

IRIS RECOGNITION AND VERIFICATION SYSTEM

Mehmet SARIOĞLU
Yusuf Arif ERDEM

Undergraduate Project Report
submitted in partial fulfillment of
the requirements for the
degree of Bachelor of Science (B.S.)

in

Electrical and Electronic Engineering Department
Eastern Mediterranean University

June 2005

Approval of the Electrical and Electronic Engineering Department

Prof. Dr. Derviş Z. Deniz
Chairman

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in
cope and quality, as an Undergraduate Project.

Assist. Prof. Dr. Hasan DEMİREL
Supervisor

Members of the examining committee

	Name	Signature
1.	Prof. Dr. Osman KÜKRER	
2.	Assoc. Prof. Dr. Erzat ERDİL	
3.	Assoc. Prof. Dr. Hüseyin ÖZKARAMANLI	
4.	Assist. Prof. Dr. Hasan DEMİREL	
5.	Assist. Prof. Dr. Mustafa UYGUROĞLU	

Date: 20/06/2005

ABSTRACT

IRIS RECOGNITION AND VERIFICATION SYSTEM

by

Mehmet SARIOĞLU, Yusuf Arif ERDEM

Electrical and Electronic Engineering Department
Eastern Mediterranean University

Supervisor: Asst. Prof. Dr. Hasan DEMİREL

Keywords: Iris Recognition, Verification, Principle Component Analyses, Image Processing, Visual Basic.net, Matlab, PicBasicPro, image acquisition, system control.

Nowadays, security systems take a very important place in public or private institutions which in security systems, visitor identification, especially for building access control, suspect identification by the police, driver's licenses and many other fields. In order to overcome the security problems biometric systems can be used. Iris recognition systems are the most accurate, reliable and efficient way to recognize and distinguish people.

This project is about Iris Recognition Systems and their application in daily life. The implementation of this system is based on image processing by using Principle Component Analyses (PCA). A graphical user interface is developed for establishing links between project components. SQL Server is used for database storage and Matlab is incorporated with the Visual Basic.net based graphical user interface to apply PCA. A key-pad based Personal Identification Number (PIN) entry hardware is developed for verification. In addition, an infrared camera is used for taking snapshots of the iris.

In this project, we will use Eigenfaces for iris recognition and verification. Furthermore, iris detection method is used to pinpoint the location of the iris and cancel out the undesired parts of the eye – picture. This process increases the speed of the algorithm and reliability of the recognition. Therefore the efficiency of the system will increase.

Acknowledgments

We would like to take this opportunity to thank our supervisor Assist.Prof.Dr. Hasan DEMİREL for encouraging us to make this project, in order to implement the knowledge we gained throughout our undergraduate studies, and benefit from these studies. Also we would like to thank Cem Cengiz, Ma'al Al-azzeH, Cem Direkoğlu and Mesut Yakup.

Finally, a special thank for all our friends in the department of Electric and Electronics Engineering for their assist while taking pictures for the training part of the project.

Table of Contents

ABSTRACT.....	2
ACKNOWLEDGMENTS	3
TABLE OF CONTENTS.....	4
LIST OF FIGURES	5
LIST OF TABLES	6
1. INTRODUCTION.....	7
2. TECHNOLOGICAL OVERVIEW	8
2.1 SCIENCE BEHIND THE TECHNOLOGY	8
2.2 WHAT IS IRIS?.....	8
3. BIOMETRIC COMPARISON	9
3.1 IRIS RECOGNITION VS. FACIAL RECOGNITION	9
3.1.1 Weaknesses of Facial Recognition.....	9
3.2 IRIS RECOGNITION VS. FINGERPRINT	9
3.2.1 Weakness of Fingerprint.....	10
3.3 IRIS RECOGNITION VS. HAND GEOMETRY.....	10
3.3.1 Weakness of Hand Geometry	11
3.4 STRENGTHS OF IRIS RECOGNITION.....	11
4. PRINCIPLE COMPONENT ANALYSIS (PCA).....	12
4.1 WORKING PRINCIPLE OF PCA.....	12
4.2 TRAINING PHASE OF PCA	13
4.3 TEST PHASE OF PCA	16
5. OPERATION OF SYSTEM.....	17
5.1 SYSTEM SOFTWARE	17
5.1.1 Main Program (Visual Basic.net).....	17
5.1.2 Matlab Functions Used in the Main Program.....	21
5.1.3 SQL Server.....	23
5.1.4 Pic Basic Pro and Proteus	23
5.2 SYSTEM HARDWARE.....	23
5.2.1 PIC, LCD and Key – pad	24
5.2.2 Camera	25
5.3 WORKING PROCEDURE	25
6. EXPERIMENTAL RESULTS AND DISCUSSIONS	28
6.1 METHODOLOGY	28
6.2 IRIS RECOGNITION PERFORMANCES	29
6.3 IRIS VERIFICATION	31
7. DIFFICULTIES IN APPLICATION	32
8. CONCLUSION.....	33
9. FUTURE WORK.....	34
REFERENCES.....	35
APPENDIX.....	36
VISUAL BASIC CODES.....	36
PIC BASIC PRO CODES	55

List of Figures

Figure – 1: Eye	8
Figure – 2: B/W to Matrix form.....	13
Figure – 3: Image matrix.....	13
Figure – 4: Difference Matrix	13
Figure – 5: First 20 eigenfaces with highest eigenvalues.....	15
Figure – 6: Main forms of Eye – Judge	17
Figure – 7: Program interface and enrolment.....	18
Figure – 8: Normal mode and verification form	19
Figure – 9: User delete form.....	20
Figure – 10: User details form.....	20
Figure – 11: Operating procedure for Matlab functions.....	21
Figure – 12: Iris detection stages	22
Figure – 13: PCA based iris recognition flow chart	22
Figure – 14: Database table relationship.....	23
Figure – 15: Operating procedure for hardware.....	24
Figure – 16: System hardware	24
Figure – 17: Circuit diagram of hardware.....	25
Figure – 18: Main operation procedure	25
Figure – 19: Data flow diagram of iris recognition system.....	26
Figure – 20: Eye samples from CASIA database [4].....	28
Figure – 21: Eye samples from our database.....	29
Figure – 22: Iris recognition performance graphic	30

List of Tables

Table – 1: Iris recognition performance test results.	29
Table – 2: Iris verification performance test results.	31

1. Introduction

A biometric system provides automatic recognition of an individual based on some unique characteristic owned by the individual. Biometric systems have been developed based on fingerprints, facial features, hand geometry, and the one presented in this report, the iris.

Biometric systems work based on capturing sample records, such as face image, eye image, hand shape, finger shape. These sample records are transformed using some mathematical function to build biometric training path. These training paths will provide some characteristic property for identification. Most of the biometric systems allow two operation modes, which are an enrollment mode to build database and identification mode to give access.

Nowadays, biometric systems play very important role with a maximum security is required where bank, military base, prison, airport, laboratory, intelligence office and etc. In order to produce a good biometric system, probability of having same characteristic should be minimal for individuals, so that the feature of individual does not change over time. Also the system should easily capture images in order to provide efficient access.

In this report we produced a biometric system which is based on the iris recognition. In order to complete verification process we have used Principal Component Analysis (PCA) to recognize a specified image by the help of an associated Personal Identification Number (PIN). Principal Component Analysis is one of the most successful techniques that have been used to recognize images. We have adapted PCA methodology to Iris Recognition. Also we introduced graphical user interface software by using Visual Basic.net and Adobe Photoshop CS.

This report includes fundamental information about Principal Component Analyses (PCA), biometric systems, combining software with hardware, difficulties in application and test results for Iris Recognition and Verification.

2. Technological Overview

2.1 Science behind the Technology

The iris has been historically recognized to possess characteristics unique to each individual. In the mid-1980s, two ophthalmologists--Drs. Leonard Flom and Aran Safir--proposed the concept that no two irises are alike. They researched and documented the potential of using the iris for identifying people and were awarded a patent in 1987. Soon after, the intricate and sophisticated algorithm that brought the concept to reality was developed by Dr. John Daugman [1] and patented in 1994. The original work and continued development have established Iridian's [2] iris recognition algorithm as the mathematically unrivaled means for authentication.

The technical performance capability of the iris recognition algorithm far surpasses that of any other biometric technology now available. Objective measures, such as a cross-over error rate, are at levels that cannot be reached by other biometrics. Iridian's algorithm is designed for rapid (seconds) exhaustive search of very large databases; a distinctive capability required for authentication today.

2.2 What is Iris?

The iris is the plainly visible, colored ring that surrounds the pupil. It is a muscular structure that controls the amount of light entering the eye, with intricate details that can be measured, such as striations, pits, and furrows. The iris is not to be confused with the retina, which lines the inside of the back of the eye. Fig.1 shows human eye characteristics.

No two irises are alike. There is no detailed correlation between the iris patterns of even identical twins, or the right and left

eye of an individual. The amount of information that can be measured in a single iris is much greater than fingerprints, and the accuracy is greater than DNA.

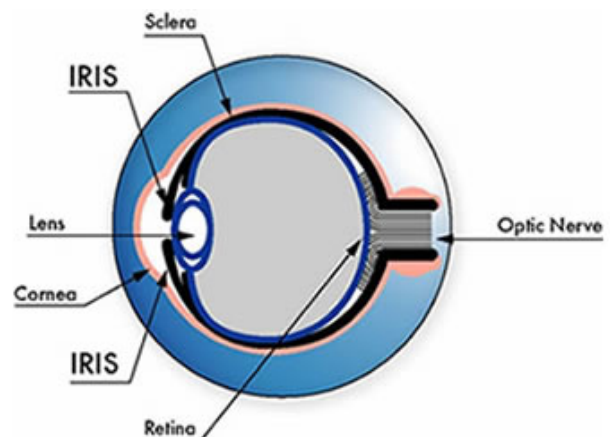


Figure – 1: Eye

3. Biometric Comparison

3.1 Iris Recognition vs. Facial Recognition

Facial recognition technology has gained publicity to scan large crowds and populations. However, facial recognition is relatively easy to fool. Age, facial hair, surgery, head coverings, and masks all affect results. For this reason, it will most likely remain a surveillance tool instead of a starting line identifier, and will not be used for critical match applications such as border control or restricted access.

3.1.1 Weaknesses of Facial Recognition

Although facial recognition has success to verification, lighting, age, glasses, hair and beard shape and face covering masks may change verification success rate. Lower success rate may occur for large populations. As a result, secondary possessing is required for surveillance operations. Also people do not know always their pictures is being taken and searched for database or picture may be taken without permission of the user.

3.2 Iris Recognition vs. Fingerprint

Fingerprint technology has been widely accepted. Fingerprint readers are not ideally suited to handle the large variation of populations that need to be enrolled. “Outliers,” or those in the population that deviate from the average, can be a natural limit to enrollment or recognition. In large-scale deployments it takes many minutes, not seconds, to conduct a single search and a search may require ancillary data (such as age, sex, etc.) to partition the database for more speed. Further, multiple candidate matches may be returned. In a high volume, high speed environment, fingerprints do not have the accuracy, reliability or ability to handle large, diverse populations as is needed for critical transportation applications such as border control or restricted access. For these reasons, the best use of fingerprint technology within the transportation industry may be background checks.

3.2.1 Weakness of Fingerprint

Fingerprint is not accurate as iris recognition. False accepts rate may occur and is approximately 1 in 100,000. Iris recognition false accepts rate is 1 in 1.2 million statistically [12]. Most of the fingerprint systems measure approximately 40 – 60 characteristics but iris recognition looks about 240 characteristic to create eigenfaces. Iris recognition can perform matching in a high speed but fingerprint search take much longer and it may require filtering may give wrong identity matches. Most of the biometric systems required physical contact with scanner device and that needs to be kept clean and this is not hygiene issue. Iris recognition has a standard but on the other hand because of large number of different symmetry means no fingerprint standard. Fingerprint readability also may be affected by the work an individual does. For example, transportation workers such as mechanics, food workers, or maintenance workers may present fingerprints that are difficult to read due to dryness or the presence of foreign substances, such as oil or dirt, on fingers. Trauma or some disease on finger may change or damage finger details meaning that user may be needed scan his or her finger many times.

3.3 Iris Recognition vs. Hand Geometry

Hand Geometry is one of the first biometric verification system including access control, sale application, employee working time logs etc. Actually it is easy to use, also expensive, but needs large equipments which may limit the application. Hand geometry carries other challenges as well. Weather, temperature and medical conditions affect hand size. Hand size and geometry change greatly over time, which is especially obvious different in the very young and very old. It always needs upgrades for hand geometry. These challenges make hand geometry unsuitable for the high volume, large population applications of the transportation industry.

3.3.1 Weakness of Hand Geometry

Hand geometry not matches with large databases. Whether and medical conditions such as pregnancy or certain medications can affect hand size. Hand size and geometry changes during the life cycle of people, especially in the very young and the very old. Most of the biometric systems required physical contact with scanner device and that needs to be kept clean and this is not hygiene issue. Big size of equipments may result difficulty in application. Expensive equipments are required.

3.4 Strengths of Iris Recognition

Iris recognition allow user to hands – free operation in application. Iris recognition has highest proven accuracy, had no false matches in over two million cross – comparison, according to Biometric Testing Final Report [12]. It allow high speed also for large populations, just look into a camera for a few seconds. The iris is stable for each individual through his or her life and do not change with age.

4. Principle Component Analysis (PCA)

Principle component analysis has been widely used for analyzing the image data. It is applied to biometric systems as a classification design. PCA enables us to measure the difference between two images while allowing expression changes. In the vector space, PCA identifies the major directions and corresponding strengths, of variation in the data. PCA performs these by computing the eigenvectors and eigenvalues of the covariance matrix of the image data. Keeping only a few eigenvectors corresponding to the largest eigenvalues, PCA can be also used as to reduce the size of the data while hold the major variation of data. For example, we may reduce the database size by using only first twenty eigenspace while our whole eigenvectors are 250.

Principle Component Analysis (PCA), also known as Eigen-XY analysis is a standard statistical technique for finding directions of maximum variations in data. These directions, called the principle components, can be used to reconstruct all of the information within the set and can be tested to which level a test image couples with an image of the training set. Principle components with smaller associated magnitudes can often be omitted, as they contribute less to the overall reconstruction of each data element. This allows sufficient representation of the original data set with a reduced set of principle components.

