

KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY
INSTITUTE OF DISTANCE LEARNING

FACE RECOGNITION USING PRINCIPAL COMPONENT ANALYSIS

KNUST

BY

GYASI SILAS BAAH (B.Sc. Mathematics)

PG 6320311

A thesis submitted to the Department of Mathematics, in partial fulfillment of the
requirements for the award of degree of

Master of Science

Industrial Mathematics

SEPTEMBER, 2013

DECLARATION

I hereby declare that, this submission is my own work towards the Master of Science degree and that, to the best of my knowledge, it contains no material previously published by another person nor material which has been accepted for the award of any degree of the University except where due acknowledgement has been made in the text.

Gyasi Silas Baah PG 6320311

KNUST

Student Name & ID

Signature

Date

Certified by:

Mr K. F. Darkwah

.....

Supervisor's Name

Signature

Date

Prof. S. K. Amponsah

.....

Head of Department's Name

Signature

Date

Prof. I.K.Dontwi

.....

.....

Dean, IDL

Signature

Date

DEDICATION

This work is dedicated to my parents, Mr. and Mrs. Edward Gyasi, my brother, Paul Osei Gyasi and my sister, Eunice Owusuaa Gyasi.

KNUST

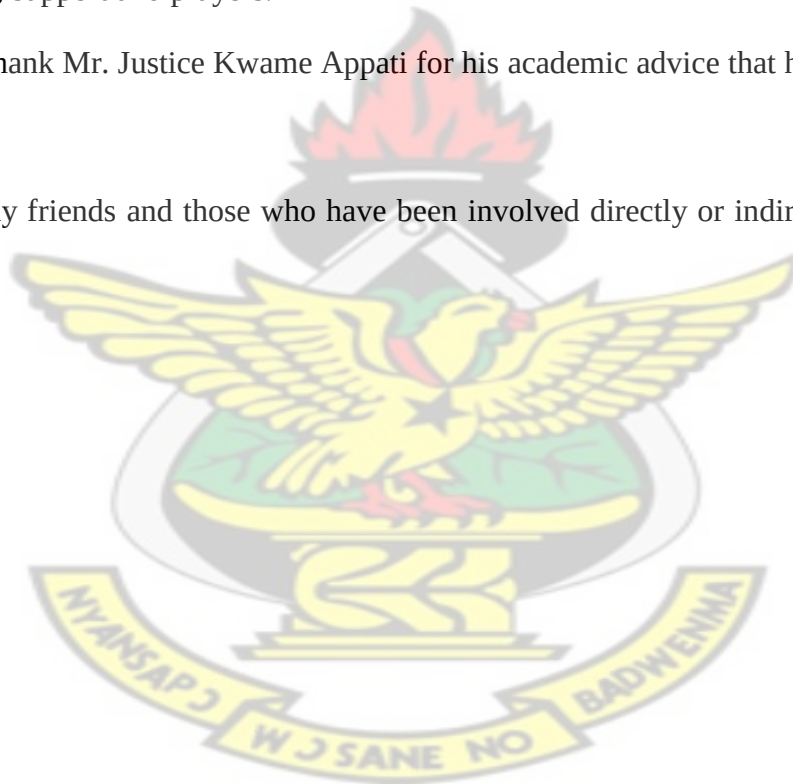


ACKNOWLEDGEMENT

To God be the glory for how far he has brought me, giving me the strength and wisdom to finish this research. I would like to express my profound gratitude to my supervisor, Mr. K. F. Darkwah, for his understanding and professionalism in guiding me throughout this work. Very special thanks go to my family, especially my parents who gave me words of encouragement, support and prayers.

I also want to thank Mr. Justice Kwame Appati for his academic advice that helped me in this project.

Lastly, to all my friends and those who have been involved directly or indirectly, thank you very much.



ABSTRACT

Face Recognition is the process of identification of a person by their facial image. This technique makes it possible to use the facial images of a person to authenticate him into a secure system, for criminal identification, for passport verification, surveillance, etc... Face recognition is a popular research area where there are different approaches studied in literature.

In this thesis, a holistic Principal Component Analysis (PCA) based method, namely Eigenface method is studied and implemented on the Faces 94 database. This approach treats face recognition as a two-dimensional recognition problem. Face images are projected onto a face space that encodes best variation among known face images. The face space is defined by eigenface which are eigenvectors of the set of faces, which may not correspond to general facial features such as eyes, nose, and lips. Face will be categorized as known or unknown face after matching with the present database. Experimental results in this thesis showed that an accuracy of 98.8158% was achieved. The variable reducing theory of PCA accounts for the smaller face space than the training set of face.

TABLE OF CONTENTS

Declaration.....	ii
Dedication	iii
Acknowledgement	iv
Abstract.....	v
Table of Contents	vi
List of Figures.....	x
CHAPTER 1.....	1
Introduction.....	1
1.0 Background Of The Study	1
1.1. Applications	4
1.2. Statement Of The Problem.....	6
1.3. Objectives Of The Study.....	6
1.4. Methodology	7
1.5. Justification Of The Study	7
1.6. Organization Of The Study	8
CHAPTER 2.....	9
Literature Review	9
2.0 Introduction.....	9
2.1 Recent Approaches To Face Recognition.....	9
2.2 Human Face Recognition.....	10
2.3 Machine Recognition Of Faces.....	11
2.3.1 Statistical Approaches.....	13

2.3.1.1 Template Matching	14
2.3.1.2 Face Detection And Recognition By Pca.....	15
2.3.1.3 Feature-Based Approaches	18
2.3.1.4 Geometry Feature Based Methods.....	20
2.3.1.5 Other Statistical Approaches.....	22
2.3.1.6 Neural Network Based Approaches.....	27
2.3.2 Hybrid Approaches	29
2.3.3 Other Approaches	29
CHAPTER 3	32
Methodology	32
3.0 Introduction.....	32
3.1 Principal Component Analysis	34
3.1.1 Deriving Pca: Eigenvectors Of Covariance Matrix	35
3.2 Eigen Face Approach.....	36
3.2.1 Eigenvectors And Eigenvalues	39
3.2.2 General Formula For The Solution Of The Eigenvalue Problem	39
3.2.2.1 Worked Example	40
3.2.3 Properties Of Eigenvectors And Eigenvalues.....	41
3.3 Calculating Eigenfaces.....	42
3.3.1 Using Eigenface To Classify A Face Image	47
3.3.1.1 Worked Example	50
3.3.2 Summary Of Eigenface Recognition	55
3.3.3 Significance Of Eigenface Approach.....	56
CHAPTER 4.....	57
4.0 Introduction.....	57
4.1 Data Collection	57
4.2 Face Feature Analysis	59

4.2.1	Machine Specifications	59
4.2.2	Face Feature Extraction Algorithm.....	59
4.3	Face Recognition Algorithm.....	70
4.4	Algorithm Performance	73
4.5	Condition For Rejection.....	74
CHAPTER 5.....		75
5.0 CONCLUSION		75
5.1	Recommendation	76
References.....		77
Appendix.....		83



KNUST



LIST OF FIGURES

Fig. 3.1(a) Example of images from the ORL database.....	46
Fig. 3.1(b) Mean face obtained from the ORL database.....	46
Fig. 4.1 First fifteen images of the 1 st person with different facial expressions.....	58
Fig. 4.2 Colour image of first person in the database.....	60
Fig. 4.3(a) Red component of Fig. 4.2.....	60
Fig. 4.3(b) Green component of Fig. 4.2.....	61
Fig. 4.3(c) Blue component of Fig. 4.2.....	61
Fig. 4.4 Matrix representation of grey image (Fig. 4.5).....	62
Fig. 4.5 Grey image of Fig. 4.2.....	63
Fig. 4.6 Vector representation of Fig. 4.4.....	63
Fig. 4.7 Matrix rep. of mean face as calculated from the dataset.....	64
Fig. 4.8 Mean face obtained from the dataset.....	65
Fig. 4.9 Matrix rep. of centered image as calculated from the dataset.....	66
Fig. 4.10 Matrix rep. of covariance matrix.....	66
Fig. 4.11 Vector representation of eigenvalues.....	67
Fig. 4.12 A plot of eigenvalues.....	67
Fig. 4.13 Matrix rep. of eigenvectors of covariance matrix.....	68
Fig. 4.14 Matrix rep. of eigenvectors of retained eigenvalues in dataset.....	68
Fig. 4.15 Matrix rep. of eigenface as calculated in the database.....	69
Fig. 4.16 Matrix rep. of projected images as calculated in the dataset.....	70
Fig. 4.17 A test image of Fig. 4.2 with corresponding equivalent image in the training set.....	73

CHAPTER 1

INTRODUCTION

This chapter provides an introduction to face recognition as a biometric technology, together with its applications.

1.0 BACKGROUND OF THE STUDY

Face recognition is the ability to recognize people by their facial characteristics. Over years it happens to be one of the increasing popular trends in image analysis and processing. This kind of technology has many benefits which causes it to be used in several different areas which push more researchers into identifying some peculiar problem still existing in its current implementation as a biometric system.

Biometrics simply refers to the term that encompasses the application of modern statistical methods to the measurements of biological objects (Oxford English Dictionary, 2004). As a check to the dangers posed by fraudulent use of identity documents, a lot of biometric technologies are emerging. Among these are:

- Voice recognition
- Fingerprint recognition
- Iris recognition
- face recognition and
- gait recognition

These are referred to as biometric identifiers which can be used for distinctive physiological and behavioural characterization of persons.

However these characteristics should satisfy the following properties:

- **Robustness:** This means that the biometric should be sufficiently invariant (Permanence) over a period of time and thus maintain a low intra-class variability.
- **Distinctiveness:** This indicates that biometric identifiers should differentiate (Uniqueness) any two persons and thus have large inter-class variability.
- **Availability:** Ideally, a biometric identifier should be possessed by every person (Universality).
- **Accessibility:** The characteristic should be easy to acquire (Collectability).

Since biometric system is essentially a pattern-recognition system, the need for data acquisition, pre-processing, data representation and decision making is required. This encompasses the extraction of physiological or behavioural characteristic from persons beforehand and using it as a template database to be used later for recognition. Biometric identifiers which are conceptually unique attributes are today portrayed as the panacea to verify person's identity.

Mainly three approaches are available to proving a person's identity and to provide "the right person with the right privileges, the right access and at the right time" (Woodward et al, 2003). These are:

- **Something you have:** The associated service or access is received through the presentation of physical objects like, keys, identity card, smart card, etc., in possession of the concerned person.
- **Something you know:** A pre-defined secret knowledge such as, password permits to access a service.
- **Something you are:** The access to a service can be achieved through the presentation of measurable biometric traits, such as biometric measures.

The third approach has some significant advantages over the first two since it is difficult to share, steal or forge and neither can someone forget it at home while to his workplace. This indeed provides a higher security level in identity prove with a high degree of certainty in response to the dangers posed by fraudulent use of identity documents.

Face recognition is an integral part of biometrics and it has a number of strengths to be recommended over its peer in certain circumstances, (Chellappa et al, 1995).

Automated face recognition is an interesting computer vision problem with many commercial and law enforcement applications. User verification and user access control, crowd surveillance, enhanced human computer interaction all become possible if an effective face recognition system can be implemented making it an area of active research to solve its outstanding problem.

Recognizing faces is something that people usually do effortlessly and without much conscious thought, yet it has remained a difficult problem in the area of computer vision, where some twenty (20) years of research is just beginning to yield useful technological solutions. As a biometric technology, automated face recognition has a number of desirable properties that drives research into practical techniques development where computers can be made to conduct facial database searches on live image as one-to-one or one-to-many verifications with unprecedented accuracy in a split-second processing. Users can be granted secure access to their computer, mobile devices, or for online e-commerce, simply by looking into their web camera.

Just like human beings, computer algorithms to perform face analysis are also divided into detection, recognition and expression understanding. Face detection is about determining the locations and sizes of faces in an image, separating them from other non-face objects.

Recognition, on the other hand, is about establishing the identity of the person from image(s) of his face. This is done with reference to a database of known faces.

The problem of face recognition can be stated as ‘identifying an individual from images of the face and encompasses a number of variations other than the most familiar applications.

Face recognition and detection can be achieved using technologies related to computer science where features extracted from a face are processed and compared with similarly processed faces present in the database. If a face is recognized it is known in the database else it is unknown. In surveillance system if an unknown face appears more than one time then it is stored in database for further recognition. These steps are very useful in criminal identification. In general, face recognition techniques can be divided into two groups based on the face representation they use:

- i. Appearance-based, which uses texture features and is applied to either whole-face or specific regions in a face image and
- ii. Feature-based, which uses geometric facial features (mouth, eyes, brows, cheeks etc.), and geometric relationships between them.

The main challenges of face recognition today are: handling rotation in depth and broad lighting changes, together with personal appearance changes. Even under good conditions, however, accuracy needs to be improved.

1.1. APPLICATIONS

Biometrics has been widely used in forensics applications such as criminal identification and prison security. It also has a strong potential to be widely adopted in civilian applications such as e-Banking, e-Commerce, and access control. Due to the huge increase in the use of

electronic transactions, e-Banking and e-Commerce are becoming the most important emerging application of biometrics. Credit card and smart card security, ATM security, check cashing and fund transfers; online transactions and web access are some of the examples of the applications. The token-based authentications used in physical access control are gradually replaced by biometric systems. Similarly, knowledge-based authentication (e.g. password) for remote login and data access applications are substituted by biometrics. Other biometric applications include welfare disbursement, immigration checkpoints, national ID, voter and driver registration, and time and attendance. Although very reliable methods of biometric personal identification exist, for example, fingerprint analysis and retinal or iris scans, these methods rely on the cooperation of the participants, whereas a personal identification system based on analysis of frontal or profile images of the face is often effective without the participant's cooperation or knowledge. Commercial and law enforcement applications of FRT range from static, controlled-format photographs to uncontrolled video images, posing a wide range of technical challenges and requiring an equally wide range of techniques from image processing, analysis, understanding, and pattern recognition. One can broadly classify FRT systems into two groups depending on whether they make use of static images or of video. Within these groups, significant differences exist, depending on the specific application. The differences are in terms of image quality, amount of background clutter (posing challenges to segmentation algorithms), variability of the images of a particular individual that must be recognized, availability of a well-defined recognition or matching criterion, and the nature, type, and amount of input from a user. A general statement of the problem of machine recognition of faces can be formulated as follows: given still or video images of a scene, identify or verify one or more persons in the scene using a stored database of faces.

1.2. STATEMENT OF THE PROBLEM

Traditionally, student's attendance in KNUST is taken manually by using attendance sheet given by lecturer in class and not a system. With this manual system, there are some cases that student can cheat by asking their friends to tick or sign for them. This occurs because the students just want to fulfill the 80% of the attendance so that they can seat for the final examination at the end of the semester. Lecturers can't monitor for all students in the class and it is difficult for a lecturer to record the attendance of students accurately and efficiently, since they are responsible to monitor the entire student's attendance for the whole semester. For those students that fail to meet the 80% of their attendance rate will be given a reminder as a warning from faculty.

Because of this problem, a system may be needed in order to record the attendance of the students more accurately without the manual trace by lecturers. A face recognition device will be provided at each class and laboratory at the faculty. This system will record the attendance of students in class when the class begins and at the end of class. This is to ensure that the students have attended the class until the lecture ended by the lecturer.

1.3. OBJECTIVES OF THE STUDY

Objectives of this thesis are:

1. To use Principal Component Analysis to efficiently and effectively reduce the dimensions of the extracted data from the face data set.
2. Using results obtained to recognize person.

1.4. METHODOLOGY

Face images from a standard database which is a secondary source data will be extracted and used for this thesis. There are a number of standard databases available for research purposes which could be used to test the performance of face recognition system. For the purposes of this project Faces 94 database is considered. It contains 152 individuals with 180 x 200 pixels in resolution and in portrait format, 20 of which are female, 112 as males and 20 as male staff taken at different angles and variations.

The method of solution will be PCA procedure.

The analysis will be coded with MATLAB which will be implemented in MATLAB 2012a version.

Literature for this thesis will include resources available on the internet, thus www.wikipedia.org, published and unpublished papers and KNUST library.

1.5. JUSTIFICATION OF THE STUDY

It saves hours of teachers' time recording attendance and lateness and provides at a glance who's in and who's out.

It also makes it easy to track poor attendance and lateness patterns, which would allow for and early intervention.

It would also solve the problem of impersonation where a student can make a false registration in someone's name.

Again, this thesis will serve as a springboard to generate interest and further research into the other aspects of biometrics in Ghana. This stems from the fact that face recognition is a multifaceted phenomenon and no one research is capable of addressing it in full. This makes this study justifiable and worthwhile.

1.6. ORGANIZATION OF THE STUDY

This thesis consists of five chapters.

Chapter 1 is the introduction, which describes the background of the study, statement of the research problem, objectives of the study, methodology, justification and organization of the study. Also Chapter 2 comprises the review of related literature. This focuses on the conceptual base of the study. Chapter 3 deals with the methodology employed in carrying out the study and Chapter 4 deals with the presentation of results and analysis. Finally, Chapter 5 presents the conclusions and recommendations of the study.

CHAPTER 2

LITERATURE REVIEW

The history of studies on human face perception and machine recognition of faces are given in this chapter. This chapter will focus on studies carried out by researchers on face recognition and Principal Component Analysis, in other areas.

