DESIGN OF AN IRIS VERIFICATION SYSTEM ON EMBEDDED BLACKFIN PROCESSOR FOR ACCESS CONTROL APPLICATION

RICHARD NG YEW FATT

MASTER OF ENGINEERING SCIENCE

FACULTY OF ENGINEERING AND SCIENCE
UNIVERSITI TUNKU ABDUL RAHMAN
APRIL 2012

DESIGN OF AN IRIS VERIFICATION SYSTEM ON EMBEDDED BLACKFIN PROCESSOR FOR ACCESS CONTROL APPLICATION

By

RICHARD NG YEW FATT

A thesis submitted to the Department of Electrical and Electronic Engineering,
Faculty of Engineering and Science,
Universiti Tunku Abdul Rahman,
In partial fulfillment of the requirements for the degree of
Master of Engineering Science
April 2012

ABSTRACT

Biometric identification employs physiological or behavioral characteristics to identify an individual. It provides a robust technique and has become the standard in identity authentication and access control. Among all the biometrics, iris has distinct phase information used for accurate recognition. Furthermore, iris verification systems can be non-invasive and user friendly as iris pattern is imaged from a distance. Due to its reliability, iris recognition is used for high security applications. For the use of iris recognition on portable devices, the user can capture the iris image while pointing the capture device at his eye. This project presents an implementation of a Digital Signal Processor (DSP) based iris verification system. The goal is to develop a fast speed and high accuracy iris verification embedded system, based on Analog Devices Blackfin DSP. The project involves two parts, software and hardware. First, the iris verification algorithm is developed in a C/C++ program. The three main stages of iris verification system are image preprocessing, feature extraction and template matching. The method implemented is evaluated on iris images taken from the CASIA iris image database version 1.0. Experimental results show that the proposed 1D Log Gabor filter and 1D advanced correlation filter have achieved high accuracy of 98.62% and 98.77% respectively. For the DSP implementation, the C/C++ code is further optimized to perform the verification in a shorter time. All the algorithms are mapped onto the DSP without compromising performance. Optimization is done through C/C++ source code tuning, loop optimization using pragmas and optimizing of conditional codes. Data cache and memory are utilized for optimization as well. The optimization gain is as high as 67.74%. Final experimental results demonstrate that the iris verification system is capable of completing a verification in less than one second. The DSP based iris verification system provides a fast and accurate authentication solution within a compact portable device.

ACKNOWLEDGEMENT

First of all, I would like to express my appreciation to my supervisor Mr. Mok Kai Ming for his guidance and advice. He is the one who introduced me to the iris recognition project. He has put in a lot of effort in arranging the project meetings and has to drive a long way from Kampar campus to Petaling Jaya campus. I also thank Dr. Tay Yong Haur, who is my co-supervisor for guiding me on how to do research. His criticism and guidance have encouraged me to work consistently to complete my project. I also thank him for his time in conducting weekly meetings to discuss our project. They have provided full support and motivation that helped me to complete my research. Thanks also go to Dr. Lo Fook Loong for reviewing my thesis.

I would like to acknowledge all the members in the Computer Vision and Information Systems (CVIS) group especially Mr. Kenny Khoo Kuan Yew, Mr. Tou Jing Yi, Mr. Ho Wing Teng, Mr. Heng Eu Sin, Mr. Lee Chee Wei, Ms. Chan Siew Keng, and Mr. Lim Hao Wooi. They have helped me a lot in enhancing my programming skill. I also thank them for their knowledge sharing and for the fun times we had throughout my studies.

A special thank goes to Mr Yap Wooi Hen, who is attached to our CVIS lab for a few months. He is a good mentor and is ever willing to share his knowledge with all of us. He has been replying to my emails regarding questions I have about digital signal processor.

I would like to thank my best friends and family for their strongest support. Thanks goes to my parents for giving me this opportunity to pursue my interests.

Finally, I would like to acknowledge the Institute of Automation, Chinese Academy of Science for providing the CASIA iris image database version 1.0.

APPROVAL SHEET

This thesis entitled **“DESIGN OF AN IRIS VERIFICATION SYSTEM ON EMBEDDED BLACKFIN PROCESSOR FOR ACCESS CONTROL APPLICATION”** was prepared by RICHARD NG YEW FATT and submitted as partial fulfillment of the requirements for the degree of Master of Engineering Science at Universiti Tunku Abdul Rahman.

Approved by:

____

(Mr. Mok Kai Ming) Date: 6th April 2012
Supervisor
Department of Computer and Communication Technology
Faculty of Information and Communication Technology
Universiti Tunku Abdul Rahman

____

(Dr. Tay Yong Haur) Date: 6th April 2012
Co-supervisor
Department of Internet Engineering and Computer Science
Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

FACULTY OF ENGINEERING AND SCIENCE
UNIVERSITI TUNKU ABDUL RAHMAN

Date: ______6th April 2012______

SUBMISSION OF THESIS

It is hereby certified that **RICHARD NG YEW FATT** (ID No: **07UEM08602**) has completed this thesis entitled **"DESIGN OF AN IRIS VERIFICATION SYSTEM ON EMBEDDED BLACKFIN PROCESSOR FOR ACCESS CONTROL APPLICATION"** under supervision of Mr Mok Kai Ming (Supervisor) from the Department of Computer and Communication Technology, Faculty of Information and Communication Technology, and Dr Tay Yong Haur (Co-supervisor) from the Department of Internet Engineering and Computer Science, Faculty of Engineering and Science.

I understand that the University will upload softcopy of my thesis in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

____
(RICHARD NG YEW FATT)

DECLARATION

I hereby declare that the thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTAR or other institutions.

Name Richard Ng Yew Fatt

Date 6th April 2012

TABLE OF CONTENTS

Page

CHAPTER

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACE	Average Correlation Energy
CMOS	Complementary Metal Oxide Semiconductor
DCT	Discrete Cosine Transform
DMA	Direct memory access
DSP	Digital Signal Processor
EBIU	External bus interface unit
EER	Equal Error Rate
FAR	False Acceptance Rate
FFT	Fast Fourier Transform
FPGA	Field programmable gate array
FRR	False Rejection Rate
IDDE	Integrated Software Development and Debugging Environment
IP	Intellectual Property
NIR	Near Infrared
ONV	Output Noise Variance
OTSDF	Optimal Trade-off Synthetic Discriminant Function
PCE	Peak to Correlation Energy
PIN	Personal Identification Number
PPI	Parallel peripheral interface
PSR	Peak to Sidelobe Ratio
ROC	Receiver Operating Curve
RTOS	Real Time Operating System
SDRAM	Synchronous Dynamic Random Access Memory

CHAPTER 1

INTRODUCTION

1.1 Background

Due to rising security concern, access control has become important in protecting resources for authorized personnel. For the traditional access control applications, users must possess identity claims, such as personal identification number (PIN), password and identity card. However, they are futile against identity theft, forgery and forgetfulness.

Biometrics is more reliable because it is based on inherent traits of a person which are difficult to steal, forge or being forgotten. This leads to the awareness of the importance of biometrics. It employs physiological or behavioral characteristics to identify an individual. The physiological characteristics are iris, fingerprint, face and hand geometry. Voice, signature, handwriting and keystroke dynamics are classified as behavioral characteristics. Biometric identification has provided a robust technique for proving identity and has become the standard in identity authentication and access control.

Recent research has shown that iris recognition can attain a high level of accuracy. There is only one false identification in 131,000 comparisons (Daugman, 1993). The high matching speed enables comparison between different irises in large database. Iris recognition is user friendly because iris can be captured from a certain distance. Furthermore, iris pattern is stable throughout life and difficult to forge. Therefore, it is the most accurate and reliable biometric identification. Iris recognition is suitably applied for high security access control and personal identity authentication.

At present, most of the prominent iris recognition algorithms are developed in General Purpose Processor (GPP) (Daugman, 1993, 2004; Wildes, 1997; Kong and Zhang, 2001; Tisse et al., 2002; Ma et al., 2002; Cui et al., 2004). Due to the growing demand for portable devices in consumer markets (PDAs, cell phone, mobile iris scanner), iris verification algorithm needs to be ported into the embedded system.

This thesis proposes an iris verification algorithm with high speed and accuracy and its implementation on embedded processor. Digital signal processor technology provides a fast authentication solution within a compact portable device.

1.2 Motivation

The main motivations for designing an embedded platform for iris verification system are:

1. Developing a robust iris verification algorithm:

   Iris verification is well suited to be applied to high security access-control system. Over the past decade, a number of investigations were carried out on iris recognition (Daugman, 1993, 2004; Wildes, 1997; Kong and Zhang, 2001; Tisse et al., 2002; Ma et al., 2002; Cui et al., 2004). However, there are still some improvements that can be achieved. For example, iris localization can be enhanced to effectively segment a non-ideal iris image. A complete solution is required to remove the eyelids, eyelashes, reflections and pupil noises. Fast iris matching is needed to compare billions of templates in seconds. Iris recognition research is motivated not only by its accuracy challenges but also by its demand for fast response by access control and identity authentication applications.

2. There is a growing demand for high-end embedded system:
   Due to the growing demands for portable devices in consumer market, iris verification algorithm needs to be ported into embedded systems. DSP technology provides a fast authentication solution within a small portable device. The portable device can be used in various environments.

1.3 Scope of Work

The scope of work for designing a digital signal processor based iris verification system includes the following:

1. The iris verification algorithm is able to verify user with high level of confidence and fast speed.
2. In the iris image preprocessing stage, the algorithm must be able to locate the iris region accurately under various environments. The noise removal

algorithm will be developed to achieve higher accuracy rate. It will be able to account for the rotation, translation, and scale variance of the iris images.

3. A publicly available database called CASIA iris image database version 1.0 will be used for evaluating the algorithm.
4. The algorithm will be mapped onto the Blackfin DSP without compromising the performance. The algorithm will be further optimized to achieve the speed requirement. The verification time should be less than one second.

1.4 Objectives

The project objective is to implement the iris verification prototype using the Blackfin Digital Signal Processor (DSP) for access control application. The iris verification system must be able to verify user identity accurately in less than one second.

Iris segmentation is developed as part of the iris verification algorithm. It must be able to localize iris region from the background rapidly regardless of varying illumination and contrast levels. It must remove the noises in the segmented iris regions, such as eyelashes, eyelids, reflections and pupil.

Biometric templates generated from the feature extraction stage must be able to capture salient information relevant to user identity. The biometric template must be generated in a compact format for efficient storage and matching. Feature matching must be able to differentiate an authentic user from an imposter with high confidence level.

All these algorithms must be able to be mapped into DSP without compromising performance. The compiler features are employed to optimize the algorithm for faster execution. The research outcome is a DSP based iris verification system that could verify an individual accurately with fast speed.

1.5 Thesis Outline

Chapter 1 introduces the background on biometrics technology and its applications. It presents the advantages of iris recognition for high security access-control applications. The motivation, scope of work and objectives are also outlined.

Chapter 2 presents the literature review of existing iris recognition algorithms. The three main stages in iris recognition system are image preprocessing, feature extraction and template matching. In addition, the implementation of iris verification algorithm on embedded system is studied.

Chapter 3 discusses the proposed algorithm for the iris verification system. Three main stages of iris verification algorithm are discussed in detail.

Chapter 4 explains various experiments that are conducted in this research. The experimental results for iris localization and recognition are illustrated in the last section.

Chapter 5 presents the implementation of the iris verification system using Blackfin processor. The optimization strategies for the iris verification system are discussed in detail. The performance profiles of the iris verification system are analyzed in the last section.

Chapter 6 concludes the thesis and the works that have been done for the research. The contributions of the research are highlighted as well. Future works are suggested for further improvement of the iris verification system.

CHAPTER 2

LITERATURE REVIEW

Iris recognition has become a popular research topic in recent years. Many approaches have been proposed for iris recognition. There are three main stages in an iris recognition system: image preprocessing, feature extraction and template matching.

2.1 Image Preprocessing

The objective of the image preprocessing stage is to segment the iris region which has an abundance of unique features. Image preprocessing is divided into three steps, iris localization, iris normalization and image enhancement. Iris localization detects the inner and outer boundaries of an iris as circles. Eyelid and eyelash that may cover part of the iris are detected and masked. Iris normalization converts iris image from Cartesian coordinates to Polar coordinates. The normalized iris image is a rectangular image with angular and radial resolution. The iris image may have low contrast and non-uniform illumination caused by the position of the light source. All these factors can be compensated by the image enhancement algorithm.

2.1.1 Iris Localization

2.1.1.1 Integro-differential operator

Integro-differential operator is proposed for locating the inner and outer boundaries of iris (Daugman, 1993, 2004). The operator computes the partial derivative of the average intensity of circle points. It searches for the circular path with maximum difference in intensity values. Since it uses raw derivative information, the operator can compute faster and does not require a specific threshold value. However, the operator may fail to locate the iris boundaries if there is noise in the image, such as reflection.

2.1.1.2 Hough Transform

Since the inner and outer boundaries of an iris can be modelled as circles, circular Hough transform is used to localize the iris (Wildes, 1997; Kong and Zhang, 2001; Tisse *et al.*, 2002; Ma *et al.*, 2002).

First, edge detection is applied to the binary image to generate the edge map. The edge map is obtained by calculating the first derivative of intensity values and thresholding the results. The formula is defined as

$$g(x,y) = \left|\nabla G_{\sigma}(x,y) * I(x,y)\right|$$

$$\text{where } \nabla \equiv \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right) \text{ and}$$

$$\mathrm{G}_{\sigma}(\mathrm{x},\mathrm{y}) = \frac{1}{2\pi\sigma^2}\mathrm{e}^{-\frac{(\mathrm{x}-\mathrm{x}_0)^2+(y-y_0)^2}{2\sigma^2}} \qquad (2.1)$$

$G_\sigma(x, y)$ denotes a two dimensional Gaussian filter of scale *σ* positioned at point *(x_0,y_0)*. Gaussian filter is applied to filter the random edges which are irrelevant to reduce false circle detection. The voting procedure is realized using circular Hough Transform in order to search for the circular contour from the edge map. Assuming a circle with centre coordinate *(x_c,y_c)* and radius *r*, each edge point on the circle casts a vote in Hough space. The circular contour of interest is defined as

$$(x_i - x_c)^2 + (y_i - y_c)^2 = r^2 \tag{2.2}$$

The centre coordinate and radius of the circle with maximum number of votes are defined as the pupil centre and iris inner boundary respectively. The disadvantage of the Hough transform algorithm is that it requires a threshold value to generate the edge map. The selected threshold value may remove some critical edge points and result in false circle detection.

2.1.1.3 Discrete Circular Active Contour

Active contour model has been used to localize iris (Ritter and Cooper, 2003). The contour consists of a number of vertices connected as a simple closed curve. First, a darkest average point is used as the centre of the first inner contour. The inner contour is moved by the internal and external forces until it reaches equilibrium. The outer contour is moved in a similar manner. The algorithm repeats until the inner and outer contours are matched. The discrete circular active contour model for the iris boundary is affected by the specular reflections from the cornea. Therefore, image pre-processing algorithm is required to remove the specular reflections.

2.1.1.4 Bisection Method

Lim *et al*. (2001) and Sung *et al*. (2004) applied the bisection method to locate the centre of the pupil. First, edge detection is applied to the iris image to extract the edge information. For every two points on the same edge component, bisection method is applied to draw the perpendicular lines to the centre point. The centre point with maximum number of line intersections is selected as the centre of the pupil. A virtual circle is drawn with reference to the centre of the pupil and the radius is increased within a certain range. Two virtual circles with the largest number of edge points are chosen as the iris inner and outer boundaries. Bisection method is affected by the non-uniform illuminations and reflections from eye-glasses, therefore image pre-processing algorithm is needed to remove the high intensity areas caused by illuminations and reflections.

