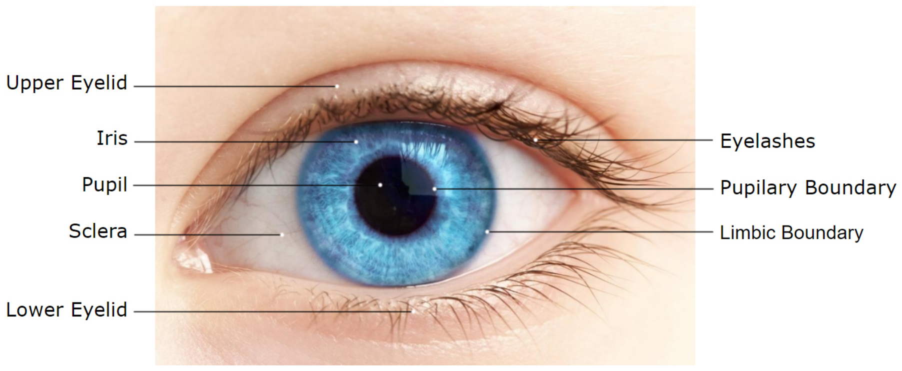


Figure 1: Features of the eye involved in iris recognition.

Of the physiological identifiers, the human iris is one of the most attractive for use. The iris is protected from damage by the cornea [1], unique due to the high level of entropy in its formation [2], and shows relatively little change over a person's lifetime once developed [1]. Its distinct and easily identifiable placement on a person makes it quick to locate and therefore advantageous for fast analysis and identification.

There are several challenges around the use of biometrics over traditional verification methods. Once lost or stolen, a biometric such as a scan of an iris cannot (within reasonable bounds) be changed as easily as a password could be. The use of a fingerprint to unlock your car may put you in more danger in the event of a theft than you would have been had you just been able to hand over your car keys. Because of this, biometrics are not always a perfect solution.

Problem Description

This project aims to develop a piece of software that can identify people by their iris. The system should locate the iris in an image of the eye, encode the iris information and then correctly select matching iris' from a database of at least 150 images. The system should operate with an identification accuracy of at least 95%. In literature many methods to locate and encode the texture information of the iris are presented with varying rates of success, and so several must be implemented and analysed to determine which has the better accuracy. For the simplicity, to simulate the presence of a user a leave-one-out strategy will be used to test the system, taking an eye image from the data set and using this as if it had been captured by a camera in a real world application.

2 Literature Review & Background Reading

For successful iris recognition, the input image requires an iris size of around 70 pixels; a size of 80-130 is generally more useful [1] but beyond that using a larger image increases computational time and encodes more information than is needed. Near-infrared images are generally preferred over the use of visible light, as corneal reflections are not captured in them [3].

After the iris image has initially been captured, the iris recognition process generally comprises of three steps; localisation, normalisation, and finally feature extraction and matching. Various methods have been presented to implement these steps, and several may use similar implementations for one step but then a different method for the next.

2.1 Localisation

The first phase of the iris localisation aims to find the pupilary and limbic borders. Accurate localisation is crucial to the iris matching; over-fitting the iris will encode parts of the eyelid and other eyelash noise as iris information, and under-fitting will mean that a subsection of the iris is distorted when normalised to the size of the entire iris. In the worst case the mask is aligned so that parts are over and under fit simultaneously.

Daugman [1] uses an Integro-Differential Operator (IDO) to locate the iris boundaries. From a set of possible centre points, the operator searches outwardly for the maximum change in average circumferential pixel value. The change is greatest as the operator moves across a border from either dark pupil pixels to grey iris pixels or from grey iris pixels into white sclera pixels, and does so in all directions at the same time. This point of maximum change identifies the correct centre point and radius. This technique is also used in [4]. [5] and [6] note that the IDO is susceptible to specular light reflections.

Wildes [7] uses a classical computer vision technique, the Hough transform, in a circular arrangement to vote on iris and pupil centre points. Edge detection generates an edge map, and using the edge points circles are drawn which vote in the Hough parameter space. Maxima in the parameter space correspond to circles identified in the image. This technique is also used in [8] and [9].

Tisse et al. [5] use a combination of the methods discussed above; edge detection and a circular Hough transform approximates the position of the eye in the global image, and then Daugman's operator is used to find a more precise pupil and iris boundary.

Noting that the contrast across the pupilary boundary is much greater than across the iris boundary, and the pupil itself is significantly darker than the rest of the eye, Zhu, Tan and Wang [10] find the pupilary boundary via thresholding, a fast and simple technique. They then locate the outer boundary by "maximizing changes of the perimeter-normalized sum of gray level values along the circle", in a similar way to [1].

2.1.1 Eyelid Removal

While in an ideal system the captured iris image would be free from occlusion by the eyelid, this is rarely the case. Images used in [1] were deemed inadequate if less than 50% of the iris is visible. In [11] and [5], the iris was cropped from the top down where upper eyelid occlusion was common, regardless of its presence in the image. In Daugman's later papers [1] and [12], eyelids (if present) are located using his IDO, integrating around an arcuate path rather than circular.

In [10], only the innermost part of the iris is used for matching to avoid eyelid occlusion. Similarly, Ma, Wang and Tan [13] found that the top 75% of a normalised iris image is the most useful for identification, due to its features and the fact that it is usually untouched by eyelid occlusion.

Masek [14] uses a linear Hough transform to fit two horizontal lines to the upper and lower eyelids. Canny edge detection is implemented, and edge information in the horizontal direction is used. As the iris and pupil are located first, the edges used as a basis for the transform are constrained to only those that lie within the iris area. This is refined in Wildes et al. [8], who use a horizontally orientated edge detector followed by an elliptical Hough transform, so to model the eyelids as two parabolic arcs that lie within the iris region.

Dehkordi and Abu-Bakar [9] use fuzzy filtering to remove noise such as eyelashes. Thresholding first identifies eyelashes in the image, and an adaptive window is placed over each eyelash pixel. The window expands based on how many other noise pixels are inside it and the density of the noise pixels inside the window determines whether the central eyelash pixel is filtered out or preserved.

Podder et al. [15] uses oriented non-maximum suppression to remove eyelashes and eyelid noise. After locating the iris, the edge points from Canny edge detection are used along with their direction from the centre point. For each edge point P, the points $X1$ and $X2$ either side of it along the radial direction are found, and if P is a maximum it is marked as an edge to keep in the new edge image.

2.2 Normalisation

In most cases, the size of the iris between images will vary depending on pupil dilation, as well as the angle and distance between the eye and the camera differing slightly. Untreated this will affect the accuracy of the iris matching process, and so is corrected through a normalisation process.

Daugman [1] uses linear mapping to transform the circular iris into a space of θ and r, a rectangular block of specific dimensions, often referred to as Daugman's 'rubber sheet' model. This accounts for stretches in the iris caused by pupil dilation, and the fact that the pupil and iris rarely share the same centre. This normalisation process is also used in [5], [9], [10], [13] and [4].

Wildes et al. [8] uses a technique to warp an acquired image $I_a(x, y)$ onto a database image $I_d(x, y)$ according to a mapping function $(u(x, y), v(x, y))$ so that the pixel value at $I_a(x, y) - (u(x, y), v(x, y))$ is as similar as possible to that at $I_d(x, y)$. [7] notes that accounting for pupil dilation is not a critical issue, as the constant lighting of the image acquisition system should (for healthy eyes at least) affect subjects in the same way, and cause the same level of pupil dilation throughout.

Boles and Boashash [2] describe normalisation requiring two images. In this case, the image with the greater iris diameter is considered a reference image. "Virtual Circles" are generated on the non-reference image using the ratio of iris diameter between the two images. Information is then extracted from the smaller iris and scaled appropriately to the size of the reference image iris.

2.3 Feature Extraction and Matching

2D Gabor wavelets are used by Daugman [1] and Dehkordi and Abu-Bakar [9] to encode the iris information. Areas of the iris are projected onto the wavelets, and the resulting phase quantisation of the area determines bits in the iris code. The iris code generated is 256 bytes long - small enough to fit onto a magnetic strip on a credit card [11] - and a corresponding mask is generated, indicating which parts of the code may have been generated from eyelashes, eyelids, areas of specular reflection or just image noise. The codes are matched by their Hamming distance, a measure of dissimilarity, which by successively bit-shifting the iris code also allows for rotational differences in the two images being compared. Zhu, Tan and Wang [10] apply histogram equalisation to the normalised iris images, and similarly use Gabor filtering and a 2D wavelet transform - however, for simplicity they use the weighted Euclidian distance to match iris codes.

Ibrahim [6] and Tze Weng et al. [16] use a Haar wavelet transform to perform iris feature extraction. In [6], high and low pass filters are passed over the iris image horizontally and vertically, and the resulting information is used to find detail coefficients in horizontal, vertical and diagonal directions. An approximation image is also produced, and used again as the input for another level of Haar filtering - all together four levels are generated and pieced together into one iris image. The average absolute deviation of the image is used for matching using a measure of the Euclidian distance. Tze Weng et al. [16] also use Haar wavelets, taking only coefficients from the the fourth level of filtering. The signs of the coefficients generate an iris code, where positive coefficients are set to ones and negative coefficients set to zeros. Codes are matched similarly to [1], using the Hamming distance, although no mask bits are used. Haar wavelets are also used in [17].

Wildes [7] discusses how distinguishing features of the iris are visible at a variety of scales, and as such a multiscale representation of the iris could capture this information. Noting that the encoding can be stored efficiently by subsampling the lower frequency bands, a Laplacian of Gaussian filters is used to encode the features of the iris. The Laplacian is defined procedurally in a cascading manner. Compared to [1] this representation is not as compact, but conversely retains more iris information. To match the representation, normalised correlation is used.

Ko et al. [4] use grey level analysis. The normalised iris is split into regions, and the average value of a region is taken as a representative. These regions are grouped in fives vertically and horizontally, and for each group the cumulative sum of representative values is generated. The maximum and minimum cumulative sum values are used to determine whether the region intensity increases or decrease, and to set bits in the iris code. A vertical and horizontal code is generated and matched by Hamming distance. This technique is faster than those in [1], [2] and [13] as the processing only involves addition and subtraction.

Circular symmetric filters (CSF), similar to Gabor filters, are used for feature extraction by Ma, Wang and Tan [13]. The filters extract local iris features from the top 75% of a normalised iris image and use this to form global feature vectors. This region of interest (ROI) gets divided into three vertical sections, which are locally filtered using CSF. Feature extraction is performed in 8x8 pixel sections, generating 384 values (the values are the absolute deviation of the 8 x 8 block), which make up a one-dimensional feature vector with 384 coefficients. The nearest feature line distance is used for classification and matching.

Whilst not used specifically in iris systems, the use of local orientation histograms has been used in other biometric applications for hand gesture recognition in [18], [19] and to recognise sign language in [20]. This involves a patch wise generation of local histograms representing image edge gradient, either at various interest points in the image, or in a dense grid uniformly across the image. Once generated, corresponding patches between images are compared and a quality of match value is generated.

Boashash and Boles [2] use one-dimensional wavelet signals to represent the iris and obtain the zero crossing of these signals for iris comparison. Low resolution levels are used so the representation is robust in a noisy environment and the computational complexity is reduced. The one-dimensional nature of the transform only captures data along the circle, excluding information that may come from patterns in the iris along the radius.

3 Implementation

3.1 Localisation

Two methods of iris and pupil localisation were identified in the literature review - an Integro-Differential Operator (IDO), used and developed specifically for this application by Daugman [1], and the circular Hough transform, used by Ma, Wang and Tang [13] and Wildes [7], a classical computer vision technique.

3.1.1 Image Preprocessing

Before the images are used they need to be processed to ensure that only necessary information is kept, and that maximum accuracy is maintained in the image itself during the stages of iris localisation and manipulation. The images are converted to be of type double, so that all the values of the pixels are between 0 (black) and 1 (white). As the images from the dataset are grey level, the additional colour channels of the image created using MATLAB's imread is disregarded.

Figure 2: Application of the imfill operation to S1033L01.

(a) Gaussian Blurring ($\sigma = 3$) before Hough localisation. (b) Histogram equalisation before IDO localisation.

Figure 3: Two preprocessing techniques applied to S1033L01.

Due to the lighting arrangement used in the image acquisition stages, an array of white spots exist in the pupilary area of the images. These are first removed using MATLAB's imcomplement and imfill functions (Fig. 2). The images are also blurred with a Gaussian filter to remove edges created from eyelashes or the skin around the eye, in order to aid the later localisation. Before the Hough transform a σ of three is used (Fig. 3a), whereas before the IDO is applied a σ of two is chosen, resulting in less of a blurring effect. In the Hough Transform, a stronger blur was needed to remove edge points detected from eyelashes, while the IDO relies on a strong contrast across the boundary to generate peaks in the differential, and therefore should not be blurred as much. Histogram Equalisation, a technique to flatten an image's histogram, is also applied to an image before it is passed to the IDO, increasing the contrast across the limbic boundary (Fig. 3b).

3.1.2 Daugman's Integro-Differential Operator

Daugman [1] presents an operator that is now one of the most widely used in commercial applications in Eq. (1):

$$\max_{(r,x_0,y_0)} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{r,x_0,y_0} \frac{I(x,y)}{2\pi r} ds \right| \tag{1}$$

where $G_\sigma(r)$ is a Gaussian blur $\frac{1}{\sqrt{2\pi}}e^{-\frac{r^2}{2}}$ of scale σ. The algorithm searches a circular path of radius r around a potential iris centre, looking for a maximum change in pixel value across the border of the circular path. At every radius r the normalised intensity of the circumferential pixels is compared to that in adjacent radii (one pixel apart), and at the end of the search the centre point of the iris is given by where the greatest change in pixel values across the circlular border was found. Once the iris boundary has been found, the pupil boundary can be found with the range of radius bound by the iris radius and possible centre points bound to the 10 x 10 pixel area around the determined iris centre point.

Finding Potential Centre Points

The IDO first needs a list of approximate pupil centre points to reduce the search space - otherwise in an image of 320 x 240 pixels (such as those used in the dataset), the operator would be applied to 768,000 centre points - the majority of which are not even remotely close to the centre of the iris, generating an unnecessary computational overhead. The function findmins achieves this using basic image processing techniques and restricting parameters that apply to all images in the dataset.

