A Phase-Only Correlation-based Iris Recognition System

by

Shlomo Kashani

Queen Mary
University of London

A report presented to the University of London in fulfilment of the requirement for the degree of

Master of Science in

Digital Signal Processing

London, United Kingdom, 2006

Disclaimer

This report is submitted as part requirement for the degree of MSc in Digital Signal Processing at the University of London. It is the product of my own labour except where indicated in the text. The report may be freely copied and distributed provided the source is acknowledged.

Signature

Abstract

Resolving the identity of an individual is a task of significant importance in various high security environments including, but not limited to, airport border control, academic biological research facilities, suspect detention amenities, prisons and military operations. Biometric identification systems utilizing iris recognition are known to be one of the most reliable [1] and least invasive techniques available.

Amongst the numerous iris recognition systems reported in the open literature, the earliest and most widely used is a system based on extracting textural characteristics utilizing a two-dimensional Gabor wavelet coefficient as suggested in [1] by Daugman. Conversely, one of the newest and least studied is a phase-based iris image matching system as suggested in [28] [33] by Miyazawa, a system which was at first used for fingerprint recognition [26] and image alignment [37]. At its core, the phase-correlation algorithm extracts the phases of the cross-power spectrum of two iris images and then obtains the phase correlation surface; if the two iris images belong *to the same human* the result would be a single peak, otherwise the result would have randomly scattered peaks.

We start with a binary edge-mapped iris image obtained using the Canny edge detector and, based on empirical results, we show how crucial is the selection of the variance used in the Gaussian smoothing operation for the correct segmentation of the iris and pupil. Afterwards, based on a small dataset from the UPOL [8] iris image database we show that at best only 70% of the pupils and 96% of the irises could be correctly segmented using a constant variance. Then, we demonstrate how the lower half of the iris only can be used to extract the characteristics of a human iris. Next, to compensate for iris size variations, different images are matched by a polar coordinates transformation to a fixed 128 x 256 pixel normalized rectangular image. Then we introduce the phase-only correlation (POC) method and two of its variants. After that, we show that assuming perfect iris segmentation, the POC function can be used as an iris matching criterion. Subsequently we prove, using analysis of variance, that the use of band limited and windowed POC is superior to plain POC. Next, we propose our own criteria, based on a threshold, for classifying iris images in the UPOL [8] database into impostors or genuine identity holders. Finally, utilizing the UPOL [8] database on which this method has not been tested before, we demonstrate the validity of this approach and present experimental results producing zero false (impostor) and perfect genuine matches.

The original contribution of this work is that we have successfully researched and implemented a new variation on a Fourier-based algorithm for iris encoding and matching and added it to MATLAB creating the first publicly available phase-based iris recognition toolbox.

Keywords: Identification of humans, iris recognition, Gaussian filtering, Canny edge detection, Hough transform, iris normalization, phase-only correlation, Gabor filters.

Acknowledgements

I would like to appreciatively acknowledge the wholehearted supervision of Dr. Mark D Plumbley from the Centre for Digital Music at the University of London throughout this report, for the many perceptive conversations and helpful comments on the text and his lectures on wavelet transforms.

I thank Dr. Mike Davies from the DSP group for his lectures on statistical signal processing and the technical discussions during the preceding semesters.

I thank Miyazawa et al. and Kenji et al. for the papers that inspired this report and for their email communications.

I thank Professor Monro from the iris recognition group at the University of Bath for his email correspondence and valuable information on iris image quantization.

I thank Ullrich Köthe, an assistant professor in the Cognitive Systems Group at the Computer Science Department of the University of Hamburg, for his help and advice on the realization of Gaussian and Gaussian derivative kernels and for his C++ signal processing toolbox.

I am grateful to my friends Tsela and Ruth from the University of London for their continued moral support and in particular I would like to acknowledge my spouse who conceived during this time.

I am grateful to Dr. Uri Kartoun from the Ben-Gurion University of the Negev for reviewing this report.

Finally, I am evermore indebted to my father for fostering, financing and encouraging my interest in science.

Portions of the research in this paper use the CASIA iris image database collected by Institute of Automation, Chinese Academy of Sciences.

Table of Contents

Disclaimer ii

Abstract iii

Acknowledgements iv

Table of Contents v

List of Figures viii

List of Tables x

Chapter 1 Introduction 1

1.1 Objective 1

1.2 Problem Description 1

1.3 Background and Related Work 2

1.3.1 Overview of Iris Recognition Algorithms 2

1.4 Method 4

1.4.1 Realization 4

1.5 Workflow 4

Chapter 2 Image Acquisition and Iris Image Databases 6

2.1 The Iris and Gross Anatomy of the Human Eye 6

2.2 Image Acquisition 8

2.2.1 Noise artefacts 8

2.3 Iris Image Databases 8

2.3.1 The UBIRIS Database 9

2.3.2 The CASIA Database 9

2.3.3 The UPOL Database 9

2.4 Experimental Results with the Processing of Noise Artefacts 10

2.4.1 Noise Generated by Specular Reflection 10

2.4.2 Noise Generated by Eyelashes 11

2.4.3 Noise Generated by Eyelids 12

2.4.4 Noise Generated by Skewed Pupil or Iris 12

Chapter 3 Iris Localization, Isolation and Segmentation 14

3.1 Introduction 14

3.2 The Canny Edge-Detection Algorithm 14

3.3 Two-Dimensional Approximations to First-Order Derivative Masks 15

3.4 Gaussian Filtering 17
3.4.1 The Continuous Gaussian Function 18
3.4.2 Experimental results, discretization of the Gaussian Function 18
3.4.3 Efficient Separable Representation 20
3.4.4 Discrete derivative of a Gaussian function 22
3.4.5 Machine Generation of Discrete Gaussian and GD Kernels 24
3.4.6 Impulse Response of a Derivative of a Gaussian 26
3.5 Experimental Results, the canny edge detector 27
3.5.1 Empirical Determination of the Gaussian Blur Standard Deviation 27
3.5.2 Experimental results, Segmentation Success Rates 29
3.6 The Hough Transform 30
3.6.1 The Line-Detecting Hough Transform 30
3.6.2 Accumulators and the Voting System 32
3.6.3 Parametric Coordinates 32
3.6.4 The Circular Hough Transform 35
Chapter 4 Coordinate Transformation and Iris Normalization 36
4.1 Overview 36
4.2 Cartesian to Polar Coordinate Transformation 36
4.2.1 Implementation of Image Index Quantization 37
4.2.2 Experimental Results 38
4.3 Iris Normalization 39
4.3.1 Minimal Sampling Requirements 40
Chapter 5 Feature Extraction and Iris Encoding 42
5.1 Overview of Digital Iris Code Generation 42
5.2 Encoding Using the Cross Spectrum Phase Correlation 42
5.2.1 Formulation of the Phase-Only Correlation Algorithm 43
5.3 Experimental Iris Matching Results Using POC 44
5.3.1 Genuine Iris Matching POC scores 45
5.3.2 Impostor Iris-Matching POC Scores 48
5.4 The Band-Limited Phase-Only Correlation Algorithm 49
5.4.1 Experimental Results Using BLPOC 50
5.5 The Windowed BLPOC Algorithm 51

Chapter 6 Discussion, Future Work and Conclusions 54

6.1 Discussion 54

6.2 Future work 55

6.3 Conclusion 56

Alternative Iris Encoding Methods 60

Encoding phase data using Gabor function 60

List of Figures

Figure 2-1 Frontal view of the right eye 7

Figure 2-2 Noise generated by specular reflection 10

Figure 2-3 False circle edge detection due to specular reflections 11

Figure 2-4 Noise generated by eyelashes........ 12

Figure 2-5 Noise generated by eyelids........ 12

Figure 2-6 Noise generated by skewed pupil or iris 13

Figure 3-1 Horizontal and vertical first-order gradients 16

Figure 3-2 Floating Point Valued Gaussian Kernel of Order 3 and Unit Variance and an Integer Valued Gaussian Kernel Convolution Mask of Order 5 and Unit Variance 19

Figure 3-3 3D plot of an isotropic sampled digital two-dimensional Gaussian kernel........ 20

Figure 3-4 Experimental results of Gaussian blur with different STDV applied to an iris image.... 22

Figure 3-5 Plot of a first derivative of a 1D Gaussian 23

Figure 3-6 Experimental results, 3D surface plot of a derivative of a Gaussian with different-sized kernels........ 24

Figure 3-7 8 Experimental results, Impulse response of a GD 27

Figure 3-9 10 Experimental results, edge-mapped iris with a small variance 28

Figure 3-11 Experimental results, edge-mapped iris with a large variance 29

Figure 3-12 Transformation of a point into a line in the parametric *ab* space........ 31

Figure 3-13 Two points in the Cartesian space transform into two lines in parametric space.......... 31

Figure 3-14 The Hough transform accumulation system........ 32

Figure 3-15 Relationship between Cartesian line and parametric point 34

Figure 3-16 Relationship between a Cartesian point and parametric sine wave........ 34

Figure 4-1 Transformation of an iris ring from Cartesian to polar coordinate system 37

Figure 4-2 Algorithmic realization of the Cartesian to polar coordinate system for a ring 38

Figure 4-3 Non-concentric circles lead to noise being transformed 39

Figure 4-4 Example of actual transformation of the lower-half iris section and selection of the area of interest 40

Figure 4-5 Examples of actual normalization using a constant number of quantized index points.. 40

Figure 5-1 Block diagram of the phase only correlation algorithm 44

Figure 5-2 Phase-only correlation functions........ 45

Figure 5-3 Genuine iris matching POC plots exhibit a distinctive peak........ 47

Figure 5-4 One way ANOVA for POC function means at different resolutions for genuine matching 47

Figure 5-5 Iris texture contrast enhancement using histogram equalization.................................... 48

Figure 5-6 Impostor iris matching POC plots exhibit several scattered noisy peaks........................ 49

Figure 5-7 Band-limited spectrum of a two-dimensional iris DFT 50

Figure 5-8 One-way ANOVA for POC and BLPOC functions of genuine matching 51

Figure 5-9 BLPOC score distribution 53

List of Tables

Table 1-1 Iris Recognition: Systems and Algorithms 2

Table 2-1 Iris Scan Camera Configuration Parameters 8

Table 3-1 Experimental results, run Time Comparison between Separable and Non-separable two-dimensional Convolution Demonstrating the Superiority of Separable Convolution 21

Table 3-2 Experimental results, the Effect of Histogram Equalization on Segmentation Failure Rates 29

Table 5-1 Genuine Iris Matching Phase Only Correlation Scores Using Varying Resolution without Histogram Equalization 46

Table 5-2 Impostor Iris Matching Phase-Only Correlation Scores 48

Table 5-3 Larger Scale Impostor Iris Matching POC and BLPOC Scores with 45 Matches 51

Table 5-4 BLPOC versus Windowed BLPOC Scores 52

Table 5-5 WBLPOC Threshold Values for Genuine and Impostor Matching, used to decide Acceptance/Rejection 52

CHAPTER 1
INTRODUCTION

This chapter begins by stating the objective of this report in Section 1.1 followed by a short introduction to iris recognition in section 1.2. Section 1.3 presents the background and related work and section 1.4 presents the suggested method and realization. Section 1.5 concludes the chapter with the workflow of this report.

1.1 Objective

The first objective of this report is to research and implement an iris-based biometric identification system using a variation of a phase-based image matching approach and to analyze the performance of an iris encoding approach utilizing the phase-only correlation function. The second objective is to implement this algorithm into a regularly used image processing software package such as MATLAB which is one of the most widely used signal and image processing packages. While in the open literature there are a great number of papers that implement Daugman's iris recognition algorithm[1], this report is inspired by the less known recent works of Takita et al.[27], Ito et al. [26] and most notably Miyazawa et al. [28][33].

1.2 Problem Description

The most commonly used biometric identification systems exploit the physiological or behavioural characteristic of a person such as speech, fingerprint, face, handwriting, ECG[1] and gait analysis. However, these features are time-variant and their performance in terms of false acceptance and false rejection rates is unacceptable[13]. Conversely, iris recognition is based on iris randomly formed patterns as a unique human identifier which remains constant throughout the course of a person's life. Biometric identification systems utilizing iris recognition are known to be one of the most reliable [1] and least invasive techniques available. Moreover, in [29], *Results from 200 Billion Iris Cross-Comparisons*, and [2] Daugman shows that in reality his iris-based recognition system produced zero false matches in several million comparisons. Equally important, in [15], “Epigenetic Randomness, Complexity and Singularity of Human Iris Pattern”, Daugman proves that the iris patterns in a person's left and right eyes are dissimilar and so are the iris patterns of monozygotic (identical) twins.

[1] An electrocardiogram (ECG or EKG, abbreviated from the German Elektrokardiogramm) is a graphic produced by an electrocardiograph, which records the electrical voltage in the heart in the form of a continuous strip graph .

1.3 Background and Related Work

In the literature, a variety of iris recognition systems have been proposed for automatic human identification [1][2][6][16][17][28][33]. One of the earliest iris recognition systems was proposed by Flom and Safir [6], but never came into realization. Conversely, several well-documented systems in the open literature which came into realization are the ones suggested by Daugman [1] [2]; Zhu, Tan and Wang [16]; Wildes [17]; and most recently Miyazawa et al. [28].Yet, even though the algorithms involved in those systems differ, common to all the above systems are four processing stages:

1) Iris localization

2) Normalization of the iris to a fixed size

3) Feature extraction

4) Feature matching

1.3.1 Overview of Iris Recognition Algorithms

The iris recognition algorithms used in several systems categorized by their respective author are presented in Table 1-1:

Table 1-1 Iris Recognition: Systems and Algorithms

	Iris localization	Feature extraction	Iris code matching
Daugman [1][2]	Integro-differential operator	two-dimensional Gabor transform	Hamming distance
Wang et al. [16]	Hough transform	Gabor filters and wavelet transform	Euclidian distance
Wildes et al. [17]	Circular Hough transform	Laplacian of a Gaussian filter	Fisher discriminant analysis
Miyazawa [28]	Integro-differential operator	two-dimensional cross-phase spectrum	Phase-only correlation function

We now describe the stages in a typical iris recognition system in some detail and the usage of these algorithms and then describe the methods we selected:

Image acquisition: In the first stage, the iris of an individual is captured into a digital iris image database using a dedicated iris scan camera. This step is extremely important as the quality of the images and the spatial location of the eye greatly influence the success rate of the system. For this project, no image-capture phase is involved. Instead, a publicly available iris image database is used and its noise artefacts are explored. At present there are only a few [9] databases which have been proven to be suitable for an iris recognition system. Most notably, these are the CASIA [9] iris image database from the Chinese Academy of Sciences composed of 2400 images collected from 60 individuals, the UBIRIS [5] database composed of 1877 images collected from 241 individuals and the Michal Dobeš and Libor Machala UPOL [8] medical iris image database composed of 384 images collected from 128 individuals.

Iris localization and segmentation: In this phase, the image is segmented. An edge detector such as the Canny edge detector, which includes three [11] main steps of Gaussian filtering, non-maximum suppression and hysteresis thresholding, is applied to the image. One of the earliest and most widely used systems suggested by Daugman [1], [2] employs an integro-differential operator to precisely detect the boundaries and radius of the iris and pupil. Later, Zhu et al. [16] and Wildes[17] utilized the circular Hough transform as an alternative to Daugman's algorithm. Most iris recognition algorithms require precise iris localization for successful extraction of the area of interest which forms a flat doughnut shape.

Cartesian to polar coordinates: The region of interest (the segmented iris image) is converted into polar coordinates, transforming it into a rectangle. The most widely used transformation system is known as the rubber sheet model and was suggested by Daugman[1]. In addition, the image is normalized to a fixed width and height to compensate for pupil size, position and orientation variations in the original images. In some cases, histogram equalization is applied to the image to enhance the contrast [28].

Noise detection and removal: In some cases, the segmentation results in a noisy image. Typically, the noise is due to the presence of eyelids and eyelashes which cover part of the iris. The purpose here is to remove any noise artefacts before the generation of the iris code.

Iris code generation/encoding: Once the image has been successfully segmented, a unique iris code must be generated for identification purposes. Daugman[1] used a two-dimensional Gabor wavelet to compute an *n* byte (2048 bits) iris code with *n* = 256 bytes being the most widely used. Other methods for encoding the iris include the use of a Laplacian or a Gaussian filter [17] with four different resolutions and phase-only correlation [28].

Storage: The iris codes have to be stored in a database for future comparison.

Iris code matching: To verify the claimed identity of a human, iris codes stored in the previous phase are compared with a newly generated iris code. Daugman[1] used a *Hamming distance* to express the fraction of mismatched bits between two iris codes. Wildes [17] utilized normalized correlation for matching, and recently Miyazawa [28] utilized the peak value of a phase-only correlation function as a similarity measure between two iris images.

1.4 Method

In this work the system is designed as follows: the iris localization step consists of an automatic segmentation based on the Canny edge detector and the circular Hough transform as proposed by [17] while using only the lower half of the iris as suggested in [28]; the extracted iris region is normalized into a fixed-size rectangle by a Cartesian to polar coordinate mapping according to the rubber sheet model proposed in [1]; regarding feature extraction and matching, the phase information of a two-dimensional cross-phase spectrum is extracted as suggested in [26][27][28] and [33]. Ultimately the key phase-only correlation function is calculated as a similarity measure for iris matching, as suggested in [28][33]. To the best of our knowledge, [28] is the only work that describes a fully automatic iris recognition system based on the phase-only correlation function.

1.4.1 Realization

In this work, we embrace a top-down Digital Signal Processing (DSP) design methodology in which the various image processing and iris recognition algorithms are realized using MATLAB. As an alternative, we also evaluated several algorithms using the Insight Toolkit C++ image processing library [14]. In our experience, MATLAB is by far easier to learn and use than C/C++ and it is extremely well suited for the simulation and rapid prototyping of an image processing algorithm. On the other hand, using C++ as the programming language has the advantages of producing fast executable code, being a cross platform and hence easily portable between diverse environments including small embedded devices, thus rendering the portability restrictions obsolete. Nevertheless, we found that the use of C++ required a huge man hour investment. Hence, we abandoned the use of C++ and implemented the complete system using MATLAB, a decision which allowed concentrating on the algorithmic issues themselves rather than having to allocate numerous hours on debugging C++ code. In our report, there are several references to MATLAB files written by the author as part of the research. All MATLAB files bare the following naming convention *iris<Function Name>.m* and when referenced appear in *italic.*

Except where noted:

I. All the tables and figures in this report are based on results generated by our research and/or based on experimental evidence collected by us.

II. All the experimental results were generated by us and mostly using MATLAB.

1.5 Workflow

The rest of this report is organized as follows. Chapter 2 discusses several iris image databases and their respective noise artefacts. Chapter 3 introduces iris localization and segmentation algorithms and the utilization of the Canny edge detector and the Hough transform in an iris recognition system. Chapter 4 comprehensively discusses the concept of Cartesian to polar transformation systems, realization issues and the notion of iris normalization. Chapter 5 introduces iris code generation algorithms, including a short discussion on Gabor filters, and systematically discusses phase-based image matching and several variants of the phase-only correlation function. Finally, we discuss our results in Chapter 6 and shall draw conclusions and propose future work in the area.

CHAPTER 2
IMAGE ACQUISITION AND IRIS IMAGE DATABASES

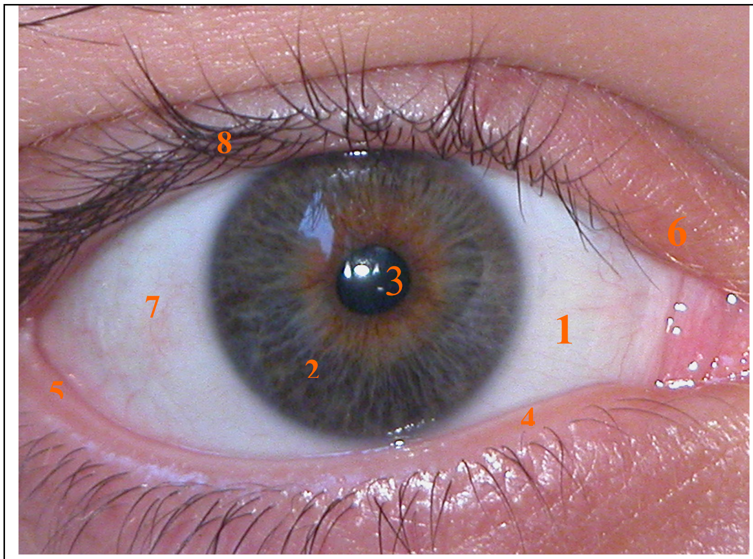
The chapter begins with a description of the gross anatomy of the human eye in section 2.1 followed by the process by which an image of the iris is captured into a digital format in section 2.2. Section 2.3 presents commonly used iris image databases and typical noise artefacts which are introduced in the process. Section 2.4 concludes the chapter with experimental results of processing common noise artefacts.

