NEAR EAST UNIVERSITY

GRADUATE SCHOOL OF APPLIED SCIENCES

PCA- AND LDA-BASED FACE RECOGNITION USING REGION-DIVISION WITH MAJORITY VOTING

Mazen Eleyan

Master's Thesis

Department of Computer Engineering

Nicosia – 2008

ABSTRACT

Face databases that contain obstacles that occlude the natural features of the face pose a special challenge for face recognition algorithms. In this thesis, our goal is to investigate whether dividing the face into regions would perform better than the classic approach of using the whole face. For a more comprehensive comparison, we created our own face image database in addition to the standard ORL database that contains 15 persons with 10 images per person. In our region-division approach, the goal is to use each part of the face as a separate feature and then combine the recognition results of all regions using majority voting to decide on the best match. In our approach, we divide each face image into five horizontal regions without removing the background. We introduce two region-division approaches. One of these approaches uses these five regions of the face with majority voting. The other uses the five regions of face and the whole face as a sixth region with majority voting. We tested these two techniques against the classic approach using both PCA and LDA feature extraction algorithms. In order to test these region-division approaches, we implemented a software system that generates sets of ten experiments randomly, and, we used the unpaired t-test to compare the results. The performance of our region-division approaches did not perform any better than the classic approach using PCA on our face image database and using both PCA and LDA on the ORL database. However, with LDA, the five-region and six-region approaches with majority voting outperformed the classic approach on our face image database with 99% statistical confidence.

Dedicated to my parents, my dear wife, and my daughter …

ACKNOWLEDGMENTS

First of all, I would like to thank my advisor Assist. Prof. Dr. Huseyin Sevay, who has been very helpful to me throughout the preparation of this thesis, starting with helpful and valuable suggestions, and ending up with the preparation of this thesis in LaTeX format. A great deal of thanks goes to my father and mother, for their endless support and encouragement, even through the most demanding phases of this research. Thanks to my fifteen fellow students who helped me in the preparation of my own face image database. And finally, a special thanks goes to my wife for her enormous support and understanding throughout the past few years.

CONTENTS

LIST OF ABBREVIATIONS

PCA	Principle Component Analysis
LDA	Linear Discriminant Analysis
ORL	Olivetti Research Laboratories
SVM	Support Vector Machines
NN	Neural Networks
ED	Euclidean Distance
NEU	Near East University
NFID	NEU Face Image Database

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Overview

This thesis addresses the issue of developing a face recognition system that uses two region-division approaches in order to achieve a higher recognition rate especially in cases where the facial features of person are modified by wearing a different hair styles or wearing glasses. One of the region-division approaches uses the five regions of the face. The second approach uses the five regions of the face but also considers the whole face as a sixth region, and it compares the performance of this approach using Principle Component Analysis (PCA) and Linear Discriminant Analysis (LDA).

In order to recognize a face, the approach presented in this thesis divides each face into five horizontal parts, applies either the PCA or the LDA method to each part, and uses the majority voting scheme to decide on the best match. In order to better demonstrate this region-division approach, we created a new face database that we will refer to as the NEU Face Image Database (NFID) in the remainder of this thesis. This new face database contains different types of occlusions of the face so that that the database poses a challenge for face recognition compared to a database that does not contain similar occlusions of the face.

Experimental results show that, for small number of classes, the PCA approach is a reliable and robust face recognition method. However, the LDA approach is more successful than the PCA approach with large face databases. In particular, using the entire face, the recognition rate with different number of training and test cases were used the classic approach with either the PCA or LDA feature extraction methods.

The recognition rates when only a single training image and 9 test images with PCA was 49.4% and with LDA the recognition rate was 18.3%. Using the five-region approach with majority voting described in this thesis, the PCA-based recognition rate was 39.9% and the LDA-based recognition rate increased to 20.9%. Using the six region approach with majority voting, the PCA-based recognition rate was 44.0%, and the LDA-based recognition rate was only 21.9%. Since only a single training image was used, these results were expected.

When 5 training images and 5 test images were used, the recognition rate for the classic whole face approach with PCA was 65.2%, and, with LDA, the recognition rate was 78.5%. Using the five region approach with majority voting, the PCA-based recognition rate increased to 67.3%, and the LDA-based recognition rate increased to 85.6%. Using the six-region approach with majority voting, the PCA-based recognition rate increased to 69.5%, and the LDA-based recognition rate increased to 88.3%.

When 9 training images and a single test image were used for testing, the recognition rate for the classic whole face with PCA was 71.3, and, with LDA, it was 78.7%. Using the five-region approach with majority voting, the PCA-based recognition rate increased to 78.0%, and the LDA-based recognition rate increased to 94.0%. Using the six-region approach with majority voting, the PCA-based recognition rate increased to 81.3%, and the LDA-based recognition rate increased to 95.3%.

The LDA experiments run using 5 training image and 5 test images and also using 9 training images and one test image showed that the region-division with majority voting approach reported in this thesis performed statistically better than the classic whole face approach with 99% confidence.

Face recognition is one of the most popular problems in the field of image analysis and understanding. Identifying a person from an unknown face is usually done by comparing the unknown face with the known faces from a face database.

The interest of researchers and engineers in the face recognition problem has grown rapidly in recent years since there is a wide range of commercial and law enforcement applications on face recognition [Zhao et al., 2003, Chellappa et al., 1995]. The increasing need for surveillance-related applications, especially due to drug traffic and terrorist activities has a great impact on the growth of the interest in the field of face recognition.

There are several application areas of face recognition in our real life such as identification of personnel using credit cards, passport checks, entrance control, criminal investigations, and so on. There exist a number of well-known appearance-based statistical methods applied for the solution of the face recognition problem. Eigenfaces based on Principle Component Analysis (PCA) is one of the first of these methods used for dimensionality reduction in compression and recognition problems. Alex P. Pentland and Mathew A. Turk of MIT suggested the Eigenfaces methodology in 1991 [Turk and Pentland, 1991].

The main idea in the Eigenfaces approach is to get the features of the face in the mathematical sense instead of the physical face feature using mathematical transforms. The eigenface algorithm finds the principle components of the covariance matrix of a set of face images. These vectors can be thought as a set of features, which together characterize the variations between face images. Each face image in the training set can be represented exactly in terms of a linear combination of the Eigenfaces. More recently, Nayar, Nene and Murase used eigenspace projection to identify objects using a turntable to view objects at different angles [Nayar et al., 1996]. Finlayson, Dueck, Funt and Drew extended grayscale Eigenfaces to color images [Finlayson et al., 1996]. Their experimental results have proven that the Eigenface method is a reliable and robust face recognition method [Nayar et al., 1996, Finlayson et al., 1996].

Another well-known approach to face recognition is the Linear Discriminant Analysis (LDA), which is also known as Fisher's Discriminant Analysis. Fisher developed the Fisher's Linear Discriminant (FLD) in the 1930's, but not until recently have Fisher Discriminants been utilized for object recognition [Fisher, 1936]. This algorithm is very similar to the Eigenface algorithm. Fisherfaces are complex modification of the Eigenfaces that maximize the ratio of the determinant of the between scatter matrix of the projected samples to the determinant of the within-class scatter matrix.

Swets and Weng used FLD to cluster images for identification in 1996 [Swets and Weng, 1996]. Also in 1997, Belhumeur et al used FLD to identify faces, by training and testing some faces under different lighting [Belhumeur et al., 1997]. The performance of their algorithm has proven that the LDA approach is more successful than the previous algorithms that have been used for face recognition.

1.2 Contribution

This thesis contributes the following:

- A face recognition method that considers a face region by region and applies majority voting to pick the best match.
- A comparison of the performance of the PCA and LDA feature extraction approaches between the classic whole face approach and the region-division approaches described in this thesis.
- A new face image database of a total of 150 face images belonging to 15 different persons (all males) that contains challenges for the classic approach that considers each face image as a whole.

1.3 Thesis Organization

The remainder of this thesis is organized as follows:

- *Chapter 2* describes the fundamentals of pattern and face recognition, which the work presented in this thesis is based upon. It also discusses work closely related to the work to the methodology described in this thesis.
- *Chapter 3* describes the methodology in this thesis for face recognition. This methodology is based upon applying PCA and LDA methods alternatively after dividing the face into five regions and then combining the recognition results from each region using majority voting.
- *Chapter 4* describes the face image databases used in the experiments, namely the ORL database and the NEU Face Image Database (NFID) created for the work reported in this thesis.
- *Chapter 5* presents the simulation results of the face recognition methodology described in this thesis and the comparison of this methodology using both PCA and LDA. This chapter also provides a discussion of these results.
- *Chapter 6* presents the conclusions and future work.

CHAPTER 2

BACKGROUND

2.1 Overview

Pattern recognition is a mathematically rigorous field with the purpose of classifying objects into one of a number of *classes*. These objects are generically termed patterns and include printed characters, speech waveforms, textures, images, and anything else one wishes to classify. The pattern recognition process is generally implemented in a manner that allows automatic recognition without human intervention. For example, a system may tell the credit card company which transactions are likely the results of unauthorized credit card use. Important application areas are image analysis, character recognition, speech analysis, man and machine diagnostics, person identification reading machine for the blind, searching for meaningful patterns, and industrial inspection.

2.2 Pattern Recognition

There are two types of features, namely, *holistic features*, where each feature is a characteristic of whole face, and *partial features* such as hair, nose, eyes, and mouth. Holistic features focus on detecting individual features such as eyes, mouth, nose and head outline, and defining face model by position, size and relationships between these features.

Wilson made the first attempt to semi-automated face recognition with a human computer system that classified faces on the basis of facial marks entered on the photographs by hand [Bledsoe, 1966]. Parameters for classification were normalized distances and ratios among points such as eye corners, mouth corners, nose tip and chin point. Later work at Bell Labs led to the development of a vector of up to 21 features, and the system created recognized faces using standard pattern classification.

The Tanaka approach improved template matching approach which measures similar features automatically, describe linear algorithm that use local template matching and global matching of fit to find and measure facial feature [Tanaka et al., 2000]. Kamel et al described a system for content-based retrieval of facial images from an image database [Kamel et al., 1993]. This system includes feature extraction based on expert-assisted feature selection, spatial feature measurement, feature and shape representation, feature information compression and organization, search procedures, and pattern-matching techniques. The system uses novel data structures to represent the extracted information. These structures include attributed graphs for representing local features and their relationships.

The approach in [Burt, 1988] is based on multi-resolution template matching using a special-purpose computer built to calculate multi-resolution pyramid images quickly, and this approach has been demonstrated to identify people in near real-time.

[Kotani et al., 2006] used a vector quantization histogram method for recognition. In this approach, the face is divided into different regions, and the histograms resulting from vector quantization are used to recognize faces.

Rose emphasizes the work of Nobel Laureate and Herbert Simon whose central finding was that pattern recognition is critical in most human decision making tasks ”The more relevant patterns at your disposal, the better your decisions will be.” [Rose, 1998] Recently Anil et al introduced pattern recognition as the best way of utilizing available sensors, processors and domain knowledge to make decisions automatically [Jain et al., 2000].

Construction of a pattern recognition system involves learning from a set of example patterns. This learning process has two forms: *supervised* and *unsupervised*. If the classes of the example patterns are already known, the learning process is termed as *supervised pattern recognition*. That is, the correct classification of an individual pattern is used to evaluate the performance of the system. On the other hand, if the classes are not known a priori, then an *unsupervised pattern recognition* system must not only produce a classification procedure but also define the classes themselves.

In general, pattern recognition is performed in two phases. The first phase involves the extraction of features, where observation X of a pattern is transformed into a vector Y, whose components are called *features*. Y is generally more tractable for the system than X, but it needs to contain most of the information necessary for classification of the patterns.

The second phase of pattern recognition involves the classification of the feature vectors. A *classifier* partitions the feature space of Y into disjoint regions, each corresponding to a pattern class. If the feature vector of a specific observation lies in the region R_k, the observation is assigned to class C_k. Thus, the partition specifies the class membership of the observations. Constructing the classifier in a supervised pattern recognition system is relatively simple, because the class memberships of a set of example patterns are already known. This feedback knowledge can then be used to both train and test the classifier. If the class membership is unknown, however, construction is much more difficult and will involve clustering.

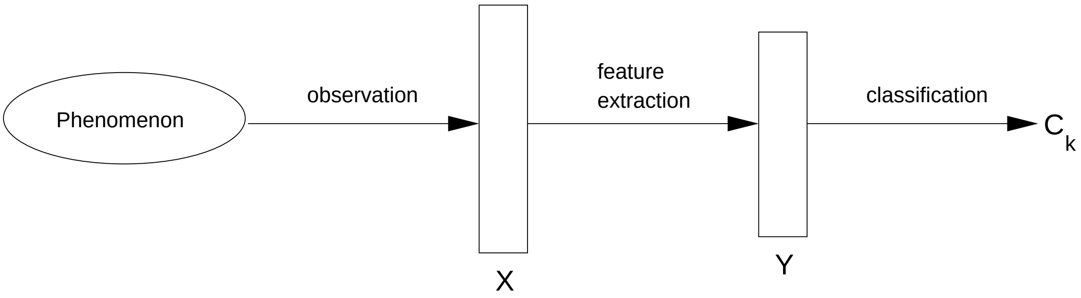


Figure 2.1: Pattern recognition process

Figure 2.1 illustrates the general pattern recognition process. First, an observation vector X is recorded while observing a phenomenon. Feature extraction from X produces a smaller vector Y that encapsulates the salient features of the original observation. Finally, classification determines the best class label C_k that may be assigned to Y.

2.3 Clustering

Clustering is the process of constructing a classifier for unsupervised pattern recognition. Here, the problem is not only to classify the given data, but also, at the same time, to define the classes themselves. In the general sense, clusters are defined as groups of similar points according to some measure of similarity. Usually, similarity is defined as proximity of the points as measured by a distance function, such as the Euclidean distance, of feature vectors in the *feature space*. However, measures of other properties, such as vector direction, may also be used. The method of finding the clusters may have a heuristic basis or may be dependent on minimization of a mathematical clustering criterion.

In the field of digital signal processing, vector quantization is a clustering method that uses the Euclidean distance measure. However, many new terms are used. Furthermore, the distance of each sample to the mean of its enclosing cluster is no longer a measure of similarity but rather a measure of distortion. The goal of vector quantization is to find the set of quantization levels that minimizes the average distortion over all samples.

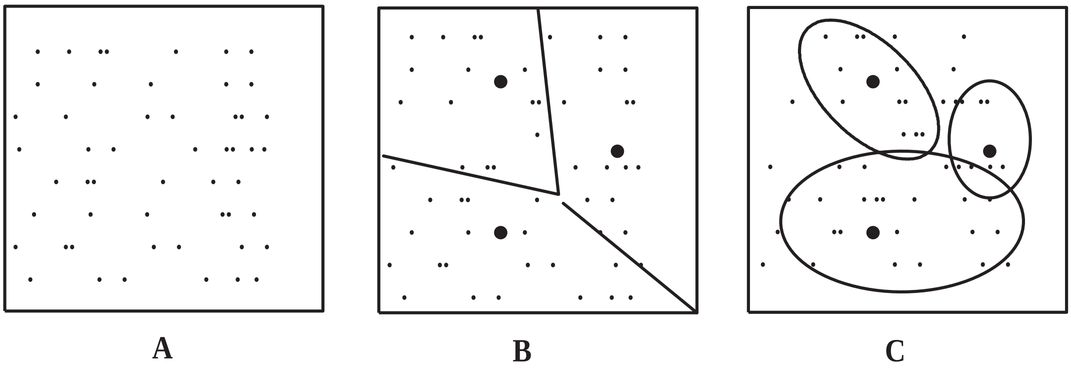


Figure 2.2: Clustering of data in pattern recognition where (A) is the collection of data, (B) is the vector quantization showing the associated mean, and (C) is the Gaussian distribution of the clusters.

Figure 2.2 illustrates the clustering process. First, all the data are collected into a single set of observation vectors. Next, vector quantization groups the data points into clusters. Finally, assuming a Gaussian distribution in each cluster, the estimated mean and covariance of each cluster are computed.

2.4 Face Recognition

Face recognition is a type of pattern recognition task where a face classified as either ″known″ or ″unknown,″ after comparing that face with the images of known individuals stored in the database. Since there is variability in data due to the random variation across people as well as the systematic variations due to pose, lighting conditions, and so on, face recognition becomes a challenging task.

Computational models of face recognition must address several difficult problems. These difficulties arise from the fact that faces must be represented in a way that best utilizes the available face information to distinguish a particular face from all other faces in the database. Face pose is a particularly difficult problem in this respect because all the faces look like each other, since they contain two eyes, a mouth, a nose, and other features located at about the same place.

Figure 2.3 shows the main computational stages of a face recognition system.

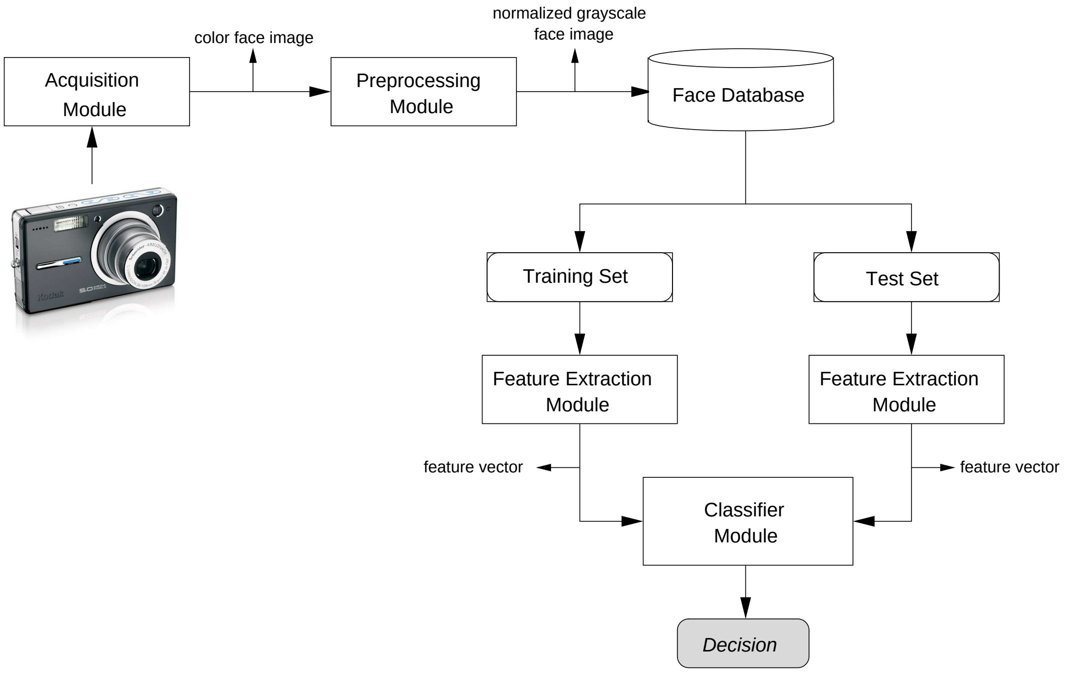


Figure 2.3: The main computational stages of a face recognition system (modified from a figure in [Eleyan, 2004])

The system starts with the Acquisition Module where images are captured with a digital camera or any image capturing device. In the second phase, captured images are sent through the Preprocessing Module to meet the standards required by a given recognition system. The Preprocessing Module may perform tasks such as color-to-grayscale conversion, image resizing, and illumination and background removal in order to *normalize* the input image. Then the normalized images are added to the Face Database. Some of the images in the face database are used as the Training Set of the system and the rest will be the Test Set. The Feature Extraction Module takes as input a normalized image and outputs a mathematical model of that input image that expresses the most important features in that image, thereby reducing its dimensionality. For example, techniques such as

Principal Components Analysis (PCA) and the Linear Discriminant Analysis (LDA) can be used as feature extractors. Finally, the Classifier Module compares the feature vectors between a test image and all the training images and decides which training image is closest to that test image.