As the pupil is the darkest part of the image, and the image is grey-level - made up of values from zero to one - the function then excludes all the pixels with a value greater than 0.5 as these are deemed too light to be a pupil or iris centre point. Areas of camera light reflection will have been filled in with values less than 0.5, and so if the centre is covered by light reflection it will not be ruled out of the search. The list of possible centre points is then reduced again by removing any points that are too close to the edge of the image based on the minimum size of an iris, and then reduced further by removing any points that are not local minima compared. As eyelashes are typically very dark, they can be accepted as centre points, so once more the list is reduced by taking the median co-ordinate position of the points, and removing any points that have either a row or column position that lies outside a 20 x 20 pixel area centred on the median. This method accurately narrows down the original 768,000 points to approximately four hundred possible centre points, highlighted red in Fig. 4.

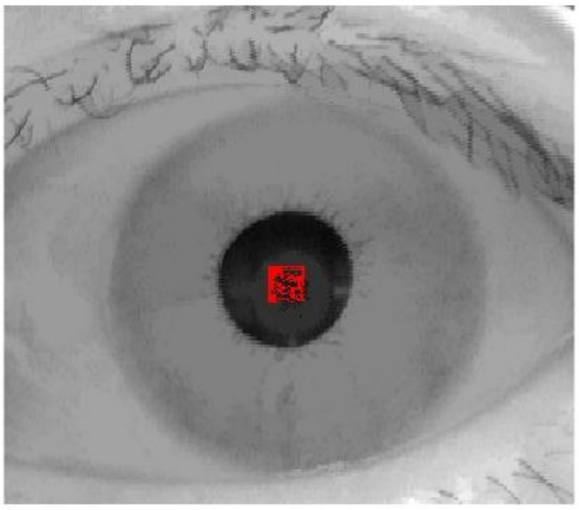


Figure 4: 265 possible centres identified in S1025L01.

Using the Operator

The centre points are passed onto an implementation of the IDO. For every possible centre point the function generates masks corresponding to circles of increasing radius around said points - where the masks have a value of one on the circle border and zero elsewhere. The masks are the same size as the eye image, so multiplication of the mask and eye image returns an image where any pixels not on the circumference of the circle are set to zero. The number of pixels kept by the mask and the total circumferential value are found using the MATLAB function sum; together these are used to find the average circumferential pixel value.

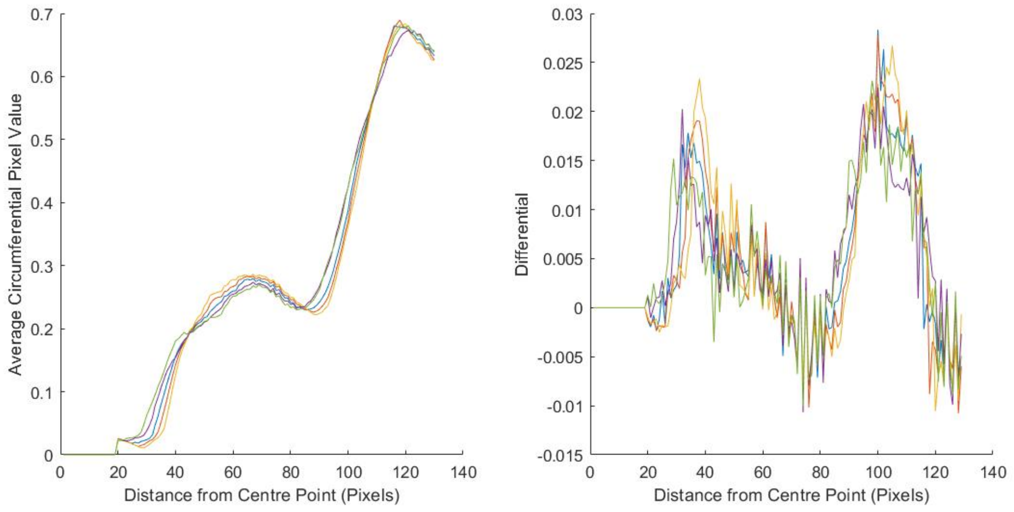


Figure 5: The IDO run on S1025L01 with five predetermined centre points.

Fig. 5 shows the average circumferential pixel value at a given radius and the corresponding differential for five points known to be close to the actual centre. The implementation runs in about 16 seconds and generally locates the pupil boundary very accurately; it sometimes struggles to identify the iris boundary, as in many of the images there is less contrast over the limbic boundary than the pupilary boundary.

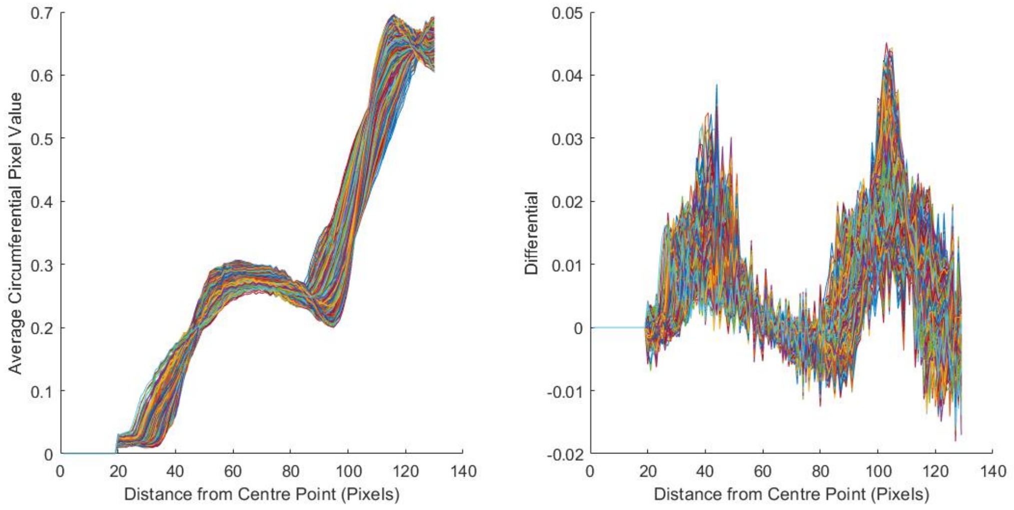


Figure 6: The IDO run in one pass with 265 points located by findmins.

Fig. 6 shows this for all points located by findmins. Fig. 7 shows the iris successfully located, using the peaks of the differential in Fig. 6 and the radius and centre points for those peaks. In the final implementation, rather than running the operator once with a single range of radius before establishing a threshold in the IDO data to search above and below for the two peaks, the operator is run twice with a pupil radius range of 20 to 60 pixels and an iris radius range of 100 to 130 pixels. These ranges are dependent on the input images but work effectively with the CASIA iris set.

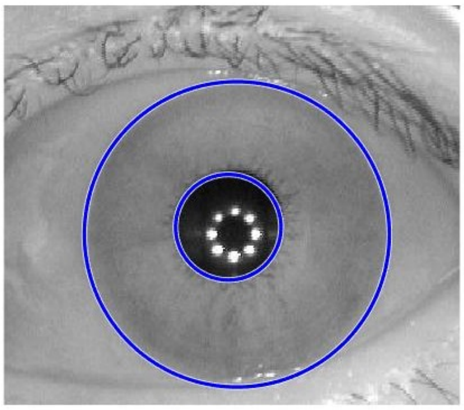


Figure 7: Correct localisation of the pupilary and limbic boundry with the IDO in S1025L01.

3.1.3 The Circular Hough Transform

Ma, Wang and Tang [13] and Wildes [7] both describe a different approach to iris localisation, making use of classic edge detection techniques and an adapted Hough transform. First, Canny edge detection [21] finds accurate edge locations in the image through applying a Gaussian filter, a Sobel operator [22], non-maximum suppression (thinning the identified edges to only indicate the sharpest change of edge intensity value), thresholding (identifying strong and weak edges) and hysteresis (removing identified weak edge points that are not directly neighbours of strong edge points). Gradient orientated edge detectors are used; for example, for identifying the eyelids, the edge detector is tuned for locating horizontal edges, and for identifying the limbic boundary the edge detector is tuned vertically.

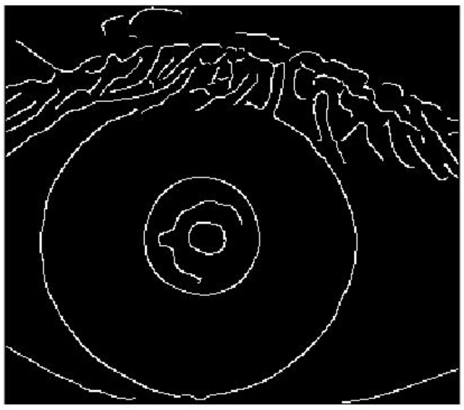


Figure 8: Canny edge detection applied to S1079R07.

The circular Hough transform takes the equation of a circle with three parameters in Eq. (2) and changes this to a parametric representation in Eq. (3):

$$(x-a)^2 + (y-b)^2 = r^2 \tag{2}$$

$$\begin{aligned} x &= a + r\cos(\theta) \\ y &= b + r\sin(\theta) \end{aligned} \tag{3}$$

where a and b are co-ordinates of edge points in the image.

Using all the edge points in the image as circle centre points, with increasing radius the Hough transform draws circles that vote on locations in the parameter space. Afterwards, local maxima in the voting space identify parameters of circles in the original image. A similar approach is seen in [9].

As three parameters need to be found (x and y position, and circle radius) a three-dimensional accumulator is used of size *height* x *width* x *maximumRadius*, where each element holds a number of votes for the parameter tuple it represents. The maximum in the accumulator generates a peak corresponding to the best circle location and radius, shown in Fig. 9 as a 'hot spot' and in Fig. 10 as a vertical spike.

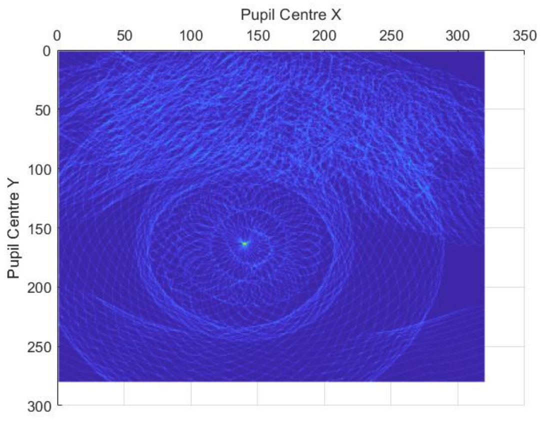


Figure 9: Two-dimensional view of the Hough accumulator space at $r = 41$ for S1079R07.

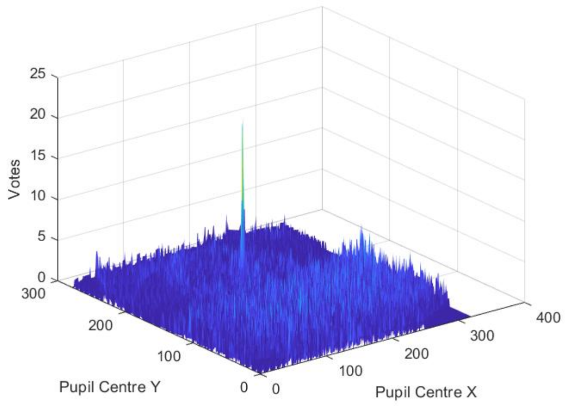


Figure 10: Three-dimensional view of the Hough accumulator space at $r = 41$ for S1079R07.

3.1.4 Eyelid Localisation

A simple way to locate and remove eyelids is to exclude the area they commonly occlude regardless of whether they are present or not. Once localised, the iris is cropped by two horizontal lines that lie a fixed number of pixels - in this case twenty - inside the iris boundary. Cropping similar to this is used in [11], where a region at the top and a 45° notch at the bottom is ignored. A Hough transform can be used to fit a straight line or an arc to the eyelids, as used in [14] and [8], but this was not implemented.

3.2 Normalisation

The 'Rubber Sheet' Model

In order to use the located iris region the texture information must be normalised into a polar array. This must account for the fact that the pupil and iris in an image are rarely concentric, and pupil dilation can affect the thickness of the iris. To do this a normalisation function is used, which implements Daugman's Rubber sheet model.

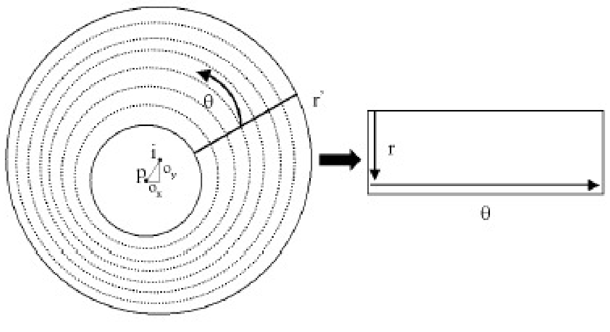


Figure 11: Daugman's Rubber Sheet Normalisation (Reproduced from Kazakov [23], Chapter 5, Page 20).

The model converts the Cartesian co-ordinates of the iris image into a 'doubly-dimensionless' [1] arrangement of r and θ. This provides an exact mapping for all the pixels in the area specified as the iris. The remapping can be represented as:

$$\begin{aligned} x(r,\theta) &= (1-r)x_p(\theta) + rx_l(\theta) \\ y(r,\theta) &= (1-r)y_p(\theta) + ry_l(\theta) \end{aligned} \tag{4}$$

where $(x_p(\theta), y_p(\theta))$ and $(x_l(\theta), y_l(\theta))$ are the pupilary and limbic boundaries in the direction of θ, where also θ is an angle between $[0, 2\pi]$ and r an interval between $[0, 1]$.

Masek [14] and Shamsi et al. [24] both use a formula to scale points on the iris based on the iris and pupil centre point displacement and the point's angle from the centre in Eq. (5), using Eq. (6):

$$r' = \sqrt{\alpha}\beta \pm \sqrt{\alpha\beta^2 - \alpha - r_I{}^2} \tag{5}$$

$$\begin{aligned} \alpha &= o_x{}^2 + o_y{}^2 \\ \beta &= \cos\pi - \arctan\left(\frac{o_y}{o_x}\right) - \theta \end{aligned} \tag{6}$$

where r' is the distance between the edge of the pupil and the edge of the iris at angle θ, o_y and o_x are the relative displacement of the two centre points in the y and x direction, and r_I is the radius of the iris. The implementation takes arguments of radial and angular resolution r and θ and first calculates the displacement between the pupil and iris centre point in the y and x direction. [23] notes that in the event there is no displacement between the pupil and iris centre, $r' = \pm\sqrt{-r_I^2}$, an imaginary number that will not satisfy the normalisation. Therefore in the case of no displacement, $\beta = \cos\pi - (\pi/2) - \theta$ is used.