2.1.1.5 Black Hole Search Method

Black hole search method is used to compute the centre and area of a pupil (Teo and Ewe, 2005; Grabowski *et al*., 2006). Since the pupil is the darkest region in the image, the approach applies threshold segmentation method to find the region. A threshold is defined to identify the dark areas in the iris image. These dark areas are called “black holes”. The centre of mass of these black holes is computed from the global image. The area of pupil is made up of the total number of these black holes within the region. The radius of the pupil can be calculated from the circle area formula. Black hole search method is not suitable for images with dark iris. The area of dark iris would be detected instead of the area of pupil.

2.1.2 Eyelid Detection

Cui *et al*. (2004) proposed texture segmentation to detect upper and lower eyelids. The energy of different frequencies at each region is computed to segment the eyelashes. The region with high frequency is considered as the eyelash area. The upper eyelashes are fitted with a parabolic arc. The parabolic arc shows the position of the upper eyelid. For lower eyelid detection, the histogram of the original image is used. The lower eyelid area is segmented to compute the edge points of the lower eyelid. The lower eyelid is a parabolic arc that fits with the edge points.

Chen et al. (2006) used Daubechies wavelets method to decompose the original iris image into four frequency bands. Canny edge detection is applied to each band image. To minimize the influence of eyelashes, Canny edge detector is tuned to the horizontal direction. To detect the upper eyelid, edges outside the upper iris boundary area are removed. The edge points that are close to each other within a certain distance are connected. The longest connected edge that fits with a parabolic arc is defined as the upper eyelid. To detect the lower eyelid, the steps are repeated with lower iris boundary area.

2.1.3 Eyelash Detection

Gabor filter and variance of intensity approaches are proposed for eyelash detection (Kong and Zhang, 2001). The eyelashes are categorized into separable eyelashes and multiple eyelashes. Separable eyelashes are detected using 1D Gabor filters. A low output value is obtained from the convolution of the separable eyelashes with the Gabor filters. For multiple eyelashes, the variance of intensity is very small. If the variance of intensity in a window is smaller than a threshold, the centre of the window is considered as the eyelashes.

According to Huang *et al.* (2004), both the edge and region information are used for eyelash detection. To speed up iris segmentation, the iris is roughly localized using filtering, edge detection and Hough transform. The localized iris is normalized to a rectangular image. A bank of Gabor filters is applied to the image to extract the edge information. The edge information is combined with the region information to detect the eyelashes and pupil noise regions.

2.1.4 Iris Normalization

Iris may be captured in different sizes with varying imaging distance. Due to illumination variations, the radial size of the pupil may also be different. In some cases, the pupil and iris may be non-concentric. Therefore, the iris region needs to be normalized to compensate for these variations. The homogeneous rubber sheet model algorithm remaps each pixel in the localized iris region from the Cartesian coordinates to polar coordinates (Daugman, 1993, 2004). The remapping formulas are defined in Equation (2.3).

$$
\begin{aligned}
& I(x, y) \rightarrow I(r, \theta) \\
& r \in [R_p, R_n(\theta)]\,,\ \theta \in [0, 2\pi] \\
& x_i = x_c + r\cos(\theta) \\
& y_i = y_c + r\sin(\theta)
\end{aligned}
\qquad (2.3)
$$

where *(x_i, y_i)* denotes the polar coordinates of a point inside iris region, *(x_c, y_c)* and R_p are the centre coordinates and radius of the pupil respectively, *$R_n(\theta)$* is the distance from pupil centre to the iris outer boundary which is in the function of θ, *$I(x,y)$* represents a pixel in Cartesian coordinate system and $I(r, \theta)$ represents a pixel in Polar coordinate system.

2.1.5 Image Enhancement

The normalized iris image has low contrast and non-uniform illumination caused by the light source position. The image needs to be enhanced to compensate for these factors. Local histogram analysis is applied to the normalized iris image to reduce the effect of non-uniform illumination (Zhu *et al.*, 2000).

2.2 Feature Extraction

Iris has an abundance of unique texture features, especially inside the inner part of the iris. Feature extraction transforms the enhanced iris image into a set of discriminant signatures. The significant features are encoded into templates for identification.

2.2.1 Gabor Filters

Daugman (1993, 2004) proposed 2D Gabor filters to extract iris features. Gabor filter's impulse response is defined as a harmonic function multiplied by a Gaussian function. It provides optimum localization in both spatial and frequency domains.

Each isolated iris pattern is demodulated to extract its phase information using quadrature 2D Gabor wavelets. The phase information is extracted for recognizing irises because it is discriminating. It does not depend on extraneous factors, such as imaging contrast, illumination and camera gain (Daugman, 2004).

1D Log Gabor filter is also used to extract the frequency information which represents the iris textures. A Log Gabor filter is a Gaussian transfer function on a logarithmic scale (Field, 1987). It is a band pass filter that removes the DC components caused by background brightness. The 1D Log

Gabor filter on the linear frequency scale has a transfer function as shown in Equation (2.4).

$$G(w) = \exp((-\log(w/w_0)^2)/2(\log(k/w_0))^2) \qquad (2.4)$$

where ω_0 denotes the filter's centre frequency and k denotes the bandwidth of the filter. The plot of 1D Log Gabor filter in frequency domain is shown in Figure 2.1.

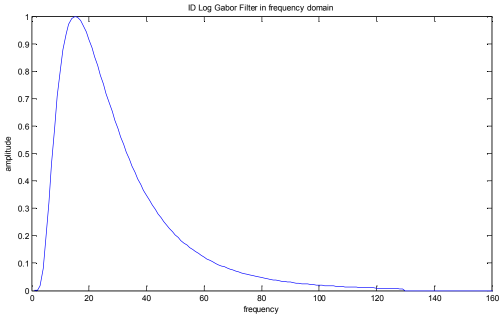


Figure 2.1: 1D Log Gabor filter in frequency domain

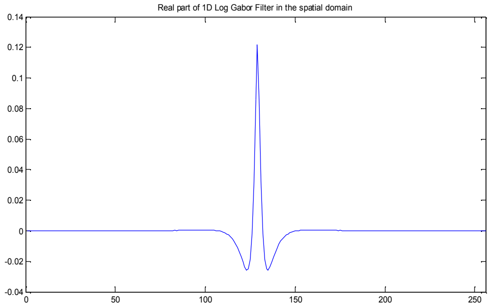


Figure 2.2: Real part of 1D Log Gabor filter in the spatial domain

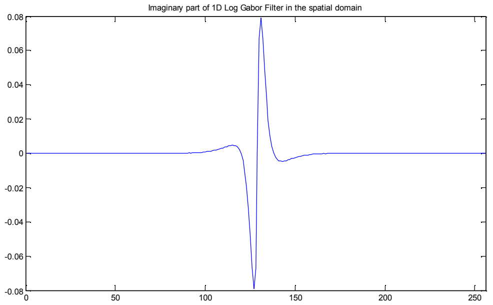
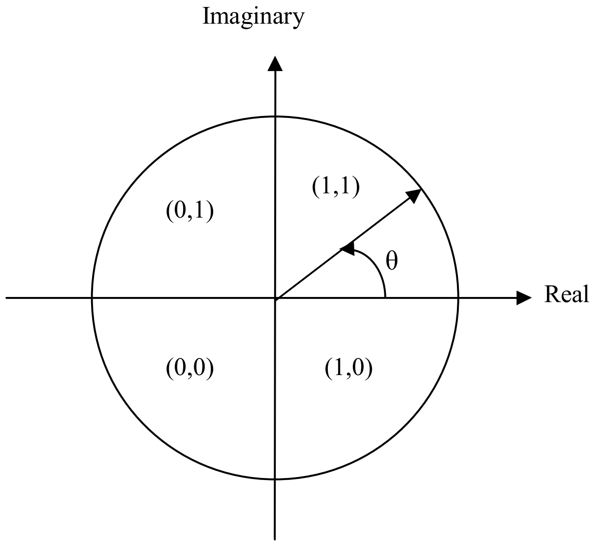


Figure 2.3: Imaginary part of 1D Log Gabor filter in the spatial domain

After applying 1D Log Gabor filter on each row of the enhanced iris image, a series of real and imaginary numbers is generated. The phase information is quantized into four quadrants in the complex plane as shown in Figure 2.4. The equations of the phase demodulation process are defined in Equation (2.5). The template consists of binary digits only.



Phase Demodulation

01 00 10 11 00 11 01 10 10 10 10 11 00 11 11 00 01 01
10 10 00 00 00 00 11 11 10 01 01 01 11 10 11 00 10 10

Figure 2.4: Phase demodulation process

$$T_{RE}(x,y)=1 \text{ if Real}(f(x,y)\geq 0)$$

$$T_{RE}(x,y)=0 \text{ if Real}(f(x,y)<0)$$

$$T_{IM}(x,y)=1 \text{ if Imaginary}(f(x,y)\geq 0)$$

$$T_{IM}(x,y)=0 \text{ if Imaginary}(f(x,y)<0) \qquad (2.5)$$

where *f(x,y)* denotes the filtered image after performing inverse Fast Fourier Transform (FFT).

2.2.2 Wavelet Transform

Wavelet transform decomposes the iris region into components with different resolutions. The commonly used wavelets are Daubechies, Biorthogonal, Haar and Mexican Hat wavelet (Poursaberi and Araabi, 2005, 2007; Chen *et al*., 2006; Rydgren *et al*., 2004; Boles and Boashash, 1998; Sanchez-Avila *et al.,* 2002).

A bank of wavelet filters is applied to the normalized iris region. Each filter is tuned for each resolution with each wavelet defined by scaling functions. The output of the filters is encoded to generate a compact biometric template.

The advantage of wavelet transform over Fourier transform is that it has both space and frequency resolution. The features are localized in both space and frequency domains with varying window sizes.

2.2.3 Laplacian of Gaussian Filter

Laplacian of Gaussian filters are used to extract features by decomposing the iris region (Wildes, 1997; Wildes *et al*., 1994). A cascade of Gaussian-like filters is applied to the iris image. The four level Laplacian of Gaussian filters are constructed to generate a compact biometric template. This approach compresses the data to obtain significant information. The compressed data can be stored and processed effectively.

2.2.4 Key Local Variation

Key local variations are used to represent the characteristics of the iris (Ma *et al*., 2004). The normalized iris image is decomposed into a set of 1D intensity signals. Dyadic wavelet transform is applied to each intensity signal. Local extrema of the wavelet transform results correspond to sharp intensity variations of the original signal. The local maximum and minimum points are encoded into a feature vector. The feature vector is then converted to a binary template.

2.2.5 Discrete Cosine Transform

This iris coding method is based on differences of discrete cosine transform (DCT) coefficients of angular patches from normalized iris image (Monro *et al*., 2007). The normalized iris image is divided into diagonal 8x12 patches. The average over width is windowed using a Hanning window to reduce the effects of noise. A similar Hanning window and DCT is applied to the patch along its length. The differences between the DCT coefficients of adjacent patches are obtained. A binary template is generated from the zero crossings of the differences between the DCT coefficients. This iris coding method has low complexity and good interclass separation. It is superior to other approaches in terms of both speed and accuracy.

2.2.6 Advanced Correlation filter

Advanced correlation filter is designed for recognition of irises from the same class (Thornton J. *et al.*, 2005; Vijaya Kumar B.V.K. *et al.*, 2003). First, the cross-correlation between the test image and the filter is performed. The output correlation plane should contain a sharp peak if there is match between the test image and the filter. There is no discernible peak if the test image is from an imposter. One of the advanced correlation filter is called Optimal Trade-off Synthetic Discriminant Function (OTSDF). The trade-off between average correlation energy (ACE) and output noise variance (ONV) can be optimized using this filter. ACE calculates the average energy of the filter output. The filter output represented by the training images should be reduced to lower the sidelobes. The equation for ACE is below.

$$ACE = \mathbf{h}^{+}\mathbf{Dh} \qquad (2.6)$$

where **h** is a vector representing the correlation filter, **D** is a diagonal matrix with the average power spectrum of the training images and the superscript + denotes the complex conjugate transpose. ONV measures the variance of the noise at the correlation output. The ONV equation is defined in Equation (2.7) if the noise is white.

$$ONV = \mathbf{h}^{+}\mathbf{h} \qquad (2.7)$$

In order to optimize both ACE and ONV, the vector of the correlation filter is defined in Equation (2.8).

$$\mathbf{h = A^{-1}X(X^{+}A^{-1}X)^{-1}u} \qquad (2.8)$$

where $A=\alpha\mathbf{I}+\sqrt{}(1-\alpha^2)\mathbf{D}$, vector **u** is the predefined peak constraints (1 for authentics and 0 for imposters) and matrix X is the vectorized 2D Fourier transform of the training images.

2.3 Template Matching

The templates generated from the feature extraction stage need a corresponding matching metric. The matching metric compares the similarity between the templates. A threshold is set to differentiate between intra-class and inter-class comparisons.

2.3.1 Hamming Distance

Hamming distance is defined as the fractional measure of dissimilarity between two binary templates (Daugman, 1993, 2004). A Hamming distance value of zero would represent a perfect match. The two templates that are completely independent would give a Hamming distance near to 0.5. A threshold is set to decide whether the two templates are from the same iris or different irises.

The fractional Hamming Distance is sum of the exclusive-OR between two templates over the total number of bits. Noise mask is used in the calculation to exclude the noise regions.

$$HD=\frac{\left\|(templateA\otimes templateB)\cap maskA\cap maskB\right\|}{\left\|maskA\cap maskB\right\|} \tag{2.9}$$

where $\otimes$ denotes the boolean exclusive-OR operator (XOR), $\bigcap$ denotes the bitwise AND operator, maskA and maskB are two noise masks corresponding to the two matching templates. For the noise mask, “1” represents the iris region and “0” represents the noise region.

The advantage of Hamming distance is fast matching speed because the templates are in binary format. Hamming distance is suitable for comparisons of millions of templates in large database.

2.3.2 Weighted Euclidean Distance

Weighted Euclidean distance is used to compare two templates for matching purpose (Zhu *et al.*, 2000). Weighted Euclidean Distance is defined as a measure of similarity between two templates. The distance between two feature points is calculated. An iris template is compared with all templates in the database. The two templates are matched if the Weighted Euclidean Distance is minimum. The Weigthed Euclidean Distance is defined in Equation (2.10).

$$WED(k) = \sum_{i=1}^{N} \frac{(f_i - f_i^{(k)})^2}{(\delta_i^{(k)})^2} \tag{2.10}$$

where f_{i} denotes the i^{th} feature of the test iris, $f_{\mathrm{i}}^{(\mathrm{k})}$ and $\delta_{\mathrm{i}}^{(\mathrm{k})}$ denote the i^{th} feature and its standard deviation of iris k, N denotes the total number of features extracted from an iris.

2.3.3 Nearest Feature Line

Nearest feature line is an efficient classification method in iris matching stage (Ma *et al.*, 2002). Feature line passes through any two feature points of the same class. The feature line extracts more variations of the feature vector than the feature points (Li and Lu, 1999). The distance from a feature point to the feature line is calculated. The nearest feature line distance will be used in the classification stage.

2.3.4 Peak to Correlation Energy

After cross correlating a test image with the advanced correlation filter, Peak to Correlation Energy (PCE) is used as a matching metric (Thornton J. *et al.*, 2005; Vijaya Kumar B.V.K. *et al.*, 2003). It is a metric used to measure the degree of match for the correlation output. It is defined in Equation (2.11).

$$\boldsymbol{PCE} = (\mathrm{peak} - \mu)/\sigma \qquad (2.11)$$

where μ and σ denote mean and standard deviation of the correlation output.

2.3.5 Peak to Sidelobe Ratio

Another metric used to measure the degree of similarity of correlation planes is called Peak to Sidelobe Ratio (Chong *et al.*, 2005). A test image from the same class will give a large PSR while the different class test image will produce a very low PSR. The PSR is defined in Equation (2.12).

$$\boldsymbol{PSR} = (mean(mask) - mean(sidelobe)) / \sigma(sidelobe) \qquad (2.12)$$

First, the correlation peak near the origin is located. The mean value of the central mask which is centred at the peak is calculated. The sidelobe is the annular region between the central mask and a larger square region. After locating the sidelobe, the mean and standard deviation of the sidelobe are computed.