2.4.1 Major Computational Stages of A Face Recognition System

A face recognition system comprises the following modules:

- **Acquisition Module:** This is the entry point of the face recognition process. It is the module where the face image under consideration is presented to the system. An acquisition module can acquire a face image from several different environments.
- **Preprocessing Module:** By means of early vision techniques, face images are normalized and enhanced to improve the recognition performance of the system. The following preprocessing steps can be implemented in a face recognition system:
    - *Image size normalization:* Because the Principal Components Analysis (PCA) and the Linear Discriminant Analysis (LDA) involve multiplication of arrays, it is important to normalize the size of all images. This is done by resizing all images to a default image size such as 112×92 pixels as in the ORL database we use in this work to guarantee that information about the eyes, nose, and mouth is not lost in potentially small versions of images.
    - *Illumination normalization:* The general purpose of illumination normalization is to decrease lighting effect when the observed images are captured in different lighting environments. A common approach is to adjust observed images to approximate the ones captured under a standard lighting condition.
    - *Histogram equalization:* Histogram equalization is a process of adjusting the image so that each intensity level contains an equal number of pixels such that the appearance of the image is improved by balancing light and dark areas.
    - *Median filtering:* Median filtering is used to reduce the noise in an image.
    - *High-pass filtering:* High-pass filtering removes low frequency gradients in the image, without affecting high frequency gradients.

  - *Background removal:* Removal of the background is used so that only the face can be considered.
  - *Translational and rotational normalization:* It is possible to work on a face image in which the head is shifted or rotated. The head plays a very important role in the determination of facial features especially for face recognition systems that are based on the frontal views of faces. It may be desirable that the pr-processing module determine and normalize the shifts and rotations in the head position.

- **Feature Extraction Module:** After preprocessing, it uses the normalized face image to find the key features that are going to be used for classification.
- **Classification Module:** Given the test face image and a set of preprocessed training images from a face database, the Classification Module performs a comparison based on a distance metric and decides on the class of that test image.

2.4.2 Difficulties of Recognition and Identification

Facial features used for face recognition need to remain invariant to factors unrelated to a person's identity in order for a face to be a source of useful biometrics. Existing practice suggests these factors are difficult to handle, and yet the exact source of difficulties remains unclear. The literature concentrates on specific difficulties that influence face image acquisition and individual differences in subjects [Gross et al., 2001].

- **Viewing angle**

  The face has a 3-dimensional shape. As the camera pose changes, the appearance of the face can experience projective deformation, which leads to stretching and foreshortening of different part of face. Camera pose changes may also cause self occlusion and disocclusion of parts of the face. If faces are only seen from one viewing angle, in general, it may be difficult to recognize them from different angles.

- **Illumination**

  Illumination variation is an inevitable difficulty in face recognition. Ambient lighting is likely to change greatly during the same day or from day to day, and it may change between indoors and outdoors. Due to the 3-dimensional shape of the face, direct lighting can cast dark shadows that accentuate or diminish some facial features.

- **Expression**

  The face is a non-rigid object, and different expressions can cause the appearance of a face to change. Given that the number of possible changes in facial expression is in the thousands, a good recognition system needs to be able to handle a range of changes in the same face well. Yet, the influence of facial expression on recognition is not yet a well-understood subject.

- **Occlusion**

  The face may be occluded by other objects in the environment or by sunglasses or even hair, for example. However, we would like a recognition system to be able to recognize individuals that may intentionally or intentionally occlude their faces from a complete view.

- **Time Effects**

  Faces change over time due to aging, hair style, makeup, muscle tension, presence or absence of facial hair, glasses, or other facial jewelry, and a face recognition system must adjust to these changes as best as it can.

- **Individual Factors**

  Day-to-day changes in mood and use of makeup may potentially effect recognition. An algorithm may be more or less sensitive for men or women or members of different ethnic groups. Practically, females may be more difficult to recognize because of greater use and day-to-day variation in makeup or hair style. Male and female faces differ in both local features and in shape. Men's faces tend to have thicker eyebrows and greater texture in the beard region. In women's faces, the distance between the eyes and brows is greater, the protuberance of the nose smaller, and the chin is narrower than in men. It is hard to identify, however, the sensitivity of specific algorithms to these factors.

2.5 Fundamental Problems in Pattern Recognition

The design of an automatic pattern recognition system requires great attention to several issues. The first issue has to do with dealing with the representation of the input data, which can be measured from the objects to be recognized. Each measured quantity describes a characteristic of the pattern or object. In other words, a pattern vector that describes the input data has to be formed. The pattern vectors contain all the measured information available about the patterns. The set of patterns belonging to the same class corresponds to an ensemble of points scattered within some region of the measurement space.

The second issue in pattern recognition systems concerns the extraction of characteristic features from the received input data and the dimensionality reduction of pattern vectors. This is referred as the feature extraction problem. The elements of intraset features which are common to all pattern classes under consideration carry no discriminatory information and can be ignored. If a complete set of discriminatory features for each pattern class can be determined from the measured data, the recognition and classification of patterns will present some difficulty. Automatic recognition may be reduced to a simple matching process. However, in most pattern recognition problems that arise in practice, the determination of a complete set discriminatory features is extremely difficult.

The third issue in pattern recognition system design involves the determination of the optimum decision procedures, which are needed in the identification and classification process. After expressing the observed data in the form of measurement vectors in the pattern space, we want the machine to decide to which pattern class these data belong. Let the system be capable of recognizing N different pattern classes. Then the pattern space can be considered as consisting of N regions, each of which encloses the measurement of a class. The recognition problem can now be viewed as that of generating the decision boundaries that separate the N pattern classes on basis of the observed measurement vectors. These decision boundaries are generally determined by decision functions.

2.6 Sources of Variation

The reasons for the image of a face being different can be divided into two categories: Intrinsic variation and extrinsic variation. Intrinsic variation is independent of any observer and due to physical nature of face only. Extrinsic variation, which is dependent of observer and on interaction of light with face (image dependent on external factors).

The intrinsic sources of variation include identity, expression, speech, gender, and age. Non-rigid body motions due to speech or facial expression are peculiar to an individual and not due to any external factors and are therefore intrinsic. Examples of extrinsic sources of variation are viewing geometry (pose), illumination (shading, self-shadows, color, specular highlights), imaging process (resolution, focus, imaging noise, and perspective effects), effects of other objects (occlusion, shadowing, and indirect illumination). Pose is one of the most significant sources of variation. As pose is determined by relative three-dimensional position and orientation of the observer, it is extrinsic.

2.7 Recent Applications of Face Recognition

Yan and Osadciw proposed a method that fuses the classification results from the components of the face (eyes, nose and mouth) and from the whole face image, instead of concatenating the face feature and the modular features for a single classifier [Yan and Osadciw, 2007]. Yang and Wang present a new face recognition system based on Local Binary Patterns with the Hamming distance constraint [Yang and Wang, 2007]. Yang and Liu present a General Discriminant Model (GDM) for color face recognition. The GDM model involves two sets of variables, namely a set of color component combination coefficients for color image representation and a set of projection basis vectors for image discrimination [Yang and Liu, 2007]. Hemery et al analyze the benefit and the limitations of using biometrics in order to replace the PIN code authentication for banking [Hemery et al., 2008]. Sirlantzis et al present two methods for creating multiple classifier systems based on an initial transformation of the original features to the binary domain and subsequent quantization [Sirlantzis et al., 2008].

2.8 Summary

This chapter presented the background of face recognition from the perspective of the work reported in this thesis. This chapter also described the main computational stages of a face recognition system and the difficulties of recognition and identification in general. Work closely related to our study will be discussed in detail in the next chapter.

CHAPTER 3

METHODOLOGY

3.1 Overview

The purpose of pattern recognition is to classify objects into one of a number of classes. These objects are generically termed patterns and may include printed characters, speech waveforms, textures, or images. Pattern recognition is usually implemented such that a recognition system can work without human intervention. For example, a system may tell a credit card company which transactions are likely the results of unauthorized use. Important application areas of pattern recognition are image analysis, character recognition, speech analysis, man and machine diagnostics, person identification development of reading machines for the blind, searching for meaningful patterns, and industrial inspection.

Many algorithms and approaches exist for solving the problem of human face recognition. In this thesis, our goal is to develop a system that can recognize faces whose appearance changes according to different factors such as hair style or what is worn by a person such as glasses and hats, especially for cases where the treatment of the whole face may fail to produce correct recognition. Instead of considering the whole face at once, in our approach, we divide the face into five horizontal regions, and apply a near-complete recognition on each region except for the final decision, and consider the recognition result from each facial region as a vote for the corresponding person. Finally, the person who receives the highest vote is chosen as the best match.

To provide a perspective on the angle of our region-division approach that uses majority voting, we compare the recognition performance of two techniques, namely the

Principle component analysis (PCA) and Linear Discriminant Analysis (LDA).

3.2 Principle Component Analysis (PCA)

Principle Component Analysis (PCA) is a dimensionality reduction technique that is used for image recognition and compression. It is also known as Karhunen-Loeve transformation (KLT) or eigenspace projection [Turk and Pentland, 1991]. The main goal of PCA is dimensionality reduction. Therefore, the eigenvectors of the covariance should be found in order to reach the solution. The eigenvectors correspond to the directions of the principle components of the original data, and their statistical significance is given by their corresponding eigenvalues.

The first introduction of a low-dimensional characterization of faces was developed at Brown University by Kirby and Sirovich in 1987 [Kirby and Sirovich, 1990]. This was later extended to eigenspace projection for face recognition by Pentland, Turk, Moghaddam and Starner at the Vision and Modeling Group of MIT in 1994 [Pentland et al., 1994]. More recently, Nayar et al used eigenspace projection to identify objects using a lookup table to view objects at different angles [Nayar et al., 1996]. Finlayson et al extended grayscale eigenfaces to color images [Finlayson et al., 1996].

3.2.1 Calculating Eigenfaces

Consider an $N \times N$ face image $\Gamma(x, y)$ as a vector of dimension N^2, so the image can be thought as a point in N^2 dimensional space. A database of M images can therefore be mapped to a collection of points in this high-dimensional "face space" as $\Gamma_1, \Gamma_2, \Gamma_3, \ldots, \Gamma_M$.

To compute Eigenfaces, first an average of all training images needs to be computed. The average image Ψ is computed using *Equation 3.1*:

$$\Psi = \frac{1}{M} \sum_{n=1}^{M} \Gamma_n \tag{3.1}$$

Each image Γ_i differs from the average image Ψ by the vector $\Phi_i = \Gamma_i - \Psi \quad \text{where} \quad i = 1, 2, \ldots, M$. The covariance matrix C of the data set is defined by *Equation 3.2*.

$$C = \frac{1}{M} = \sum_{n=1}^{M} \Phi_n \Phi_n^T = AA^T \tag{3.2}$$

where the matrix $A = [\Phi_1, \Phi_2, \ldots, \Phi_M]$. The matrix C has a dimension of $N^2 \times N^2$ eigenvectors and eigenvalues, and, for typical image sizes, this size would be a very high value. Therefore, we need a computationally feasible method to determine these eigenvectors.

If the number of data points in the image space is less than the dimension of space $M < N^2$, there will be only $(M-1)$ meaningful eigenvectors, and the rest of the eigenvectors will have eigenvalues of zeros.

Consider eigenvectors ν_i of $A^T A$ such that

$$A^T A \nu_i = \mu_i \nu_i \tag{3.3}$$

Multiplying both sides by A, we have

$$A A^T A \nu_i = \mu_i A \nu_i \tag{3.4}$$

From *Equation 3.4*, we can see that $A\nu_i$ are the eigenvectors of $C = AA^T$. Then we can construct a $M \times M$ matrix:

$$L = A^T A \tag{3.5}$$

Then we can find the M eigenvectors ν_i of L. These vectors determine linear combinations of the M training set face images to form Eigenfaces u_i.

$$u_i = \sum_{k=1}^{M} \nu_{ik} \Phi_k \tag{3.6}$$

Eigenvectors u_i are in fact images, and they are called *eigenfaces*, and the eigenvectors with the highest eigenvalues are more useful than the rest. Therefore those $M^I < M$ eigenfaces that are most significant are used for constructing the "face subspace" for projections that are used in identifying or classifying images.

Using Eigenfaces for Projection and Classification

In order to identify an input image, first of all, that image needs to be transformed into the face space by taking inner products with eigenfaces. These inner products compose a vector, which is the representation of input image in face space. Then we compare this vector with those of known images using Euclidean distance. The input test image is then classified as the closest match.

A new face image (Γ) is transformed into its eigenface components (i.e., projected into "face space") using the following operation:

$$\omega_k = u_k^T(\Gamma - \Psi) \quad k = 1, 2, \ldots, M^I \tag{3.7}$$

where ω_k is the k-th coordinate of the Φ in the new "face space." The above equation describes point-by-point image multiplication and summation, resulting in the scalar value (dimension 1×1) defined as a *weight* that describes each face. The weights form a feature vector given by *Equation 3.8*:

$$\Omega^T = [\omega_1, \omega_2, \ldots, \omega_M] \tag{3.8}$$

Vector Ω^T in *Equation 3.8* describes the contribution of each eigenface in representing the input test face image, where the eigenfaces represent the basis set for face images. This vector is used in finding the class that the input image belongs to, if there are more than one image describing a person, otherwise it is used to find to which single image is most similar to the test image. This decision is reached by computing the Euclidean distance between input image and training images in the face database (i.e., class projections), as given in *Equation 3.9*.

$$\varepsilon_k = \left\|\Omega - \Omega_k\right\|^2 \tag{3.9}$$

where Ω_k describes k-th face class, which is the average of the eigenface representation of the face images for each individual in the training set. A face will be classified as belonging to some class if the minimum ε_k is below some specified threshold. Otherwise, that image will be classified as unknown face.

3.2.2 A PCA Example

Identifying images through eigenspace projection takes three basic steps. First, the eigenspace must be created using training images. Second, the training images are projected into the eigenspace. Finally, the test images are identified by projecting them into the eigenspace and comparing them to the projected training images.

- **Create an Eigenspace**

  The following steps create an eigenspace:

  1. **Center data:** Each of the training images must be centered. To do so, the average (mean) image is subtracted from each of the training images. The mean image is a column vector such that each entry is the mean of all corresponding pixels of all training images.
  2. **Create a data matrix:** Once the training images are centered, they are combined into a data matrix of size $N \times P$, where P is the number of training images and each column is a single.
  3. **Create covariance matrix:** The data matrix is multiplied by its transpose to create a covariance matrix.
  4. **Compute the eigenvalues and eigenvectors:** The eigenvalues and corresponding eigenvectors are computed for the covariance matrix.

     $$\Omega V = \Lambda V$$

     where V is the set of eigenvectors associated with the eigenvalues Λ.
  5. **Order eigenvectors:** Order the eigenvectors $\nu_i \in V$g according to their corresponding eigenvalues λ_ig $\in$ gΛg from high to low. Keep only the eigenvectors associated with non-zero eigenvalues. This matrix of eigenvectors is the eigenspace V, where each column of V is an eigenvector.

     $$V = \nu_1 | \nu_2 | \ldots | \nu_P$$

- **Project Training Images**

  Each of the centered training images $\overline{x^i}$ is projected onto the eigenspace. To project an image onto the eigenspace, calculate the dot product of the image with each of the ordered eigenvectors.

$$\widehat{x^i} = V^T \overline{x^i} \tag{3.10}$$

Therefore, the dot product of the image and the first eigenvector will be the first value in the new vector. The new vector of the projected image will contain as many values as eigenvectors.

- **Identify Test Images**

  Each test image is first mean centered by subtracting the mean image, and is then projected into the same eigenspace defined by V.

  $$\overline{y}^i = y^i - m, \quad \text{where} \quad m = \frac{1}{P} \sum_{i=1}^{P} y^i \tag{3.11}$$

  and

  $$\widehat{y}^i = V^T \overline{y}^i \tag{3.12}$$

  The projected test image is compared to every projected training image, and the training image that is found to be closest to the test image is used to identify the training image. The images can be compared using any number of similarity metrics such as the Euclidean distance.