Iterating the angle around the pupil, r' is calculated, and a one-dimensional vector of values that range from the pupil radius to the iris radius in r steps is generated, $radSteps$. Using these steps and a polar to Cartesian transform, two vectors containing x and y positions are created, $colValues$ and $rowValues$, such that $colValues[i]$ and $rowValues[i]$ together identify a Cartesian co-ordinate for $(radSteps[i], theta)$. At the end of the iteration $colValues$ and $rowValues$ are copied to the $theta$ column of two-dimensional arrays $allCol$ and $allRow$. This is repeated for all $theta$, and once row and column values have been generated for all $theta$, interpolation is used on the three-dimensional arrays to build the normalised image (Figure 12).

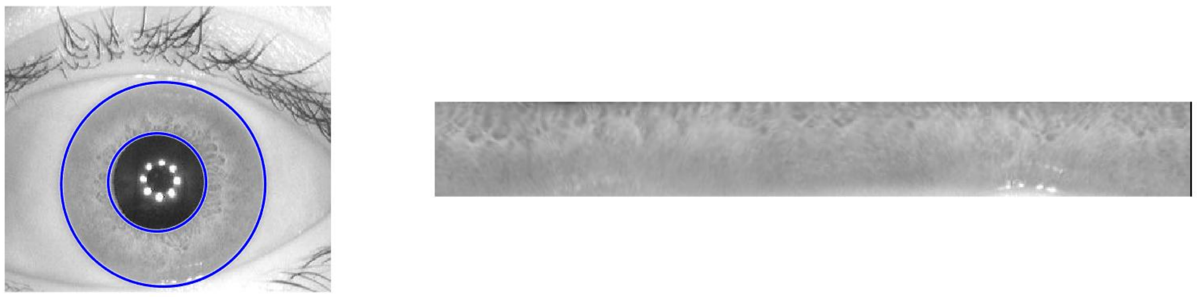


Figure 12: Normalisation of S1199L09.jpg.

A shortcoming of the rubber sheet model is that in the event of occlusion, parts of the image that cover the iris are interpreted as the iris itself and included in the normalised image, shown in Fig. 13.

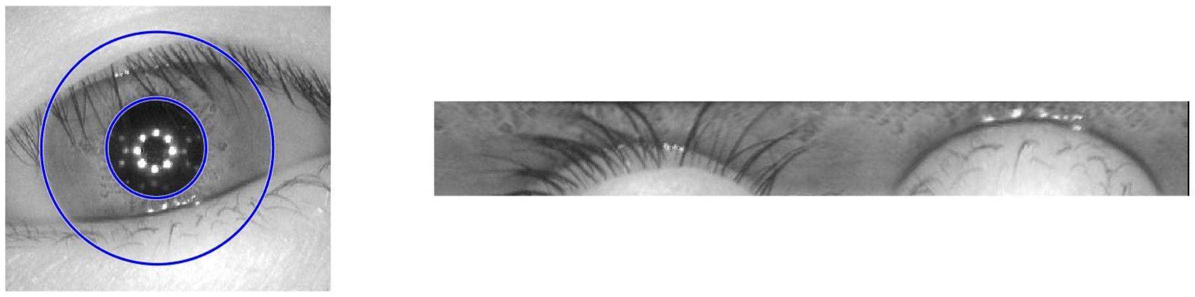


Figure 13: Normalisation of S1144L01.jpg effected by included eyelids.

To account for this a corresponding mask image is generated. It indicates parts of the normalised image that may have been drawn from outside the original image (in cases where the circle for the iris border overlaps the edge of the image) or parts that may have been occluded. Fig. 14 shows an iris that has been localised and cropped to remove eyelids, with a mask where ones indicate where these eyelids would be in the normalised image.

Figure 14: Normalisation of S1019L01.jpg, with a bit mask to indicate bad parts of the normalised image.

3.3 Feature Extraction & Matching

3.3.1 2D Gabor Wavelets

Daugman [1] encodes the extracted iris information using 2D Gabor wavelets, shown in Eq. (7):

$$h_{\{\mathrm{Re,Im}\}} = \mathrm{sgn}_{\{Re,Im\}} \int_{\rho}\int_{\phi} I(\rho,\phi)e^{-i\omega(\theta_0-\phi)} \cdot e^{-(r_0-\rho)^2/\alpha^2} e^{-(\theta_0-\phi)^2/\beta^2} \rho d\rho d\phi \tag{7}$$

where $h_{\{\mathrm{Re,Im}\}}$ is a complex-valued bit; $I(\rho,\phi)$ is the raw iris image; α and β are the multiscale two dimensional wavelet size parameters; ω is wavelet frequency; and (r_0,θ_0) represent the polar coordinates of each region of iris for which the phasor coordinates $h_{\{\mathrm{Re,Im}\}}$ are computed. The real and imaginary parts of the resulting convolution are used to set bits in the iris code as follows:

$$\begin{aligned}
h_{Re} &= 1\,\text{if Re}\int_{\rho}\int_{\phi} I(\rho,\phi)e^{-i\omega(\theta_0-\phi)} \cdot e^{-(r_0-\rho)^2/\alpha^2} e^{-(\theta_0-\phi)^2/\beta^2} \rho d\rho d\phi \geq 0\\
h_{Re} &= 0\,\text{if Re}\int_{\rho}\int_{\phi} I(\rho,\phi)e^{-i\omega(\theta_0-\phi)} \cdot e^{-(r_0-\rho)^2/\alpha^2} e^{-(\theta_0-\phi)^2/\beta^2} \rho d\rho d\phi < 0\\
h_{Im} &= 1\,\text{if Im}\int_{\rho}\int_{\phi} I(\rho,\phi)e^{-i\omega(\theta_0-\phi)} \cdot e^{-(r_0-\rho)^2/\alpha^2} e^{-(\theta_0-\phi)^2/\beta^2} \rho d\rho d\phi \geq 0\\
h_{Im} &= 0\,\text{if Im}\int_{\rho}\int_{\phi} I(\rho,\phi)e^{-i\omega(\theta_0-\phi)} \cdot e^{-(r_0-\rho)^2/\alpha^2} e^{-(\theta_0-\phi)^2/\beta^2} \rho d\rho d\phi < 0
\end{aligned} \tag{8}$$

MATLAB's inbuilt function gabor allows creation of a bank of gabor filters with a given wavelength (measured in cycles per pixels) and orientation. This can then be convolved with the image to return magnitude and phase information. The magnitude information is discarded, because unlike phase information it can be affected by image contrast, illumination and camera gain [1]. Bits are set in the iris code depending on the phase angle (Fig. 15).

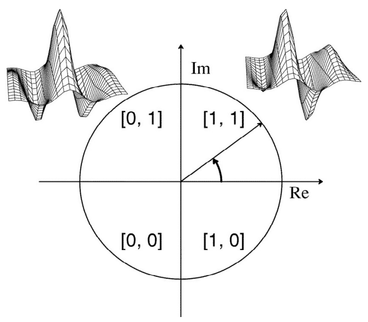


Figure 15: The phase demodulation process for building Gabor iris codes (reproduced from Daugman [1], Page 22).

For an m x n iris image, an m x $2n$ size iris code is returned. The number of bits in the iris code is directly linked to the size of the normalised iris image. [1] uses 2048 bits in the iris code; this generated from 1024 pixels in the normalised iris image restricts the normalised image dimensions.

Matching the Iris Codes

The Hamming distance between two iris codes is used to check for a match using a certain tolerance of variation, given in Eq. (9):

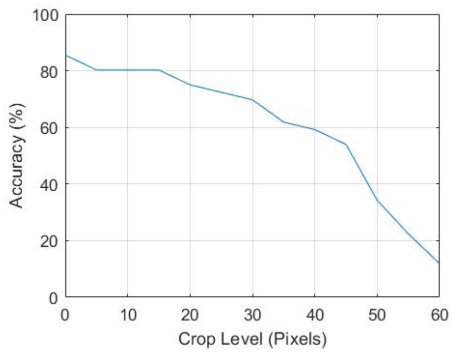
$$\mathrm{HD} = \frac{\|(codeA \oplus codeB) \bigcap maskA \bigcap maskB\|}{\|maskA \bigcap maskB\|} \tag{9}$$

where *codeA* and *codeB* are the iris codes and *maskA* and *maskB* are their corresponding masks to exclude non-iris information. The low computational requirement of the bit comparisons means exhaustive searches of large databases can be completed quickly. A Hamming distance of zero indicates that the two images are identical, and one that they are opposite. A distance of 0.5 is expected for uncorrelated eyes [1].

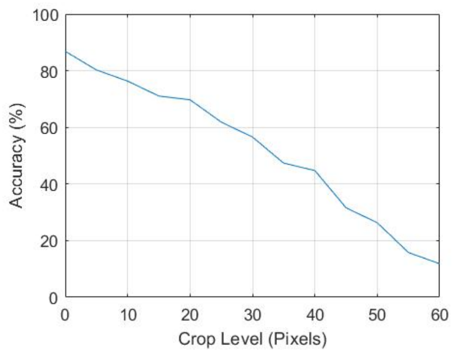
The comparison is performed several times with different versions of the same code. To account for radial rotation in the original image the iris patterns must be shifted during comparisons and the lowest Hamming distance taken. A shift of 16 pixels (corresponding to 8 pixels in the normalised image) in steps of two was used to the left and right with a 32 x 512 pixel code, catering to roughly $\pm3^\circ$ of rotation in the original eye image. A local function is used to perform the shifting. The two masks are combined using a bitwise OR to generate the combined mask, and from the combined mask the total number of bits that will be compared is obtained and stored. The combined mask is then inverted, so that zeros indicate the parts to keep, and then the two codes are OR'd together before being combined via a bitwise AND with the combined mask. The summation of this result will give the number of different bits; this is then normalised by dividing by the total number of compared bits to give the Hamming distance.

Accounting for Eyelid Occlusion

Introducing noise into the eye image via eyelid occlusion has a direct effect on the success of matching rates, and so cropping is used to exclude commonly occluded areas. A set of eyes predominantly with full, circular iris visible were selected, and one by one matched with others in the set without the use of masking bits. The subject image being matched was covered slightly with random noise in order to simulate a level of upper eyelid occlusion. The effect of varying this level and therefore the effect of an eyelid is shown in Fig. 16. Using cropping to generate corresponding mask bits to ignore this eyelid effect in the matching phase steadied the matching rate, shown in Fig. 17.

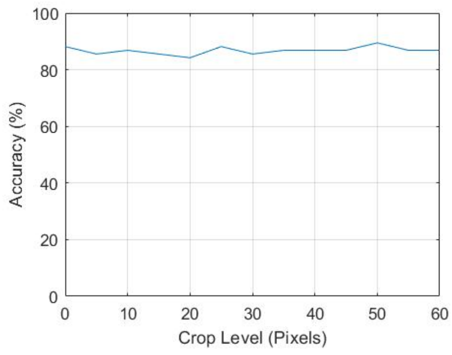


(a) Upper Cropping

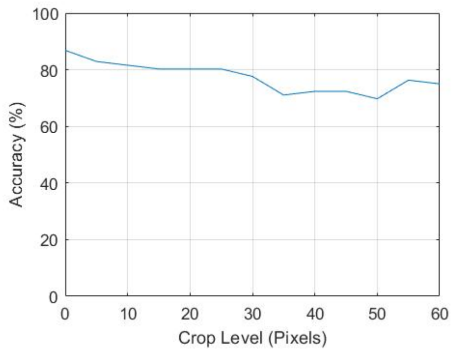


(b) Lower Cropping

Figure 16: Match rates with upper and lower occlusion.



(a) Upper Cropping



(b) Lower Cropping

Figure 17: Iris match rates with upper and lower occlusion using Gabor wavelets and mask bits.

3.3.2 Cumulative-Sum Analysis

Feature extraction is performed using the cumulative sums of fixed size regions in Ko et al. [4]. A rectangular normalised iris of size 30 x 250 pixels is divided into consecutive regions of 3 x 10 pixels. These regions will later be grouped vertically and horizontally in sets of five, so the initial normalised image size is chosen so that the dividing and grouping can be performed exactly covering the original image. A representative value is taken for each region as the mean value of the region. Grouping first horizontally, the first five regions are taken along the row. The group average is determined from the representative values. The cumulative sum is calculated for the group as in Eq. (10):

$$\begin{aligned} S_0 &= 0 \\ S_i &= S_{i-1} + (X_i - \overline{X}) \quad \text{for i } = 1, 2, ..., 5. \end{aligned} \tag{10}$$

where X_i is the representative value of region i and $\overline{X}$ is the average representative value of the group. after discarding S_0 these cumulative-sum values CS are used to set the iris code. The code section is initialised as a one-dimensional row vector containing five zeros. The index of the minimum and maximum cumulative-sum values I_{min} and I_{max} are found. If the index of the maximum value is greater than the index of the minimum value, the cumulative-sum values from I_{min} to I_{max} inclusive are deemed to form an upward slope, where the iris pattern changes from dark to light, and so their corresponding code bits are set to one. Conversely, if I_{min} is greater than I_{max}, the cumulative-sum slope is downward and the code bits are set to two. Values outside the range of I_{min} to I_{max} are left as zero. Code generation is illustrated in Fig. 18. This process is repeated grouping vertically so that two codes of size 10 x 25 are formed, one for either direction.

CS	I_{min}	I_{max}	Code
-20 4 8 15 1	1	4	1 1 1 1 0
12 44 15 -2 2	4	2	0 2 2 2 0

Figure 18: Cumulative-Sum Code Generation.

Matching the Iris Codes

Matching the iris codes is achieved through a measure of Hamming distance. The principle is similar to that discussed in Sec. 3.3.1 when used with Gabor wavelets, but is computed differently by Eq. (11):

$$HD = \frac{1}{2N}\left[\left(\sum_{i=1}^{N} A_h(i) \oplus B_h(i)) + \sum_{i=1}^{N} A_v(i) \oplus B_v(i)\right]\right) \tag{11}$$

where N is the number of values used from the codes, A_h and B_h are the two horizontal iris codes, and A_v and B_v are the two vertical iris codes. Furthermore, the measurement excludes cells which have a zero value at the same position in both of the codes being compared; for example, if $A_h(i) = 0$, and $B_h(i) = 0$, the cell at position i is cleared in A_h and B_h, and also in A_v and B_v so that the number of cells being compared in the vertical code comparison is the same as that in the horizontal comparison. The number of cells used, N, must therefore be calculated taking this into consideration.