2.1 The Iris and Gross Anatomy of the Human Eye

The iris is the coloured annular section of the eye encircling the pupil [3] which begins its physiological formation during the third month of gestation, reaches an almost complete distinguishable structure by the eight month of gestation and remains stable for life after 3 years [21]. Most iris recognition systems are based on the time-invariant characteristics of iris patterns as a unique human identifier. This was first introduced in a study conducted by Flom and Safir[6] which concluded that these patterns are formed randomly and the probability of finding two individuals with identical iris patterns approaches zero. This argument was also verified by numerous other studies but most notably by an independent study conducted by Daugman [2] which gathered statistics on 9 million irides (the plural of iris)

This section presents a brief introduction to the anatomy of the eye in the context of iris recognition and heavily draws from *Principles of Anatomy and Physiology* [3, pp. 457-461] and the Association for Macular Diseases Web site [4].

A frontal view of the right eye based on image number img_81_2_1 from the UBIRIS [5] database is presented in Figure 1-1 (left), with the key parts marked on the image with their respective names (right).



1-Sclera

2-Iris

3-Pupil

4,5-Lower eyelid

6-Upper eyelid

7-Cornea

8-Eyelashes

Figure 2-1 Frontal view of the right eye

A description of each anatomical part follows:

1. The sclera (*skleros* is Latin for hard), also known as the white of the eye, is made of a tough rigid tissue which serves to protect the eye's inner parts.

2. The iris is the coloured ring-shaped section between the pupil (inner boundary) and the sclera (outer boundary). It controls the amount of light that reaches the optical nerves and does so by dilation and contraction of the pupil. In the context of biometric identification, the iris is the only information bearing part of the eye. One of the most critical algorithms in an iris recognition system is the localization of the iris.

3. The pupil is a dark circular section located at the centre of the iris which opens and closes to allow varying amounts of light to enter the eye. To correctly localize the iris, the radius and centre of the pupil must be precisely evaluated.

4-6. The lower and upper eyelids protect the eye from foreign objects and also serve to regulate excessive light from reaching the eye. The eyelids are regarded as noise as they obstruct and cover parts of the iris by forming a parabola and make the iris localization process more involved.

7. The cornea is a transparent curved coat that covers the iris and the front of the eye. Its main function is in focusing the irradiated light rays that reach the eye.

8. The eyelashes extend from each eyelid and serve as a protection mechanism against foreign objects.

2.2 Image Acquisition

In the first stage of a biometric identification process, the iris of an individual is captured into a digital iris image database. With the aid of dedicated iris scan cameras, it is possible to acquire iris images in a rapid, simple and cost-effective way. For instance, a typical 5 megapixel digital configuration which was used for the acquisition of the UBIRIS [5] iris image database [5, p. 3] is presented in Table 2-1.

Table 2-1 Iris Scan Camera Configuration Parameters

Camera Model Nikon E5700	**Software** E5700v1.0
Colour Representation RGB	**Focal Length** 71 mm
Exposure Time 1/30 sec.	**ISO Speed** ISO-200
Images Width 2560 pixels	**Height** 1704 pixels
Horizontal Resolution 300 dpi	**Vertical Resolution** 300 dpi
Bit Depth 24 bit	**Format** JPEG lossless compression

The digital iris scan camera is placed very close to the eye and uses either an auxiliary infrared [9] or flash unit [5][8] to illuminate and capture a very high-resolution digital replica of the eye. Coloured contact lenses and glasses do not restrict the acquisition process since the colour of the iris bears no entropy and in any iris recognition system the image is first converted to grey-scale.

2.2.1 Noise artefacts

The acquisition step is extremely important as several noisy artefacts which may be introduced as a result of a non-perfect process may possibly degrade the success of the subsequent steps in which the iris is segmented and extracted. To illustrate this concept we first describe the iris databases that were researched in this work and then we present several typical classes of noisy artefacts which were introduced into those iris image databases [5][8][9] using actual iris scan cameras. {**Expand per Marks request**}

2.3 Iris Image Databases

In this work, no image capture phase has been incorporated. As an alternative, several publicly available iris image databases were experimented with and used. The reason is twofold: no dedicated iris scan camera was available to the author and the variety of excellent publicly available databases for academic research reduced the time allocated for image acquisition. There are only a few database [7] which have been reported in the literature and proven to be of high quality appropriate for iris recognition system. These are the CASIA [9], which has been comprehensively studied in numerous reports [1][12][18][28], the UBIRIS [5] and the UPOL [8] iris image databases. Several, but not all, noise artefacts which characterize these databases will be introduced later in section 2.4. The virtues and shortcomings of each these databases are now discussed.

2.3.1 The UBIRIS Database

The UBIRIS [5] database, which is also known as a noisy database, had noisy artefacts introduced to it intentionally as opposed to other existing databases which contain only random noise. The authors of the database regard the other database ([8][9]) as noise free [5, p. 1] even though, as we shall shortly see, they do contain several random noise artefacts. Nonetheless, it has the advantage of being the only noisy database amongst the publicly available databases and currently is also the largest publicly available iris database. The database is composed of 1877 images collected from 241 individuals in two separate sessions in which the first does not contain noise artefacts and the second contains noise artefacts such as focus, reflection and luminosity problems.

2.3.2 The CASIA Database

The CASIA database is one of the most referenced and most widely used in the open literature and was collected by the Chinese Academy of Sciences Institute of Automation. Version 1.0 consists of 756 iris images collected from 108 individuals in different sessions.

A variety of iris recognition systems based their research and experimental results on this database [1][12][18][28]. Two of its main merits are (1) the images were obtained using a near infrared camera, a process which completely avoids noise artefacts in the form of specular reflections on the pupil which are common in the UPOL database, and (2) the pupil in all the images was manually darkened and marked to ease the segmentation processing step.

2.3.3 The UPOL Database

The UPOL [8] iris image database consists of 3 x 128 texture-rich iris images (i.e., 3 x 64 left and 3 x 64 right) which have 24-bit depth in RGB mode with a resolution of 576 x 768 pixels. The file format used was PNG and the irides were scanned by a TOPCON TRC50IA optical device connected with a SONY DXC-950P 3CCD camera. The main purpose of this database, according to the researchers at the Laboratory of Image Analysis of Human Eye Iris and Retina [8], is to conduct medical research for early glaucoma diagnostics using a *fundus* camera, which is a specialized microscope connected to a camera. Amongst the three databases reported here, the UPOL seems to be the least studied as there are almost no reports in the open literature specifying the use of this database and it is also the only database which uses a microscope for the acquisition of the iris images.

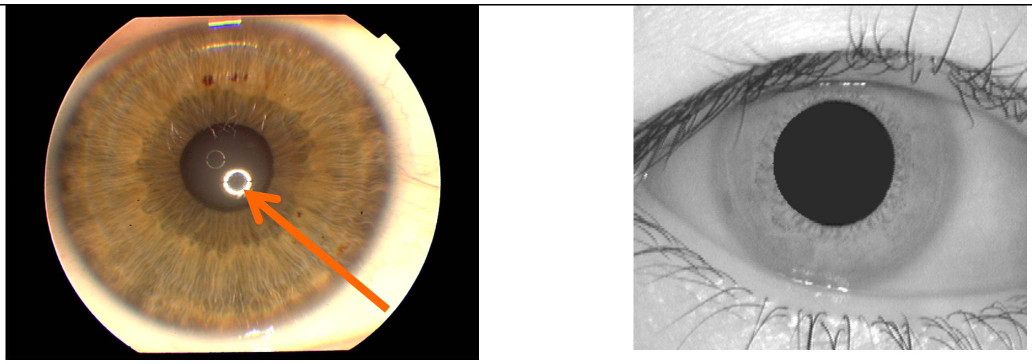
Because phase-based image matching had not been previously applied on this database, and because there are not many reports in the open literature using this database it was chosen to be the database on which the research will be conducted.

2.4 Experimental Results with the Processing of Noise Artefacts

2.4.1 Noise Generated by Specular Reflection

When using a digital camera, as in the case of using a traditional film camera, the iris scan camera unit may or may not be equipped with a flash unit. In the former case, the use of a flash unit may possibly bring in superfluous noisy artefacts such as red-eye and specular reflections which were not part of the original scene [10]. The latter case may not be at all possible since it would mandate the use of very long exposure times in which the iris may move rapidly and result in a blurred image. However, when creating the CASIA [9] iris image database, the researchers utilized a camera which is sensitive to near-infrared light (the wavelength range for infrared light is from about 1 mm to 750 nm.), hence rendering the flash unit obsolete.

Figure 2-1 presents the formation (a) using image number 049R_2 from the UPOL [8] database or the absence (b) using image number 105_2_4 from the CASIA [9] database of noise generated by *specular reflection*. The surface of the eye can be regarded as a *specular surface*. When light from an electrical flash unit hits a specular surface it will be reflected at an angle equal to the angle of the incident light angle (a direct consequence of Snell's law).



(a) Specular reflections form bright highlights on the surface of the pupil. The camera used is sensitive to the visible light band only.

(b) No specular reflections are evident when using a near-infrared camera as no flash unit is utilized.

Figure 2-2 Noise generated by specular reflection

The problem with using a flash unit is presented in Figure 2-3 using (1) image number 049R_2 from the UPOL [8] database with several processed versions (2,3,4) on which typical (the Canny edge detector and the Hough transform will be discussed in detail in Chapter 3.) image preprocessing algorithms were applied. As can be observed (bottom right), when the image is processed with a Canny edge detector the reflections form a circular shape. The problem is that it may be erroneously identified by a circle detection algorithm (such as the Hough transform) as the actual circle bounding the pupil. This fault will of course result in the erroneous detection of the radius of the annular section of the iris, which will in turn lead to the generation of a wrong iris code.

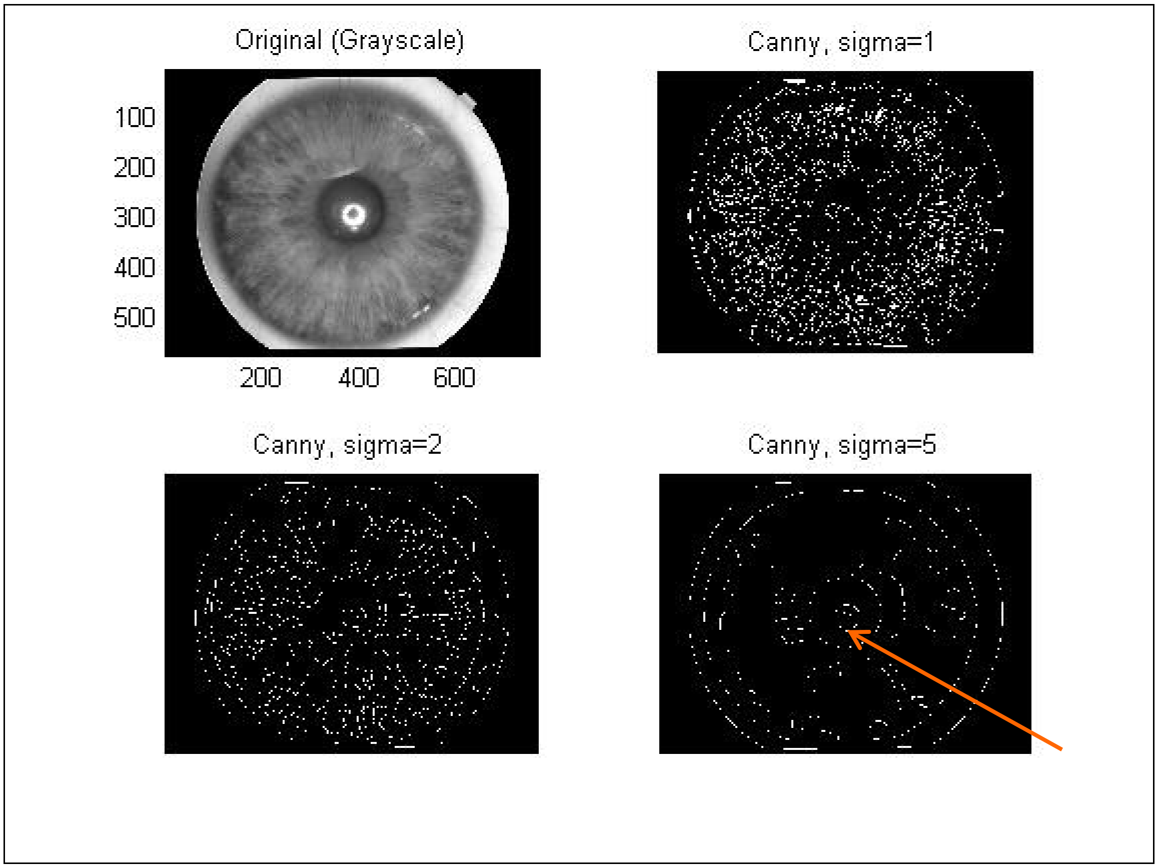


Figure 2-3 False circle edge detection due to specular reflections

This type of noise does not exist in either the UBIRIS [5] or the CASIA [9] databases and is common in the UPOL [8] database. Correspondence with the authors of the UPOL database sheds some light on the nature of the problem:

2.4.2 Noise Generated by Eyelashes

The noise generated by the eyelashes based on (a) image number 004_2_3 from the CASIA [9] database and (b) image number Img_106_1_3 from the UBIRIS [5] database is presented in Figure 2-4. The upper eyelashes extend from the eyelids and occlude parts of the iris (more notable on the left image). This type of noise must be removed by a pre-processing step. This type of noise artefact does not exist in the UPOL [8] database.

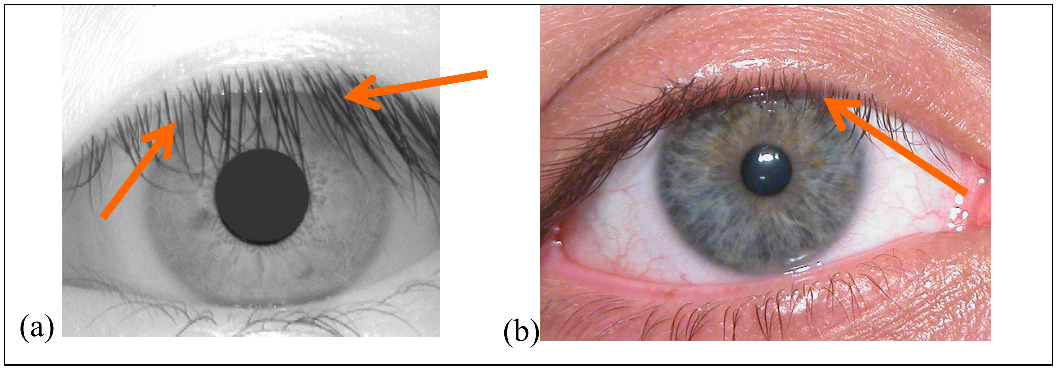


Figure 2-4 Noise generated by eyelashes

2.4.3 Noise Generated by Eyelids

The noise generated by the eyelids based on (a) image number 046_1_1 from the CASIA [9] database and (b) image number Img_32_1_4 from the UBIRIS [5] database is presented in Figure 2-5. The upper eyelids, which form a shape of a parabola, occlude parts of the iris (more notable on the right image). This type of noise, as in the case of noise generated by the eyelashes, must be removed by a pre-processing step. This type of noise does not exist in the UPOL [8] database.

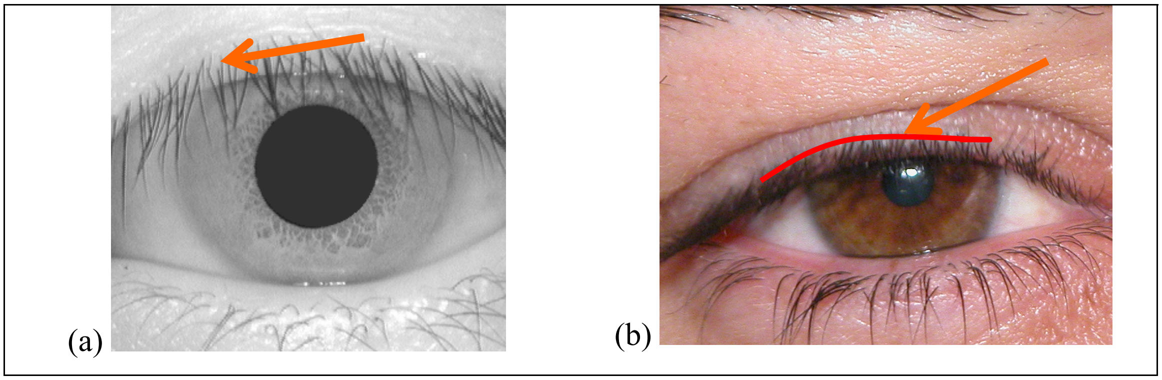
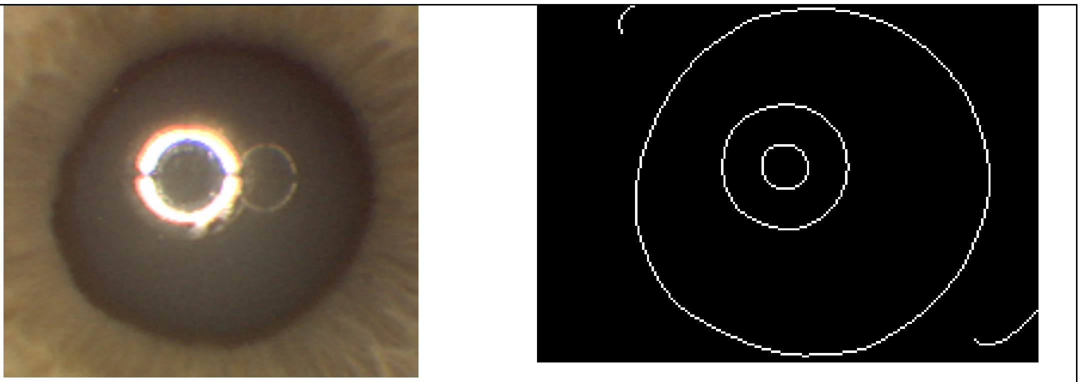


Figure 2-5 Noise generated by eyelids

2.4.4 Noise Generated by Skewed Pupil or Iris

When the pupil does not form a perfect circle, e.g., when it is skewed, the circle-detecting algorithms will detect a circle as expected, but due to the skewness some parts of the pupil may lie outside or inside the detected circle. The unperfected circular shape of the pupil based on (a) a zoomed version of image number 027L_2 from the UPOL [8] database and (b) the edges detected by the Canny edge detector inside the pupil is presented in Figure 2-6. Note how the edge-mapped image (b) is skewed towards the lower left. The case of a skewed iris resembles the case of the skewed pupil.



(a) Zoomed pupil (*Not* an iris)

(b) Edge-mapped image

Figure 2-6 Noise generated by skewed pupil or iris

CHAPTER 3
IRIS LOCALIZATION, ISOLATION AND SEGMENTATION

This chapter starts with a short introduction in section 3.1 followed by a discussion on the Canny edge detector in section 3.2. Section 3.3 presents first-order derivative masks and section 3.4 thoroughly discusses Gaussian smoothing and Gaussian derivative kernels, which are crucial to the fast and correct operation of the Canny edge detector. The section also addresses state-of-the-art algorithms for the computer generation of Gaussian kernels. Section 3.5 addresses more aspects of the Canny edge detector and section 3.6 presents the empirical results for finding the optimal variance for the Gaussian smoothing kernel on the UPOL database and the actual segmentation success rates. Section 3.7 introduces the Hough transform and section 3.8 concludes the chapter.

3.1 Introduction

After the proper acquisition of an iris image, the next phase is to localize the pupil and the iris. To accomplish this goal, several image processing algorithms are utilized. The image is first segmented by applying an edge detector such as the Canny edge detector. Then, for computational purposes, both the inner and the outer boundaries of the iris can be approximated by a circle; thus the boundaries and radius of the iris and pupil are located by one of the circular edge-detection algorithms such as the circular Hough transform [5] or the Daugman integro-differential operator [1]. Following a successful segmentation using the preceding steps, the iris is extracted as explained in Chapter 4.