  The following is an example of identifying images through eigenspace projection. Let the four images in *Figure 3.1(a)–Figure 3.1(d)* be training images, and let the additional image in *Figure 3.1(e)* be a test image. The four training images and the mean image are:

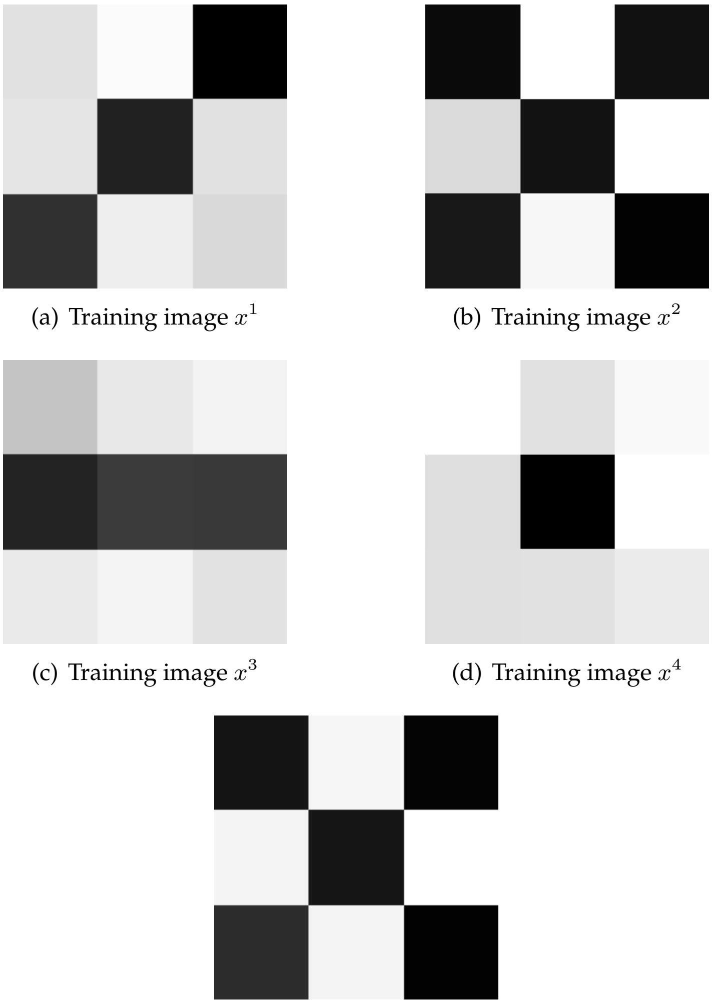
$$x^1 = \begin{bmatrix} 225 \\ 229 \\ 48 \\ 251 \\ 33 \\ 238 \\ 0 \\ 225 \\ 217 \end{bmatrix} \quad x^2 = \begin{bmatrix} 10 \\ 219 \\ 24 \\ 255 \\ 18 \\ 247 \\ 17 \\ 255 \\ 2 \end{bmatrix} \quad x^3 = \begin{bmatrix} 196 \\ 35 \\ 234 \\ 232 \\ 59 \\ 244 \\ 243 \\ 57 \\ 226 \end{bmatrix} \quad x^4 = \begin{bmatrix} 255 \\ 223 \\ 224 \\ 225 \\ 0 \\ 225 \\ 249 \\ 255 \\ 235 \end{bmatrix} \quad m = \begin{bmatrix} 171.50 \\ 176.50 \\ 132.50 \\ 240.75 \\ 27.50 \\ 238.50 \\ 127.25 \\ 198.00 \\ 170.00 \end{bmatrix}$$

The centered images are:

$$\overline{x}^1 = \begin{bmatrix} 53.5000 \\ 52.5000 \\ -84.5000 \\ 10.2500 \\ 5.5000 \\ -0.5000 \\ -127.2500 \\ 27.0000 \\ 47.0000 \end{bmatrix} \quad \overline{x}^2 = \begin{bmatrix} -161.5000 \\ 42.5000 \\ -108.5000 \\ 14.2500 \\ -9.5000 \\ 8.5000 \\ -110.2500 \\ 57.0000 \\ -168.0000 \end{bmatrix} \quad \overline{x}^3 = \begin{bmatrix} 24.5000 \\ -141.5000 \\ 101.5000 \\ -8.7500 \\ 31.5000 \\ 5.5000 \\ 115.7500 \\ -141.0000 \\ 56.0000 \end{bmatrix} \quad \overline{x}^4 = \begin{bmatrix} 83.5000 \\ 46.5000 \\ 91.5000 \\ -15.7500 \\ -27.5000 \\ -13.5000 \\ 121.7500 \\ 57.0000 \\ 65.0000 \end{bmatrix}$$

Combine all the centered training images into one data matrix:

$$\overline{X} = \begin{bmatrix} 53.5000 & -161.5000 & 24.5000 & 83.5000 \\ 52.5000 & 42.5000 & -141.5000 & 46.5000 \\ -84.5000 & -108.5000 & 101.5000 & 91.5000 \\ 10.2500 & 14.2500 & -8.7500 & -15.7500 \\ 5.5000 & -9.5000 & 31.5000 & -27.5000 \\ -0.5000 & 8.5000 & 5.5000 & -13.5000 \\ -127.2500 & -110.2500 & 115.7500 & 121.7500 \\ 27.0000 & 57.0000 & -141.0000 & 57.0000 \\ 47.0000 & -168.0000 & 56.0000 & 65.0000 \end{bmatrix}$$



(a) Training image x^1

(b) Training image x^2

(c) Training image x^3

(d) Training image x^4

(e) Test image y^1

Figure 3.1: Four training patterns and and one test pattern for PCA

Next, compute the covariance matrix:

$$\Omega = \overline{X}\,\overline{X}^T = \begin{bmatrix} 3.6517 & -0.3639 & 2.3129 & -0.3282 & 0.0304 & -0.2392 & 2.4000 & -0.6456 & 3.6446 \\ -0.3639 & 2.6747 & -1.9155 & 0.1650 & -0.5851 & -0.1071 & -2.2083 & 2.6442 & -0.9574 \\ 2.3129 & -1.9155 & 3.7587 & -0.4742 & 0.1247 & -0.1557 & 4.5603 & -1.7562 & 2.5888 \\ -0.3282 & 0.1650 & -0.4742 & 0.0633 & 0.0078 & 0.0280 & -0.5806 & 0.1425 & -0.3426 \\ 0.0304 & -0.5851 & 0.1247 & 0.0078 & 0.1869 & 0.0461 & 0.0645 & -0.6402 & 0.1831 \\ -0.2392 & -0.1071 & -0.1557 & 0.0280 & 0.0461 & 0.0285 & -0.1880 & -0.1074 & -0.2021 \\ 2.4000 & -2.2083 & 4.5603 & -0.5806 & 0.0645 & -0.1880 & 5.6569 & -1.9101 & 2.6937 \\ -0.6456 & 2.6442 & -1.7562 & 0.1425 & -0.6402 & -0.1074 & -1.9101 & 2.7108 & -1.2498 \\ 3.6446 & -0.9574 & 2.5888 & -0.3426 & 0.1831 & -0.2021 & 2.6937 & -1.2498 & 3.7794 \end{bmatrix}$$

The ordered non-zero eigenvectors of the covariance matrix and the corresponding eigenvalues are:

$$\nu_1 = \begin{bmatrix} 0.3672 \\ -0.2731 \\ 0.4795 \\ -0.0592 \\ 0.0343 \\ -0.0192 \\ 0.5582 \\ -0.2725 \\ 0.4120 \end{bmatrix} \qquad \nu_2 = \begin{bmatrix} 0.5447 \\ 0.5275 \\ -0.0600 \\ -0.0148 \\ -0.1165 \\ -0.0621 \\ -0.1251 \\ 0.4644 \\ 0.4153 \end{bmatrix} \qquad \nu_3 = \begin{bmatrix} 0.2728 \\ -0.2926 \\ -0.3084 \\ 0.0593 \\ 0.2031 \\ 0.0422 \\ -0.5752 \\ -0.4682 \\ 0.3850 \end{bmatrix}$$

$$\lambda_1 = 0.2533 \qquad \lambda_2 = 0.4752 \qquad \lambda_3 = 1.5226$$

The eigenspace is defined by the projection matrix, V:

$$V = \begin{bmatrix} 0.3672 & 0.5447 & 0.2728 \\ -0.2731 & 0.5275 & -0.2926 \\ 0.4795 & -0.0600 & -0.3084 \\ -0.0592 & -0.0148 & 0.0593 \\ 0.0343 & -0.1165 & 0.2031 \\ -0.0192 & -0.0621 & 0.0422 \\ 0.5582 & -0.1251 & -0.5752 \\ -0.2725 & 0.4644 & -0.4682 \\ 0.4120 & 0.4153 & 0.3850 \end{bmatrix}$$

The four centered training images projected into eigenspace are:

$$\widehat{x}^1 = V^T \overline{x}^1 = \begin{bmatrix} -94.6419 \\ 109.1201 \\ 105.6437 \end{bmatrix} \qquad \widehat{x}^2 = V^T \overline{x}^2 = \begin{bmatrix} -270.5630 \\ -88.1799 \\ -51.7143 \end{bmatrix}$$

$$\widehat{x}^3 = V^T \overline{x}^3 = \begin{bmatrix} 223.9086 \\ -127.9723 \\ 43.8963 \end{bmatrix} \qquad \widehat{x}^4 = V^T \overline{x}^4 = \begin{bmatrix} 141.2963 \\ 107.0321 \\ -97.8257 \end{bmatrix}$$

The test image viewed as a vector and the centered test image are:

$$y^1 = \begin{bmatrix} 20 \\ 244 \\ 44 \\ 246 \\ 21 \\ 244 \\ 4 \\ 255 \\ 2 \end{bmatrix} \qquad \overline{y}^1 = \begin{bmatrix} -151.5000 \\ 67.5000 \\ -88.5000 \\ 5.2500 \\ -6.5000 \\ 5.5000 \\ -123.2500 \\ 57.0000 \\ -168.0000 \end{bmatrix}$$

The projected test image is:

$$\widehat{y}^i = V^T \overline{y}^1 \begin{bmatrix} -270.6914 \\ -69.1469 \\ -55.0417 \end{bmatrix}$$

The Euclidean distance of the test image y_1 from the four training images x_1, x_2, x_3 and x_4 are 297.64, 19.32, 507.81 and 450.12 respectively. By comparing these distance values, the second training image x_2 is found to be closest to the test image y_1. Therefore, the test image y_1 is identified as belonging to the same class of images as the second training image x_2. One can validate this result by visually inspecting the original test and training images.

3.2.3 Summary of PCA

The steps in finding the principle components can be summarized as follows:

- Form a face library that consists of the face images of known individuals.
- Select a training set that includes a number of images (M) for each person with variation in expression and lighting.
- Compute the mean (average) of the training set and subtract it from each image in the training set.
- Compute the covariance matrix

$$C = \frac{1}{M} = \sum_{n=1}^{M} \Phi_n \Phi_n^T = AA^T$$

- Determine eigenvalues and eigenvectors of the covariance matrix C.
- Choose the M^I eigenvectors with the highest eigenvalues.
- Combine the normalized training set of images to produce M^I eigenfaces. Store these eigenfaces for later use.
- Determine the projection vectors of the training images.

- The projected test image is compared to every projected training image by calculating the Euclidean distance.

- The result is the training image, which is closest to the test image.

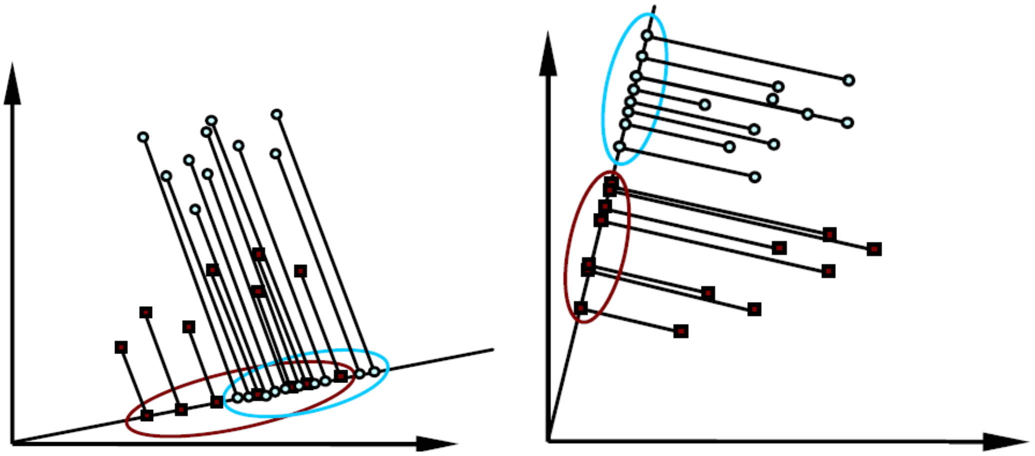
- Compute the system accuracy rate.

3.3 Linear Discriminant Analysis (LDA)

The LDA approach or Fisher discriminants group images of the same class and separates images of different classes. Images are projected from N-dimensional space (where N is the number of pixels in the image) to $C-1$ dimensional space (where C is the number of classes of images). For example, consider two sets of points in 2-dimensional space that are projected onto a single line. Depending on the direction of the line, the points can either be mixed together (*Figure 3.2(a)*)) or separated (*Figure 3.2(b)*). Fisher discriminants find the line that best separates the points. To identify a test image, the projected test image is compared to each projected training image, and the test image is identified as the closest training image.

As with eigenspace projection, training images are projected into a subspace. The test images are projected into the same subspace and identified using a similarity measure. What differs is how the subspace is calculated. Following are the steps to follow to find the Fisher discriminants for a set of images.

The within-class scatter matrix, also called intra-personal, represents variations in appearance of the same individual due to different lighting and face expression, while the between-class scatter matrix, also called the extra-personal, represents variations in appearance due to a difference in identity. By applying this method, we find the projection directions that on one hand maximize the Euclidean distance between the face images of different classes on the other hand minimize the distance between the face images of the same class. The ratio is maximized when the column vectors of the projection matrix are the eigenvalues of $S_w^{-1}S_b$. In face recognition tasks this method cannot be applied directly since the dimension of the sample space is typically larger than the number of samples in the training set, in other words is singular in this case. This problem is also known as the “small sample size problem” [Huang et al., 2002].



(a) Points are mixed when projected onto a line

(b) Points can be grouped when projected onto a line

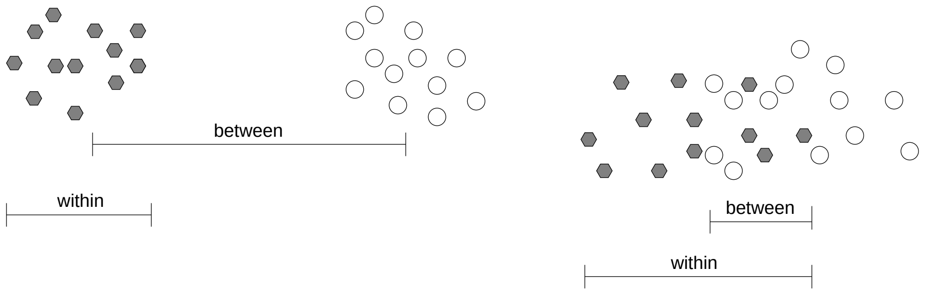
Figure 3.2: Linear Discriminant Analysis (LDA) example demonstrating the scattering of points in Fisher space based on projections on a line that best separates the points [Eleyan and Demirel, 2006]

In the last two decades numerous methods, have been proposed to over come this problem. Tian et al [Tian et al., 1986] used the Pseudo-Inverse method to solve the singularity problem by replacing S_w^{-1} with its Pseudo inverse. Perturbation method has been used to solve this problem [Cheng et al., 1992, Chen et al., 2000], where a small Perturbation matrix is added to in order to make it non-singular. Cheng et al [Cheng et al., 1992] proposed the Rank Decomposition method based on successive eigen-decompositions of the total scatter matrix S_b and the between class scatter matrix. Chen et al [Chen et al., 2000] proposed the Null space method; it can be used when the dimension of the sample space is larger than the rank of the within-class scatter matrix. Another method, the Principal Component Analysis + Null space method was proposed by Huang et al [Huang et al., 2002] for dealing with small sample size problem. In this method, PCA is applied to remove the null space of S_b and S_w, which contains the intersection of the null spaces of and. Then, the optimal projection vectors are found in the remaining lower-dimensional space by using the Null space method.

The problem that may arise during recognition that the within-class scatter matrix sometimes becomes singular, so, in order to overcome this problem and make the within class scatter matrix nonsingular, the Pseudo-Inverse method [Tian et al., 1986] has been used in this thesis.

In terms of face recognition, maximizing the between class scatter matrix and minimizing the within class scatter matrix means grouping images of the same class and separate images of different classes. Images are projected from a N-dimensional space, where N is the number of pixels in the image, to a $M-1$ dimensional space, where M is the number of classes of images.

Figure 3.3 shows good and bad class separation.



(a) Good class separation (b) Bad class separation

Figure 3.3: Linear Discriminant Analysis (LDA) class separation example demonstrating good and bad class separation

3.3.1 Calculating Fisherfaces

Unlike the PCA method that extracts features to best represent face images, the Linear Discriminant Analysis (LDA) method tries to find the subspace that best discriminates different face classes. It is achieved by maximizing the between-class scatter matrix S_b, while minimizing the within-class scatter matrix S_w in the projective subspace.

S_w and S_b are defined as

$$S_w = \sum_{j=1}^{C} \sum_{i=1}^{N_j} (X_i^j - \mu_j)(X_i^j - \mu_j)^T \tag{3.13}$$

where X_i^j is the ith sample of class j, μ_j is the mean of class j, C is the number of classes, and N_j is the number of samples in class j.

$$S_b = \sum_{j=1}^{C} (\mu_j - \mu)(\mu_j - \mu)^T \tag{3.14}$$

where μ represents the mean of all classes.

The subspace for LDA is spanned by a set of vectors W= $[W_1, W_2, \dots, W_d]$ satisfying

$$W = \text{argmax}\left|\frac{W^T S_b W}{W^T S_w W}\right| \tag{3.15}$$

The within class scatter matrix represents how face images are distributed closely within classes and between class scatter matrix describes how classes are separated each other. When face images are projected into the Discriminant vectors W, face images should be distributed closely within classes and should be separated between classes, as much as possible. In other words, these Discriminant vectors minimize the denominator and maximize the numerator in *Equation 3.15*.

W can therefore be constructed by the eigenvectors of ${S_w}^{-1} S_b$.

There are various methods to solve the problem of LDA as mentioned in the related work section, such as the pseudo inverse method, the subspace method, or the null space method. In this thesis, the singularity problem (Small Sample Size problem) has been solved by using the pseudo inverse method.

The approach is similar to the eigenface method, which makes use of projection of training images into a subspace. The test images are projected into the same subspace and identified using a similarity measure. What differs is how subspace is calculated.

The face which has the minimum distance with the test face image is labeled with the identity of that image. The minimum distance can be found using the Euclidean distance method.

3.3.2 The Pseudo-Inverse Method

Pseudo inverse is one of the most successful methods that have been used to overcome the singularity or "small sample size" (SSS) problem arising from the small number of available training samples compared to the dimensionality of the sample space. This problem occurs in calculating the within classes scatter matrix.

The method can be expressed numerically as follows;

A set of linear algebraic equations can be written as

$$Ax = b \tag{3.16}$$

where A is a matrix of coefficients, $m \times n$ and $m \times 1$ is some form of a system output vector. The vector x is what we usually solve for. If $m = n$, then there are as many equations as the number of unknowns, and there is a good chance of solving for x. That is

$$A^{-1}Ax = A^{-1}b \tag{3.17}$$

$$x = A^{-1}Ax = A^{-1}b \tag{3.18}$$

Here, we compute the inverse of A. This can prove to be a challenging task, however, since there are many situations where the inverse of A does not exist. In these cases the inverse can be computed using the pseudo inverse based on singular value decomposition (SVD) method which can turn a singular problem into a non-singular one. Vector x in *Equation 3.16* can also be solved for by using the transpose of A.

That is

$$A^{T}Ax = A^{T}b \tag{3.19}$$

$$x = (A^{T}A)^{-1}A^{T}b \tag{3.20}$$

where the inverse of A is expressed as

$$A^{+} = (A^{T}A)^{-1}A^{T} \tag{3.21}$$

where A^{+} is called the more *Penrose pseudo inverse*.

3.3.3 An LDA Example

As with eigenspace projection, training images are projected into a subspace. The test images are projected into the same subspace and identified using a similarity measure. What differs is how the subspace is calculated. Following are the steps to follow to find the Fisher discriminants for a set of images.