Accounting for Eyelid Occlusion

Eyelid occlusion is managed in [4] at the normalisation stage. The normalised rectangular iris only contains image data from the left and right side of the iris, specifically the area from 45°to to 315°and 135°to 225°, where 0°could be represented as the three o'clock position on a clock face. However, it is possible to use mask data that is generated during normalisation in the matching process after resizing it to match the dimensions of the code. A new method has been derived, where the cell positions in the code are compared with the excluded areas in the mask; then in calculating the Hamming distance, if a cell's position lies in an area of exclusion in the mask, cells with that position in the horizontal and vertical codes are removed (in a similar fashion as pairs of zero values are in [4]).

3.3.3 Haar Wavelets

Haar wavelets can also be used to encode iris information in [16]. The Haar wavelet's mother function is defined in Eq. (12), and the wavelet itself is shown in Fig. 19:

$$\psi(t) = \begin{cases} 1 & 0 \leq t < \frac{1}{2}, \\ -1 & \frac{1}{2} \leq t < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

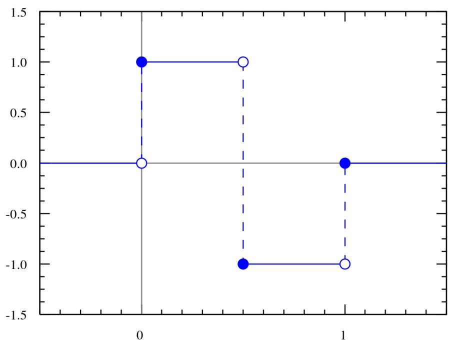


Figure 19: The Haar wavelet.

The filtering of an m x n image using Haar wavelets produces four $0.5m$ x $0.5n$ images. These are the horizontal, vertical and diagonal detail images, as well as a lower resolution approximation image. The approximation image can then be used again for further wavelet decomposition to a specified level. Due to the size reduction it is common for decomposed images to be represented as in Fig. 20, where a_i, h_i, v_i and d_i are the approximation, horizontal, vertical and diagonal detail images at level i.

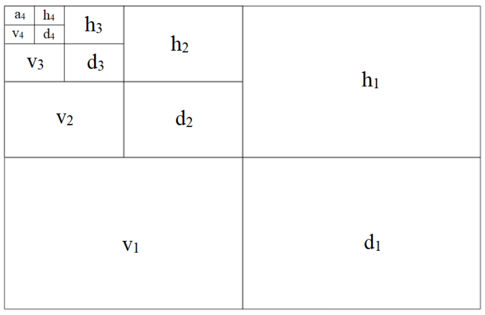


Figure 20: Haar decomposition structure.

MATLAB's haart2 function allows the filtering of an image using Haar wavelets to a specified level, returning the coefficients of all images at each level. Initially, as in [16] a normalised 60 x 450 pixel iris image is histogram equalised and then decomposed using Haar wavelets up to the fourth level. Using the first or second levels of the decomposition would still involve a large number of coefficients, and so the fourth is taken to minimise the final code length. At the fourth level, the horizontal image is 4 x 29 pixels in size, and the vertical and diagonal images 4 x 28 pixels; the three images are then appended together into one 4 x 85 pixel image, yielding 340 coefficients. The approximation images are discarded as they are used in generation of successive levels. The image is altered into a usable iris code by converting positive coefficients to one and negative coefficients to zero.

(a) Combined horizontal, vertical and diagonal details.

(b) The resulting code.

Figure 21: Generation of the iris codes using Haar detail images from S1003R01.

Matching the Iris Codes

The codes generated are matched using their Hamming distance - as described in Sec. 3.3.1 and 3.3.2 - however once again computed differently, shown in Eq. (13):

$$HD = \frac{1}{N} \sum_{i=1}^{N} (codeA_i \oplus codeB_i) \tag{13}$$

where N is the number of coefficients used and *codeA* and *codeB* are the two codes being compared.

Accounting for Eyelid Occlusion

In [16] it is noted that a perfect matching score was unobtainable due to image noise, such as eyelashes not being segmented properly. As well as this, with the code being only 4 pixels in height, a 1 x n pixel mask cannot easily be used as it would exclude 25% of the iris vertically; in fact, such a mask could only exclude areas vertically in quarter intervals, so generally the mask would over or underestimate vertical occlusion significantly and introduce more matching errors than it prevents - therefore no cropping of eyelid areas is performed in the localisation stages prior to this encoding method.

3.3.4 Local Gradient Histograms

Local regions in the normalised iris image can be described directly by their pixel values. One way to do this is to generate local histograms of pixel intensity at various points in the image. As mentioned in Sec. 3.3.1, image descriptors need to be invariant to illumination and contrast, else elements of specular reflection on the iris would greatly influence the matching effectiveness of pixel intensity descriptors. As well as this, for grey scale iris images in particular, individual descriptors would not be very distinguishable due to the relatively consistent pixel intensity throughout images in the dataset.

Instead, an approach using a histogram of oriented gradients (HoG) was proposed by Dalal and Triggs [25], where local gradient histograms are used as image descriptors for person matching - but the principle can also be applied to iris recognition. The gradient directions of an image can be calculated with a Sobel operator [22], however it was found that "Simple 1-D $[-1, 0, 1]$ masks at $\sigma = 0$ work best" in their application, and further using Gaussian smoothing before extracting the gradient information in fact reduced performance. Before the histograms are generated, images to be matched are first normalised to size 32 x 256 pixels as described in Sec. 3.2.

From a set of image gradient directions in the x and y directions, the gradient direction θ at a specific pixel f can be calculated with:

$$\theta = \arctan\left(\frac{\delta f}{\delta y}, \frac{\delta f}{\delta x}\right) \tag{14}$$

and the magnitude of the gradient with

$$m = \sqrt{\left(\frac{\delta f}{\delta x}\right)^2 + \left(\frac{\delta f}{\delta y}\right)^2} \tag{15}$$

A fixed size square window is passed over the image, and the gradient directions of the pixels in the window are grouped into a number of buckets representing angles between 0° and 180° in 22.5° increments. As edges are symmetrical, their absolute value is used, and the magnitudes of these angles act as weights to form histograms, that are taken as the descriptor at that window location.

Matching the Gradient Histograms

To compare images, the difference between the corresponding buckets of complementing histograms are calculated as in Eq. (16):

$$|A.histogram_{m,n}.bucket_p - B.histogram_{m,n}.bucket_p| \tag{16}$$

The summation of these values yields a goodness-of-match value, where a lower value indicates a closer match.

Accounting for Eyelid Occlusion

In order to account for bad patches of the eyelids the method was adapted to make use of the iris masks generated from the normalisation process (Sec. 3.2). To do this, for each region in the normalised image the window passed over, the same region in the normalised mask was checked to see if it contained various proportions of mask bits. If the masking level of the region was above a percentage threshold, the histogram was set to a NaN value, and then during comparison whenever either of the image's histograms were NaN, they were skipped. The resulting match value was also normalised through division by the number of successful comparisons on the two images (i.e. the number of window locations that were not masked above a threshold in either of the images).

3.4 Building a Database

In order for the system to be able to perform identification and verification a database of subjects needs to be created. The structure of the dataset was maintained for the database, with images for matching being generated and saved in the same folder as the source files. A naming convention was followed:

- 'S1001L01.jpg' - the original 280 x 320 subject eye image
- 'S1001L01_n.jpg' - the normalised iris image
- 'S1001L01_c.jpg' - the iris code
- 'S1001L01_m.jpg' - the iris code noise mask (if applicable)

Original images and the normalised images were kept for reference, so they can be displayed in the GUI (Sec. 3.5). Each folder also contained a file params.csv, for holding the iris and pupil location details for all the images of that eye, in the following format:

$$id,\ pupilRow,\ pupilCol,\ PupilRad,\ irisRow,\ irisCol,\ irisRad \tag{17}$$

where *id* is the number of the image in the folder (e.g. for image S1022R05 the *id* field would be '5'). This allows for faster reconstuction of the database in case it is damaged, as the localisation process does not have to be repeated.

Loading the Database

The database is loaded as a MATLAB struct object, a data type that groups using fields that can be identified by name with the dot notation. As memory consumption is not a priority, the whole database is loaded initially to save disk reading during the system's use. The struct object is defined with fields as follows:

- *name* - the name of the image, such as 'S1007R01'
- *original* - the original 280 x 320 subject eye image
- *normalised* - the normalised m x n iris image
- *code* - the iris code
- *mask* - the iris code noise mask (if applicable)
- *pupilRow* - the row co-ordinate of the pupil's centre point in the original image
- *pupilCol* - the column co-ordinate of the pupil's centre point in the original image
- *pupilRad* - the radius of the pupil in the original image
- *irisRow* - the row co-ordinate of the iris' centre point in the original image
- *irisCol* - the column co-ordinate of the iris' centre point in the original image
- *irisRad* - the radius of the iris' in the original image

3.5 Building a GUI

While the system could be run as a sequence of function calls from MATLAB's command window, this requires knowledge of how the functions take arguments and return values, and the order they must be called in. Therefore, for user convenience a GUI can be used to hide the complexities of the underlying logic while still providing full control over the system. 'App Designer' is a development environment in MATLAB for building applications that link to MATLAB code. Components can be dragged and dropped onto the window design view, and linked to callback functions in the code view that run upon user input through buttons, sliders, or other components that mimic the style of scientific instrument panels.

A single application was built for both tasks of verification and identification, shown in Fig. 22. Controls are grouped on the left hand side, where the user can first build a database from the dataset or load an existing one. Two tabs determine whether the system is in identification mode (where the user can perform one-to-one matching by selecting both the subject and the query images) or verification mode (where the user selects a subject image and one-to-many many matching determines the correct subject). When images are loaded by the user they are automatically localised according to the selected method in the localisation button group, and then normalised, encoded, and displayed. If a method of localisation fails, the user can reattempt localisation of either image with the other method. File names are displayed in the image captions, and a match/identification indicator is used to determine if the match or verification outcome was correct.

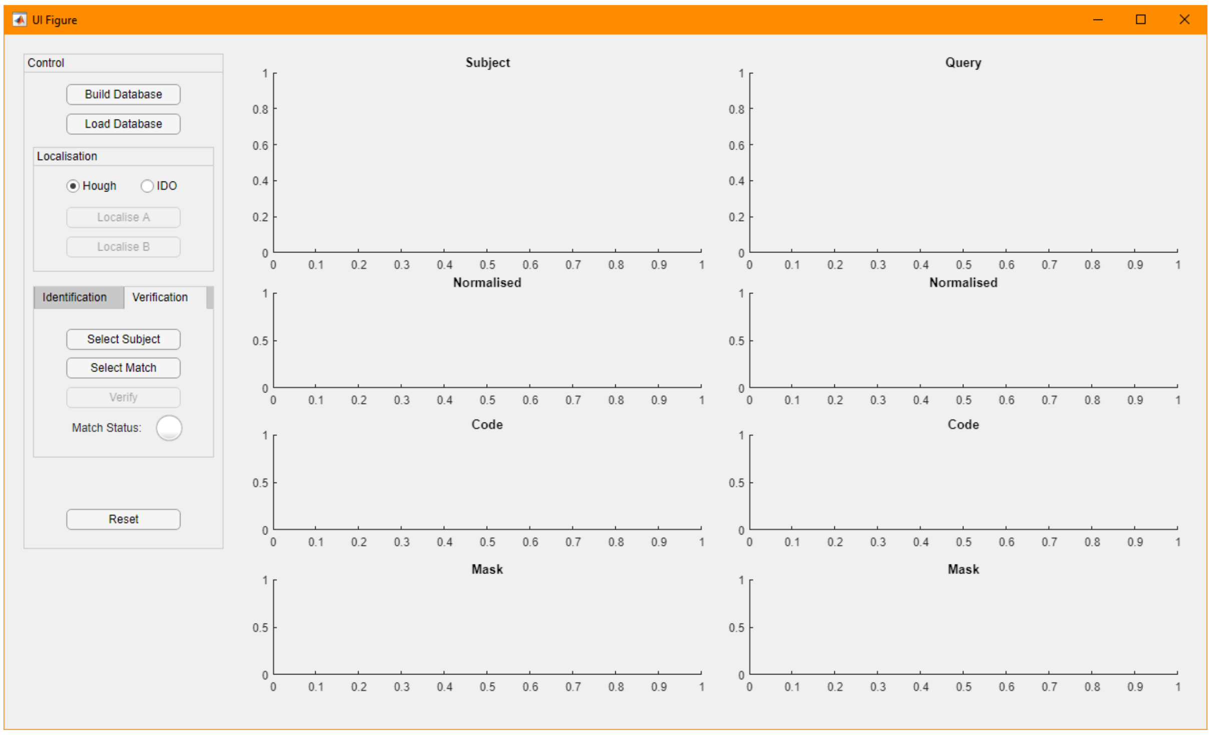


Figure 22: The skeleton GUI ready for the final methods to be implemented.

4 Implementation Issues

4.1 The Dataset

The dataset used contains a wide range of iris images, which, while useful for testing the accuracy of the system, can also cause erroneous matches due to image quality. The sheer scale of the dataset meant that during the development of the system, the testing of various features had to be done on a small set of subjects due to time constraints. Moreover, some of the eyes were not accurately localisable by my algorithms. Subsets *D1*, *D2* and *D3* were created for testing - successive sets of eyes that increase in size and localisation difficulty. A listing of their contents can be found in Appendix A.

4.2 Ambiguity in matching by 2D Gabor Wavelets

As discussed in Sec. 3.3.1, encoding via Gabor wavelets is implemented in [1] using "2D Gabor wavelets"; that is, more than one wavelet. The author's use of multiple wavelets was unclear in that:

- The author describes that his implementation results in an iris code that is just 2048 bits long. Using one wavelet would require a normalised image of 1024 bits to generate this code, therefore using multiple wavelets on the whole image would need a reduction in image size (e.g two wavelets applied globally to an image of 512 bits) in order to maintain the same resulting iris code size. Alternatively, the implementation may use multiple wavelets at various positions on the iris image, and at various scales, but like the wavelet frequency these specifics are alluded to but not defined.
- If more than one wavelet was to be used globally on the image and the size restriction ignored, the resulting iris code would not only be much bigger but complicate the Hamming distance matching procedure. If several global encodings were appended together as one long code, the shifting involved in finding the lowest distance would have to account for this and a more time consuming set of local shifts would have to be performed. Combined with an increased code size this would further reduce match time and while accuracy rates may increase if implemented correctly, the time taken for the implementation to run would not be viable, even in the context of the project.