3.2 The Canny Edge-Detection Algorithm

In his well-known paper arising from his masters degree at MIT, Canny [11] proposed an optimal edge detector, which is regarded as one of the best and most widely used edge detectors. The Canny edge detector was built on the earlier work of Marr and Hildreth [34], who conducted research on modelling the human visual perception and is based on a set of three different optimality criteria. In his paper Canny asserts that an optimal filter that meets the criteria can be approximated by the first derivative of a two-dimensional Gaussian function in the spatial domain [11]. The Canny edge detection algorithm consists of a chain of four key operations which are applied in succession:

I. Blurring by a convolution with a two-dimensional Gaussian filter

II. Differentiation of the blurred image in both the x- and y-axis and calculation of the gradient magnitude

III. Application of non-maxima suppression in a direction perpendicular to an edge

IV. Edge linking by double thresholding (also known as *hysteresis thresholding*)

We start with a description of the derivative operator with respect to edge detection in images in section 3.3 as a basis for blurring by a convolution and the concepts introduced later. We do not however discuss non-maxima suppression and edge linking by double thresholding, topics which are extensively discussed in the open literature.

3.3 Two-Dimensional Approximations to First-Order Derivative Masks

The gradient or the derivative method detects edges by looking for the maximum and minimum in the first derivative of the image. Given an image represented by a continuous two-dimensional function $f(x,y)$, the gradient of $f(x,y)$ is defined as

$$\nabla f(x,y) = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right] \qquad \textbf{3.1}$$

where $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$ are the partial derivatives in the x- and y-axis respectively and (x,y) is a two-dimensional pixel point. The gradient $\nabla f(x,y)$ points to a path of maximal change of the function $f(x,y)$. Hence the gradient of an image is based on the correct computation of the *partial derivatives* at each pixel location. In the digital domain, the first-order differential would have to be *approximated* (more on this issue later). The two directional derivatives can also be written separately in each axis as

$$\begin{aligned} \nabla f_x(x,y) &= \frac{\partial f}{\partial x} \\ \nabla f_y(x,y) &= \frac{\partial f}{\partial y} \end{aligned} \qquad \textbf{3.2}$$

For the purpose of edge detection we are mostly interested in the gradient magnitude (also known as the *strength of the edge*), defined as:

$$\|\nabla f(x,y)\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \qquad \textbf{3.3}$$

The result of the equation is in a floating point representation. However, in practice it is converted (rounded off) to an integral representation in a range suitable for grey-map images. For computational efficiency, common practice is to approximate the gradient with absolute values instead of using a square root as in:

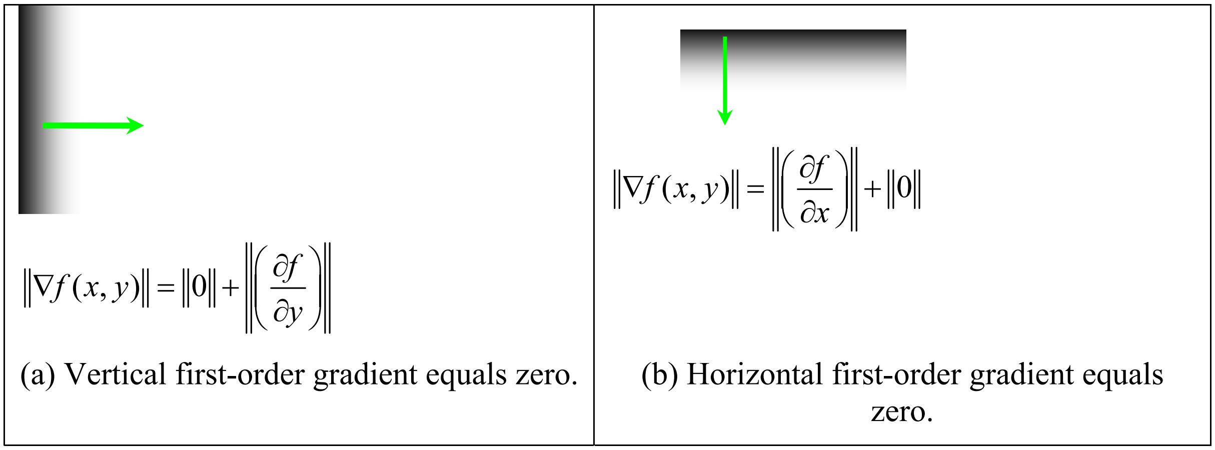
$$\|\nabla f(x,y)\| = \left\|\left(\frac{\partial f}{\partial x}\right)\right\| + \left\|\left(\frac{\partial f}{\partial y}\right)\right\| \qquad \textbf{3.4}$$

Note, however, that the result is not isotropic and hence the approximation was not used. The gradient vector direction or orientation is defined as:

$$\phi(x,y) = \tan^{-1}\left(\frac{\left(\frac{\partial f}{\partial y}\right)}{\left(\frac{\partial f}{\partial x}\right)}\right) \qquad \textbf{3.5}$$

A high gradient value will be observed when there is a fast alteration in the image brightness levels, which means there is either a transition between a dim region and a bright region or a transition between a bright region and a dim region. Thus, an edge can be detected whenever there is a rapid change occurring. It can be measured by calculating the derivative, which for the largest abrupt change will have its maximum magnitude or, equally, would have a zero second derivative.

This concept is presented in Figure 3-1 by showing that when there is a change in the horizontal direction (a) only the gradient in the *y*-axis direction, $\left(\frac{\partial f}{\partial y}\right)$, is larger than zero and consistently, in the vertical direction (b) only the gradient in the *x-axis direction*, $\left(\frac{\partial f}{\partial x}\right)$, is larger than zero.



$$\|\nabla f(x,y)\| = \|0\| + \left\|\left(\frac{\partial f}{\partial y}\right)\right\|$$

(a) Vertical first-order gradient equals zero.

$$\|\nabla f(x,y)\| = \left\|\left(\frac{\partial f}{\partial x}\right)\right\| + \|0\|$$

(b) Horizontal first-order gradient equals zero.

Figure 3-1 Horizontal and vertical first-order gradients

This representation falls well within definitions in the analogue domain. However, when processing images on a computer, a discrete *digital* domain representation is called for and it can be obtained by approximating or sampling the analogue domain.

Assume now that the function $f(x,y)$ representing the image is discrete. The derivatives of the image can be calculated by convolving the image with a discrete filter that approximates the analogue derivative. A typical approximation to the analogue derivatives is given by

$$\begin{aligned}\Delta_x &= I(x,y) - I(x-1,y)\\ \Delta_y &= I(x,y) - I(x,y-1)\end{aligned} \qquad \textbf{3.6}$$

However, equation 3.6 does not calculate the gradient at point (x,y) but rather at point $(x-.5, y-.5)$ and is anti-symmetric. A better alternative is presented by equation 3.7, which is symmetric but introduces a delay to the computation

$$\begin{aligned}\Delta_x &= I(x+1,y) - I(x-1,y)\\ \Delta_y &= I(x,y+1) - I(x,y-1)\end{aligned} \qquad \textbf{3.7}$$

The intuition behind equation 3.7 is that a horizontal edge can be found by calculating the difference between one pixel brightness and the adjacent (+1) pixel brightness either climbing up or down from the first. This operation is equivalent to convolving the image with the convolution masks of the form:

$$h_{horizontal} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \qquad \textbf{3.8}$$

$$h_{vertical} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}^{Transpose} \qquad \textbf{3.9}$$

Typical first-derivative masks are the Sobel, Prewitt and Roberts operators.

3.4 Gaussian Filtering

The first-order derivative filter masks presented in the previous section are susceptible to noise. The edge filters act as high pass-filters that not only amplify spatial high frequencies such as edges but also boost the noise level. It is possible to first limit the rate at which brightness intensities change by a smoothing operation which attenuates high spatial frequencies. By far the most popular smoothing operation in computer vision is a convolution with a two-dimensional Gaussian kernel mask [30]. This section discusses the

Gaussian function and the Gaussian derivative and illustrates a number of methods that can be applied to generate discrete Gaussian kernels on a machine.

3.4.1 The Continuous Gaussian Function

Denote the parameter $\sigma > 0$ as the *standard deviation* (STDV, Standard deviation is a statistical measure of the variability around the mean) where $\sigma^2 = \frac{\sum (x - \overline{x})^2}{n-1}$ is the *Variance* defined as the average of the square of the distance of each value from the mean and $\overline{x}$ is the average of all the values; then the continuous one-dimensional Gaussian function is defined as

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \qquad \textbf{3.10}$$

The two-dimensional Gaussian function is defined as

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad \textbf{3.11}$$

Some of the unique properties and merits of the Gaussian function $G(x, y)$ are as follows:

I. It is symmetric, rotationally invariant and emphasizes nearby pixels over distant ones

II. It never crosses zero and hence does not introduce new zeros when used in edge detection

III. It acts as a *low pass-filter* that removes very high frequencies from the image due to noise

IV. Its Fourier transform is again a Gaussian and, most importantly,

V. It is a separable function

Some of these properties will be discussed in the subsequent sections.

3.4.2 Experimental results, discretization of the Gaussian Function

In this section we present our own experimental results with the discretization of the Gaussian function. Theoretically, the continuous Gaussian kernel is non-zero everywhere, which necessitates a substantially large convolution kernel. Nevertheless, in reality it is in effect zero more than about four standard deviations from the mean and hence can be truncated. To use the filter in a digital domain, it must be sampled and quantized (i.e., a process of discretization). When deciding on the digital filter kernel size it is usually sufficient to use a filter which extends to *three standard deviations* on each side around its mean. This guarantees an error of less than *0.00135* on each tail of the Gaussian distribution. Thus, for an odd-length filter, the kernel size can be evaluated as:

$$n_{order} = 2*\,(3*\sigma)\,+1 \qquad \textbf{3.12}$$

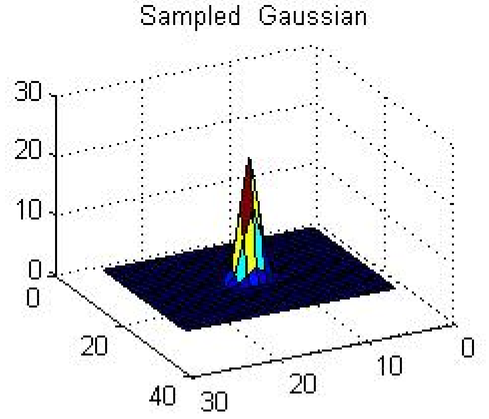
$$\text{(Error less than 0.00135)}$$

Possibly the easiest solution to the discretization of the Gaussian function is by direct uniform sampling. A MATLAB script was written to generate and experiment with a two-dimensional Gaussian kernel of varying STDV and varying kernel size using uniform sampling. The values of such a kernel with a unit STDV and a kernel size of 5 are presented in Figure 3-2(a):

$(a)\begin{bmatrix} 0.0751 & 0.1238 & 0.0751 \\ 0.1238 & 0.2042 & 0.1238 \\ 0.0751 & 0.1238 & 0.0751 \end{bmatrix}$	$(b)\ (342)^{-1}*\begin{pmatrix} 1 & 5 & 8 & 5 & 1 \\ 5 & 20 & 33 & 20 & 5 \\ 8 & 33 & 54 & 33 & 8 \\ 5 & 20 & 33 & 20 & 5 \\ 1 & 5 & 8 & 5 & 1 \end{pmatrix}$

Figure 3-2 Floating Point Valued Gaussian Kernel of Order 3 and Unit Variance and an Integer Valued Gaussian Kernel Convolution Mask of Order 5 and Unit Variance

This floating point kernel can be converted into an integral representation (Table 3-1(b)) using only integers which can be beneficial on a non-floating-point CPU. The process starts by finding the reciprocal of the smallest value (in this case 1 / .002969 = 333), multiplying the matrix by that value and rounding off to the nearest integer. The number 342 is the sum of all the elements in the matrix and serves as a normalization factor. In addition, the numbers can be rounded to the nearest integral power of 2, allowing the smoothing operation to be executed *using only shift operations,* which is an extremely fast operation on most CPUs. A 3D stem plot of a digital two-dimensional Gaussian kernel equivalent of a continuous Gaussian function is presented in Figure 3-3.



A digital sampled Gaussian with kernel of order 7 and STDV of 1. The larger the kernel, the closer the digital kernel to its analogue equivalent.

Figure 3-3 3D plot of an isotropic sampled digital two-dimensional Gaussian kernel

3.4.3 Efficient Separable Representation

In this section we present our own experimental results with the efficient mathematical representation of two-dimensional convolution. Two-dimensional Gaussian blur operations are used in a wide range of image-processing systems and for large kernel size the filtering execution time is intolerable [30]. The function $G(x,y)$, however, is distinctive in the sense that it is a rotationally symmetric or *isotropic* function that can be separated by dimension; i.e., it is *separable* since it can be decomposed into a product of two one-dimensional Gaussian filters, one depending only on the variable x and the other depending only on y:

$$G(x,y) = \frac{1}{2\pi\sigma^2}(e^{-\frac{x^2}{2\sigma^2}})*(e^{-\frac{y^2}{2\sigma^2}}) = G(x)*G(y) \qquad \textbf{3.13}$$

This result is of great significance as the two-dimensional smoothing or blurring operation which is a convolution in a two-dimensional space can be performed by the successive application of two one-dimensional Gaussian convolutions. Mathematically a $n*n$ kernel is split into two kernels of $n*1$ and $1*n$, respectively. Thus, the number of multiplications necessary is reduced from

$$\Theta(n*n) \qquad \textbf{3.14}$$

to

$$\Theta(2*n) \qquad \textbf{3.15}$$

The last equation considerably decreases the computational complexity especially for large kernels. For instance, if *n=20*, the separable implementation requires only 40 multiplications instead of 400 with an approximately tenfold increase in computational efficiency. In [30],Waltz shows that by separating a Gaussian blur into row and column operations, an increase in computational efficiency on the order of ten can be observed. His findings are in accordance with the theoretical results of equation 3.15 and our own empirical results of an experiment of 100 runs are presented in Table 3-1 (MATLAB: *irisSeparableConvolution.m*):

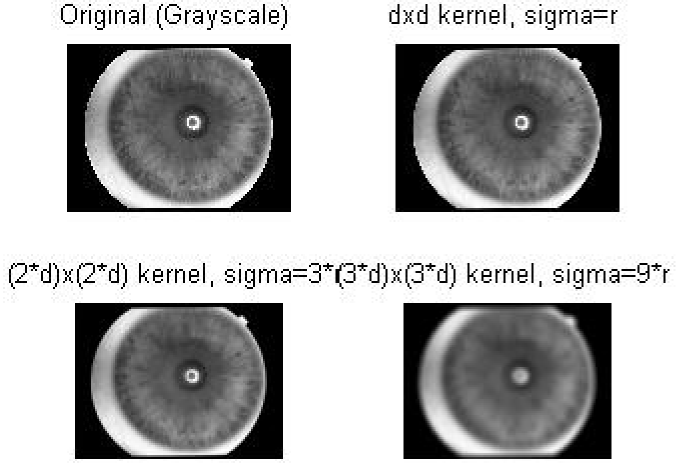
Table 3-1 Experimental results, run Time Comparison between Separable and Non-separable two-dimensional Convolution Demonstrating the Superiority of Separable Convolution

Kernel size/Sigma	Separable		Non-separable
1,5	0.1090		0.6720
3,20	0.1870		0.6250
5,40	**0.3430**	**much smaller than**	**2.8750**
5,80	**0.7030**	**much smaller than**	**12.3910**

Equally important, a distinctive feature of a Gaussian kernel called the *semi-group* property is that convolving a Gaussian with a second Gaussian produces another Gaussian with STDV that is given by:

$$\sigma_{new-Gaussian} = \sqrt{(\sigma_1{}^2 + \sigma_2{}^2)} \qquad \textbf{3.16}$$

This is a significant result since convolution is an expensive operation, especially when large kernels are involved and with this result larger Gaussian kernels can be constructed by repeated application of *smaller* kernels, which is a much more computationally efficient operation [30]. This result could also be used for blurring at different scales (the STDV σ of the Gaussian derivative function, which is also known as *scale*) as locating the pupil, for instance, would need a kernel of $m*m$ while locating the larger iris would need a kernel of $n*n$ where $m << n$. Our own experimental result of the blurring effect with varying values of STDV, based on iris image number 027L_2 from the UPOL [8] database, is presented in Figure 3-4.



For a constant kernel size, the larger the variance (bottom right) the larger the blurring effect observed (MATLAB: *irisGaussianBlur.m*)

Figure 3-4 Experimental results of Gaussian blur with different STDV applied to an iris image

3.4.4 Discrete derivative of a Gaussian function

The digital finite difference filter which estimates an analogue derivative does not produce satisfactory results. This is because a finite difference filter has an abrupt response to rapid changes due to *noise*, in addition to those due to the image signal itself. A first-order derivative edge detector that is less prone to noise is the *derivative of a Gaussian* (GD[2]). The derivative of a Gaussian edge detector performs Gaussian smoothing before application of the derivative. Both operations of Gaussian filtering and differentiation can be performed by convolving with a one-dimensional mask of the form presented in Figure 3-5 and mathematically given by:

$$\frac{\partial G(x)}{\partial x} = \frac{\partial}{\partial x}\left(\frac{x}{\sqrt{2\pi}\sigma}\right)e^{-\frac{x^2}{2\sigma^2}} = -\frac{x}{\sqrt{2\pi}\sigma^3}e^{-\frac{x^2}{2\sigma^2}} \qquad \textbf{3.17}$$

[2] We use the acronym *GD* in order to differentiate between its close relative *the difference of a Gaussian (DOG).*

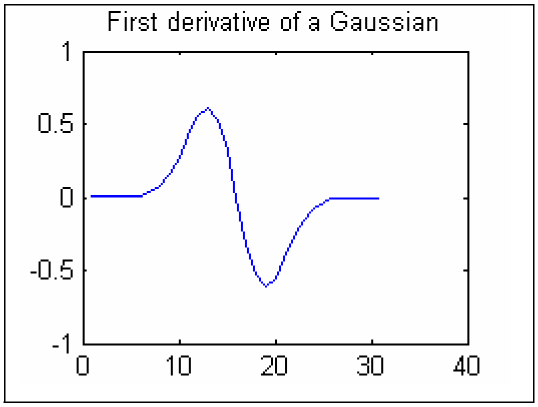


Figure 3-5 Plot of a first derivative of a 1D Gaussian

And similarly a two-dimensional mask:

$$\frac{\partial G(x,y)}{\partial x} = -\frac{x}{\sigma^2}\left[\frac{1}{2\pi\sigma^2}e^{\frac{-(x^2+y^2)}{2\sigma^2}}\right] \qquad \textbf{3.18}$$

(A directional derivative of a two-dimensional Gaussian along the *x*-axis)

$$\frac{\partial G(x,y)}{\partial y} = -\frac{y}{\sigma^2}\left[\frac{1}{2\pi\sigma^2}e^{\frac{-(x^2+y^2)}{2\sigma^2}}\right] \qquad \textbf{3.19}$$

(A directional derivative of a two-dimensional Gaussian along the *y*-axis)

Our experimental results of the cross section of the GD edge operator with a kernel size of 7 x 7 (left) and a kernel size of 19 x 19 (right) are presented in Figure 3-6. The wider the function the wider the edges detected. A narrow function will detect sharp edges and more detail:

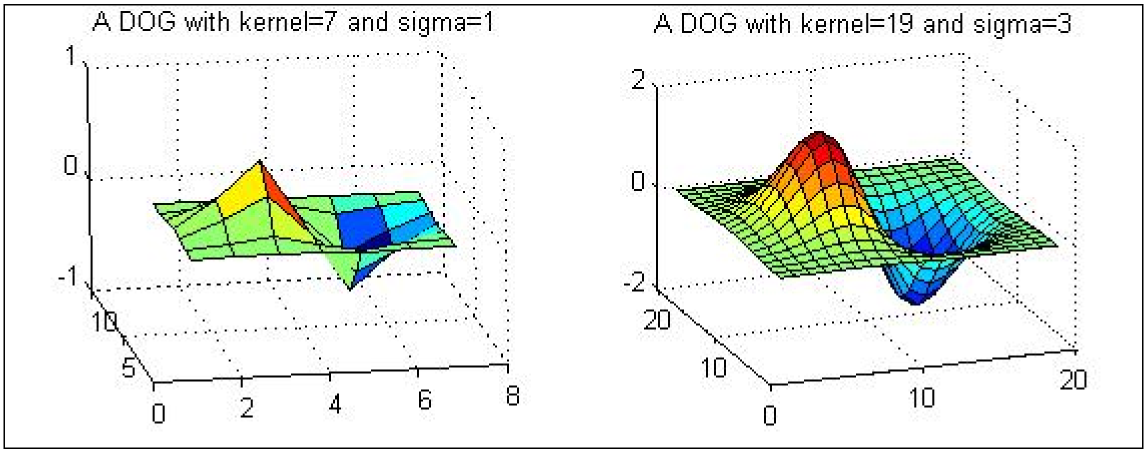


Figure 3-6 Experimental results, 3D surface plot of a derivative of a Gaussian with different-sized kernels

A very important point to note is that *each* of the two two-dimensional directional derivatives (matrix form) has to be convolved with an image to obtain the norm of the gradient. However, using the separability property derived earlier we can perform two one-dimensional convolutions instead of one costly two-dimensional convolution. That is, the first 1D derivative of a Gaussian function convolved with a 1D Gaussian blur function convolved with the image function is equivalent to the image function convolved with the first derivative of a two-dimensional directional Gaussian blur function. The assertion is expressed mathematically in equations (3.20) and (3.21) for the directional derivative in the *x*-axis and the *y*-axis, respectively:

$$(\mathrm{I} \oplus \frac{\partial G^{2D}(x,y)}{\partial x}) = (\left[\mathrm{I} \oplus \frac{\partial G^{1D}(x)}{\partial x} \right] \oplus G^{1D}(y)) \qquad \mathbf{3.20}$$

$$(\mathrm{I} \oplus \frac{\partial G^{2D}(x,y)}{\partial y}) = (\left[\mathrm{I} \oplus \frac{\partial G^{1D}(y)}{\partial y} \right] \oplus G^{1D}(x)) \qquad \mathbf{3.21}$$

where *I* is the image, *G* is the Gaussian mask and the operator $\oplus$ indicates the convolution operation. The computational savings are of the order mentioned for the same analysis in the case of an ordinary two-dimensional Gaussian.