1. **Calculate the within class scatter matrix:** The within class scatter matrix measures

the amount of scatter between items in the same class. For the i^{th} class, a scatter matrix (S_i) is calculated as the sum of the covariance matrices of the centered images in that class.

$$S_i = \sum_{x \varepsilon X_i} (x - m_i)(x - m_i)^T \tag{3.22}$$

where m_i is the mean of the images in the class. The within class scatter matrix (S_W) is the sum of all the scatter matrices.

$$S_w = \sum_{i=1}^{C} S_i \tag{3.23}$$

where C is the number of classes.

2. **Calculate the between-class scatter matrix:** The between-class scatter matrix (S_B) measures the amount of scatter between classes. It is calculated as the sum of the covariance matrices of the difference between the total mean and the mean of each class.

$$S_b = \sum_{i=1}^{C} n_i(m_i - m)(m_i - m)^T \tag{3.24}$$

where n_i is the number of images in the class, m_i is the mean of the images in the class and m is the mean of all the images.

3. **Solve the generalized eigenvalue problem:** Solve for the generalized eigenvectors (V) and eigenvalues (Λ) of the within class and between class scatter matrices.

$$S_b V = \Lambda S_w V \tag{3.25}$$

4. **Keep first** $(C - 1)$ **eigenvectors:** Sort the eigenvectors by their associated eigenvalues from high to low and keep the first $C - 1$ eigenvectors. These eigenvectors form the Fisher basis vectors.

5. **Project images onto Fisher basis vectors:** Project all the original (i.e. not centered) images onto the Fisher basis vectors by calculating the dot product of the image

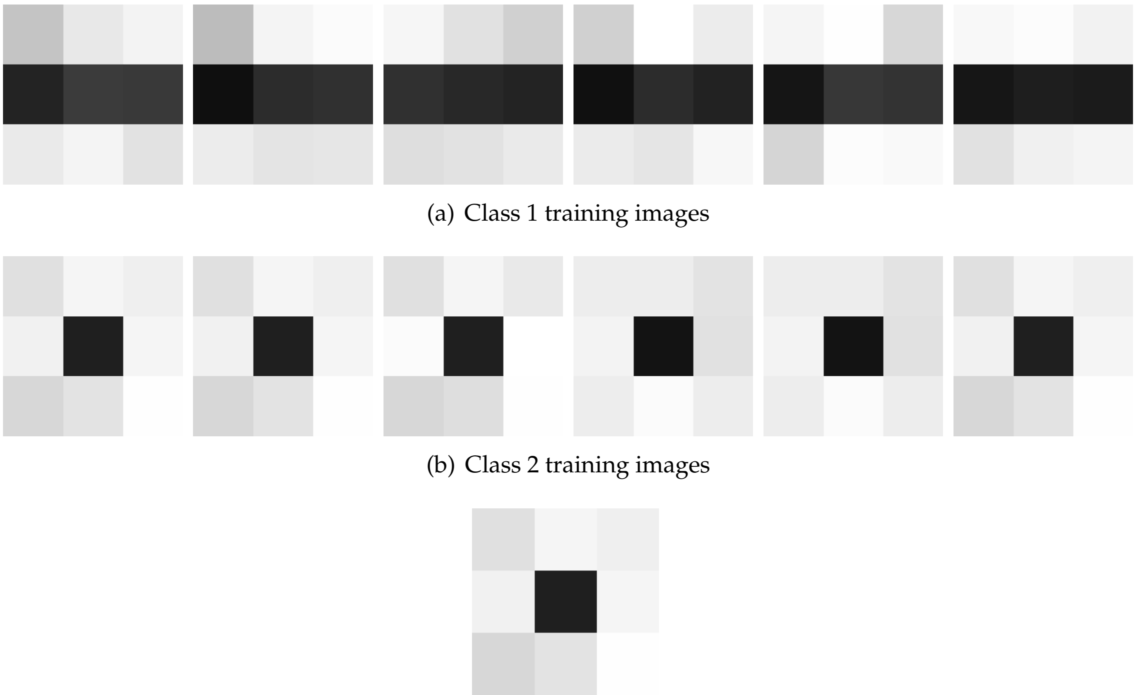
with each of the Fisher basis vectors. The original images are projected onto this line because these are the points that the line has been created to discriminate, not the centered images.

Following is an example of calculating the Fisher discriminants for a set of images. Let the twelve images in *Figure 3.4* be training images. There are two classes. Images x^1–x^6 are in the first class and images x^7–x^{12} are in the second class. The training images viewed as vectors are:

$$x^1 = \begin{bmatrix} 196 \\ 35 \\ 234 \\ 232 \\ 59 \\ 244 \\ 243 \\ 57 \\ 226 \end{bmatrix} \quad x^2 = \begin{bmatrix} 188 \\ 15 \\ 236 \\ 244 \\ 44 \\ 228 \\ 251 \\ 48 \\ 230 \end{bmatrix} \quad x^3 = \begin{bmatrix} 246 \\ 48 \\ 222 \\ 225 \\ 40 \\ 226 \\ 208 \\ 35 \\ 234 \end{bmatrix} \quad x^4 = \begin{bmatrix} 208 \\ 16 \\ 235 \\ 255 \\ 44 \\ 229 \\ 236 \\ 34 \\ 247 \end{bmatrix} \quad x^5 = \begin{bmatrix} 245 \\ 21 \\ 213 \\ 254 \\ 55 \\ 252 \\ 215 \\ 51 \\ 249 \end{bmatrix} \quad x^6 = \begin{bmatrix} 248 \\ 22 \\ 225 \\ 252 \\ 30 \\ 240 \\ 242 \\ 27 \\ 244 \end{bmatrix}$$

$$x^7 = \begin{bmatrix} 255 \\ 223 \\ 224 \\ 255 \\ 0 \\ 255 \\ 249 \\ 255 \\ 235 \end{bmatrix} \quad x^8 = \begin{bmatrix} 234 \\ 255 \\ 205 \\ 251 \\ 0 \\ 251 \\ 238 \\ 253 \\ 240 \end{bmatrix} \quad x^9 = \begin{bmatrix} 232 \\ 255 \\ 231 \\ 247 \\ 38 \\ 246 \\ 190 \\ 236 \\ 250 \end{bmatrix} \quad x^{10} = \begin{bmatrix} 255 \\ 241 \\ 208 \\ 255 \\ 28 \\ 255 \\ 194 \\ 234 \\ 188 \end{bmatrix} \quad x^{11} = \begin{bmatrix} 237 \\ 243 \\ 237 \\ 237 \\ 19 \\ 251 \\ 227 \\ 225 \\ 237 \end{bmatrix} \quad x^{12} = \begin{bmatrix} 224 \\ 251 \\ 215 \\ 245 \\ 31 \\ 222 \\ 233 \\ 255 \\ 254 \end{bmatrix}$$

The scatter matrices S_1 and S_2 are as follows:



(a) Class 1 training images

(b) Class 2 training images

(c) Test image

Figure 3.4: Training images belonging to two different classes and a test image to be used with the LDA approach

$$S_1 = \begin{bmatrix} 3.8088 & 0.5892 & -1.0935 & 0.1397 & -0.5957 & 0.3945 & -1.6945 & -0.8330 & 0.7713 \\ 0.5892 & 0.8268 & -0.1485 & -0.7177 & 0.0467 & -0.0865 & -0.6335 & 0.0100 & -0.2773 \\ -1.0935 & -0.1485 & 0.4175 & -0.0560 & -0.0050 & -0.2555 & 0.6165 & 0.0340 & -0.2310 \\ 0.1397 & -0.7177 & -0.0560 & 0.7893 & -0.1033 & 0.2100 & 0.2790 & -0.1650 & 0.4777 \\ -0.5957 & 0.0467 & -0.0050 & -0.1033 & 0.5473 & 0.2760 & -0.0700 & 0.5620 & -0.1297 \\ 0.3945 & -0.0865 & -0.2555 & 0.2100 & 0.2760 & 0.5475 & -0.0855 & 0.2820 & 0.1440 \\ -1.6945 & -0.6335 & 0.6165 & 0.2790 & -0.0700 & -0.0855 & 1.4615 & 0.1120 & -0.2800 \\ -0.8330 & 0.0100 & 0.0340 & -0.1650 & 0.5620 & 0.2820 & 0.1120 & 0.6800 & -0.2630 \\ 0.7713 & -0.2773 & -0.2310 & 0.4777 & -0.1297 & 0.1440 & -0.2800 & -0.2630 & 0.4613 \end{bmatrix}$$

$$S_2 = \begin{bmatrix} 0.8135 & -0.6210 & -0.0890 & 0.2820 & -0.3790 & 0.6110 & -0.0465 & -0.0970 & -1.1680 \\ -0.6210 & 0.7393 & -0.1440 & -0.1573 & 0.4547 & -0.3367 & -0.5863 & -0.0900 & 0.4960 \\ -0.0890 & -0.1440 & 0.8200 & -0.2840 & 0.2500 & 0.0580 & -0.1180 & -0.4370 & 0.5930 \\ 0.2820 & -0.1573 & -0.2840 & 0.2373 & -0.1827 & 0.1567 & -0.0147 & 0.2200 & -0.4060 \\ -0.3790 & 0.4547 & 0.2500 & -0.1827 & 1.3073 & -0.4743 & -1.5447 & -0.4880 & -0.0030 \\ 0.6110 & -0.3367 & 0.0580 & 0.1567 & -0.4743 & 0.7853 & -0.1673 & -0.3010 & -0.8400 \\ -0.0465 & -0.5863 & -0.1180 & -0.0147 & -1.5447 & -0.1673 & 2.9388 & 1.0020 & 1.1340 \\ -0.0970 & -0.0900 & -0.4370 & 0.2200 & -0.4880 & -0.3010 & 1.0020 & 0.8420 & 0.5600 \\ -1.1680 & 0.4960 & 0.5930 & -0.4060 & -0.0030 & -0.8400 & 1.1340 & 0.5600 & 2.8180 \end{bmatrix}$$

The within class scatter matrix is:

$$S_w = \begin{bmatrix} 4.6223 & -0.0318 & -1.1825 & 0.4217 & -0.9747 & 1.0055 & -1.7410 & -0.9300 & -0.3967 \\ -0.0318 & 1.5662 & -0.2925 & -0.8750 & 0.5013 & -0.4232 & -1.2198 & -0.0800 & 0.2187 \\ -1.1825 & -0.2925 & 1.2375 & -0.3400 & 0.2450 & -0.1975 & 0.4985 & -0.4030 & 0.3620 \\ 0.4217 & -0.8750 & -0.3400 & 1.0267 & -0.2860 & 0.3667 & 0.2643 & 0.0550 & 0.0717 \\ -0.9747 & 0.5013 & 0.2450 & -0.2860 & 1.8547 & -0.1983 & -1.6147 & 0.0740 & -0.1327 \\ 1.0055 & -0.4232 & -0.1975 & 0.3667 & -0.1983 & 1.3328 & -0.2528 & -0.0190 & -0.6960 \\ -1.7410 & -1.2198 & 0.4985 & 0.2643 & -1.6147 & -0.2528 & 4.4003 & 1.1140 & 0.8540 \\ -0.9300 & -0.0800 & -0.4030 & 0.0550 & 0.0740 & -0.0190 & 1.1140 & 1.5220 & 0.2970 \\ -0.3967 & 0.2187 & 0.3620 & 0.0717 & -0.1327 & -0.6960 & 0.8540 & 0.2970 & 3.2793 \end{bmatrix}$$

The mean of each class and the total mean are:

$$m^1 = \begin{bmatrix} 221.8333 \\ 26.1667 \\ 227.5000 \\ 243.6667 \\ 45.3333 \\ 236.5000 \\ 232.5000 \\ 42.0000 \\ 238.3333 \end{bmatrix} \quad m^2 = \begin{bmatrix} 239.5000 \\ 244.6667 \\ 220.0000 \\ 248.3333 \\ 19.3333 \\ 246.6667 \\ 221.8333 \\ 243.0000 \\ 234.0000 \end{bmatrix} \quad m = \begin{bmatrix} 230.6667 \\ 135.4167 \\ 223.7500 \\ 246.0000 \\ 32.3333 \\ 241.5833 \\ 227.1667 \\ 142.5000 \\ 236.1667 \end{bmatrix}$$

The between class scatter matrix is:

$$S_b = \begin{bmatrix} 0.0094 & 0.1158 & -0.0040 & 0.0025 & -0.0138 & 0.0054 & -0.0057 & 0.1065 & -0.0023 \\ 0.1158 & 1.4323 & -0.0492 & 0.0306 & -0.1704 & 0.0666 & -0.0699 & 1.3176 & -0.0284 \\ -0.0040 & -0.0492 & 0.0017 & -0.0011 & 0.0059 & -0.0023 & 0.0024 & -0.0452 & 0.0010 \\ 0.0025 & 0.0306 & -0.0011 & 0.0007 & -0.0036 & 0.0014 & -0.0015 & 0.0281 & -0.0006 \\ -0.0138 & -0.1704 & 0.0059 & -0.0036 & 0.0203 & -0.0079 & 0.0083 & -0.1568 & 0.0034 \\ 0.0054 & 0.0666 & -0.0023 & 0.0014 & -0.0079 & 0.0031 & -0.0033 & 0.0613 & -0.0013 \\ -0.0057 & -0.0699 & 0.0024 & -0.0015 & 0.0083 & -0.0033 & 0.0034 & -0.0643 & 0.0014 \\ 0.1065 & 1.3176 & -0.0452 & 0.0281 & -0.1568 & 0.0613 & -0.0643 & 1.2120 & -0.0261 \\ -0.0023 & -0.0284 & 0.0010 & -0.0006 & 0.0034 & -0.0013 & 0.0014 & -0.0261 & 0.0006 \end{bmatrix}$$

Since there are two classes, only one eigenvector is kept. The non-zero eigenvector and corresponding eigenvalue of $S_B V = \lambda \quad S_w V$ are:

$$\nu_1 = \begin{bmatrix} 0.1847 \\ 0.5185 \\ 0.5270 \\ 0.5571 \\ 0.0099 \\ -0.1053 \\ 0.0836 \\ 0.2611 \\ -0.1503 \end{bmatrix}$$

Table 3.1 shows the values of the images projected onto the first eigenvector. *Figure 5* shows a plot of the points, illustrating the separation between the two classes.

Table 3.1: The values of the images projected onto the first eigenvector

Pattern	X_1	X_2	X_3	X_4	X_5	X_6
Class 1	283.0480	278.1952	280.6345	280.4389	277.7812	281.8256
Pattern	X_7	X_8	X_9	X_10	X_11	X_12
Class2	448.0635	446.7615	448.8117	446.2217	442.5680	446.1270
Y	438.3069					

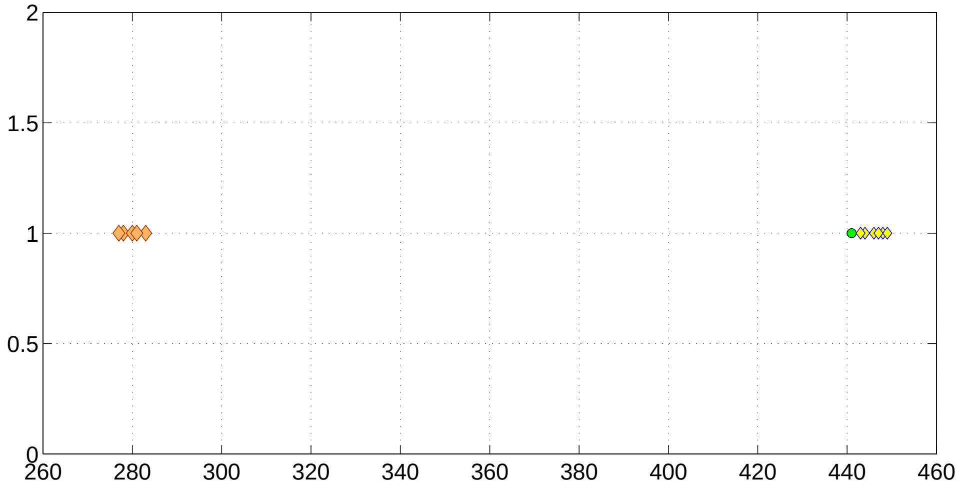


Figure 3.5: Plot of the images projected onto Fisher basis vectors

3.3.4 Summary of LDA

The steps followed in the evaluation of the Fisherfaces are as follows:

- Read images (faces) in the database and divide them into two sets, one for training and the other for testing.
- Images for training
- Create the training set and find the average of each class and find the average for each person's training images, μ_j.
- Find average of the training images in the database, μ.
- Apply the Linear Discriminant Analysis to find the within-scatter matrix and between-scatter matrix:
- The within-scatter matrix:

$$S_w = \sum_{j=1}^{C} \sum_{i=1}^{N_j} (X_i^j - \mu_j)(X_i^j - \mu_j)^T \tag{3.26}$$

- The between-scatter matrix:

$$S_b = \sum_{j=1}^{C} (\mu_j - \mu)(\mu_j - \mu)^T \tag{3.27}$$

- Find the associated eigenvectors using the between scatter matrix and within scatter matrix: ${S_w}^{-1} S_b$.
- Transform the test images and all the faces in the database into the face space created in the previous step.
- Using Euclidean distance method, find the identity of the test image: $\varepsilon_k = ||\Omega - \Omega_k t||^2$
- Compute the system accuracy rate.

3.4 Region-Division with Majority Voting

Instead of applying PCA or LDA directly on face databases [Turk and Pentland, 1991, Belhumeur et al., 1997, Pentland et al., 1994], this thesis presents a slightly different approach, where we divide each face image in the database into five horizontal regions, namely the forehead, eyes, nose, mouth, and chin regions, and then we apply PCA or LDA on each separate region locally and use majority voting method over the Euclidean distance values from each algorithm to decide on a match. That is, no recognition is done on a part-by-part basis, and, instead, each facial region generates a vote for the person recognized using that region only. Therefore, instead of having one database, we have five separate databases, one for each facial region. *Figure 3.6* depicts the architecture of our system.