Because of this, the decision was made to implement 2D Gabor wavelets for encoding using just one wavelet. While this may not present a solution that is as accurate as it could be, or even as fast, the matching accuracy is sufficient for the level of this project.

4.3 Wilde's Gaussian Laplacian

Wildes [7] applies a Laplacian of Gaussian filters to the image to extract feature and texture information at a variety of scales. Gaussian filters are generated procedurally and used to create a set of low-pass images. The low pass images are then used to generate the Laplacian pyramid. This encoding technique was implemented successfully.

Using the mean intensity and standard deviation of the images, matching is performed using the normalised correlation between corresponding 8 x 8 pixel sections of the two images, generating a group of block correlations for all images in the Laplacian pyramid. Median values are used to represent each layer of the pyramid.

To determine the goodness of match of two images the four correlation values must be combined. [7] uses Linear Discriminant Analysis (LDA) to minimise inter-class variance and maximise between-class variance, so that a "separation point" can be used to classify images. As the match values are generated from a comparison, each individual subject eye requires its own specific classifier. This classifier categorises correlation values between the subject eye template it represents and an unknown query image into an accept/reject judgement.

Each classifier must be trained on two disjoint sets of genuine and imposter images. In the CASIA dataset, most subjects have between nine and ten images. Subjects with less than this could not be used as there were not enough images to create the two sufficiently large sets. Because of this, the implementation was discarded as the lack of training data per eye meant that classifiers could not be generated effectively, and in many cases could not be generated at all.

5 Results and Discussion

5.1 Localisation

Finding circles with the Hough transform proved to be much faster than when using the IDO. The IDO completes in approximately 16 seconds, whereas the Hough transform completes in approximately two seconds as a result of its lower computational complexity - where the IDO performs integration around a circle and finds maxima in the differential, the Hough transform is simply a voting algorithm. For most cases, the Hough transform is adequate for localisation but excessive eyelash coverage can throw the voting off. However, in many cases the IDO localises the boundaries more accurately than the Hough transform, specifically when the eye is slightly occluded by the eyelid. It also responds less to eyelash occlusion, where the Hough transform often interprets the noise generated from eyelashes as edge points that could lie on the iris borders.

5.2 Feature Extraction and Matching

Measuring Implementation Effectiveness

The effectiveness of the implemented methods (and a system as a whole) can be measured in the number of mistakes made when matching. Mistakes are categorised as false acceptances, where the system determines that two different eyes are the same, and false rejections, where the system determines that two images of the same eye are from different people [26]. The decision threshold t used in matching determines the false acceptance rate (FAR) and false rejection rate (FRR). This is shown in Fig. 23, where hypotheses H_0 is that the subject image does not come from the same person as the template, and H_1 is that it does; and the decisions D_0 and D_1 are that a person is not and is who they claim to be. Therefore, a false acceptance occurs when D_1 is decided when H_0 is true, and a false rejection occurs when D_0 is decided when H_1 is true [26].

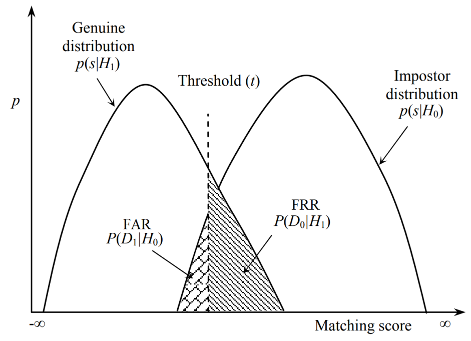


Figure 23: FAR and FRR at threshold t over genuine and imposter score distributions (adapted from Jain, Ross and Prabhakar [26], Page 5).

The threshold that minimises both the FAR and the FRR is known as the equal error rate (EER), where the number of false acceptances and rejections are equal. In consumer applications this is often taken as the decision threshold. In commercial applications the threshold may be changed specifically to minimise the FAR, such as in high security identification, or to minimise the FRR, such as in forensic identification or for user convenience.

$$FAR = \frac{imposter\,scores\,below\,threshold}{all\,imposter\,scores}$$
$$FRR = \frac{genuine\,scores\,above\,threshold}{all\,genuine\,scores} \tag{18}$$

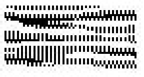
For listings of the CASIA database subsets D1, D2 and D3 that are used for testing in these sections, see Appendix A.

Gabor Wavelet Results

Using a series of wavelengths λ and orientations θ, images in *D1* were localised, normalised at the resolution implied by [1], then encoded and matched. A very low success rate was found when identifying with *D1* using these resolutions at threshold 0.32, shown in Tab. 1.

Table 1: Matching rates on normalised iris of size 16 x 64 and 32 x 32.

Normalised 16 x 64 (Code 16 x 128)					Normalised 32 x 32 (Code 32 x 64)				
λ	θ	FAR (%)	FRR (%)	Accuracy (%)	λ	θ	FAR (%)	FRR (%)	Accuracy (%)
2	0	2.67	97.07	84.91	2	0	3.80	93.87	84.35
2	45	0.32	99.73	86.60	2	45	0.04	100.00	86.81
2	90	1.92	96.67	85.61	2	90	6.26	88.27	82.95
2	135	0.36	100.00	86.53	2	135	0.08	100.00	86.77
4	0	4.93	91.47	83.68	4	0	6.32	90.27	82.63
4	45	13.66	88.00	76.56	4	45	11.64	87.20	78.42
4	90	8.85	86.93	80.88	4	90	8.28	86.40	81.44
4	135	14.53	78.27	77.09	4	135	12.22	85.73	78.11
6	0	10.95	86.40	79.12	6	0	20.14	81.20	71.82
6	45	38.34	47.33	60.46	6	45	34.87	65.07	61.16
6	90	19.01	76.13	73.47	6	90	20.83	70.27	72.67
6	135	38.16	55.33	59.58	6	135	35.56	64.00	60.70
8	0	24.91	75.33	68.46	8	0	34.81	72.93	60.18
8	45	51.07	34.40	51.09	8	45	50.34	56.53	48.84
8	90	40.99	67.87	55.47	8	90	30.67	53.87	66.28
8	135	52.12	44.27	48.91	8	135	50.22	48.00	50.07
10	0	37.05	65.60	59.19	10	0	44.63	51.87	54.42
10	45	59.79	29.37	44.19	10	45	54.75	51.20	45.72
10	90	61.76	51.07	39.65	10	90	36.93	52.00	61.09
10	135	55.99	38.99	46.23	10	135	56.61	41.33	45.40


(a) 32 x 64 Code, 90°, $\lambda = 6$

(b) 16 x 128 Code, 135°, $\lambda = 2$

Figure 24: Two inadequate iris codes generated from Gabor wavelets.

In a similar fashion a variety of sizes along with wavelet frequencies and orientations were tested with *D1* and *D2*. Using larger image sizes proved to be much more successful, as less detail was lost in the normalisation process and the best overall accuracy (97.37% with *D1*, 92.36% with *D2*) was found using a normalised image 64 x 512 pixels wide with a Gabor filter oriented at 45° with a wavelength of 14 (Fig. 25b). Using a normalised image of this size results in very slow identification, so a 32 x 256 pixel image was used. For an image this size the best accuracy (97.37% with *D1*, 88.98% with *D2*) was found with a wavelength parameter of 6 and vertical (0°) orientation (Fig. 25a).

(a) 32 x 512 Code, 0°, $\lambda = 6$

(b) 64 x 1024 Code, 45°, $\lambda = 14$

Figure 25: Two successful iris codes generated from Gabor wavelets.

As expected, cropping eyelids had no effect on accuracy identifying subjects in *D1*, but increased accuracy by 1.1% when using *D2* to generate the best overall accuracy above.

Table 2: Hamming Distance matching speeds.

Size of Image	Size of Code	Time (s)
32 x 32	32 x 64 - 256 Bytes	0.068
16 x 64	16 x 128 - 256 Bytes	0.078
16 x 128	16 x 256 - 512 Bytes	0.192
32 x 256	32 x 512 - 2,048 Bytes	0.493
64 x 512	64 x 1024 - 8,192 Bytes	1.481

Increasing the image size reduces the speed of the matching as anticipated. The time taken to perform one hundred comparisons at various sizes is detailed in Tab. 2. While increasing the size of the bit code does improve the accuracy of the match, the overhead of holding larger image files (two 8kB files per eye, compared to two 2kB files when using a 32 x 512 code or two 256 byte files in [1]) and the additional computational time needed for determining the Hamming distance between these larger bit codes means that using a 64 x 1024 bit code is not an effective solution. However, this does not imply it is impossible - [10] use a normalised image of this size, eventually generating a feature vector "of length 1,536". A 32 x 512 cell code still presents a favourable accuracy rate, with the format taking up a quarter of the space needed by the 64 x 512 code and matching much faster.

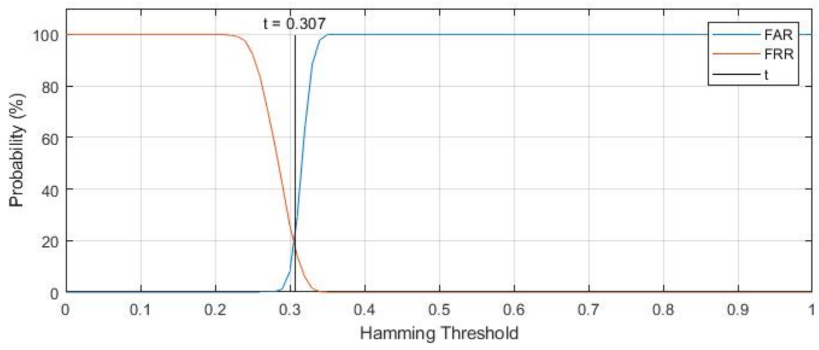


Figure 26: FAR and FRR relating to the Hamming distance threshold for the Gabor wavelet method.

Fig. 26 shows the FAR and FRR based on the Hamming distance threshold chosen when using a 32 x 512 cell code. The EER lies at a threshold of 0.31. Fig. 27 shows the Hamming distance decision environment.

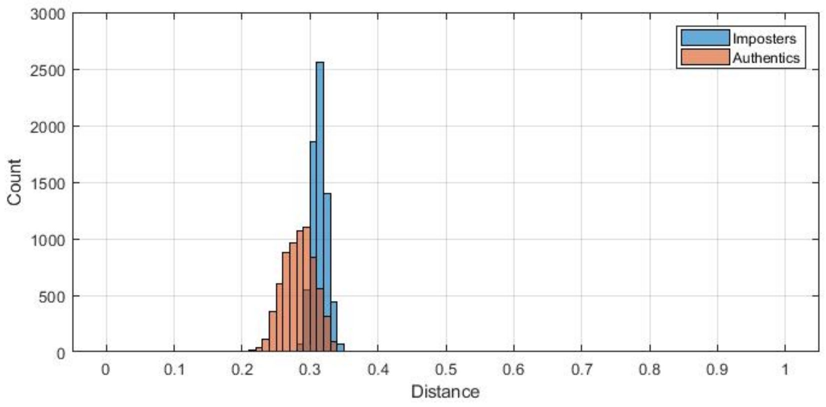


Figure 27: Hamming Distance Results for the Gabor wavelet method.

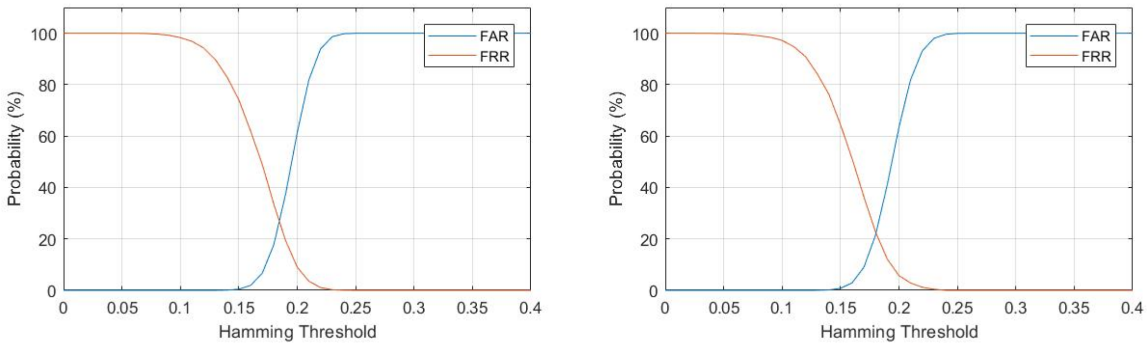
Cumulative Sum Results

Accuracy rates are shown in Tab. 3. Using the new masking method increased the accuracy by up to 2.5%, but was less effective in situations where iris occlusion was more prevalent e.g. when using *D3*. The masking method is most effective when the occlusion is minor.

Table 3: Accuracy Rates (%) using Masking vs. Cropping.

Dataset	Cropping	Masking
D1	90.79	93.42
D2	71.40	72.72
D3	70.26	71.88

Fig. 28 shows the FAR and FRR based on the Hamming distance threshold for the cropping and masking method. The EER is lower when using the proposed masking method (FAR 9%, FRR 36.32% vs. FAR 6.57%, FRR 49.07% when cropping), but the threshold at which it lies - 0.18 - remains the same.



(a) Cropping (b) Masking

Figure 28: FAR and FRR relating to the Hamming distance threshold for the cumulative-sum method.

Fig. 29 shows the distribution of Hamming distances when matching the set *D2* using the masking method. The lower accuracy rate of this implementation overall is visible in the overlap of the genuine and imposter classes.

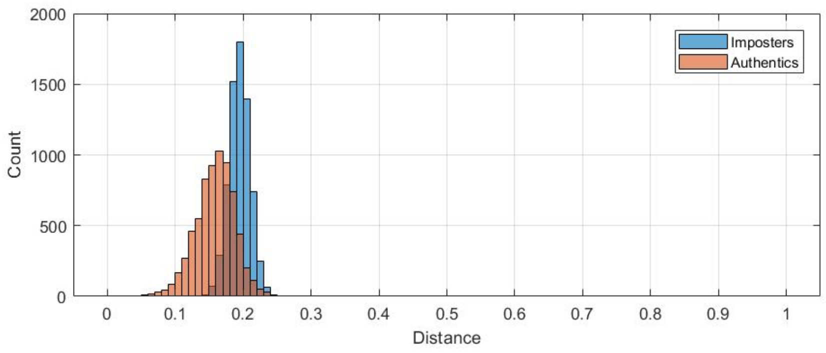


Figure 29: Hamming distance results for the cumulative-sum method.