2.4 Embedded Systems for Iris Verification

In commercial access control application, embedded systems are preferred because of low-power and portability. In general, there are two types of embedded system implementation: hardware-based (FPGA, VLSI) and software-based (DSP, microcontroller).

Liu-Jimenez *et al*. (2004) proposed a new architecture for an iris authentication system on FPGA. It is based on the biometric co-processor which performs the biometric tasks. The processor can avoid software attacks and thus improve the security of the system.

Liu-Jimenez *et al*. (2005) proposed a new biometric architecture based on a combination of hardware and software. The architecture, which is implemented on FPGA, provides a more secure and faster solution. The pipelined architecture reduces the processing time by 80%.

An architecture dedicated to iris identification based on wavelet packet analysis is introduced by Ea *et al.* (2005). The architecture is implemented on System-on-a-programmable-chip (SOPC). The SOPC consists of a NIOS processor and hardware intellectual property (IP). The wavelet IP is implemented on Altera Stratix Board EP1S25F780C5.

For DSP implementations, the iris recognition systems are dominated by commercial solutions. Analog Devices Inc. (2007d) uses Blackfin DSP to drive the authentication algorithms. The iris capture camera and embedded iris authentication engine board are housed inside a computer mouse. The registration and authentication data are stored inside DSP and thus eliminate threats from the hackers.

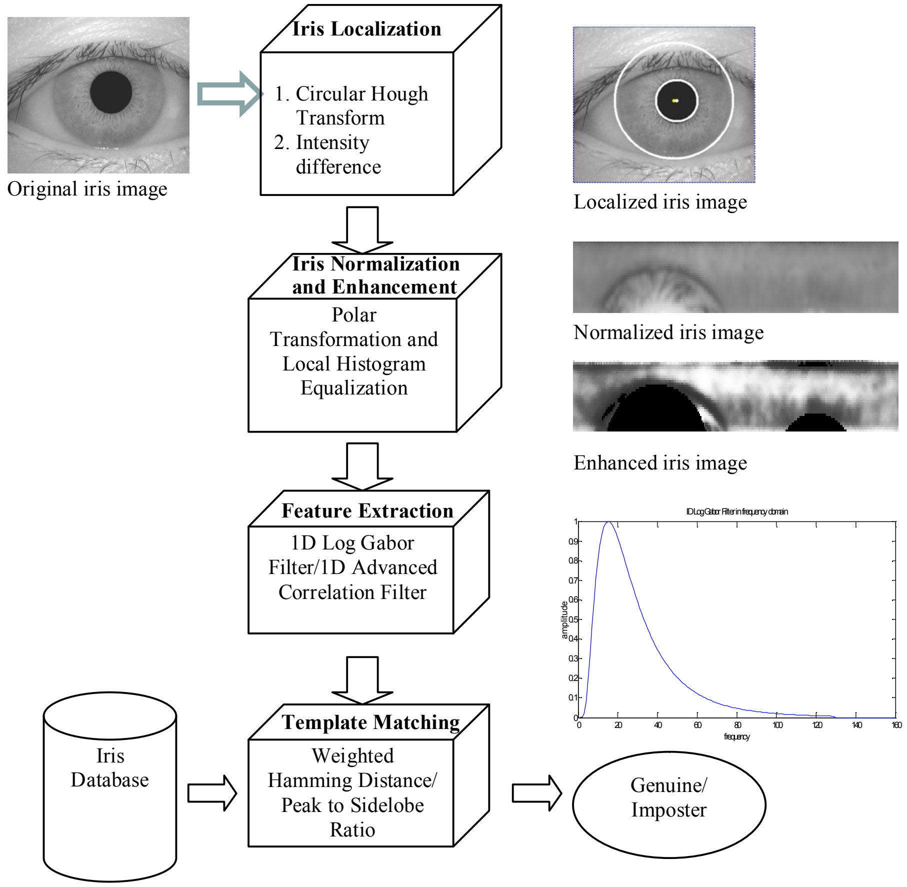
Iritech Inc. (2007) implements iris recognition on high performance embedded systems. It uses Texas Instrument's DM642 DSP processor. It employs multimodal biometrics, which include iris and face identification algorithms. The multimodal biometrics enhances the accuracy of the iris recognition system.

Miyazawa *et al.* (2006) developed an iris recognition system based on DSP. The prototype utilizes the DSP technology to achieve real time iris recognition. A fixed point DSP is used because it runs at higher clock frequency with lower power consumption.

CHAPTER 3

IRIS VERIFICATION ALGORITHM

This chapter presents the major stages of the iris verification system. The block diagram of the proposed algorithms is illustrated in Figure 3.1.



Original iris image
Iris Localization
1. Circular Hough Transform
2. Intensity difference
Localized iris image
Iris Normalization and Enhancement
Polar Transformation and Local Histogram Equalization
Normalized iris image
Enhanced iris image
Feature Extraction
1D Log Gabor Filter/1D Advanced Correlation Filter
ID Log Gabor Filter in frequency domain
amplitude
frequency
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
20
40
60
80
100
120
140
160
Iris Database
Template Matching
Weighted Hamming Distance/ Peak to Sidelobe Ratio
Genuine/ Imposter

Figure 3.1: Block diagram of the proposed algorithms

3.1 Image Preprocessing

The proposed approach gives a solution for compensating all four types of noises to achieve higher recognition rate. It consists of four parts: (a) Pupil is localized using automatic thresholding and circular Hough Transform methods. (b) Two search regions from iris inner boundary to image boundary are located based on pupil center as origin. Intensity difference is computed on the search regions to detect the iris outer boundary. (c) Two search regions are selected based on pupil position and Sobel edge detector is used to detect the upper and lower eyelids. (d) Thresholding is implemented to remove eyelashes, reflection and pupil noises. The method is evaluated using the iris images taken from the CASIA iris image database version 1.0 (CASIA, 2007). Figure 3.2 illustrates the three main steps of iris image preprocessing stage.

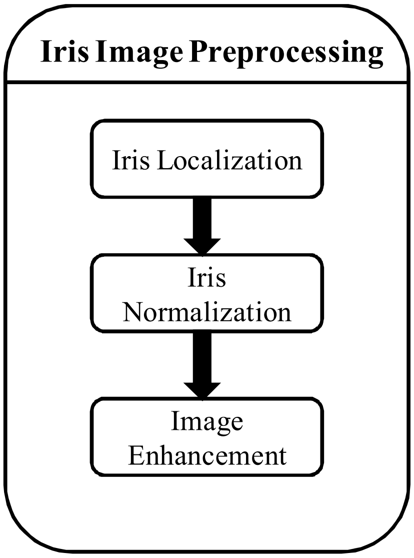


Figure 3.2: Three main steps of iris image preprocessing stage

3.1.1 CASIA Iris Database

The proposed algorithm was evaluated on CASIA iris image database version 1.0 (CASIA, 2007). There are 756 iris images from 108 different people. For each eye, seven images are captured in two sessions. The resolution of the iris image is 320×280 pixels. All the images are captured in 8-bit gray scale mode using homemade digital optical sensor. Near infrared illumination is used together with the optical sensor. The examples of iris images from CASIA database are shown in Figure 3.3.

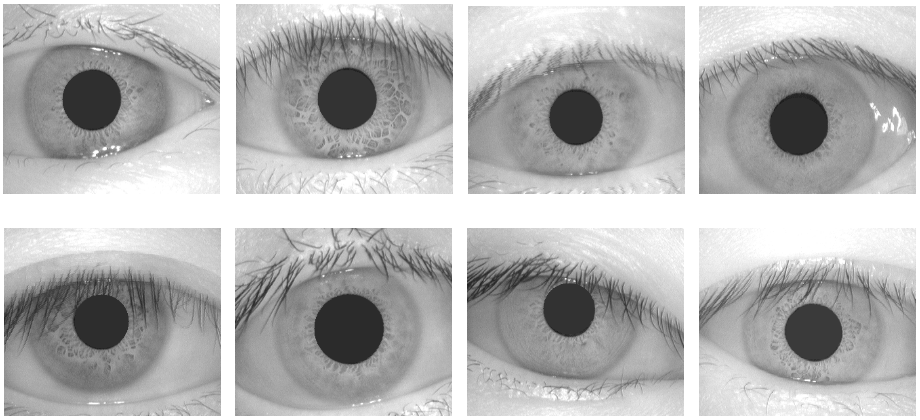


Figure 3.3: Examples of iris images from CASIA database

3.1.2 Iris Inner Boundary Localization

First, the pupil is detected using thresholding operation. From the gray scale eye image, thresholding is performed to generate a binary image which contains the pupil. Automatic threshold is chosen because the intensity values vary depending on imaging contrast, illumination and camera gain.

The proposed segmentation algorithm begins by finding the darkest region in the iris image. The iris image is divided into 20×20 pixels rectangular blocks. Then, the rectangular block with minimum average intensity is identified. The minimum average intensity is used as the automatic threshold of the binary image. Due to the presence of the non-uniform illumination inside pupil region, the automatic threshold is increased by a constant value. From our experiments, we tested with different values of 5, 10, 15, 20, 25, and 30. The pupil detection rate is highest when the threshold is increased by 10. Figure 3.4 illustrates the process of detecting the pupil region.

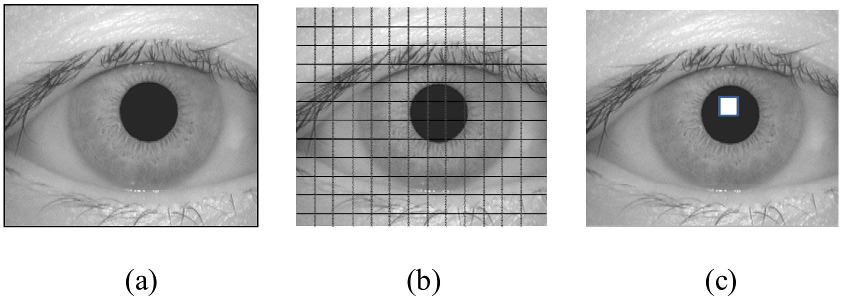


Figure 3.4: (a) Original iris image. (b) Iris image with 20×20 pixels rectangular blocks. (c) Rectangular block with minimum average intensity

Morphological operator is applied to the binary image to remove the reflections inside the pupil region and dark spots caused by eyelashes. Morphological gradient detects the sharp gray-level transitions in the image for boundary extraction. Morphological gradient of the image is the difference between the dilation and erosion. Dilation fills up holes to remove the illumination inside the pupil region. Erosion removes the spurious pixels, which are the dark spots caused by eyelashes. Dilation image is subtracted by the erosion image to obtain the boundary of the pupil region. Figure 3.5(b) shows the binary image after thresholding and morphological operator.

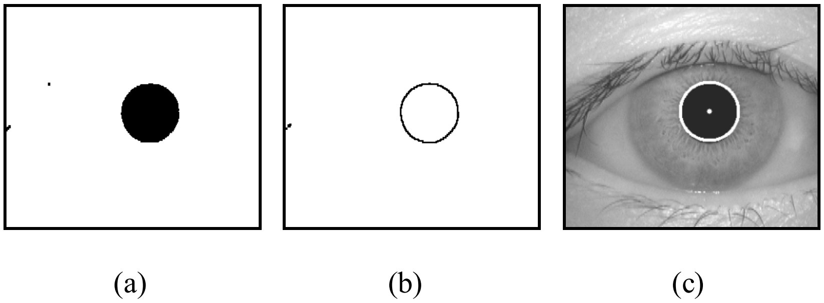
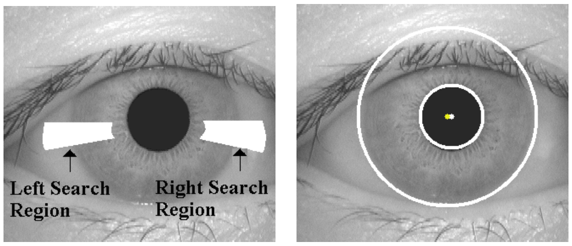


Figure 3.5: (a) Binary image after thresholding. (b) Binary image after thresholding and morphological operator. (c) Pupil localization

Since the iris inner boundary can be approximately modelled as a circle, circular Hough transform is used to localize it (Wildes, 1997; Kong and Zhang, 2001). Circular Hough Transform is used to find the centre coordinate and radius of the pupil. If the number of votes is less than a certain threshold set by the circular Hough Transform, it is assumed that the eye is heavily occluded by eyelids, or that it is in defocused or in motion blurred condition.

3.1.3 Iris Outer Boundary Localization

In order to locate the iris outer boundary, the proposed method selects two search regions including the iris outer boundary. Localization is limited within the two search regions to reduce computational time. The two search regions are defined around the outer boundary, thus this method can reduce incorrect localization. The right and left search regions are defined as shown in Figure 3.6(a).



(a) (b)

Figure 3.6: (a) Right and left search regions of the iris image. (b) Iris inner and outer boundaries localization

The pupil centre is referred as origin. The search region is a sector with radius from pupil boundary to a maximum radius. Maximum radius is defined as the distance from pupil centre to boundary of the right or left search region.

$$r_{right} = \min(width - x_c, y_c, height - y_c, \max\_\ threshold\ ) \tag{3.1}$$

$$r_{left} = \min(x_c, y_c, height - y_c, \max\_threshold) \tag{3.2}$$

where r_{right} denotes the maximum radius of the right search region and r_{left} denotes the maximum radius of the left search region. Maximum threshold is defined based on the iris size. The minimum radius of the search regions starts ten pixels away from the pupil boundary. This is to avoid the effect caused by the pupil noise. In order to avoid occlusion caused by eyelashes, upper and lower eyelids, the two search regions are selected on the lower iris region.

The intensities of each radius in the search region are added up according to Equation (3.3). The sum of intensities of each radius is calculated to reduce the effect caused by noise and variation of iris texture. The negative sign in the Equation (3.3) indicates that the y-coordinate starts from top to bottom of the image.

$$I[r] = \sum_{\theta = m}^{n} (y_c - r \sin \theta) * width \; + (x_c + r \cos \theta) \qquad (3.3)$$

Finally, the iris outer boundary, R_{iris} can be calculated using Equation (3.6). The right and left iris boundaries are the maximum difference between the sum of intensities of two outer radii and two inner radii. The iris outer boundary is the average of the distances from pupil centre to right iris boundary, R_r and left iris boundary, R_l.

$$R_r = \arg\max_r \{I[r+2] + I[r+1] - I[r-1] - I[r-2]\} \qquad (3.4)$$

$$R_l = \arg\max_r \{I[r+2] + I[r+1] - I[r-1] - I[r-2]\} \qquad (3.5)$$

$$R_{iris} = \frac{R_r + R_l}{2} \qquad (3.6)$$

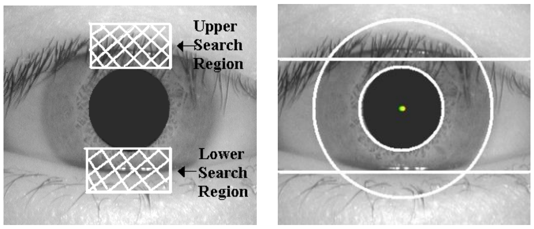
The iris centre (x_{iris}, y_{iris}) is defined in Equation (3.7) and (3.8). The x-coordinate of the iris centre shifts from the pupil centre depending on the difference between R_r and R_l. The y-coordinate of the iris centre is assumed to be the same as the pupil centre.

$$x_{iris} = x_c + \frac{(R_r - R_l)}{2} \qquad (3.7)$$

$$y_{iris} = y_c \qquad (3.8)$$

3.1.4 Upper and Lower Eyelids Detection

Similar to iris outer boundary localization, the proposed method selects two search regions to detect upper and lower eyelids. This method confines the search regions to reduce the possibility of incorrect eyelids detection and speed up the detection. The upper and lower search regions are labelled as in Figure 3.7(a). The pupil centre, iris inner and outer boundaries are used as references to select the two search regions.