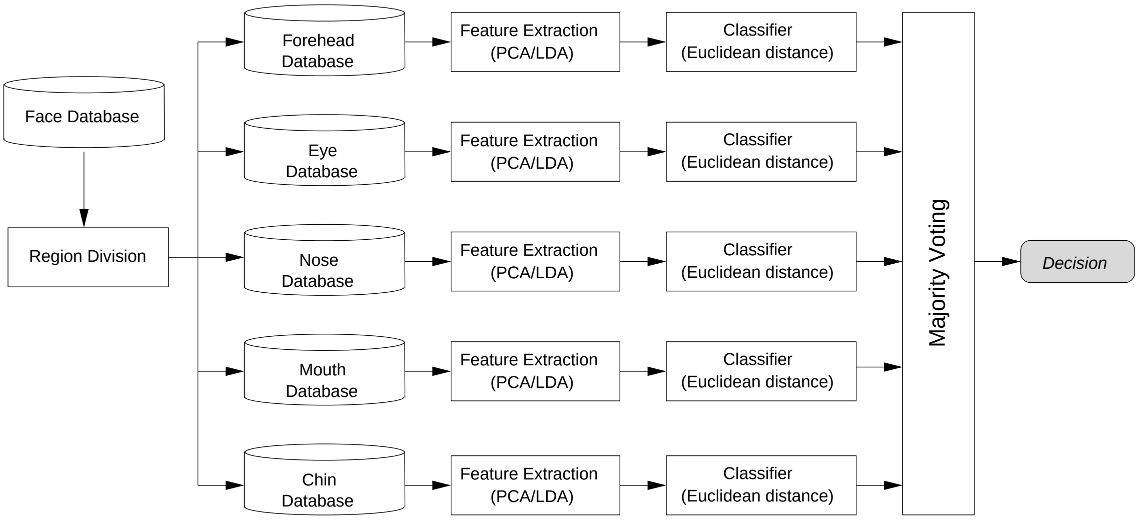


Figure 3.6: The architecture of the proposed region-division approach that uses majority voting

After applying PCA or LDA as alternate feature extractors, we use Euclidean distance over the resulting vectors in the classifier stage. Instead of taking the best value among the results and declare the person as it is usually done, our approach takes the best four values from each facial region, and then we apply majority voting in order to decide on the best match.

Our system is based upon two main algorithms, one for dividing face images into regions and the other for executing the tests.

- **Database Preparation Algorithm:** In this algorithm, the system divides whole-face images belonging to each person into five regions, namely the forehead region, eye region, nose region, mouth region, and chin region. After dividing a single face image into these five regions, our system saves each region image into a separate image database.

```
RegionDivide( Database )
FOR EACH person $n = 1$ to 15 DO
   FOR EACH image $i = 1$ to 10 DO
      Divide $\text{image}_{n_i}$ in given whole-face database Database into five regions: $\{\text{image}_{n_{i_{forehead}}}$,
      $\text{image}_{n_{i_{eye}}}$, $\text{image}_{n_{i_{nose}}}$, $\text{image}_{n_{i_{mouth}}}$, $\text{image}_{n_{i_{chin}}}\}$
      Add $\text{image}_{n_{i_{forehead}}}$ to $\text{Database}_{forehead}$
      Add $\text{image}_{n_{i_{eye}}}$ to $\text{Database}_{eye}$
      Add $\text{image}_{n_{i_{nose}}}$ to $\text{Database}_{nose}$
      Add $\text{image}_{n_{i_{mouth}}}$ to $\text{Database}_{mouth}$
      Add $\text{image}_{n_{i_{chin}}}$ to $\text{Database}_{chin}$
   END
END
```

- **Recognition Algorithm:** This algorithm runs a recognition test for a given test image t using feature extraction method M using the training database TrainDatabase. The algorithm returns the ID of the person that is recognized by the system.

```
Recognize( $t$, $M$, TrainDatabase )
Divide test image $t$ into five regions: $R$ =$\{\text{image}_{forehead}$, $\text{image}_{eye}$, $\text{image}_{nose}$,
$\text{image}_{mouth}$, $\text{image}_{chin}\}$
RegionDivide( TrainDatabase )
FOR EACH test image region $t_s$ in $R$ DO
   FOR EACH $\text{Database}_s$ resulting from RegionDivide( TrainDatabase ) DO
      Process $\text{Database}_s$ using $M$ (PCA or LDA)
      Compute the projection vector $\overline{p_s}$ for test image $t_s$ in $R$
      FOR EACH training image $k$ in $\text{Database}_s$ DO
```

```
            Compute the projection vector $\overline{p_k}$ for $k$
            Compute Euclidean distance $d_{sk}$ between $\overline{p_s}$ and $\overline{p_k}$
        END
    END
END
$D = \{\}$
FOR EACH region $r$ in $\{forehead, eye, nose, mouth, chin\}$ DO
    Determine the smallest/best Euclidean distance value $d_r$
    Add $d_r$ to $D$
END
Determine the best Euclidean distance value $d_{best}$ in $D$
FOR EACH person $p$ in TrainDatabase DO
    Initialize the total votes for recognition $C_p$ to 0
END
FOR EACH distance value $d_r$ in $D$ DO
    Add 1 (vote) $C_p$ where $p$ is a person in TrainDatabase
END
Return the person ID $p$ in TrainDatabase with the highest total vote $C_p$
```

3.5 Evaluation Method nad Experiment Design

In order to evaluate the ideas presented in this thesis, we will compare the performance of our region-division techniques that employ majority voting to the performance of the classic technique that uses the whole face. One of our region-division methods uses all the five regions a face is divided into (*Figure 4.2*, *Chapter 4*). The second region-division technique also uses the same five facial regions, but it also uses the entire face as a sixth region.

Since there are 10 images per person in both the ORL database and the NFID, we conduct tests that vary the number of training images between 1 and 9, as the number of test images vary from 9 to 1. In addition, we conduct each experiment with both PCA and LDA.

Our system randomly generates 10 test scenarios for each run with a given number of training/test images, feature extraction method (PCA/LDA), and face recognition approach (5 Region/6 Region/Whole Face). Therefore, for each given number of training/test images, we run three sets of experiments, one for each face recognition approach. Each run contains 10 different experiments where, in each experiment, each given test image is compared to all available training images in the given database. For example, in the NFID, when two test images are used per person (i.e., number of training images per person will then be 8), each test image will be compared to 15×8 (120) training images.

To test for statistical significance, we use the student's *t-test*. Since each run has 10 scenarios, we compare the ten recognition rate averages for each such run to those obtained from another run. For this comparison, we use the *unpaired* t-test, because each test scenario is not repeated.

3.6 Implementation

We implemented the two region-division approaches with majority voting in Matlab version 7.1. One of the region-division approaches uses the five regions of the face. The second approach uses the five regions of the face but also considers the whole face as a sixth region presented in this thesis. A desktop computer was used for testing with an Intel Celeron 2.8GHz CPU, 512MB main memory, and an 80GB hard disk.

3.7 Summary

This chapter presented our proposed approach for face recognition that uses the two region-division approaches with majority voting scheme. Since we test this approach using PCA and LDA, this chapter also presents the PCA and LDA algorithms for face recognition.

CHAPTER 4

FACE IMAGE DATABASES

4.1 Overview

Our work in this thesis focus mainly in applying PCA and LDA algorithms using ORL (Olivetti Research Laboratories) database [Website, 1994] and our new face database, which we prepared for this purpose. We divided the faces in the database into five regions: forehead region, eye region, nose region, mouth region, and chin region. This chapter discusses the details of the new database.

4.2 Camera Details

We used a SAMSUNG Digimax 301 digital camera with 3.2 megapixel resolution to shoot these photos. Camera features are listed in *Table 4.1*:

4.3 NEU Face Image Database (NFID)

A total of 150 face images where taken against a homogeneous dark background by the author between December 2007 and March 2008. The images belong to 15 students studying at the Near East University, Cyprus. Each student has had his face image captured 10 times where each of the 10 images represents a different facial expression (open/closed eyes, smiling/not smiling), facial details (glasses/no glasses, beard/shaven, hat/without hat), varying lighting, and head pose (tilting and rotation). *Figure 4.1* shows an example subset of the NEU Face Image Database.

Model	SAMSUNG Digimax 301
Pixel Resolution	3.2 MP
PC Camera function	Yes
Digital Camera Function	Yes
Image Resolution	1600×1200, 1280×1024, 1024×768
Lens	Free focus, 50mm equivalent
Shutter	$1/4$ sec $\sim 1/1000$sec
Focusing Range	Normal: 1.5m–$\inf$, Macro: 30–50cm, Text: 11 13cm
Digital Zoom	Yes
Flash	Built-in
File Format	Compressed JPEG
Quality Level	Super fine, Fine, Normal
Connectivity	Mini USB
LCD Monitor	Status LCD

Table 4.1: Specifications of the digital camera used to capture the images used in the NEU Face Image Database

All the images in the NEU Database were preprocessed by resizing them to 90×120 pixels and converting them to grayscale bitmap images, as demonstrated in *Figure 4.2*.

Figure 4.3 is the entire content of the NEU Face Image Database (NFID). The database contains 10 images per person with poses similar to that in the ORL database. Moreover, this database contains subjects wearing beards, hats, and different types of eyewear.

Figure 4.4, *Figure 4.5*, *Figure 4.6*, *Figure 4.7*, and *Figure 4.8* depict the face region databases generated from the NFID.

4.4 The ORL Database

For testing the algorithms performance based on the number of classes, number of Eigenfaces, Fisherfaces, pose and accessories the ORL (Olivetti Research Laboratory) database has been used, which is freely distributed on the Internet (See *Figure 4.9*). The ORL database contains a set of face images taken between April 1992 and April 1994 at the lab. There are 10 face images of 40 subjects where taken against a dark background. The faces vary in lighting, facial expressions, and facial details (glasses/ no glasses) and pose (straight looking, left-looking and right looking). Each face here will be again divided to 5 regions same as what we did with our own face image database. The results of both databases are compared in *Chapter 5*.

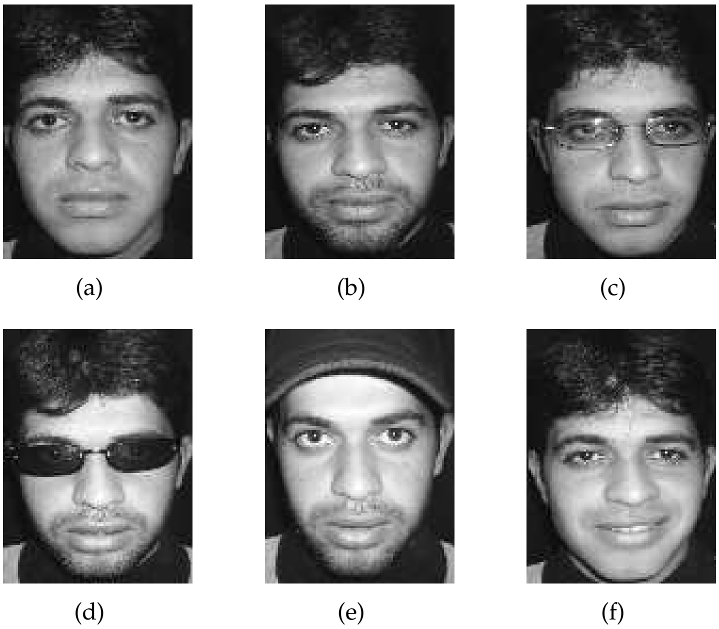
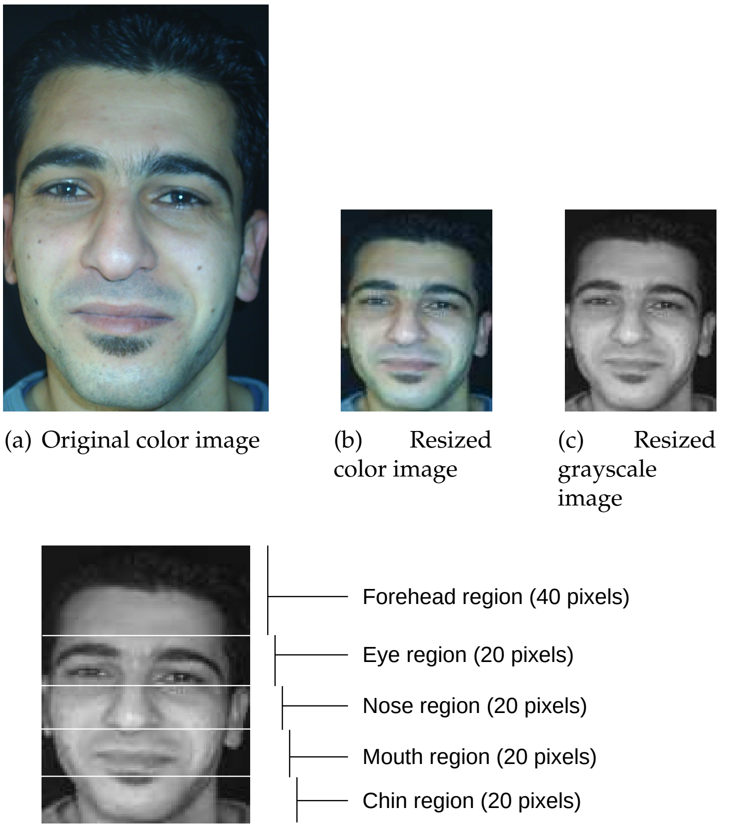

(a) (b) (c) (d) (e) (f)

Figure 4.1: An example set of images from the NEU Database. Image (a) shows the person shaven, image (b) wearing a beard, image (c) shaven but with light-colored glasses, image (d) wearing a beard but with dark sunglasses, (e) wearing a beard but with a baseball hat, and, finally, image (f) shaven and smiling.

4.5 Summary

In this chapter, we showed our new prepared face database together with, the ORL (Olivetti Research Laboratories) database. We divided the faces in the database to five parts; forehead region, eyes region, nose region, mouth region, and chin region. These regions will be supplied separately to the recognition system and the results will be evaluated as it will be shown in next chapter.



(a) Original color image

(b) Resized color image

(c) Resized grayscale image

(d) Division of the face image

Figure 4.2: An example image for the NEU Database. (a) Each image starts with its color original. (b) Then the color original is scaled down to 90 × 120 pixels. (c) Finally, color is replaced by grayscale values (90 × 120 pixels), and the five parts (forehead, eyes, nose, mouth, chin) are created with the given fixed region-division scheme.

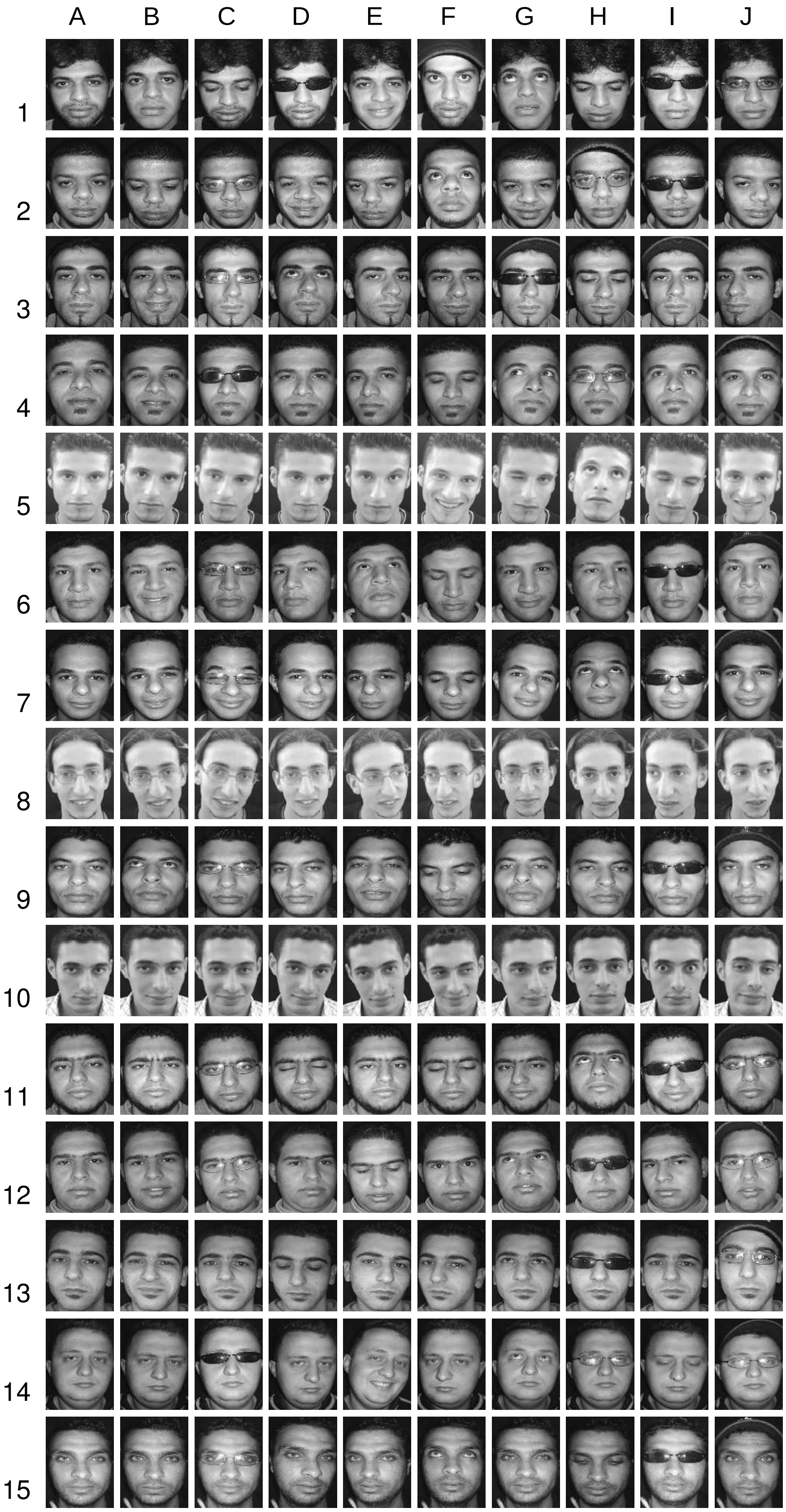


Figure 4.3: NEU Face Image Database. The database has images of 15 persons. For each person, the database contains 10 variations each for a total of 150 images

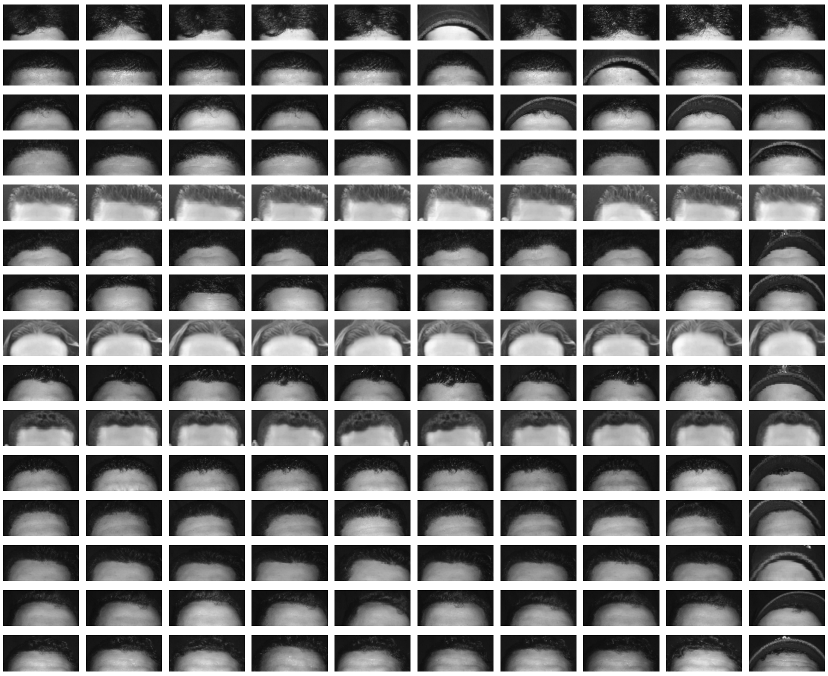


Figure 4.4: NEU Face Image Database (Forehead Region)