Haar Wavelet Results

Using Haar wavelets for identification proved 98.68% accurate with *D1*, dropping slightly to 93.71% when using *D2*. Matching 1000 codes took 0.034s, due to the fast Hamming distance calculation that did not involve shifting (unlike the Gabor wavelet method implemented earlier). As the code is made up of three concatenated codes (each derived from the entire iris), shifting the entire code in matching actually reduced the accuracy with *D2*. Shifting the three segments individually also reduced accuracy with *D2* to 92.74%. In a 28-29 pixel wide image of the iris each pixel accounts for roughly 3.5°of the eye, so shifting the code by just one pixel catered for more rotation than necessary and introduced errors.

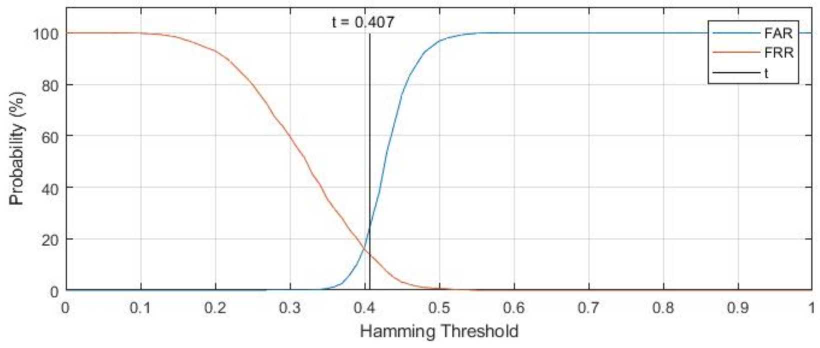


Figure 30: FAR and FRR relating to the Hamming distance threshold for the Haar wavelet method.

Fig. 30 shows the FAR and FRR based on the Hamming distance threshold chosen, with the EER at a threshold of 0.41 using *D2* (which is higher than in the Gabor wavelet method) and in [16] where it is 0.32 - this is likely due to using a lower quality set of test data, as the EER lies at 0.38 when using *D1*. Fig. 31 shows the Hamming distance decision environment.

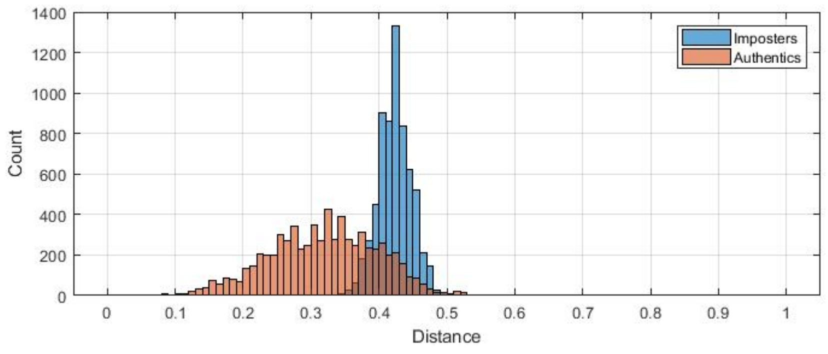


Figure 31: Hamming Distance Results for the Haar wavelet method.

Local Gradient Histogram Results

Without using any techniques to isolate the eyelid region in the normalised images, an 94.74% successful match rate was achieved using dataset *D1* and a window size of 16 pixels. Using *D2*, the accuracy dropped to 88.68%.

In an effort to improve the matching success rate, edge magnitude information was discarded - instead of the magnitude being used as a vote weight, constant vote weights for every angle were used. Initially matching eyes through this method was very successful - returning a 99.70% successful match rate using set *D1*, and only seeing a slight decrease in accuracy to 95.74% when using *D2*. As well as this, the time taken to generate the histograms was reduced, as angles in the gradient image did not have to be linked to the magnitude image, but as expected the matching time remained unchanged.

Table 4: Accuracy rates (%) using HoG.

		Window Size (Pixels)			
Dataset	Histogram	4	8	16	32
D1	Unweighted	98.68	99.70	97.37	82.89
	Weighted	81.58	88.16	94.74	90.79
D2	Unweighted	95.20	95.74	91.01	70.41
	Weighted	70.41	88.01	88.68	78.05

The window size determines the number of edge points that are included in the histogram, which therefore affects the matching rate. Tab. 4 shows the matching success rate for various sized square windows on the two previously mentioned subsets, generating weighted and unweighted histograms. Multiples of the normalised image size were used in order to eliminate the need to deal with wrapping the window at the edge of the image.

Using masks did not consistently increase the accuracy with various sets of images that were tested, when using a window size of 8 x 8 pixels and thresholds 1%, 2%, 5%, 10% or 15%. Any thresholds above 15% had no effect on the accuracy as in no case was the windowed area more than 15% covered by the corresponding mask bits.

Fig. 32 shows the FAR and FRR based on the similarity distance threshold. The EER is slightly lower when using the proposed masking method, lying at a threshold position of 4699.

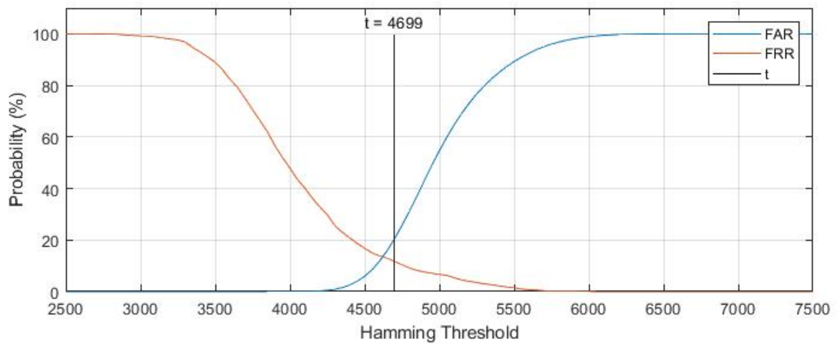


Figure 32: FAR and FRR relating to the similarity distance threshold for the HoG method.

Fig. 33 shows the distribution of Hamming distances when matching the set *D2* using the masking method.

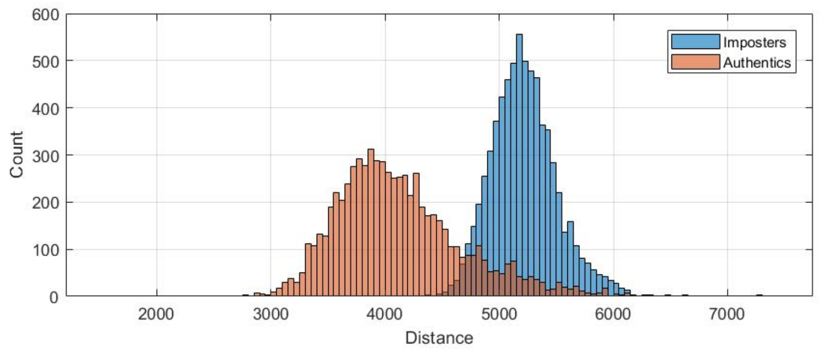


Figure 33: Hamming distance results for the HoG method.

Feature Extraction and Matching Comparison

Overall, the use of a gradient-based decomposition was found to be the best performing method, with the accuracy of 95.74%. Haar Wavelets were also relatively successful. The Gabor wavelets were less accurate, and the lowest success rate came from cumulative-sum analysis. In all cases, the accuracies are lower than those reported in their papers [1], [4], [16], [25]. This was expected due to testing with a larger set of subjects of generally more varied occlusion, because of to the knock-on effect of sub-optimal localisation or normalisation in the matching phases. Key scores from different methods of encoding and matching using *D2* are given in Tab. 5.

Table 5: Identification accuracy, FAR and FRR (at EER), encoding size and comparison speed for all implemented methods.

				Iris Code Size		
Method	Accuracy (%)	FAR (%)	FRR (%)	Coefficients	Bits	Time (s)
HoG	95.74	10.19	16.27	1,024	8,192	0.114
Haar wavelets	93.71	9.37	17.06	340	340	0.034
Gabor wavelets	92.36	3.85	25.39	16,384 (x2)	16,384 (x2)	26.81
Gabor wavelets	88.98	7.92	25.94	4,096 (x2)	4,096 (x2)	7.225
Cumulative-sum	72.72	11.29	33.89	500 (x2)	4000 (x2)	0.066

Matching speed is particularly important during identification. The time taken to compare a code with one thousand other codes in each of the implementations is detailed above in the 'Time (s)' field. The shifting necessary when using Gabor wavelets slowed down the overall implementation speed, as several comparisons were made for evaluating the match between a subject and a query. Had these shifts not been performed, the performance would have been proportional to that in the Haar comparisons, which used no shifts but a much smaller code.

The number of coefficients in the codes generated varied between methods. As decomposition in the Haar wavelet method significantly reduced the size of the image the code was generated from, the code was the smallest overall. Even though it did not use masks, the HoG method used a greater number of coefficients than the cumulative-sum method did with a mask; codes generated from the Gabor wavelet method were by far the largest, due to the size of the normalised image the code was generated from. The code generated from Haar encoding is also entirely binary, whereas codes from the HoG and cumulative-sum methods use unsigned integers, each requiring eight bits for storage. The Gabor code is also binary, but again the fact it is generated from a much larger normalised iris image means its size is affected accordingly. One thousand Haar wavelet codes can be stored in 42.5kB, resulting in a very small database, whereas this same space could only cater for 41 Gabor iris codes and their masks.

Cumulative-sum analysis was the most straightforward method to implement for iris encoding and matching. As noted in Sec. 2.3, the processing only involves addition and subtraction and is clearly explained in [4], but the implementation's poor performance overall can be partially attributed this simplicity. Likewise, the HoG implementation was relatively straightforward, and while not previously applied to iris recognition, the technique is well documented for other image and pattern matching applications - and unlike the cumulative-sum method it performed very well. The use of Gabor wavelets was the most complex, due to the lack of clarification on wavelet size, orientation, position and other key parameters in [1] (discussed further in Sec. 4.2), and so these all had to be determined in testing. Haar wavelets were much simpler to implement as these parameters did not need to be determined.

5.3 Final Implementation

In the final system, the IDO is used for initially building the database, and then in operation the Hough transform is the primary method of localisation. If the system does not locate the iris adequately with the Hough transform, the user can relocate the iris using the IDO. Encoding and matching for identification and verification is implemented using a HoG-based decomposition.

The codes used are stored as a cell array of row vectors that represent the histograms, and therefore cannot be stored as images as originally planned in Sec. 3.4. Instead, codes are stored as .mat objects, binary MATLAB files that store workspace variables for later use. While individual .mat files for each code could be stored instead of the .jpg codes, it is more convenient to store the entire database struct as one larger .mat object. This means the database can be loaded altogether, rather than through iterating a file structure and naming scheme to find each code, which would then be added to a newly created struct at runtime. When the database is built, only the normalised images are stored alongside the original images. No masks are involved, so the database is significantly smaller in size.

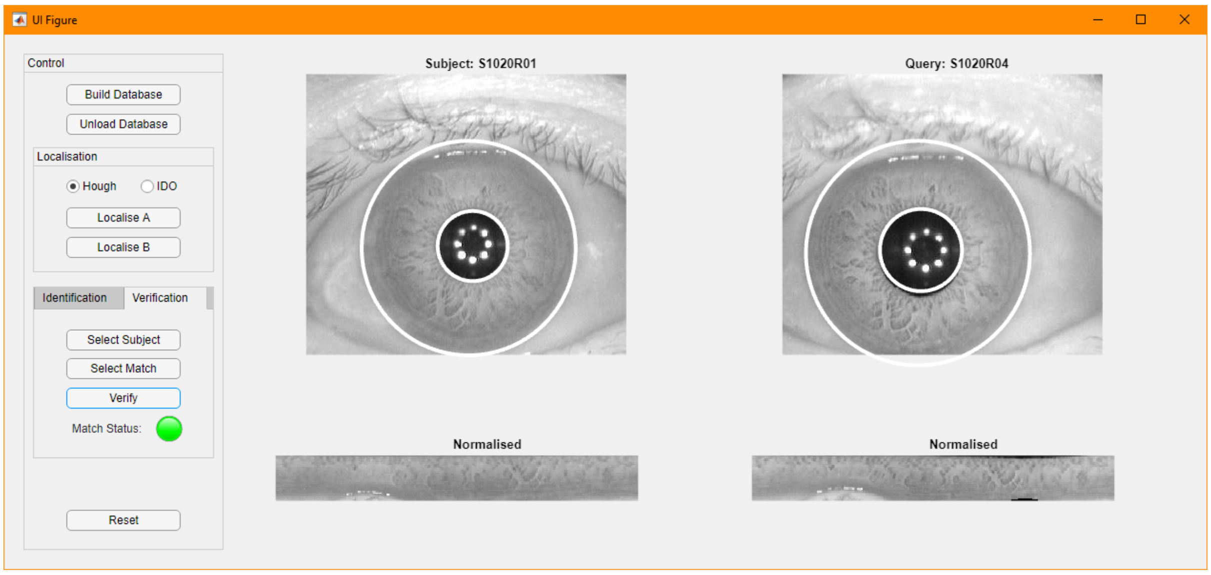


Figure 34: Successful verification in the final system.

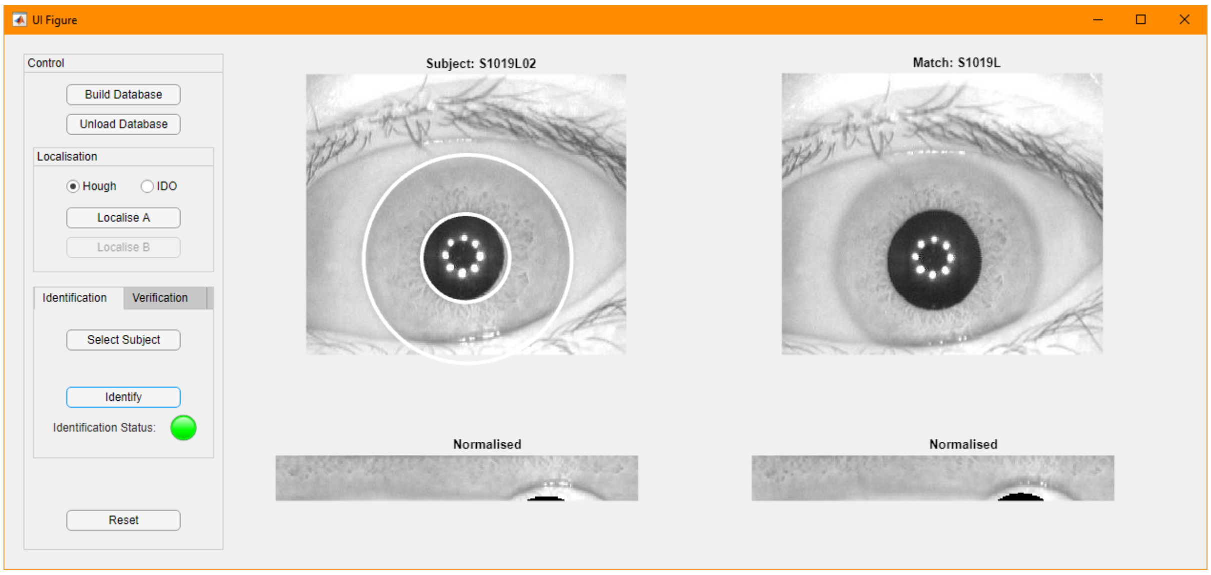


Figure 35: Successful identification in the final system.