2.0 INTRODUCTION

Face Recognition has been an interesting issue for both neuroscientists and computer engineers dealing with artificial intelligence (AI). A healthy human can detect a face easily and identify that face, whereas for a computer to recognize faces, the face area should be detected and recognition comes next. Hence, for a computer to recognize faces the photographs should be taken in a controlled environment; a uniform background and identical poses makes the problem easy to solve.

2.1 RECENT APPROACHES TO FACE RECOGNITION

Face recognition has been an active research area over last 30 years. This research spans several disciplines such as image processing, pattern recognition, computer vision, and neural networks. It has been studied by scientists from different areas of psychophysical sciences and those from different areas of computer sciences. Psychologists and neuroscientists mainly deal with the human perception part of the topic, whereas engineers studying on machine recognition of human faces deal with the computational aspects of face recognition. Face recognition has applications mainly in the fields of biometrics, access control, law enforcement, and security and surveillance systems.

2.2 HUMAN FACE RECOGNITION

When building artificial face recognition systems, scientists try to understand the architecture of human face recognition system. Focusing on the methodology of human face recognition system may be useful to understand the basic system. However, the human face recognition system utilizes more than that of the machine recognition system which is just 2-D data. The human face recognition system uses some data obtained from some or all of the senses; visual, auditory, tactile, etc. All these data is used either individually or collectively for storage and remembering of faces. In many cases, the surroundings also play an important role in human face recognition system. It is hard for a machine recognition system to handle so much data and their combinations. However, it is also hard for a human to remember many faces due to storage limitations. For a human face recognition system the important feature is its parallel processing capacity.

The issue “which features humans use for face recognition” has been studied and it has been argued that both global and local features are used for face recognition. It is harder for humans to recognize faces which they consider as neither “attractive” nor “unattractive”.

Both holistic and feature information are important for the human face recognition system. Studies suggest the possibility of global descriptions serving as a front end for better feature-based perception. Chellappa et al (1995). If there are dominant features present such as big ears, a small nose, etc. holistic descriptions may not be used. Also, recent studies show that an inverted face (i.e. all the intensity values are subtracted from 255 to obtain the inverse image in the grey scale) is much harder to recognize than a normal face.

Hair, eyes, mouth, face outline have been determined to be more important than nose for perceiving and remembering faces. It has also been found that the upper part of the face is more useful than the lower part of the face for recognition. Also, aesthetic attributes (e.g. beauty, attractiveness, pleasantness, etc.) play an important role in face recognition; the more attractive the faces are easily remembered.

For humans, photographic negatives of faces are difficult to recognize. But, there is not much study on why it is difficult to recognize negative images of human faces.

Bruce (1999) conducted a study on the direction of illumination and showed the importance of top lighting. He demonstrated that it is easier for humans to recognize faces illuminated from top to bottom than the faces illuminated from bottom to top.

2.3 MACHINE RECOGNITION OF FACES

Although studies on human face recognition were expected to be a reference on machine recognition of faces, research on machine recognition of faces has developed independent of studies on human face recognition.

Chellappa et al. (1995) posited that, during 1970's, typical pattern classification techniques, which use measurements between features in faces or face profiles, were used. During the 1980's, work on face recognition remained nearly stable. Since the early 1990's, research interest on machine recognition of faces has grown tremendously. The reasons may be;

- An increase in emphasis on civilian/commercial research projects,
- The studies on neural network classifiers with emphasis on real-time computation and adaptation,

- The availability of real time hardware,
- The growing need for surveillance applications.

The basic question relevant for face classification is that; what form the structural code (for encoding the face) should take to achieve face recognition. Two major approaches are used for machine identification of human faces; geometrical local feature based methods, and holistic template matching based systems. Also, combinations of these two methods, namely hybrid methods, are used. The first approach, the geometrical local feature based one, extracts and measures discrete local features (such as eye, nose, mouth, hair, etc.) for retrieving and identifying faces. One of the well-known geometrical-local feature based methods is the Elastic Bunch Graph Matching (EBGM) technique. The other approach, the holistic one, conceptually related to template matching, attempts to identify faces using global representations; Huang (1998). Holistic methods approach the face image as a whole and try to extract features from the whole face region. In this approach, as in the previous approach, the pattern classifiers are applied to classify the image after extracting the features. One of the methods to extract features in a holistic system is applying statistical methods such as Principal Component Analysis (PCA) to the whole image. PCA can also be applied to a face image locally; in that case the approach is not holistic.

With the problem being a classification problem, training the face recognition system with images from the known individuals and classifying the newly coming test images into one of the classes is the main aspect of the face recognition systems. This problem seems to be easily solved by humans where limited memory can be the main problem; whereas the problems for a machine face recognition system are:

1. Facial expression change

2. Illumination change
3. Aging
4. Pose change
5. Scaling factor (i.e. size of the image)
6. Frontal vs. profile
7. Presence and absence of spectacles, beard, moustache etc.
8. Occlusion due to scarf, mask or obstacles in front.

Appropriate methods should be applied to reduce the dimension of the studied space. Working on higher dimension causes overfitting where the system starts to memorize. Also, computational complexity would be an **important problem** when working on large databases.

Kanade's (1973) face recognition system was the first and is still one of the few systems in which all the steps in the recognition process were automated, using a top-down control strategy directed by a generic model of expected feature characteristics. His system calculated a set of facial parameters from a single face image and used a pattern classification technique to match the face from a known set, a purely statistical approach depending primarily on local histogram analysis and absolute gray-scale values.

2.3.1 STATISTICAL APPROACHES

Statistical methods include template matching based systems where the training and test images are matched by measuring the correlation between them. Moreover, statistical methods include the projection based methods such as Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), etc. In fact, projection based systems came out due to

the shortcomings of the straightforward template matching based approaches; that is, trying to carry out the required classification task in a space of extremely high dimensionality.

2.3.1.1 TEMPLATE MATCHING

Huang (1998) explained that template matching is conceptually related to holistic approach which attempts to identify faces using global representations. These types of methods approach the face image as a whole and try to extract features from the whole face region and then classify the image by applying a pattern classifier. One of the methods used to extract features in a holistic system, is based on statistical approaches which are discussed in the following section.

A simple version of template matching is that a test image represented as a two-dimensional array of intensity values is compared using a suitable metric, such as the Euclidean distance, with a single template representing the whole face. There are several other more sophisticated versions of template matching on face recognition. One can use more than one face template from different viewpoints to represent an individual's face.

Brunelli and Poggio (1993) suggested that the optimal strategy for face recognition is holistic and corresponds to template matching. In their study, they compared a geometric feature based technique with a template matching based system. In the simplest form of template matching, the image (as 2-D intensity values) is compared with a single template representing the whole face using a distance metric.

2.3.1.2 FACE DETECTION AND RECOGNITION BY PCA

In Karl Pearson (1901), PCA is a standard statistical technique that can be used to reduce the dimensionality of a data set, and is useful when obtained data have some redundancy. This result in reduction of variables into smaller number of variables called Principal Components which will account for most of the variance in the observed variable.

Sirovich and Kirby (1987) were the first to utilize Principal Components Analysis (PCA) to economically represent face images. They demonstrated that any particular face can be efficiently represented along the standard face picture (eigenpicture) coordinate space, and that any face can be approximately reconstructed by using just a small collection of eigenpictures and the corresponding projections along each eigenpicture.

Turk and Pentland (1991) discovered that while using the eigenfaces technique, the residual error could be used to detect faces in images, a discovery that enabled real-time automated face recognition systems. Although the approach was somewhat constrained by environmental factors, it nonetheless created significant interest in furthering development of automated face recognition technologies.

Starnier et al (1994) posited that, an important feature of PCA is that one can reconstruct any original image from the training set by combining the eigenfaces. They stipulated that, by means of PCA one can transform each original image of the training set into a corresponding eigen face. The original face image can be reconstructed from eigenfaces if all the eigenfaces (features) are added in the right proportion. Each eigenface represents only certain features of the face, which may or may not be present in the original image.

Samal and Iyengar (1995) proposed a PCA face detection scheme based on face silhouettes. Instead of eigenfaces, they generated eigensilhouettes and combined this with standard image processing techniques (edge detection thinning, thresholding) and the generalized Hough transform. They reported a detection rate of 92% on a dataset of 129 images (63 face images and 66 general images) where in the case of the face images, the face occupies most of the image.

Belhumeur et al. (1997) argued that although PCA appears to work well when a single image of each individual is available, but when multiple images per person are present, by choosing the projection which maximizes total scatter, PCA retains unwanted variations due to lighting and facial expression.

Kanade (1973), employed simple image processing methods to extract a vector of 16 facial parameters - which were ratios of distances, areas and angles (to compensate for the varying size of the pictures) - and used a simple Euclidean distance measure for matching to achieve a peak performance of 75% on a database of 20 different people using 2 images per person (one for reference and one for testing).

Cox et al. (1996) reported a recognition performance of 95% on a database of 685 images (a single image for each individual) using a 30-dimensional feature vector derived from 35 facial features. However, the facial features were manually extracted, so it is reasonable to assume that the recognition performance would have been much lower if an automated, and hence less precise, feature extraction method had been adopted.

Moghaddam and Pentland (1995) used the Eigenface Method for image coding of human faces for potential applications such as video telephony, database image compression and face recognition.

Lee et al. (1999) proposed a method using PCA which detects the head of an individual in a complex background and then recognize the person by comparing the characteristics of the face to those of known individuals.

Crowley and Schwerdt (1999), in a study suggested that, PCA is used for coding and compression for video streams of talking heads. They suggest that a typical video sequence of a talking head can often be coded in less than 16 dimensions.

Moghaddam et al. (1999), (1998), and (2000), suggested the Bayesian PCA method. By this system, the eigenface method based on simple subspace-restricted norms is extended to use a probabilistic measure of similarity. Also, another difference from the standard Eigenface approach is that this method uses the image differences in the training and test stages. The difference of each image belonging to the same individual with each other is fed into the system as intrapersonal difference, and the difference of one image with an image from different class is fed into the system as extra personal difference. Finally, when a test image comes, it is subtracted from each image in the database and each difference is fed into the system. For the biggest similarity (i.e. smallest difference) with one of the training images, the test image is decided to be in that class. The mathematical theory is mainly studied in Moghaddam and Pentland (1997).

Moghaddam (2002) introduced his study on several techniques; Principal Component Analysis (PCA), Independent Component Analysis (ICA), and nonlinear Kernel PCA (KPCA). He examined and tested these systems using the FERET database. He argued that the experimental results demonstrate the simplicity, computational economy and performance superiority of the Bayesian PCA method over other methods.

Chung et al. (1999) suggested the use of PCA and Gabor Filters together. Their method consists of two parts: In the first part, Gabor Filters are used to extract facial features from the original image on predefined fiducial points. In the second part, PCA is used to classify the facial features optimally. They suggested the use of combining these two methods in order to overcome the shortcomings of PCA. They argued that, when raw images are used as a matrix of PCA, the eigenspace cannot reflect the correlation of facial feature well, as original face images have deformation due to in-plane, in-depth rotation and illumination and contrast variation. Also they argued that, they have overcome these problems using Gabor Filters in extracting facial features.

2.3.1.3 FEATURE-BASED APPROACHES

Bobis et al. (1999) studied on a feature based face recognition system. They suggested that a face can be recognized by extracting the relative position and other parameters of distinctive features such as eyes, mouth, nose and chin. The system described the overall geometrical configuration of face features by a vector of numerical data representing position and size of main facial features. First, they extracted eyes coordinates. The interocular distance and eyes position is used to determine size and position of the areas of search for face features. In these areas binary thresholding is performed, system modifies threshold automatically to detect

features. In order to find their coordinates, discontinuities are searched for in the binary image. They claimed that, their experimental results showed that their method is robust, valid for numerous kind of facial image in real scene, works in real time with low hardware requirements and the whole process is conducted automatically.

Cagnoni and Poggi (1999) suggested a feature-based approach instead of a holistic approach to face recognition. They applied the eigenface method to sub-images (eye, nose, and mouth). They also applied a rotation correction to the faces in order to obtain better results.

Guan and Szu (1999) compared the performance of PCA and ICA on face images. They argue that, ICA encodes face images with statistically independent variables, which are not necessarily associated with the orthogonal axes, while PCA is always associated with orthogonal eigenvectors.

Duda et al. (2001) suggested that while PCA seeks directions in feature space that best represent the data in a sum-squared error sense, ICA seeks directions that are most independent from each other.

Martinez (1999) proposed a different approach based on identifying frontal faces. Their approach divides a face image into n different regions, analyzes each region with PCA, and then uses a Bayesian approach to find the best possible global match between a probe and database image. The relationship between the n parts is modeled by using Hidden Markov Models (HMMs).

2.3.1.4 GEOMETRY FEATURE BASED METHODS

Geometry feature based methods uses the facial feature measures such as distance between eyes, ratio of distance between eyes and nose etc., but it is significantly different from the feature-based techniques that it constructs the topological graph using the facial features of each subject.

Lades et al. (1997) presented dynamic link architecture for distortion invariant object recognition which employs elastic graph matching to find the closed stored graph. Objects were represented with sparse graphs whose vertices were labeled with geometrical distances. In this system, individual faces were represented by a rectangular graph, each node labeled with a set of complex Gabor wavelet coefficients, called a jet. Only the magnitudes of the coefficients were used for matching and recognition. When recognizing a face of a new image, each graph in the model gallery was matched to the image separately and the best match indicated the recognized person. They presented good results with a database of 87 subjects and test images composed of different expressions and faces turned 15 degree. The matching process was computationally expensive, taking roughly 25 seconds to compare an image with 87 stored objects when using a parallel machine with 23 transputers.

Wiskott et al. (1997) extended this system to handle larger galleries and larger variations in pose and to increase the matching accuracy. Firstly, they use the phase of the complex Gabor wavelet coefficients to achieve an accurate location of the nodes and to disambiguate patterns which would be similar in the magnitudes of the coefficient. Secondly, they employ object adapted graphs, so that nodes refer to specific facial landmarks, called fiducial points.

The correspondences between two faces can be found across large viewpoint changes. Thirdly, a new data structure called the bunch graph was introduced which serves as

generalized representation of faces by combining jets of a small set of individual faces. This allows the system to find the fiducial points in one matching process, which eliminates the need for matching each model graph individually. This also reduces computational effort significantly. It offers good performance of about 98% for FERET database. But the drawback in this feature matching approach is that it requires manual intervention to select the fiducial points in the facial image and it requires precise location of those points.

Cagnoni and Poggi (1999) suggested a feature based approach instead of a holistic approach to face recognition. They applied the eigenface method to sub-images (eye, nose and mouth). They also applied a rotation correction to the faces in order to obtain better results.

Fischler and Elschlager (1973) attempted to measure similar faces automatically. They described a linear embedding algorithm that used both local feature template matching and a global measure of fit to find and measure facial features. This template matching has been continued and improved.

Yuille et al (1989) continued and improved the work on template matching. Their strategy is based on ‘‘deformable templates’’, which are parametrized models of the face and its features in which the parameter values are determined by interactions with the image.

Choudhury et al. (1999) described a multimodal person recognition system. This system consists of a face recognition module, a speaker identification module, and a classifier fusion module. It has the following characteristics:

- (1) The face recognition module can detect and compensate for pose variations; the speaker identification module can detect and compensate for changes in the auditory background;

- (2) the most reliable video frames and audio clips are selected for recognition;
- (3) 3D information about the head obtained through SfM is used to detect the presence of an actual person as opposed to an image of that person.

In Li and Chellappa (2001), a face verification system based on tracking facial features was presented. The basic idea of this approach is to exploit the temporal information available in a video sequence to improve face recognition. First, the feature points defined by Gabor attributes on a regular 2D grid are tracked. Then, the trajectories of these tracked feature points are exploited to identify the person presented in a short video sequence.

2.3.1.5 OTHER STATISTICAL APPROACHES

Cevikalp et al. (2005) solved small sample size problem of LDA by optimizing a variant of Fisher's criterion. It searched for the optimal projection vectors in the null space of within class. So, to find the optimal projection vectors in the null space of it projects the face samples onto the null space of to generate common vectors for each class and then obtain the projection vectors by performing PCA on common vectors. A new set of vectors, called as discriminative common vectors, are obtained by projecting face samples on the projection vectors. Thus each class is represented by a single discriminative common vector. Among two algorithms to extract the discriminant common vectors for representing each person in the training set of face database, one algorithm uses within-class scatter matrix of the samples in the training set while the other uses the subspace methods and the Gram-Schmidt orthogonalization procedures to obtain the discriminative common vectors. These discriminative common vectors are used for classification of new faces.

In Zhang et al (2004) the 2D PCA is based on 2D image matrices instead of 1D vector. The image covariance matrix is constructed directly from the original image. Let A denote the image with m rows and n columns. The image matrix is projected on to n -dimension column vector x as, $y = Ax$ where y is an m dimensional column vector called the projected feature vector of the image A . The column vector x is chosen such that the generalized total scatter criterion is maximized. This means that total scatter of the projected vectors is maximized.