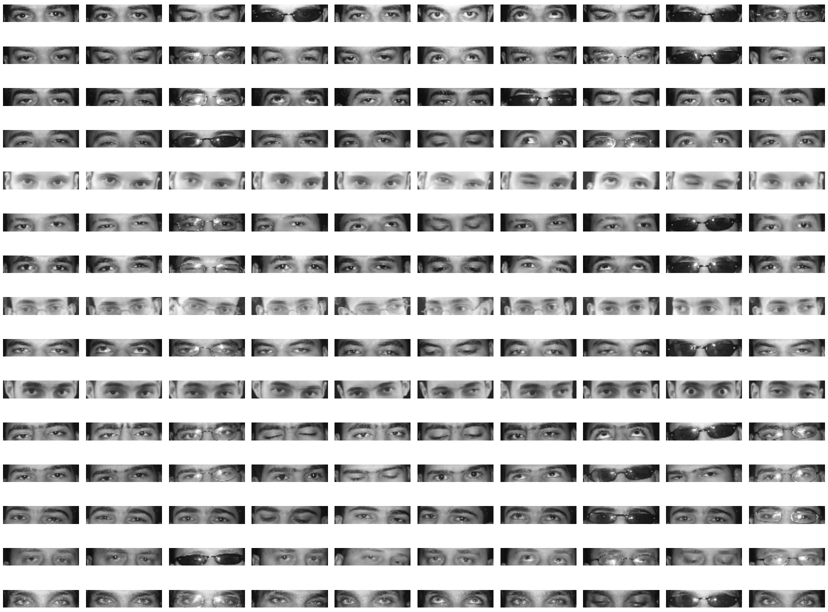


Figure 4.5: NEU Face Image Database (Eye Region)

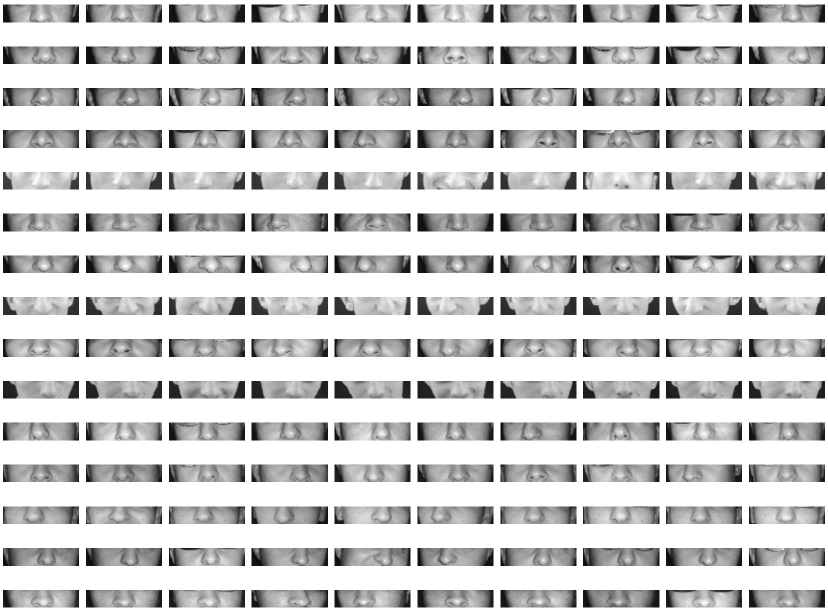


Figure 4.6: NEU Face Image Database (Nose Region)

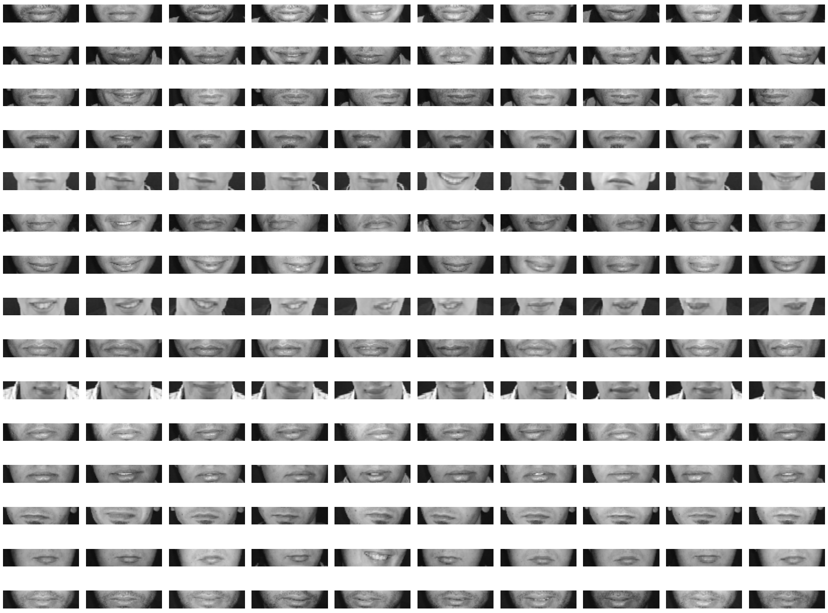


Figure 4.7: NEU Face Image Database (Mouth Region)

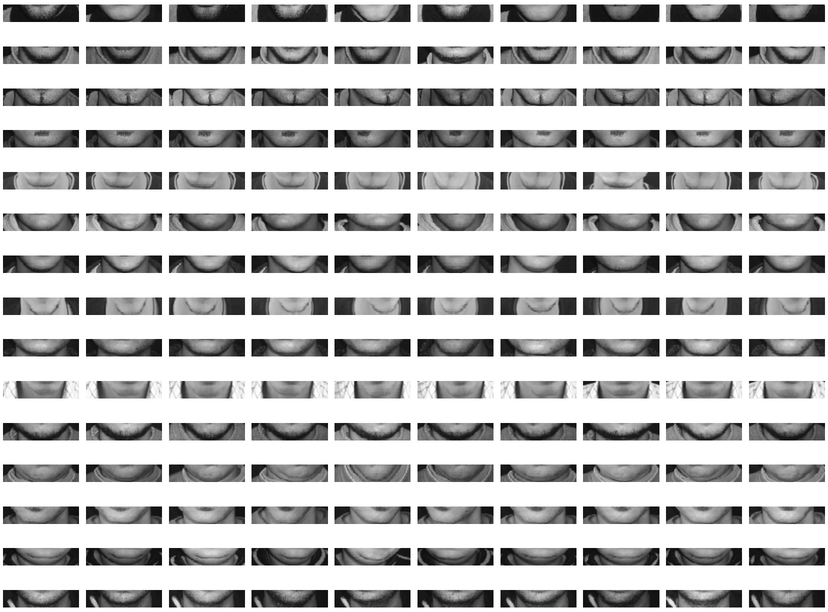


Figure 4.8: NEU Face Image Database (Chin Region)

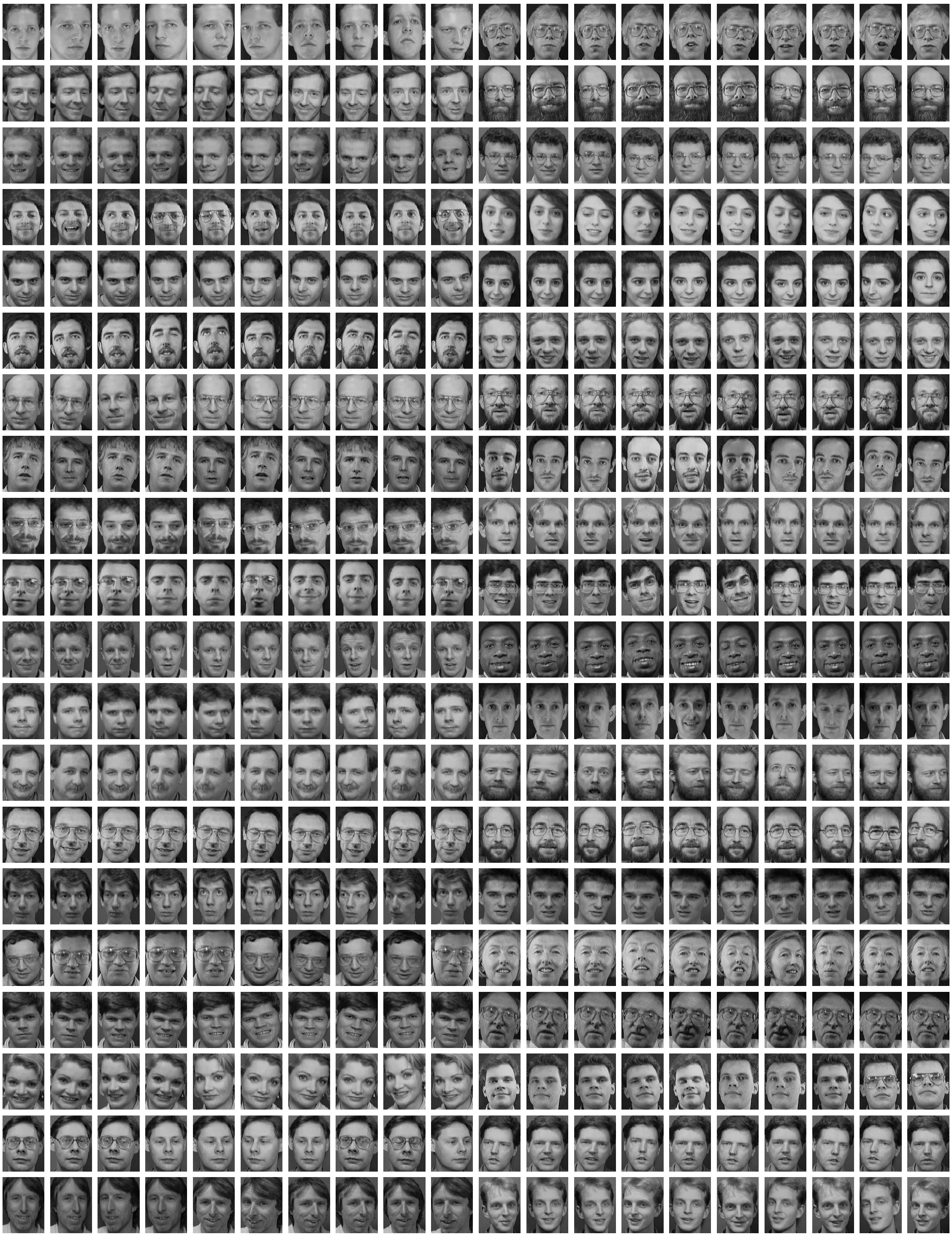


Figure 4.9: ORL Face Image Database. The database has images of 40 persons. For each person, the database contains 10 variations each for a total of 400 images

CHAPTER 5

RESULTS and DISCUSSION

5.1 Overview

This chapter presents the experiments we carried out in our study and the results obtained from these experiments. We used two separate databases, namely the ORL database and our own database that we refer to as the NEU Face Image Database (NFID) (See *Chapter 4*).

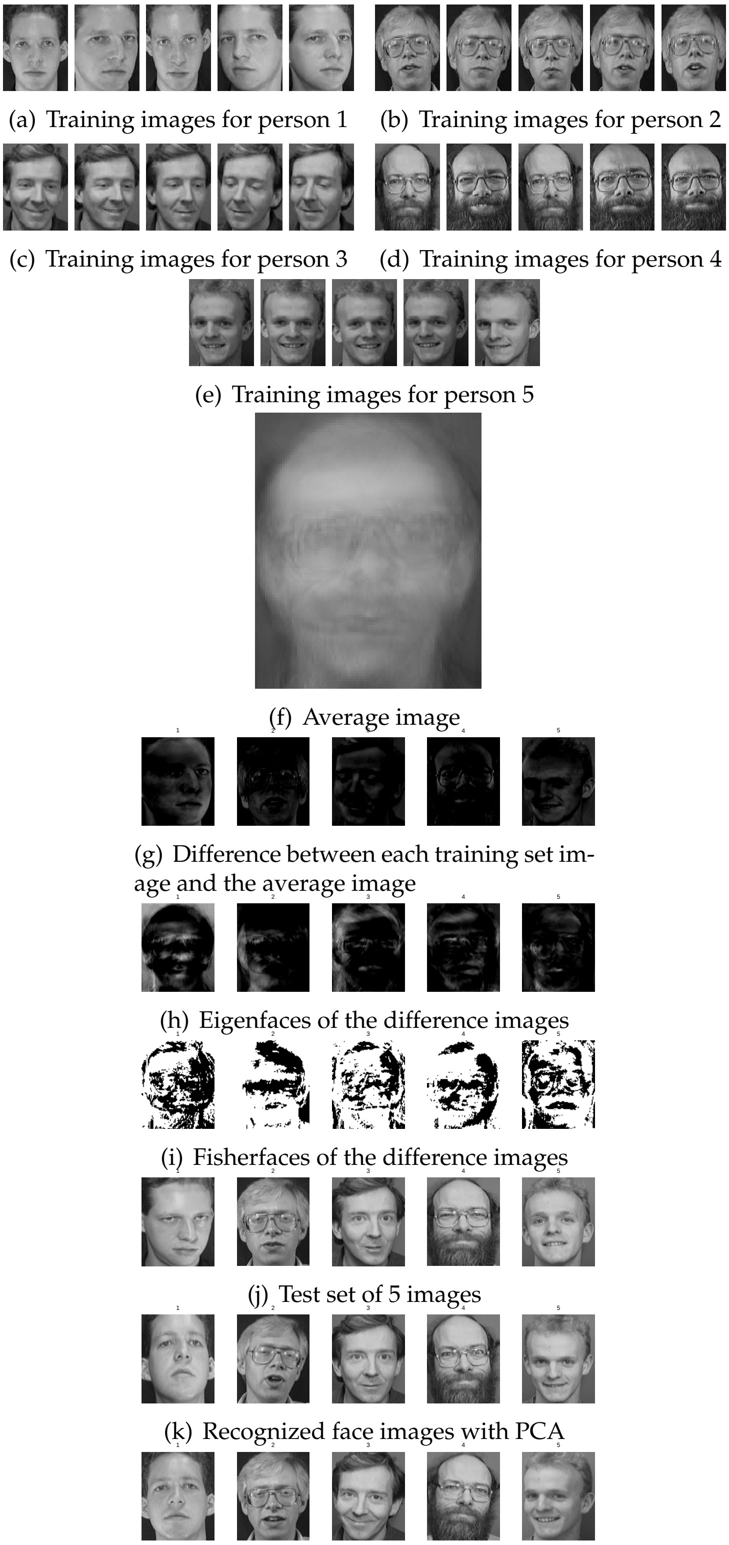
The aim is to show that, existing schemes that use the whole face do not perform well when faces are occluded by what is worn on the face such as beards, glasses, or hats and that, using our proposed region-division with majority voting approach results in better performance than using the classic whole face approach.

5.2 Face Recognition Examples

In this section, we illustrate the way our system works with the help of a few examples.

5.2.1 Whole Face Example

Figure 5.1 depicts the entire sequence of the classic whole face approach to face recognition for a small experiment conducted using a small set of images from the ORL database. Five training images are used for each person, and, then, for each person, one test image is used. As *Figure 5.1* indicates, all 5 test images were correctly identified using both PCA and LDA feature extraction method.



(a) Training images for person 1 (b) Training images for person 2

(c) Training images for person 3 (d) Training images for person 4

(e) Training images for person 5

(f) Average image

(g) Difference between each training set image and the average image

(h) Eigenfaces of the difference images

(i) Fisherfaces of the difference images

(j) Test set of 5 images

(k) Recognized face images with PCA

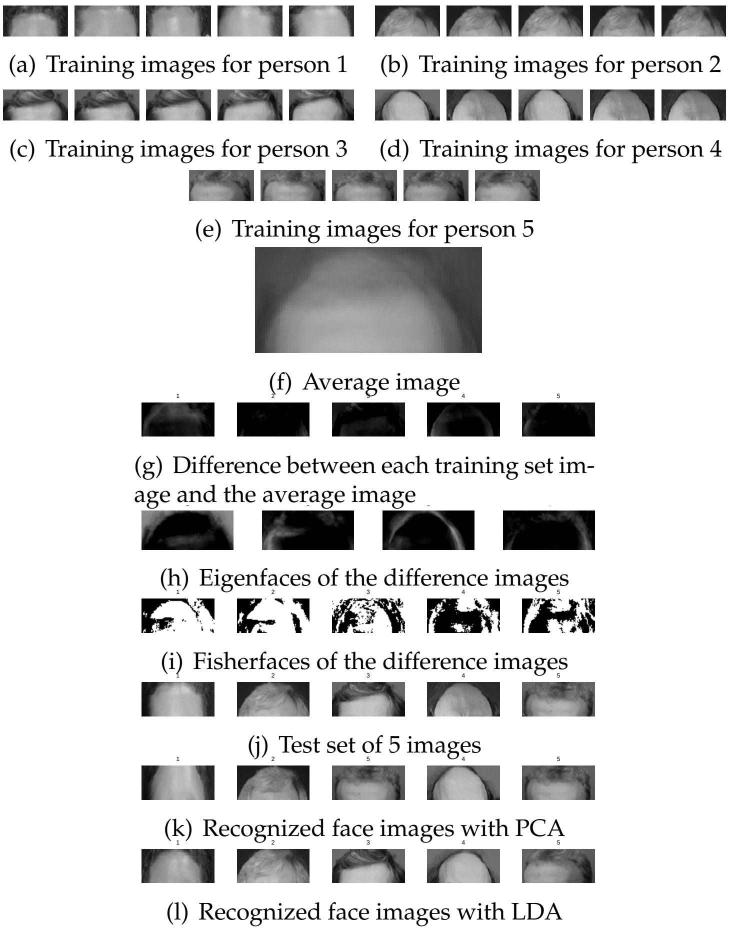
(l) Recognized face images with LDA

Figure 5.1: ORL whole face example using both PCA and LDA. The input to the system are the five sets of face images, each with five variations. Although PCA and LDA recognize different images as most similar to the test images (j), the results (k & l) are 100% correct.

5.2.2 Region-based PCA and LDA Examples

This section provides recognition examples using region-based division in order to provide a perspective for how our system works.