Principle components are found by computing the eigenvalues and eigenvectors of the covariance matrix associated with the data. Eigenvectors with largest associated eigenvalues are the principle components that describe the most variation in the data set. Principle component coefficients of data elements are found by projecting each datum onto the eigenspace of the covariance matrix.

4.1 Working Principle of PCA

In order to use PCA for image processing, firstly it is required to convert RGB (Red – Green – Blue) picture to gray scale (black and white). It is also possible to use RGB picture but instead of using 3D matrices, 2D matrices are better to evaluate. Given a gray scale image, that can be represented as a matrix of intensity values. We can convert it into a vector by placing one column of the image on top of another. Eye image is a two-dimensional array of size $x * y$, where x and y are width and height of the image. Each image can be represented as a vector of dimension $x * y$.

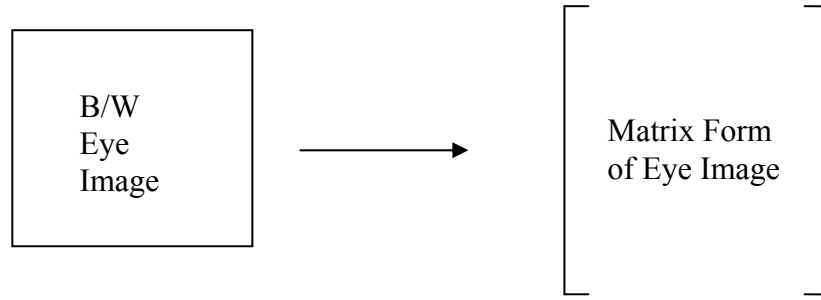


Figure – 2: B/W to Matrix form

4.2 Training Phase of PCA

Let Γ_n be an image from the collection of M images in database. P_1, P_2, P_3 to P_n in Fig.3 refers vector form of each picture.

$$\Gamma_n = \begin{bmatrix} P_1 & P_2 & P_3 & \dots & P_n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

Figure – 3: Image matrix

Average image Ψ is defined as:

$$\Psi = \frac{1}{M} \sum_{n=1}^M \Gamma_n \quad (4.2.1)$$

After getting average column vector matrices, we required to build difference matrix. Each image Γ_i differ from the average image Ψ by the vector:

$$\Phi_i = \Gamma_i - \Psi \quad (i=1,2,3,\dots,n) \quad (4.2.2)$$

$$\Phi = \begin{bmatrix} \Phi_1 & \Phi_2 & \dots & \Phi_n \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

Figure – 4: Difference Matrix

The covariance matrix of the data set is defined as:

$$C = \frac{1}{M} \sum_{n=1}^M \Phi_n \Phi_n^T = \Lambda \Lambda^T \quad (4.2.2)$$

Where Λ is

$$\Lambda = [\Phi_1 \Phi_2 \dots \Phi_M] \quad (4.2.3)$$

After getting covariance matrix, we can build the eigenvectors.

If we consider eigenvectors v_i of $\Lambda^T \Lambda$ such that

$$\Lambda^T \Lambda v_i = \mu_i v_i \quad (4.2.4)$$

When we multiply both sides of eq. (4.1.4) by Λ we obtain:

$$\Lambda \Lambda^T \Lambda v_i = \mu_i \Lambda v_i \quad (4.2.5)$$

Hence, we can construct $M \times M$ matrix:

$$L = \Lambda^T \Lambda \quad (4.2.6)$$

and find M eigenvectors v_i of L . If we now define eigenvectors of C as u_i , then equations (4.2.4) and (4.2.5) imply that:

$$u_i = \sum_{k=1}^M v_{ik} \Phi_k \quad (4.2.7)$$

or

$$u_i = \Lambda v_i \quad (4.2.8)$$

Eigenvectors u_i , are actually images, and they are called eigenfaces. Among those eigenfaces ones with the higher eigenvalues are more useful. Therefore those $M' < M$ eigenfaces that are most significant are used for constructing the “eye subspace” for image projections which are used in iris identifying, classifying or recognizing.

If we reshape the eigenfaces, they give us the ghost pictures. In Fig.5, we can observe first twenty ghost pictures for eigenfaces.

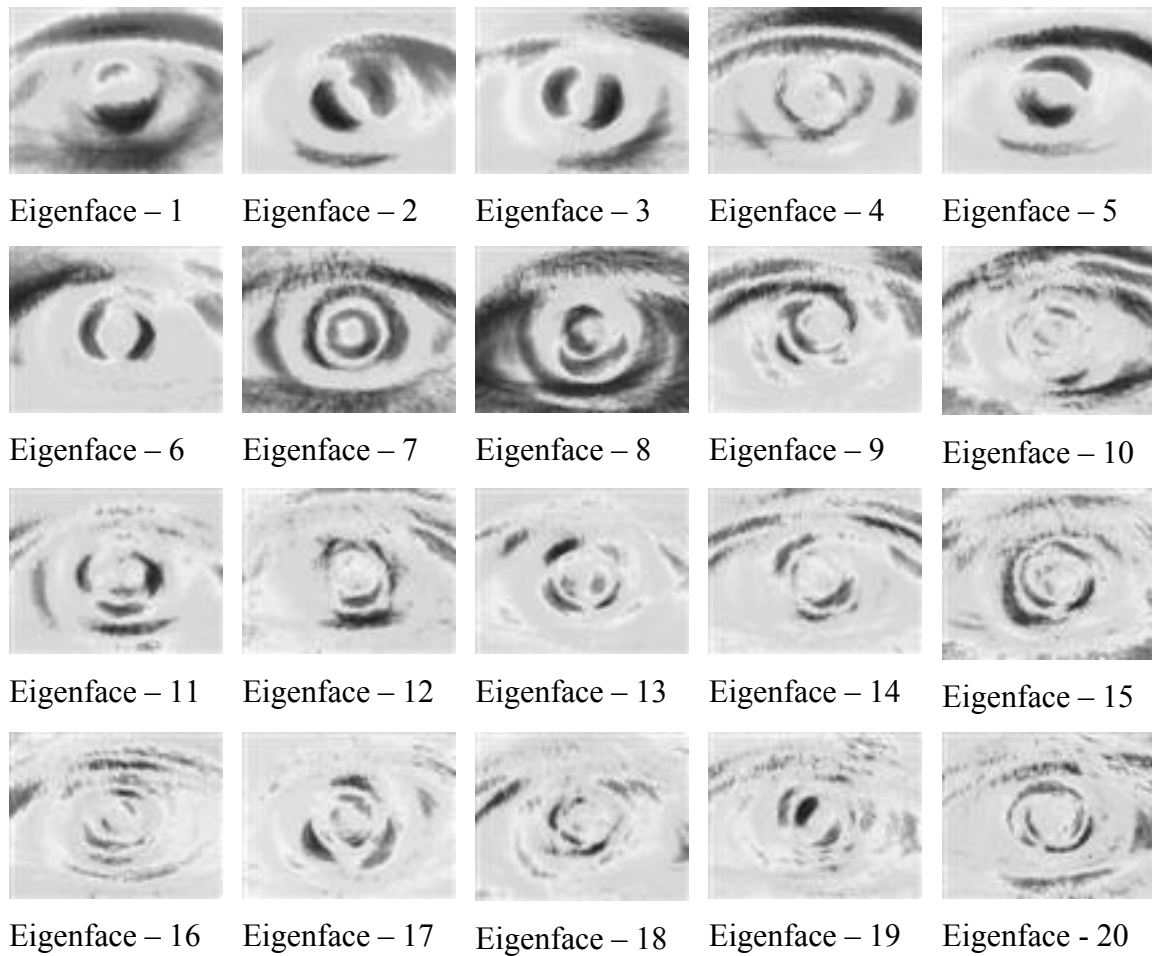


Figure - 5: First 20 eigenfaces with highest eigenvalues.

4.3 Test Phase of PCA

After finishing training phase and getting eigenfaces, let us move on the test phase to proof our PCA theory.

Although we have M pictures and eigenfaces in our database, it is not required to use all of these eigenfaces. We may use M' of them ($M' \ll M$).

In order to test any picture to whether it is a sample of the training part or not, we should to transform that picture to eigenface form by the following operation:

$$\omega_k = u_k^T (\Gamma - \Psi) \quad k = 1, 2, \dots, M' \quad (4.3.1)$$

ω_k is the k -th coordinate of the Φ in the new “eye space”. Eq. (4.3.1) describes point-by-point image multiplications and summations, resulting in the scalar value (dimension 1×1), previously defined as a weight that describes each face. Those weights form a vector

$$\Omega^T = [\omega_1 \ \omega_2 \ \dots \ \omega_{M'}] \quad (4.3.2)$$

gives us the projection vector.

Vector Ω describes contribution of each eigenface in representing the input (test) image, where the eigenfaces represent basis set for eye images.

This vector is used in finding the class that input image belongs to if there are more than one images describing a person, otherwise it is used to find to which single image is most similar. This will be done by building Euclidian distance between test image and projection image classes. If we compare the distance between by using ω values for training and test phase, it will give us the distance of the test picture according to the training phase.

$$\epsilon_k = \|\Omega - \Omega_k\|_2 \quad (4.3.3)$$

Here Ω_k describes k -th eye class, which is average of the eigenface representation of eye images of each individual. An eye will be classified to some class if the minimum ϵ_k is below some specified threshold, otherwise it will be classified as “unknown” picture.

5. Operation of System

In order to construct this iris recognition system (Eye – Judge), it is required to use some programming languages and some peripheral devices. We have used VisualBasic.net, SQL, Matlab and PicBasicPro programming languages to constitute software part of the project and PIC, key-pad, LCD and camera for the hardware part of the project. We have also used some programs which are Proteus and Adobe Photoshop. In order to combine system software and hardware, RS232 serial communication protocol is needed.

5.1 System Software

The main part of our project has been constituted by the software. The software controls database and hardware to fulfill main operating procedures. Fig.6 shows main forms of the Eye – Judge Software.

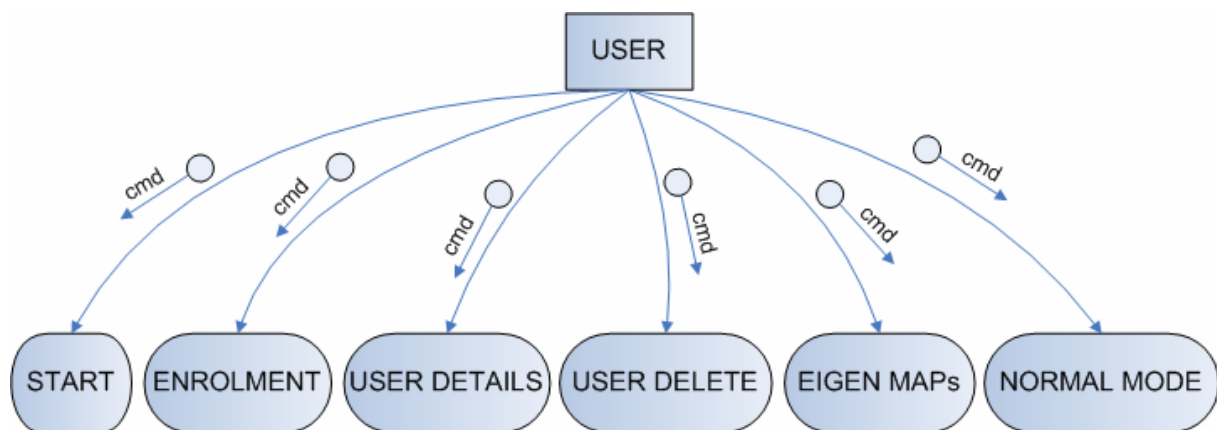


Figure – 6: Main forms of Eye – Judge

5.1.1 Main Program (Visual Basic.net)

Visual Basic.net gives an opportunity to programmer to introduce visual interfaced program and it uses SQL database efficiently that is why we have used Visual Basic.net programming language. We have designed the program interface by using Adobe Photoshop 8 CS as shown in Fig.7.

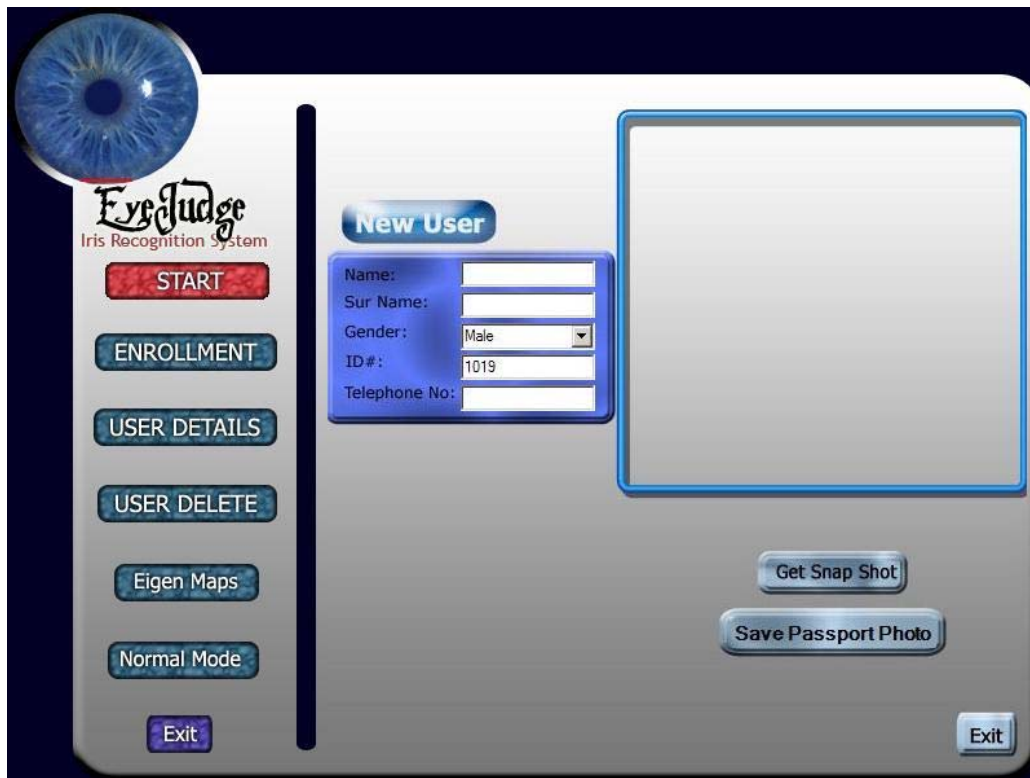


Figure – 7: Program interface and enrolment

The main program has been written by using Visual Basic.net programming language. We have many forms in our program. Each form has been designed for different assignment. The program also has three different skins to serve user requirement which one of them (Normal Mode) is shown in Fig.8. Firstly, the eyes' images have to be taken with the camera to build database. This procedure can be made easily by using "Enrolment" form.