(a) (b)

Figure 3.7:(a) Upper and lower search regions of the iris image. (b) Upper and lower eyelids detection

The search regions are confined within the iris inner and outer boundaries. Sobel edge detection is applied to the search regions to detect the eyelids. In order to reduce the false edge detection caused by eyelashes, Sobel kernel is tuned to the horizontal direction.

Table 3.1: Sobel kernel tuned to horizontal direction

-1	-2	-1
0	0	0
1	2	1

After edge detection, the edge image is generated. The eyelids are detected using linear Hough Transform method. The method calculates the total number of edge points in every horizontal row inside the search regions. The horizontal row with maximum number of edge points is selected as eyelid boundary. If the maximum number of edge points is less than a predefined threshold, it is assumed that the eyelid is not presented in the search regions. The eyelids detection process is illustrated in Figure 3.8.

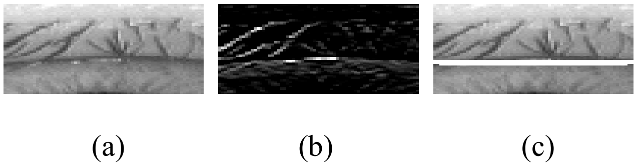


Figure 3.8:(a) Upper search region of the iris image. (b) Upper search region after Sobel edge detection. (c) Upper eyelid detection

In the proposed method, the eyelid boundaries are approximately modelled as straight line which allows for faster computation. Figure 3.9 shows the examples of the localized iris images from CASIA iris database.

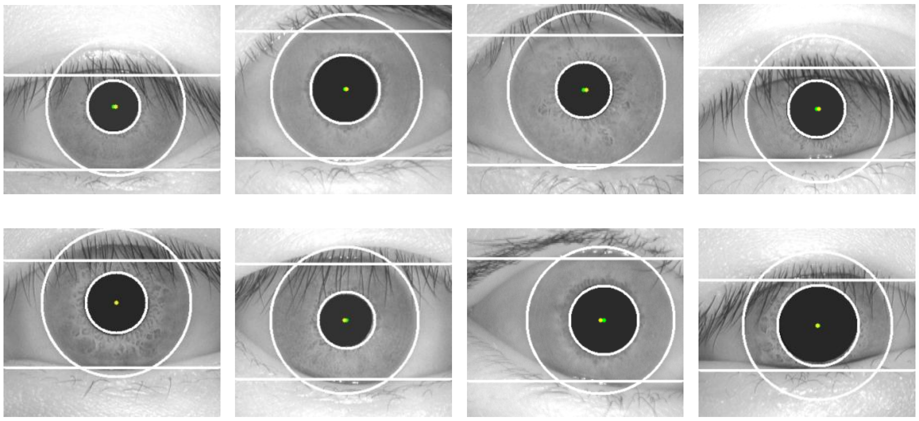


Figure 3.9: The examples of localized iris images from CASIA iris image database

3.1.5 Eyelashes, Reflection and Pupil Noise Removal

Eyelashes appear randomly inside the iris region. It is difficult to detect the eyelashes effectively. However, the eyelashes are observed to have low intensity values. A thresholding technique is implemented to segment the eyelashes accurately.

In general, iris imaging device uses near infrared light as illumination source. Near Infrared (NIR) illuminator is used to reveal complex textures for darkly pigmented irises. Reflection regions are characterized by high intensity values close to 255. A high threshold value is chosen to separate the reflection noise.

The pupil is not necessary a circular region. When the pupil boundary is approximately modelled as circle, some parts of the pupil will exist inside the normalized iris region as noise. Similar to eyelashes and reflection detection, an appropriate threshold is defined to remove the pupil noise. Figure 3.10 illustrates the normalized iris image with pupil, eyelashes and reflection noises.

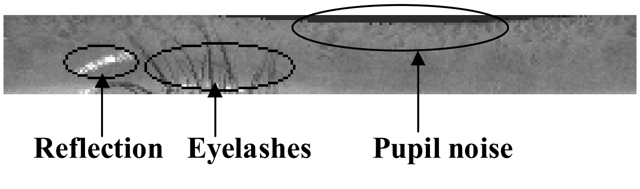


Figure 3.10: Normalized iris image with pupil, eyelashes and reflection noises

3.1.6 Normalization and Enhancement

Normalization remaps each pixel in the localized iris region from the Cartesian coordinates to polar coordinates. The non-concentric polar representation is normalized to a fixed size rectangular block. In the experiments, the size of the normalized iris image is 256×32 pixels. An iris image normalization example and the iris normalization technique are illustrated in Figure 3.11 and Figure 3.12 respectively.

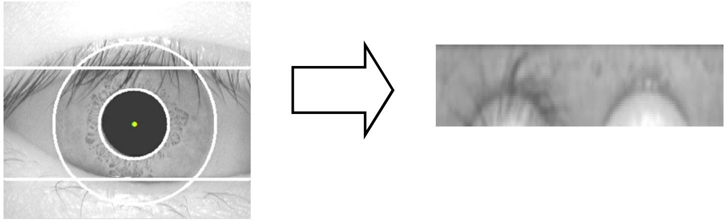


Figure 3.11: An example of iris image normalization

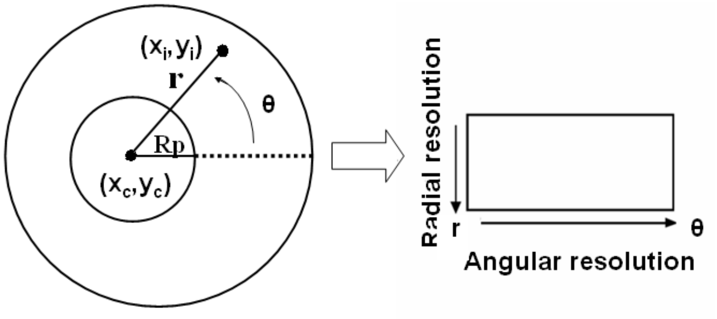


Figure 3.12: Normalization process

The pupil centre is used as the reference point and the remapping formulas are defined in Equation (2.3). The distance from pupil centre to the iris outer boundary, $R_n(\theta)$, which is a function of θ, is calculated using Equation (3.9) and (3.10).

$$d = \sqrt{(x_d - x_c)^2 + (y_d - y_c)^2} \tag{3.9}$$

$$R_n(\theta) = \sqrt{R_s{}^2 - (d \times \sin\theta)^2} + d \times \cos\theta \tag{3.10}$$

where *(x_d,y_d)* and R_s denote centre coordinates and radius of the iris respectively, and (x_c,y_c) is the centre coordinates of the pupil.

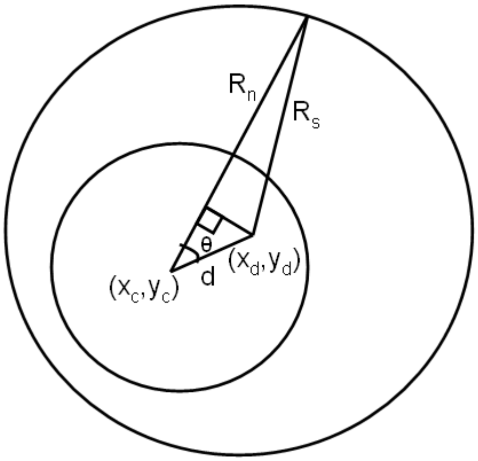


Figure 3.13: Geometry representation for iris normalization

The normalized iris image has low contrast and non-uniform illumination caused by the light source position. Local histogram equalization is applied to the normalized iris image to normalize brightness and increase the contrast of the image. Thus, the effect of non-uniform illumination is reduced and a well-distributed texture image is obtained. Figure 3.14 shows the enhanced iris image after local histogram equalization.

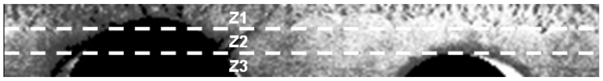


Figure 3.14: Enhanced iris image

The iris region is divided into three zones according to the characteristic of the iris texture. The three zones have the same size. Zones Z1, Z2 and Z3 are illustrated in Figure 3.14 and Figure 3.15.

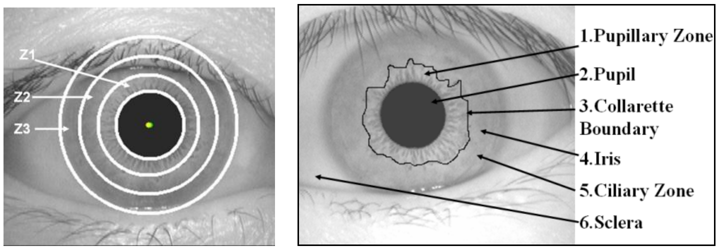


Figure 3.15:(a) Three zones of the iris image. (b) Pupillary zone, collarette boundary and ciliary zone of the iris

The iris is divided into two major regions: pupillary zone and ciliary zone. Zone Z1 is the pupillary zone which contains an abundance of textures. Zone Z2 is the collarette boundary that separates pupillary zone and ciliary zone. Zone Z3 corresponds to the ciliary zone with the flattest textures. The iris is categorized into different zones so that iris features can be analyzed and extracted more effectively.

3.2 Feature Extraction

3.2.1 1D Log Gabor filter

1D Log Gabor filter is used to extract the frequency information which represents the iris textures. First, the 2D normalized iris image is decomposed into 1D intensity signals. 1D Log Gabor filter is multiplied with each 1D intensity signal in the frequency domain as in Equation (3.11).

$$F_i(w) = G(w) \bullet I_i(w) \tag{3.11}$$

where $F_i(w)$ is the i^{th} row of filtered image, *G(w)* is the 1D Log Gabor filter and $I_i(w)$ is the i^{th} row of enhanced iris image in frequency domain.

The iris features are extracted based on its anatomical structure. It is observed that the inner zone, Z1 contains the finest iris texture. The variation of the fine texture indicates that it contains high frequency components. The high frequency information can be extracted using Log Gabor filter with high centre frequency, ω_0. The middle zone, Z2 has a larger block of texture due to the presence of the collarette boundary. It is processed using Log Gabor filter with a lower centre frequency. The flattest texture appears in the outer zone, Z3. The flat texture has low frequency components and therefore the coarsest Log Gabor filter with lowest centre frequency is used to capture the local details of the outer zone, Z3. By analysing the iris texture, the most significant iris features can be extracted with less redundancy.

After applying 1D Log Gabor filters, a series of real and imaginary numbers is generated. The phase information is quantized into four quadrants in the complex plane. Each quadrant is encoded with two bits phase information. Therefore, each pixel in the enhanced iris image is demodulated into two bits code in the template. The phase demodulation method used to encode iris template is illustrated in Figure 2.4.

1D intensity signals are used because the information density is the highest in the angular direction, which corresponds to the horizontal row in the enhanced iris image (Ma *et al.*, 2003).

The binary template is encoded for matching purpose. The template contains no amplitude information of the iris. Actual iris image cannot be reconstructed from the template. Therefore, the use of binary template would avoid the showing of original iris image.

Furthermore, the binary template can be stored and processed effectively using DSP technology. Because the binary template matching is computationally efficient, it is suitable for comparisons of millions of templates in large database. Therefore, this algorithm is suitable to be ported to the DSP embedded system.

3.2.2 1D Advanced Correlation filter

Since the information density is the highest in the angular direction, 1D advanced correlation filter is proposed to extract the iris features. First, 2D normalized iris image is decomposed into 1D intensity signals. One correlation filter is designed using the Fourier Transform of each 1D intensity signal.

Correlation filters capture the spatial frequency of the iris image. They are implemented in frequency domain array and 1D FFT is used to speed up the computation. The training of advanced correlation filters is illustrated in Figure 3.16. Correlation filters are designed for each iris class in the training set. All iris images that are in the same class are captured from the same eye.

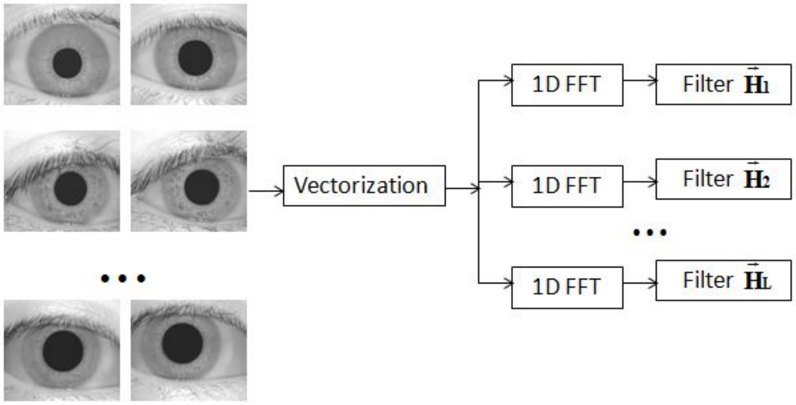

Vectorization
1D FFT
1D FFT
1D FFT
Filter $\vec{\mathbf{H}}_1$
Filter $\vec{\mathbf{H}}_2$
Filter $\vec{\mathbf{H}}_L$

Figure 3.16: Training of the 1D advanced correlation filters

The advanced correlation filters are designed using Optimal Trade-off Synthetic Discriminant Function (OTSDF), which is defined in Equation (2.8). After the advanced correlation filters are designed for each iris class in the training set, the feature vector of the test image can be extracted. The feature vector extraction is illustrated in Figure 3.17. The advanced correlation filter designed for the *l*-th iris class is denoted as $\vec{h}$. The correlation output is defined in Equation (3.12).

$$\vec{c}_i(n) = \vec{x}_i(n) * \vec{h}(n) \tag{3.12}$$

where $\vec{x}_i(n)$ is the 1D feature of the *i*-th test image, and * denotes cross correlation function of the two 1D signals. The output correlation plane should contain a sharp peak near the origin if the test image belongs to the same iris class.

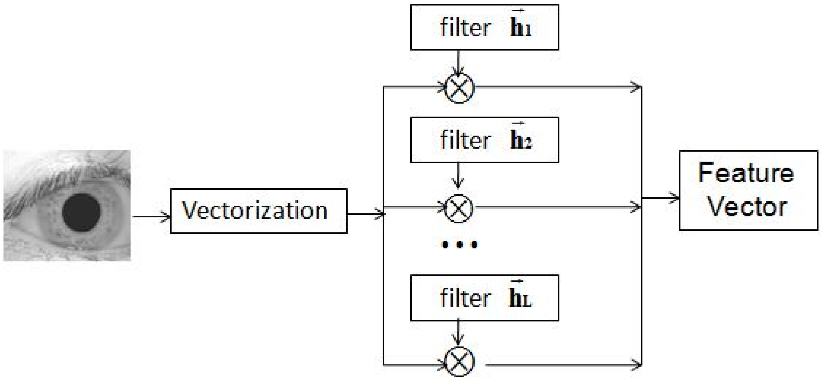


Figure 3.17: Feature vector extraction in 1D advanced correlation filters

OTSDF filter is chosen because it can optimize trade-off between average correlation energy (ACE) and output noise variance (ONV). By reducing the average correlation energy, the filter can produce a sharp peak at the output correlation plane for the training images. By reducing the output noise variance, it can tolerate the additive white noise.

The advanced correlation filter is chosen due to its rotation invariance property. It can compensate for the rotation of iris caused by head tilt or rotation of the camera. The location of correlation peak shows the relative shift between the test image and the filter. Furthermore, the correlation filter can tolerate the within-class variations. Therefore, the OTSDF filter can achieve higher recognition rate than other correlation filter even if the test images contain noises.

3.3 Template Matching

3.3.1 Total Hamming Distance

Hamming distance is used as the measure of dissimilarity between two binary templates encoded by 1D Log Gabor filter. Total Hamming Distance (THD) is the summation of Hamming distance from three different zones with different weightings.

$$THD = \alpha HD_1 + \beta HD_2 + \gamma HD_3 \qquad (3.13)$$

where HD_i, i=1,2,3 denotes the Hamming distance between two templates computed from three different zones, Z1, Z2 and Z3. α, β and γ represent the weightings of the Hamming distance for zone Z1, Z2, and Z3 respectively. The weightings must satisfy the condition defined in Equation (3.14).

$$\alpha + \beta + \gamma = 1 \qquad (3.14)$$

α, β and γ have decreasing weightings because the inner zone provides more texture information than the outer zones. Zone Z1 contains the most significant features that contribute to the recognition. Zone Z3 has less discriminating information because its texture is the flattest and it is often occluded by eyelids and eyelashes.