Forehead Example:



(a) Training images for person 1 (b) Training images for person 2

(c) Training images for person 3 (d) Training images for person 4

(e) Training images for person 5

(f) Average image

(g) Difference between each training set image and the average image

(h) Eigenfaces of the difference images

(i) Fisherfaces of the difference images

(j) Test set of 5 images

(k) Recognized face images with PCA

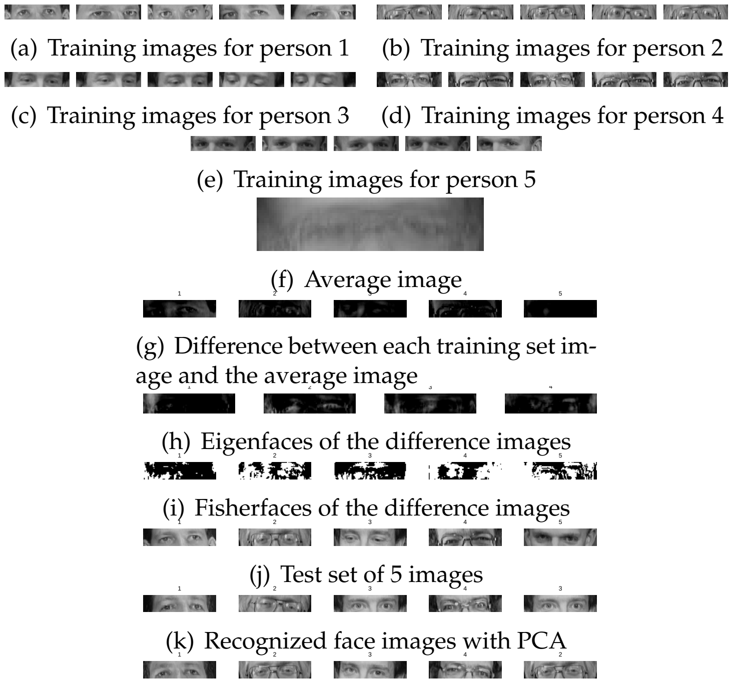
(l) Recognized face images with LDA

Figure 5.2: ORL forehead region example using both PCA and LDA (*Figure 5.1(a)*–*Figure 5.1(e)* are corresponding full-face images). The input to the system are the five sets of face images, each with five variations.

Figure 5.2 depicts how region-division works with only a single region at work, in this case, the forehead. Similar to the previous example, the figure depicts a small experiment conducted using a small set of images from the ORL database. Five training images are used for each person, and, then, for each person, one test image is used. Using PCA, all but the third person were recognized correctly. The third person (*Figure 5.2* (j), third image from left) was recognized incorrectly as the fifth person (*Figure 5.2* (e)). Using

LDA, on the other hand, all persons were recognized correctly.

Eye Example:



(a) Training images for person 1 (b) Training images for person 2

(c) Training images for person 3 (d) Training images for person 4

(e) Training images for person 5

(f) Average image

(g) Difference between each training set image and the average image

(h) Eigenfaces of the difference images

(i) Fisherfaces of the difference images

(j) Test set of 5 images

(k) Recognized face images with PCA

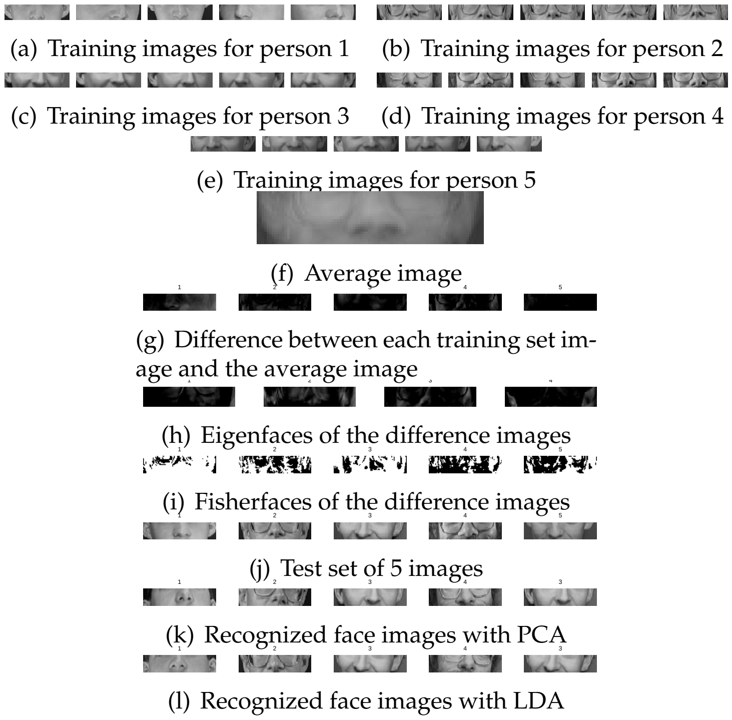
(l) Recognized face images with LDA

Figure 5.3: ORL eye region example using both PCA and LDA (*Figure 5.1(a)–Figure 5.1(e)* are corresponding full-face images). The input to the system are the five sets of face images, each with five variations.

Figure 5.3 depicts a small experiment conducted using a small set of eye images extracted from the ORL database. Five training eye images are used for each person, and, then, for each person, one test image is used. Using PCA, all but the fifth person were recognized correctly. The fifth person (*Figure 5.3* (j), fifth image from left) was recognized incorrectly as the third person (*Figure 5.3* (c)). Similarly, using LDA, all but the fifth person were recognized correctly. The fifth person (*Figure 5.3* (j), fifth image from left) was recognized incorrectly as the second person (*Figure 5.3* (b)).

Nose Example:

Figure 5.4 depicts a small experiment conducted using a small set of nose images extracted from the ORL database. Five training eye images are used for each person, and, then, for each person, one test image is used. Using PCA, all but the fifth person were



(a) Training images for person 1 (b) Training images for person 2

(c) Training images for person 3 (d) Training images for person 4

(e) Training images for person 5

(f) Average image

(g) Difference between each training set image and the average image

(h) Eigenfaces of the difference images

(i) Fisherfaces of the difference images

(j) Test set of 5 images

(k) Recognized face images with PCA

(l) Recognized face images with LDA

Figure 5.4: ORL nose region example using both PCA and LDA. The input to the system are the five sets of face images, each with five variations.

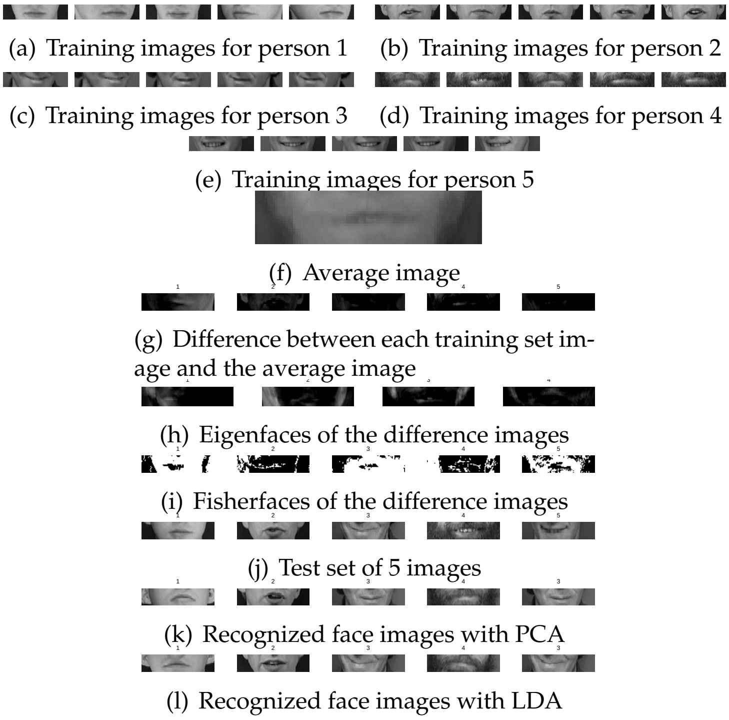
recognized correctly. The fifth person (*Figure 5.4* (j), fifth image from left) was recognized incorrectly as the third person (*Figure 5.4* (c)). Similarly, using LDA, all but the fifth person were recognized correctly. The fifth person (*Figure 5.4* (j), fifth image from left) was also recognized incorrectly as the third person (*Figure 5.4* (c)).

Mouth Example:

Figure 5.5 depicts a small experiment conducted using a small set of mouth images extracted from the ORL database. Five training eye images are used for each person, and, then, for each person, one test image is used. Using both PCA and LDA, all but the fifth person were recognized correctly. The fifth person (*Figure 5.5* (j), fifth image from left) was recognized incorrectly as the third person (*Figure 5.5* (c)).

Chin Example:

Figure 5.6 depicts a small experiment conducted using a small set of mouth images extracted from the ORL database. Five training eye images are used for each person, and, then, for each person, one test image is used. Using both PCA and LDA, all but the third



(a) Training images for person 1 (b) Training images for person 2

(c) Training images for person 3 (d) Training images for person 4

(e) Training images for person 5

(f) Average image

(g) Difference between each training set image and the average image

(h) Eigenfaces of the difference images

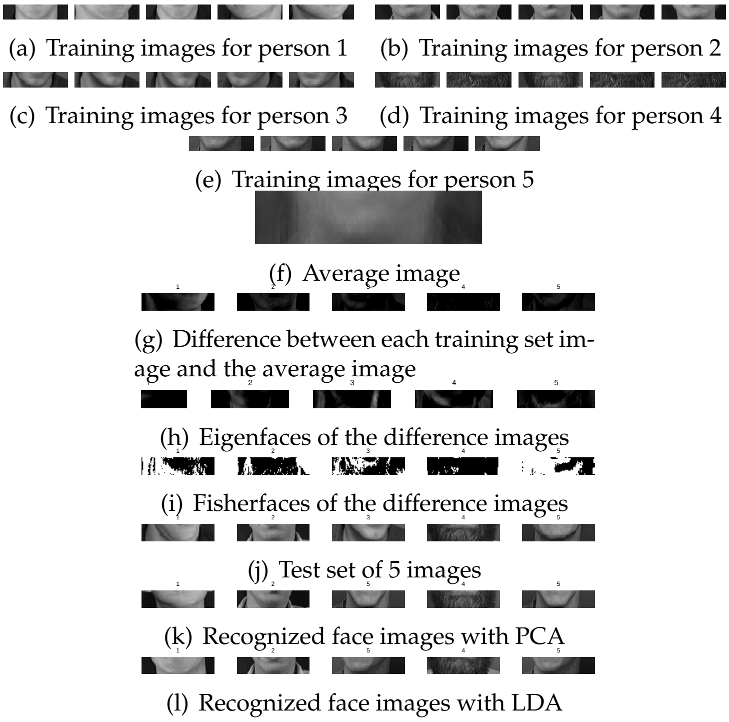
(i) Fisherfaces of the difference images

(j) Test set of 5 images

(k) Recognized face images with PCA

(l) Recognized face images with LDA

Figure 5.5: ORL mouth region example using both PCA and LDA. The input to the system are the five sets of face images, each with five variations.



(a) Training images for person 1 (b) Training images for person 2

(c) Training images for person 3 (d) Training images for person 4

(e) Training images for person 5

(f) Average image

(g) Difference between each training set image and the average image

(h) Eigenfaces of the difference images

(i) Fisherfaces of the difference images

(j) Test set of 5 images

(k) Recognized face images with PCA

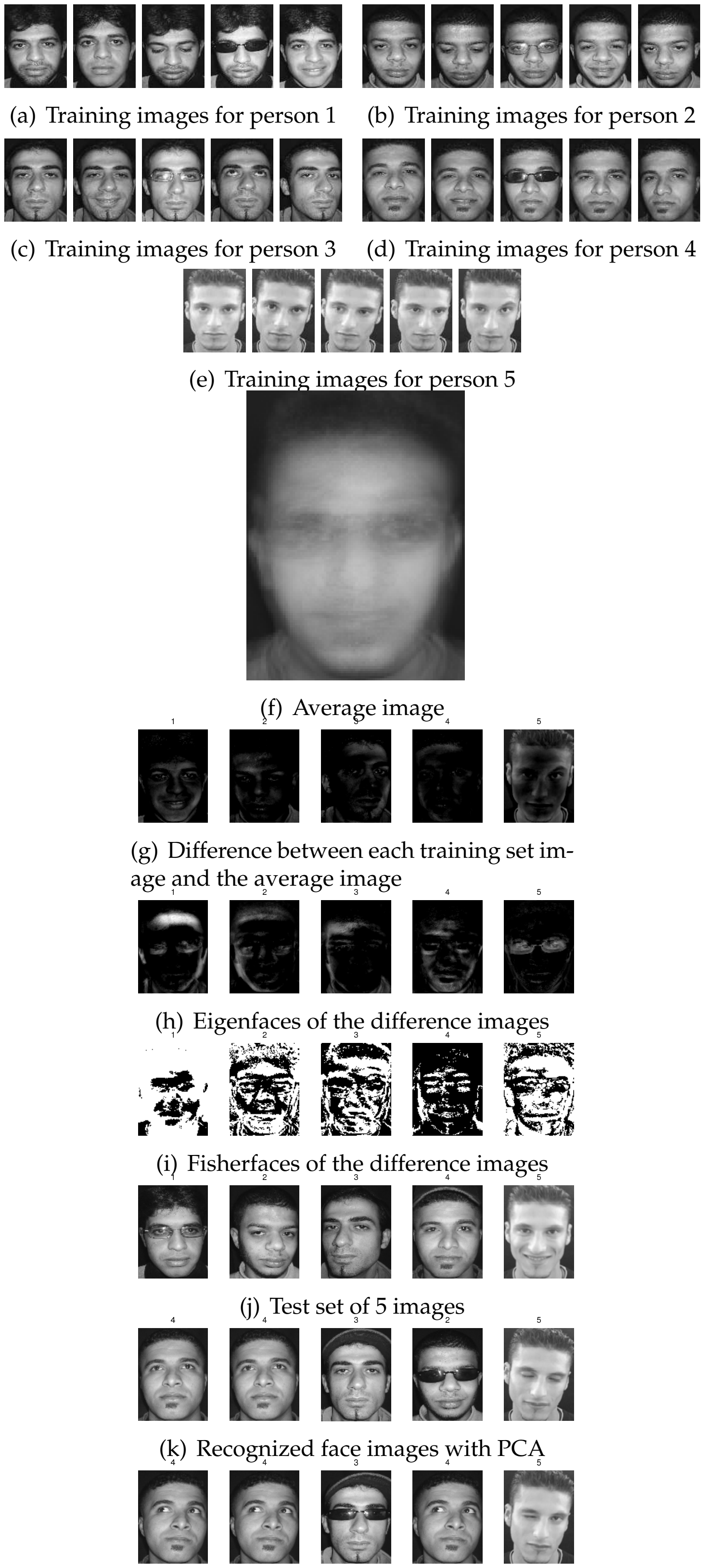
(l) Recognized face images with LDA

Figure 5.6: ORL chin region example using both PCA and LDA. The input to the system are the five sets of face images, each with five variations.

person were recognized correctly. The third person (*Figure 5.6* (j), third image from left) was recognized incorrectly as the fifth person (*Figure 5.6* (e)).

5.2.3 A Counter-Example Using the Classic Whole Face Approach

This section presents a counter-example of using the classic whole-face approach. The experiment run involved images of five persons from the NEU Face Image Database (NFID). For each person, the experiment used 5 training images and 5 test images. The recognition rate using PCA was $2/5$, and using LDA, the recognition rate was $3/5$. *Figure 5.7* is a summary of this experiment.



(a) Training images for person 1 (b) Training images for person 2

(c) Training images for person 3 (d) Training images for person 4

(e) Training images for person 5

(f) Average image

(g) Difference between each training set image and the average image

(h) Eigenfaces of the difference images

(i) Fisherfaces of the difference images

(j) Test set of 5 images

(k) Recognized face images with PCA

(l) Recognized face images with LDA

Figure 5.7: Counter-example where the full-face application of PCA and LDA fails to recognize the images from the NEU Face Image Database correctly. PCA recognition rate $= \frac{2}{5}$, LDA recognition rate $= \frac{3}{5}$

5.3 Simulation Results for PCA

This section presents the results of performing whole-face recognition using PCA on the ORL face image database and the NEU Face Image Database (NFID).

5.3.1 ORL Database and NEU Face Image Database (NFID)

Firstly, we used our NEU Face Image Database to perform PCA-based whole face recognition and compare the performance to that with the ORL database. In the ORL experiments, we used 10 images of 40 different individuals from the ORL database. We varied the number of training images from 1 to 9, so the number of test images varied between 9 and 1 always totaling 10. We applied the same to the NFID. *Table 5.1* shows the results of applying PCA-based recognition to both of these face image databases.

Table 5.1: PCA recognition rates for the NEU Face Image Database (NFID) and the ORL database

# training images	#test images	NFID	ORL
1	9	49.4	58.2
2	8	54.6	72.4
3	7	62.0	79.6
4	6	64.7	84.0
5	5	65.2	89.2
6	4	66.2	91.9
7	3	71.8	92.4
8	2	67.7	94.0
9	1	71.3	96.8

Figure 5.8 is a graph of the results given in *Table 5.1*. From *Figure 5.8* and *Table 5.1*, one can observe that recognition rates for the ORL database were very close. As the number of training images increased, the recognition results for both the ORL database and NFID increased, but the recognition rates for the ORL database were higher than that for NFID. Therefore, we can conclude from this analysis that applying whole face recognition on the NFID does not work well. That is, these results confirm that the NFID is more challenging database than the ORL due to different types of occlusions present in the images of the NFID.

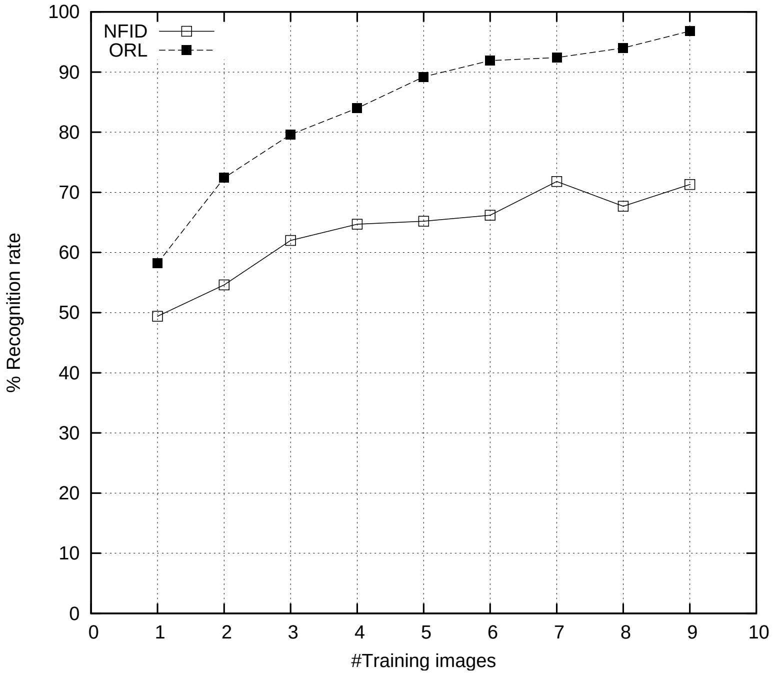


Figure 5.8: PCA recognition rates varying with respect to number of training images using the whole face as input (NEU Face Image Database (NFID),ORL database)

5.3.2 Region-based PCA Recognition

We also investigated how recognition rate would vary if we use only one of the five facial parts at a time for recognition in order to gain a perspective on the potential contribution of each facial part to the overall recognition rate.