Administrator has to fill required information and get passport photo to identify user. After taking passport photo, the program will alert administrator to get ten eye pictures. After getting snapshot of eye images these will be add to SQL database. Program gives ID number automatically. Administrator may enter many users without build eigenmaps after each user. Building eigenmaps will be automatically done after pressing exit button for enrolment form.

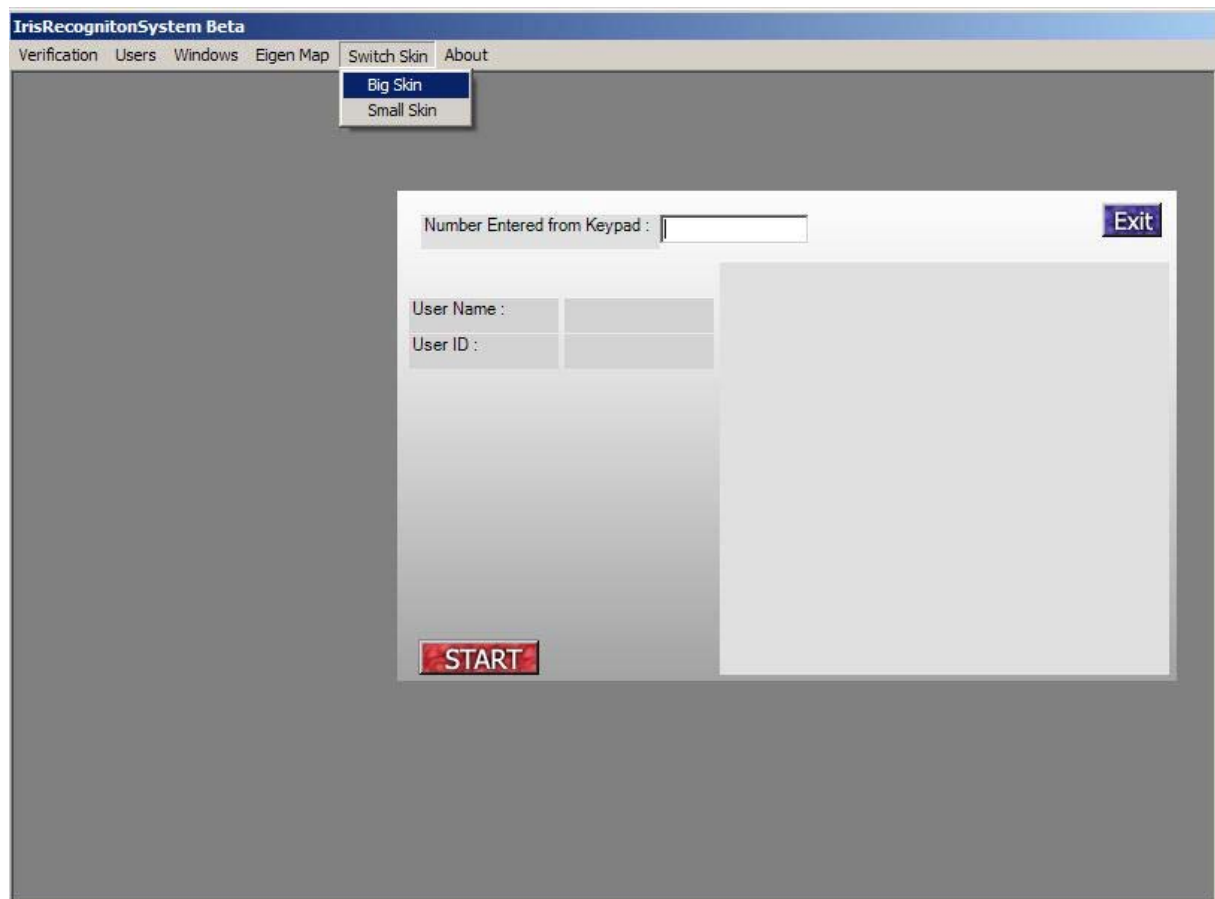


Figure – 8: Normal mode and verification form

After creating database and building eigenmaps we can make verification. It is required to run verification form for recognition. When administrator press “START” button, shown in Fig.8, the program will automatically initialize hardware by sending security codes for emergency cases and to terminate main program by RS232 interface. It will also send predefined time to predicate how long door will be open after giving access to the user. Later on, if any user come and enter his or her ID number, main program gets that specific ID from the hardware and checks whether the user is registered or not. If user is reregistered, program gets snapshot of the eye through the camera and gives access to user if user eye image meet with his or her ID. In addition, there is a simulation part for testing without using the hardware except camera.

It is also possible to remove any registered user by using “Delete User” form. After deleting the user, the program will build eigenmaps again. As you see in Fig.9, the user may be removed by entering UserID.

The form is titled "Enter user ID to delete". It contains a text input field with the value "1004", a "Delete" button, and an "Exit" button. To the right, there is a "NewDataSet" window displaying a list of users with their IDs and names.

ID	Name
1004	Emre
1005	Yusuf
1006	Mehmet
1007	Burak
1008	Alper
1009	Rıdvan
1012	İbrahim
1014	Ercan
1015	Recep
1016	Kemal
1017	Alper
1018	Burak
1019	Neyre
1020	Hakan
1021	Yakup
1022	Selman

Figure – 9: User delete form

Administrator may reach user details by pressing “User Details” button. User details form shown in Fig.10, also gives an option to look out user logs and it is also possible to know which pictures are belonging to which user.

The form is titled "Users". It contains a "NewDataSet" window displaying a table of user details. The table has columns for ID, Name, Surname, and a numeric value. There are also two links: "Photo to User" and "Log to User".

ID	Name	Surname	Value
1004	Emre	Göküş	1
1005	Yusuf	ERDEM	2
1006	Mehmet	SARIOĞLU	3
1007	Burak	KÖLE	4
1008	Alper	AYDIN	5
1009	Rıdvan	EKER	6
1012	İbrahim	ÖLÇAL	7
1014	Ercan	TUNALI	8
1015	Recep	YILMAZ	9
1016	Kemal	YURT	10
1017	Alper	AVDAN	11
1018	Burak	DOĞAN	12
1019	Neyre	TEKBIYIK	13
1020	Hakan	TOZKOPARA	14
1021	Yakup	FAKIOĞLU	15
1022	Selman	ÖZCAN	16

Figure – 10: User details form

It is required to set some values by using “Properties” form which can only be seen in normal mode. Firstly, administrator has to fix threshold value according to quality of images; it is directly related with camera, and also size of the eigenmaps. If we get sharp and high quality snapshots we may reduce the size of eigenmaps, so the program is getting faster and we can save memory and hard disk space. Also administrator has to define COM port for hardware and also has to define how long door will be open after giving access to the user.

5.1.2 Matlab Functions Used in the Main Program

Matlab realizes PCA procedures for iris recognition and verification. It is also an interface between camera and main program. Matlab takes snapshot of the iris image and transfers it to the main program. We have used Matlab M – Files by creating dll (eyejudgedll.dll) file for using in main program. Combination of dll with other component has been shown in Fig.11. In order to increase success of the verification we need to detect iris part of the eye image. Also iris detection has been made by Matlab. Fig.12 shows working procedure of the PCA.

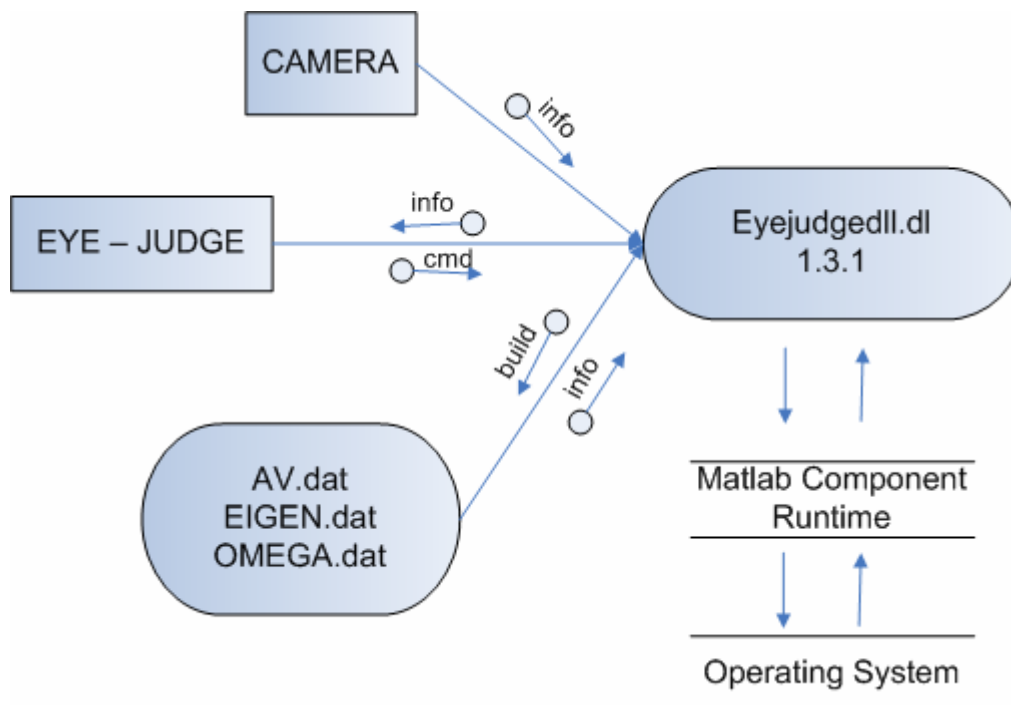


Figure – 11: Operating procedure for Matlab functions

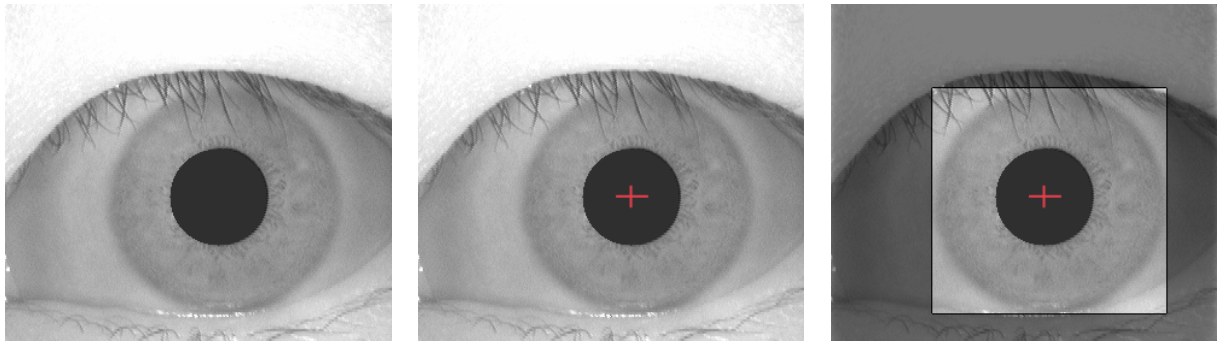


Figure – 12: Iris detection stages

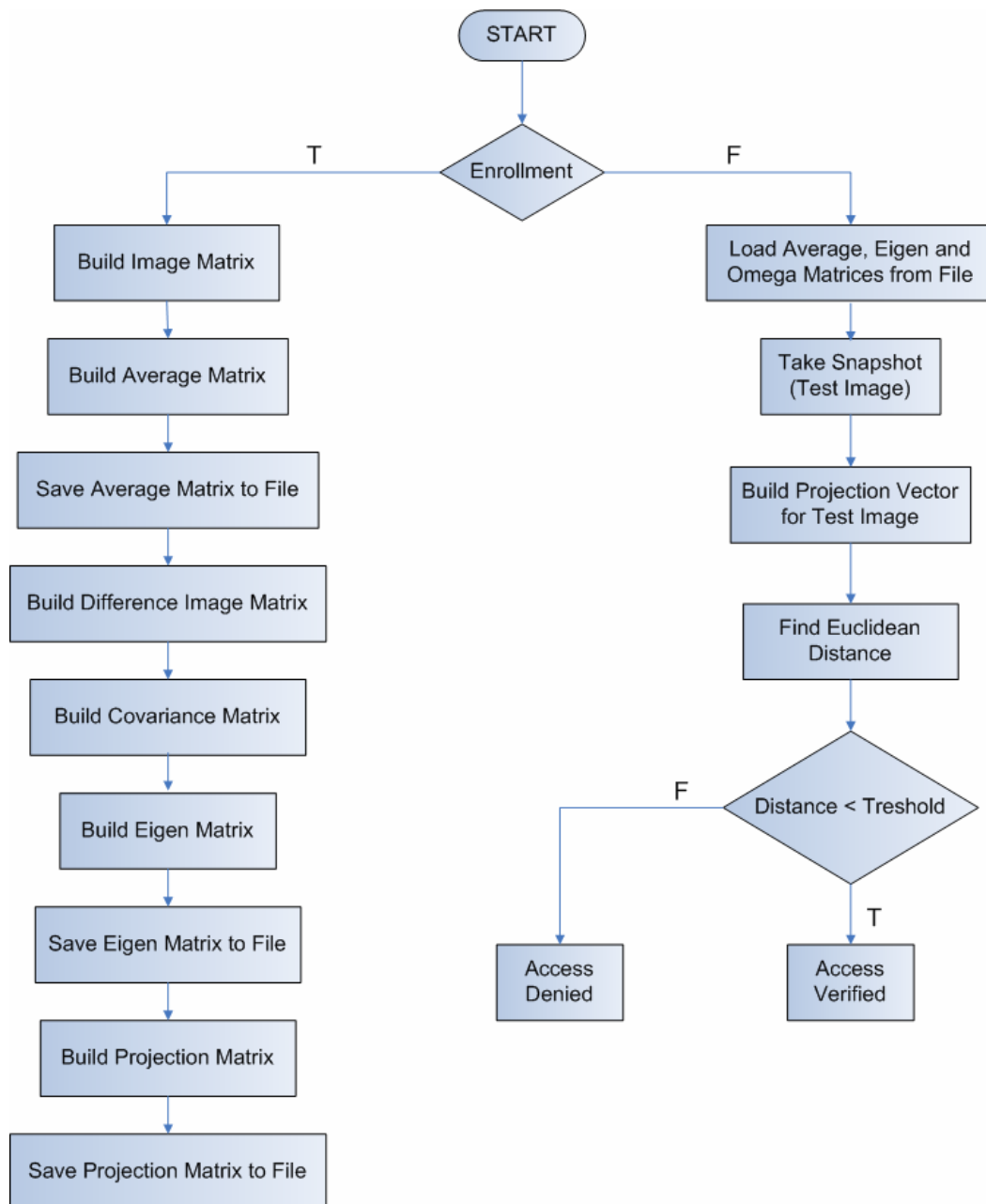


Figure – 13: PCA based iris recognition flow chart

5.1.3 SQL Server

SQL Server has been used to make original eye pictures secure and it is one of the fastest database program. We need to use SQL server to store original images, user detail and logs. Primary key relation of database with Eye – Judge has been shown in Fig.13.