3.4.5 Machine Generation of Discrete Gaussian and GD Kernels

Methods for the generation of Gaussian kernels include approximation by a repeated convolution of a binomial in a manner similar to the generation of Pascal triangle coefficients [11], approximation by a two-sided recursive *infinite impulse response* (IIR) filter [11] and by means of fast convolution using the *fast Fourier transform* (FFT). Methods for the generation of a GD kernel include analytic derivation using Hermite polynomials, discretization using Deriche recurrence, discretization of the heat equation and finally by an FFT. In this section we review the state of the art in the machine generation of discrete Gaussian and GD kernels.

Generation Using Hermite Polynomials

In [31], "A Gaussian Derivative Based Transform", Bloom and Reed assert that the n^{th} GD can be expressed analytically as the product of a *Hermite polynomial* and the original Gaussian. Hence, the family of 1D Gaussians can be constructed recursively using [19]:

$$\frac{\partial^n G(x)}{\partial x} = \frac{\partial^n}{\partial x}\left(\frac{x}{\sqrt{2\pi\sigma}}\right)e^{-\frac{x^2}{2\sigma^2}} = h(n)G(x) \qquad \mathbf{3.22}$$

where the function $h(n)$ is the n^{th}-order Hermite polynomial.

This method of Gaussian kernel generation is realized in practice, for instance, in the VIGRA[19] C++ image processing library as follows:

$$\begin{aligned} &h^{(0)}(x) = 1 \\ &h^{(1)}(x) = \text{-}x / s \wedge 2 \\ &. \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathbf{3.23} \\ &h^{(n+1)}(x) = \text{-}1/s \wedge 2 * [\, x * \ h^{(n)}(x) + n * h^{(n\text{-}1)}(x)] \end{aligned}$$

where the result of the derivative at recurrence level (n) is used to express the derivative at recurrence level (n+1). Naturally, an analytically derived first-order GD convolution mask using this technique is superior to the one obtained by sampling a Gaussian and then approximating differentiation by a first-order digital filter such as the Sobel operator.

Generation Using Deriche Recurrence

Another algorithm for Gaussian derivative convolutions was developed by Deriche [22] in his report *Recursively Implementing the Gaussian and Its Derivatives*. Deriche presented an efficient recursive approximation to the sampled Gaussian given in the form of 2nd order IIR filters. The error measure [pp. 12, 22] used was the mean square error (MSE) and experimental results proved that the error is on the order of 10^-4.

Generation Using Discretization of the Diffusion Equation

Another technique was presented in [23] by Lindeberg, titled *Scale Spaced Approach in Computer Vision,* who concentrated on the direct discretization of the *diffusion equation,* also known as the *heat equation*. The implementation in the ITK image processing library follows this route.

Generation Using a Fourier Domain Transformation

The kernel generation can be computed in the Fourier domain since a convolution in the temporal domain reduces to a product in the Fourier domain. The Fourier transform of a derivative is the Fourier transform of the function multiplied by the factor $(j\omega)$:

$$\mathbb{F}\left\{\frac{\partial G}{\partial x}\right\} = j\omega_x \mathbb{F}\left\{G\right\}, \mathbb{F}\left\{\frac{\partial^n G}{\partial x^n}\right\} = (i\omega_x)^n \mathbb{F}\left\{G\right\} \qquad \textbf{3.24}$$

where $\hat{F}\{\ \}$ denotes the Fourier transform. Hence the Gaussian derivative of an image in the Fourier domain becomes:

$$\mathbb{F}\left\{I \oplus G_x\right\} = \mathbb{F}\left\{I\right\} \cdot (i\omega)^n \mathbb{F}\left\{G_\omega\right\} \qquad \textbf{3.25}$$

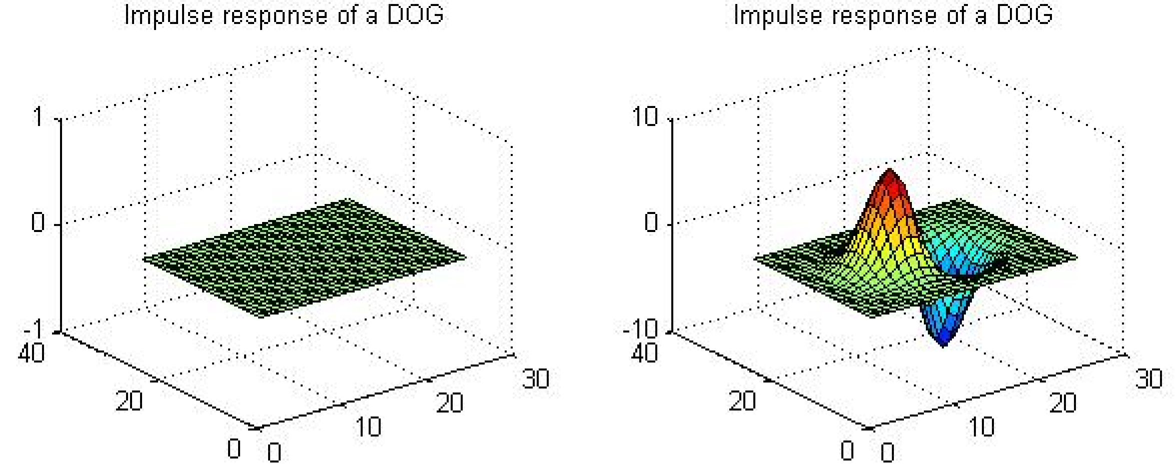
It is possible to go back to the time domain using an inverse Fourier transform as in:

$$I \oplus G_x = \mathbb{F}^{-1}\left\{\ \mathbb{F}\left\{L\right\}\ \cdot (i\omega)^n \mathbb{F}\left\{G_\omega\right\}\ \right\} \qquad \textbf{3.26}$$

where the operator $\oplus$ indicates the convolution operation.

3.4.6 Impulse Response of a Derivative of a Gaussian

The effect of applying a GD filter can be studied by observing its so-called *impulse response*. For this, we create in MATLAB a synthetic image in which only the central pixel has a non-zero value and convolve it with a GD mask. Our experimental results with the impulse response with a zero-value central pixel (upper left) and central pixel of value 5 (upper right) are presented in Figure 3-7. Note that in the case of zero-value central pixels the impulse response is flat whereas in the other case it grows considerably with the growth of the central pixel.



(a) No edge detected

(b) Even a small variation in brightness is detected by a first-order derivative.

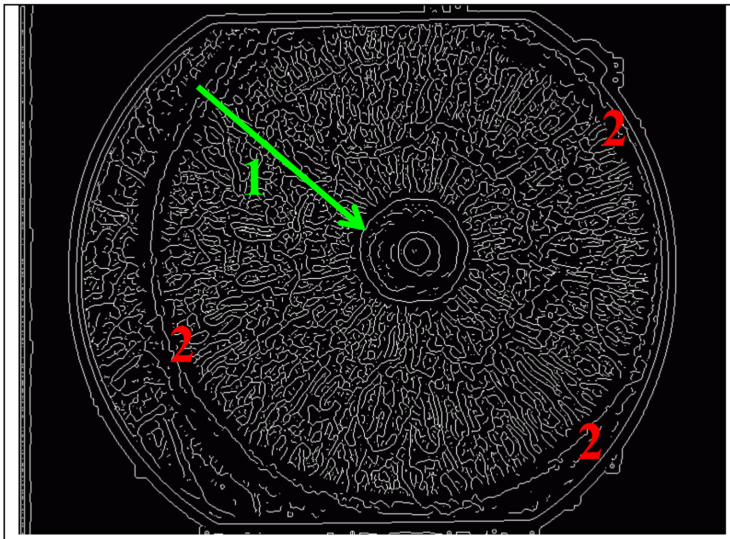
Figure 3-7 8 Experimental results, Impulse response of a GD

3.5 Experimental Results, the canny edge detector

This section presents our experimental results with the canny edge detector. The correct segmentation of the pupil and the iris depends heavily on the selection of the standard deviation of the Gaussian smoothing filter used in the Canny edge detector. A small variance means that smaller and finer segments would be identified, although it may sometimes identify false edges. In contrast, larger Gaussian kernels would be less sensitive to noise but would have a much higher run-time overhead. Furthermore, it must be underlined that as the radius of the pupil and iris differ considerably and due to inter-iris differences between humans, the actual value of the STDV must be determined empirically. And yet, one must note that even after finding those parameters they are of little value to a system which incorporates a *different* iris image database, as those parameters (radius, STDV) vary considerably between iris image databases. On top of that, an empirically determined STDV found using a first derivative of a Gaussian method will not yield the same segmentation results with a second derivative of a Gaussian method which is used in some systems that employ the Canny edge detector. Having said that, we now present the experimental results of applying the Canny edge detector to several iris images from the UPOL database using MATLAB and the ITK image processing library.

3.5.1 Empirical Determination of the Gaussian Blur Standard Deviation

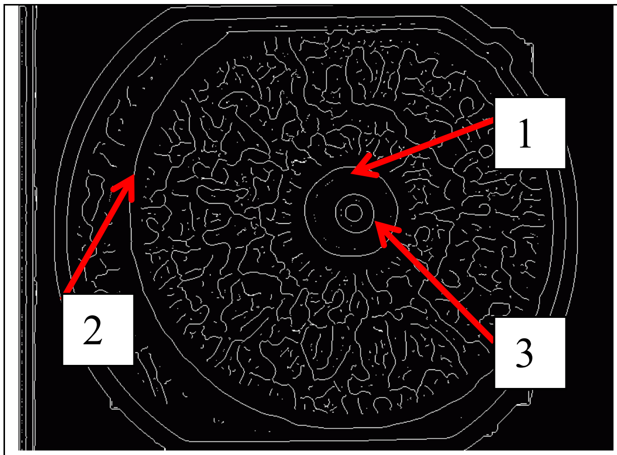
An edged-mapped iris image number 045R_3 from the UPOL database processed by the Canny edge detector class of the ITK library with a variance of 5.0 is presented in Figure 3-8.



Many noise artefacts exist when a small variance is used for the Gaussian smoothing operation. Note that due to the small variance used, (1) noise artefacts in the form of a semi-circle exist inside the circle bounding the pupil and (2) noise in the form of small edges contaminate the boundary of the iris bounding circle.

Figure 3-9 10 Experimental results, edge-mapped iris with a small variance

Image number 045R_3 was further processed by the Canny detector, but with a variance of 30.0. Our experimental result is presented in Figure 3-9. Note that due to the large variance used, the noise artefacts in the form of a semi-circle are absent, and noise in the form of small edges no longer contaminates the boundary of the iris-bounding circle.



The number of noise artefacts decreases substantially when a large variance is used for the Gaussian smoothing operation. Note how well defined (1) the circles bounding the pupil and (2) the iris are. Note also that (3) there are two circles inside the pupil which are considered as noise and might be falsely identified as the circle bounding the pupil. Yet, using a larger kernel means longer run-time overhead.

Figure 3-11 Experimental results, edge-mapped iris with a large variance

Furthermore, the circle bounding the pupil and the outer circle bounding the iris are clearly marked on the edge-mapped image. The correct segmentation of the pupil and iris is crucial for the success of the circle-detecting algorithms introduced later.

3.5.2 Experimental results, Segmentation Success Rates

We conducted several experiments on a subset of 48 iris images (known as the *iris9_16* set) from the UPOL database to obtain statistics on the number of successfully segmented images using different values for the STDV, where success is measured by the correct segmentation of the circles bounding the pupil and the iris. As not all images were successfully segmented, we tested the effect of applying *histogram equalization* on the binary image *before* the application of the Canny edge detector in an attempt to enhance the contrast and improve the results, as suggested in [25]. Our experimental results of the segmentation *failure* rates are presented in Table 3-2.

Table 3-2 Experimental results, the Effect of Histogram Equalization on Segmentation Failure Rates

Segmented shape	With equalization	No equalization
Pupil	34/48 **(70%)**	16/48 **(33%)**
Iris	2/48 **(4.16%)**	2/48 **(4.16%)**

As can be observed, utilizing the UPOL database the application of histogram equalization for pupil detection results in a *lower* success rate. On the other hand there is no effect on the iris boundary detection. The result was verified and found to be consistent in other subsets of the database (such as *iris57_64*) and can be explained by the fact that the specular reflections present to a large extent in the centre of the pupil have an adverse effect on the spread and range of the pixels around the pupil, making the detection of edges by the Canny edge detector when using histogram equalization more complex. Still it is worthwhile to note that there is a relatively high failure rate of 33% in the detection of the edges of the pupil due to the existence of noise factors such as specular reflections and due to lack of variation in contrast between the pupil and its surrounding pixels.

The selection of the STDV used is also crucial for the correct detection of the pupil and even a difference in the magnitude of 1 in the STDV can lead to failure. This result is presented in Figure 3-9, where a STDV of (a) 3 and (b) 5 was used for the Gaussian blur operation. The image number 014L_3 is from the *iris9_16* subset and while the detection fails with SDTV of 5, it succeeds with an STDV of 3. Of course it is not possible in real-time to know which images would react better to one or another STDV and hence we opt for the STDV with the *lowest average* failure rate.

It is important to note that in the case of several iris image databases, such as the CASIA, the pupil was marked manually and is the largest and darkest area in the image and it can be separated from the surrounding iris by simple thresholding. Conversely, the UPOL database is substantially noisier in this sense due to a high number of illumination artefacts and specular reflections which are scattered on the pupil, making the precise iris localization much more difficult. Thus, experiments conducted with the CASIA in the open literature report almost perfect segmentation rates.

Note that this discussion does not refer to the detection of the circles, which is the subject of our next section. Rather, it only refers to the correct marking of the edges of the circles by the Canny edge detector.

3.6 The Hough Transform

The Hough transform is an algorithm for discovering objects of arbitrary shape within a pre-processed edge-mapped image. In [20], Hough introduced and patented an efficient method for finding straight lines in a binary image. Its core operation is a transformation between a Cartesian space in which the line is defined and a parameter space (also know as *Hough space)* in which the Hough transform of the line (or an arbitrary shape as we shall see) is defined.

3.6.1 The Line-Detecting Hough Transform

We wish to detect a set of points laying on a straight line. Given the points (x_j, y_j) and (x_i, y_i), one of the simplest forms of expressing a line that passes through them is the slope-intercept form:

$$y = ax + b \qquad \textbf{3.27}$$

This *line* which is defined on the xy space (Figure 3-10 left) has two parameters which define a *point* in the two-dimensional ab space (Figure 3-9 right). Observe the point (x_i, y_i), which has an infinite number of straight lines passing through it but all of them are bound to the constraint that $y_i = ax_i + b$ or

$$b = -ax_i + y_i \qquad \textbf{3.28}$$

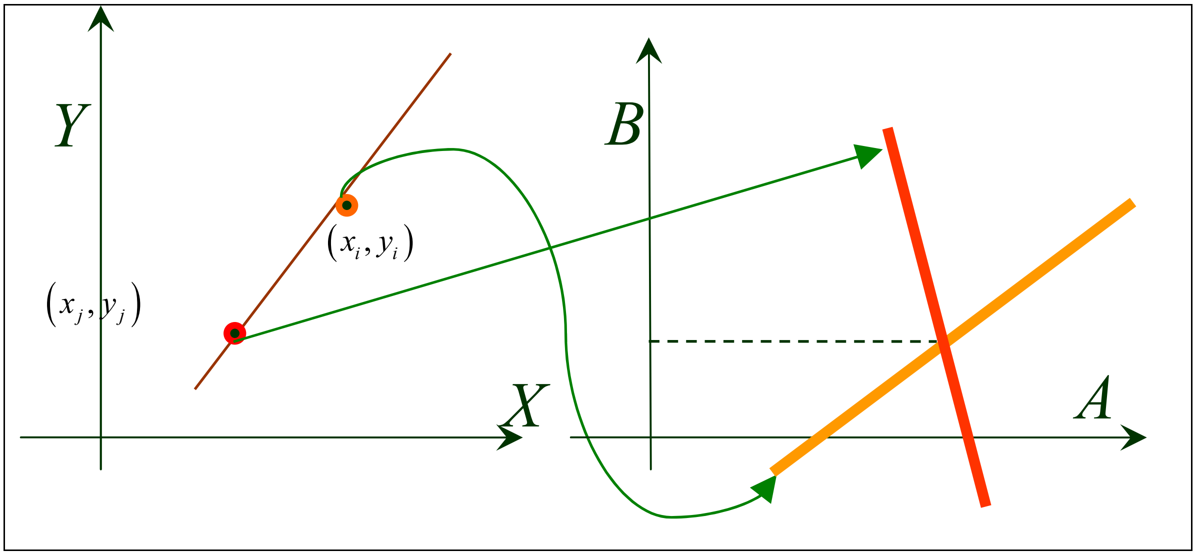


Figure 3-12 Transformation of a point into a line in the parametric *ab* space

The meaning of equation 3.28 is that every *point* in the xy space corresponds to a *line* in the ab parametric space. Likewise, two or more *points* (Figure 3-10) on a *line* in the xy space will define two or more *lines* in the ab space which will intersect on *a single point* that represents the line on which those two points are in the xy space. This assertion can be better exemplified using a MATLAB plot. In Figure 3-11 we have the two points (7,-2) and (3,2) on a xy space coordinate system (Figure 3-11 left) with the corresponding representation of the points using equation 3.28 (Figure 3-11 right) on the ab space.