In order to account for rotational variance during imaging, the user template is shifted right and left bit-wise during matching. Each bit shifting in the template corresponds to rotation of the iris by an angle depends on the angular resolution. Ten Hamming distances are calculated from successive shifts of the user template. The lowest Hamming distance is chosen as the best match between the two templates.

Finally, a threshold is set to determine whether the templates are from same iris or different irises. If the Total Hamming Distance is lower than the predefined threshold, the user will gain access from the iris verification system. However, if the Total Hamming Distance is higher than the predefined threshold, the user's access will be denied and the user will be considered as an imposter.

3.3.2 Peak to Sidelobe Ratio

After cross correlation between the 1D advanced correlation filter and 1D intensity signal of the test image, the output correlation plane is generated. The peak sharpness of the correlation plane is measured using Peak to Sidelobe Ratio as defined in Equation (2.12). An authentic iris image produces a large PSR while an imposter image yields a very low PSR.

Total Peak to Sidelobe Ratio is the summation of all PSR from each row of the correlation output plane.

$$PSR = \delta \sum_{i=1}^{N/3} PSR_i + \varepsilon \sum_{i=N/3+1}^{2N/3} PSR_i + \zeta \sum_{i=2N/3+1}^{N} PSR_i \qquad (3.15)$$

where PSR$_i$ denotes the Peak to Sidelobe Ratio of the ith row of correlation outputs. N is the total number of rows of the output plane. δ, ε and ζ represent the weightings of the Peak to Sidelobe Ratio for zone Z1, Z2, and Z3 respectively. The weightings must satisfy the condition defined in Equation (3.16).

$$\delta + \varepsilon + \zeta = 1 \tag{3.16}$$

δ, ε and ζ have decreasing weightings because inner zone provides more texture information than the outer zones.

Finally, a threshold is set to distinguish whether the test image is from the same or different iris class. If the PSR exceeds the predefined threshold, the test image is from the same iris class. However, if the PSR is lower than the predefined threshold, the test image is from different iris class.

PSR locates the peak sharpness of the correlation output. The location of the correlation peak indicates the relative shift of the test image. Therefore, PSR can account for the rotational variance of the test iris image. Furthermore, PSR is invariant to any uniform scale changes in illumination of the test image.

3.4 Summary

In this chapter, the image preprocessing, feature extraction and template matching of an iris verification system are presented. For the iris localization stage, automatic thresholding and circular Hough Transform algorithms are applied to the iris image to detect the iris inner boundary. Iris outer boundary is detected using the proposed search region and intensity gradient methods. Upper and lower search regions are defined to limit the searching space when detecting the eyelids. Sobel edge detector and linear Hough Transform are implemented to obtain the eyelids position. A thresholding method is applied to remove the eyelashes, pupil and reflection noises. After the iris region is correctly segmented, the iris region is normalized into a fixed rectangular block before local histogram equalization is applied. The significant iris features are extracted from the enhanced iris image. 1D Log Gabor filter with different centre frequency is applied to three different iris zones to extract the features of the iris. Total Hamming Distance calculates the dissimilarity between two iris templates. For the 1D advanced correlation filter, cross correlation between the filter and the test image is performed to generate the correlation output. Peak to sidelobe ratio is used to measure the peak sharpness of the correlation plane. A threshold is set to distinguish whether the test image is from an authentic user or an imposter.

CHAPTER 4

RESULTS AND ANALYSIS

Section 4.1 presents the experimental setup for the iris verification system. The performance evaluation is described in Section 4.2. Section 4.3 explains the experimental results for iris localization. Experiment using 1D Log Gabor Filter and Total Hamming Distance is discussed in Section 4.4 while experiment using 1D Advanced Correlation filter and Peak to Sidelobe Ratio is discussed in Section 4.5. Finally, the summary is presented in Section 4.6.

4.1 Experimental Setup

The experiments are performed using CASIA iris image database version 1.0 (CASIA, 2007) as used in the work of Miyazawa K. *et al.* (Miyazawa K. *et al.*, 2006), Thornton J. *et al.* (Thornton J. *et al.*, 2005), Cui J. *et al.* (Cui J. *et al.*, 2004), Yuan X. *et al.* (Yuan X. *et al.*, 2007), Zhu Y. *et al.* (Zhu Y. *et al.*, 2000), and Huang J. *et al.* (Huang J. *et al.*, 2004). The prototype is run on a computer with Pentium 4 2.4 GHz CPU and 4GB RAM.

The main tools used for the experiments are MATLAB (R2006b) and Microsoft Visual Studio 2008. MATLAB is used for testing some of the algorithms. 1D Log Gabor filter response is plotted quickly in Matlab. Different Log Gabor filter parameters are evaluated to obtain the most suitable response for iris feature extraction. Moreover, circular Hough Transform and Fast Fourier Transform (FFT) are tested using Matlab to verify their functionality.

The iris verification algorithm is developed in C/C++ language. The software tool used is Microsoft Visual Studio 2008 with its richness in predefined functions and standard libraries.

For the computer vision applications, Intel® Open Computer Vision Library (Intel Corporation, 2007) is integrated into the algorithm. It is an open source computer vision library developed by Intel Corporation. It consists of various image processing algorithms, including edge detection, segmentation, object identification, histogram analysis, morphological operators and so forth.

4.2 Performance Evaluation

The recognition performance is evaluated by plotting the Receiver Operating Curve (ROC). The three measures for ROC curve are False Rejection Rate, False Acceptance Rate and Equal Error Rate.

1) False Rejection Rate (FRR): The fraction of the number of rejected genuine attempts divided by total number of genuine attempts.

2) False Acceptance Rate (FAR): The fraction of the number of falsely accepted imposter attempts divided by total number of imposter attempts.

3) Equal Error Rate (EER): The intersection point on the ROC curve where FAR is equal to FRR.

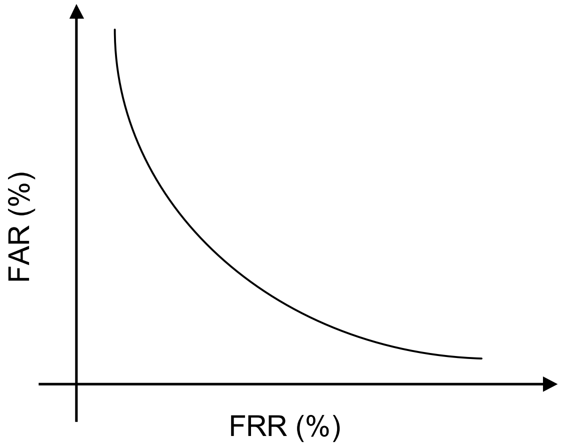


Figure 4.1: An example plot of ROC curve

Figure 4.1 shows the example plot of ROC curve. The values of FRR and FAR are threshold dependent. By adjusting the threshold, a list of FRR and FAR values is plotted on the ROC curve. The relation between FAR and FRR is illustrated in Figure 4.2.

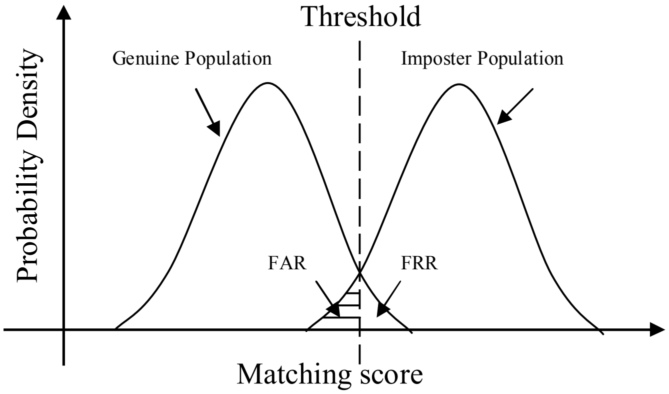


Figure 4.2: The relation between FAR and FRR and threshold value

If the threshold is shifted to the right, the FAR will increase but the FRR will decrease. However, if the threshold is shifted to the left, the FRR will increase but the FAR will decrease. A high FAR will increase the risk of granting access to unauthorized personnel. On the other hand, a high FRR will cause inconvenience to user access as the probability of rejecting a genuine attempt is increased.

Therefore, EER is a threshold independent performance measure that is commonly used. The highest performance security system has the lowest EER value.

4.3 Iris Localization results

The circular Hough Transform and proposed intensity gradient method have performed well on the iris localization. The proposed algorithm was evaluated using CASIA iris image database version 1.0 (CASIA, 2007). The results of iris inner and outer boundaries detection are shown in Table 4.1. Table 4.2 shows the results of upper and lower eyelids detection.

Table 4.1: Comparison of iris inner and outer boundaries detection rate with other algorithms

Method	Iris inner boundary detection rate	Iris outer boundary detection rate
Cui *et al.* (2004)	99.34%	99.34%
Xu *et al.* (2006)	98.42%	98.42%
Proposed	99.07%	98.68%

Table 4.2: Comparison of upper and lower eyelids detection rate with other algorithms

Method	Upper eyelid detection rate	Lower eyelid detection rate
Cui *et al.* (2004)	97.35%	93.39%
Xu *et al.* (2006)	98.52%	98.52%
Proposed	95.77%	95.37%

In Table 4.1 and Table 4.2, the iris boundaries and eyelids detection rates are observed by eyes because there is no standard method for evaluating the detection results. Since the iris segmentation results on CASIA iris image database version 1.0 is shown in Cui *et al.* (2004) and Xu *et al.* (2006), the performance of the proposed method is compared with their methods in Table 4.1 and Table 4.2. It can be concluded that the proposed method is comparable with their methods.

The detection rates of the iris inner and outer boundaries are 99.07% and 98.68% respectively. The false localization of iris inner boundary is caused by the pupil that is not a perfect circle. The algorithms try to find the best circle which fits the pupil boundary. Iris outer boundary is detected incorrectly due to the presence of eyelashes and the iris outer boundary is too near to the image boundary.

The accuracy of upper and lower eyelids detection are 95.77% and 95.37% as shown in Table 4.2. The eyelid boundaries are usually covered by eyelashes. Furthermore, it is difficult to model the eyelid boundaries using parabolic shape. The presence of skin fold also causes false eyelids detection.

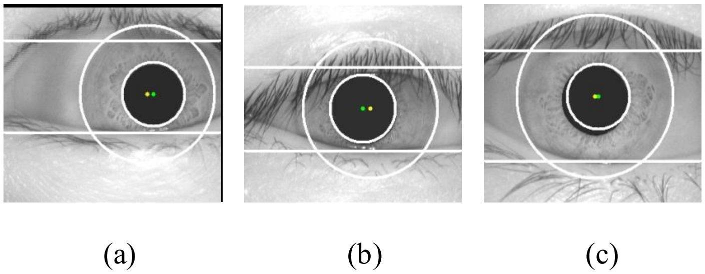

(a) (b) (c)

Figure 4.3: Inaccurate segmentation due to (a) iris outer boundary near to image boundary. (b) presence of eyelashes. (c) pupil is not a perfect circle

The proposed iris localization method is accurate because it utilizes the iris anatomical structure characteristics. The circular Hough Transform and intensity gradient methods are less complex and thus can reduce the computational cost.

4.4 Experiment using 1D Log Gabor Filter and Total Hamming Distance

For the feature extraction stage, 1D Log Gabor filter is used to extract the discriminating frequency information which represents the iris textures. For the template matching stage, Total Hamming Distance calculates the dissimilarity between two iris templates. The experiments were carried out by using CASIA iris image database version 1.0 (CASIA, 2007). Figure 4.4 shows the ROC curve plotted to evaluate the recognition performance using 1D Log Gabor filter.

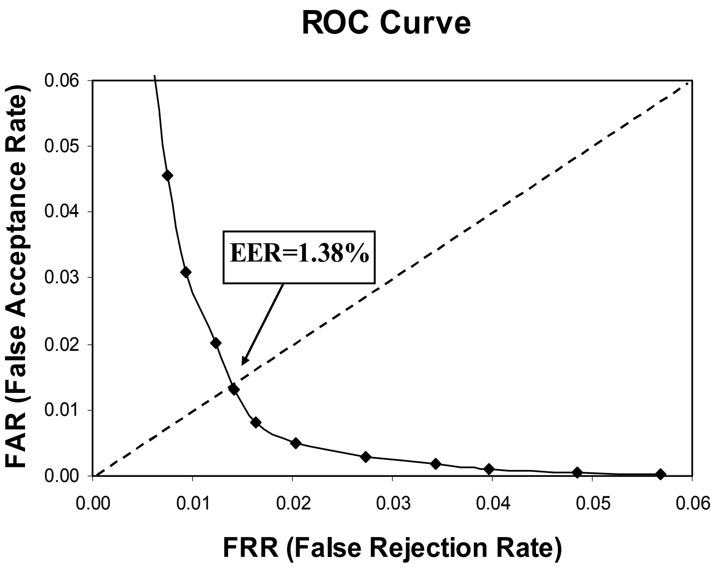
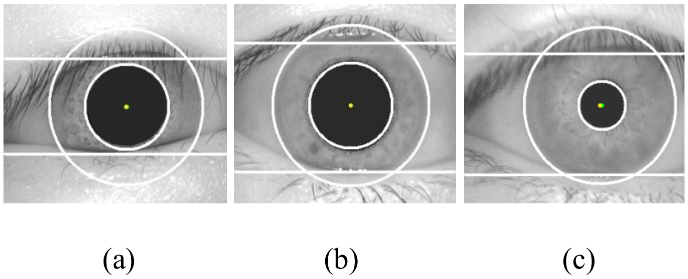


Figure 4.4: ROC curve for iris recognition results using 1D Log Gabor filter

ROC curve is plotted to measure the recognition accuracy. From the experimental results, the algorithm shows an overall accuracy of 98.62% with Equal Error Rate (EER) of 1.38%. It is noted that the result is not perfect due to the low quality of the iris images. The iris region is heavily occluded by eyelids and eyelashes or distorted much due to pupil dilation and constriction. Some of the iris images are defocused or are motion blurred as shown in Figure 4.5.



(a) (b) (c)

Figure 4.5: (a) A heavily occluded eye. (b) A defocused eye. (c) A motion blurred eye

4.5 Experiment using 1D Advanced Correlation filter and Peak to Sidelobe Ratio

For second method, 1D advanced correlation filter is applied on the test image to generate the correlation output. Peak to sidelobe ratio is used to measure the peak sharpness of the correlation plane. The experiments were carried out by using CASIA iris image database version 1.0 (CASIA, 2007). There are 108 iris classes with 7 images each. The first iris image from each class is selected as training image while the other 6 images are used as testing images.

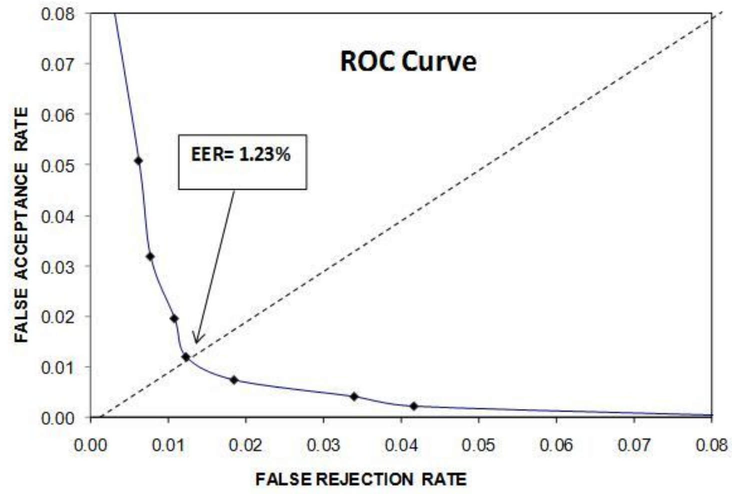


Figure 4.6: ROC curve for iris recognition results using 1D advanced correlation filter

From our research, we found that the 1D advanced correlation filter algorithm has better results than the 1D Log Gabor algorithm. The Equal Error Rate is only 1.23% and its performance degrades slowly due to the presence of noise and eyelid occlusion. This shows that the correlation filter is noise tolerant. The noise inside the iris image is assumed as additive white noise. The trade-off between output variance noise and average correlation energy can be optimized to obtain the best recognition accuracy.