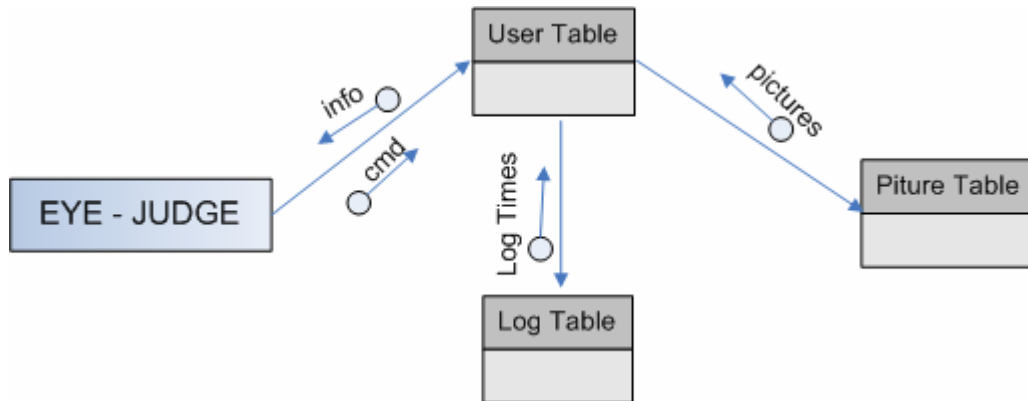


Figure – 14: Database table relationship

5.1.4 Pic Basic Pro and Proteus

Pic Basic Pro is the programming language for programming PIC microchips. We have used a 16F877 PIC to scan key-pad and drive LCD. It is difficult to test written program for PIC. Programmer must program PIC each time after making any small change. In order to come over this difficulty we have used Proteus circuit designer and simulation program. It enables user to construct and test the circuit with very large component library.

5.2 System Hardware

In our project, we have some peripheral devices which are PIC, LCD, key – pad and camera. If you look at Fig.14, it clearly shows how main program (Eye – Judge) controls hardware of the system.

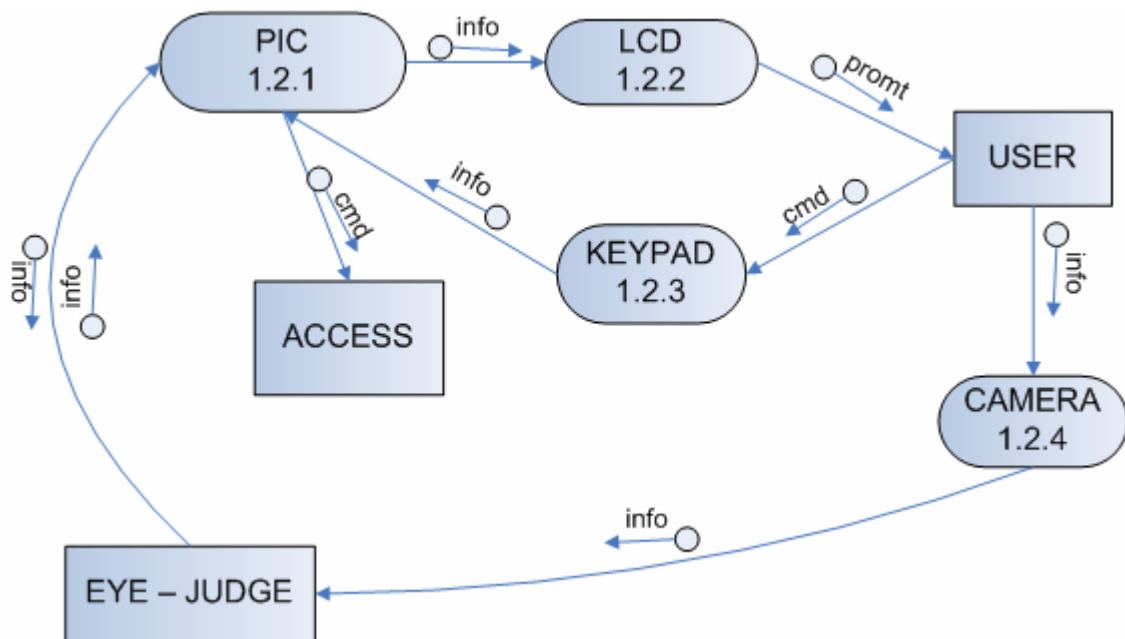


Figure – 15: Operating procedure for hardware

5.2.1 PIC, LCD and Key – pad

PIC is one of the most important parts of the hardware. PIC scans key-pad to get user ID to send it to main program through RS232 communication port. A LCD is included to the hardware to give guide information to the user and it is driven by the PIC. We used 4x20 dot matrix LCD. The key-pad is standard matrix form key-pad. It is working based on matrix scanning procedure.

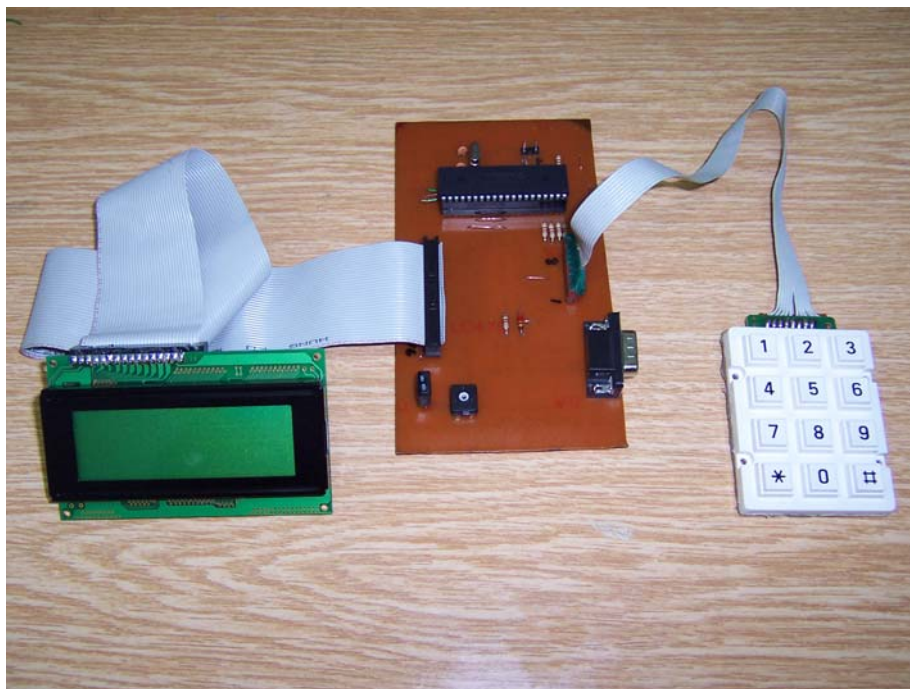


Figure – 16: System hardware

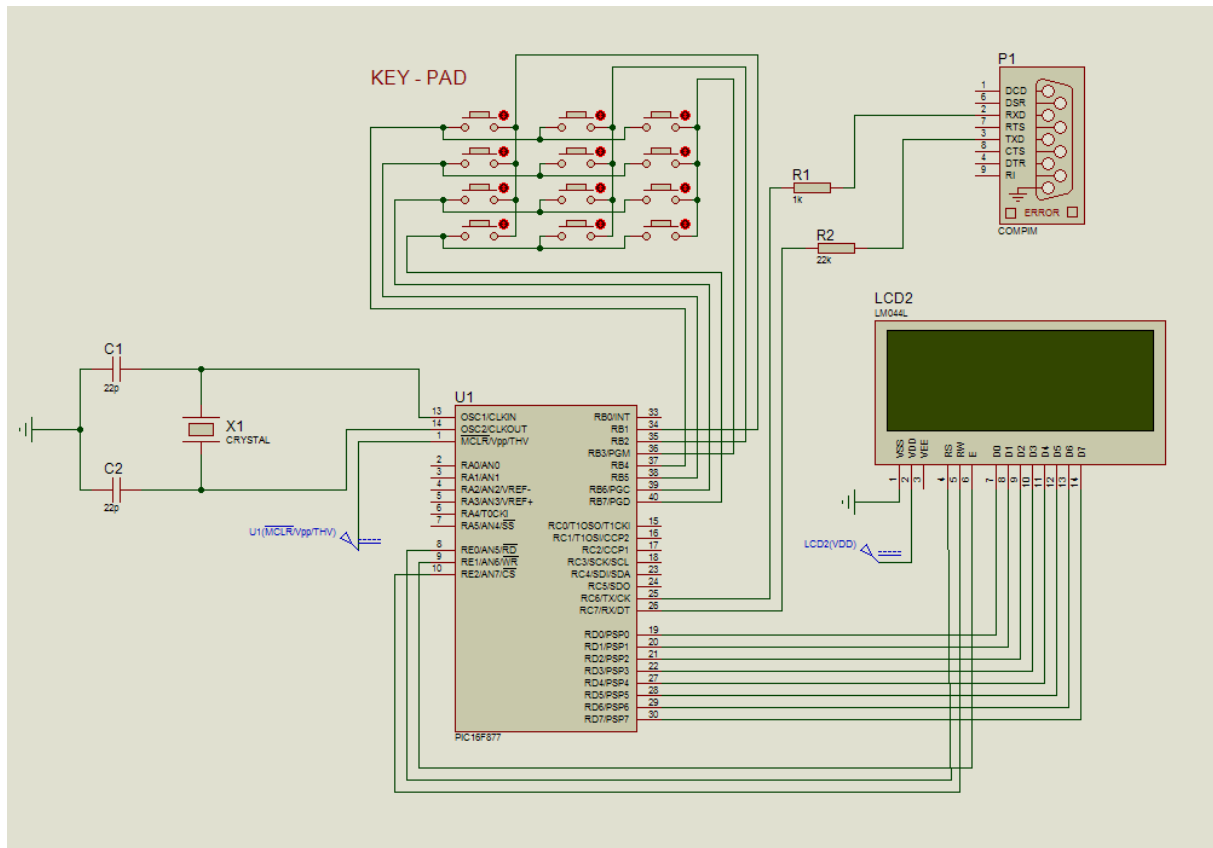


Figure – 17: Circuit diagram of hardware

5.2.2 Camera

We have used a hand camera in order to get snapshot of eye images. It is not a simple camera. We have used infrared light to predicate iris details. We have taken eye images in dark place by putting an eye hat in front of the camera because of eye surface reflects daylight and also other light. We also used fire wire cable while transferring pictures from camera to program because fire wire communication is faster than other communication types and high speed allow us to take high quality image with 720x576 resolution.

5.3 Working Procedure

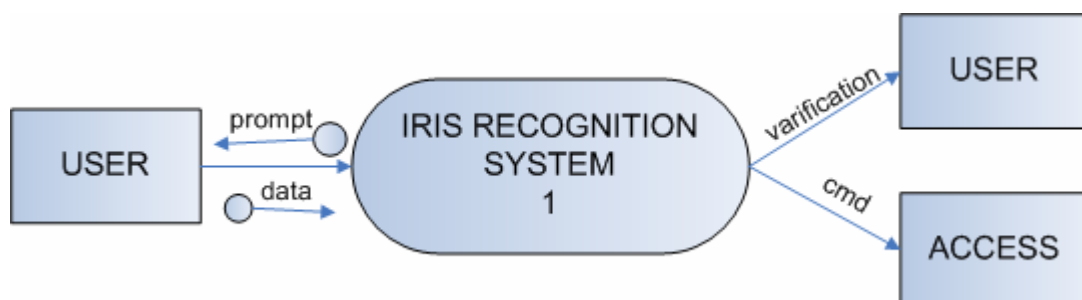


Figure – 18: Main operation procedure

Fig.15 shows main operation procedure of the iris recognition system and Fig.16 shows details of Iris Recognition System. Firstly, the eyes' images have to be taken with the camera to build database. Camera directly connected to computer through fire wire protocol. When the system is ready after building database, the main program sends security, program terminate code and delay time to describe how long the door remains open. Security code is required to open door manually if any problem occur on PC or server. Terminate code is required to send key command to main program if we need to stop the system because programmer may change security codes or add/remove any other user.