Zhang et al (2004) showed that the scatter criterion is satisfied when x is chosen as the orthonormal eigenvectors of the image covariance matrix G_t . For each eigenvector x_s , there exists a projected feature vector y_s . Therefore, the principal components of a 2D-PCA is a vector unlike that of PCA which is a scalar. If S principal components are used (corresponding to the largest S eigenvalues of G_t), the principal component vectors obtained can be represented as, $B = [y_1, y_2, \dots, y_S]$. Thus, B is the feature matrix of the image sample A . Unlike the conventional PCA, the 2D-PCA does not involve computation of a large correlation matrix and therefore is relatively less computation intensive. But on the other hand, it requires more memory for storing feature matrices.

Yang, Zhang et al. (2004) tested 2D-PCA method on ORL, AR and Yale face databases. For the ORL database, the authors used two strategies for experiments:

(a) 5 image samples per class for training and (b) leave one out strategy for observing the average performance. In case (a) the recognition rate is 96% and in case of (b) the same is reported to be 98.3% for ORL. Leave one out strategy was adopted for Yale database and maximum accuracy is reported to be 84.24%.

Li and Yuan (2005), in the recently proposed 2D-LDA showed the image is not reordered as a column vector. The process of projection of an image using 2D-LDA is given as follows. An image A of m rows and n columns is projected as $y = Ax$, where y is the projected vector and x is the projection vector. Optimal projection vectors are chosen when Fisher's criterion is maximized. The classification using nearest neighbor strategy is similar to the classification using 2D-PCA.

Appearance-based approaches for face recognition like PCA, LDA, and probabilistic subspace view a 2D face image as a vector in image space. A set of face images $\{x_i\}$ can be represented as an $M \times N$ matrix $X = [x_1, x_2, \dots, x_N]$, where M is total number of pixels in the images and N is the total number of samples. Each of the face images x_i belongs to one of the C classes 1, 2, 3, ..., C .

Hong and Jain (1998) showed that there is substantial related work in multimodal biometrics. They used face and fingerprint in multimodal biometric identification. Others have used face and voice. However, use of the face and ear in combination seems more relevant to surveillance applications.

Turk and Pentland (1991) developed the traditional PCA algorithm used in the eigenfaces approach. This method computes the eigenvectors and eigenvalues of a covariance matrix and solves an eigenvalues system problem. This method performs no discrimination but instead the eigenvectors that correspond to the largest eigenvalues are used in as base vectors in a low-dimensional space.

Etemad and Chellappa (1996) proposed a method on appliance of Linear/Fisher Discriminant Analysis for the face recognition process. LDA is carried out via scatter matrix analysis. The aim is to find the optimal projection which maximizes between class scatter of the face data and minimizes within class scatter of the face data. As in the case of PCA, where the eigenfaces are calculated by the eigenvalue analysis, the projections of LDA are calculated by the generalized eigenvalue equation.

Belhumeur et al. (1997) studied a method which combines PCA and LDA. This method consists of two steps; the face image is projected into the eigenface space which is constructed by PCA, and then the eigenface space projected vectors are projected into the LDA classification space to construct a linear classifier. In this method, the choice of the number of eigenfaces used for the first step is critical; the choice enables the system to generate class separable features via LDA from the eigenface space representation. The generalization/overfitting problem can be solved in this manner. In these studies, a weighted distance metric guided by the LDA eigenvalues was also employed to improve the performance of the subspace LDA method.

Podilchuk and Zang (1996) proposed a method which finds the feature vectors using Discrete Cosine Transform (DCT). Their system tries to detect the critical areas of the face. The system is based on matching the image to a map of invariant facial attributes associated with specific areas of the face. They claim that this technique is quite robust, since it relies on global operations over a whole region of the face. A codebook of feature vectors or codewords is determined for each person from the training set. They examine recognition performance based on feature selection, number of features or codebook size, and feature dimensionality.

Duda et al. (2001) tried several block-based transformations for feature selection and the K-means algorithm to generate the code words for each codebook. They argue that the block-based DCT coefficients produce good low-dimensional feature vectors with high recognition performance. This brings the possibility of performing face recognition directly on a DCT-based compressed bit stream without having to decode the image.

Eickeler et al. (1999) suggested a system based on Pseudo 2-D Hidden Markov Models (HMMs) and coefficient of the 2-D DCT as the features. A major advantage of their approach is that the system works directly on JPEG-compressed face images, i.e. it uses the DCT-coefficients provided by the JPEG standard. Thus, it does not need any further decompressing of the image.

Phillips (1999) applied SVM to face recognition. Face recognition is a K-class problem, where K is the number of known individuals; and SVM is a binary classification method. By reformulating the face recognition problem and reinterpreting the output of the SVM classifier, they developed a SVM-based face recognition algorithm. They formulated the face recognition problem in difference space, which models dissimilarities between two facial images. In difference space, they formulated the face recognition as a two class problem. The classes are; dissimilarities between faces of the same person and dissimilarities between faces of different people. By modifying the interpretation of the decision surface generated by SVM, they generated a similarity metric between faces, learned from examples of differences between faces.

2.3.1.6 NEURAL NETWORK BASED APPROACHES

Artificial Neural Network (ANN) Bishop (1995); Mammone (1993), is a powerful tool for pattern recognition problems. The use of neural networks (NN) in faces has addressed several problems: gender classification, face recognition and classification of facial expressions.

Kohonen (1988) outlined one of the earliest demonstrations of NN for face recall applications which was reported in Kohonen's associative map. Using a small set of face images, accurate recall was reported even when input image is very noisy or when portions of the images are missing. A few NN based face recognition techniques are discussed in the following.

Stonham (1984) reported a single layer adaptive NN (one for each person) for face recognition, expression analysis and face verification. A system named Wilke, Aleksander and Stonham's recognition devise (WISARD) was devised. It needs typically 200-400 presentations for training each classifier where the training patterns included translation and variation in facial expressions. One classifier was constructed corresponding to one subject in the database. Classification was achieved by determining the classifier that was giving the highest response for the given input image.

Much of the present literature on face recognition with neural networks presents results with only a small number of classes (often below 20).

Lawrence et al. (1997) presented a hybrid neural network solution which combines local image sampling, a self-organizing map (SOM) and a convolutional neural network. The SOM provides a quantization of the image samples into a topological space are also nearby in the output space, thereby providing dimensionality reduction and invariance to minor changes in the image sample. The convolutional neural network provides partial invariance to translation, rotation, scale and deformation. The recognizer provides a measure of confidence

in its output. The classification error approaches zero when rejecting as few as 10% of the examples on a database of 400 images which contains a high degree of variability in expression, pose and facial details.

In Dai and Nakano (1998), a Hopfield memory model for the facial images is organized and the optimal procedure of learning is determined. A method for face recognition using Hopfield memory model combined with the pattern matching is proposed. It shows better performance of database having 20 faces of 40 subjects.

Lin et al. (1997) proposed a face recognition technique based on Probabilistic Decision based Neural network (PDBNN). It adopts a hierarchical network structures with nonlinear basis functions and competitive credit assignment scheme. It demonstrated a successful application of PDBNN on FERET and ORL databases.

Gutta et al. (2000) described the application of mixtures of experts on gender and ethnic classification of human faces and pose classification and showed their feasibility on the FERET database. The mixture consists of ensembles of radial basis functions (RBFs). Inductive Decision Trees (IDTs) and SVMs implement the "gating network" components for deciding which of the experts should be used to determine the classification output and to restrict the support of the input space. Experimental results yield good results on gender, ethnic and pose classification, which can be effectively used in face recognition.

2.3.2 HYBRID APPROACHES

This type of approach uses both statistical pattern recognition techniques and neural networks.

Examples for hybrid approaches include the combination of PCA and Radial Basis Function (RBF) neural network.

Er et al (1999) suggested the use of RBF on the data extracted by discriminant eigenfeatures. They used a hybrid learning algorithm to decrease the dimension of the search space in the gradient method, which is crucial on optimization of high dimension problem. First, they tried to extract the face features by both the PCA and LDA methods. Next, they presented a hybrid learning algorithm to train the RBF Neural Networks so the dimension of the search space is significantly decreased in the gradient method.

Thomaz et al (1998) also studied on combining PCA and RBF neural network. Their system for face recognition consists of a PCA stage which inputs the projections of a face image over the principal components into a RBF network acting as a classifier.

2.3.3 OTHER APPROACHES

Chellappa et al. (1995) used range images in face recognition. In this method data is obtained by scanning the individual with a laser scanner system. This system also has the depth information so the system processes 3-dimensional data to classify face images.

Yoshitomi et al. (1997) used thermal sensors to detect temperature distribution of a face. In this method, the front view face in input image is normalized in terms of location and size, followed by measuring the temperature distribution, the locally averaged temperature and the

shape factors of face. The measured temperature distribution and the locally averaged temperature are separately used as input data to feed a neural network and supervised classification is used to identify the face. The disadvantage of visible ray image analysis is that the performance is strongly influenced by lighting condition including variation of shadow, reflection and darkness. These can be overcome by this method which uses infrared rays.

Liposcak and Loncaric (1999) worked on profile images instead of frontal images. Their method is based on the representation of the original and morphological derived profile images. Their aim was to use the profile outline that bounds the face and the hair. They take a gray-level profile image and threshold it to produce a binary image representing the face region. They normalized the area and orientation of this shape using dilation and erosion. Then, they simulate hair growth and haircut and produce two new profile silhouettes. From these three profile shapes they obtain the feature vectors. After normalizing the vector components, they use the Euclidean distance measure for measuring the similarity of the feature vectors derived from different profiles.

Cagnoni and Poggi (1999) suggested a feature based approach instead of a holistic approach to face recognition. They applied the eigenface method to sub-images (eye, nose and mouth). They also applied a rotation correction to the faces in order to obtain better results.

Fischler and Elschlager (1973) attempted to measure similar faces automatically. They described a linear embedding algorithm that used both local feature template matching and a global measure of fit to find and measure facial features. This template matching has been continued and improved.

Li and Chellappa (2001) presented, a face verification system based on tracking facial features. The basic idea of this approach is to exploit the temporal information available in a video sequence to improve face recognition. First, the feature points defined by Gabor attributes on a regular 2D grid are tracked. Then, the trajectories of these tracked feature points are exploited to identify the person presented in a short video sequence.

Brunelli and Poggio (1993) computed a vector of 35 geometric features from a database of 47 people (4 images per person) and reported a 90% recognition rate. However, they also reported 100% recognition accuracy for the same database using a simple template-matching approach. Nevertheless, many of the findings have important consequences for engineers who design algorithms and systems for machine recognition of human faces.

During the past 5 to 8 years, much research has been concentrated on video based face recognition. The still image problem has several inherent advantages and disadvantages. For applications such as drivers' licenses, due to the controlled nature of the image acquisition process, the segmentation problem is rather easy. However, if only a static picture of an airport scene is available, automatic location and segmentation of a face could pose serious challenges to any segmentation algorithm. On the other hand, if a video sequence is available, segmentation of a moving person can be more easily accomplished using motion as a cue. But the small size and low image quality of faces captured from video can significantly increase the difficulty in recognition. Recognizing a 3D object from its 2D images poses many challenges. The illumination and pose problems are two prominent issues for appearance- or image-based approaches.

CHAPTER 3

METHODOLOGY

This chapter is dedicated to the explanation of a principal component analysis method that is used in face recognition, on which the proposed face recognition system is based.

3.0 INTRODUCTION

Much of the previous work on automated face recognition has ignored the issue of just what aspects of the face stimulus are important for face recognition. This suggests the use of an information theory approach of coding and decoding of face images, emphasizing the significant local and global features. Such features may or may not be directly related to our intuitive notion of face features such as the eyes, nose, lips, and hair.

In the language of information theory, the relevant information in a face image is extracted, encoded as efficiently as possible, and then compared with a database of models encoded similarly. A simple approach to extracting the information contained in an image of a face is to somehow capture the variation in a collection of face images, independent of any judgment of features, and use this information to encode and compare individual face images.

In mathematical terms, the principal components of the distribution of faces, or the eigenvectors of the covariance matrix of the set of face images, treating an image as point (or vector) in a very high dimensional space is sought.

The eigenvectors are ordered, each one accounting for a different amount of the variation among the face images.

These eigenvectors can be thought of as a set of features that together characterize the variation between face images. Each image location contributes more or less to each

eigenvector, so that it is possible to display these eigenvectors as a sort of ghostly face image which is called an "eigenface".

Each individual face can be represented exactly in terms of a linear combination of the eigenfaces. Each face can also be approximated using only the "best" eigenfaces, those that have the largest eigenvalues, and which therefore account for the most variance within the set of face images. The best M eigenfaces span an M-dimensional subspace which we call the "face space" of all possible images.

Kirby and Sirovich (1987, 1990) developed a technique for efficiently representing pictures of faces using principal component analysis. Starting with an ensemble of original face images, they calculated a best coordinate system for image compression, where each coordinate is actually an image that they termed an "eigenpicture". They argued that, at least in principle, any collection of face images can be approximately reconstructed by storing a small collection of weights for each face, and a small set of standard pictures (the eigenpictures). The weights describing each face are found by projecting the face image onto each eigenpicture.

In this thesis, the method which was proposed by Turk and Pentland (1991) has been followed in order to develop a face recognition system based on the eigenfaces approach. They argued that, if a multitude of face images can be reconstructed by weighted sum of a small collection of characteristic features or eigenpictures, perhaps an efficient way to learn and recognize faces would be to build up the characteristic features by experience over time and recognize particular faces by comparing the feature weights needed to approximately reconstruct them with the weights associated with known individuals. Therefore, each individual is characterized by a small set of feature or eigenpicture weights needed to

describe and reconstruct them. This is an extremely compact representation when compared with the images themselves.

3.1 PRINCIPAL COMPONENT ANALYSIS

Principal Component Analysis (PCA) is a dimensionality reduction technique based on extracting the desired number of principal components of the multi-dimensional data. The purpose of PCA is to reduce the large dimensionality of the data space (observed variables) to smaller intrinsic dimensionality of feature space (independent variables), which are needed to describe the data economically. This is the case when there is a strong correlation between observed variables. The first principal component is the linear combination of the original dimensions that has the maximum variance; the n -th principal component is the linear combination with the highest variance, subject to being orthogonal to the $n-1$ first principal components.

An important and largely unsolved problem in dimensionality reduction is the choice of the intrinsic dimensionality of the principal manifold. No analytical derivation of this number for a complex natural visual signal is available to date. To simplify this problem, it is common to assume that in the noisy embedding of the signal of interest (in our case, a point sampled from the facespace) in a high-dimensional space, the signal-to-noise ratio is high. Statistically, that means that the variance of the data along the principal modes of the manifold is high compared to the variance within the complementary space.

This assumption relates to the eigenspectrum - the set of the eigenvalues of the data covariance matrix. The i -th eigenvalue is equal to the variance along the i -th principal component; thus, a reasonable algorithm for detecting k is to search for the location along the decreasing eigenspectrum where the value of λ drops significantly.

3.1.1 DERIVING PCA: EIGENVECTORS OF COVARIANCE MATRIX

The first algebraic solution to PCA could be derived using linear algebra. This solution is based on an important property of eigenvector decomposition. The data set is X , an $m \times n$ matrix, where m is the number of measurement types and n is the number of data trials. The

goal is to find some orthonormal matrix P where $Y=PX$ such that $S_Y \equiv \frac{1}{n-1} YY^T$ is

diagonalized. The rows of P are the principal components of X .

Begin by rewriting S_Y in terms of our variable of choice P .

$$S_Y = \frac{1}{n-1} YY^T \quad (3.1)$$

$$= \frac{1}{n-1} (PX)(PX)^T \quad (3.2)$$

$$= \frac{1}{n-1} PXX^T P^T \quad (3.3)$$

$$= \frac{1}{n-1} P (XX^T) P^T \quad (3.4)$$

$$S_Y = \frac{1}{n-1} PAP^T. \quad (3.5)$$

Note that a new matrix $A=XX^T$, is defined, where A is symmetric. The roadmap is to recognize that a symmetric matrix (A) is diagonalized by an orthogonal matrix of its eigenvectors.

A symmetric matrix A can be written as EDE^T , where D is a diagonal matrix and E is a matrix of eigenvectors of A arranged as columns.

The matrix A has $r \leq m$ orthonormal eigenvectors where r is the rank of the matrix. We select the matrix P to be a matrix where each Row p_i is an eigenvector of XX^T

By this selection, $P = E^T$. Hence $A = P^TDP$.

With this relation and the fact that $P^{-1} = P^T$ since the inverse of orthonormal matrix is its transpose, we can finish evaluating S_Y as follows;

$$\begin{aligned} S_Y &= \frac{1}{n-1} P A P^T \\ &= \frac{1}{n-1} P (P^T D P) P^T \end{aligned} \quad (3.6)$$

$$= \frac{1}{n-1} (P P^T) D (P P^T) \quad (3.7)$$

$$= \frac{1}{n-1} (P P^{-1}) D (P P^{-1}) \quad (3.8)$$

$$S_Y = D \quad (3.9)$$

It is evident that the choice of P diagonalizes S_Y .

3.2 EIGEN FACE APPROACH

An image space can be thought of as a space having dimensions equal to the number of pixels making up the image and having values in the range of the pixels values. Thus, for example for a grey scale image of size $(N_x \times N_y)$, the dimension of the image space is P , P being N_x times N_y .

An image can be thought as a point in the image space by converting the image to a long vector by concatenating each column of the image one after the other.

When all the face images are converted into vectors, they will group at a certain location in the image space as they have similar structure, having eye, nose and mouth in common and their relative position correlated. This correlation is the main point to start the eigenface analysis.