4.6 Summary

In this chapter, the experimental results for iris localization and recognition are presented. The proposed iris localization method is comparable to that of Cui *et al.* (2004) and Xu *et al.* (2006). The experimental results show that the proposed iris localization method is effective. It uses a less complex algorithm but produces results comparable to the other algorithms. The proposed 1D Log Gabor filter and Total Hamming Distance method have achieved a high recognition rate of 98.62%. The recognition rate for the proposed 1D advanced correlation filter and Peak to Sidelobe ratio is 98.77%. In conclusion, the innovative iris recognition algorithms based on texture analysis and advanced correlation filter presented in this research are comparable to existing algorithms.

CHAPTER 5

BLACKFIN BASED IMPLEMENTATION AND OPTIMIZATION OF IRIS VERIFICATION SYSTEM

The first section of this chapter addresses the challenges of implementing the DSP based iris verification system. The second section describes the design flow for the embedded iris verification system. The implementation detail of the iris verification system using Blackfin processor is also presented. The iris verification algorithm is ported from PC onto the Blackfin evaluation board. The optimization strategies for the DSP system are discussed in detail. C/C++ source code tuning, compiler's pragmas, and conditional code optimization are implemented to obtain the best code execution performance for the processor. The performance profiles of the iris verification system are shown in the last section. The performance is evaluated in terms of speed before and after optimization.

5.1 DSP Design Challenge

It is a challenging task to develop an iris verification system on a digital signal processor (DSP) platform. The porting of the iris verification algorithm onto a DSP platform must be carefully designed to meet stringent requirements, including accuracy and speed. The algorithm needs to be optimized for DSP without compromising performance. This is the main challenge in this project, since it is difficult to meet all the requirements without having to make trade-offs. For example, most DSP platform has very limited amount of computation power and memory. It is necessary to make trade-offs between the speed and memory.

To cope with the design challenges, an integrated software development and debugging environment (IDDE) is used in the DSP development. It consists of C/C++ compiler, assembler, expert linker, loader, run-time library, simulator and hardware emulator. The development tools help to manage the project efficiently and meet changing development needs. These tools provide an easy-to-use interface which speeds up development time and reduces design error.

5.2 Embedded Systems Design Flows

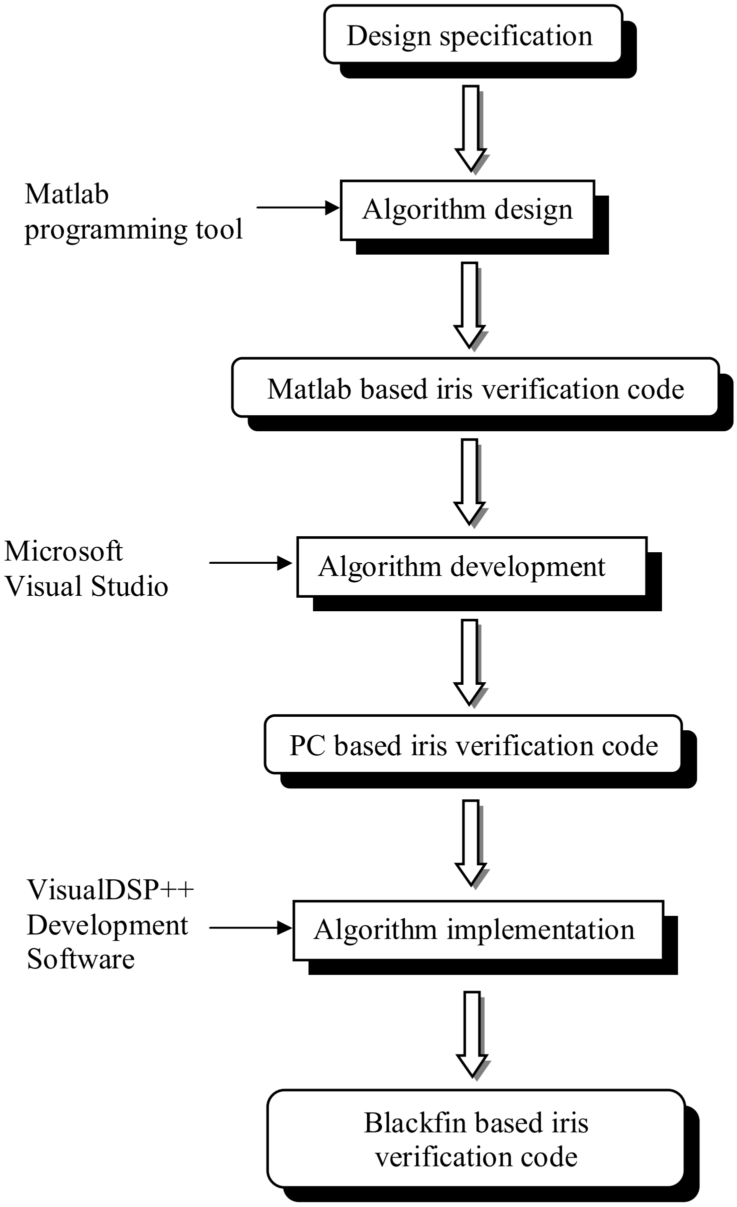

Design specification
Matlab programming tool
Algorithm design
Matlab based iris verification code
Microsoft Visual Studio
Algorithm development
PC based iris verification code
VisualDSP++ Development Software
Algorithm implementation
Blackfin based iris verification code

Figure 5.1: Design flow for embedded iris verification system

Figure 5.1 illustrates the design flow for the embedded iris verification system. The first design phase is the algorithm design and testing. Iris localization, feature extraction and template matching algorithms are developed to meet the specification. In order to implement a robust algorithm, prominent algorithms available from open source are investigated. The algorithms are analyzed and tested using Matlab programming language. The purpose of exploring algorithms using Matlab is to conceptualise and evaluate the algorithms without having to focus on the implementation details.

At design phase II, the iris verification system is implemented on the PC platform. Matlab code is rewritten in C/C++ and debugged using Microsoft Visual Studio 2008. Major works include:

1. Study of fundamental algorithm such as the Fast Fourier Transform (FFT) and rewriting it into C/C++ code.
2. Selected functions from OpenCV library are integrated with the algorithm to perform simple image processing operations. For instance, Hough circle transform function is called to detect the iris inner boundary. After the algorithm is developed, its performance will be evaluated in terms of accuracy and speed. Once the system is verified, it will be implemented on the Blackfin platform for real world application.

At the final design phase, the DSP implementation and optimization are completed using VisualDSP++ software. When porting the iris verification algorithm onto the DSP platform, the iris verification recognition rate of the Blackfin implementation must be equal with the PC-based system. Finally, processor architecture and memory layout are exploited in order to optimize the Blackfin implementation. The software and hardware optimization must be able to reduce the processor cycles to achieve faster verification time without compromising performance.

5.3 Implementation on Blackfin Processor

In the iris verification implementation, the PC-based iris verification algorithm is ported into the Blackfin processor. The mapping of C/C++ source code into Blackfin processor must be completed without compromising performance. First, the code is compiled and run successfully in VisualDSP++. Then, the result of the Blackfin-based algorithm is compared with the PC-based result. This is to verify that the PC platform and the Blackfin platform produce the same result.

For the implementation on Blackfin processor, the verification time is less than one second to process a 320×280 grayscale iris image. The iris verification stages consist of iris segmentation, normalization and template matching. The 1D Log Gabor filter and Total Hamming Distance method are chosen to be implemented on Blackfin processor. This is because the encoded binary template is only 16 K bytes and it is possible to avoid the showing of the actual iris images. The binary template is small in size and can be stored efficiently. The advantage of using Total Hamming Distance is the fast matching speed. Therefore, 1D Log Gabor filter and Total Hamming Distance are used for implementing iris recognition device using DSP technology.

Figure 5.2 illustrates the system architecture of the embedded iris verification system. The embedded iris verification system is implemented on ADSP-BF561 EZ-KIT LITE evaluation board, which is based on the Blackfin ADSP-BF561 dual-core processor. The system comprises of three parts, video decoding, DSP processing and video encoding.

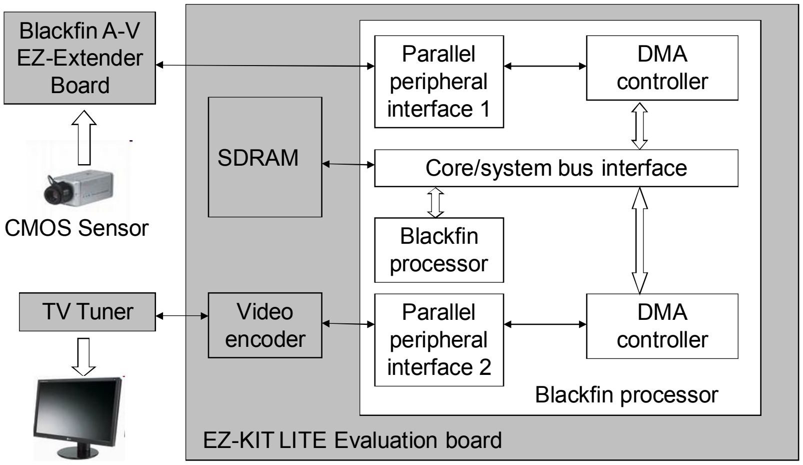

Blackfin A-V EZ-Extender Board
CMOS Sensor
TV Tuner
SDRAM
Video encoder
Parallel peripheral interface 1
DMA controller
Core/system bus interface
Blackfin processor
Parallel peripheral interface 2
DMA controller
Blackfin processor
EZ-KIT LITE Evaluation board

Figure 5.2: System architecture of the embedded iris verification system

5.3.1 Video Decoder

The decoding part captures iris images using Complementary Metal Oxide Semiconductor (CMOS) sensor and transmits data to the DSP board via Blackfin A-V EZ-Extender card. First, the iris image is captured using CMOS image sensor. The Blackfin A-V EZ-Extender decodes the image into digital format. Parallel peripheral interface (PPI) serves as a video interface between the video decoder and the Blackfin processor. The data is transferred from the PPI to SDRAM via Direct Memory Access (DMA) channel and external bus interface unit (EBIU). The data is buffered in the SDRAM for further image processing. However, in this project, the experiment loads offline iris images from the dataset into the SDRAM for verification purpose. The executable code and data is loaded from PC into the EZ-KIT Lite evaluation board via USB interface.

5.3.2 DSP Processing

The iris image data stored in the SDRAM is transferred into core for processing via EBIU. The iris images are verified using the iris verification code which is stored in Level 1 (L1) and Level 2 (L2) on-chip instruction memories. On-chip memory supports instruction execution at the fastest speed with minimum latency. DSP is well suited to compute the Fast Fourier Transform (FFT) algorithm which is required in the feature extraction stage. 16-bit fixed point DSP is used because it runs at higher clock frequency with lower power consumption than a floating point DSP.

5.3.3 Video Encoder

In the encoding part, the localized iris images and the verification results are transferred to the ADV7179 video encoder. The images are transferred in NTSC frame format via the video channel. Multiple peripherals are chained to build up the video channel. The peripherals include SDRAM, EBIU, DMA channel and PPI. The video encoder converts the digital data into its corresponding analog TV signal output. A TV tuner is used to convert the analog signal into digital video format for display on the computer monitor. Figure 5.3 displays the Blackfin-based prototype of the iris verification system.

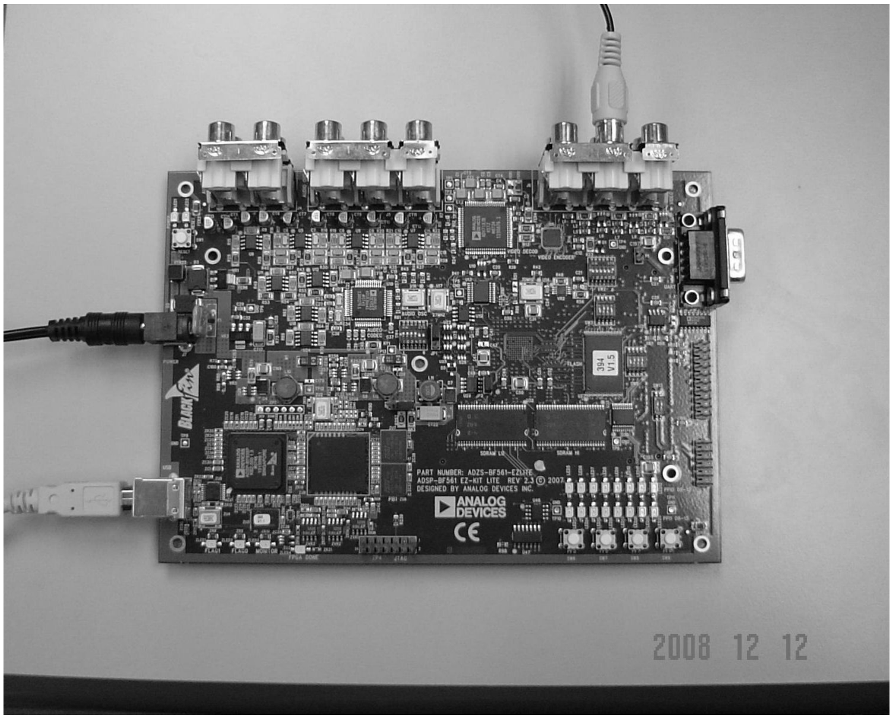


Figure 5.3: Blackfin-based iris verification system

5.4 Optimization on Blackfin Processor

In order to achieve less than one second execution time, the DSP-based iris verification algorithm needs to be optimized. Code optimization must be carried out without altering the original code behaviour. Using VisualDSP++, a statistical profiler identifies the areas of the program that spend most of the processing time. By optimizing these areas of the program, the best code execution performance can be obtained.

5.5 C/C++ Source Code Tuning

At this stage, the iris verification algorithm has been successfully compiled and run in VisualDSP++. However, the algorithm has not achieved the targeted performance, where the verification time should take less than one second. The objective of C/C++ source code tuning is to present the algorithm in a way that gives the optimizer visibility of the data and operations. Hence, the algorithm can be safely manipulated to obtain the best execution performance (Analog Devices Inc., 2003).

VisualDSP++ compiler generates efficient code from the straightforward C code. In default setting, the C/C++ compiler is in non-optimized mode. The automatic optimization can be enabled by adding -0 compiler switch to the compiler invocation. The compiler optimization can be configured to optimize for speed, or code size, or a factor between speed and code size. The output of the optimizer is the correct code that executes faster and has smaller size.

Tuning C/C++ source code begins by identifying the areas of the program that are computationally intensive. Statistical profiler provided in VisualDSP++ is used to find those areas of code. The areas of program that are most frequently executed are optimized to provide the largest gains.