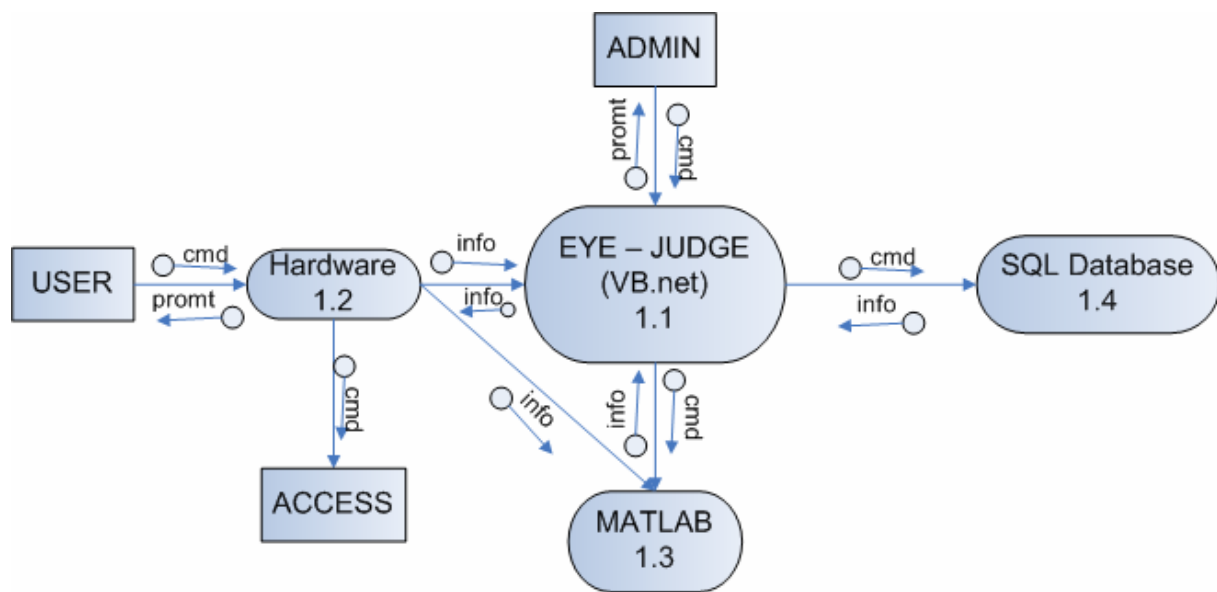


Figure – 19: Data flow diagram of iris recognition system

After initializing hardware, the system is ready to make verification. The procedure starts with entering ID number by the user using key-pad. ID number automatically is sent to main program by PIC through RS232 serial communication protocol. Until this moment, main program always checks serial port whether new ID number is entered or not. The main program checks ID number from the database whether it is a registered user or not. If the user do not registered before to get access, program sends an error code to PIC and PIC prompts to user “You are not a registered user” by using LCD. If the user has a registered ID, program prompts user to make a closer look to the camera and press a snapshot button when he or she is ready to give a eye photo. After pressing button to take snapshot, PIC sends a command to main program to take snapshot. Eye photo is converted to iris form by making iris detection.

After this point main program realizes PCA procedure to compare taken picture to verify. If PCA procedure verifies the snapshot iris picture, it sends a command to PIC to give access. Getting access command from the main program starts new procedure for PIC. It sends a control signal to the door and door be remain open until defined time limit reach. If access not granted by the main program it sends a command to PIC and PIC will prompt user to “Access Denied” through the LCD.

This operation procedure continues until any security code entered. Administrator user may control the door by entering security code. If security code is entered door opens and remain open until “*” button pressed from the key-pad.

6. Experimental Results and Discussions

We have collected some experimental and theoretical test results to measure success of system.

6.1 Methodology

In order to get these test results we have used two different databases. In early development period of the project we have used CASIA [4] database. Two samples eye image of CASIA database is shown in Fig.17 and two sample eye image of our own database is shown in Fig.18. The quality of the eye images taken from CASIA has been very high. We used three training images and one test image. Although we did not make iris detection we get 72% of success rate. After creating our own database we had fifteen different person and we used nine training pictures and one test picture for each person. One of these people was using lens and two of them have blue colored eye. The recognition success without entering PIN number is 100% by using first seventeen Eigen faces. We get this success rate by using Principle Component Analysis (PCA) method. Integration of the hardware and taking PIN number from user success rate is also 100%.

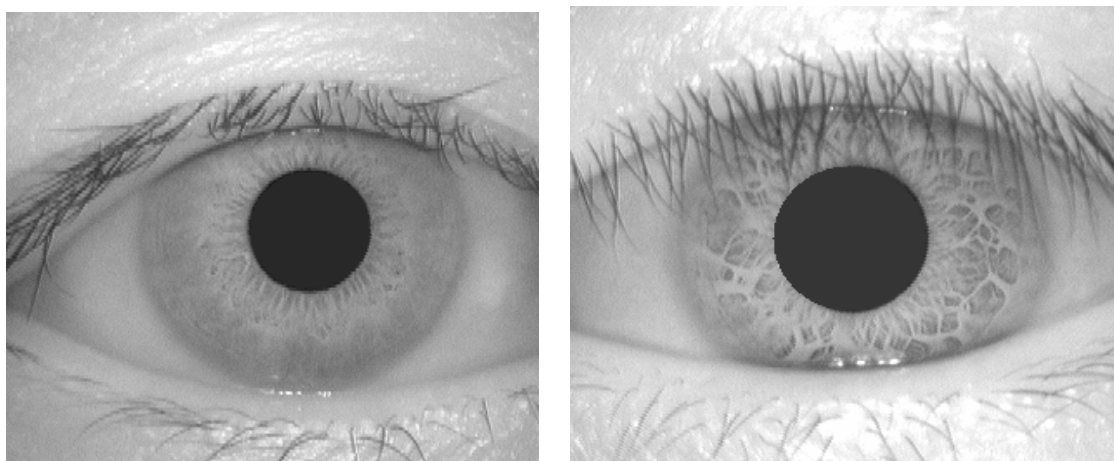


Figure – 20: Eye samples from CASIA database [4]

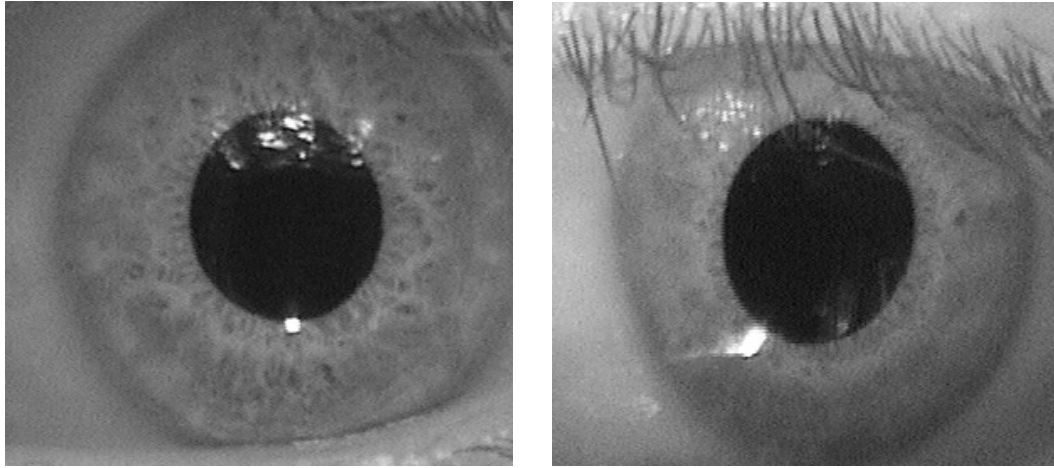


Figure – 21: Eye samples from our database

6.2 Iris Recognition Performances

Test result shows that iris recognition is the most reliable and secure biometric verification system. Table – 1 shows the verification success for first fifty eigenface without using keypad. The table clearly shows that using first fifteen eigenface is enough to verification. Despite first fifteen eigenface is enough to make verification we are using first twenty eigenface to guarantee our verification success 100%. Eigenfaces are directly related with the pictures on the training phase but we are using just first twenty of them. Although building eigenfaces takes little time (related with size of the database) verification may guarantee in 3 seconds. Building eigenfaces is required only if any user removed or registered to system and this process do not effect the verification run time.

# of Used Eigenface	Success Rate	# of Used Eigenface	Success Rate	# of Used Eigenface	Success Rate	# of Used Eigenface	Success Rate	# of Used Eigenface	Success Rate
1	7,11	11	93,75	21	100	31	100	41	100
2	64	12	100	22	100	32	100	42	100
3	56,85	13	100	23	100	33	100	43	100
4	85,33	14	93,75	24	100	34	100	44	100
5	71,07	15	100	25	100	35	100	45	100
6	78,17	16	100	26	100	36	100	46	100
7	78,17	17	100	27	100	37	100	47	100
8	92,39	18	100	28	100	38	100	48	100
9	93,75	19	100	29	100	39	100	49	100
10	93,75	20	100	30	100	40	100	50	100

Table – 1: Iris recognition performance test results.

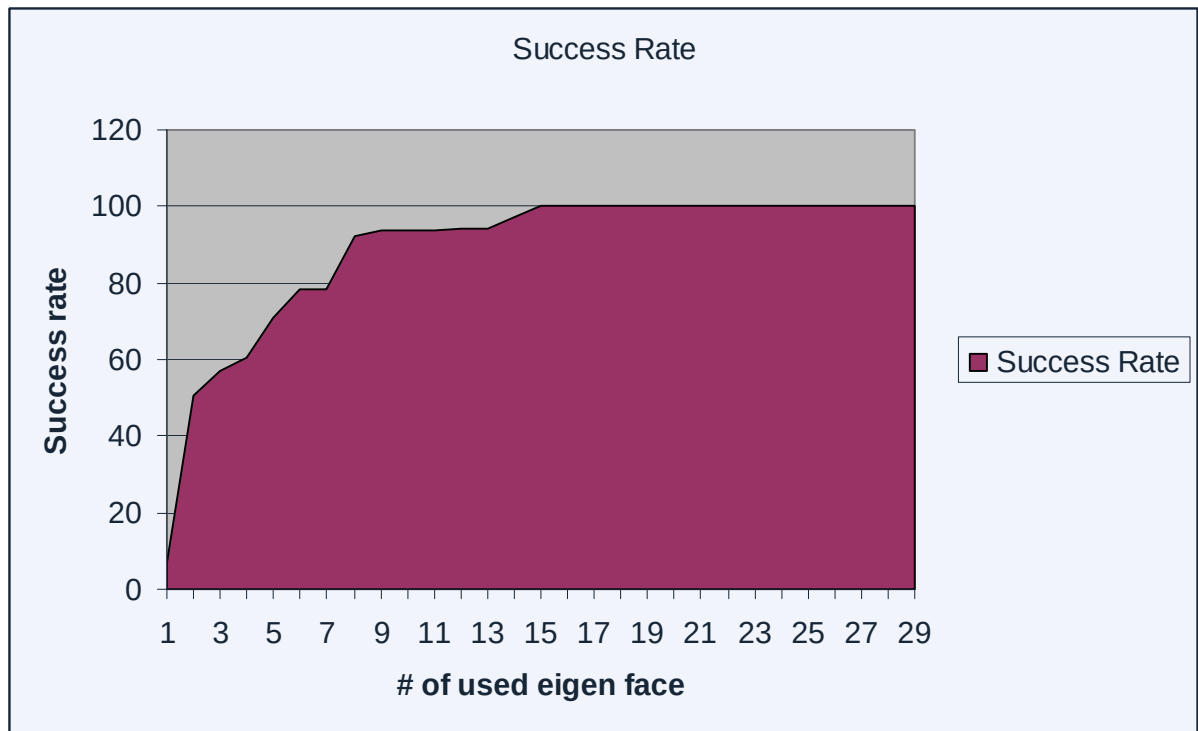


Figure – 22: Iris recognition performance graphic

6.3 Iris Verification

Verification test has confirmed our recognition results. We made our verification test for only 16th user. Test results in Table – 2 clearly shows Euclidean distance between 16th user and other users getting regular and uniform while used Eigenfaces are increasing. This regularity increases verification success rate. If we increase number of Eigen faces it did not make any difference to make verification more reliable that is why we are using first twenty Eigen faces.

	1. Eigen	2. Eigen	3. Eigen	4. Eigen	5. Eigen	6. Eigen	7. Eigen	8. Eigen	9. Eigen	10. Eigen
1	0.3162	0.4628	0.5696	0.5924	0.6884	0.7329	0.7876	0.7914	0.7955	0.8063
2	0.0061	0.1696	0.2880	0.2887	0.2892	0.4396	0.5522	0.5725	0.5826	0.5836
3	0.1438	0.3551	0.4091	0.4204	0.4248	0.4331	0.4516	0.4529	0.4534	0.4607
4	0.0103	0.1314	0.2458	0.2486	0.3330	0.5168	0.5170	0.5194	0.5427	0.5436
5	0.1014	0.1041	0.4570	0.4649	0.4697	0.5463	0.5531	0.6041	0.6423	0.6492
6	0.1136	0.1450	0.2117	0.2741	0.2972	0.4945	0.5060	0.5123	0.5134	0.5218
7	0.0719	0.2924	0.3898	0.4150	0.4266	0.6060	0.6206	0.6322	0.6324	0.6357
8	0.0144	0.0251	0.2932	0.3554	0.3559	0.4422	0.5512	0.6195	0.6312	0.6349
9	0.3639	0.3726	0.7724	0.8591	0.8755	0.9316	0.9317	0.9317	0.9342	0.9378
10	0.0118	0.0282	0.4166	0.4188	0.4235	0.5371	0.6322	0.7147	0.7175	0.7224
11	0.2259	0.2701	0.4030	0.4362	0.4366	0.4367	0.4555	0.4599	0.4671	0.4800
12	0.4130	0.4259	0.4954	0.5408	0.5426	0.6163	0.6182	0.6480	0.6494	0.6786
13	0.0094	0.1990	0.2639	0.3872	0.3933	0.4071	0.4113	0.4335	0.4627	0.4628
14	0.6086	0.6342	0.8973	0.8998	0.9154	0.9633	0.9640	1.0075	1.0084	1.0300
15	0.1397	0.5622	0.6075	0.6086	0.6143	0.6430	0.6448	0.7004	0.7005	0.7019
16	0.0017	0.0421	0.0528	0.0540	0.0652	0.0729	0.1508	0.1560	0.1854	0.1884
	11. Eigen	12. Eigen	13. Eigen	14. Eigen	15. Eigen	16. Eigen	17. Eigen	18. Eigen	19. Eigen	20. Eigen
1	0.8066	0.8079	0.8298	0.8305	0.8305	0.8445	0.8446	0.8448	0.8481	0.8486
2	0.5836	0.5863	0.6591	0.6675	0.6757	0.6812	0.6824	0.6951	0.6954	0.6961
3	0.4608	0.4612	0.4627	0.4821	0.4956	0.5544	0.5560	0.5564	0.5570	0.5612
4	0.5436	0.5443	0.5660	0.5681	0.5688	0.5838	0.5901	0.5934	0.5948	0.6042
5	0.6495	0.6498	0.6857	0.6981	0.7064	0.7065	0.7076	0.7205	0.7220	0.7277
6	0.5227	0.5280	0.5389	0.5564	0.5627	0.5642	0.5659	0.5734	0.5824	0.5825
7	0.6366	0.6366	0.6585	0.6588	0.6678	0.6891	0.6979	0.6985	0.7032	0.7045
8	0.6362	0.6377	0.6596	0.6599	0.6606	0.6628	0.6666	0.6735	0.6741	0.6745
9	0.9378	0.9473	0.9650	0.9656	0.9658	0.9770	0.9770	0.9845	0.9849	0.9850
10	0.7494	0.7629	0.7764	0.7771	0.7819	0.7939	0.8018	0.8081	0.8091	0.8091
11	0.4800	0.4804	0.5381	0.5393	0.5411	0.5438	0.5577	0.5583	0.5586	0.5586
12	0.7014	0.7037	0.7269	0.7271	0.7390	0.7598	0.7626	0.7704	0.7704	0.7709
13	0.4695	0.4710	0.4712	0.4936	0.4964	0.5017	0.5144	0.5218	0.5218	0.5227
14	1.0301	1.0308	1.0597	1.0603	1.0603	1.0718	1.0726	1.0726	1.0726	1.0727
15	0.7181	0.7209	0.7538	0.7601	0.7604	0.7745	0.7838	0.7883	0.7883	0.7889
16	0.1886	0.1898	0.2025	0.2109	0.2109	0.2114	0.2119	0.2211	0.2240	0.2241