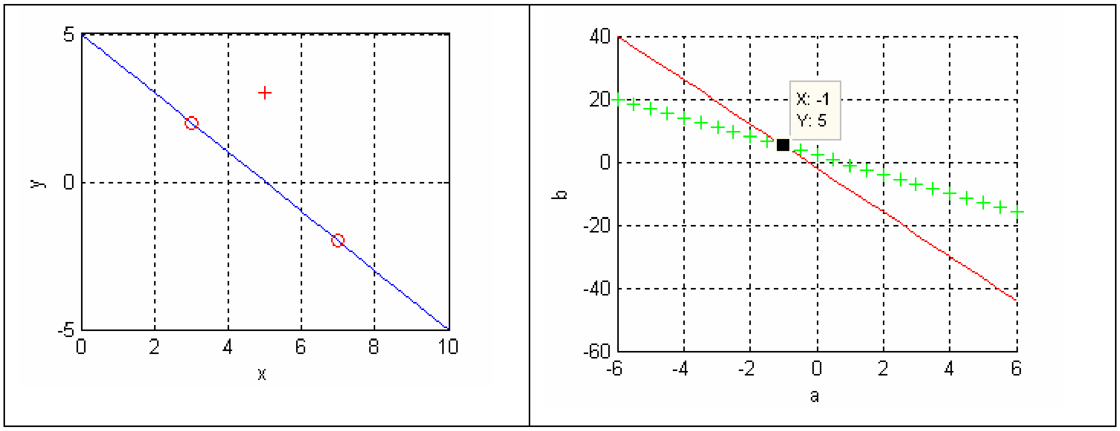


Figure 3-13 Two points in the Cartesian space transform into two lines in parametric space

The line is defined by the points (left) given by $y = -1 * x + 5$ and these same parameters, (-1, 5), are the intersection point of the lines in the *ab* space. This gives rise to a method called a *voting system*, which finds the point with the highest number of intersections, hence giving the shape with the highest probability of being a line. The voting system is discussed next.

3.6.2 Accumulators and the Voting System

One of the most important stages in the Hough transform is a voting system. By letting each point in the xy space cast a vote in the ab space for each possible line passing through it, we save the votes into an accumulation array for each corresponding point. That is, the Hough transform is a voting process where each pixel of the image votes for all potential lines passing through that point. Refer to Figure 3-12 and, following our previous example, assume for instance that we have the three points (7,-2), (3, 2) and (5,0) on a two-dimensional Cartesian coordinate system (left) with the corresponding representation of the points using equation 3.28 (right).

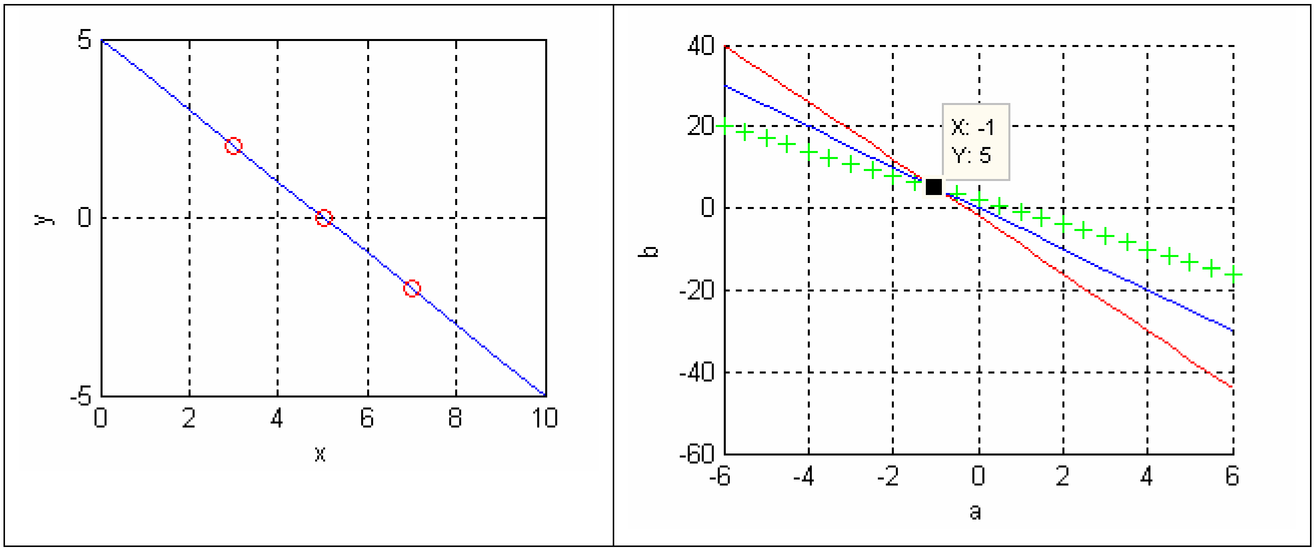


Figure 3-14 The Hough transform accumulation system

As can be observed in Figure 3-12, the three points on the line which is given by $y = -1 * x + 5$ (Figure 3-12 left) generate three lines or *votes* which coincide on a single point (-1,5) in the *ab* space (Figure 3-12 right). These votes can be accumulated in an accumulator array $H(a,b)$, which is also known as the Hough space; for every point *(x, y)*, each element that lies on the corresponding point in the *(a, b)* accumulator array can be incremented. Members are incremented in the accumulator array when lines pass through multiple points. At the end of the accumulation process, the cell that has the maximum value corresponds to the line that passes through the most number of points in the source image.

3.6.3 Parametric Coordinates

The major drawback of the slope-intercept form of a line, presented in the previous section, is that vertical lines cannot be represented since $a \to \infty$ and for horizontal lines $a = 0$. In order to avoid these singularities the line can be transformed to a parametric presentation, a transformation which is given by the following equation

$$x\cos\theta + y\sin\theta = r \qquad \textbf{3.29}$$

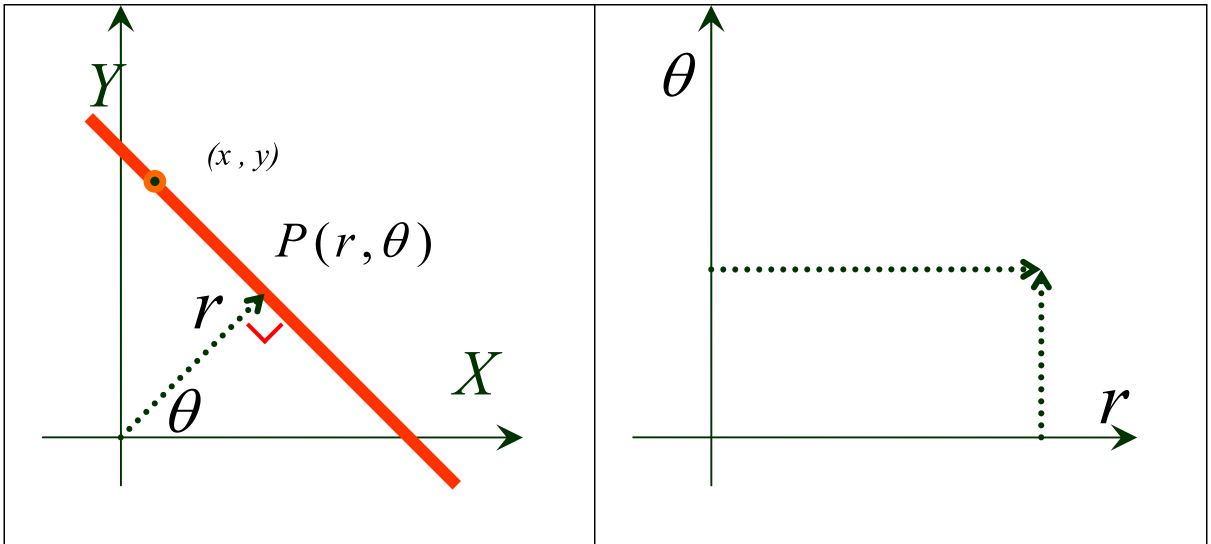
where the angle θ is bounded by $[0, 2*\pi]$ and made by the normal r to the straight line with the positive *x*-direction and r is the vertical distance of the straight line from the origin. For any given line, θ and r are constants. The geometrical interpretation is that the pair (r, θ) corresponds to a *straight line* in the (x, y) plane:

$$y = \frac{r}{\sin\theta} - x\cot\theta \qquad \textbf{3.30}$$

And that the pair (x, y) corresponds to a *sinusoidal wave* in the (r, θ) plane:

$$\begin{aligned} r &= \rho\left(\frac{x}{\rho}\cos\theta + \frac{y}{\rho}\sin\theta\right) \\ \rho &= \sqrt{x^2 + y^2} \\ r &= \rho\left(\cos\alpha\cos\theta + \sin\alpha\sin\theta\right) \\ r &= \rho\cos\left(\theta - \alpha\right) \end{aligned} \qquad \textbf{3.31}$$

The transformation between the image plane and the parametric space is known as the Hough transform. This concept is presented in Figure 3-13, where a *line* (a) in the Cartesian (x, y) coordinates plane (left) is transformed into a *point* (b) in the parametric (r, θ) coordinates plane (right).

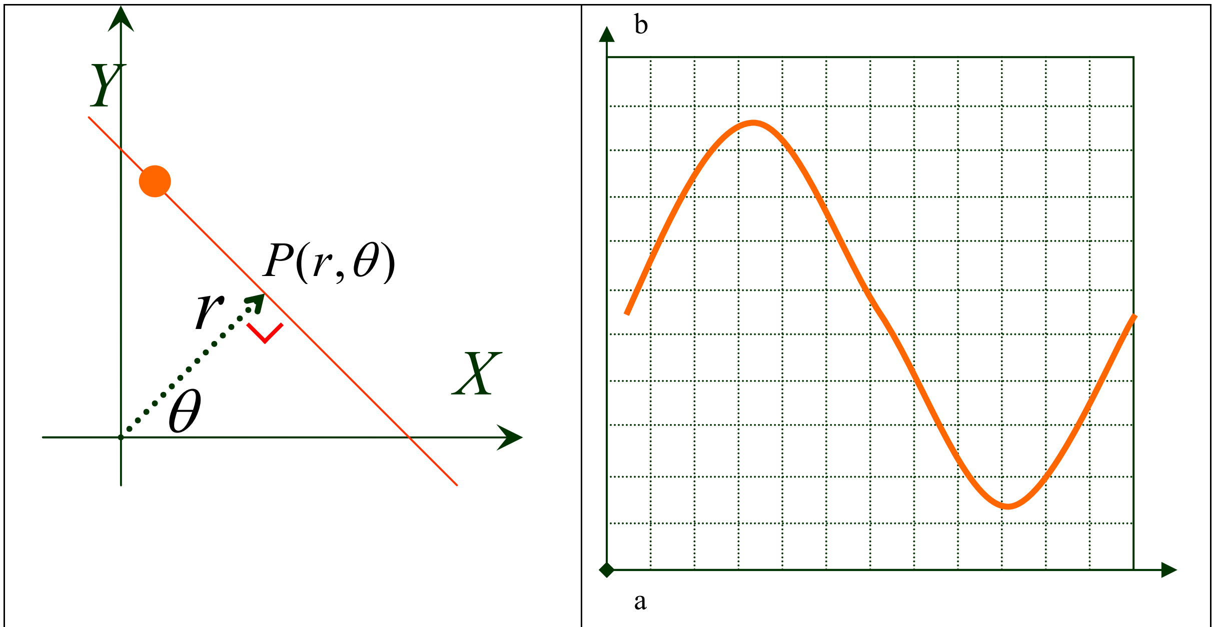


(a) A line in a two-dimensional Cartesian space. One of many possible lines through *(x, y)*.

(b) The Hough transform of the line is a point in the parametric space.

Figure 3-15 Relationship between Cartesian line and parametric point

The Hough transform of a point in the image plane is a sinusoidal curve in the Hough space (Figure 3-13).



(a) A point *(x, y)* in a two-dimensional Cartesian space.

(b) The Hough transform of a point is a sine wave in the parametric space.

Figure 3-16 Relationship between a Cartesian point and parametric sine wave

3.6.4 The Circular Hough Transform

The line detection method can be extended to the detection of other shapes such as circles. In iris recognition systems the *circular Hough transform* and the Daugman [1] integro-differential operator are the most widely used circle detection algorithms. The iris localization step, which consists of an automatic segmentation based on the Hough transform, was proposed in [17] by Wildes and was successfully used in numerous other reports. We now present the algorithmic steps of the circular Hough transform. Let us assume that (1) we are given a pre-processed edge-mapped image with clearly marked edges as a result of applying the Canny edge detector and that (2) we want to find circles of known radius in this image. The Hough transform can be used to determine the parameters of a circle when a number of points that fall on the edge are known. Just as a line can be expressed parametrically, so can a circle. A circle with radius R and centre (a,b) can be constructed with a parametric equation as follows:

$$\begin{aligned} x &= a + R\ \cos(\phi) \\ y &= b + R\ \sin(\phi) \end{aligned} \qquad \textbf{3.32}$$

If we let the angle ϕ pass through the full $2*\pi$ degree range, then the points $(x,\ y)$ trace the outer limits of a circle. Equation 3.32 has three parameters: a, b, r. A typical edge mapped iris image contains countless points, some of which lay on the perimeters the pupil and iris. The purpose of the circular Hough transform is to discover these parameters triplets a, b, r in order to fully describe each circle. In a similar fashion to the linear Hough transform, the curvature obtained in the Hough space for each edge mapped pixel is a right circular cone. Because we have not realized the Circular Hough transform in our system, we do not pursue this topic any further. For more information on the circular Hough transform please refer to [20][25].

CHAPTER 4
COORDINATE TRANSFORMATION AND IRIS NORMALIZATION

This chapter presents a short overview of the normalization process in section 4.1 followed by a systematic development of the Cartesian to polar transformation system, ring extraction, implementation details and experimental results in section 4.2. Section 4.3 concludes the chapter with iris image normalization, quantization and optimal resolution determination.

4.1 Overview

Upon the successful determination of the centre and radius of the pupil, an annular ring such as an iris area of fixed radius has to be extracted. To accomplish that, we use a Cartesian to polar coordinate transformation. We base our discussion on two relaxed assumptions: (a) the circles bounding the pupil and the iris were successfully marked using the Canny edge detector (b) The radii and centre of these circles were perfectly located using the circular Hough transform. As found earlier we know that the segmentation process can have a failure rate of almost 33%, so the former assumption is not fully correct and the latter assumption holds true only in ideal lab conditions. With these assumptions, the next step is to transform the doughnut-shaped ring into a rectangle using a polar/Cartesian transformation and normalize it so that rings with different outer and inner radii will have the same transformed rectangular size.

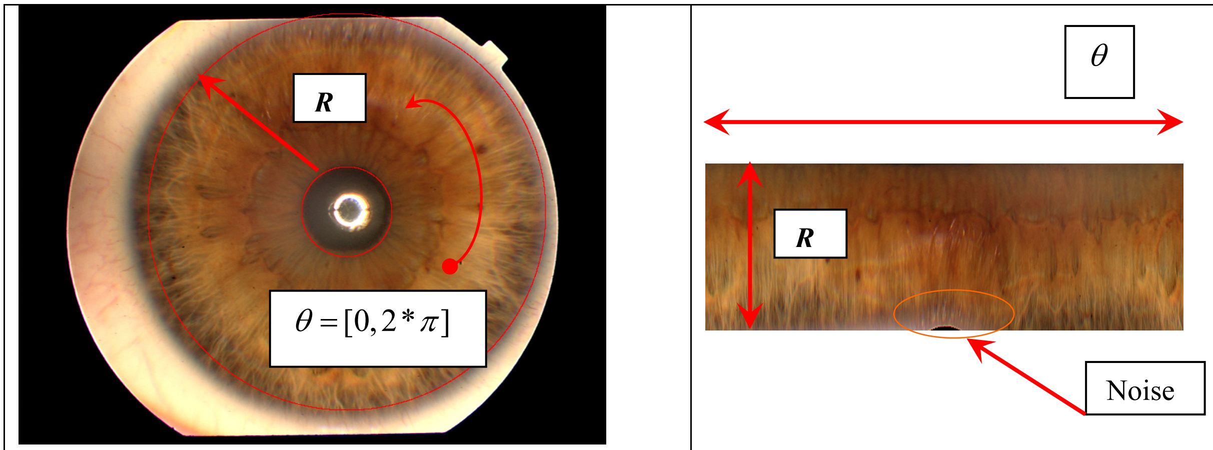
4.2 Cartesian to Polar Coordinate Transformation

We demonstrate the concept and the shortcomings of Cartesian to polar coordinate transformation in the context of iris recognition by manually detecting the radius of the circles around the pupil and the iris. That is, we manually segmented the iris rather than using a fully automatic system. We use the simplifying assumption that the pupil and the iris are *concentric*[3] but as we shall see, this assumption leads to errors since in several cases it is *not* true. A *circle* with radius cOuterRadius and centre (a,b) can be constructed using a polar coordinate system as follows:

$$\begin{aligned} x &= a + \text{cOuterRadius}*\cos(\phi) \\ y &= b + \text{cOuterRadius}*\sin(\phi) \end{aligned} \qquad \textbf{4.1}$$

where the angle θ is bounded by $[0, 2*\pi]$ and the points $(x,\, y)$ trace the pixels of a circle with radius (cOuterRadius).

[3] Concentric circles share the same centre but have different radius.



(a) Manually marked circles on a Cartesian coordinate system for image number 045R_3. The centre, outer radius and inner radius are given by [-17 45], 260 and 59 pixels, respectively.

(b) The resulting transformed rectangle in the polar coordinate system. *R* is equal to 201 = (260 - 59). θ = 1635 pixels and was calculated using equation 4.3.

Figure 4-1 Transformation of an iris ring from Cartesian to polar coordinate system

However, we are interested in constructing a *ring* and not a circle. Consequently, instead of having a constant radius (cOuterRadius) we let the angle (ϕ) pass through the full 0 to $2*\pi$ degree range. The interval or bin between each adjacent angle is given by:

$$(\Delta\phi) = 1/(2*\pi*(\text{cOuterRadius})) \tag{4.2}$$

The transformation algorithm is as follows: For each fixed value of the angle (ϕ) we iterate through all pixels with a radius starting with the inner radius (cInnerRadius), a step size of $(\Delta\phi)$ and ending with the outer radius (cOuterRadius), thus exhaustively spanning all pixels of the ring. Each such pixel is copied to the corresponding two-dimensional pixel location in the polar coordinate system. Hence, each circular ring of the iris is a row of the new polar transformed image.

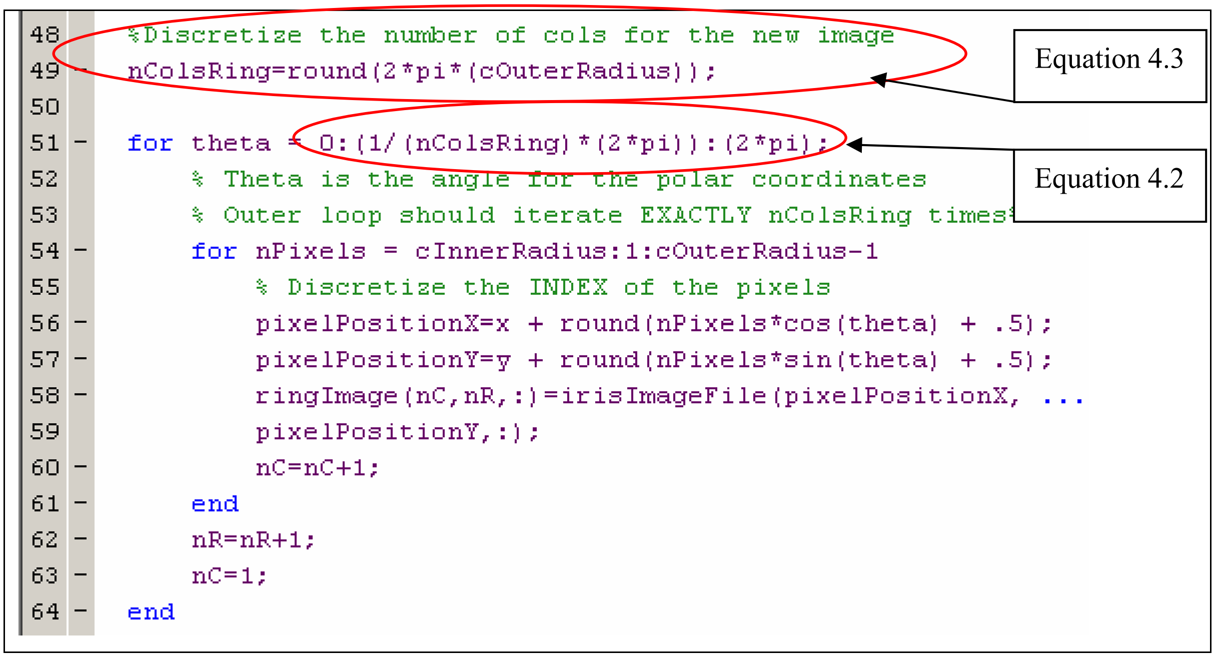
The transformation system is presented in Figure 4-1, where the ring with an outer perimeter of $2*\pi*(\text{cOuterRadius})$, inner perimeter of $2*\pi*(\text{cInnerRadius})$ and radius of $\text{R=(cOuterRadius-cInnerRadius)}$ is transformed into a rectangle with R rows and of $2*\pi*(\text{cOuterRadius})$ columns. For this particular case the centre, outer radius and inner radius are given by [-17 45], 260 and 59 pixels, respectively, and the circles almost coincide.