The eigenface method tries to find a lower dimensional space for the representation of the face images by eliminating the variance due to non-face images; that is, it tries to focus on the variation just coming out of the variation between the face images.

Eigenface method is the implementation of Principal Component Analysis (PCA) over images. In this method, the features of the studied images are obtained by looking for the maximum deviation of each image from the mean image. This variance is obtained by getting the eigenvectors of the covariance matrix of all the images.

The eigenface space is obtained by applying the eigenface method to the training images. Later, the training images are projected into the eigenface space. Next, the test image is projected into this new space and the distance of the projected test image to the training images is used to classify the test image. In the standard eigenface procedure suggested by Turk and Pentland (1991), Nearest Mean Classifier is used for the classification of test images. It is an adequate and efficient method to be used in face recognition due to its simplicity, speed and learning capability. Eigenfaces are a set of eigenvectors used in the computer vision problem of human face recognition. They refer to an appearance based approach to face recognition that seeks to capture the variation in a collection of face images and use this information to encode and compare images of individual faces in a holistic manner.

The eigenfaces are principal components of a distribution of faces, or equivalently, the eigenvectors of the covariance matrix of the set of the face images, where an image with N by N pixels is considered a point in N^2 dimensional space. This suggests that coding and decoding of face images may give information of face images emphasizing the significance of features. These features may or may not be related to facial features such as eyes, nose, lips and hairs. Relevant information in a face image needs to be extracted, efficiently encoded and compared with a database of faces encoded similarly. A simple approach to extracting

the information content in an image of a face is to somehow capture the variation in a collection of face images. Principal components of the distribution of faces, or the eigenvectors of the covariance matrix of the set of face images should be found. Each image location contributes to each eigenvector, so that the eigenvector as a sort of face can be displayed. Each face image can be represented exactly in terms of linear combination of the eigenfaces. The number of possible eigenfaces is equal to the number of face image in the training set. The faces can also be approximated by using best eigenface, those that have the largest eigenvalues, and which therefore account for most variance between the set of face images. The primary reason for using fewer eigenfaces is computational efficiency.

The whole recognition process involves two steps:

- a. Initialization process
- b. Recognition process

The Initialization process involves the following operations:

1. Acquire the initial set of face images called training set.
2. Calculate the eigenfaces from the training set, keeping only the highest eigenvalues. These M images define the face space. As new faces are experienced, the eigenfaces can be updated or recalculated.
3. Calculate the corresponding distribution in M -dimensional weight space for each known individual, by projecting their face images on to the “face space”.

These operations can be performed from time to time whenever there is a free excess operational capacity. This data can be cached which can be used in the further steps eliminating the overhead of re-initializing, decreasing execution time thereby increasing the performance of the entire system.

Having initialized the system, the next process involves the steps,

1. Calculate a set of weights based on the input image and the M eigenfaces by projecting the input image onto each of the eigenfaces
2. Determine if the image is a face at all (known or unknown) by checking to see if the image is sufficiently close to a “free space”.
3. If it is a face, then classify the weight pattern as either a known person or as unknown.
4. Update the eigenfaces or weights as either a known or unknown.

If the same unknown person face is seen several times then calculate the characteristic weight pattern and incorporate into known faces. The last step is not usually a requirement of every system and hence the steps are left optional and can be implemented as and when there is a requirement.

3.2.1 EIGENVECTORS AND EIGENVALUES

For a square matrix A , λ is an eigenvalue of A if there exists a non-zero vector x such that $Ax = \lambda x$. In this case, x is called an eigenvector corresponding to λ , and the pair (λ, x) is called an eigenpair for A . Eigenvalues and eigenvectors are also known as, respectively, *characteristic roots* and *characteristic vectors*, or *latent roots* and *latent vectors*. The dominant eigenvector of a matrix is an eigenvector corresponding to the eigenvalue of largest magnitude of that matrix.

3.2.2 GENERAL FORMULA FOR THE SOLUTION OF THE EIGENVALUE PROBLEM

One can find eigenvalues and eigenvectors by treating a matrix as a system of linear equations and solving for the values of the variables that make up the components of the

eigenvector. For a square matrix A and I the identity matrix, $\det(A - \lambda I) = 0$. (det means the determinant of matrix). This equation is called the characteristic equation.

$= (a_{1,1} - \lambda)(a_{2,2} - \lambda) \cdots (a_{n,n} - \lambda) = 0$, the characteristic equation then reads

$$\det(A - \lambda I) = \det \left(\begin{bmatrix} a_{1,1} & 0 & \cdots & 0 \\ 0 & a_{2,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & a_{n,n} \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix} \right) \quad (3.10)$$

$$= \det \begin{bmatrix} a_{1,1} - \lambda & 0 & \cdots & 0 \\ 0 & a_{2,2} - \lambda & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & a_{n,n} - \lambda \end{bmatrix} \quad (3.11)$$

$$= (a_{1,1} - \lambda)(a_{2,2} - \lambda) \cdots (a_{n,n} - \lambda) = 0 \quad (3.12)$$

The solutions to the characteristic equation are the eigenvalues $\lambda_i = a_{i,i} (i=1, \dots, n)$

For the eigenvectors x_i of A , they are the non-zero solutions of the equations $(A - \lambda I)x = 0$, for the values of λ .

3.2.2.1 WORKED EXAMPLE

For a 2x2 matrix A , $A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$

$$\det(A - \lambda I) = 0$$

$$\det(A - \lambda I) = 0 = \det \begin{bmatrix} 2 - \lambda & 1 \\ 1 & 2 - \lambda \end{bmatrix} = 0$$

Calculating the determinant, this yields the quadratic equation $\lambda^2 - 4\lambda + 3 = 0$

$\lambda_1 = 1$ and $\lambda_2 = 3$ are the eigenvalues for the matrix A .

$$(A - \lambda I)x = 0$$

For $\lambda_1 = 1$

$$\left(\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} - 1 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \left(\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$x_1 + x_2 = 0 \Rightarrow x_1 = -x_2 \text{ Therefore } x_1 = -1 \text{ and } x_2 = 1$$

$$x = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \text{ is the eigenvector of the eigenvalue } \lambda_1 = 1$$

For $\lambda_2 = 3$

$$\left(\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} - 3 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \left(\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} - \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$-x_1 + x_2 = 0 \Rightarrow x_1 = x_2 \text{ Therefore } x_1 = 1 \text{ and } x_2 = 1$$

$$x = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \text{ is the eigenvector of the eigenvalue } \lambda_2 = 3$$

Since $\lambda_2 = 3$ is the largest eigenvalue of the matrix A, the dominant eigenvector is $x = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

3.2.3 PROPERTIES OF EIGENVECTORS AND EIGENVALUES

- Eigenvectors can only be found for square matrices and not every square matrix has eigenvectors.
- Given an $m \times m$ matrix (with eigenvectors), we can find m eigenvectors.
- All eigenvectors of a symmetric matrix are perpendicular to each other, no matter how many dimensions we have.
- In practice eigenvectors are normalized to have unit length.

3.3 CALCULATING EIGENFACES

Let a face image $I(x, y)$ be a two-dimensional $N \times N$ array of 8-bit intensity values. An image may also be considered as a vector of dimension N^2 , so that a typical image of size 256×256 becomes a vector of dimension 65,536, or equivalently a point in 65,536-dimensional space. An ensemble of images, then, maps to a collection of points in this huge space.

Images of faces, being similar in overall configuration, will not be randomly distributed in this huge image space and thus can be described by a relatively low dimensional subspace. The main idea of the principal component analysis (or Karhunen-Loeve expansion) is to find the vectors that best account for the distribution of face images within the entire image space.

These vectors define the subspace of face images, which are called "face space". Each vector is of length N^2 , describes an $N \times N$ image, and is a linear combination of the original face images. Because these vectors are the eigenvectors of the covariance matrix corresponding to the original face images, and because they are face-like in appearance, they are referred to as "eigenfaces".

Definitions:

An $N \times N$ matrix A is said to have an eigenvector X , and corresponding eigenvalue λ if

$$AX = \lambda X. \quad (3.13)$$

It is obvious that, Eq. (3.13) can hold only if

$$\det|A - \lambda I| = 0 \quad (3.14)$$

which, if expanded out, is an N th degree polynomial in λ whose roots are the eigenvalues. This proves that there are always N (not necessarily distinct) eigenvalues. Equal eigenvalues coming from multiple roots are called "degenerate".

A matrix is called symmetric if it is equal to its transpose,

$$A = A^T \text{ or } a_{ij} = a_{ji} \quad (3.15)$$

It is termed orthogonal if its transpose equals its inverse,

$$A^T A = AA^T = I \quad (3.16)$$

A real matrix is called normal if it commutes with its transpose,

$$AA^T = A^T A. \quad (3.17)$$

In mathematical terms;

$$\text{Image } I : (N_x \times N_y) \text{ pixels} \quad (3.18)$$

The image matrix I of size $(N_x \times N_y)$ pixels is converted to the image vector Γ of size $(P \times 1)$ where $P = (N_x \times N_y)$; that is the image matrix is reconstructed by adding each column one after the other. Thus, this two dimensional vector is changed to one dimensional vector.

$$\text{Training Set } \Gamma = [\Gamma_1, \Gamma_2, \dots, \Gamma_{M_t}] \quad (3.19)$$

is the training set of image vectors and its size is $(P \times M_t)$ where M_t is the number of the training images.

$$\text{For example: } \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 2 \\ 1 \end{bmatrix}$$

Each face image is represented by the vector Γ_i

$$\Gamma_1 = \begin{bmatrix} 1 \\ -2 \\ 1 \\ -3 \end{bmatrix} \quad \Gamma_2 = \begin{bmatrix} 1 \\ 3 \\ -1 \\ 2 \end{bmatrix} \quad \Gamma_3 = \begin{bmatrix} 2 \\ 1 \\ -2 \\ 3 \end{bmatrix} \quad \dots \quad \Gamma_m = \begin{bmatrix} 1 \\ 2 \\ 2 \\ 1 \end{bmatrix}$$

Mean Face

$$\Psi = \frac{1}{M_t} \sum_{i=1}^{M_t} \Gamma_i \quad (3.20)$$

is the arithmetic average of the training image vectors at each pixel point. Its size is (P x 1).

$$\begin{bmatrix} 1 \\ -2 \\ 1 \\ -3 \end{bmatrix} + \begin{bmatrix} 1 \\ 3 \\ -1 \\ 2 \end{bmatrix} + \begin{bmatrix} 2 \\ 1 \\ -2 \\ 3 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 4 \\ 0 \\ 3 \end{bmatrix}$$

$$\Psi = (\Gamma_1 + \Gamma_2 + \Gamma_3 + \dots + \Gamma_M)/M$$

Mean subtracted image

$$\Phi = \Gamma - \Psi \quad (3.21)$$

is the difference of the training image from the mean image (size P x 1). Thus, each face differs from the average by $\Phi = \Gamma - \Psi$ which is also called mean centered image.

$$\Phi_1 = \begin{bmatrix} 2 \\ -1 \\ -1 \\ 0 \end{bmatrix} \quad \Phi_2 = \begin{bmatrix} 2 \\ 4 \\ -3 \\ 5 \end{bmatrix} \quad \Phi_3 = \begin{bmatrix} 3 \\ 2 \\ -4 \\ 6 \end{bmatrix} \quad \dots \quad \Phi_M = \begin{bmatrix} 2 \\ 3 \\ 0 \\ 4 \end{bmatrix}$$

$$\text{Difference Matrix } A = [\Phi_1, \Phi_2, \dots, \Phi_{M_t}] \quad (3.22)$$

is the matrix of all the mean subtracted training image vectors and its size is (P x M_t).

$$\text{Covariance Matrix } X = A \cdot A^T = \frac{1}{M_t} \sum_{i=1}^{M_t} \Phi_i \Phi_i^T \quad (3.23)$$

$A = [\Phi_1, \Phi_2, \dots, \Phi_{M_t}]$ is the covariance matrix of the training image vectors. Size is (P x P).

An important property of the Eigenface method is obtaining the eigenvectors of the covariance matrix. For a face image of size (N_x x N_y) pixels, the covariance matrix is of size (P x P), P being (N_x x N_y). This covariance matrix is very hard to work with due to its huge dimension causing computational complexity. On the other hand, Eigenface method

calculates the eigenvectors of the ($M_t \times M_t$) matrix, M_t being the number of face images, and obtains ($P \times P$) matrix using the eigenvectors of the ($M_t \times M_t$) matrix.

Initially, a matrix Y is defined as,

$$Y = A^T \cdot A = \frac{1}{M_t} \sum_{i=1}^{M_t} \Gamma_i^T \Gamma_i \quad (3.24)$$

which is of size ($M_t \times M_t$).

Then, the eigenvectors v_i and the eigenvalues μ_i of Y are obtained,

$$Y \cdot v_i = \mu_i \cdot v_i \quad (3.25)$$

The value of Y is put in this equation,

$$A^T \cdot A \cdot v_i = \mu_i \cdot v_i \quad (3.26)$$

Both sides are pre-multiplied by A ,

$$A \cdot A^T \cdot A \cdot v_i = A \cdot \mu_i \cdot v_i \quad (3.27)$$

The necessary matrix arrangements are made,

$$A \cdot A^T \cdot A \cdot v_i = \mu_i \cdot A \cdot v_i \quad (3.28)$$

(as μ_i is a scalar, this arrangement can be done)

$$X \cdot A \cdot v_i = \mu_i \cdot A \cdot v_i \quad (3.29)$$

Now, group $A \cdot v_i$ and call a variable $v_i = A \cdot v_i$. It is easy to see that

$$v_i = A \cdot v_i \quad (3.30)$$

is one of the eigenvectors of $X = A \cdot A^T$ and its size is $(P \times 1)$.

Thus, it is possible to obtain the eigenvectors of X by using the eigenvectors of Y . A matrix of size $(M_t \times M_t)$ is utilized instead of a matrix of size $(P \times P)$ (i.e. $[\{N_x \times N_y\} \times \{N_x \times N_y\}]$). This formulation brings substantial computational efficiency.

Fig. 3.1(a) and Fig. 3.1(b) below are facial images and the mean face respectively from the ORL database.

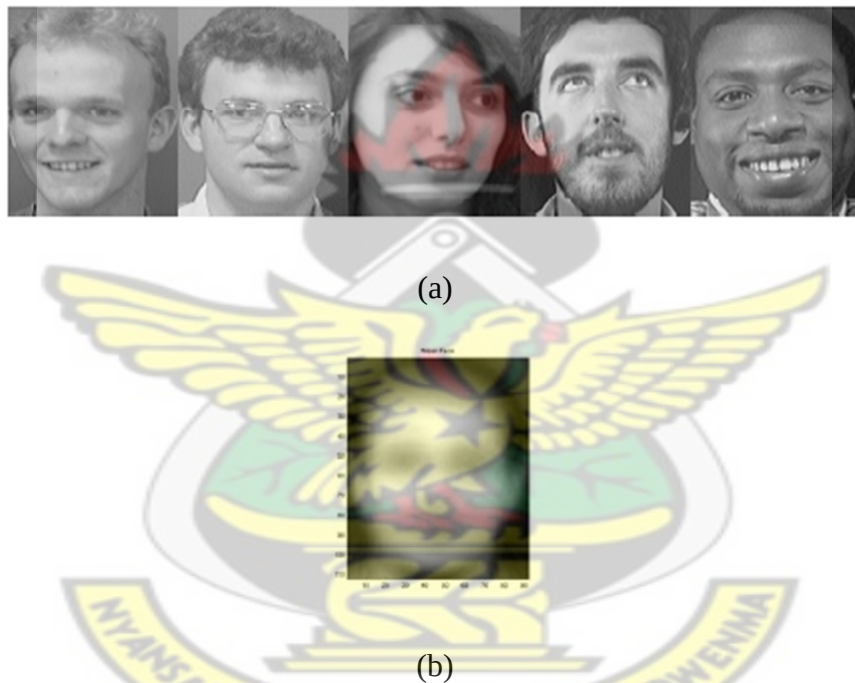


Fig 3.1 (a) Example of images from the ORL database,
(b) Mean face obtained from the ORL database.

The eigenfaces are in fact $(P \times 1)$ vectors for the computations; in order to see what they look like, they are rearranged as $(N_x \times N_y)$ matrices.

Instead of using M_t of the eigenfaces, $M' \leq M_t$ of the eigenfaces can be used for the eigenface projection. This is achieved to eliminate some of the eigenvectors with small eigenvalues, which contribute less variance in the data.

Eigenvectors can be considered as the vectors pointing in the direction of the maximum variance and the value of the variance the eigenvector represents is directly proportional to the value of the eigenvalue (i.e. the larger the eigenvalue indicates the larger variance the eigenvector represents). Hence, the eigenvectors are sorted with respect to their corresponding eigenvalues. The eigenvector having the largest eigenvalue is marked as the first eigenvector, and so on. In this manner, the most generalizing eigenvector comes first in the eigenvector matrix.