Table – 2: Iris verification performance test results.

Note: The values on the table are multiplied with 10-4

7. Difficulties in Application

In order to bring project to this point we have faced many problems. These problems occurred mostly for hardware part. We have faced also software problems but they were easy to come over.

Firstly, we tested our PCA verification procedure by using CASIA eye database pictures. We used only three pictures for training set and one picture for test set. Those pictures have very high quality and all of them have taken with same light. According to those pictures our PCA procedure was working properly and we get 72% success. For biometric systems, this success rate is not enough to construct secure system. In order to increase success rate, we have made iris detection to remove undesired part of the pictures (eye lid, eyelash etc.). After getting only iris details from the CASIA eye pictures by making iris detection, we have increased our success rate to 100%. This success rate made us very happy until moved on practice to our program by using our own hardware and camera. When we started to get eye pictures, we have seen that iris details can not easily appreciable because of reflection of day light or other light sources. As we all know eye surface reflects lights and it is shining under lightly places. In order to come over this problem we have taken pictures in dark places by using infrared light. We have introduced a small box to put it in front of the camera to create dark ambiance. Using infrared light has solved the shining problem on eye surface and it has made iris details more comprehensible.

Secondly, we have faced a problem while using serial port. We used a notebook computer and most of the latest models not have a RS232 port. In order to come over this problem, we used a USB-to-RS232 converter. Also we used fire wire connection for camera that is why we used a notebook computer. At the early application of our system we had used USB cable to camera connection but we have seen that resolution was not enough by using USB cable that is why we need to use fire wire cable. Fire wire also transfer snapshot photo very fast and its resolution is very high than the USB cable. High resolution occurs because of high transfer speed.

8. Conclusion

In conclusion, this report is giving detailed information about Principle Component Analyses (PCA) and how to implement it by explaining on figures. Also it gives test result for both iris recognition and verification.

This thesis has presented an iris recognition system, which was tested using two databases of grayscale eye images in order to verify the claimed performance of iris recognition technology, which are CASIA database and our own database.

Firstly, an automatic segmentation algorithm was presented, which would localize the iris region from unwanted parts of the eye (eyelash, eyelid). Automatic segmentation was achieved some simple codes. Firstly, we found the center of the pupil, after we have cut eye image according to center of the pupil. Therefore we have got clearer picture.

Next, we applied PCA on the segmented iris image. So PCA gives the most accurate results. Before iris detection PCA gives 72% success rate for 3 training image of 145 eye images. But after iris detection, success rate has increased to 100%.

We also developed very useful and easy to use graphical user interfaced program by integrating to them hardware. This system can be used for security purposes where higher security control is required.

As a result, PCA is applicable for iris recognition system.

9. Future Work

Future development of the software is quite possible, and in fact is very helpful in order to increase its efficiency, and flexibility in use.

Matlab is powerful software when it comes to mathematical operations but it uses lots of vectors and matrix and these matrices and vectors uses too much memory, hard disc and slow down the processor unit of the computer. Thus, implementing our project by using C#.net or Visual Basic.net to save memory and time. Also program will work independent from the Matlab.

Since we used a stand alone PC to control any gate, if we construct a Local Area Network (LAN) by keeping irises, user and log information on main server, ever PC may reach these files and may given access any gate on the LAN.

Also it may add specific time groups to indicate which user will be given access at which time user get access.

Wireless communication may be added instead of RS232 communication or we may change RS232 protocol with USB protocol to make interface between PIC and computer.

We may get snap shots of eye image by measuring the distance of the user's eye by using sensor.

References

- [1] John Daughman. <http://www.cl.cam.ac.uk/users/jgd1000/>
- [2] Iridian Technologies. <http://www.iridiantech.com>
- [3] Lindsay I. Smith February 26, 2002
- [4] CASIA Iris Image Database. From <http://www.sinobiometrics.com/casia%20iris.htm>
- [5] Planet Math. <http://planetmath.org/encyclopedia/PrincipleComponentsAnalysis.html>
- [6] Jaypeetex. <http://www.jaypeetex.com/>
- [7] Connexions. <http://cnx.rice.edu>
- [8] Chinese Academy of Sciences – Institute of Automation. Database of 756 Greyscale Eye Images. <http://www.sinobiometrics.com> Version 1.0, 2003.
- [9] P. Kovesi. MATLAB Functions for Computer Vision and Image Analysis. Available at: <http://www.cs.uwa.edu.au/~pk/Research/MatlabFns/index.html>
- [10] Mathworks. <http://www.mathworks.com>
- [11] LG Electronics / Iris Technology Division. <http://www.lgiris.com>
- [12] Biometric Product Testing Final Report (19 March 2001, Center for Mathematics and Scientific Computing, National Physics Laboratory, UK).
<ftp://ftp.iridiantech.com/pdf/cesg-performance-trials-final-report.pdf>

APPENDIX

Visual Basic Codes

Enrollment Form:

```
Private Sub NewUser_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    'Dim frmNewU As New NewUser
    'frmNewU.Close()
    Dim frmAbout As New About
    frmAbout.Close()
    Dim frmEigen As New EigenMapEyes
    frmEigen.Close()
    Dim frmShowAll As New ShowAllUsers
    frmShowAll.Close()
    Dim frmDel As New DeleteUser
    frmDel.Close()

    Dim c As Integer
    Try
        Dim daI As SqlDataAdapter = New SqlDataAdapter("select UserID
from UserTablo", conn)
        Dim dsI As DataSet = New DataSet
        Catch ex As Exception
            txtID.Text = "1001"
            txtID.Enabled = False
        End Try

        pictureSnap.Enabled = True

        pictureSnap.Visible = False
        pbRefresh.Visible = False

        cbGender.SelectedIndex = 0

    Try

        Dim strSqlO As String = "SELECT UserResimNo FROM UserTablo"
        Dim daa As SqlDataAdapter = New SqlDataAdapter(strSqlO, irisConn2)
        Dim dss As DataSet = New DataSet
        daa.Fill(dss, "sayac")
        otonumaraUT = dss.Tables("sayac").Rows.Count()

        Dim strSqlR As String = "SELECT PhotoID FROM PhotoTablo"
        Dim dar As SqlDataAdapter = New SqlDataAdapter(strSqlR, irisConn2)
        Dim dsr As DataSet = New DataSet
        dar.Fill(dsr, "sayac2")
        ottol = otonumaraUT

        otto2 = otonumaraRT
        Catch ex As Exception
            MsgBox("SQL Database ERROR! Please contact with your
administrator.")
        End Try
    End Sub
```

```

Private Sub lblExit_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles lblExit.Click
    pictureSnap.Enabled = True

    Dim counterPict As Integer
    Dim j As Integer = 0
    Dim jj As Integer = 1
    Try

        Dim irConn As New SqlConnection(conn)
        Dim cmd As New SqlCommand("SELECT PhotoGoz FROM PhotoTablo",
irConn)
        da = New SqlDataAdapter(cmd)
        cbd = New SqlCommandBuilder(da)
        dsPictures = New DataSet
        da.Fill(dsPictures, "goz")

        Catch ex As Exception
            MsgBox("SQL Database ERROR! Please contact with your
administrator.")
        End Try
        lblRefresh.Visible = True
        lblRefresh.Text = "Updating Iris Database"

        pbRefresh.Minimum = 0
        pbRefresh.Maximum = counterPict + 10
        pbRefresh.Visible = True

        While j < counterPict
            Dim arrPic() As Byte =
CType(dsPictures.Tables("goz").Rows(j)("PhotoGoz"), Byte())
            Dim mss As New MemoryStream(arrPic)
            pictureSnap.Image = Image.FromStream(mss)
            pictureSnap.Image.Save("c:\irisrecognition_YM\databaseT\r_eye"
& j + 1 & ".bmp")
            mss.Close()
            j = j + 1
            pbRefresh.Value = j
            Application.DoEvents()
        End While
        kişiSayisi = counterPict / 10
        pbRefresh.Value = j + 9
        Application.DoEvents()

        pictureSnap.Enabled = False

        pbRefresh.Value = j + 10
        Application.DoEvents()

        pbRefresh.Visible = False
        pictureSnap.Enabled = False
        pictureSnap.Visible = False
        Try
            Kill("c:\irisrecognition_ym\temp\*.bmp")
        Catch ex As Exception
            MsgBox("File delete ERROR!")
        End Try

```

```

        Me.Close()
    End Sub

    Private Sub lblSave_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles lblSave.Click

        If txtName.Text = "" Or txtSName.Text = "" Or txtID.Text = "" Then
            Label1.Visible = True
        Else
            Label1.Visible = False
            If gDegisken = 0 Then
Dim ms2 As New MemoryStream
pictureSnap.Image.Save(ms2, pictureSnap.Image.RawFormat)
Dim arrImage2() As Byte = ms2.GetBuffer
            Dim cins As String
            If cbGender.SelectedIndex = 0 Then
                cins = "Mr."
            End If
            If cbGender.SelectedIndex = 1 Then
                cins = "Mrs."
            End If

            otto1 = otto1 + 1
            Try

                Dim strSql2 As String = " INSERT INTO
UserTablo(UserID,UserAd,UserSoyad,UserResimNo,UserResim,UserTelefon,UserCin
s)" & _
                "VALUES ('" &
txtID.Text & "','" & txtName.Text & "','" & txtSName.Text & "','" & otto1 &
"',@UserResim,'" & txtTelephone.Text & "','" & cins & "')"
                Dim cmd2 As New SqlCommand(strSql2, irisConn2)
                With cmd2
                    .Parameters.Add(New SqlParameter("@UserResim",
SqlDbType.Image)).Value = arrImage2
                End With
                irisConn2.Open()
                irisConn2.Close()
            Catch ex As Exception
                MsgBox("SQL Database ERROR! Please contact with your
administrator.")
            End Try
            ElseIf gDegisken <= 10 And gDegisken > 0 Then
                Dim ms As New MemoryStream
                Dim arrImage() As Byte = ms.GetBuffer
                ms.Close()

                otto2 = otto2 + 1
                Try

                    Dim irisConn As New SqlConnection(conn)
                    Dim strSql As String = " INSERT INTO
PhotoTablo(PhotoID,UserID,PhotoGoz)" & _
                    "VALUES ('" & otto2
& "','" & txtID.Text & "')"
                    Dim cmd As New SqlCommand(strSql, irisConn)
                    irisConn.Open()
                    cmd.ExecuteNonQuery()
                    irisConn.Close()

```

```

        Catch ex As Exception
            MsgBox("SQL Database ERROR! Please contact with your
administrator.")
        End Try
    End If

    If gDegisken <= 10 Then
        gDegisken = gDegisken + 1
        If gDegisken <= 10 Then
            lblSave.Text = "Save " & gDegisken & ". Picture"
        End If
    End If
    If gDegisken = 11 Then
        lblSave.Enabled = False
        MsgBox("You Snap Shot 10 Pictures!")

    End If
End If
End Sub

Private Sub lblNewUser_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles lblNewUser.Click
    gDegisken = 0
    Dim c As Integer
    Try
        Dim daI As SqlDataAdapter = New SqlDataAdapter("select UserID
from UserTablo", conn)
        Dim dsI As DataSet = New DataSet
        daI.Fill(dsI, "IdC")
        c = dsI.Tables("IdC").Rows.Count

        txtID.Text = dsI.Tables("IdC").Rows(c - 1)("UserID") + 1
        txtID.Enabled = False
    Catch ex As Exception
        txtID.Text = "1001"
        txtID.Enabled = False
    End Try
    lblSave.Enabled = True
    txtName.Text = ""
    txtSName.Text = ""

    txtTelephone.Text = ""
    lblSave.Text = "Save Passport Photo"
End Sub

Private Sub lblGetSnapShot_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles lblGetSnapShot.Click
    pictureSnap.Visible = True
    Dim i As Integer = 1
    Dim d2 As New IO.FileInfo("C:\Irisrecognition_YM\temp\eye1.bmp")
    Dim res As String
    Dim d1 As IO.FileInfo

    Try
        If gDegisken = 0 Then
            d1 = New IO.FileInfo("C:\Irisrecognition_YM\temp\eyeS.jpg")

            recognition.mysnapportre()
            res = ".jpg"
        Else