6 Conclusion

Reflecting on the finished project, the goals set out in the original project brief (Appendix C), and the proposed implementation in the progress report (Appendix D), the project was successful in its aims, however not implemented entirely as initially planned.

While in Sec. 3.1 the implemented localisation methods are those described in the proposed system, the four methods of extraction and matching in Sec. 3.3 are not the same as in Appendix D. Only one of the proposed methods was implemented successfully; another partially, one abandoned (see Sec. 4.3) and one left unattempted. After further reading, different methods were chosen that captured a variety of information types:

- Phase based encoding with 2D Gabor wavelets, proposed Daugman [1]
- Local intensity encoding through cumulative-sum analysis, proposed by Ko et al. [4]
- Detail-decomposition based encoding with Haar wavelets, proposed by Tze Weng et al. [16]
- Gradient based encoding through local gradient histograms, proposed by Dalal and Triggs [25]

Despite these changes, the initial project goal to 'implement different methods to encode the texture information of the iris' was still met. The final local gradient histogram approach achieved a 99.70% accuracy over the *D1* set of 76 images, and a 95.74% accuracy over the *D2* set of 1034 images - an accuracy and reliability that satisfies the project goal to be able to identify a match from approximately 150 other images, with an identification accuracy above 95%.

An interface was created that allows the user to choose an image from the dataset to verify or identify, and the entire iris recognition process is visibly broken down into the stages laid out in this report. Matching success was measured through identification accuracy rate, the False Accept Rate (FAR) and the False Reject Rate (FRR).

7 Future Work

There are several areas for development of the project that could be explored in the future:

- The use of a camera to capture subject eye images in real time
- An improved eyelid segmentation technique, such as using a Hough transform in [14] and [8]
- An improved eyelash removal technique, such as fuzzy filtering in [9]
- Investigations into the effect of off-angle iris images, contact lenses, glasses and surface glaze on identification and verification accuracy
- Restricting the iris code size when using Gabor wavelets to that in [1], or the use of multiple wavelets instead of just one (discussed in Sec. 4.2)

8 Project Management

Being new to the MATLAB environment, to minimise damage in the case of file corruption or loss I produced daily backups of my MATLAB code on an external drive, and produced my report using ShareLaTeX[1], an free online LaTeX editor and compiler that allows for downloading of the source files generated and output .pdf files.

Fig. 36 details how I allocated my time for the project in mid-December in my project brief. Fig. 37 shows how this time was actually used in the following months. I tried to stay a week ahead of schedule throughout the project, allowing spare time in the case of any unexpected illness or delays. At the end of January, my project took a back seat as I focused on my semester 1 exams, and also saw less attention than intended as I was out of the country at the start of February. The localisation stages (particularly fine-tuning the IDO) took longer than I had anticipated over the Christmas break and so starting work

[1]www.sharelatex.com

on the extraction and matching phases was delayed, and I found myself using time I had set aside as mentioned earlier. Other parts such as building the UI took much less time than I had expected, and stages like building the database and writing the report covered a longer time frame due to the need for multiple iterations and refinements. I believe the use of the Gantt chart allowed me to stay on target for the project deadline very effectively.

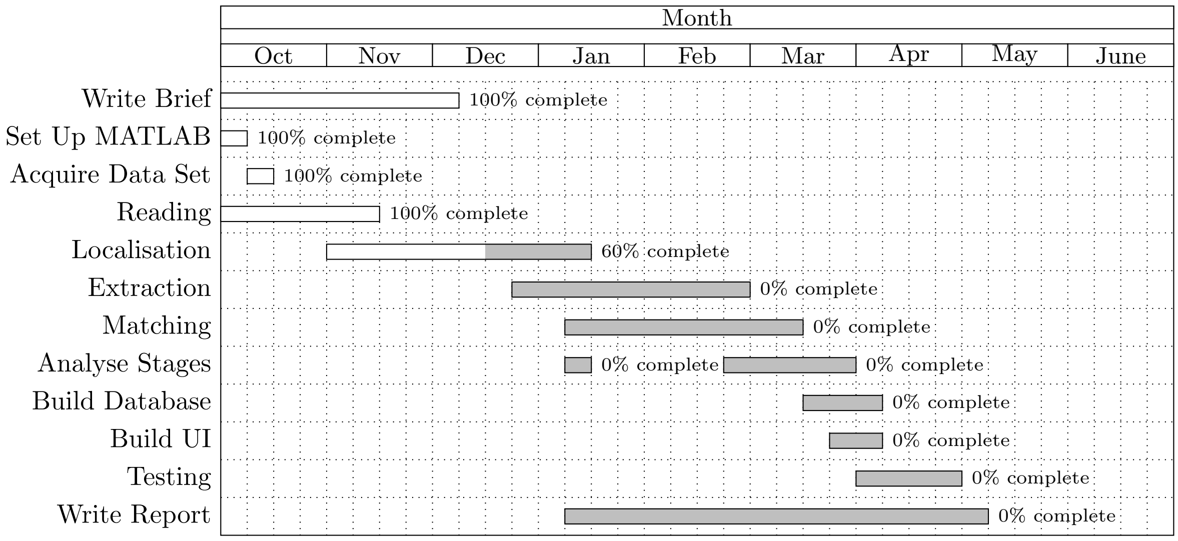


Figure 36: Gantt Chart of Planned Work (December 2017).

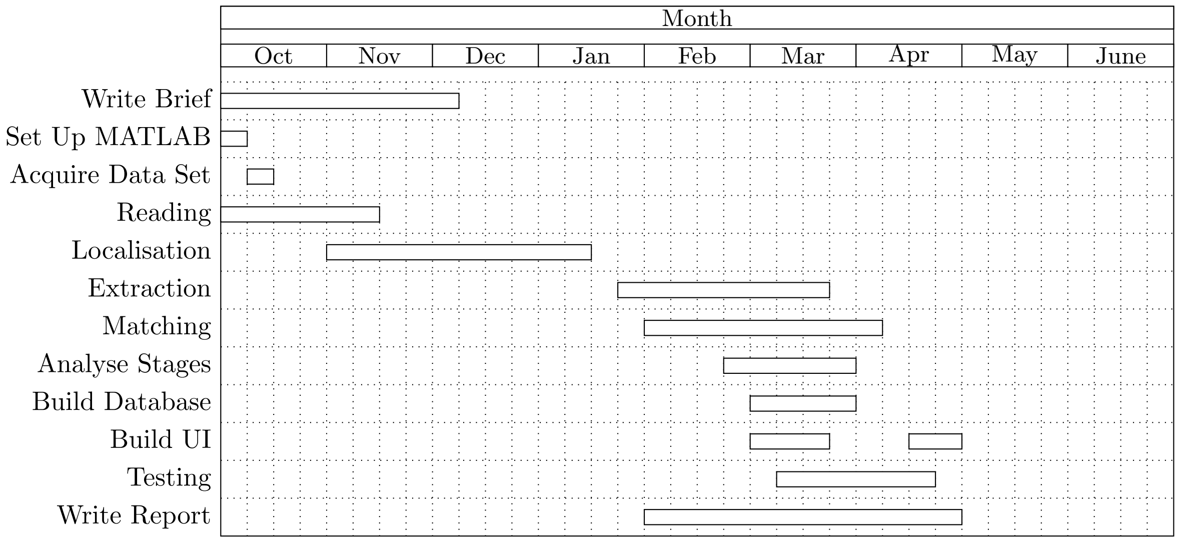


Figure 37: Gantt Chart of Actual Work (May 2018).

References

[1] J. Daugman, "How iris recognition works," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 1, pp. 21–30, Jan. 2004, ISSN: 1051-8215. DOI: 10.1109/TCSVT.2003.818350.

[2] W. W. Boles and B. Boashash, "A human identification technique using images of the iris and wavelet transform," *IEEE Transactions on Signal Processing*, vol. 46, no. 4, pp. 1185–1188, Apr. 1998, ISSN: 1053-587X. DOI: 10.1109/78.668573.

[3] M. A. M. Abdullah, J. A. Chambers, W. L. Woo, and S. S. Dlay, "Iris biometrie: Is the near-infrared spectrum always the best?" In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, Nov. 2015, pp. 816–819. DOI: 10.1109/ACPR.2015.7486616.

[4] J.-G. Ko, Y.-H. Gil, J.-H. Yoo, and K. Chung, "A novel and efficient feature extraction method for iris recognition," vol. 29, pp. 399–401, Jun. 2007.

[5] C.-l. Tisse, L. Martin, L. Torres, M. Robert, *et al.*, "Person identification technique using human iris recognition," in *Proc. Vision Interface*, 2002, pp. 294–299.

[6] A. A. Ibrahim, "Iris recognition using haar wavelet transform," *J. Al-Nahrain Univ. Sci*, vol. 17, no. 1, pp. 180–186, 2014.

[7] R. P. Wildes, "Iris recognition: An emerging biometric technology," *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1348–1363, Sep. 1997, ISSN: 0018-9219. DOI: 10.1109/5.628669.

[8] R. P. Wildes, J. C. Asmuth, G. L. Green, S. C. Hsu, R. J. Kolczynski, J. R. Matey, and S. E. McBride, "A system for automated iris recognition," in *Proceedings of 1994 IEEE Workshop on Applications of Computer Vision*, Dec. 1994, pp. 121–128. DOI: 10.1109/ACV.1994.341298.

[9] A. B. Dehkordi and S. A. R. Abu-Bakar, "Noise reduction for iris recognition using adaptive fuzzy filtering," in *2015 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, Oct. 2015, pp. 399–403. DOI: 10.1109/ICSIPA.2015.7412223.

[10] Y. Zhu, T. Tan, and Y. Wang, "Biometric personal identification based on iris patterns," in *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, vol. 2, 2000, 801–804 vol.2. DOI: 10.1109/ICPR.2000.906197.

[11] J. G. Daugman, "High confidence visual recognition of persons by a test of statistical independence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1148–1161, Nov. 1993, ISSN: 0162-8828. DOI: 10.1109/34.244676.

[12] J. Daugman, "High confidence recognition of persons by iris patterns," in *Proceedings IEEE 35th Annual 2001 International Carnahan Conference on Security Technology (Cat. No.01CH37186)*, Oct. 2001, pp. 254–263. DOI: 10.1109/.2001.962841.

[13] L. Ma, Y. Wang, and T. Tan, "Iris recognition using circular symmetric filters," in *Object recognition supported by user interaction for service robots*, vol. 2, 2002, 414–417 vol.2. DOI: 10.1109/ICPR.2002.1048327.

[14] L. Masek, "Recognition of human iris patterns for biometric identification," School of Computer Science and Software Engineering, University of Western Australia, Tech. Rep., 2003.

[15] P. Podder, T. Z. Khan, M. H. Khan, M. M. Rahman, R. Ahmed, and M. S. Rahman, "An efficient iris segmentation model based on eyelids and eyelashes detection in iris recognition system," in *2015 International Conference on Computer Communication and Informatics (ICCCI)*, Jan. 2015, pp. 1–7. DOI: 10.1109/ICCCI.2015.7218078.

[16] T. W. Ng, T. L. Tay, and S. W. Khor, "Iris recognition using rapid haar wavelet decomposition," in *2010 2nd International Conference on Signal Processing Systems*, vol. 1, Jul. 2010, pp. V1-820-V1-823. DOI: 10.1109/ICSPS.2010.5555246.

[17] S. Lim, K. Lee, O. Byeon, and T. Kim, "Efficient iris recognition through improvement of feature vector and classifier," *Journal of Electronics and Telecommunication Research Institute*, vol. 23, no. 2, pp. 61–70, 2001.

[18] W. T. Freeman and M. Roth, "Orientation histograms for hand gesture recognition," in *International workshop on automatic face and gesture recognition*, vol. 12, 1995, pp. 296–301.

[19] W. T. Freeman, K.-i. Tanaka, J. Ohta, and K. Kyuma, "Computer vision for computer games," in *Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on*, IEEE, 1996, pp. 100–105.

[20] H.-J. Lee and J.-H. Chung, "Hand gesture recognition using orientation histogram," in *TENCON 99. Proceedings of the IEEE Region 10 Conference*, vol. 2, Dec. 1999, 1355–1358 vol.2. DOI: `10.1109/TENCON.1999.818681`.

[21] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986, ISSN: 0162-8828. DOI: `10.1109/TPAMI.1986.4767851`.

[22] I. Sobel, "An isotropic 3 3 image gradient operator," Feb. 2014.

[23] T. Kazakov, "Iris detection and normalization," School of Computer Science, University of Birmingham, Tech. Rep., May 2011.

[24] M. Shamsi, P. B. Saad, I. Subariah, and A. Rasouli, "A new accurate technique for iris boundary detection," *WSEAS Transactions on Computers*, vol. 9, no. 6, pp. 654–663, 2010.

[25] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, Jun. 2005, 886–893 vol. 1. DOI: `10.1109/CVPR.2005.177`.

[26] A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," *IEEE Transactions on circuits and systems for video technology*, vol. 14, no. 1, pp. 4–20, 2004.

[27] *CASIA Iris Image Database*, `http://biometrics.idealtest.org`.

Appendices

A Subsets

A.1 *D1*

D1 is a small collection of 76 images from 12 eyes. These eyes are all free from occlusion. This set was generally used in the first test of each implementation iteration, as due to its size exhaustive matching could be completed quickly.

Table 6: Images included in *D1*.