3.3.1 USING EIGENFACE TO CLASSIFY A FACE IMAGE

In the next step, the training images are projected into the eigenface space and thus the weight of each eigenvector to represent the image in the eigenface space is calculated. This weight is simply the dot product of each image with each of the eigenvectors. A new face image (Γ) is transformed into its eigenface components (projected onto "face space") by a simple operation

$$\text{Projection } \omega_k = v_k^T \cdot \Phi = v_k^T \cdot (\Gamma - \Psi) \quad (3.31)$$

is the projection of a training image on each of the eigenvectors where $k = 1, 2, \dots, M'$

$$\text{Weight Matrix } \Omega = [\omega_1, \omega_2, \dots, \omega_{M'}]^T \quad (3.32)$$

is the representation of the training image in the eigenface space and its size is $(M' \times 1)$.

At this point, the images are just composed of weights in the eigenface space, simply like they have pixel values in the image space. The important aspect of the eigenface transform lies in this property.

Each image is represented by an image of size $(N_x \times N_y)$ in the image space, whereas the same image is represented by a vector of size $(M' \times 1)$ in the eigenface space. Moreover, having the dimension structure related to the variance of the data in hand makes the eigenface representation a generalized representation of the data. This makes the algorithm a solution to the “curse of dimensionality” problem seen in the standard pattern recognition task.

When a new test image is to be classified, it is also mean subtracted and projected onto the eigenface space and Nearest Mean algorithm (Turk and Pentland 1991) is used for the classification of the test image vector in the standard eigenface method; that is, the test image is assumed to belong to the nearest class by calculating the Euclidean distance of the test image vector to the mean of each class of the training image vectors.

Test image vector Γ_T (3.33)

is the test image vector of size $(P \times 1)$.

Mean subtracted image $\Phi_T = \Gamma_T - \Psi$ (3.34)

is the difference of the test image from the mean image (size $P \times 1$).

Projection $\omega_k = v_k^T \cdot \Phi_T = v_k^T \cdot (\Gamma - \Psi)$ (3.35)

is the projection of a training image on each of the eigenvectors where $k = 1, 2, \dots, M'$

Weight Matrix $\Omega_T = [\omega \ \omega \dots \omega_{M'}]^T$ (3.36)

is the representation of the test image in the eigenface space and its size is $(M' \times 1)$.

The training and test image vectors can be reconstructed by a back transformation from the eigenface space to the image vector space.

$$\textbf{Reconstructed image vector } \Gamma_f = v \cdot \Omega + \Psi = \Phi_f + \Psi \quad (3.37)$$

If possible, it is better to work on a database of more than one image per individual in order to increase the robustness to minor changes in expression, illumination and slight variations of view angles. A class of images for an individual can be formed and this class can be considered as the representative image vector of that class. For an individual having q_i images in the database, the average of the projections of each class is the mean of all the projected image vectors in that class. In mathematical terms;

$$\textbf{Average Class Projection } \Omega_\Psi = \frac{1}{q_i} \sum_{i=1}^{q_i} \Omega_i \quad (3.38)$$

This average class projection can be used as one of the vectors (representing an image class instead of an image vector) to compare with the test image vector.

A similarity measure is defined as the distance between the test image vector and i-th face class;

$$\textbf{Similarity } \delta_i = \|\Omega_T - \Omega_{\Psi_i}\| = \sqrt{\sum_{k=1}^{M_t} (\Omega_{Tk} - \Omega_{\Psi_{ik}})^2} \quad (3.39)$$

is the Euclidean distance between projections.

A distance threshold may be defined for the maximum allowable distance from any face class, which is half of the distance between the two most distant classes;

$$\text{Distance Threshold } \Theta = \frac{1}{2} \max(\|\Omega_{\Psi_i} - \Omega_{\Psi_j}\|) \quad (3.40)$$

Classification procedure of the Eigenface method ensures that face image vectors should fall close to their reconstructions, whereas non-face image vectors should fall far away. Hence, a distance measure is defined;

$$\text{Distance measure } \varepsilon^2 = \|\Phi - \Phi_f\|^2 \quad (3.41)$$

is the distance between the mean subtracted image and the reconstructed image.

The recognition of an image knowing these two measures δ_i and ε can be done as follows:

- (a) If $\varepsilon > \Theta \Rightarrow$ the image is not a face (independent of the values of δ_i),
- (b) If $\varepsilon < \Theta$ and for all i , $\delta_i > \Theta \Rightarrow$ the image is an unknown face,
- (c) If $\varepsilon < \Theta$ and for one of i , $\delta_i < \Theta \Rightarrow$ the image belongs to the training face class i .

This is the standard Eigenface approach suggested by Turk and Pentland.

3.3.1.1 WORKED EXAMPLE

This section treats an example of the PCA process with three matrices, which are assumed to be images that make up a database.

1. Assume A, B and C are the matrix representation of images in a database

$$A = \begin{pmatrix} 8 & 1 & 7 \\ 10 & 9 & 8 \\ 7 & 10 & 8 \end{pmatrix} \quad B = \begin{pmatrix} 4 & 8 & 1 \\ 7 & 1 & 1 \\ 2 & 3 & 9 \end{pmatrix} \quad C = \begin{pmatrix} 7 & 1 & 8 \\ 4 & 5 & 8 \\ 10 & 4 & 2 \end{pmatrix}$$

2. Reshape matrices such that each row represents one matrix

$$R = \begin{pmatrix} 8 & 10 & 7 & 1 & 9 & 10 & 7 & 8 & 8 \\ 4 & 7 & 2 & 8 & 1 & 3 & 1 & 1 & 9 \\ 7 & 4 & 10 & 1 & 5 & 4 & 8 & 8 & 2 \end{pmatrix}$$

3. Calculate the mean of the reshaped matrix using the formula:

$$\Psi = \frac{1}{M_t} \sum_{i=1}^{M_t} \Gamma_i$$

This results in the mean vector:

6.3333 7.0000 6.3333 3.3333 5.0000 5.6667 5.3333 5.6667 6.3333

4. Subtract the mean from the reshaped matrix to form the mean centered image matrix,

C.

$$C = \begin{pmatrix} 1.6667 & 3.0000 & 0.6667 & -2.3333 & 4.0000 & 4.3333 & 1.6667 & 2.3333 & 1.6667 \\ -2.3333 & 0 & -4.3333 & 4.6667 & -4.0000 & -2.6667 & -4.3333 & -4.6667 & 2.6667 \\ 0.6667 & -3.0000 & 3.6667 & -2.3333 & 0 & -1.6667 & 2.6667 & 2.3333 & -4.3333 \end{pmatrix}$$

5. Calculate the covariance matrix, L of the mean centered matrix, C using the formula:

$$L = C' * C, \text{ where } C' \text{ is the transpose of } C.$$

$$L = \begin{pmatrix} 1.6667 & -2.3333 & 0.6667 \\ 3 & 0 & -3 \\ 0.6667 & -4.3333 & 3.6667 \\ -2.3333 & 4.6667 & -2.3333 \\ 4 & -4 & 0 \\ 4.3333 & -2.6667 & -1.6667 \\ 1.6667 & -4.3333 & 2.6667 \\ 2.3333 & -4.6667 & 2.3333 \\ 1.6667 & 2.6667 & -4.3333 \end{pmatrix} * \begin{pmatrix} 1.6667 & 3 & 0.6667 & \dots & \dots & \dots & 1.6667 & 2.3333 & 1.6667 \\ -2.3333 & 0 & -4.3333 & \dots & \dots & \dots & -4.3333 & -4.6667 & 2.6667 \\ 0.6667 & -3 & 3.6667 & \dots & \dots & \dots & 2.6667 & 2.3333 & -4.3333 \end{pmatrix}$$

KNUST

This result in:

$$L = \begin{pmatrix} 8.6667 & 3 & 13.6667 & -16.3333 & 16 & 12.3333 & 14.6667 & 16.3333 & -6.3333 \\ 3 & 18 & -9 & 0 & 12 & 18 & -3 & 0 & 18 \\ 13.6667 & -9 & 32.6667 & -30.3333 & 20 & 8.3333 & 29.6667 & 30.3333 & -26.3333 \\ -16.3333 & 0 & -30.3333 & 32.6667 & -28 & -18.6667 & -30.3333 & -32.6667 & 18.6667 \\ 16 & 12 & 20 & -28 & 32 & 28 & 24 & 28 & -4 \\ 12.3333 & 18 & 8.3333 & -18.6667 & 28 & 28.6667 & 14.3333 & 18.6667 & 7.3333 \\ 14.6667 & -3 & 29.6667 & -30.3333 & 24 & 14.3333 & 28.6667 & 30.3333 & -20.3333 \\ -16.3333 & 0 & 30.3333 & -32.6667 & 28 & 18.6667 & 30.3333 & 32.6667 & -18.6667 \\ -6.3333 & 18 & -26.3333 & 18.6667 & -4 & 7.3333 & -20.3333 & -18.6667 & 28.6667 \end{pmatrix}$$

6. Calculate the eigenvalues and eigenvectors of the covariance matrix, L by finding

$$\det|L - \lambda I| = 0$$

This produces the eigenvalues:

$$\begin{pmatrix} -0.0000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.0000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.0000 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.0000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 67.4930 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 175.1736 \end{pmatrix}$$

And corresponding eigenvectors:

$$\begin{pmatrix} 0.0615 & -0.0125 & 0.8605 & -0.4408 & -0.0279 & 0.0766 & -0.0285 & 0.0833 & -0.2163 \\ 0.1824 & -0.0722 & 0.1118 & 0.4025 & -0.5082 & 0.3024 & 0.4130 & 0.5164 & -0.0026 \\ -0.0132 & 0.1089 & 0.0147 & 0.3140 & 0.0739 & 0.7103 & -0.3874 & -0.2634 & -0.3997 \\ -0.1620 & 0.2204 & 0.3214 & 0.3549 & 0.6186 & 0.1552 & 0.3205 & 0.0056 & 0.4318 \\ 0.2755 & 0.4967 & 0.0701 & 0.3334 & 0.1723 & -0.4609 & -0.2562 & 0.3395 & -0.3719 \\ 0.1564 & -0.4511 & -0.2380 & -0.2161 & 0.5555 & 0.1830 & 0.0335 & 0.5132 & -0.2493 \\ -0.1216 & 0.5140 & -0.2404 & -0.3390 & 0.0400 & 0.1372 & 0.5993 & -0.0913 & -0.4005 \\ -0.7129 & -0.3287 & 0.1214 & 0.3020 & -0.0034 & -0.2727 & 0.1300 & -0.0056 & -0.4318 \\ -0.5596 & 0.3343 & -0.0968 & -0.2347 & -0.1139 & 0.1871 & -0.3647 & 0.5196 & 0.2442 \end{pmatrix}$$

7. The eigenvalues are sorted and those with values less than 0 are eliminated. The eigenvectors that correspond to the retained eigenvalues then become:

$$V = \begin{pmatrix} 0.0833 & -0.2163 \\ 0.5164 & -0.0026 \\ -0.2634 & -0.3997 \\ 0.0056 & 0.4318 \\ 0.3395 & -0.3719 \\ 0.5132 & -0.2493 \\ -0.0913 & -0.4005 \\ -0.0056 & -0.4318 \\ 0.5196 & 0.2442 \end{pmatrix}$$

8. Calculate the eigenface by multiplying the centered image matrix by the eigenvectors of the retained eigenvalues.

Thus, eigenface = C * V

Eigenface=

$$\begin{pmatrix} 1.6667 & 3.0000 & 0.6667 & -2.3333 & 4.0000 & 4.3333 & 1.6667 & 2.3333 & 1.6667 \\ -2.3333 & 0 & -4.3333 & 4.6667 & -4.0000 & -2.6667 & -4.3333 & -4.6667 & 2.6667 \\ 0.6667 & -3.0000 & 3.6667 & -2.3333 & 0 & -1.6667 & 2.6667 & 2.3333 & -4.3333 \end{pmatrix} \times \begin{pmatrix} 0.0833 & -0.2163 \\ 0.5164 & -0.0026 \\ -0.2634 & -0.3997 \\ 0.0056 & 0.4318 \\ 0.3395 & -0.3719 \\ 0.5132 & -0.2493 \\ -0.0913 & -0.4005 \\ -0.0056 & -0.4318 \\ 0.5196 & 0.2442 \end{pmatrix}$$

$$\text{Hence, eigenface} = \begin{pmatrix} 5.7820 & -5.4784 \\ 0.0539 & 10.8062 \\ -5.8360 & -5.3297 \end{pmatrix}$$

9. The centered images are projected unto a face space by multiplying it by the eigenface. Thus projected image, P = eigenface' x centered image.

$$P = \begin{pmatrix} 5.7820 & 0.0539 & -5.8360 \\ -5.4784 & 10.8062 & -5.3297 \end{pmatrix} \times \begin{pmatrix} 1.6667 & 3.0000 & 0.6667 & \dots & \dots & \dots & \dots & \dots & 1.6667 \\ -2.3333 & 0 & -4.3333 & \dots & \dots & \dots & \dots & \dots & 2.6667 \\ 0.6667 & -3.0000 & 3.6667 & \dots & \dots & \dots & \dots & \dots & -4.3333 \end{pmatrix}$$

Thus, projected image then becomes:

5.6202	34.8539	-17.7776	0.3776	22.9122	34.6381	-6.1597	-0.3776	35.0697
-37.8971	-0.4516	-70.0148	75.6437	-65.1385	-43.6766	-70.1653	-75.6437	42.7733

It is obvious that, the dimension of the initial matrix has been reduced by the process, PCA.

This projected image forms the feature vector for the recognition process.

3.3.2 SUMMARY OF EIGENFACE RECOGNITION

To summarize the eigenfaces approach to face recognition involves the following steps:

Collect a set of characteristic face images of the known individuals. This set should include a number of images of each person, with some variation in expression and in the lighting. (Say four images of ten people, so $M = 40$)

Calculate the (40×40) matrix L , find its eigenvalues and eigenvectors, and choose the M' eigenvectors with the highest associated eigenvalues. (Let $M' = 10$ in example).

Combine the normalized set of images to produce the $(M' = 10)$ eigenfaces μ_k .

For each known individual, calculate the class vector Ω_k by averaging pattern vector Ω calculated from the original (four) images of the individual. Choose a threshold θ_ϵ that defines the maximum allowable distance from any face class, and a threshold θ_ϵ that defines the maximum allowable distance from face space.

For each new image to be identified, calculate its pattern vector Ω , the distance ϵ_I to each known class, and the distance ϵ to face space. If the minimum distance $\epsilon_k < \theta_\epsilon$ and the distance $\epsilon < \theta_\epsilon$, classify the input face as the individual associated with class vector ω_k . If the minimum distance $\epsilon_k = \theta_\epsilon$ but distance $\epsilon < \theta_\epsilon$, then the image may be classify as "unknown", and optionally used to begin a new face class.

If new image is classified as a known individual, this image may be added to the original set of familiar face images, and the eigenfaces may be recalculated. This gives the opportunity to modify the face space as the system encounters more instances of the known faces.

3.3.3 SIGNIFICANCE OF EIGENFACE APPROACH

The standard Eigenface Approach for Face Recognition has been seen. Here, M eigenvectors for representing training set images was found out. Now it is important to choose only M' eigenvectors from these M eigenvectors, such that M' is less than M , to represent face space spanned by images. This will reduce the face space dimensionality and enhance speed for face recognition. Here we are reducing the dimensionality of face images.

One can choose only M' eigenvectors with highest eigenvalues. Now as the higher eigenvalues represent maximum face variation in the corresponding eigenvector direction, it is important to consider this eigenvector for face space representation. Since, the lower eigenvalues does not provide much information about face variations in corresponding eigenvector direction, such small eigenvalues can be neglected to further reduce the dimension of face space. This does not affect the success rate much and is acceptable depending on the application of face recognition. Here only M' eigenvectors with higher eigenvalues are chosen for defining face space. The most important point here is to choose value of M' (Maximum eigenvalues chosen from Covariance Matrix) so that it does not result in high error rates for Face recognition process. The M' should be chosen such that error rates does not increase much and are acceptable depending on application for face recognition.

CHAPTER 4

EXPERIMENTAL ANALYSIS AND RESULTS

4.0 INTRODUCTION

In this section, experimental results that were obtained from the proposed face recognition system are given.

4.1 DATA COLLECTION

Facial images were taken from Faces 94 database. It is made up of 152 individuals who comprise 20 female, 112 male and 20 male staff in separate directories. Separate directories were merged into one folder to achieve different lighting effect. The subjects were sitting at approximately the same distance from the camera and were asked to speak while a sequence of twenty images was taken. The speech was used to introduce moderate and natural facial expression variation.

Twenty images of each person were taken at a resolution of 180 by 200 pixels in a portrait format on a plain green background. Each image is indexed 1 to 20 for the 20 images per person. In all, a total of 3040 images make up the database for this thesis. For the purposes of recognition and testing the performance of the algorithm, the dataset is divided into two: the training set and testing set. The training set contains 15 images of the 152 images while the testing set contains the remaining 5 images which is usually 80% and 20% respectively as recommended by literature. This resulted in 2280 training images and 760 test images.

Fig.4.1 below shows the first 15 images of the 1st person with different facial expression.



Fig 4.1: First 15 images of the 1st person that form the training set.

4.2 FACE FEATURE ANALYSIS

Face feature algorithm with experimental results are explained in this section of the chapter.