Next, function inlining is used for small and frequently-executed functions. It avoids program flow latencies, function entry and exit instructions, and parameter passing overheads. As illustrated below, the variable that has a fixed value throughout the program is declared as constant.

```
void evRFFT(const uchar *in, EvComplex *out, const EvComplex w[],
int N )
{
        int i, j, k, m;
        const int N2 = N >> 1;
        const float Wn = 2*PI/N;
        int Len1, Len2, baseidx;
        ...

}
```

For the source code tuning, condition branches such as if-else construct, are relocated outside the loops. If a loop has conditional code, long control-flow latencies may be incurred if the decision has to branch against the compiler’s prediction. An example of conditional statement inside a loop is taken from the iris verification algorithm.

```
for( i = 0; i <= n/2; i++ )
    {
        if( type == 1)
        {
               ...
        }
        else
        {
               ...
        }
    }
```

After code tuning, the conditional code is relocated outside the *for* loop. The optimized code is illustrated below.

```
if( type == 1)
   {    for( i = 0; i <= n/2; i++ )
         {
               ...
         }
   }
else
   {
        for( i = 0; i <= n/2; i++ )
         {
               ...
         }
   }
```

The nested loops are rewritten so that the inner loop iterates more than the outer loop. This is because the optimizer focuses on improving the performance of the inner loop. Iris verification algorithm has a few nested loops where the outer loop iterates many times while the inner loop iterates a few times only. An example of similar nested loop is illustrated below.

```
for( degree=0; degree <360; degree+= 1)
        {      for( r = radius ; r<radius +3; r++ )
               {
                       ...
               }
        }
```

The nested loop is rewritten so that the inner loop has more iterations than the outer loop.

```
for( r = radius ; r<radius +3; r++ )
        {      for( degree=0; degree <360; degree+= 1)
               {
                       ...
               }
        }
```

5.6 Level-1 Code Optimization

The iris verification algorithm after code tuning is benchmarked using a statistical profiler. Statistical profiler identifies the areas of the program that are most frequently executed. Optimizing these areas of code would provide the largest gain. The optimization strategy focuses on the speed performance. Optimization is carried out until the targeted speed performance is reached.

From the statistical profiling, it is found that the algorithm spends most of its execution time inside for loop construct. Therefore, loop optimization is carried out in order to achieve largest performance gain. The optimization of a loop reaps a benefit for every iteration of that loop.

5.6.1 Loop Optimization using Pragmas

The Blackfin processor allows the code to execute more than one iteration in parallel. Loop vectorization is performed by calling the pragmas provided by the compiler. Loop optimization pragma (#pragma vector_for, #pragma all_aligned, #pragma loop_count) is placed before the loops to guide the compiler to perform optimization. The examples of loop statements are for, while and do statements.

#pragma vector_for assures the compiler that there are no loop-carried dependencies. It tells the compiler that all the iteration may be run in parallel. The compiler checks various properties of the loop and vectorizes the loop whenever it is safe to do so.

#pragma all_aligned is inserted so that every pointer variable in the loop is aligned on a word boundary at the beginning of the first iteration. An example of pragma keyword applied in the code is shown below.

```
#pragma vector_for
#pragma all_aligned
    for(i=0;i<AREA; i++)
    {
           ...
    }
```

#pragma loop_count tells the compiler about the loop's iteration counts. The pragma includes information of minimum count, maximum count and a multiple of constant times. Using the iteration count range, the compiler makes more reliable decisions for the optimization strategy. It decides whether the loop is worth unrolling and whether the loop needs to be generated for odd iterations. If the loop count is a multiple of some constant, the loop can be vectorized without the need for conditionally-executed iterations. The section of code using loop count pragma is shown below.

```
for( i = 0; i <= width - 4; i += 4, s += 4 )
    {
          ...
         #pragma loop_count (5,10)
         for( k = 1, j = cn; k <= ksize2; k++, j += cn )
         {
               ...
         }
          ...
    }
```

5.6.2 Optimizing Conditional Code

A pipelined processor executes instructions at the fastest speed when the program is relatively linear. If the conditional statement is placed in the program, it will incur branch latency caused by pipeline flushing. The compiler attempts to decide whether the condition is usually branched to true or to false. It arranges for the most efficient path of execution which is most commonly executed.

Blackfin processor has incorporated static branch prediction for conditional statements. Programmer can use expected_true and expected_false built-in functions to control the compiler's behaviour. These functions tell the compiler which condition is most likely to be executed. For example, the following code checks whether the region of interest is inside an image.

```
for( i = 0, rad=0; i <WIDTH; i++, rad+= THETA)
        {
        ...
        for( r2 = rpi3,j=0 ; j < HEIGHT; r2+= r_step,j++ )
                {
                if(expected_true(y>gray->origin && y<gray->height))
                        ...
                }
        ...
        }
```

The condition `y>gray->origin && y<gray->height` is expected to be true in most cases. Then, the expected_true built-in function is placed before the condition statement. The compiler arranges for the true cases to be executed most efficiently.

5.7 Level-2 Code Optimization

The hierarchical memory structure of Blackfin processor can be utilized to minimize the latency caused by memory access.

Due to the limited amount of internal memory, the iris images, templates and masks are stored in the SDRAM. SDRAM provides a 128MB of memory but operates at longer latency than the on-chip memory. The high speed processor is idle while fetching data from the external memory. This may leads to slow execution speed of the program. Fortunately, the iris verification algorithm can be fit into the L1 and L2 instruction memories. Data cache and memory layout optimization are implemented for faster accessing of the data in the external memory.

5.7.1 Data Cache

The data stored in the L1 internal memories can be fetched with single-cycle accesses. However, there is limited amount of L1 and L2 on-chips memories. There are only 64K bytes of data SRAM in each processor core. In the iris verification system, the iris images, templates and masks are too large to fit into the internal memory. These data are stored in the SDRAM with 128M bytes of memory but operates at longer latency.

Therefore, the data cache is enabled to transfer data from the SDRAM to L1 memory for faster processing. The most frequently used data is transferred to the L1 internal memory by the cache controller. Data in the cache can be accessed efficiently within a processor single cycle.

Blackfin ADSP-BF561 has 64K bytes of L1 data memory in each core. 32K bytes of the data memory is set as the data SRAM. The remaining 32K bytes of data memory can be configured either as data SRAM or data cache. In the iris verification system, 16K bytes of data bank A is configured as data cache.

Static cache management is implemented to manage the cache. Cacheability protection lookaside buffers (CPLBs) table are also configured with its descriptors to define cacheable external memory. The CPLB descriptors are fitted with 32 fixed entries, which cover the entire cacheable external memory, from 0x00000000 to 0x2000000 (32 M bytes). Figure 5.4 illustrates the optimized L1 memory layout for the iris verification system.

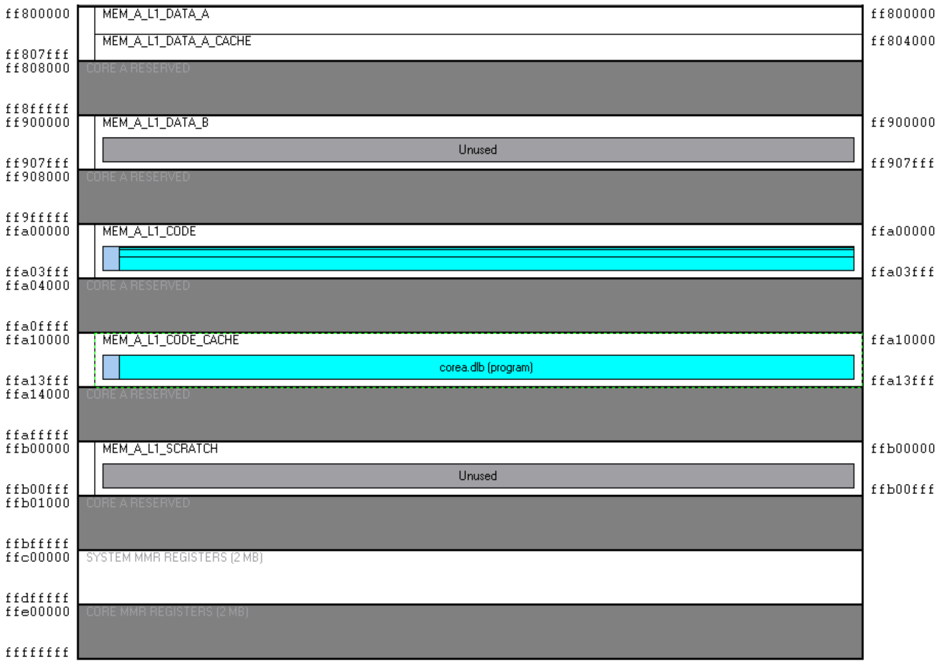


Figure 5.4: Internal L1 memory layout after the data cache is turned on using Expert Linker

5.7.2 Memory Layout Optimization

The Blackfin ADSP-BF561 supports a hierarchical memory model with different size and performance parameters. The L1 memory can be accessed in single cycle. L2 memory provides larger space but with higher latency. SDRAM has the longest memory access latency. The memory layout can be utilized to achieve better performance.

By default, there is one system heap used by calls to allocation functions like malloc in C. Multiple heaps are implemented in the iris verification system. Multiple heaps are needed to serve allocations either using fast-but-scarce memory or slower-but-plentiful memory. Frequently accessed data structures are allocated in the L1 internal memory. If the data structure is too large, it will be allocated in the SDRAM during compile time.

Iris images are stored in SDRAM sub-bank 0, while the iris templates, masks and NTSC frame are mapped into SDRAM sub-bank 1. Blackfin processor supports two memory operations in one cycle if the addresses are located in two different sub-banks. The memory layout after optimization is shown in Figure 5.5.

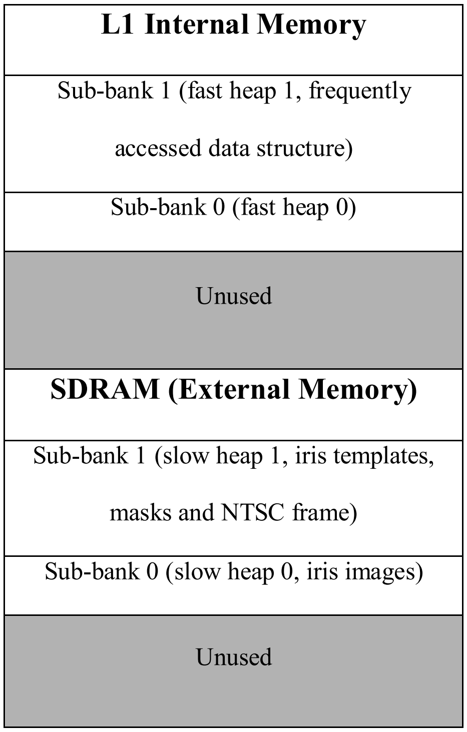

L1 Internal Memory
Sub-bank 1 (fast heap 1, frequently accessed data structure)
Sub-bank 0 (fast heap 0)
Unused
SDRAM (External Memory)
Sub-bank 1 (slow heap 1, iris templates, masks and NTSC frame)
Sub-bank 0 (slow heap 0, iris images)
Unused

Figure 5.5: Optimized memory layout for iris verification system

5.8 Experiments

The iris verification code is stored in Level 1 (L1) and Level 2 (L2) on-chip memories. On-chip memory supports instruction execution at the fastest speed with minimum latency. The test image is loaded from the PC to the evaluation board. The test image size is 320×280. For verification purpose, the iris templates and iris masks are loaded into the SDRAM due to their large sizes.

5.9 Performance Profile

The performance profile for the original iris verification algorithm is gathered using statistical profiler tool. The DSP runs at clock frequency 525 MHz. Table 5.1 shows the performance profile of the iris verification system before and after optimization. Before optimization, the experimental results reveal that the iris verification system has not achieved the speed requirement of less than one second. It shows that the iris segmentation stage consumes 67.2% of the total computation time. This is because the iris segmentation scans the whole image for the iris region. Therefore, optimization is focused on the iris segmentation stage.

Table 5.1: Comparison of iris inner and outer boundaries detection rate with other algorithm

Iris Verification Stages	Before Optimization		After Level-1 Optimization		After Level-2 Optimization		Optimization Gain (%)
	Number of Cycles	Comp. Time (ms)	Number of Cycles	Comp. Time (ms)	Number of Cycles	Comp. Time (ms)	
Iris segmentation	519,440,248	989.41	363,003,760	691.44	134,215,137	255.65	74.16
Iris normalisation	59,208,968	112.78	56,923,634	108.43	34,935,278	66.54	41.00
Iris verification	194,721,961	370.89	120,643,451	229.80	80,344,065	153.04	58.74
All stages	773,371,177	1473.08	540,570,845	1029.67	249,494,480	475.23	67.74

First, the compiler optimization is turned on and the source code tuning is implemented. In Level-l optimization, the strategies include loop optimization using pragmas and conditional statement optimization. The computation time for iris verification system after Level-1 optimization is shown in Table 5.1. The experimental results confirmed that the compiler optimization produces an equivalent code with higher performance. The total computation time of iris verification algorithm drops by about 30%. The total computation time is 1.029 second, but it has not met the speed requirement of the iris verification system.

Table 5.1 also shows the result of the iris verification system after Level-2 optimization. It is noted that the iris verification algorithm has achieved the desired speed performance. The total verification time is 470.23 milliseconds. From the experimental results, it is obvious that the iris verification algorithm has achieved significant speed improvement, by saving up to 554.44 milliseconds (291,076,365 cycles). It is 53.85% faster than the iris verification system before Level-2 optimization. Cycle saving at Level-2 optimization is attributed to the cache enabled for memory access. The performance bottleneck is caused by the frequent access to the slower external memory. After the data cache is enabled, the data is transferred from the SDRAM to L1 memory for faster processing.

For the implementation on Blackfin platform, the PC-based iris verification algorithm is ported to benchmark accuracy. The PC-based iris verification algorithm shows an overall accuracy of 98.62%. Multiple regression tests are performed during the implementation and optimization of iris verification system. This is to ensure that implementation and optimization of iris verification algorithm does not alter the original code behaviour.

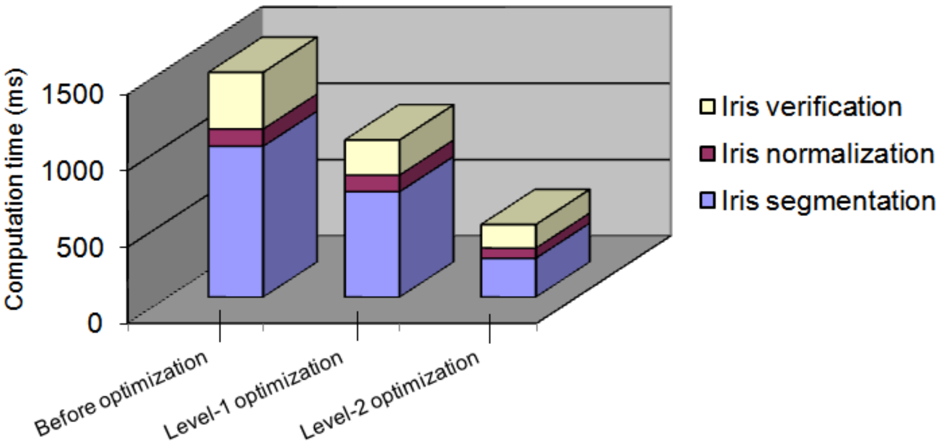
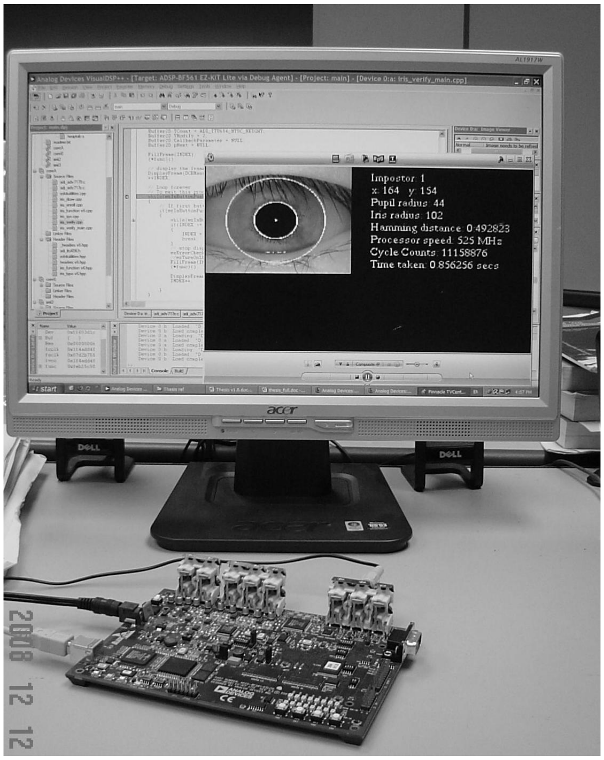


Figure 5.6: Execution time for all iris verification stages before and after optimization



```
Impostor: 1
x: 164   y: 154
Pupil radius: 44
Iris radius: 102
Hamming distance: 0.492823
Processor speed: 525 MHz
Cycle Counts: 11158876
Time taken: 0.856256 secs
```

Figure 5.7: Final prototype of the iris verification system

CHAPTER 6

CONCLUSION AND FUTURE WORKS

6.1 Conclusion

In this project, a robust iris verification algorithm is developed and implemented on the Blackfin DSP. First, the iris verification algorithm is developed to process the grayscale iris images of 320×280. For the iris localization stage, circular Hough Transform algorithm is applied to the iris image to localize the iris inner boundary. Iris outer boundary is detected using the proposed intensity gradient method. Upper and lower search regions are proposed to limit the searching space when detecting the eyelids. Sobel edge detector and linear Hough Transform are implemented to detect the eyelids. A thresholding method is applied to remove the eyelashes, pupil and reflection noises. An effective iris segmentation method for iris recognition system is proposed in this thesis. It proposes a solution for compensating all types of noises to achieve higher recognition rate.