```

```

        d1 = New IO.FileInfo("C:\Irisrecognition_YM\temp\eyeS.bmp")

        recognition.snapgoz()
        res = ".bmp"
    End If

    d1.MoveTo("C:\Irisrecognition_YM\temp\eye" & "1" & res)
    pictureSnap.Image =
Image.FromFile("C:\Irisrecognition_YM\temp\eye" & "1" & res)

    Catch ex As Exception
        Try
            While IO.File.Exists("C:\Irisrecognition_YM\temp\eye" & i &
res)
                If IO.File.Exists("C:\Irisrecognition_YM\temp\eye" & i
+ 1 & res) Then
                    i = i + 1
                Else
                    d1.MoveTo("C:\Irisrecognition_YM\temp\eye" & i + 1
& res)
                    pictureSnap.Image =
Image.FromFile("C:\Irisrecognition_YM\temp\eye" & i + 1 & res)
                End If
            End While
        Catch ht As Exception
            pictureSnap.Image =
Image.FromFile("C:\Irisrecognition_YM\temp\eye" & i & res)
        End Try
    End Try

End Sub

```

Users Form:

```
Private Sub ShowAllUsers_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    Dim frmNewU As New NewUser
    frmNewU.Close()
    Dim frmAbout As New About
    frmAbout.Close()
    Dim frmEigen As New EigenMapEyes
    frmEigen.Close()
    Dim frmDel As New DeleteUser
    frmDel.Close()

    daUser = New SqlDataAdapter("select
UserID,UserAd,UserSoyad,UserResimNo from UserTablo", conn)
    daPhoto = New SqlDataAdapter("select UserID,PhotoID from
PhotoTablo", conn)
    daLog = New SqlDataAdapter("select UserID,LogSaat,LogTarih from
LogTablo", conn)

    dtUser = New DataTable("Users")
    dtPhoto = New DataTable("Photos")
    dtLog = New DataTable("Logs")

    ds = New DataSet

    daUser.Fill(dtUser)
    daPhoto.Fill(dtPhoto)
    daLog.Fill(dtLog)

    dr = New DataRelation("Photo_to_User", dtUser.Columns("UserID"),
dtPhoto.Columns("UserID"))
    dr2 = New DataRelation("Log_to_User", dtUser.Columns("UserID"),
dtLog.Columns("UserID"))

    ds.Tables.Add(dtUser)
    ds.Tables.Add(dtPhoto)
    ds.Tables.Add(dtLog)

    ds.Relations.Add(dr)
    ds.Relations.Add(dr2)

    UsersDG.DataSource = ds

End Sub
```

Delete User Form:

```
Private Sub DeleteUser_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
```

```
    Dim frmNewU As New NewUser
    frmNewU.Close()
    Dim frmAbout As New About
    frmAbout.Close()
    Dim frmEigen As New EigenMapEyes
    frmEigen.Close()
    Dim frmShowAll As New ShowAllUsers
    frmShowAll.Close()
    pbDel.Visible = False
    Label2.Visible = False
    Dim strSql As String = "select UserID,UserAd from UserTablo"
    Dim da As SqlDataAdapter = New SqlDataAdapter(strSql, conn)
    Dim ds As DataSet = New DataSet
    da.Fill(ds)
    dgDelete.DataSource = ds
```

```
End Sub
```

```
Private Sub btnDelete_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnDelete.Click
```

```
    Label2.Visible = True
    Label2.Text = "Deleting..."
    pbDel.Visible = True
    pbDel.Minimum = 0
    pbDel.Maximum = 100
    Dim strSqlL As String = "DELETE FROM LogTablo WHERE UserID = '" & txtIDd.Text & "'"
    Dim strSql As String = "DELETE FROM UserTablo WHERE UserID = '" & txtIDd.Text & "'"
```

```
    Dim cmd As SqlCommand = New SqlCommand(strSql, conn)
    Try
```

```
        Dim strSqlP As String = "DELETE FROM PhotoTablo WHERE UserID = '" & txtIDd.Text & "'"
```

```
        Dim cmdP As SqlCommand = New SqlCommand(strSqlP, conn)
        pbDel.Value = 2
        Application.DoEvents()
        cmdP.ExecuteNonQuery()
        cmdP.Connection.Close()
        pbDel.Value = 6
        Application.DoEvents()
        cmdL.Connection.Open()
        cmdL.ExecuteNonQuery()
        cmdL.Connection.Close()
```

```
        pbDel.Value = 10
        Application.DoEvents()
```

```
        cmd = New SqlCommand(strSql, conn)
        cmd.Connection.Open()
        cmd.ExecuteNonQuery()
        cmd.Connection.Close()
```

```

        Catch ex As Exception
            MsgBox("SQL Database ERROR! Please contact with your
administrator.")
        End Try
        pbDel.Value = 15
        Application.DoEvents()

        Dim da As SqlDataAdapter = New SqlDataAdapter("Select * from
PhotoTablo", conn)
        Dim ds As DataSet = New DataSet
        da.Fill(ds, "Ph")
        Dim photoC As Integer = ds.Tables("Ph").Rows.Count()
        pbDel.Value = 18
        Application.DoEvents()

        da = New SqlDataAdapter("select * from UserTablo", conn)
        Dim dsU As DataSet = New DataSet
        Dim userC As Integer = dsU.Tables("Us").Rows.Count
        pbDel.Value = 20
        Application.DoEvents()
        cmd = New SqlCommand("delete from PhotoTablo", conn)
        cmd.Connection.Open()
        cmd.ExecuteNonQuery()
        cmd.Connection.Close()
        pbDel.Value = 30
        Application.DoEvents()

        cmd = New SqlCommand("delete from UserTablo", conn)
        cmd.Connection.Open()
        cmd.ExecuteNonQuery()
        cmd.Connection.Close()
        pbDel.Value = 35
        Application.DoEvents()
        Dim k As Integer = 0
        Dim arrImage() As Byte
        Dim l As Integer = 0
        Dim cmd2 As SqlCommand
        Dim cmd3 As SqlCommand

        While l < userC
            arrImage = CType(ds.Tables("Ph").Rows(l) ("PhotoGoz"), Byte())
            dsU.Tables("Us").Rows(l) ("UserResimNo") = l + 1
            cmd2 = New SqlCommand(" INSERT INTO
UserTablo (UserID,UserAd,UserSoyad,UserResimNo,UserResim,UserTelefon,UserCin
s)" & _
"VALUES ('" & dsU.Tables("Us").Rows(l) ("UserID") & "','" &
dsU.Tables("Us").Rows(l) ("UserAd") & "','" &
dsU.Tables("Us").Rows(l) ("UserSoyad") & "','" &
dsU.Tables("Us").Rows(l) ("UserResimNo") & "','@UserResim,'" &
dsU.Tables("Us").Rows(l) ("UserTelefon") & "','" &
dsU.Tables("Us").Rows(l) ("UserCins") & "')", conn)
            cmd2.Parameters.Add(New SqlParameter("@",
SqlDbType.Image)).Value = arrImage

            cmd2.Connection.Open()
            cmd2.ExecuteNonQuery()
            cmd2.Connection.Close()

            l = l + 1
        End While

```

```

pbDel.Value = 60
Application.DoEvents()

Dim m As Integer
l = 0
While k < photoC
    If l Mod 10 = 0 Then
        l = l + 1
    End If
    m = l - 1
    arrImage = CType(ds.Tables("Ph").Rows(k) ("PhotoGoz"), Byte())
    cmd3 = New SqlCommand("INSERT INTO PhotoTablo
(PhotoID,UserID,PhotoGoz) VALUES" & _
("'" & dsU.Tables("Us").Rows(m) ("UserResimNo") + k & "',''" &
ds.Tables("Ph").Rows(k) ("UserID") & "','@PhotoGoz)", conn)
    cmd3.Parameters.Add(New SqlParameter("@PhotoGoz",
SqlDbType.Image)).Value = arrImage
    cmd3.Connection.Open()
    cmd3.ExecuteNonQuery()
    cmd3.Connection.Close()
    k = k + 1
End While

pbDel.Value = 90
Application.DoEvents()
Dim strSql2 As String = "select UserID,UserAd from UserTablo"
Dim da2 As SqlDataAdapter = New SqlDataAdapter(strSql2, conn)
Dim ds2 As DataSet = New DataSet

da2.Fill(ds2)
dgDelete.DataSource = ds2

pbDel.Value = 100
Application.DoEvents()
pbDel.Visible = False
pbDel.Visible = False
Label2.Visible = False

End Sub

Private Sub btnExit_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnExit.Click
    Label2.Enabled = True
    Label2.Visible = True
    Label2.Text = "Updating Iris Database..."

    ks = 0

    Dim dae As SqlDataAdapter = New SqlDataAdapter(cmd)
    Dim dse As DataSet = New DataSet
    dae.Fill(dse, "goz")
    counterPict = dse.Tables("goz").Rows.Count
    pbDel.Minimum = 0
    pbDel.Maximum = counterPict + 10
    pbDel.Visible = True
    ks = counterPict / 10
    If ks = 0 Then
        Me.Close()
    Else

```

```

        recognition = New EyeJudgeDLL.EyeJudgeDLLclassClass

        While j < counterPict
            Dim arrPic() As Byte =
CType(dse.Tables("goz").Rows(j)("PhotoGoz"), Byte())
            Dim mss As New MemoryStream(arrPic)
            pictureSnap.Image = Image.FromStream(mss)

pictureSnap.Image.Save("c:\irisrecognition_YM\databaseT\r_eye" & j + 1 &
".jpg")

            mss.Close()
            j = j + 1
            pbDel.Value = j
            Application.DoEvents()
        End While
        recognition.saveeigenavomegat(ks, 20)
        pbDel.Value = j + 10
        Application.DoEvents()
        j = 1
        While j <= counterPict
            j = j + 1
        End While
        Me.Close()
    End If
End Sub

Dim tiklanan_nokta As Point
Private Sub DeleteUser_MouseDown(ByVal sender As System.Object, ByVal e
As System.Windows.Forms.MouseEventArgs) Handles MyBase.MouseDown
    tiklanan_nokta = New Point(e.X, e.Y)

End Sub

Private Sub DeleteUser_MouseMove(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles MyBase.MouseMove
    If e.Button = MouseButtons.Left Then
        Dim koordinatlar As Point
        koordinatlar = Control.MousePosition
        koordinatlar.Offset(-tiklanan_nokta.X, -tiklanan_nokta.Y)
        Me.Location = koordinatlar
    End If
End Sub

```

Normal Mode:

```
Private Sub RunSystem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles RunSystem.Click
    Dim frmVerification As New Verification
    frmVerification.MdiParent = Me
    frmVerification.Show()

End Sub

Private Sub exitM_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles exitM.Click

End
End Sub

Private Sub allUsersM_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles allUsersM.Click
    Dim frmUsers As New ShowAllUsers
    frmUsers.MdiParent = Me
    frmUsers.Show()
End Sub

Private Sub newUserM_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles newUserM.Click
    Dim frmNUser As New NewUser
    frmNUser.MdiParent = Me
    frmNUser.Show()
End Sub

Private Sub mDelete_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles mDelete.Click
    Dim frmDUser As New DeleteUser
    frmDUser.MdiParent = Me
    frmDUser.Show()
End Sub

Private Sub IrisRecognitionSystem_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
    Dim frmNewU As New NewUser
    frmNewU.Close()
    Dim frmAbout As New About
    frmAbout.Close()
    Dim frmEigen As New EigenMapEyes
    frmEigen.Close()
    Dim frmShowAll As New ShowAllUsers
    frmShowAll.Close()
    Dim frmDel As New DeleteUser
    frmDel.Close()
End Sub

Private Sub MenuItem8_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MenuItem8.Click
    Dim frmAbout As New About
    frmAbout.Show()
End Sub
```

```

    Private Sub MenuItem9_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MenuItem9.Click
        Dim frmP As New MainP
        frmP.Show()
        Me.Close()
    End Sub

    Private Sub MenuItem10_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MenuItem10.Click
        Dim frmS As New Main
        frmS.Show()
        Me.Close()
    End Sub

    Private Sub vFromC_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles vFromC.Click
        Dim frmS As New VerifyPC
        frmS.Show()

    End Sub

```

Small Skin:

```
Private Sub Main_MouseDown(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles MyBase.MouseDown
    tiklanan_nokta = New Point(e.X, e.Y)

End Sub

Private Sub Main_MouseMove(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles MyBase.MouseMove
    If e.Button = MouseButtons.Left Then
        Dim koordinatlar As Point
        koordinatlar = Control.MousePosition
        koordinatlar.Offset(-tiklanan_nokta.X, -tiklanan_nokta.Y)
        Me.Location = koordinatlar
    End If
End Sub

Private Sub Label1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label1.Click
    End
End Sub

Private Sub lblRun_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles lblRun.Click
    Dim frmVerification As New Verification
    frmVerification.Show()
End Sub

Private Sub lblUsers_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles lblUsers.Click
    Dim frmUsers As New ShowAllUsers
    frmUsers.Show()
End Sub

Private Sub lblNUser_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles lblNUser.Click
    Dim frmNUser As New NewUser
    frmNUser.Show()
End Sub

Private Sub lblDel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles lblDel.Click
    Dim frmDUser As New DeleteUser
    frmDUser.Show()
End Sub

Private Sub lblNormal_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles lblNormal.Click
    Dim frmN As New IrisRecognitionSystem
    frmN.Show()
    Me.Visible = False

End Sub

Private Sub Label2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label2.Click
    Dim frmE As New EigenMapEyes
    frmE.Show()
End Sub
```

```

    Private Sub lblAbout_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles lblAbout.Click
        Dim frmAbout As New About
        frmAbout.Show()
    End Sub

    Private Sub Main_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim frmNewU As New NewUser
        frmNewU.Close()
        Dim frmAbout As New About
        frmAbout.Close()
        Dim frmEigen As New EigenMapEyes
        frmEigen.Close()
        Dim frmShowAll As New ShowAllUsers
        frmShowAll.Close()
        Dim frmDel As New DeleteUser
        frmDel.Close()
    End Sub