4.2.1 MACHINE SPECIFICATIONS

With the chosen dataset, a Matlab code was written using Matlab 2012a version and executed in Toshiba with the following specifications:

- Operating System: Windows 7 Ultimate 64-bit
- System Model: Satellite C655
- BIOS: InsydeH20 Version 1.90
- Processor: Intel(R) Core(TM) i3 CPU
- Memory: 4GB RAM
- HDD: 520 GB
- DirectX Version: DirectX 11

4.2.2 FACE FEATURE EXTRACTION ALGORITHM

The following are steps of the algorithm:

Step 1(a): Obtain the matrix representation of images using the code line:

Input image = imread (image); (See appendix)

Images at hand are in colour format which is 3 dimensions (3D), thus, RGB (Red, Green, and Blue) on the x, y, and z axis. Fig 4.3(a), Fig. 4.3(b) and Fig.4.3(c) are the Red, Blue and Green matrix components respectively of a colour image (Fig. 4.2). Fig. 4.2 below is the colour image of a person in the dataset.

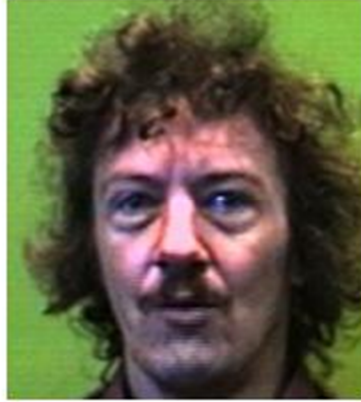


Fig. 4.2: Colour image of the first person in the dataset

The matrix below is the red component of Fig. 4.2

$$R = \begin{pmatrix} 125 & 125 & 125 & 128 & \dots & \dots & \dots & \dots & 126 & 122 & 123 & 128 \\ 128 & 127 & 127 & 130 & \dots & \dots & \dots & \dots & 129 & 126 & 127 & 132 \\ 128 & 127 & 128 & 130 & \dots & \dots & \dots & \dots & 130 & 128 & 130 & 134 \\ 125 & 125 & 126 & 128 & \dots & \dots & \dots & \dots & 127 & 127 & 129 & 131 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 109 & 107 & 107 & 106 & \dots & \dots & \dots & \dots & 103 & 101 & 101 & 101 \\ 110 & 108 & 109 & 108 & \dots & \dots & \dots & \dots & 105 & 103 & 103 & 103 \\ 110 & 107 & 108 & 108 & \dots & \dots & \dots & \dots & 104 & 103 & 103 & 103 \\ 107 & 104 & 103 & 106 & \dots & \dots & \dots & \dots & 102 & 101 & 101 & 101 \end{pmatrix}$$

Fig. 4.3(a): Red component of Fig. 4.2

This matrix below is the green component of Fig 4.2

$$G = \begin{pmatrix} 185 & 185 & 185 & 185 & \dots & \dots & \dots & \dots & 185 & 180 & 181 & 186 \\ 188 & 187 & 187 & 190 & \dots & \dots & \dots & \dots & 188 & 184 & 185 & 190 \\ 188 & 187 & 188 & 190 & \dots & \dots & \dots & \dots & 189 & 187 & 189 & 192 \\ 185 & 185 & 186 & 188 & \dots & \dots & \dots & \dots & 186 & 186 & 188 & 189 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \dots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \ddots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \ddots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \dots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 127 & 127 & 127 & 129 & \dots & \dots & \dots & \dots & 130 & 130 & 130 & 130 \\ 127 & 127 & 129 & 131 & \dots & \dots & \dots & \dots & 132 & 132 & 132 & 132 \\ 126 & 126 & 128 & 131 & \dots & \dots & \dots & \dots & 133 & 134 & 134 & 134 \\ 123 & 123 & 125 & 128 & \dots & \dots & \dots & \dots & 131 & 132 & 132 & 132 \end{pmatrix}$$

Fig. 4.3(b): Green component of Fig. 4.2

The matrix below shows the blue component of Fig. 4.2

$$B = \begin{pmatrix} 71 & 71 & 72 & 75 & \dots & \dots & \dots & \dots & 79 & 77 & 78 & 83 \\ 74 & 73 & 74 & 77 & \dots & \dots & \dots & \dots & 82 & 81 & 82 & 87 \\ 75 & 74 & 75 & 77 & \dots & \dots & \dots & \dots & 83 & 81 & 83 & 89 \\ 72 & 72 & 73 & 75 & \dots & \dots & \dots & \dots & 80 & 80 & 82 & 86 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \dots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \ddots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \ddots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \dots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 45 & 42 & 38 & 38 & \dots & \dots & \dots & \dots & 37 & 37 & 37 & 37 \\ 49 & 45 & 42 & 41 & \dots & \dots & \dots & \dots & 39 & 39 & 39 & 39 \\ 51 & 45 & 43 & 43 & \dots & \dots & \dots & \dots & 40 & 40 & 40 & 40 \\ 50 & 42 & 40 & 43 & \dots & \dots & \dots & \dots & 38 & 38 & 38 & 38 \end{pmatrix}$$

Fig. 4.3(c) Blue component of Fig. 4.2

Step 1(b). Convert colour image of Fig. 4.2 to grey image of Fig. 4.5 and extract its matrix representation for easy computation. The code line that extracts the matrix representation of the grey image is given below:

`imread (image); (See appendix )`

This is the matrix representation of the grey image of Fig. 4.5

154	154	154	157	160	...	...	...	...	...	156	152	159	155	151
157	156	156	159	162	...	...	...	...	...	159	155	162	158	155
157	156	157	159	161	...	...	...	...	...	159	156	161	159	157
154	154	155	157	157	...	...	...	...	...	156	155	157	156	156
154	155	156	156	156	...	...	...	...	...	156	157	156	156	156
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
137	137	137	137	138	...	...	...	...	...	137	137	136	136	136
139	138	137	136	136	...	...	...	...	...	135	134	134	134	135
143	142	140	138	137	...	...	...	...	...	136	136	136	136	136
142	141	140	139	138	...	...	...	...	...	138	139	137	137	137
138	137	136	135	136	...	...	...	...	...	138	139	136	136	136

Fig 4.4: Matrix representation of a grey image (Fig. 4.5)

The image below represents the grey image of Fig. 4.2. The conversion from colour image to grey image is done by the matlab code line:

`img = rgb2gray(img);` (See appendix)



Fig. 4.5 Grey image of Fig. 4.2

The pixel intensity value on the grey scale runs from 0 to 255. Thus, the closer the value is to 0, the darker the image and the closer the value is to 1, the brighter the image.

Step 2. Reshape image from 2D to 1D in order to create the face database with each row representing the image of a person. This creates a row vector of dimension 1 x 36000. Reshaping the image from 2D to 1D is done by the code line:

```
temp = reshape (img', irow * icol,1); (See appendix)
```

This yields the result as shown in Fig. 4.6 which is the resulting vector after Fig. 4.4 has been transformed into a 1D image.

```
temp = (154 154 154 157 160 ... ... 109 112 112 112 112)
```

Fig 4.6: Vector representation of Fig. 4.4.

The process in steps 1 and 2 are repeated for all images in the training set to get temp i for image i in training set.

Step 3(a). Compute the average face image (mean face) of the database. This is done with the code line:

$$m = \text{mean}(T, 2), \text{ (See appendix)}$$

Where m is the average face image and T is the training set in the database. This line in the code executes the formula:

$$m = \frac{1}{M_t} \sum_{i=1}^{M_t} \text{temp}_i$$

Where M_t is the number of training images in the dataset and temp_i is an image vector in the dataset.

Step 3(b). The mean face, m , is a vector which is reshaped into 2D to be displayed as a matrix and an image. The code line that reshapes the mean face from a vector to a matrix is given by the code line:

$$F = \text{reshape}(irow, icol); \text{ (See appendix)}$$

The figure below shows the matrix representation of the mean face (average face) matrix as calculated from the dataset.

$$m = \begin{pmatrix} 121.5943 & 123.4965 & 123.9132 & \dots & \dots & \dots & 90.6526 & 89.0127 & 89.4553 \\ 121.7684 & 123.6939 & 124.0886 & \dots & \dots & \dots & 90.1697 & 88.7496 & 89.2614 \\ 121.9706 & 123.9149 & 124.2715 & \dots & \dots & \dots & 89.6294 & 88.5140 & 89.1285 \\ \vdots & \vdots & \vdots & \ddots & \dots & \dots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \dots & \dots & \dots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \dots & \dots & \dots & \vdots & \vdots & \vdots \\ 122.6303 & 124.5018 & 124.8943 & \dots & \dots & \dots & 80.6728 & 80.1254 & 80.8645 \\ 123.2250 & 124.9092 & 125.1864 & \dots & \dots & \dots & 81.1364 & 81.1171 & 82.0241 \\ 123.1039 & 124.7627 & 125.0421 & \dots & \dots & \dots & 81.6377 & 81.5820 & 82.4991 \end{pmatrix}$$

Fig 4.7: Matrix representation of the mean face as calculated from the database

The figure below shows is the image of the mean face matrix as calculated from the dataset



Fig 4.8: Mean face obtained from the database.

Step 3(c). Subtract the mean face vector as calculated from each image of the database created in step 2, to form the centered image. Computing the average face image is done by the code line:

```
A = [ ]
for i=1:Train_Number
temp = double (tempi – m)
A = [A temp];
end. (See appendix)
```

Where A is the centered image. The code executes the formula:

$$A = \text{temp} - m$$

The figure below is the matrix representation of the centered image as calculated from the database.

$$A = \begin{pmatrix} 32.4057 & 33.4057 & 34.4057 & 37.4057 & \dots & \dots & \dots & \dots & 33.4057 & 42.4057 & 37.4057 & 38.4057 \\ 32.2316 & 32.2316 & 33.2316 & 36.2316 & \dots & \dots & \dots & \dots & 34.2316 & 42.2316 & 38.2316 & 37.2316 \\ 32.0294 & 31.0294 & 32.0294 & 34.0294 & \dots & \dots & \dots & \dots & 36.0294 & 43.0294 & 38.0294 & 35.0294 \\ 34.9404 & 29.9404 & 30.9404 & 32.9404 & \dots & \dots & \dots & \dots & 37.9404 & 42.9404 & 36.9404 & 33.9404 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \dots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \ddots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \ddots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \dots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 31.2781 & 28.2781 & 30.2781 & 25.2781 & \dots & \dots & \dots & \dots & -70.7219 & -70.7219 & -70.7219 & -70.7219 \\ 30.2886 & 28.2886 & 30.2886 & 26.2886 & \dots & \dots & \dots & \dots & -71.7114 & -71.7114 & -71.7114 & -71.7114 \\ 29.4949 & 27.4949 & 29.4949 & 27.4949 & \dots & \dots & \dots & \dots & -72.5053 & -72.5053 & -72.5053 & -72.5053 \\ 29.3939 & 28.3939 & 30.3939 & 30.3939 & \dots & \dots & \dots & \dots & -71.6061 & -72.6061 & -73.6061 & -72.6061 \end{pmatrix}$$

Fig. 4.9: Matrix representation of centered image as calculated from the database

Step 4. Calculate the covariance of the resulting output in step 3. This is done by the code line:

$$L = A' * A; \text{ (See appendix)}$$

Which executes the formula:

$$X = A \cdot A^T$$

In Fig. 4.10 below the result of the covariance matrix of the centered image as calculated from the database is shown.

$$L = 1.0e+07 \begin{pmatrix} 6.0372 & 5.6938 & 5.7500 & 5.8863 & \dots & \dots & \dots & \dots & 1.5207 & 1.5844 & 1.6258 & 1.5226 \\ 5.6938 & 6.0929 & 6.0674 & 5.9781 & \dots & \dots & \dots & \dots & 1.6537 & 1.7222 & 1.7540 & 1.6643 \\ 5.7500 & 6.0674 & 6.1013 & 6.0073 & \dots & \dots & \dots & \dots & 1.6170 & 1.6791 & 1.7177 & 1.6283 \\ 5.8863 & 5.9781 & 6.0073 & 6.0764 & \dots & \dots & \dots & \dots & 1.5745 & 1.6419 & 1.6802 & 1.5860 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \dots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \ddots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \ddots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \dots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 1.5207 & 1.6537 & 1.6170 & 1.5745 & \dots & \dots & \dots & \dots & 1.1108 & 1.1109 & 1.1055 & 1.1028 \\ 1.5844 & 1.7222 & 1.6791 & 1.6419 & \dots & \dots & \dots & \dots & 1.1109 & 1.1454 & 1.1254 & 1.1235 \\ 1.6258 & 1.7540 & 1.7177 & 1.6802 & \dots & \dots & \dots & \dots & 1.1105 & 1.1254 & 1.1334 & 1.1217 \\ 1.5226 & 1.6643 & 1.6283 & 1.5860 & \dots & \dots & \dots & \dots & 1.1028 & 1.1235 & 1.1217 & 1.1359 \end{pmatrix}$$

Fig 4.10: Matrix representation of the covariance matrix as calculated from the database

Step 5. Calculate the eigenvalues and eigenvectors of the covariance matrix. This is done by the code line:

$$[V \ D] = \text{eig} (L); \text{ (See appendix)}$$

where V is the eigenvector and D is the eigenvalue. This line in the code executes the formula:

$$\det|L - \lambda I| = 0$$

The result of the eigenvalues as calculated in the database is shown below in Fig. 4.11.

$$D = 1.0e+04(-0.0000 \ 5.2922 \ 5.4617 \ 5.6255) \dots\dots\dots 1.0e+10(0.8681 \ 1.2861 \ 2.4836 \ 5.3323)$$

Fig 4.11: Vector representation of eigenvalues in the database

Fig. 4.12 below is a plot of the eigenvalues as calculated from the database.

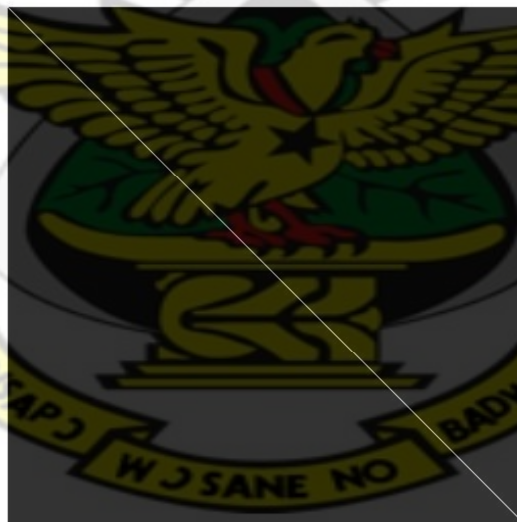


Fig. 4.12: A plot of eigenvalues as calculated in the database

The matrix representation of the eigenvectors as calculated from the database is shown in the figure below:

$$V = \begin{pmatrix} -0.0209 & 0.0012 & 0.0015 & -0.0016 & \dots & \dots & \dots & \dots & -0.0074 & 0.0046 & -0.0173 & -0.0001 \\ -0.0209 & 0.0132 & -0.0087 & -0.0068 & \dots & \dots & \dots & \dots & -0.0111 & 0.0056 & -0.0169 & 0.0004 \\ -0.0209 & -0.0060 & 0.0040 & -0.0026 & \dots & \dots & \dots & \dots & -0.0107 & 0.0051 & -0.0171 & 0.0004 \\ -0.0209 & -0.0066 & 0.0026 & 0.0031 & \dots & \dots & \dots & \dots & -0.0095 & 0.0047 & -0.0173 & 0.0002 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \dots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \ddots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \ddots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \dots & \ddots & \vdots & \vdots & \vdots & \vdots \\ -0.0209 & -0.0008 & -0.0058 & 0.0089 & \dots & \dots & \dots & \dots & -0.0406 & 0.0524 & -0.0096 & 0.0123 \\ -0.0209 & -0.0023 & 0.0020 & -0.0046 & \dots & \dots & \dots & \dots & -0.0431 & 0.0505 & -0.0111 & 0.0123 \\ -0.0209 & 0.0001 & 0.0037 & -0.0033 & \dots & \dots & \dots & \dots & -0.0426 & 0.0509 & -0.0108 & 0.0122 \\ -0.0209 & -0.0015 & 0.0011 & 0.0015 & \dots & \dots & \dots & \dots & -0.0437 & 0.0511 & -0.0105 & 0.0125 \end{pmatrix}$$

Fig 4.13: Matrix representation of eigenvectors of covariance matrix as calculated in the database.

Step 6. All eigenvalues are sorted and those with values less than 0 are eliminated. The lines in the code that returns the eigenvectors with corresponding retained eigenvalues are given by:

```
for i=1: size (V, 2)
    If (D (i, i) > 0)
        L_eig = [L_eig, V (i, i)];
    end
end (See appendix)
```

In Fig. 4.14, below, results of the eigenvectors whose eigenvalues have been retained are shown.

$$L_eig = \begin{pmatrix} 0.0012 & 0.0015 & -0.0016 & 0.0052 & \dots & \dots & \dots & \dots & -0.0074 & 0.0046 & -0.0173 & -0.0001 \\ 0.0132 & -0.0087 & -0.0068 & -0.0057 & \dots & \dots & \dots & \dots & -0.0111 & 0.0056 & -0.0169 & 0.0004 \\ -0.0060 & 0.0042 & 0.0026 & 0.0066 & \dots & \dots & \dots & \dots & -0.0107 & 0.0051 & -0.0171 & 0.0004 \\ -0.0066 & 0.0026 & 0.0031 & 0.0009 & \dots & \dots & \dots & \dots & -0.0095 & 0.0047 & -0.0173 & 0.0002 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \dots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \ddots & \dots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \ddots & \dots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \dots & \ddots & \vdots & \vdots & \vdots & \vdots \\ -0.0008 & -0.0058 & 0.0089 & -0.0054 & \dots & \dots & \dots & \dots & -0.0406 & 0.0524 & -0.0096 & 0.0123 \\ -0.0023 & 0.0020 & -0.0046 & 0.0049 & \dots & \dots & \dots & \dots & -0.0431 & 0.0505 & -0.0111 & 0.0123 \\ 0.0001 & 0.0037 & -0.0033 & 0.0024 & \dots & \dots & \dots & \dots & -0.0426 & 0.0509 & -0.0108 & 0.0122 \\ -0.0015 & 0.0011 & 0.0015 & -0.0050 & \dots & \dots & \dots & \dots & -0.0437 & 0.0511 & -0.0105 & 0.0125 \end{pmatrix}$$

Fig. 4.14: Matrix representation of eigenvectors of retained eigenvalues in the dataset.