4.2.1 Implementation of Image Index Quantization

The value of the number of columns $\text{nColsRing=}2*\pi*(\text{cOuterRadius})$ of the transformed image involves multiplication by the irrational number π and hence results in a *floating* *point* number representation. Conversely, pixels must be accessed using a *discrete* index number representation and hence this value is quantized according to:

$$\text{nColsRing}=\lceil 2*\pi*(\text{cOuterRadius})\rceil \qquad \textbf{4.3}$$

where the operator $\lceil \text{floating point number} \rceil$ rounds down a floating point number to the nearest integral pixel index. A small code snipped (*irisGetAnnularSection.m*) from the actual algorithm realization in MATLAB further exemplifies this concept and is presented in Figure 4-2.



```
%Discretize the number of cols for the new image
nColsRing=round(2*pi*(cOuterRadius));

for theta = 0:(1/(nColsRing)*(2*pi)):(2*pi);
    % Theta is the angle for the polar coordinates
    % Outer loop should iterate EXACTLY nColsRing times
    for nPixels = cInnerRadius:1:cOuterRadius-1
        % Discretize the INDEX of the pixels
        pixelPositionX=x + round(nPixels*cos(theta) + .5);
        pixelPositionY=y + round(nPixels*sin(theta) + .5);
        ringImage(nC,nR,:)=irisImageFile(pixelPositionX, ...
        pixelPositionY,:);
        nC=nC+1;
    end
    nR=nR+1;
    nC=1;
end
```

Figure 4-2 Algorithmic realization of the Cartesian to polar coordinate system for a ring

Moreover, the angle ϕ is also quantized into bins according to:

$$(\Delta\phi)_{quqntized} = \lceil 1/(2*\pi*(\text{cOuterRadius})) \rceil \qquad \textbf{4.4}$$

Our implementation allows for the easy modification of the number of bins to suit the needs of other iris image databases.

4.2.2 Experimental Results

We found that the assumption that both the pupil and the iris are concentric was violated in several images in the UPOL database. Failure to take into account different centres for the pupil and the iris results in the incorrect transformation of the ring to polar coordinates and the introduction of noise as sections inside the pupil or outside the iris may be included in the transformed image. In the example depicted in Figure 4-1, based on image 045R_3, the circles are almost perfectly concentric. In contrast, Figure 4-3, based on image 015L_2, presents a different scenario of non-coinciding circles. In the latter case, the transformed image will introduce non-iris regions that corrupt the normalized representation. To actually avoid this type of noise the algorithm must take into account a ring with non-uniform radius while still under the constraint that the result of the transformation is a rectangle. To realize this constraint, we later modify our algorithm to utilize a rule of using *n* equally spaced points on the radius normal, irrespective of the actual radius length. Yet, we manually selected images which are almost coinciding to minimize this noise artefact.

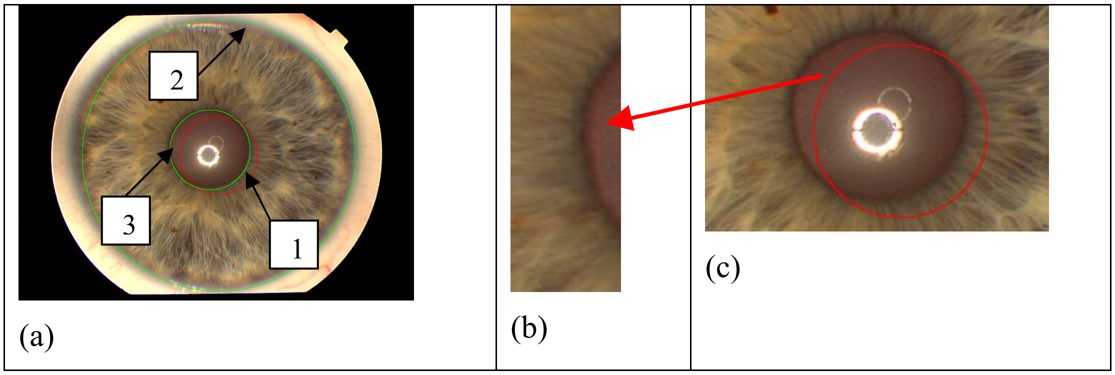


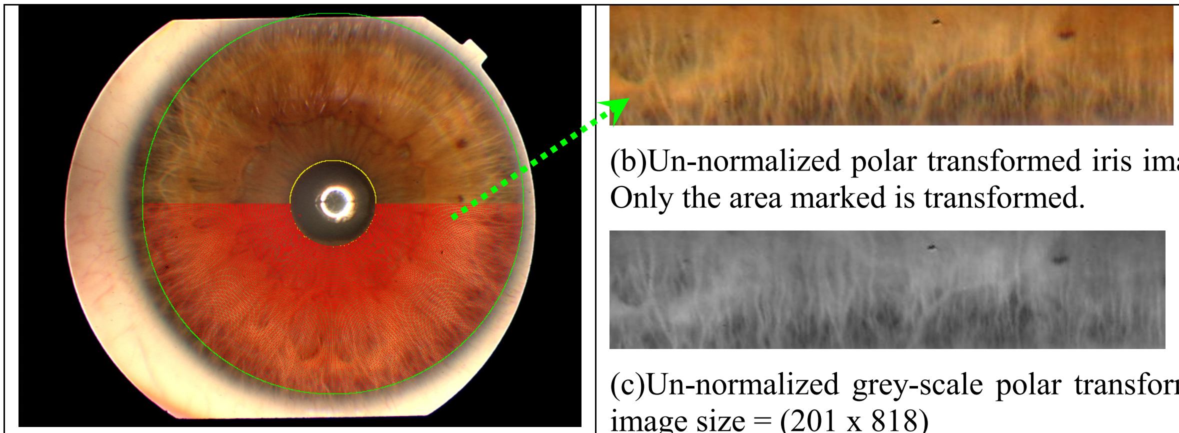
Figure 4-3 Non-concentric circles lead to noise being transformed

In Figure 4-3(a), the inner circle (1), which has the same centre as the outer circle (2), does not form a perfect dissection of the iris. Perfect dissection would have been achieved if an inner circle (3) with the non-coinciding centre had been used instead.

4.3 Iris Normalization

At this stage the polar transformed image is normalized to a fixed width and height to compensate for pupil size, position and orientation variations in the original images. We need only use the *lower half* of the iris region, as processing it will have smaller run time and memory footprints and we empirically found that it would still maintain enough entropy for iris encoding. In [28], the lower half of iris is extracted in order to avoid noise artefacts present in the upper half of the iris.

Evaluate the results obtained in Figure 4-1 depicting the transformation of the complete ring to the results obtained in Figure 4-4 illustrating the transformation of the lower half only. The latter result does not contain the noise artefact (refer to Figure 4-1; a small black spot is marked as noise) while still maintaining enough pixels for iris encoding. Moreover, the image has a resolution of (201*818) pixels, which is half the size of the original image.



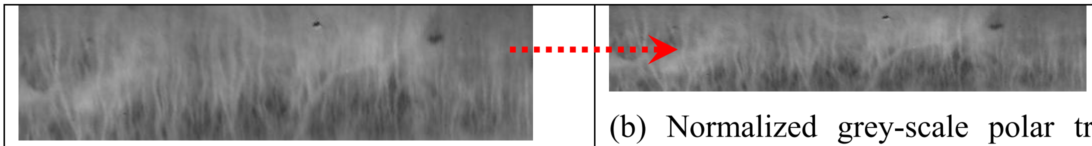
(a) Original iris image with programmatically marked area of interest.

(b)Un-normalized polar transformed iris image. Only the area marked is transformed.

(c)Un-normalized grey-scale polar transformed image size = (201 x 818)

Figure 4-4 Example of actual transformation of the lower-half iris section and selection of the area of interest

At this stage the polar transformed image is also converted to grey-scale format as the colour of the iris bears no entropy [2]. The crucial part of the normalization process is achieved by refining the algorithm presented earlier to use a constant number of quantized index points. That is, the angle is circumferentially quantized into a fixed number of bins and the radial extent of the annulus is radially sampled at a fixed number of steps regardless of the dilation or contraction of the iris due to variations in pupil size. This concept, which is similarly used in [12] and [28], is presented in Figure 4-5.



(a) Un-normalized grey-scale polar transformed image (201 x 818)

(b) Normalized grey-scale polar transformed image (128 x 512)

Figure 4-5 Examples of actual normalization using a constant number of quantized index points

4.3.1 Minimal Sampling Requirements

The original iris images in the UPOL database have a constant resolution of 768 x 576 pixels and the normalization process transforms the iris to a constant width and height regardless of the actual area of the iris section. As expected, the resolution must be determined in a manner that does not lose iris information. That is, we do not wish to degrade the ability of the normalized image to generate a unique iris code. Regarding the determination of the *minimum* number of pixels needed for encoding the iris, we consulted

Professor Monro of the Iris Recognition Group at the University of Bath [D. M. Monro, personal email communication, July 11, 2006], and based our results on [32].

According to Monro's extensive study [32] conducted on the CASIA database even if we maintain 99% of the iris image power radially and more circumferentially it would not be sufficient to successfully encode the iris. Furthermore, he asserts that at least 99.99% of the iris image power must be maintained by using 171 Fourier coefficients circumferentially and 16 radially. He concludes that a constant number of 80 rows and 512 columns would be a suitable choice for the normalized polar transformed iris image. For practical reasons of efficient sub-sampling we opt for an integral power of 2 and use 128 pixels radially and 512 pixels circumferentially, as recommended in [32], for our initial experiments.

Equally important is the empirical determination of the optimal radial and circumferential resolutions with the optimality criteria being maximal phase-only correlation value. This issue is dealt with in section 5.4 of chapter 5.

CHAPTER 5
FEATURE EXTRACTION AND IRIS ENCODING

This chapter starts with a short overview of iris code generation in section 5.1, followed by iris encoding using Gabor wavelets in appendix A. Section 5.2 presents the phase-only correlation method followed by the experimental results with impostor and genuine matches presented in section 5.3. Sections 5.4 and 5.5 present the band-limited and windowed POC methods which prove to be more efficient than plain POC. Section 5.6 concludes the chapter.

5.1 Overview of Digital Iris Code Generation

Having normalized the polar-transformed iris image, the next stage is to generate a unique binary iris code based on the transformed image. First, features need to be extracted from the iris texture to generate an iris code and then this code has to be matched against existing codes to determine if the identity is genuine or that of an impostor. To our knowledge, only one system [28][33] has been developed to-date which uses phase-only correlation for iris encoding and matching.

The most widely used and most extensively studied feature extraction method the is the two-dimensional Gabor wavelet as suggested in [1] and presented in appendix A. Other methods include the use of Gabor filters and wavelet transform[16], a Laplacian of a Gaussian filter [17], independent component analysis (ICA) [36] and two-dimensional cross-phase spectrum function [28][33].

5.2 Encoding Using the Cross Spectrum Phase Correlation

In many situations, we are provided with two images which are instances of the same object but acquired from an altered angle and position. In [37] Kuglin and Hines were the first to introduce an image alignment method based on the phase correlation of two images for obtaining the displacement between two given images. In [26], Ito et al. propose an efficient matching algorithm based on the phase components of two-dimensional *discrete Fourier transforms* (DFT) of a normalized fingerprint image. Later, in [28], Miyazawa builds upon this work and introduces the technique of phase-based iris matching using the POC function. The function, which requires the calculation of the cross-correlation function in the two-dimensional spatial Fourier domain, was successfully used for image registration [27], fingerprint [26] and iris [28] matching. Moreover, in [26][28] the authors report having built a commercially available product in cooperation with the Yamatake Corporation of Japan using the well-known CASIS iris image database for training.

5.2.1 Formulation of the Phase-Only Correlation Algorithm

The algorithm is based on the premise that the differences between two images are purely translational and builds upon the time-shifting property of the Fourier transform. This property states that a shift in the image plane (time) corresponds to an exponential factor in Fourier domain (frequency) as in

$$f(x-\Delta x, y-\Delta y) \xleftrightarrow[F.T.]{} F(\omega_X, \omega_Y) e^{-j2\pi[\Delta x\omega_x + \Delta y\omega_y]} \tag{5.1}$$

Let the discrete two-dimensional function $g(x,y)$ be a normalized iris image and let $f(x,y)$ be a time-shifted version as in

$$g(x,y) = f(x-\Delta x, y-\Delta y) \tag{5.2}$$

(The two functions $g(x,y)$ and $f(x,y)$ only differ by a shift)

Applying a Fourier transform on both sides we get

$$G(\omega_X, \omega_Y) = F(\omega_X, \omega_Y) e^{-j2\pi[\Delta x\omega_x + \Delta y\omega_y]} \tag{5.3}$$

(Their amplitude spectrum is the same but they have a different phase spectrum)

Then the normalized *cross-phase spectrum* function or *ratio* maintains only the information on the phase shift and is mathematically defined as

$$\phi(u,v) = \frac{F^*(f)\cdot G(f)}{\left|\left[F^*(f)\cdot G(f)\right]\right|} \tag{5.4}$$

where (u,v) are the Fourier domain coordinates, $G(x,y)$ and $F(x,y)$ are the Fourier transforms of two signals as previously defined, the brackets $|\ |$ are intended to indicate the normal and the operator * indicates the complex conjugate.

Computing the Inverse two-dimensional Discrete Fourier Transform (IDFT) of $\phi(u,v)$ results in a surface where the location of the peak value identifies the translations. That is, taking the inverse transform gives an impulse function which is centred at the shifts $(\Delta x, \Delta y)$ [28]. Hence the POC surface is obtained by applying the IDFT to the normalized cross-power spectrum function as follows:

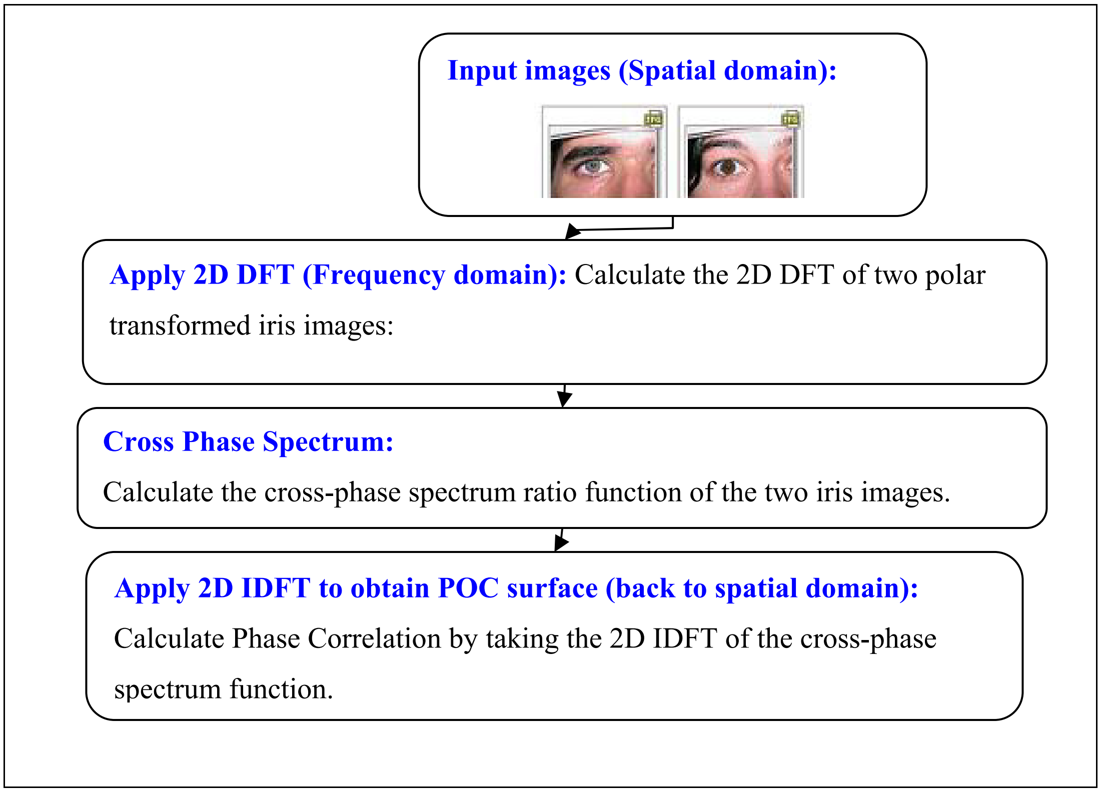
$$\mathbb{F}^{-1}\{\phi(u,v)\} = \delta_2(x+\Delta x, y+\Delta y) \qquad \textbf{5.5}$$

We wish to match the discrete images $g(x,y)$, $f(x,y)$ with the images indexed using (x,y) where *x, y* are integers. Two important characteristics of the POC functions that aid in that are [28]:

I. The *location* of the POC function peak results from translational differences due to image shifts.

II. The *height* of the POC function peak indicates the similarity of the images and hence can be used as a matching score.

Figure 5-1 presents the block diagram of the algorithm for generating the POC function:



Input images (Spatial domain):

Apply 2D DFT (Frequency domain): Calculate the 2D DFT of two polar transformed iris images:

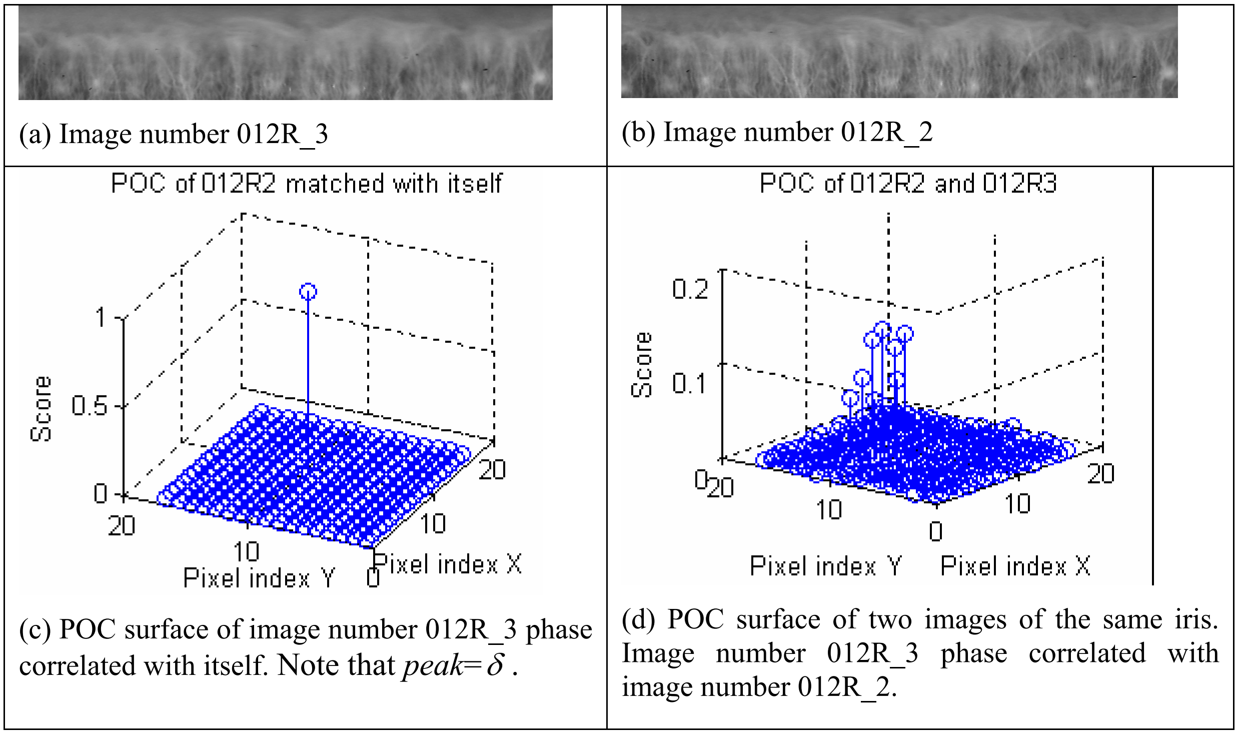
Cross Phase Spectrum:
Calculate the cross-phase spectrum ratio function of the two iris images.