```

Verification Form:

```
Private recognition As EyeJudgeDLL.EyeJudgeDLLclassClass
Dim conn As SqlConnection = New SqlConnection("user
id=?;password=?;initial catalog=IrisRecognitionDB;data source=?)
Private Sub optCom1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles optCom1.CheckedChanged,
optCom2.CheckedChanged
    If (sender Is optCom1) Then
        miComPort = 1
    Else
        miComPort = 2
    End If
End Sub

Private Sub Verification1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    recognition = New EyeJudgeDLL.EyeJudgeDLLclassClass
    recognition.load1(1, eigenM)
    recognition.load3(1, omegaW)
    Dim da As SqlDataAdapter = New SqlDataAdapter("SELECT * FROM
CodeTablo", conn)
    Dim ds As DataSet = New DataSet
    codeS = ds.Tables("codes").Rows(0)(0)
    codeT = ds.Tables("codes").Rows(0)(1)
End Sub

Function GetSnapShotAndVerify(ByVal rNO As Integer, ByVal onay As Integer)
    Dim rmse, tolerans As Double
    Dim flag As Integer = 0

odak:

    d1.MoveTo("C:\Irisrecognition_YM\temp\eye" & "1" & ".bmp")
    pbVerify.Image =
Image.FromFile("C:\Irisrecognition_YM\temp\eye" & "1" & ".bmp")

    Catch ex As Exception
        Try
            While IO.File.Exists("C:\Irisrecognition_YM\temp\eye" & i &
".bmp")
                If IO.File.Exists("C:\Irisrecognition_YM\temp\eye" & i
+ 1 & ".bmp") Then
                    i = i + 1
                Else
                    d1.MoveTo("C:\Irisrecognition_YM\temp\eye" & i + 1
& ".bmp")
                    pbVerify.Image =
Image.FromFile("C:\Irisrecognition_YM\temp\eye" & i + 1 & ".bmp")
                End If
            End While
            Catch ht As Exception
                pbVerify.Image =
Image.FromFile("C:\Irisrecognition_YM\temp\eye" & i & ".bmp")
            End Try
        End Try
    Try
        pbVerify.Image.Save("c:\irisrecognition_YM\temp\eyeS.bmp")
    Catch ex As Exception
    End Try
```

```

da.Fill(ds, "Kullanıcı")
toplamKS = ds.Tables("Kullanıcı").Rows.Count()

recognition.gozkarsilastir(1, oPutK, average, eigenM, omegaW, rNO,
toplamKS, 20)
If rmse < Val(txtTol.Text) Then
    onay = 1
Else
    If flag < 3 Then
        flag = flag + 1
        GoTo odak
    End If
    onay = 2
End If
Return onay
End Function

Function terminateP()
    SeriPort.Close()
    Me.Close()
End Function

Private Sub btnStart_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnStart.Click
    SeriPort = New Rs232
    Try
        '// Setup parameters
        With SeriPort
            .Port = miComPort
            .BaudRate = 2400
            .DataBit = 8
            .StopBit = Rs232.DataStopBit.StopBit_1
            .Parity = Rs232.DataParity.Parity_None
        End With
        .Timeout = 3000
        '// Initializes port

        SeriPort.Open()
        Dim deger As Double
        Dim count, i, rNO, onay As Integer
        Dim pinNo, uIsim, cins, saat, tarih As String
        count = 0
        i = 0
        If flag = 0 Then
            SeriPort.Write("A" & codeS)
            SeriPort.Write("A" & codeT)
            SeriPort.Write("A" & Val(txtSure.Text))
        End If
        flag = 1
        While deger <> 60000
            Try
                deger = 9
                While deger = 9
                    Try
                        SeriPort.Write(0)
                    Catch ex As Exception
                        Label6.Text = "Seri Port 5 BYTE Read Error!"
                    End Try
                End While
            End Try
        End While
    End Try
End Sub

```

```

        If deger = 60000 Then
            terminateP()
            SeriPort.Write(0)
            Me.Close()
        End If
        deger = deger - 10000

        pinNo = deger
        txtDeger.AppendText(", " & pinNo)
        Dim strSql As String = "SELECT
UserID,UserSoyad,UserCins,UserResimNo FROM UserTablo WHERE UserID='" &
pinNo & "'"

        Dim da As SqlDataAdapter = New SqlDataAdapter(strSql, conn)
        Dim ds As DataSet = New DataSet
        If ds.Tables("kullanıcı").Rows.Count <> 0 Then
            rNO = ds.Tables("kullanıcı").Rows(0) ("UserResimNo")
            Dim yaz As String = "A" & cins & uIsim & "$"
            lblName.Text = cins & uIsim
            ds.Clear()
            ds.Reset()
            SeriPort.Write(yaz)
            While deger <> 3
                Try
                    SeriPort.Read(1)
                    deger = SeriPort.InputStreamString
                    SeriPort.Write(0)
                Catch ex As Exception
                    Label6.Text = "Seri Port 1 BYTE Read Error!"
                End Try

            End While

            If deger = 3 Then
                onay = GetSnapShotAndVerify(rNO, onay)
                If onay = 1 Then
                    SeriPort.Write("A1")
                    SeriPort.Write(0)
                    Try
                        Dim strSql2 As String = " INSERT INTO
LogTablo(UserID,LogSaat,LogTarih)" & _
                        "VALUES ('" & pinNo & "','" & saat &
                        "','" & tarih & "')"
                        Dim cmd2 As New SqlCommand(strSql2, conn)
                        conn.Open()
                        cmd2.ExecuteNonQuery()
                        conn.Close()
                    Catch ex As Exception
                        MsgBox("SQL Database ERROR! Please contact
with your administrator.")
                    End Try
                Else

                    SeriPort.Write("A2")
                    SeriPort.Write(0)

                End If
            End If
        Else

```

```

        SeriPort.Write("A1$")
        SeriPort.Write(0)
    End If
Catch ex As Exception
    GoTo loo
End Try
pinNo = 0

End While
End Sub

Private Sub btnExit_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnExit.Click
    Me.Close()
End Sub

```

About Form:

```
Private Sub About_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    Dim frmNewU As New NewUser
    frmNewU.Close()
    Dim frmEigen As New EigenMapEyes
    frmEigen.Close()
    Dim frmShowAll As New ShowAllUsers
    frmShowAll.Close()
    Dim frmDel As New DeleteUser
    frmDel.Close()

End Sub
```

Pic Basic Pro Codes

```
***** LCD PARAMETERS *****
DEFINE CHAR_PACING 3000      '1000 us delay for serial communication
DEFINE LCD_BITS 8           'Data BUS of LCD is 8-bit
DEFINE LCD_DREG PORTD       'Data BUS of LCD is located at PORT-D
DEFINE LCD_DBIT 0           'First data bit is 0. bit
DEFINE LCD_RSREG PORTE      'LCD yazmaç seçimi E portu
DEFINE LCD_RSBIT 0          'at 0. bit
DEFINE LCD_EREG PORTE       'Enable pin of LCD located at PORT-E
DEFINE LCD_EBIT 1           'at 1. bit
DEFINE LCD_LINES 4          'Lines number of LCD
DEFINE LCD_COMMANDUS 2000   'Command delay time is 2000 us
DEFINE LCD_DATAUS 50        'Data delay time is 50 us

***** STARTING PROCEDURES and DEFINING VARIABLES *****

INCLUDE "modedefs.bas"

ADCON1 = 7
Low PORTE.2
Pause 100
flag          VAR BIT
flag1         VAR BIT
counter       VAR BYTE
access        VAR BYTE
dt            VAR BYTE
scode        VAR BYTE[10]
scodet        VAR BYTE[10]
scodeb        VAR BYTE[10]
scount        VAR BYTE
acount        VAR BYTE
ID            VAR BYTE[4]
IDS           VAR WORD
UNAME         VAR BYTE[20]
Sver          VAR BYTE
ster          VAR BYTE
SO            VAR PORTC.6
SI            VAR PORTC.7

***** MAIN PROGRAM *****
Start:
SerOut SO,N2400,[10]
TRISB=%11110000
TRISA=%00000000
LCDOut $fe," Please wait",$fe,$c0,"System initializing"
SerIn SI,N2400,[A]
```

```

For scout=0 TO 9
    SerIn SI,N2400,scode[scout]
Next scout
SerIn SI,N2400,["A"]
For scout=0 TO 9
    SerIn SI,N2400,scodet[scout]
Next scout
SerIn SI,N2400,["A"]
SerIn SI,N2400,dt
program:
    sver=0
    ster=0
    access=0
    flag=0
    flag1=0
    account=0
    counter=0
    PORTB=0
    LCDOut "IRIS RECOGNITION",$fe,"SYSTEM",$fe,"PLEASE ENTER YOUR ID"
loop:
    High PORTB.1
    IF PORTB.4=1 Then
        ID[counter]=1
        GoSub keypadr
    EndIF
    IF PORTB.5=1 Then
        ID[counter]=4
        GoSub keypadr
    EndIF
    IF PORTB.6=1 Then
        ID[counter]=7
        GoSub keypadr
    EndIF
    IF PORTB.7=1 Then
        GoTo program
    EndIF
    Low PORTB.1
    High PORTB.2
    IF PORTB.4=1 Then
        ID[counter]=2
        GoSub keypadr
    EndIF
    IF PORTB.5=1 Then
        ID[counter]=5
        GoSub keypadr
    EndIF
    IF PORTB.6=1 Then
        ID[counter]=8
        GoSub keypadr
    EndIF

```

```

IF PORTB.7=1 Then
    ID[counter]=0
    GoSub keypadr
EndIF
Low PORTB.2
High PORTB.3
IF PORTB.4=1 Then
    ID[counter]=3
    GoSub keypadr
EndIF
IF PORTB.5=1 Then
    ID[counter]=6
    GoSub keypadr
EndIF
IF PORTB.6=1 Then
    ID[counter]=9
    GoSub keypadr
EndIF
IF PORTB.7=1 Then
    IF counter=0 Then
        Pause 300
        GoSub secure
    EndIF
EndIF
Low PORTB.3
IF counter<4 Then
    GoTo loop
EndIF
LCDOut $fe,$1," Please wait",$fe,$c0,"Your ID is verifying"
SerOut SO,N2400,[#1]
IDS=1000*ID[0]+100*ID[1]+10*ID[2]+ID[3]
SerOut SO,N2400,[#IDS]
SerIn SI,N2400,["A"]           'Waiting for User name
For counter=0 TO 20
    SerIn SI,N2400,UNAME[counter]
    account=account+1
    IF UNAME[counter]="$" Then
        counter=20
    EndIF
Next counter
IF UNAME[0]="1" Then
LCDOut," Your ID is not",$fe,"REGISTERED"
    Pause 2000
    GoTo program
EndIF
LCDOut $fe,$1," WELCOME",$fe,$c0
account=account-2
For counter=0 TO account
    LCDOut UNAME[counter]
Next counter

```

```

LCDOut $fe,$d4,"Press # to snapshot"
PORTB=0
High PORTB.3
IF PORTB.7=1 Then
    SerOut SO,N2400,[#3]
    GoTo label
EndIF
GoTo ws
Low PORTB.3
label:
    LCDOut $fe,1,"  Please wait",$fe,$c3,"Getting access"
    SI,N2400,["A"]
    SerIn SI,N2400,access
    IF access=49 Then
        '1 'in ASCII karşılığı 49
        LCDOut $fe,1," ACCESS VERIFIED"
        GoSub door
        GoTo program
    Else
        LCDOut $fe,1," ACCESS DENIED"
        Pause 2000
        GoTo program
    EndIF
End
***** KEY-PAD Routine *****
keypadr:
    IF flag=0 Then
        LCDOut $fe,1'Clear LCD
        LCDOut "PRESS * for RE-Enter"
        LCDOut $fe,$c8
    EndIF
    counter=counter+1
    Pause 300
    flag=1
    Return
secure:
    scount=0
    LCDOut $fe,1,"Enter security code",$fe,$c5
secloop:
    High PORTB.1
    IF PORTB.4=1 Then
        scodeb[scount]="1"
        GoSub s
    EndIF
    IF PORTB.5=1 Then
        scodeb[scount]="4"
        GoSub s
    EndIF
    EndIF
    Low PORTB.1

```

```

High PORTB.2
IF PORTB.4=1 Then
    scodeb[scount]="2"
    GoSub s
ENDIF
IF PORTB.5=1 Then
    scodeb[scount]="5"
    GoSub s
ENDIF
IF PORTB.6=1 Then
    scodeb[scount]="8"
    GoSub s
ENDIF
IF PORTB.7=1 Then
    scodeb[scount]=0
    GoSub s
ENDIF
Low PORTB.2
High PORTB.3
IF PORTB.4=1 Then
    scodeb[scount]="3"
    GoSub s
ENDIF
IF PORTB.5=1 Then
    scodeb[scount]="6"
    GoSub s
ENDIF
IF PORTB.6=1 Then
    scodeb[scount]="9"
    GoSub s
ENDIF
Low PORTB.3
IF scount<10 Then
    GoTo secloop
ENDIF
IF sver=9 Then
    flag1=1
    LCDOut $fe,1," EMERGENCY-CASE",$fe,$c5,"ACTIVATED"
    High PORTB.1
back:
    IF PORTB.7=1 Then 'pressed key is "*"
        LCDOut $fe,1," EMERGENCY-
CASE",$fe,$c5,"TERMINATED",$fe,$d4," DOOR IS CLOSING"
        Pause 3000
        GoSub door
        GoTo program
    EndIF
    GoTo back
EndIF

```

```

    IF ster=9 Then
        LCDOut $fe,1," SYSTEM TERMINATED"
        SerOut SO,N2400,["60000"]
    EndIF
    LCDOut $fe,1,"  WRONG CODE"
    Pause 2000
    GoTo program
    Return
S:
    LCDOut "*"
    IF scodeb[scount]=scode[scount] Then
        EndIF
    IF scodeb[scount]=scodet[scount] Then
        EndIF
    Pause 300
    scount=scount+1
    Return
door:
    High PORTA.0
    Pause 300
    Low PORTA.0
    IF flag1=0 Then
        For counter=49 TO dt
            Pause 1000
        Next counter
        High PORTA.0
        Pause 300
        Low PORTA.0
    EndIF
    Return

```