Step 7. Calculate the eigenface by multiplying the eigenvectors whose eigenvalues have been retained by the centered images. It is done by the code line:

$$\text{Eigenfaces} = A * L_eig; \text{ (See appendix)}$$

Fig. 4.15 below shows the matrix representation of the eigenface as calculated from the database.

$$\text{eigenfaces} = \begin{pmatrix} 1.2628 & -0.8880 & 2.2918 & \dots & \dots & \dots & 23.9000 & -37.4000 & -335.200 \\ 1.9884 & -0.1741 & -0.6752 & \dots & \dots & \dots & -23.8000 & -37.4000 & -333.800 \\ 2.7122 & 1.2593 & -2.0094 & \dots & \dots & \dots & -25.1000 & -38.0000 & -334.5000 \\ \vdots & \vdots & \vdots & \ddots & \dots & \dots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \dots & \ddots & \dots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \dots & \dots & \ddots & \vdots & \vdots & \vdots \\ 1.4249 & -1.1811 & 0.5765 & \dots & \dots & \dots & 535.6272 & -762.3666 & -627.3352 \\ 0.0406 & -0.8787 & 1.2970 & \dots & \dots & \dots & 513.6269 & -777.9330 & -636.2142 \\ -0.9808 & 0.3165 & 2.0371 & \dots & \dots & \dots & 495.4350 & -795.8310 & -640.5101 \end{pmatrix}$$

Fig. 4.15: Matrix representation of the eigenface as calculated from the database

Step 8. The projected image is then formed by multiplying the eigenface by the centered image. Thus,

$$\text{Projected image} = \text{eigenface}' \times \text{centered image}$$

The code line for the projected image is given by:

```
ProjectedImages = [ ]

for i =1:Train_Number

    temp = Eigenfaces'*A(:,i);

    ProjectedImages = [ProjectedImages temp];

end. (See appendix)
```

This projected vector of each face forms the feature vector for face recognition. Fig. 4.16 shows in matrix representation, the result of the projected image as calculated from the database.

$$proging = \begin{pmatrix} 62.5469 & 698.4231 & -3154990 & \dots & \dots & \dots & -41.0615 & -119.7481 & 7.8709 \\ 82.9130 & -473.7044 & 226.9828 & \dots & \dots & \dots & -316.1269 & 109.3103 & 201.2534 \\ -87.3100 & -380.8487 & 145.5649 & \dots & \dots & \dots & 502.0833 & -259.7416 & -183.0195 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots \\ 7.212e+07 & 6.512e+07 & -6.016e+07 & \dots & \dots & \dots & 6.7434e+08 & 6.4967e+08 & 6.5519e+08 \\ -4.2087e+08 & -4.2382e+08 & -4.3021e+08 & \dots & \dots & \dots & -2.3787e+08 & -2.7673e+08 & -2.682e+08 \\ 2.036e+07 & 1.910e+07 & 1.103e+07 & \dots & \dots & \dots & 6.5497e+08 & 6.5669e+08 & 6.4935e+08 \end{pmatrix}$$

Fig. 4.16: Matrix representation of projected images as calculated from the database

This feature vector obtained is kept in the system for face recognition comparison.

4.3 FACE RECOGNITION ALGORITHM

This algorithm shows the processes involved when a new image is to be recognized.

At the recognition stage:

Step 1. The eigenfaces, the image index along with the projected image are loaded into the computer system.

Step 2. The image to be tested is allowed to go through the feature extraction process of section 4.2.2 in order to get its feature vector. This is given by the code line:

```
InputImage = imread (TestImage);

Temp = InputImage (:, :, 1);

[irow icol] = size (temp);

InImage = reshape (temp', irow*icol,1);

Difference = double (InImage)-m;

ProjectedTestImage = Eigenfaces'*Difference;

end (See appendix)
```

Step 3. This feature vector of the test image is then compared with the projected images of the training set in the database.

Step 4. The Euclidean distance between this tested image and the projected image in the database is then calculated. The code line for the calculation is given below:

```
Euc_dist = [ ];

Train_Number = size (Eigenfaces, 2);

for i = 1:Train_Number

    if i ~ = img_map(input_idx)

        q = ProjectedImages (: , i);

        temp = (norm (ProjectedTestImage-q))^2;

        Euc_dist = [Euc_dist temp];

    end

end (See appendix)
```

This code executes the formula:

$$\mathcal{E} = \|\Omega - \Omega_k\|,$$

where Ω = projected test image and Ω_k = projected image.

The Euclidean distances between the test projected image and the projected image of the training set as calculated by the system is given below:

1.0e+18(0.1573 1.8727 2.8569 ... 3.4800 3.8355 ... 7.5558 0.9497 0.9297)

Step 5. The smallest Euclidean distance that corresponds to an image in the training set is assumed to be a match. It is selected along with its index. This is done using the code line:

```
min (Euc-dist); (See appendix)
```

The code executes: $\min (d_1, d_2, d_3, d_4, \dots, d_n)$, where $d_1, d_2, d_3, d_4, \dots, d_n$ are the various Euclidean distances found.

This index is then compared to the original index of the projected image in the database. In this thesis, the minimum of all the Euclidean distances was found to be 1.1951e+17.

Step 6. If the index of the test projected image and the projected image of the training set happens to be the same, then there is a valid match. Otherwise, it failed the test of success.

The code lines for the matching are given below as:

```
[Euc_dist_min, Recognized_index ] = min (Euc_dist);
original_idx = img_map (Recognized_index);
if original_idx == matched_index);
    match = 1;
else
    match = 0
end
```

OutputName = strcat (char (img_idx (Recognized_index)),'.jpg'); (See appendix)

Fig. 4.17 below shows a test image that has found its match in the training set.



Fig. 4.17: A test image of Fig. 4.2 with corresponding equivalent in the training set.

This process is repeated for the rest of the images to be tested.

4.4 ALGORITHM PERFORMANCE

To calculate the accuracy rate during the testing stage, each image in the testing set is compared unto the eigenface of the training set and the index of the image with the smallest Euclidean distance is assumed to be a match, otherwise a mismatch. This process is repeated for the remaining 759 images of the test set.

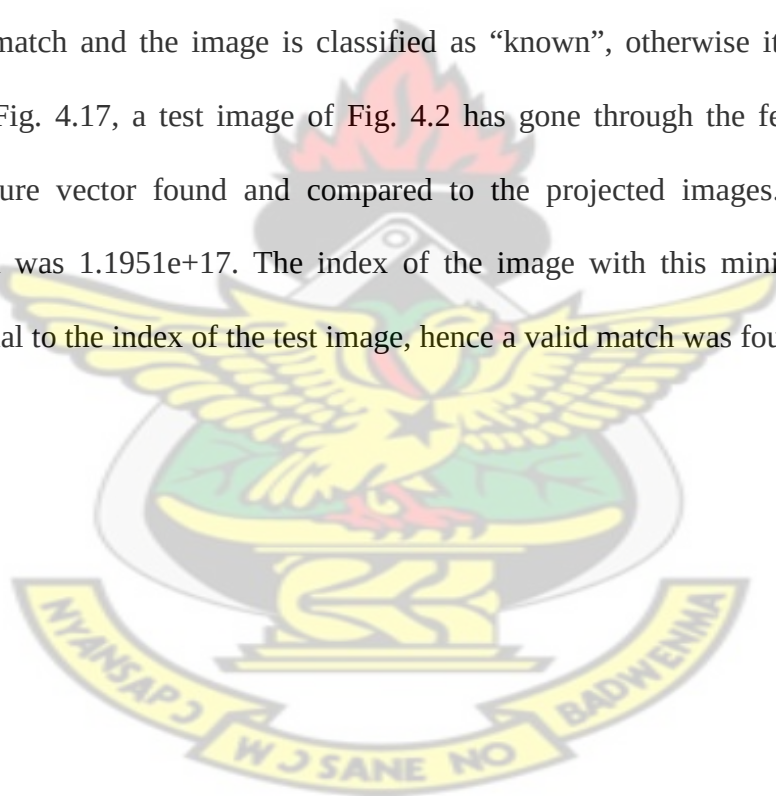
However, during matching with the test set, the algorithm is prone to errors which perhaps maybe due to the variation in the images. To check for this error of the algorithm, all the images found to match the training set are counted and their total match divided by the total number of images in the test set and their percentage is recorded.

$$\text{Thus, accuracy rate} = \frac{\text{No.of matches found}}{\text{No.of test images}} \times 100\%$$

In this thesis, an accuracy rate of 98.8158% was achieved with an error margin of 1.842%.

4.5 CONDITION FOR REJECTION

Classification of a new face image is performed by allowing the new face image to go through the feature extraction process in order to get its feature vector. The feature vector of the new test image is then compared to the feature vector of the training set. This comparison is based on the Euclidean distance between the two images. The smallest or the minimum of the Euclidean distances obtained is chosen and assumed to be the match. The index of the image with the smallest Euclidean distance is then compared with the index of the new test image. If the index of the new test image is equal to the index of the index in the training set, then there is a match and the image is classified as “known”, otherwise it is classified as “unknown”. In Fig. 4.17, a test image of Fig. 4.2 has gone through the feature extraction process, its feature vector found and compared to the projected images. The minimum Euclidean found was $1.1951e+17$. The index of the image with this minimum Euclidean distance was equal to the index of the test image, hence a valid match was found.



CHAPTER 5

CONCLUSION AND RECOMMENDATION

This chapter summarizes the research done in this thesis, together with directions for future work.

5.0 CONCLUSION

Face Recognition using Principal Component Analysis was motivated by information theory. Facial images of 152 individuals were taken from Faces 94 database which formed the data for this research. Principal component analysis, as a method was performed on the data, and a Matlab code of Matlab 2012a version was written as found in the appendix, and implemented on Toshiba.

The code applied the method, which effectively reduced the dimension of the data (matrix) from an initial 36000×3040 as found in Fig. 4.4 which is the training set in section 4.2.2, to a final 2279×2279 , the projected image in Fig. 4.16, which formed the feature vector for recognition.

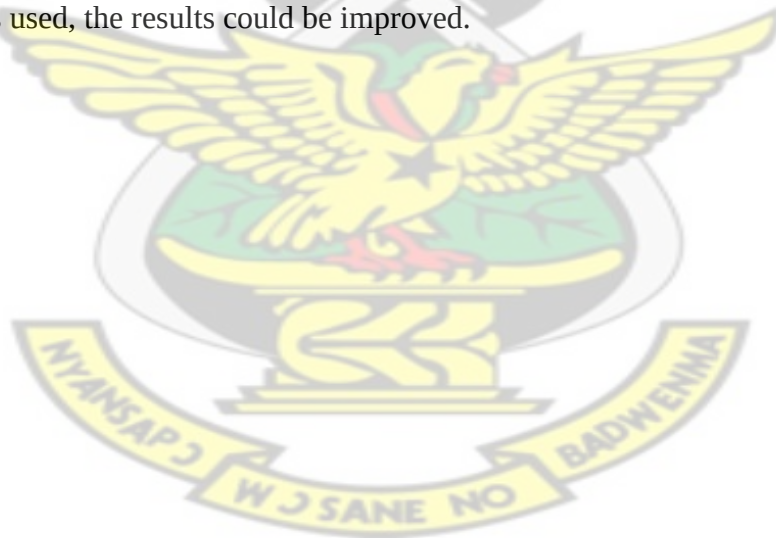
This final matrix of the projected image (Fig. 4.16) was used to form the feature vector for the recognition process. Based on the projected image, recognition was successful as the system achieved 98.8158% accuracy rate. An example of this is found in Fig. 4.17 where a test image has been successfully recognized by the system.

The eigenface approach, an application of principal component analysis to facial images, provides a practical solution that is well fitted for the problem of face recognition. It is fast, relatively simple, and works well in a constrained environment.

5.1 RECOMMENDATION

This project is based on the eigenface approach, an application of principal component analysis and gives an accuracy maximum of 98.8158%.

- The system is useful, fast and simple. I therefore recommend it to KNUST and stakeholders in various fields such as biometrics, access control, law enforcement, and security and surveillance systems.
- For the system to be effective, gadgets needs to be acquired and these include: computers, servers for storage, and cameras for taking pictures. In addition, personnel should be trained on the usage of the system in order to effectively run it.
- The whole software is dependent on the database and the database is dependent on resolution of camera. So if good resolution digital camera or good resolution analog camera is used, the results could be improved.



REFERENCES

1. Atalay, I. and Ballikaya, F. (1993), "Development of an Image Processing Environment for the 80x86 Platform", B.Sc. Thesis, Istanbul Technical University, Turkey.
2. Belhumeur, P. N., Hespanha, J. P., and Kriegman, D. J. (1997), "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, No. 7.
3. Bishop, C. M (1995), "Neural Networks for Pattern Recognition", Oxford University Press Inc., New York.
4. Bobis, C. S., Gonzalez, R. C., Cancelas, J. A., Alvarez, I. and Enguita, J. M. (1999) "Face Recognition Using Binary Thresholding for Features Extraction", IEEE, Kansas City, pages 1077-1080.
5. Bruce, V. (1999) "Identification of Human Faces", Image Processing and Its Applications, Conference Publication No. 465, IEEE, pages 615-619.
6. Brunelli, R. and Poggio, T. (1993) "Face Recognition: Features versus Templates", IEEE Transactions, Kansas City, PAMI, 15(10), pages 1042-1052.
7. Cagnoni, S. and Poggi, A. (1999) "A Modified Modular Eigenspace Approach to Face Recognition", IEEE, Kansas City, pages 490-495.
8. Cevikalp, H., Neamtu, M., Wilkes, M. and Barkana, A. (2005) "Discriminative Common Vectors for Face Recognition", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 27, No. 1, pages 4-13.
9. Chellappa, R., Wilson, C. L., and Sirohey, S. (1995), "Human and Machine Recognition of Faces: A Survey", Proceedings of the IEEE, Kansas City, Vol. 83, No. 5.

10. Choudhury, T., Clarkson, B., Jebara, T., and Pentland, A. (1999), "Multimodal person recognition using unconstrained audio and video", Proceedings, International Conference on Audio- and Video-Based Person Authentication, pages 176–181.
11. Chung, K. C., Kee, S. C. and Kim, S. R.(1999) "Face Recognition using Principal Component Analysis of Gabor Filter Responses", IEEE, Kansas City, pages 53-57.
12. Crowley, J. L. and Schwerdt, K. (1999) "Robust Tracking and Compression for Video Communication", IEEE, Kansas City, pages 2-9.
13. Cox, I. J., Ghosn, J., and Yianilos, P. N. (1996) "Feature-based face recognition using mixture distance", In Proceedings, IEEE Conference on Computer Vision and Pattern Recognition, Kansas City, pages 209–216.
14. Dai, Y. and Nakano, Y. (1998), "Recognition of Facial Images with Low Resolution using a Hopfield Memory Model", Pattern Recognition, 31(2), pages 159-167.
15. Duda, R. O., Hart, P. E. and Stork, D. G. (2001) "Pattern Classification", John Wiley & Sons, 2nd Edition.
16. Eickeler, S., Müller, S. and Rigoll, G. (1999) "High Quality Face Recognition in JPEG Compressed Images", IEEE, Kansas City, pages 672-676.
17. Er, M. J., Wu, J. S. and Lu, J. (1999) "Face Recognition Using Radial Basis Function (RBF) Neural Networks", Proceedings of the 38th Conference on Decision & Control, Phoenix, Arizona USA, IEEE, Kansas City, pages 2162-2167.
18. Etemad, K. and Chellappa, R. (1996), "Face Recognition Using Discriminant Eigenvectors", IEEE, Kansas City, pages 2148-2151.
19. Fischler, M. A. and Elschlager, R. A. (1973), "The representation and matching of pictorial structures", IEEE Trans. on Computers, Kansas City, c-22.1.
20. Guan, A. X. and Szu, H. H. (1999), "A Local Face Statistics Recognition Methodology beyond ICA and/or PCA", IEEE, Kansas City, pages 1016-1027.