Apply 2D IDFT to obtain POC surface (back to spatial domain):
Calculate Phase Correlation by taking the 2D IDFT of the cross-phase spectrum function.

Figure 5-1 Block diagram of the phase only correlation algorithm

5.3 Experimental Iris Matching Results Using POC

To empirically verify the correctness of the algorithms, we realized the algorithms and conducted several experiments in which we measured the iris matching POC scores for different iris image combinations. This section describes a set of experiments for evaluating iris matching performance using the POC function and the UPOL iris image database



(a) Image number 012R_3

(b) Image number 012R_2

(c) POC surface of image number 012R_3 phase correlated with itself. Note that *peak*= δ .

(d) POC surface of two images of the same iris. Image number 012R_3 phase correlated with image number 012R_2.

Figure 5-2 Phase-only correlation functions

In the first experiment we correlated an iris image with itself to verify that the result is a two-dimensional delta (δ) function. We correlated image 012R_3 (a) with itself and as evident in Figure 5-2(c) the result indeed agrees with the theorem [26, 27, 28], resulting in a single two-dimensional impulse with unit amplitude at the centre (*note:* the DC component of the FFT matrix was shifted and down sampled for display purposes). As stated earlier, the height of the peak is used as a similarity measure for iris image matching and in [28] Miyazawa proves that the POC measure has significantly stronger discrimination ability than does a standard correlation function. In the second experiment, image number 012R_2 (b) was correlated with image number 012R_3, which is another iris image replica of the same eye. The result (Figure 5-2(d)) suggests a distinctive peak with amplitude 0.12 and agrees with the proposed algorithm as well as having the same characteristics of the graphs presented in [26, 27, 28].

5.3.1 Genuine Iris Matching POC scores

In the third experiment we conducted an extensive number of genuine iris matching simulations in MATLAB on an iris image subset to empirically determine the effect and optimal values of:

I. Varying resolutions of 128/256 pixels radially and 512/1024 pixels circumferentially and their combinations.

II. The application of contrast enhancement via histogram equalization.

The POC function values resulting from this experiment are presented in Table 5-1 (*MATLAB: irisSimulateIrisMatchingPOC.m*).

Table 5-1 Genuine Iris Matching Phase Only Correlation Scores Using Varying Resolution without Histogram Equalization

Experiment	**128x256**	**128x512**	**256x1024**	**128x2048**	**256x2048**	**BLPOC**
(1) 012R_2,012R_3	**0.12906**	0.10656	0.070975	0.057786	0.042873	0.25134
(2) 045L_1,045L_3	**0.057527**	0.049067	0.028953	0.030319	0.025542	0.16687
(3) 002L_2,002L_3	**0.061944**	0.046336	0.027822	0.027963	0.02537	0.19547
(4) 006L_2,006L_1	**0.14358**	0.08958	0.058441	0.051866	0.036864	0.29293
(5) 010L_1,010L_2	**0.083314**	0.051471	0.039814	0.03877	0.029684	0.19493
(6) 010L_2,010L_3	**0.076047**	0.041667	0.038455	0.03024	0.032873	0.17657
(7) 010L_1,010L_3	**0.1737**	0.10589	0.070318	0.069357	0.043413	0.39989
(8) 047R1,047R2	**0.088553**	0.06301	0.040952	0.032127	0.021176	0.2001
(9) 047R1,047R3	**0.10602**	0.078637	0.077193	0.067006	0.064185	0.27359
(10) 047R2,047R3	**0.081883**	0.048047	0.029618	0.034526	0.031212	0.18699
MEAN	**0.10016**	**0.068027**	**0.048254**	**0.043996**	**0.035319**	**0.23387**
MAX	**0.1737**	**0.10656**	**0.077193**	**0.069357**	**0.064185**	**0.39989**
MIN	**0.057527**	**0.041667**				**0.16687**

As can be observed, using a higher resolution radially and/or circumferentially *decreases* the POC value in all cases. Refer to Figure 5-3, and note that even when the POC function value is small it still exhibits a distinctive delta function-like peak, a result which fully agrees with the theory.

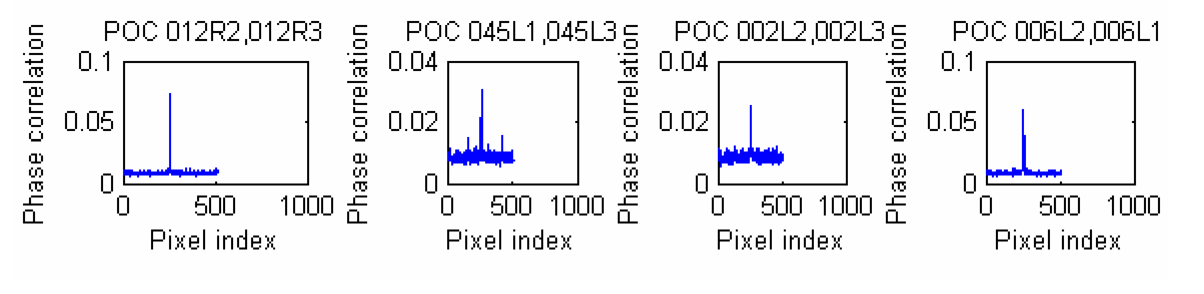
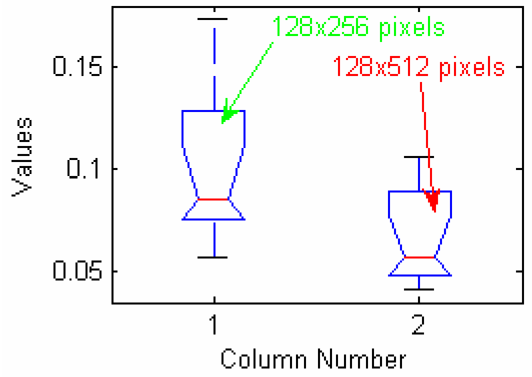
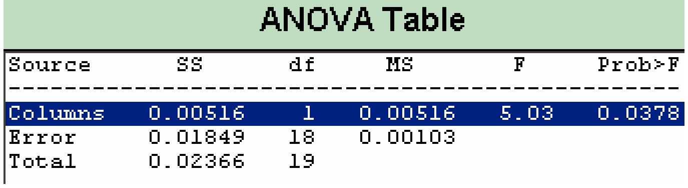


Figure 5-3 Genuine iris matching POC plots exhibit a distinctive peak

We conducted a study using one-way analysis of variance [35] (ANOVA) to determine if the strength of the POC function with a resolution of 128 x 256 was equal to the strength of the POC function with a resolution of 128 x 512. That is, does increasing the resolution really deteriorate the results? We can answer that by running a one-way ANOVA for comparing the *means* of the POC function under the null hypothesis that all the values of the POC function are drawn from the *same population*. Naturally, we are interested in the rejection of the null hypothesis, which would indicate a statistically significant result. It is common to reject the null hypothesis if the *p*-value (value returned by ANOVA) is close to zero or less than 0.05. The result of the ANOVA is presented in Figure 5-4.



(a) One way ANOVA for comparing the POC means (*irisOneWayANOVA.m*).



ANOVA Table

```
Source      SS       df      MS          F       Prob>F
-------------------------------------------------------
Columns     0.00516   1     0.00516     5.03     0.0378
Error       0.01849  18     0.00103
Total       0.02366  19
```

(b) The *p*-value for this specific study is 0.0378, which is very close to zero and below the rejection boundary (0.05), so we reject the hypothesis and conclude that higher resolution does indeed deteriorate the value of the POC function.

Figure 5-4 One way ANOVA for POC function means at different resolutions for genuine matching

We conclude that the optimal resolution for genuine iris matching is 128 pixels radially and 256 pixels circumferentially. *Note*: As manually segmenting the irides was an extremely time-consuming operation even with the aid of the semiautomatic MATLAB scripts, the genuine iris-matching experiments were conducted only on a small (10 unique irides) representative set from the UPOL database. The author hopes that future research will focus on that issue.

In some cases the contrast of the iris texture is low and may be improved by using local histogram equalization as suggested in [28]. However, using the UPOL database we found only a slight increase in the POC values with no statistical evidence to further support that assertion. The visual effect of histogram equalization on normalized images number 012R_2 (a) and image number 045L_1 (b) is presented in Figure 5-5.

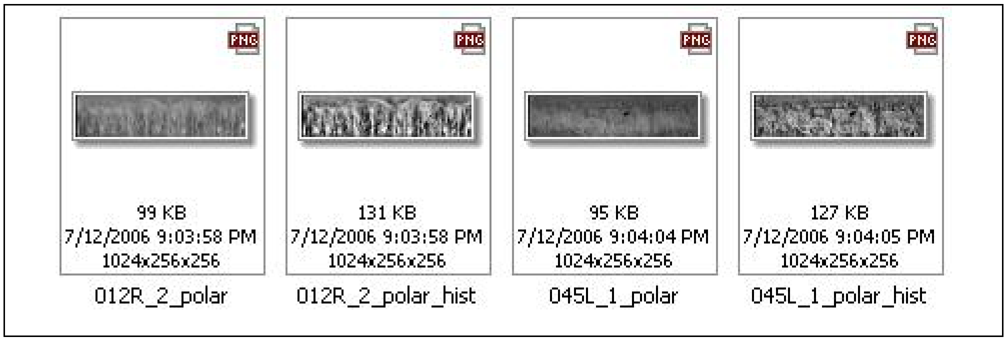


Figure 5-5 Iris texture contrast enhancement using histogram equalization

5.3.2 Impostor Iris-Matching POC Scores

In the fourth experiment we conducted impostor iris-matching simulations on the same iris image dataset (10 unique irides) as in the third experiment. The POC function values resulting from this experiment are presented in Table 5-2 (MATLAB: *irisSimulateIrisMatchingPOC.m*).

Table 5-2 Impostor Iris Matching Phase-Only Correlation Scores

Experiment	**128x256**	**128x512**	**256x1024**	**128x2048**	**256x2048**	**BLPOC**
MEAN	0.028458	0.019611	0.014914	0.014297	0.012814	0.050003
MAX	0.035146	0.025175	0.017159	0.0178	0.016618	0.067037
MIN	0.023785	0.016018				0.040616

Refer to Figure 5-6, and note that even when the POC function value is large it does not exhibit a distinctive peak (genuine matching exhibited a distinctive peak). Rather, the POC function has several noisy peaks scattered across the spectrum. This result, which fully agrees with the theory, suggests a method for differentiating between genuine and impostor matching:

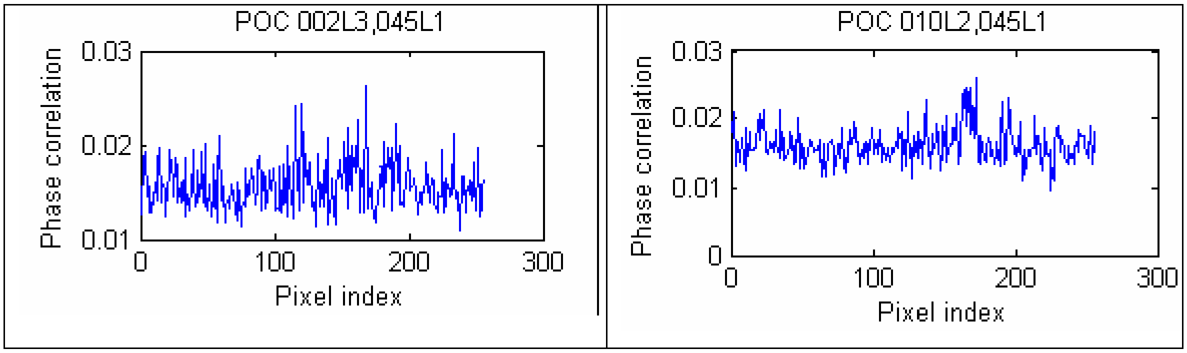


Figure 5-6 Impostor iris matching POC plots exhibit several scattered noisy peaks

First, it is important to note that in contrast to genuine matching, when impostors are matched we are interested in the *lowest* POC value. As can be observed by looking at Table 5-2, higher resolutions yield *lower* POC scores (a result which is consistent with the genuine matching results). However, there is a trade-off that has to be balanced as we cannot increase and decrease the POC score simultaneously.

With the optimal resolution of 128 x 256 for the case of genuine matching we ask whether the *maximum* value of the POC function in the impostor case (0.035146) is *lower* than the *minimum* value in the case of the genuine case (0.057527). Since it actually is lower, we can use a threshold to determine if someone is an impostor or the genuine identity holder.

5.4 The Band-Limited Phase-Only Correlation Algorithm

In [26][28] the authors suggest that the two-dimensional DFT of the normalized iris image used in the calculation of the POC has superfluous phase components in high frequencies which result in inferior values for the POC function. They present a modified version of the POC function which limits the bands in the Fourier domain.

The modified version of the POC function is a *band-limited phase-only correlation* (BLPOC) function. It is used to refine the results and obtain higher peaks when matching fingerprint and iris images. The suggestion is to band limit the two-dimensional DFT of each image with *U*V* pixels *before* the application of the two-dimensional IDFT for obtaining the POC. The concept is illustrated in Figure 5-7.

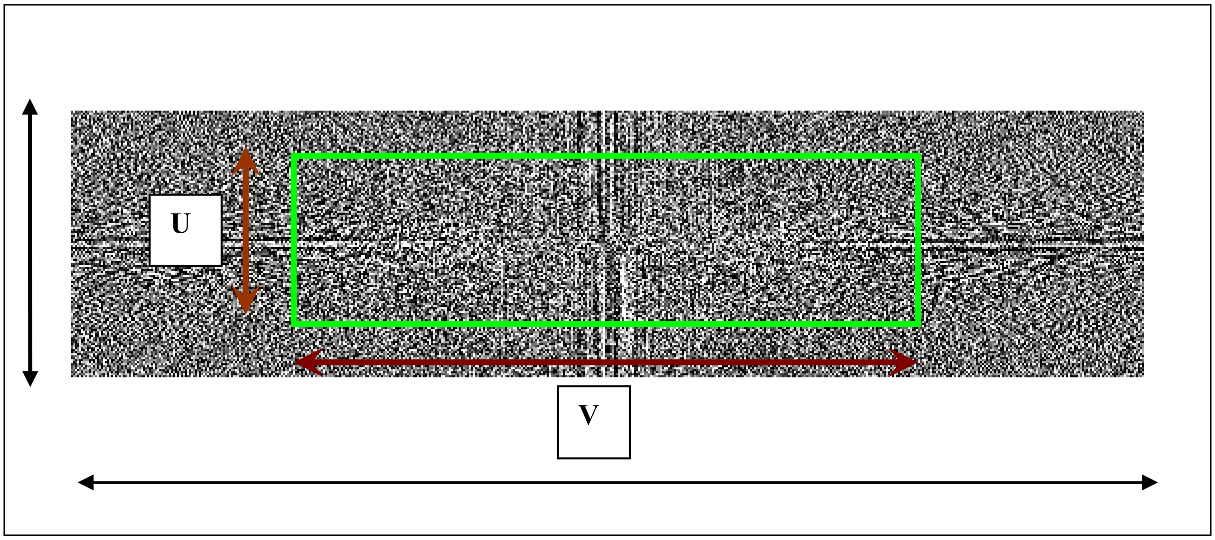
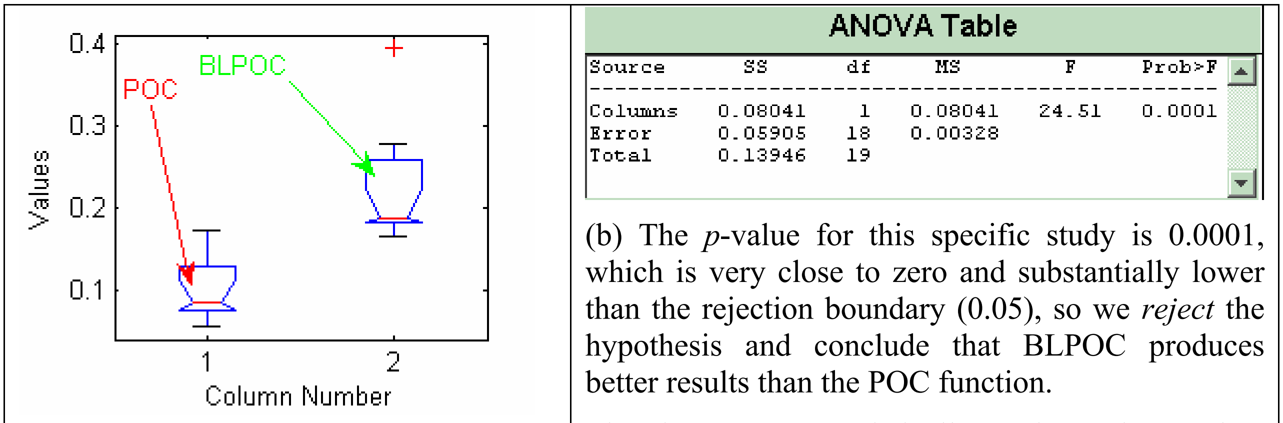


Figure 5-7 Band-limited spectrum of a two-dimensional iris DFT

As illustrated in 5-7, instead of using the full bandwidth of the transformed image in the Fourier domain, we limit the bandwidth to *U* pixels radially and *V* pixels circumferentially. In [28], using the CACIS databases a ratio of 0.6 circumferentially and 0.2 radially was empirically found to be optimal. Note that the band-limited Fourier domain images still need to be transformed back to the spatial domain using an inverse two-dimensional DFT, as was the case with the unmodified POC function.

5.4.1 Experimental Results Using BLPOC

The matching scores of the BLPOC function for genuine and impostor matching are presented in the last column of Tables 5-1 and 5-2, respectively. We modified the MATLAB realization of the POC function to allow band limiting in the Fourier domain. Experimental evaluation of the UPOL database shows that a ratio of 0.5 circumferentially and 0.5 radially is optimal, resulting in U = 128 pixels radially and V = 64 pixels circumferentially. The BLPOC values are much higher than the POC results for genuine matching and using the one-way ANOVA presented in Figure 5-8 the result proves to be statistically significant at $p=0.001$ level.



```
Source        SS       df      MS        F      Prob>F
-------------------------------------------------------
Columns    0.08041     1    0.08041    24.51    0.0001
Error      0.05905    18    0.00328
Total      0.13946    19
```

(a) One way ANOVA for comparing the POC and BLPOC means.

(b) The *p*-value for this specific study is 0.0001, which is very close to zero and substantially lower than the rejection boundary (0.05), so we *reject* the hypothesis and conclude that BLPOC produces better results than the POC function.

That is, BLPOC statistically performs better than POC (*irisOneWayANOVA.m*).

Figure 5-8 One-way ANOVA for POC and BLPOC functions of genuine matching

However, BLPOC values also increased substantially for impostor matching. We next expanded on the impostors experiment and conducted a larger-scale impostor matching experiment having 45 iris comparisons to get a better idea of the statistics related to impostor matching in Table 5-3 (*MATLAB:iris_impostor_a.txt*, *iris_impostor_b.txt, irisSimulateIrisMatchingPOCLargeScale.m*).

Table 5-3 Larger Scale Impostor Iris Matching POC and BLPOC Scores with 45 Matches

Experiment	**POC** **impostor**	**BLPOC** **impostor**	**POC** **genuine**	**BLPOC** **genuine**
MEAN	0.028458	**0.047844**	0.10016	**0.23387**
MAX	0.035146	**0.067092**	0.1737	**0.39989**
MIN	0.023785	**0.038676**	0.057527	**0.16687**

As can be observed by looking the second and third columns of Table 5-3, the statistics the BLPOC values for impostors are still high. A solution for lowering the high peaks is by a further refinement to the BLPOC function as suggested in [28] and is the topic of the next section.

5.5 The Windowed BLPOC Algorithm

The idea behind the windowed BLPOC function is to calculate the pinnacle of the iris-matching score inside a small (k * k) rectangular window centred on the origin of the BLPOC surface. This eliminates high BLPOC peaks that are a result of impostor matching without having an adverse effect on genuine matching BLPOC peaks which do not have high values outside the rectangular window. We modified our implementation (*MATLAB*: *irisWindowedBandLimitedPOC.m)* to allow the generation of windowed BLPOC and compared the performance of the windowed versus regular BLPOC function using 45 iris matches. The results are summarized in Table 5-4 and show that the BLPOC values actually decrease. An ANOVA study proved that the results are significant with a *p*-value which is practically zero. In addition, the fourth column of Table 5-4 presents the windowed BLPOC function values for genuine iris matching.