Second, after the iris region is correctly segmented, the iris region is normalized into a fixed rectangular block before image enhancement algorithm is applied. The significant iris features are extracted from the enhanced iris image. 1D Log Gabor filter and 1D advanced correlation filter are employed to capture the spatial frequency of the iris images. Total Hamming Distance and Peak to Sidelobe Ratio are used to classify the matching score from 1D Log Gabor filter and 1D advanced correlation filter respectively. For 1D Log Gabor filter method, the encoded binary template is only 16 K bytes and is able to avoid storing of actual iris images. The binary templates are suitable for implementing iris recognition devices using DSP technology. The approach has achieved a high recognition rate up to 98.62% on CASIA iris image database version 1.0 (CASIA, 2007). For 1D advanced correlation filter method, the recognition rate is 98.77% on the same database.

Third, the iris verification prototype is implemented on the EZ-KIT Lite evaluation board. The iris verification algorithm is successfully ported into the evaluation board without compromising performance. Integrated software development and debugging environment is used to manage the project efficiently. The IDDE tools help to develop the project from algorithm design to the DSP-based implementation. The Blackfin-based platform is evaluated to ensure that code behaviour is the same as that of the PC-based version.

Finally, three phases of optimization are implemented to ensure that the iris verification system conforms to the speed requirement. The areas of the program that take up most of the processing time are optimized to obtain the best code execution performance. Loop optimization is carried out in order to achieve largest performance gain. The optimization of a loop reaps a benefit for every iteration of that loop. Hardware level optimization techniques implemented includes data cache and memory layout optimization. The optimized iris verification system has achieved the desired speed performance.

The total verification time is 470.23 milliseconds to process an iris image. The total optimization gain is as high as 67.74%.

6.2 Thesis Contribution

The main contribution is the implementation of a DSP-based iris verification system prototype. A fast and accurate iris verification system is implemented within a compact hardware module. It is suitable for access control application. The execution time for the iris verification is less than one second. The generated template size is 16 K bytes, which is suitable for DSP implementation.

An effective iris segmentation method for iris recognition system is proposed in this thesis. Circular Hough Transform is used to locate the iris inner boundary. The proposed method makes use of the search regions to locate the iris outer boundary and eyelids. The iris segmentation localize iris region correctly for identification purpose.

A fast and robust iris verification algorithm is developed. The algorithm has high accuracy and is able to remove noises, such as eyelids, eyelashes, pupil and illumination. Furthermore, it enhances the iris image by improving its contrast using histogram equalization. The algorithm also compensates for iris image translation, scale and rotation variance.

Implementation framework for iris verification system is presented. The implementation workflow is applied to map the algorithm onto the DSP without compromising performance. Software and hardware toolchains are employed to speed up the workflow. The toolchains can help to reduce error and manage the project effectively.

The segment of program that consumes most of the computation time is identified before the optimization is applied. C/C++ source code tuning is done to obtain the best code execution performance. The optimizer is used to generate efficient code from the straightforward C code. Function inlining is applied to small and frequently-executed functions. Conditional branches such as if-else construct, are relocated outside the loops to avoid latency. The nested loops are rewritten so that the inner loop iterates more than the outer loop.

Loop optimization is carried out in order to achieve the largest performance gain. The optimization of a loop reaps a benefit for every iteration of that loop. Loop vectorization is performed by calling the pragmas provided by the compiler. Static branch prediction produces the most efficient instruction sequence by choosing the most commonly executed path.

The hierarchical memory structure is utilized to minimize the latency caused by memory access. Data cache is enabled to transfer most frequently used data from SDRAM to L1 memory for faster processing. The memory layout is utilized to achieve higher speed performance. Multiple heaps are defined to serve allocations either using fast-but-scarce memory or slower-but-plentiful memory. Data are stored in two different memory blocks because Blackfin processor supports two memory operations in one cycle if the addresses are located in two different blocks.

6.3 Future Works

Although much work has been accomplished in this research, there are still some improvements for future work. The algorithm was evaluated on CASIA iris database version 1.0 (CASIA, 2007). The database has limited number of iris images and the images are captured under controlled condition. The CASIA iris image database contains irises from Asian population only. More experiments with more iris images from diversified populations can be conducted in the future. The experiments will be conducted in different realistic environments to develop more robust iris verification algorithms.

In the iris verification system, the image frame is downloaded from PC to EZ-KIT LITE evaluation board for testing and debugging. The system operates in offline mode. However, it is preferable to use on-board image sensor for realistic applications.

In order to meet speed and power consumption requirements, heterogeneous DSP/FPGA architecture is the trend in high-performance embedded computing. DSP can focus on performing iris segmentation and FFT algorithms. Meanwhile, the matching algorithm can be mapped on FPGA modules to achieve parallelism by pipelined architecture. Thus, millions of comparisons between templates in large database could be performed in seconds. Hardware and software co-design allows designers to customize the hardware and software architecture to meet the requirements of the applications.

Another area to work on is the real-time application of the iris verification system. Real Time Operating System (RTOS) can be mapped to Blackfin DSP board. The examples of RTOS used in Blackfin processor are Fusion RTOS, uClinux, and Integrity. RTOS can schedule processes on real time system. It manages the hardware resources well to meet the strict timing requirement. RTOS allows dual-core Blackfin processor to perform pipeline algorithms on both cores simultaneously to speed the processing of the iris verification algorithm.

BIBLIOGRAPHY

Analog Devices Inc. (2003). Tuning C Source Code for the Blackfin® Processor Compiler. *Engineer-to-Engineer Note, EE-149*, Norwood.

Analog Devices Inc. (2007a). ADSP-BF561 Blackfin Processor Hardware Reference Rev. 1.1, Norwood, USA, 82-000561-01.

Analog Devices Inc. (2007b). VisualDSP++ 5.0 C/C++ compiler and library manual for Blackfin processors Rev. 5.0. Norwood, USA.

Analog Devices Inc. (2007c). VisualDSP++ 5.0 User's Guide Rev. 3.0. Norwood, USA, 82-000420-02.

Analog Device Inc. (2007d). *Qritek Chooses Blackfin to Drive Iris Authentication Algorithm.* URL: http://www.analog.com/processors/blackfin/overview/customerStories/qritek/qritekIndex.html. Accessed on 6th November 2007.

Analog Devices Inc. (2008a). ADSP-BF561 EZ-KIT LITE Evaluation System Manual. Revision 3.2, Part Number 82-000811-01.

Analog Devices Inc. (2008b). VisualDSP++ 5.0 Linker and utilities manual Rev. 3.1. Norwood, USA, 82-000420-03.

Analog Devices Inc. (2008c). Software Development Kit (SDK). URL: http://www.analog.com/en/embedded-processing-dsp/software-and-reference-designs/content/software_development_kit_downloads/fca.html. Accessed on 20th March 2008.

Boles W. and Boashash B. (1998). A human identification technique using images of the iris and wavelet transform. *IEEE Transactions on Signal Processing*, vol. 46, no. 4, pp. 1185-1188.

Chen Y., Dass S., and Jain A. (2006). Localized Iris Image Quality Using 2D Wavelets. *Proceedings of International Conference on Biometrics*, LNCS 3832, pp. 373-381.

Chong S.C., Teoh A.B.J., and Chek D.L.N. (2005). Iris Authentication Using Privatized Advanced Correlation Filter. *LNCS 3832*, pp. 382-388.

Cui J., Wang Y., Tan T., Ma L., and Sun Z. (2004). A Fast and Robust Iris Localization Method Based on Texture Segmentation. *SPIE Defense and Security Symposium*, vol. 5404, pp. 401-408.

Daugman J. (1993). High Confidence Visual Recognition of Persons by a Test of Statistical Independence. *IEEE Tans. Pattern Analysis and Machine Intelligence*, vol.15, pp.1148-1161.

Daugman J. (2004). How iris recognition works. *IEEE Trans. CSVT*, vol. 14, no. 1, pp. 21-30.

Ea T., Valentian A., Rossant F., Amiel F. and Amara A. (2005). Algorithm Implementation for Iris Identification. *MWSCAS 2005*, pp. 1207-1210.

Field D. (1987). Relations between the statistics of natural images and the response properties of cortical cells. *Journal of the Optical Society of America*, Vol. 4, Issue 12, pp. 2379-2394.

Huang J., Wang Y., Tan T., and Cui J. (2004). A New Iris Segmentation Method for Recognition. *Proceedings of the 17th International Conference on Pattern Recognition*, pp. 554-557.

Institute of Automation, Chinese Academy of Sciences. (2007). *CASIA Iris Image Database*. URL: http://www.cbsr.ia.ac.cn/IrisDatabase.htm. Accessed on 1st August 2007.

Intel Corporation. (2007). *Intel® Open Computer Vision Library*. URL: http://www.intel.com/technology/computing/opencv. Accessed on 10th December 2007.

Iritech Inc. (2007). *Embedded System*. URL: http://www.iritech.com/product_1-1.htm. Accessed on 7th November 2007.

Kong W. and Zhang D. (2001). Accurate iris segmentation based on novel reflection and eyelash detection model. *Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing*, pp. 263-266.

Li S.Z. and Lu J. (1999). Face Recognition Using the Nearest Feature Line Method. *IEEE Transactions on Neural Network*, 10(2), pp. 439-443.

Lim S., Lee K., Byeon O., and Kim T. (2001). Efficient Iris Recognition through Improvement of Feature Vector and Classifier. *ETRI Journal*, vol. 23, no.2, pp. 61-70.

Liu-Jimenez J., Sanchez-Reillo R. and Sanchez-Avila C. (2004). Biometric Co-Processor for an Authentication System using Iris Biometrics. *Proceedings of 38th Annual 2004 International Carnahan Conference on Security Technology,* pp. 131-135.

Liu-Jimenez J., Sanchez-Reillo R.and Sanchez-Avila C. (2005). Full Hardware Solution For Processing Iris Biometrics. *Proceedings of 39th*

Annual 2005 International Carnahan Conference on Security Technology, pp.157-163.

Ma L., Tan T., Wang Y., and Zhang D. (2004). Efficient Iris Recognition by Characterizing Key Local Variations. *IEEE Trans. Image Processing*, vol 13, no.6, pp. 739-750.

Ma L., Wang Y., and Tan T. (2002). Iris recognition using circular symmetric filters. *International Conference on Pattern Recognition*, vol 2, pp.414-417.

Ma L., Tan T., Wang Y., and Zhang D. (2003). Personal identification based on iris texture analysis. *IEEE Trans. On Pattern Analysis and Machine Intelligence*, vol 25, no.12, pp. 1519-1533.

Miyazawa K., Ito K., Aoki T., Kobayashi K. and Katsumata A. (2006). An Iris Recognition System using Phase-based Image Matching. *ICIP 2006,* pp.325-328.

Monro D. M., Rakshit S., and Zhang D. (2007). DCT-Based Iris Recognition. *IEEE Transactions on PAMI*, vol. 29, no. 4, pp. 586-595.

Poursaberi A. and Araabi B.N. (2005). A Novel Iris Recognition System Using Morphological Edge Detector and Wavelet Phase Features. *GVIP (05),* No. V6, pp. 9-15.

Ritter N. and Cooper J. (2003). Locating the iris: A first step to registration and identification. *Proceedings of the 9th IASTED International Conference on Signal and Image processing*, pp. 507-512.

Rydgren E., Ea T., Amiel F., Rossant F., and Amara A. (2004). Iris Feature Extraction Using Wavelet Packets. *IEEE International Conference on Image Processing*, vol. 2, pp. 861-864.

Sanchez-Avila C., Sanchez-Reillo R., and Martin-Roche D. De (2002). Iris-Based Biometric Recognition Using Dyadic Wavelet Transform. *IEEE AESS System Magazines*, vol. 17, no. 10, pp. 3-6.

Sung H., Lim J., Park J., and Lee Y. (2004). Iris Recognition Using Collarette Boundary Localization. *Proceedings of the 17th International Conference on Pattern Recognition*, vol. 4, pp. 857-860.

Teo C.C. and Ewe H.T. (2005). An Efficient One-Dimensional Fractal Analysis for Iris Recognition. *Proceedings of the 13th WSCG International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2005*, pp. 157-160.

Thornton J., Savvides M., and Vijaya Kumar B.V.K. (2005). Robust Iris Recognition using Advanced Correlation Techniques. *LNCS 3656,* pp.1098-1105.

Tisse C., Martin L., Torres L., and Robert M. (2002). Person identification technique using human iris recognition. *International Conference on Vision Interface, Calgary, Canada,* pp. 294-299.

Vijaya Kumar B.V.K., Xie C.Y., and Thornton J. (2003). Iris Verification Using Correlation Filters. *LNCS 2688,* pp. 697-705.

Wildes R., Asmuth J., Green G., Hsu S., Kolczynski R., Matey J., and McBride S. (1994). A System for Automated Iris Recognition. *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pp. 121-128.

Wildes R.P. (1997). Iris Recognition: An Emerging Biometric Technology. *Proceedings of the IEEE*, vol.85, pp.1348-1363.

Xu G.Z., Zhang Z.F. and Ma Y. (2006). Improving the Performance of Iris Recogniton System Using Eyelids and Eyelashes Detection and Iris Image Enhancement. *Proceedings of the 5th IEEE International Conference on Cognitive Informatics*, pp. 871-876.

Yan Y., and Zhang Y.J. (2008). 1D correlation filter based class-dependence feature analysis for face recognition. *Pattern Recognition Journal*, vol.41, pp. 3834-3841.

Yap W.H. (2007). Design of a digital signal processor based face verification system for access control application. Master Thesis, Universiti Teknologi Malaysia, Malaysia.

Yuan X. and Shi P. (2007). Efficient iris recognition system based on iris anatomical structure. *IEICE Electronic Express*, vol. 4, pp.555-560.

Zhu Y., Tan T., and Wang Y. (2000). Biometric Personal Identification Based on Iris Patterns. *Proceedings of the 15th International Conference on Pattern Recognition*, vol. 2, pp. 2801-2804.

APPENDIX A

List of Publications

1. Ng, R. Y. F., Tay, Y. H., and Mok, K. M. (2008). Iris Recognition Algorithms Based on Texture Analysis. *Proceedings International Symposium on Information Technology 2008 (ITSIM 2008).* 2, pp. 904-908.

2. Ng, R. Y. F., Tay, Y. H., and Mok, K. M. (2008). A Review of Iris Recognition Algorithms. *Proceedings International Symposium on Information Technology 2008 (ITSIM 2008).* 2, pp. 835-841.

3. Ng, R. Y. F., Tay, Y. H., and Mok, K. M. (2008). An Effective Segmentation Method for Iris Recognition System. *The 5th International Conference on Visual Information Engineering (VIE 2008)*, pp. 548-553.

4. Ng, R. Y. F., Tay, Y. H., and Mok, K. M. (2009). DSP-Based Implementation and Optimization of an Iris Verification Algorithm using Textural Feature. *Proceedings 6th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2009).*