21. Gutta, S., Huang, J. R. J., Jonathan, P. and Wechsler, H. (2000), "Mixture of Experts for Classification of Gender, Ethnic Origin, and Pose of Human Faces" IEEE, Trans. on Neural Networks, 11, Kansas City, pages 948-959.
22. Hong, L. and Jain A. (1998), "Integrating faces and fingerprints for person identification", IEEE Trans. on Pattern Analysis and Machine Intelligence, Kansas City, Vol. 20, No. 12, pages 1295–1307.
23. Huang, J. (1998), "Detection Strategies for Face Recognition Using Learning and Evolution", PhD. Thesis, George Mason University, Washington.
24. Kanade, T. (1973), "Picture processing system by computer complex and recognition of human faces", Dept. of Information Science, Kyoto University.
25. Kirby, M. and Sirovich, L. (1990), "Application of the Karhunen-Loeve procedure for the characterization of human faces", IEEE Transactions on Pattern Analysis and Machine Intelligence 12.
26. Kohonen, T. (1988), "Self-Organization and Associative Memory", Springer-Verlag, New York.
27. Lades, M., Vorbruggen, J. C., Buhmann, J., Lange, J., Von ser Malburg, C., Wurtz, R. P. and Konen, W. (1997), "Distortion Invariant Object Recognition in the Dynamic Link Architecture", IEEE Transactions on Computers, 42, Kansas City, pages 300-310.
28. Lawrence, S, Giles, C. L., Tsoi, A. C. and Back, A. D. (1997) "Face Recognition: A Convolutional Neural Network Approach", IEEE Trans. on Neural Networks, 8(1), pages 98-112.
29. Lee, S. J., Yung, S. B., Kwon, J. W. and Hong, S. H. (1999) "Face Detection and Recognition Using PCA", pp. 84-87, IEEE TENCON.

30. Li, B. and Chellappa, R. (2001), "Face verification through tracking facial features", J. Opt. Soc. Am.18.
31. Li, M. and Yuan, B. (2005) 2D-LDA: "A Statistical Linear Discriminant Analysis for Image Matrix", Pattern Recognition Letters, 26: pages 527-532.
32. Lin, S-H., Kung, S-Y. and Lin, J. L. (1997), "Face Recognition/Detection by Probabilistic Decision-based Neural Network", IEEE Trans. on Neural Networks, 8(1), pages 114-131.
33. Liposcak, Z. and Loncaric, S. (1999) "Face Recognition from Profiles Using Morphological Operations", IEEE, Kansas City, pages 47-52.
34. Mammone, R. J. (1993), "Artificial Neural Networks for Speech and Vision", Chapman and Hall, Cambridge.
35. Martinez, A. (1999) "Face Image Retrieval Using HMMs", IEEE, Kansas City, pages 35-39.
36. Moghaddam, B., Jebara, T. and Pentland, A. (2000) "Bayesian Face Recognition", Pattern Recognition, Vol. 33, No. 11, pages 1771-1782.
37. Moghaddam, B. and Pentland, A. (1995) "Probabilistic Visual Learning for Object Detection", The 5th International Conference on Computer Vision, Cambridge, MA.
38. Moghaddam, B., Jebara, T. and Pentland, A. (1999) "Bayesian Modeling of Facial Similarity", Advances in Neural Information Processing Systems 11, MIT Press, Massachusetts.
39. Moghaddam, B., Wahid, W. and Pentland, A. (1998) "Beyond Eigenfaces: Probabilistic Matching for Face Recognition", The 3rd International Conference on Automatic Face & Gesture Recognition, Nara, Japan.

40. Moghaddam, B. and Pentland, A. (1997) "Probabilistic Visual Learning for Object Representation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Kansas City, Vol. 19, No. 7.
41. Moghaddam, B. and Pentland, A. (1995) "An Automatic System for Model-Based Coding of Faces", IEEE, Kansas City, pages 362-370.
42. Moghaddam, B. (2002) "Principal Manifolds and Probabilistic Subspaces for Visual Recognition", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 6.
43. Pearson, K. (1901) "On lines and planes of closest fit to systems of points in space", Philosophical Magazine Series, Vol. 6, No.2, pages 559-572.
44. Phillips, P. J. (1999) "Support Vector Machines Applied to Face Recognition", Advances in Neural Information Processing Systems 11, MIT Press, Massachusetts, pages 803-809.
45. Podilchuk, C. and Zhang, X. (1996) "Face Recognition Using DCT Based Feature Vectors", IEEE, Kansas City, pages 2144-2147.
46. Samal, A. and Iyengar, P. A. (1995), "Human face detection using silhouettes", International J. Pattern Recognition, Artificial Intel. Vol. 9, No. 6.
47. Sirovich, L. and Kirby, M. (1987) "Low-dimensional Procedure for the Characterization of Human Faces", Journal of the Optical Society of America, Vol. 4, No. 3, pages 519-524.
48. Starner, B. Moghaddam, B. and Pentland, A. (1994), "View-based and modular eigenspaces for face Recognition", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR), pages 84-91.
49. Stonham T. J. (1984), "Practical Face Recognition and Verification with WISARD", In Aspects of Face Processing, pages 426-441.

50. Thomaz, C. E., Feitosa, R. Q. and Veiga, A. (1998) "Design of Radial Basis Function Network as Classifier in Face Recognition Using Eigenfaces", IEEE, pages 118-123.
51. Turk, M. A. and Pentland, A. P. (1991) "Face recognition using eigenfaces", In IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR, pages 586-591.
52. Turk, M., and Pentland, A. (1991), "Eigenfaces for recognition", Journal of Cognitive Neuroscience, Vol. 3, pages. 71 - 86.
53. Wiskott, L., Fellous, J. M., Krüger, N. and von der Malsburg, C. (1997) "Face Recognition by Elastic Bunch Graph Matching", IEEE, pages 129-132.
54. Woodward, J. D., Horn, C.J., Gatune, J. and Thomas, A. (2003) "Biometrics: A look at Facial Recognition. RAND Documented Briefing, Tech. Rep.
55. Yoshitomi, Y. Miyaura, T., Tomito, S. and Kimura, S. (1997), "Face Identification Using Thermal Image Processing", IEEE International Workshop on Robot and Human Communication, IEEE, pages 374-379.
56. Yuille, A. L., Cohen, D. S., and Hallinan, P. W. (1989), "Feature extraction from faces using deformable templates", Proc. of CVPR.
57. Zhang, J. Y., Alejandro, F. F. and Yang, J. Y. (2004) "Two-Dimensional PCA: A New Approach to Appearance-Based Face Representation and Recognition", IEEE Tran. On Pattern Analysis and Machine Intelligence, 26 (1), pages 131-137.

APPENDIX

```

Create database
function [T,img_idx,img_map] = CreateDatabase(TrainDatabasePath)
% Align a set of face images (the training set T1, T2, ... , TM )
%
% Description: This function reshapes all 2D images of the training
database
% into 1D column vectors. Then, it puts these 1D column vectors in a row to
% construct 2D matrix 'T'.
%
%
% Argument:      TrainDatabasePath      - Path of the training database
%
% Returns:       T                      - A 2D matrix, containing all 1D
image vectors.
%
% Suppose all P images in the
training database
% have the same size of MxN. So the
length of 1D
% column vectors is MN and 'T' will
be a MNxP 2D matrix.
%
% See also: STRCMP, STRCAT, RESHAPE

% Original version from Amir Hossein Omidvarnia, October 2007
% Email: aomidvar@ece.ut.ac.ir

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% File management
1 TrainFiles = dir(TrainDatabasePath);
2 Train_Number = 0;

3 for i = 1:size(TrainFiles,1)
    if
not(strcmp(TrainFiles(i).name, '.')|strcmp(TrainFiles(i).name, '..')|strcmp(T
rainFiles(i).name, 'Thumbs.db'))
        Train_Number = Train_Number + 1; % Number of all images in the
training database
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Construction of 2D matrix from 1D image vectors
faces = 15;
% faces = 1;
Train_Number = Train_Number/faces;
T = [];
counter = 1;
for i = 1 : Train_Number
    for j = 1 : faces
        code = strcat(num2str(i),'_',num2str(j));
        img_idx{counter} = code;
        img_map(counter) = i;
        % I have chosen the name of each image in databases as a
corresponding
        % number. However, it is not mandatory!
        str1 = int2str(i);
        str2 = int2str(j);
        str = strcat('\',str1,'_',str2,'.jpg');
        str = strcat(TrainDatabasePath,str);
        img = imread(str);
    end
end

```

```

        img = rgb2gray(img);

        [irow icol] = size(img);

        temp = reshape(img',irow*icol,1); % Reshaping 2D images into 1D
image vectors
        T = [T temp]; % 'T' grows after each turn
        counter = counter + 1;
    end
end

creating eigenfaces

function [m, A, Eigenfaces] = EigenfaceCore(T)
% Use Principle Component Analysis (PCA) to determine the most
% discriminating features between images of faces.
%
% Description: This function gets a 2D matrix, containing all training
image vectors
% and returns 3 outputs which are extracted from training database.
%
% Argument:      T          - A 2D matrix, containing all 1D
image vectors.
%                  Suppose all P images in the
training database
%                  have the same size of MxN. So the
length of 1D
%                  column vectors is M*N and 'T'
will be a MNxP 2D matrix.
%
% Returns:      m          - (M*Nx1) Mean of the training
database
%                  Eigenfaces - (M*Nx(P-1)) Eigen vectors of the
covariance matrix of the training database
%                  A          - (M*NxP) Matrix of centered image
vectors
%
% See also: EIG

% Original version by Amir Hossein Omidvarnia, October 2007
%                  Email: aomidvar@ece.ut.ac.ir

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Calculating the mean image
m = mean(T,2); % Computing the average face image m = (1/P)*sum(Tj's)    (j
= 1 : P)
figure; imshow(imrotate(reshape(uint8(m'),180,200),270)),title('Mean Face')
save('meanFace.mat','m')
Train_Number = size(T,2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Calculating the deviation of each image from mean
image
A = [];
for i = 1 : Train_Number
    temp = double(T(:,i)) - m; % Computing the difference image for each
image in the training set Ai = Ti - m
    A = [A temp]; % Merging all centered images
end

```

```

figure;
imshow(imrotate(reshape(uint8(A(:,1)),180,200),270)),title('Centered
Image'))
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Snapshot method of Eigenface methodos
% We know from linear algebra theory that for a PxQ matrix, the maximum
% number of non-zero eigenvalues that the matrix can have is min(P-1,Q-1).
% Since the number of training images (P) is usually less than the number
% of pixels (M*N), the most non-zero eigenvalues that can be found are
equal
% to P-1. So we can calculate eigenvalues of A'*A (a PxP matrix) instead of
% A*A' (a M*NxM*N matrix). It is clear that the dimensions of A*A' is much
% larger than A'*A. So the dimensionality will decrease.

L = A'*A; % L is the surrogate of covariance matrix C=A*A'.
figure; imshow(uint8(L)),title('Covariance Matrix')
save('covariance.mat','m')
[V D] = eig(L); % Diagonal elements of D are the eigenvalues for both
L=A'*A and C=A*A'.
figure; imshow(uint8(D)),title('EigenValue')
save('eigenvalue.mat','D')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Sorting and eliminating eigenvalues
% All eigenvalues of matrix L are sorted and those who are less than a
% specified threshold, are eliminated. So the number of non-zero
% eigenvectors may be less than (P-1).

L_eig_vec = [];
for i = 1 : size(V,2)
    if( D(i,i)>1)
        L_eig_vec = [L_eig_vec V(:,i)];
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Calculating the eigenvectors of covariance matrix
'C'
% Eigenvectors of covariance matrix C (or so-called "Eigenfaces")
% can be recovered from L's eigenvectors.
Eigenfaces = A * L_eig_vec; % A: centered image vectors
figure; imshow(uint8(Eigenfaces)),title('EigenFace')
save('eigenFace.mat','m')

recognition

function [OutputName, match] = Recognition(TestImage, m, Eigenfaces,
img_idx, img_map, ProjectedImages, input_idx)
% Recognizing step....
%
% Description: This function compares two faces by projecting the images
into facespace and
% measuring the Euclidean distance between them.
%
% Argument:      TestImage      - Path of the input test image
%
%               m               - (M*Nx1) Mean of the training
%                               database, which is output of
'EigenfaceCore' function.
%
%               Eigenfaces      - (M*Nx(P-1)) Eigen vectors of the
%                               covariance matrix of the training
%                               database, which is output of
'EigenfaceCore' function.
%

```

```

%                               A                               - (M*NxP) Matrix of centered image
%                               vectors, which is output of
'EigenfaceCore' function.
%
% Returns:      OutputName      - Name of the recognized image in
the training database.
%
% See also: RESHAPE, STRCAT

% Original version by Amir Hossein Omidvarnia, October 2007
%                               Email: aomidvar@ece.ut.ac.ir

% %%%%%%%%%%%%% Projecting centered image vectors into facespace
% % All centered images are projected into facespace by multiplying in
% % Eigenface basis's. Projected vector of each face will be its
corresponding
% % feature vector.
%
% ProjectedImages = [];
% Train_Number = size(Eigenfaces,2);
% for i = 1 : Train_Number
%     temp = Eigenfaces'*A(:,i); % Projection of centered images into
facespace
%     ProjectedImages = [ProjectedImages temp];
% end

% %%%%%%%%%%%%% Extracting the PCA features from test image
InputImage = imread(TestImage);
temp = InputImage(:,:,1);

[irow icol] = size(temp);
InImage = reshape(temp',irow*icol,1);
Difference = double(InImage)-m; % Centered test image
ProjectedTestImage = Eigenfaces'*Difference; % Test image feature vector

% %%%%%%%%%%%%% Calculating Euclidean distances
% Euclidean distances between the projected test image and the projection
% of all centered training images are calculated. Test image is
% supposed to have minimum distance with its corresponding image in the
% training database.

Euc_dist = [];
Train_Number = size(Eigenfaces,2);
for i = 1 : Train_Number
    if i ~= img_map(input_idx)
        q = ProjectedImages(:,i);
        temp = ( norm( ProjectedTestImage - q ) )^2;
        Euc_dist = [Euc_dist temp];
    end
end

[Euc_dist_min , Recognized_index] = min(Euc_dist);
original_idx = img_map(input_idx);
matched_idx = img_map(Recognized_index);
if original_idx == matched_idx
    match = 1;
else
    match = 0;
end

```

```

OutputName = strcat(char(img_idx(Recognized_index)),'.jpg');

Init stat

clear all
clc
close all
if exist('parameters.mat','file') == 2
    delete parameters.mat
end

% You can customize and fix initial directory paths
TrainDatabasePath = 'D:\New Folder\BA ACCOUNTANT\ind maths\give Silas
Code\FINAL_DATASET\Copy of SOURCE';

[T,img_idx,img_map] = CreateDatabase(TrainDatabasePath);
[m, A, Eigenfaces] = EigenfaceCore(T);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Projecting centered image vectors into facespace
% All centered images are projected into facespace by multiplying in
% Eigenface basis's. Projected vector of each face will be its
% corresponding
% feature vector.

ProjectedImages = [];
Train_Number = size(Eigenfaces,2);
for i = 1 : Train_Number
    temp = Eigenfaces'*A(:,i); % Projection of centered images into
    facespace
    ProjectedImages = [ProjectedImages temp];
end

% clear

msgbox('Data feature extraction FINISHED. EXIT WINDOW!','SILAS FACE
RECOGNITION')

final run

% A sample script, which shows the usage of functions, included in
% PCA-based face recognition system (Eigenface method)
%
% See also: CREATEDATABASE, EIGENFACECORE, RECOGNITION

% Original version by Amir Hossein Omidvarnia, October 2007
%
% Email: aomidvar@ece.ut.ac.ir

clear all
clc

% You can customize and fix initial directory paths

```

```

% TestDatabasePath = '/home/appatech/Documents/OTHER PROJECT
DOCUMENTATION/MSc Salas_300/Matlab Code/FINAL_DATASET/FRR';
TrainDatabasePath = 'D:\New Folder\BA ACCOUNTANT\ind maths\give Silas
Code\FINAL_DATASET\Copy of SOURCE';
TestDatabasePath = 'D:\New Folder\BA ACCOUNTANT\ind maths\give Silas
Code\FINAL_DATASET\test';

train_data = load('parameters.mat');
train_data2 = load('parameters2.mat');
m = train_data.m;
A = train_data.A;
Eigenfaces = train_data.Eigenfaces;
img_idx = train_data.img_idx;
img_map = train_data.img_map;
ProjectedImages = train_data2.ProjectedImage;
n = 15;

input_idx = 1;
mTest=(5*length(img_idx)/n);
match = [];
% for i = 1:length(ProjectedImages)/n
for i = 1:length(img_idx)/n
    for j = n+1:n+5
        TestImage = strcat(TestDatabasePath,'\ ',
num2str(i), '_ ', num2str(j), '.jpg');
        im = imread(TestImage);

        [OutputName, matchVal] = Recognition(TestImage, m,
Eigenfaces, img_idx, img_map, ProjectedImages, input_idx);

        SelectedImage = strcat(TrainDatabasePath, '\ ', OutputName);
        SelectedImage = imread(SelectedImage);

        subplot(1,2,1);
        subimage(im); axis off;
        title('Test Image')

        subplot(1,2,2);
        subimage(SelectedImage); axis off;
        title('Equivalent Image')

        str = strcat('Matched image is : ', OutputName);
        disp(str)
        match = [match matchVal];
        input_idx = input_idx + 1;
    end
end
match = mTest-sum(match);
Accuracy = match/mTest*100;
Error = (100-Accuracy);

disp(['Accuracy Rate: ', num2str(Accuracy), ' and Error
Margin: ', num2str(Error)])

```