Table 5-4 BLPOC versus Windowed BLPOC Scores

Experiment	**BLPOC 128x256 (45), WINDOWED, impostor**	**BLPOC 128x256 (45), impostor**	**BLPOC 128x256 (10), windowed, genuine**
MEAN	**0.038857**	**0.047844**	**0.23387**
MAX	0.067092	0.067092	0.39989
MIN	0.026926	0.038676	0.16687

For the case of genuine matching we again ask whether the *maximum* value of the POC function in the impostor case (**0.067092**) is *lower* then the *minimum* value in the case of the genuine case (**0.16687**). Since it actually is, we can use a threshold to determine if someone is the actual identity holder. To allow a small margin of error, we arbitrarily decided on the threshold values in Table 5-5 for the BLPOC function to differentiate between genuine and impostor matching (*irisIsImposterDecisionRule.m*):

Table 5-5 WBLPOC Threshold Values for Genuine and Impostor Matching, used to decide Acceptance/Rejection

Windowed BLPOC Value	**Decision**
BLPOC < 0.08	**Impostor**
0.15>BLPOC>0.08	Further analysis required
BLPOC>0.15	**Genuine match**

With the values in Table 5-5, we determine it is possible to have a single threshold to perfectly classify the impostor and genuine groups into two distinct groups. Using these values we were able to produce zero false matches and 100% genuine matches meaning that there is a zero false acceptance and zero false rejection rates.

Figure 5-9 serves to further illustrate this concept and presents the WBLPOC score distribution along 45 impostor and 10 genuine matches:

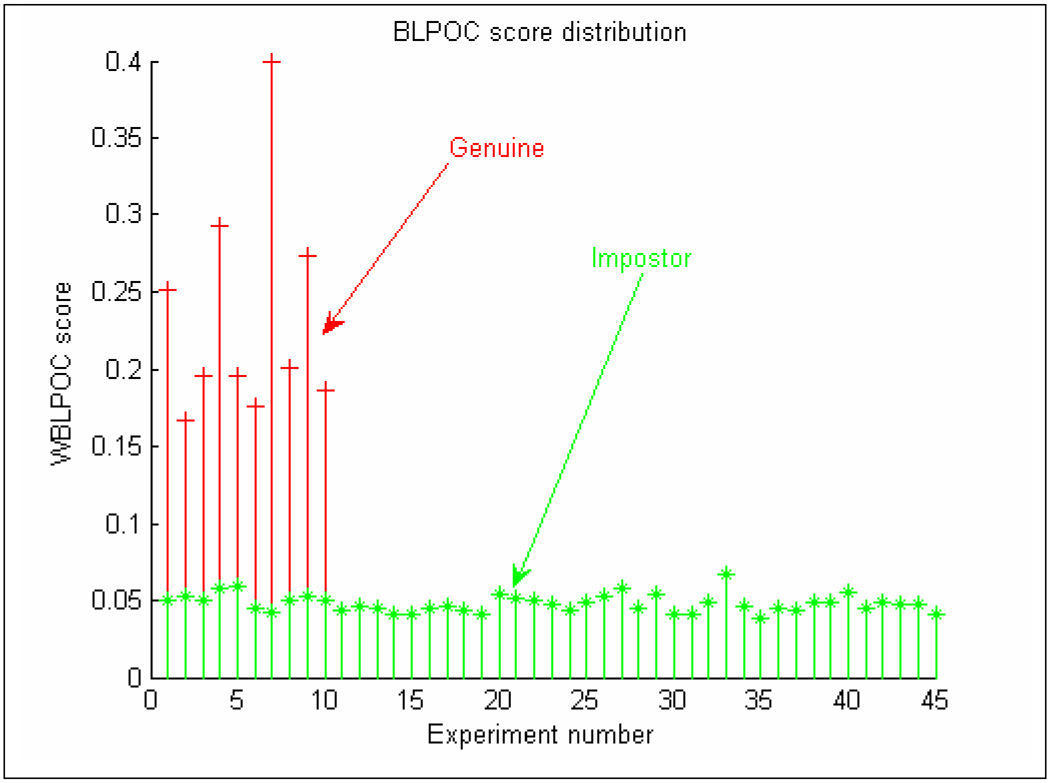


Figure 5-9 BLPOC score distribution

Refer to figure 5-9 and note that in this ideal case there is no overlap between the minimum BLPOC scores for measurements of the same person and the maximum score for measurements between different people.

Regarding the “Further analysis required” row in Table 5-5, we refer to the case in which the BLPOC value does not fall within either the impostor or the genuine match threshold regions. In [28][33], if such a case occurs, the normalized image is subjected to scale correction, a process in which the matched image is either stretched or contracted *before* the calculation of the BLPOC function. Since the two-dimensional DFT function, which is part of the BLPOC function, mandates the use of equally sized images, the smaller image of the two matched images is zero-padded to match the size of the larger image before the application of the DFT. In our implementation we *did not* have iris matches that produced border-line results and hence we have not implemented the scale correction algorithm. Nevertheless, since we only used a subset of the actual UPOL database, it is possible that if all images are tested, some values do fall within that region. This issue should be investigated by extending our system in a future work.

CHAPTER 6
DISCUSSION, FUTURE WORK AND CONCLUSIONS

6.1 Discussion

In this work we have presented a novel variation of an iris matching method using a phase-based image matching technique. Excluding the implementation of the circular Hough transform, we were able to fully accomplish our goals. Our efforts to attain a better understanding of the phase-only correlation algorithm led us to three new findings which are different from those reported in [28]. We based our new variation mostly on those findings:

I. We found no evidence to support the claim that the application of histogram equalization prior to the generation of the POC score yields superior results. In our case, this is attributed to using the very high quality images with high contrast in the UPOL database which would not benefit from further contrast enhancement. However, in [28] the CASIA [9] database was utilized which contains low contrast iris images.

II. We found using analysis of variance that regarding WBLPOC, restricting the bandwidth using ratios of 0.5 and 0.5 on the x and y-axis respectively before the application of the inverse Fourier transform yield optimal results while the corresponding ratios reported by [28][33] are 0.6 and 0.2. We assume this may be explained by the inherent variation in the iris databases and have not found scientific evidence to explain the differences.

III. In [28][33] it is reported that in the case in which the WBLPOC value is very close to genuine match threshold region, the normalized image is subjected to scale correction, a process in which the matched image is either stretched or contracted *before* the calculation of the WBLPOC function. In our implementation however, all WBLPOC scores fell within the threshold regions so we have not utilized scale correction.

Because we concentrated on iris encoding rather than on segmentation, our current system relied on the manual segmentation of the irides, limiting the evaluation of a larger database. Not implementing the circular Hough transform had not prevented us from fully evaluating phase-based image matching since our MATLAB implementation allowed for the semi automated segmentation of the iris. However, we found during this work that if the localization process is faulty, the efficiency of the overall recognition will be severely degraded. Thus realizing the Hough transform would contribute to the creation of a fully automatic and more efficient system as reported in [28][33].

We based our analysis on the simplifying assumption that the iris and pupil can be approximated by circles and that in addition they are also concentric. While this assumption provides a good approximation, it does not hold for several iris images in the UPOL, CSAIS and the UBIRIS database as some irides are better approximated by an ellipse rather than a circle and some irides do no exactly coincide with the pupil. Nevertheless, we were able to lower the influence of this issue by extracting the lower half part of the iris only thus decreasing the area which does not form an exact circle.

A major contribution and successful outcome made in this thesis is the implementation of a Fourier-based algorithm as MATLAB toolbox. One of the notable advantages of using MATLAB was that it has an extensive number of built-in image processing routines available, such as reading and writing images, 2D DFT, and inverse DFT etc. As a result, there was no need to write code for these routines. On the other hand using the ITK C++ image processing library we found that many of the build in functions have to be modified to accommodate our needs- a process which was too time consuming. We authored approximately 40 MATLAB routines creating the first publicly available phase based iris recognition toolbox which, in our opinion, could augment academic research in the field of iris recognition. Some of the highlights of the system are:

I. Handling of all iris pre-processing and post-processing operations such as file input and output, iris image display, Gaussian filtering, and statistical analysis using the MATLAB statistical signal processing toolbox.

II. Semi automatic iris segmentation using the canny edge detector and proprietary code that supports the drawing and marking of circles and disks on an iris image.

III. Fully automatic Cartesian to polar transformation with user selected image resolution, and user selection of the region of interest (either the full iris or the lower section only)

IV. Fully automatic iris encoding and matching using phase-only correlation, band limited and windowed phase-only correlation.

V. Successfully tested on subsets from the UPOL iris image database.

VI. Modular, routines can be added or used in conjunction with other databases (UBIRIS, CASIA), allowing large scale iris matching simulations.

VII. Effective, producing zero false (impostor) and perfect genuine matches.

Our MATLAB realization enabled us to fully evaluate the algorithms and provides a unique modular system which is also amongst the few publicly available iris recognition software libraries. The system can be easily adapted to use other iris image databases and would reduce hours of tedious research and development work.

6.2 Future work

We envision a number of feasible directions of future work as described below. We believe the chief weaknesses of our system and several issues worthy of investigation are:

I. Using a larger number of iris images from the UPOL database to achieve better statistical significance. Our experiments were conducted only on a subset of the UPOL database.

II. Improving the efficiency of POC, BLPOC and windowed BLPOC. No attempt has been made to optimise the functions, as more attention had been given to verifying the correctness of the algorithms.

III. Implementing the system in C++ based on the ITK image processing library. While MATLAB has many advantages, its drawbacks are code dependency on the MATLAB engine, the expensive signal processing toolbox and slow execution time. This restricts the usage of the developed algorithms to only within the original environment. Moreover, at present, while there are several MATLAB open source iris recognition libraries such as our own and [12], there is no C++ open source iris recognition library available. Using C++ has the advantage of being cross platform and hence easily portable between diverse platforms. Our initial tests with a partial implementation of the system in C++ show a promising run-time improvement over its MATLAB counterpart though the man hours required to realize the system would be much higher.

IV. Evaluating the performance of the algorithm with the UBIRIS noisy image database. Our work evaluated the performance of phase-based matching on the UPOL database and in [28] the CSAIS database was utilized. It would be beneficial to evaluate the affect of image focus, reflection, luminosity problems and other noise artefacts (which were introduced on purpose into the UBIRIS database) on the success rates of the algorithm and compare the results with those collected using the other databases.

6.3 Conclusion

Our experimental results demonstrate that our attempt to encode the iris using our original variation of a phased-based approach for iris recognition is realistic, promising and consistent with previous work found in the open literature [26][27][28] and [33]. The system produced zero false (impostor) matches using 45 iris comparisons and perfect genuine matches using 10 iris comparisons.

Bibliography

[1] Daugman, J, Computer Lab., Cambridge University, UK (2002). *How iris recognition works*. Proceedings of 2002 International Conference on Image Processing, Vol. 1.

[2] Daugman, J. (2002). The importance of being random: Statistical principles of iris recognition. *Pattern Recognition*, *36*, no.2, 279-91.

[3] Tortora, G. J., & Grabowski, S. R. (2004). *Principles of anatomy and physiology* (10th ed.). New York: Harper & Row (0060466944).

[4] Association for Macular Diseases. (2006). *The anatomy of the eye*. Retrieved February 15, 2006, from the association Web site: http://www.macula.org/anatomy

[5] Proenc¸a, H., & Alexandre, L. A. (2005). *UBIRIS*: *a noisy iris image database*. Presented at ICIAP 2005, 13th Annual International Conference on Image Analysis and Processing, Cagliari, Italy. Retrieved March 10, 2006, from http://www.di.ubi.pt/~hugomcp/doc/ubiris.pdf

[6] Flom, L., & Safir, A. (1987). Iris recognition system. *U.S. Patent No. 4,641,394*. Washington, DC: U.S. Patent and Trademark Office.

[7] Javier Ortega-Garcia, Carlos Bousono-Crespo (2004). *A survey report on existing biometric databases*. Universidad Politechnica de Madrid, BioSecure Deliverable D01-1-1_b2. (2006).Retrieved December 5, 2006, from http://www.cilab.upf.edu/biosecure1/public_docs_deli/BioSecure_Deliverable_D01-1-1_b2.pdf

[8] Dobes, M., & Machala, L. (2004). *The UPOL iris image database* 2004. Retrieved December 2006, from http://phoenix.inf.upol.cz/iris/ and http://apfyz.upol.cz/eng/obraz.html

[9] Chinese Academy of Sciences Institute of Automation. (2006) *The CASIA iris image database*. Retrieved January 2006, from the Iris image database Web site: http://www.sinobiometrics.com

[10] Cohen, M., & Toyama, K. (2006), *Digital photography with flash and no-flash image pairs*. Retrieved May 15, 2006, from http://research.microsoft.com/~hoppe/flash.pdf

[11] Canny, J. (1986) *A computational approach to edge detection*, IEEE Transactions Pattern Analysis and Machine Intelligence, *8*, 679-698.

[12] Masek, M. (2003). *Recognition of human iris patterns for biometric identification.* Honours project, the School of Computer Science and Software Engineering, The University of Western Australia.

[13] Phillips, P. J., Moon, H., Rizvi, S. A., & Rauss, P. J. (2000). The FERET evaluation methodology for face-recognition algorithms. *Transactions of Pattern Analysis and Machine Intelligence, 22*, 1090-1104.

[14] *The Insight Toolkit (ITK)*. (n.d.). Retrieved December 20, 2006, from http://www.itk.org/HTML/Download.htm

[15] Daugman, J., & Downing, C. (2001). Epigenetic randomness, complexity and singularity of human iris patterns. *Proceedings of the Royal Society B: Biological Sciences, 268*, 1737–1740.

[16] Zhu, Y., Tan, T., & Wang, Y. (2000). *Biometric personal identification based on iris patterns*. Proceedings of the 15th International Conference on Pattern Recognition, Spain, Vol. 2.

[17] Wildes, R. (1997). Iris recognition: an emerging biometric technology. *Proceedings of the IEEE, 85*, no.9.

[18] Meena1, B. R., Vatsa1, M., Singh, R., & Gupta, P. (2004). Iris based human verification algorithms. *Lecture Notes in Computer Science, 3072,* 458-466.

[19] Köthe, U. (2000). *VIGRA: A Vision with Generic Algorithms, a novel computer vision library.* PhD thesis, Universitaet Hamburg.

[20] Hough, P. V. C. (1962). A method and means for recognizing complex patterns. *U.S. Patent No. 3,069,654*. Washington, DC: U.S. Patent and Trademark Office

[21] Kronfeld, P. (1962). Gross anatomy and embryology of the eye. In H. Davson (Ed.), *The eye*. London: Academic.

[22] Deriche, R. (1993). *Recursively implementing the Gaussian and its derivatives* (Tech. Report 1893, Programme 4). Nancy, France: Institut National en Informatique et en Automatique.

[23] Lindeberg, T. (1994). *Scale space theory in computer vision*. Dordrecht, The Netherlands: Kluwer Academic.

[24] Stollnitz, E. J., DeRose, T. D., & Salesin, D. H. (1995). Wavelets for computer graphics: A primer. *IEEE Computer Graphics and Applications, 15*, no.3, 76-84.

[25] Gonzalez, R.C., & Woods, R.E. (2002). *Digital image processing* (2nd ed.). Englewood Cliffs, NJ: Prentice Hall.

[26] Ito, K., Nakajima, H., Kobayashi, K., Aoki, T., & Higuchi, T. (2004). A fingerprint matching algorithm using phase-only correlation. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, E87-A*, no.3, 682-691.

[27] Takita, K., Aoki, T, Sasaki, Y., Higuchi, T., & Kobayashi, K. (2003). High-accuracy subpixel image registration based on phase-only correlation. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, E86-A*, no.8, 1925-1934.

[28] Miyazawa, K., Ito, K., Aoki, T., Kobayashi, K. (2005). A phase-based iris recognition algorithm. *Lecture Notes in Computer Science*, *3832*, 356-365.

[29] Daugman, J. (2005). *Results from 200 billion iris cross-comparisons* (Technical Report UCAM-CL-TR-635). Cambridge, UK: University of Cambridge Computer Laboratory.

[30] Waltz, F. M., & Miller, J. W. V. (1997). *An efficient algorithm for Gaussian blur using finite-state machines.* Proc. SPIE Conference on Machine Vision Systems for Inspection and Metrology VII, Pittsburgh, Vol. 3521.

[31] Bloom, J. A., & Reed, T. R. (1996). A Gaussian derivative-based transform. *IEEE Transaction in Image Processing*, *5*, 551-553.

[32] Rakshit, S., & Monro, D. M. (2006). *Effects of sampling and compression on human iris verification*. Presented at the 2006 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2006), Department of Electronic and Electrical Engineering, University of Bath, United Kingdom.

[33] Miyazawa, K., Ito, K., Aoki, T., Kobayashi, K., & Nakajima, H. (2005). An efficient iris recognition algorithm using phase-based image matching. *Proceedings of the IEEE International Conference on Image Processing 2005,* pp. II-49-II-52.

[34] Marr, D., & Hildreth, E. (1980). A theory of edge detection. *Proceedings of the Royal Society of London* B, 207, 187 -- 217.

[35] Henry Scheffé (1999). *The Analysis of Variance* ISBN: 0-471-34505-9 Paperback, March 1999.

[36] Kwanghyuk Bae, Seung-In Noh, Jaihie Kim (2003). *Iris Feature Extraction Using Independent Component Analysis*, forth International Conference Audio-and Video-Based Biometric Person Authentication, Guildford, UK, June, 2003, pp.838-844.

[37] C. Kuglin and D. Hines (1975) *The Phase Correlation Image alignment Method*. IEEE Int. Conf. on Cybernetics and Society, 1975.

Alternative Iris Encoding Methods

Encoding phase data using Gabor function

Gabor [24] introduced a one dimensional transform in which the basis functions are sinusoids weighted by Gaussian windows. These functions known as *Gabor wavelets* are localized in both space and frequency domains and have the shape of plane waves restricted by a Gaussian envelope function. In two dimensions the Gabor functions are directional sinusoids weighted by Gaussian windows. Daugman [1] used a two dimensional Gabor transforms for encoding the iris.

One of the main reasons for using a Gabor wavelet in image processing is that it is optimal in the sense of having minimum uncertainty in the time-frequency space

The Gabor function is a sine wave modulated by a Gaussian function. The two-dimensional Gabor function also known as the *mother wavelet* is defined as:

$$g(x,y)=\left(\frac{1}{2\pi\sigma_x\sigma_y}\right)\exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right)+2\pi j\omega x\right] \qquad \textbf{6.1}$$

The Fourier transform of the Gabor function is defined as:

$$\mathbb{F}\{g(x,y)\}=G(u,v)=\exp\left\{-\frac{1}{2}\left[\frac{(u-\omega)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right]\right\} \qquad \textbf{6.2}$$

where σ_x,σ_y are the STDs of the 2D Gaussian envelope respectively along x-axis and y-axis and:

$$\begin{aligned}\sigma_u &= 1/2\pi\sigma_x,\\ \sigma_v &= 1/2\pi\sigma_y\end{aligned} \qquad \textbf{6.3}$$

The Gabor functions are known to span the whole space but at the same time the basis is not orthogonal. Using the mother wavelet $g(x,y)$ we can construct a new family of modified Gabor functions which differ by *scale* and *orientation* by using the following generating function:

$$
\begin{aligned}
& g_{mn}(x, y) = \beta^{-k} g(x', y') \\
& \beta > 1 \\
& and, m, n \in \text{int}
\end{aligned}
\qquad \textbf{6.4}
$$

where (x', y') are the rotated coordinates of (x, y) as in:

$$
\begin{aligned}
x' &= \beta^{-m}(x\cos(\theta) + y\sin(\theta)) \\
y' &= \beta^{-m}(-x\sin(\theta) + y\cos(\theta))
\end{aligned}
\qquad \textbf{6.5}
$$

and $\theta_n = (n * \pi) * (1/k)$ where k is the total number of *orientations* and m is the total number of *scales* used in the transform. The normalization factor β^{-m} guarantees that the energy does not depend on *m*. The resulting filter has the form:

$$
g(x, y) = \left(\frac{1}{2\pi\sigma_x\sigma_y} \right) \exp\left[-\frac{1}{2}\left(\frac{x'^2}{\sigma_x^2} + \frac{y'^2}{\sigma_y^2} \right) + 2\pi j\omega x' \right] \qquad \textbf{6.6}
$$