Identifier	Side	Eyes Included
007	R	01 02 03 04 05 06 07 08 09 10
011	R	01 02 03 04 05 06 07 08 09 10
015	R	01 02 03 04 05
016	L	01 02 03 04
018	R	01 02 03
025	L	01 02 03 04 05
033	L	01 02 03 04 05
057	R	01 02 03 04 05
061	R	01 02 03 04 05
071	L	01 02 03 04 05 06 07
071	R	01 02 03 04 05 06 07
081	R	01 02 03 04 05 06 07 08 09 10

A.2 *D2*

D2 is a larger set of 1034 images from 157 eyes. In some cases up to 30% of the iris in these images are occluded by eyelids or eyelashes. This subset well represents the entire CASIA set. *D2* contains all the images in *D1*, plus additionally:

Table 7: Additional images in *D2*.

ID	Side	Eyes Included
007	L	01 02 03 04 05 06 07 08 09 10
008	L	01 02 03 04 05 06 07 08 09 10 11
008	R	01 02 03 04 05 06 07 08
009	L	01
009	R	01 02 03 04 05 06 07 08
010	R	01 02 03 04 05
011	L	01 02 03 04 05 06 07 08 09
012	R	01 02 03 04 05 06
019	L	01 02 03 04 05 06 07 08 09 10 11 12 13
019	R	01 02 03 04 05 06 07 08 09 10 11 12 13
020	L	01
020	R	01 02 03 04 05
021	L	01 02 03 04 05 06
021	R	01 02
030	L	01 02 03 04 05 06 07 08 09 10
030	R	01 02 03 04 05 06 07 08 09 10
037	R	01 02 03
038	L	01 02 03 04
044	L	01 02 03 04 05 06
044	R	01
046	R	01 02 03 04
053	L	01 02 03 04 05 06 07 08 09 10 11 12 13
053	R	01 02 03 04 05 06 07 08 09 10
056	L	01 02 03 04 05
064	L	01 02 03 04 05 06 07
064	R	01 02 03 04 05 06 07
072	L	01 02 03 04 05 06 07
072	R	01 02 03 04 05 06 07
075	L	01 02 03 04 05 06
075	R	01 02 03 04 05 06

Table 8: Additional images in *D2*.

ID	Side	Eyes Included
077	R	01 02 03 04 05 06 07
079	L	01 02 03 04 05 06 07
079	R	01 02 03 04 05 06 07
083	L	01 02 03 04 05 06 07
083	R	01 02 03 04 05 06 07
007	L	01 02 03 04 05 06 07 08 09 10
008	L	01 02 03 04 05 06 07 08 09 10 11
008	R	01 02 03 04 05 06 07 08
009	L	01
009	R	01 02 03 04 05 06 07 08
010	R	01 02 03 04 05
011	L	01 02 03 04 05 06 07 08 09
012	R	01 02 03 04 05 06
019	L	01 02 03 04 05 06 07 08 09 10 11 12 13
019	R	01 02 03 04 05 06 07 08 09 10 11 12 13
020	L	01
020	R	01 02 03 04 05
021	L	01 02 03 04 05 06
021	R	01 02
030	L	01 02 03 04 05 06 07 08 09 10
030	R	01 02 03 04 05 06 07 08 09 10
037	R	01 02 03
038	L	01 02 03 04
044	L	01 02 03 04 05 06
044	R	01
046	R	01 02 03 04
053	L	01 02 03 04 05 06 07 08 09 10 11 12 13
053	R	01 02 03 04 05 06 07 08 09 10
056	L	01 02 03 04 05
064	L	01 02 03 04 05 06 07
064	R	01 02 03 04 05 06 07
072	L	01 02 03 04 05 06 07
072	R	01 02 03 04 05 06 07
075	L	01 02 03 04 05 06
075	R	01 02 03 04 05 06
077	R	01 02 03 04 05 06 07
079	L	01 02 03 04 05 06 07
079	R	01 02 03 04 05 06 07
083	L	01 02 03 04 05 06 07
083	R	01 02 03 04 05 06 07
086	L	01 02 03 04 05 06
086	R	01 02 03 04 05 06 07
088	L	01 02 03 04 05 06 07
088	R	01 02 03 04 05 06 07
091	L	01 02 03
091	R	01 02 03 04 05 06
094	L	01 02 03 04 05 06 07
094	R	01 02 03 04 05 06 07
098	L	01 02 03 04 05 06 07
098	R	01 02 03 04 05 06
101	L	01 02 03 04 05 06 07
101	R	01 02 03 04 05 06 07
102	L	01 02 03 04 05 06 07
102	R	01 02 03 04 05 06 07
103	L	01 02 03 04 05 06 07
103	R	01 02 03 04 05 06 07
104	L	01 02 03 04 05 06 07 08 09 10
104	R	01 02 03 04 05 06 07 08 09 10
106	L	01 02 03 04 05 06 07
106	R	01 02 03 04 05 06 07
108	L	01 02 03 04 05 06
108	R	01 02 03 04 05 06
111	L	01 02 03 04 05 06 07
111	R	01 02 03 04 05 06 07
115	L	01 02 03 04 05 06 07 08 09 10
115	R	01 02 03 04 05 06 07 08 09 10
126	L	01 02 03 04 05 06 07
126	R	01 02 03 04 05 06 07
127	L	01 02 03 04 05 06
127	R	01 02 03 04 05 06 07
129	L	01 02 03 04 05 06 07
129	R	01 02 03 04 05 06 07
137	L	01 02 03 04 05 06 07
137	R	01 02 03 04 05 06 07
138	L	01 02 03 04 05 06 07
138	R	01 02 03 04 05 06 07
142	L	01 02 03 04 05 06 07
142	R	01 02 03 04 05 06 07
144	L	01 02 03 04 05 06 07
144	R	01 02 03 04 05 06 07
145	L	01 02 03 04 05 06 07
145	R	01 02 03 04 05 06 07
146	L	01 02 03 04 05 06 07
146	R	01 02 03 04 05 06 07
147	R	01 02 03 04 05
148	L	01 02 03 04 05 06 07
148	R	01 02 03 04 05 06 07
157	R	01 02 03 04 05 06 07 08 09 10
158	R	01 02 03 04 05
162	L	01 02 03 04 05 06 07 08 09 10
162	R	01 02 03 04 05 06 07 08 09 10
164	L	01 02 03 04 05
164	R	01 02 03 04 05
165	L	01 02 03 04 05 06 07 08 09
165	R	01 02 03 04 05 06 07 08 09
174	L	01 02 03 04 05 06
176	L	01 02 03 04 05 06 07 08
180	R	01 02 03 04 05 06
181	L	01 02 03 04 05 06 07
182	L	01 02 03 04 05 06 07 08 09
182	R	01 02 03 04 05 06 07 08 09 10
184	R	01 02 03 04 05
185	L	01 02 03 04 05 06
187	R	01 02 03 04 05 06 07

Table 9: Additional images in *D2*.

ID	Side	Eyes Included
191	L	01 02 03 04 05 06 07 08 09 10
191	R	01 02 03 04 05 06 07 08 09 10
192	L	01
192	R	01 02 03 04 05 06
194	R	01 02 03 04 05
197	L	01 02 03 04 05
197	R	01
199	L	01 02 03 04 05 06 07 08 09 10
199	R	01 02 03 04 05 06 07 08 09 10
202	L	01
202	R	01 02 03 04 05
203	L	01 02 03 04 05
203	R	01
207	R	01 02 03 04 05
208	L	01 02 03 04 05 06 07
208	R	01
209	L	01 02 03 04 05 06 07 08 09 10
209	R	01 02 03 04 05 06 07 08 09 10
210	L	01 02 03 04 05 06 07
211	L	01 02 03 04 05 06 07 08 09 10 11 12 13
211	R	01 02
212	L	01 02 03 04 05 06 07 08
212	R	01
214	L	01 02 03 04 05 06 07
214	R	01
217	L	01 02 03 04
218	L	01 02 03
218	R	01 02 03 04 05 06 07 08 09 10 11 12 13
220	L	01 02 03 04 05 06
225	L	01 02 03 04 05 06 07
227	L	01 02 03 04 05
227	R	01
228	L	01 02 03 04 05 06 07
228	R	01
230	R	01 02 03 04 05 06 07 08 09 10
231	L	01 02 03 04 05 06 07 08
235	R	01 02 03 04
239	L	01 02 03 04 05 06 07 08 09 10
239	R	01 02 03 04 05 06 07 08 09 10
240	L	01 02 03 04 05
241	R	01 02 03
248	L	01
248	R	01 02 03 04 05 06 07
249	R	01 02

A.3 *D3*

D3 is again larger than *D2*, adding images with greater occlusion levels. The purpose of using this set was too see how the increasing the number of occluded eyes further effected the accuracy rates compared to those seen in *D3*. The entire set is made up of 1143 images from 170 eyes.

Table 10: Additional images in *D3*.

Identifier	Side	Eyes Included
013	L	01 02 03 04 05 06 07
036	L	01 02 03 04 05 06 07 08 09 10
036	R	01 02 03 04 05 06 07 08 09 10
042	L	01 02 03 04 05 06 07 08 09 10
042	R	01 02 03 04 05 06 07 08 09 10
081	L	01 02 03 04 05 06 07 08 09 10
136	L	01 02 03 04 05 06 07 08 09 10
136	R	01 02 03 04 05 06 07 08 09 10
141	L	01 02 03 04 05 06
141	R	01 02 03 04 05 06

B Project Archive Contents

- D1
  - *Build structure of dataset D1*
  - database.mat
- Database Manipulation
  - buildDatabase.m
  - buildSizeDatabase.m
  - loadDatabase.m
- Feature Extraction
  - 2D Gabor Wavelets
    - compareTwoEyes.m
    - getGaborRates.m
    - getHammingDist.m
    - match.m
    - testEncoding.m
    - testEncodingDaugmanSizes.m
    - testEyelidEffect.m
    - wavelet2DExtract.m
  - Cumulative Sum Analysis
    - getCumulativeCode.m
    - getCumulativeHammingDist.m
    - getCumulativeHammingDistWithMask.m
    - getCumulativeRates.m
    - testCumulative.m
    - testCumulativeWithMask.m
  - Gradient Histograms
    - compareHistograms.m
    - getGradientRates.m
    - gradientHistogram.m
    - gradientHistogram2.m
    - gradientHistogramEyelids.m
    - testHistogram.m

cont.

- Feature Extraction *(Cont.)*
  - Haar Wavelets
    - getHammingDistHaar.m
    - haarEncode.m
    - testHaar.m
  - Wildes Gaussian Laplacian
    - buildClassifier.m
    - buildClassifiers.m
    - gaussianLaplacian.m
    - getTrainingFormat.m
    - getWeightForClass.m
    - normalisedCorrelation.m
    - testWildes.m
    - testWildesSimple.m
  - combine.m
  - getEER.m
  - plotFARFRR.m
- Hough
  - fixedRadHough.m
  - varRadHough.m
- IDO
  - findMins.m
  - IDO.m
  - IDOWithDisplay.m
- Normalisation
  - alignIris.m
  - normalise.m
  - normaliseSameCentre.m
  - normaliseSlow.m
- localise.m
- test.m
- irissystem.mlapp

C Project Brief

An Iris Recognition System

Harry Talbot
Supervisor: Sasan Mahmoodi

October 12, 2017

Problem Overview

Biometric systems for human identification have proven to be more secure than traditional password orientated methods, and while more complex they are becoming more practical. Many biometric identifiers exist - fingerprints, facial structure, voice and gait; the human iris being one of the most attractive for use. It is protected from damage by the cornea, unique due to the high level of entropy in its formation, and shows relatively little change over a persons lifetime once developed. Its distinct and easily identifiable placement on a person makes it quick to locate and therefore advantageous for fast analysis and identification.

Most systems follow a similar pattern; iris localisation, normalisation, feature extraction and matching. Most of the variation among systems is in the feature extraction - there are several algorithms, each with various levels of granularity, that will influence the overall system accuracy.

Goals

The aim of this project is to develop a piece of software that can identify people by their iris, and correctly select matching iris' from a database of around 150 images. The software should implement different methods to encode the texture information of the iris from the image.

Scope

The images should all be of the naked eye, and any with glasses, contact lenses (coloured or clear) or excessive covering by hair or eyelashes will not be correctly identified. The eyes in the image should fill the majority of the image space; the program will not extract the iris from an image of a persons face, for example. The system will be unlikely to match an iris when the given image is too low a resolution (the iris itself being less than 50 pixels wide).

D Proposed System from Progress Report

Similar to the literature reviewed earlier the system will have several stages. Different methods will be implemented for each stage before their inclusion in the final system.

Localisation

I will implement the localisation techniques earlier and determine which is the best for the final application. These techniques are:

- An Integrodifferential Operator, proposed by Daugman [1]
- A Circular Hough Transform, proposed by Wildes [7]

Feature Extraction and Matching

Similarly, I will implement several feature extraction and matching techniques. These are:

- Normalisation, followed by encoding using 2D Gabor Wavelets and matching by Hamming Distance, proposed by Daugman [1]
- A Laplacian of Gaussian filters and normalised correlation, proposed by Wildes [7]
- Circular Symmetric Filters and nearest feature line classification, proposed by Ma, Wang and Tan [13]
- Gabor filtering and a 2D wavelet transform, with matching by Euclidian distance, proposed by Zhu, Tan and Wang [10]

User Interface

The system should have a graphical user interface that enables the user to see the iris localisation process broken down into various steps. For the sake of simulation, the interface will allow users to select a subject from the data set to act as, and an iris image from that subject to be the test image; the system will then identify which subject the chosen image belongs to solely from matching with the rest of the database. Matching success will be measured by whether the subject identified by the system is the subject chosen initially by the user.

Data Set

I will be using the CASIA-IrisV4 data set [27], more specifically one subsection CASIA-Iris-Interval. This data set is collection of 2,639 very clear images, built from 249 subjects (2 iris per subject, approximately 3-7 images per iris) and is well suited for applications involving the detailed texture of the iris. In the image acquisition stages a circular array of IR LEDs was used to illuminate the subject's eye. The data set is also free to download after registration at the Biometrics Ideal Test website[2].

Implementation

The project will be implemented in MATLAB, an interpreted language that is widely used in engineering and mathematical fields. This was chosen over Java, a language I am much more familiar with, as it has a number of well-documented inbuilt functions and generally lends itself better to implementing algorithms over Java's object-oriented approach. Where a MATLAB licence is usually quite expensive, as a member of ECS I am able to install one copy on my personal machine and also use the software in the Zepler labs.

[2]http://biometrics.idealtest.org/index.jsp