ORL Database

Table 5.2 presents for recognition performance of individual facial parts in the case of the ORL database as well as the whole face. We can see that the chin region provided the best results with 88.5% recognition rate when 9 training and 1 test images were used. Still, the recognition performance using the whole face was better with 96.8%. Note that the results for the whole face are repeated here from *Table 5.1*.

Figure 5.9 is a graph of the results given in *Table 5.3*. The *Figure 5.9* and *Table 5.3* present PCA recognition rates varying with respect to the number of training images using the whole face, five facial regions with majority voting, and using six regions (5 facial regions and the classic whole face as a sixth region) with majority voting for the ORL database. From The *Figure 5.9* and *Table 5.3*, one can observe that the two region-

Table 5.2: PCA recognition results for the ORL database using the whole face alone and each facial part on its own. Thus, this table provides a perspective on the potential relative contribution of each part to the overall recognition.

#training images	#test images	Whole Face	Forehead Region	Eye Region	Nose Region	Mouth Region	Chin Region
1	9	58.2	48.3	36.1	33.6	31.4	42.2
2	8	72.4	61.9	46.0	46.5	44.9	58.7
3	7	79.6	69.1	53.1	55.3	51.9	66.5
4	6	84.0	73.0	60.8	59.4	56.1	72.6
5	5	89.2	77.5	63.4	66.3	60.8	76.7
6	4	91.9	80.3	66.0	69.2	63.8	79.8
7	3	92.4	80.8	69.0	72.7	68.8	83.6
8	2	94.0	84.0	72.3	77.3	71.3	86.9
9	1	96.8	82.5	72.3	78.3	75.3	88.5

division approaches did not perform any better than the classic whole-face approach.

Table 5.3: PCA recognition rates for the ORL Database

#training images	#test images	Whole Face	5 Regions (MV)	6 Regions (MV)
1	9	58.2	48.6	52.7
2	8	72.4	65.7	72.2
3	7	79.6	74.7	78.7
4	6	84.0	82.9	83.3
5	5	89.2	86.1	88.2
6	4	91.9	88.6	91.3
7	3	92.4	91.9	94.3
8	2	94.0	94.4	95.9
9	1	96.8	95.3	96.8

5.3.3 NEU Face Image Database (NFID)

Table 5.4 presents for recognition performance of five individual facial parts in the case of the NEU Face Image Database (NFID) as well as the whole face. As in the above experiment, the chin region provided the best results with 78% recognition rate when 9 training and 1 test images were used. As stated previously, the recognition performance using the whole face with 9 training images were a mere 71.3%. Note that the results for the whole face are repeated here from *Table 5.1*.

Figure 5.10 is a graph of the results given in *Table 5.5*. The *Figure 5.10* and *Table 5.5* present PCA recognition rates varying with respect to the number of training images us-

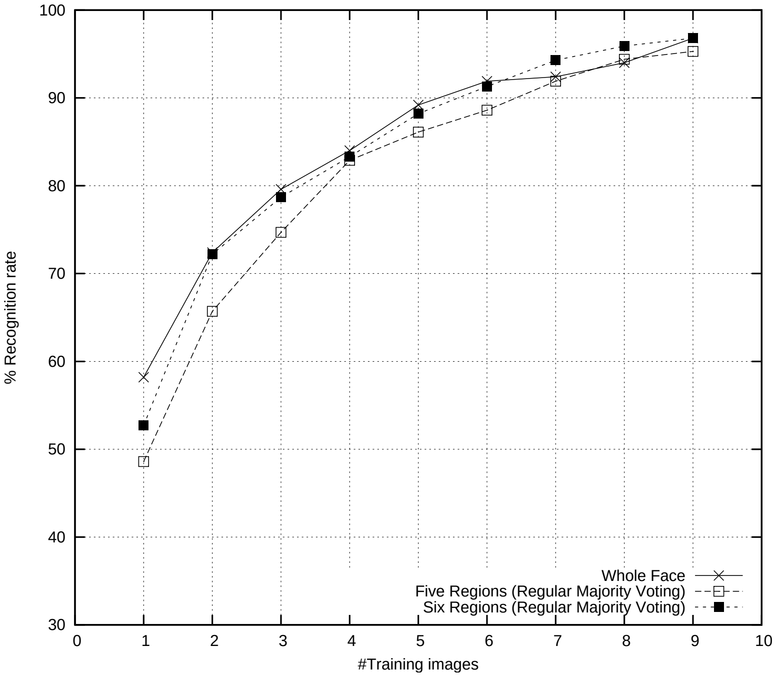


Figure 5.9: PCA recognition rates varying with respect to number of training images using the whole face, five-regions of the face, and five regions of the face combined with the whole face as input (ORL database)

ing the whole face, five facial regions with majority voting, and using six regions (5 facial regions and the classic whole face as a sixth region) with majority the NEU Face Image Database (NIFD). From *Figure 5.10* and *Table 5.5*, one can observe that the recognition rates for the five facial regions with majority voting and using six regions with majority voting better than the classic whole-face. Tabletabl:overall-results-ttest depicts the results of statistical significance tests using the student's t-test.

5.4 Simulation results for LDA

This section presents the results of performing whole-face recognition using LDA on the ORL face image database and the NEU Face Image Database (NFID).

5.4.1 ORL Database and NEU Face Image Database (NFID)

Firstly, we used our NEU Face Image Database (NFID) to perform LDA-based whole-face recognition and compare the performance to that with the ORL database. In the ORL experiments, we again used 10 images of 40 different individuals from the ORL

Table 5.4: PCA recognition results for the NEU Face Image Database using the whole face alone and each facial part on its own. Thus, this table provides a perspective on the potential relative contribution of each part to the overall recognition.

#training images	#test images	Whole Face	Forehead Region	Eye Region	Nose Region	Mouth Region	Chin Region
1	9	49.4	32.1	35.6	30.7	27.6	51.0
2	8	54.6	36.8	39.3	35.4	35.2	61.0
3	7	62.0	38.2	40.5	41.8	39.3	66.4
4	6	64.7	40.6	44.7	42.3	39.8	68.6
5	5	65.2	42.3	47.2	46.5	42.0	69.9
6	4	66.2	42.5	48.2	47.0	46.0	73.0
7	3	71.8	43.3	49.1	50.2	48.0	75.8
8	2	67.7	44.3	51.0	49.3	49.0	77.7
9	1	71.3	38.7	52.7	54.0	48.0	78.0

Table 5.5: PCA recognition rates for the NFID

#training images	#test images	Whole Face	5 Regions (MV)	6 Regions (MV)
1	9	49.4	39.9	44.0
2	8	54.6	51.6	56.3
3	7	62.0	63.4	59.0
4	6	64.7	64.2	68.8
5	5	65.2	67.3	69.5
6	4	66.2	68.7	71.0
7	3	71.8	74.0	78.0
8	2	67.7	73.7	77.0
9	1	71.3	78.0	81.3

database. We varied the number of training images from 1 to 9, i.e., the number of test images varied between 9 and 1. We applied the same to the NFID. *Table 5.6* shows the results of applying LDA-based recognition to both of these face image databases.

As in the case of PCA (*Table 5.1*), results in *Table 5.6* and *Figure 5.11* demonstrate that higher recognition rates were possible with the ORL database than with the NEU Face Image Database (NFID). This also confirms that the NEU Face Image Database (NFID) involves difficulties that severely affect recognition performance. As may be observed from *Figure 4.3*, face in the NEU Face Image Database (NFID) contain occlusions or obstacles such as a hat, sunglasses, and beards for the same faces.

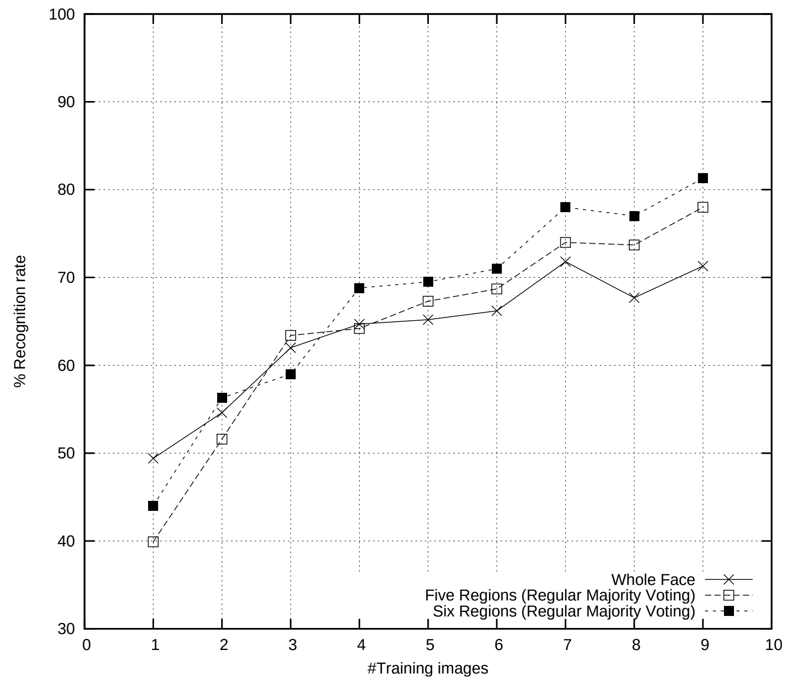


Figure 5.10: PCA recognition rates varying with respect to number of training images using the whole face, five-regions of the face, and five regions of the face combined with the whole face as input (NFID)

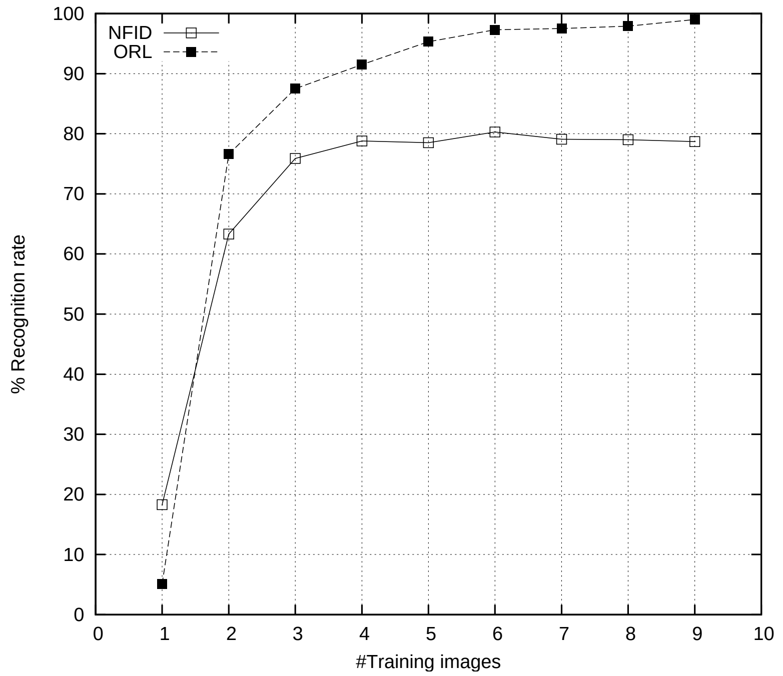


Figure 5.11: LDA recognition rates varying with respect to number of training images using the whole face as input (ORL database, NEU Face Image Database (NFID))

Table 5.6: LDA recognition rates for the NEU Face Image Database (NFID) and the ORL Database

#training images	#test images	NFID	ORL
1	9	18.3	5.1
2	8	63.3	76.6
3	7	75.9	87.5
4	6	78.8	91.5
5	5	78.5	95.3
6	4	80.3	97.3
7	3	79.1	97.5
8	2	79.0	97.9
9	1	78.7	99.0

5.4.2 Region-based LDA Recognition

In this section, we present an analysis similar to what was presented in *Section 5.3.2* except that LDA is used in this case instead of PCA.

ORL Database

Table 5.7 presents the recognition performance of individual facial parts in the case of the ORL database. The recognition rate obtained when the full face is used is duplicated here from *Table 5.6* for comparison. Interestingly, using only the forehead region, the recognition rate was 94.3% when 9 images were used for training, and, using the whole face, the recognition rate was 99%.

Table 5.7: LDA recognition results for the ORL Database using the whole face alone and each facial part on its own. Thus, this table provides a perspective on the potential relative contribution of each part to the overall recognition.

#training images	#test images	Whole Face	Forehead Region	Eye Region	Nose Region	Mouth Region	Chin Region
1	9	5.1	2.4	3.9	3.1	4.1	3.4
2	8	76.6	70.3	44.0	36.8	38.1	47.8
3	7	87.5	79.0	57.0	51.5	54.6	62.5
4	6	91.5	85.8	65.9	58.8	63.0	71.9
5	5	95.3	89.0	70.8	67.8	68.8	77.5
6	4	97.3	91.0	75.8	71.1	75.6	81.1
7	3	97.5	93.2	78.7	75.9	79.5	83.0
8	2	97.9	93.5	83.0	79.3	82.4	86.6
9	1	99.0	94.3	83.8	83.0	81.8	86.5

Figure 5.12 is a graph of the results given in *Table 5.8*. The *Figure 5.12* and *Table 5.8*

present the LDA-based recognition rates varying with respect to the number of training images using the classic whole-face, five facial regions with majority voting and using six regions with majority for the ORL database. From *Figure 5.12* and *Table 5.8*, one can observe that the two region-division approaches did not perform better than the classic whole-face approach.

Table 5.8: LDA recognition rates for the ORL Database

#training images	#test images	Whole Face	5 Regions (MV)	6 Regions (MV)
1	9	5.1	4.2	6.1
2	8	76.6	60.4	68.0
3	7	87.5	76.9	83.9
4	6	91.5	85.0	89.1
5	5	95.3	90.7	93.2
6	4	97.3	92.6	95.8
7	3	97.5	94.5	97.4
8	2	97.9	96.4	98.1
9	1	99.0	96.5	98.5

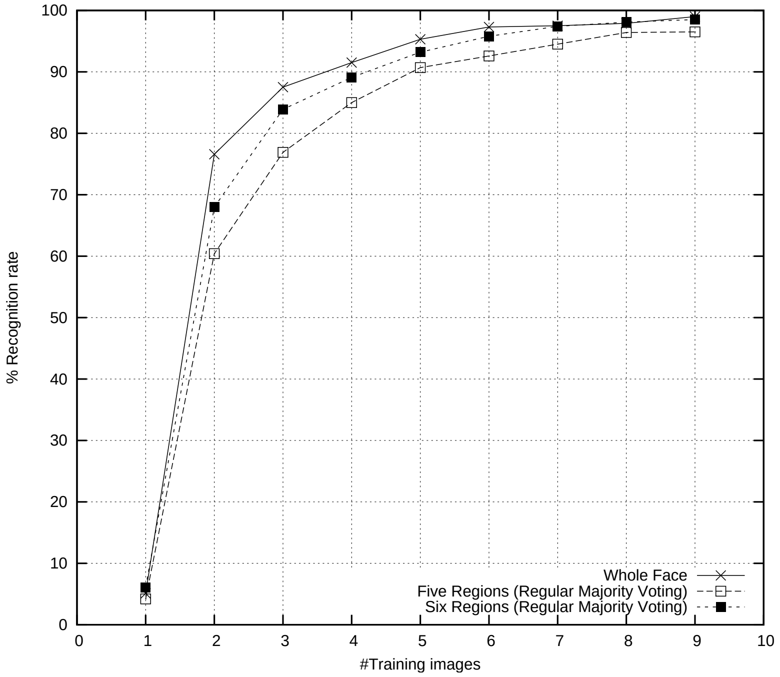


Figure 5.12: LDA recognition rates varying with respect to number of training images using the whole face, five-regions of the face, and five regions of the face combined with the whole face as input (ORL)

NEU Face Image Database (NFID)

Table 5.9 presents for recognition performance of five individual facial parts when the NEU Face Image Database (NFID) is used. The whole-face performance is duplicated here from *Table 5.6* for comparison. The performance obtained using only the chin region was higher than the performance obtained for all cases. Similarly, using only the chin region, in the experiment with 9 training images, the recognition rate was 92.7% compared to 78.7% when the whole face was used.

Table 5.9: LDA recognition results for the NEU Face Image Database using the whole face alone and each facial part on its own. Thus, this table provides a perspective on the potential relative contribution of each part to the overall recognition.

#training images	#test images	Whole Face	Forehead Region	Eye Region	Nose Region	Mouth Region	Chin Region
1	9	18.3	16.1	14.7	16.4	16.5	19.2
2	8	63.3	48.7	40.6	40.5	42.6	72.8
3	7	75.9	57.4	47.0	51.5	54.2	82.8
4	6	78.8	61.6	51.4	54.4	58.3	86.8
5	5	78.5	64.4	51.7	56.5	63.2	89.6
6	4	80.3	67.0	52.8	56.3	65.3	91.2
7	3	79.1	70.9	55.8	57.8	63.1	91.1
8	2	79.0	67.3	57.3	59.3	60.7	92.3
9	1	78.7	76.7	50.0	60.0	67.3	92.7

Figure 5.13 is a graph of the results given in *Table 5.10*. The *Figure 5.13* and *Table 5.10* present the LDA-based recognition rates varying with respect to the number of training images using the classic whole-face approach, five facial regions with majority voting approach, and six regions with majority voting approach for the NIFD. From *Figure 5.13* and *Table 5.10*, one can observe that the recognition rates for the five facial regions with majority voting and using six regions with majority voting were statistically better than the classic whole-face, as depicted in Tabletabl:overall-results-ttest.

5.5 Summary of Results for the NEU Face Image Database (NFID)

Table 5.11 depicts the summary of the recognition results for all PCA- and LDA-based experiments using a single training image and 9 test images. The two region-division approaches did not improve the performance compared to the whole-face approach. As expected, since the number of training images was low, PCA performed relatively better

Table 5.10: LDA recognition rates for the NFID

#training images	#test images	Whole Face	5 Regions (MV)	6 Regions (MV)
1	9	18.3	20.9	21.9
2	8	63.3	60.7	64.5
3	7	75.9	78.6	81.9
4	6	78.8	83.9	87.2
5	5	78.5	85.6	88.3
6	4	80.3	84.3	87.5
7	3	79.1	89.3	92.0
8	2	79.0	88.7	89.3
9	1	78.7	94.0	95.3

than LDA.

Table 5.11: Comparison of the recognition rate results for the NEU Face Image Database (NFID) between the whole-face approach and two region-division approaches with majority voting using the PCA and LDA methods with 1 training image and 9 test image. One of the region-division approaches uses the five-regions of the face. The second approach uses the five-regions of the face but also considers the whole face as a sixth region.

Technique	PCA	LDA
Whole Face	49.4	18.3
Five Regions (Majority Voting)	39.9	20.9
Six Regions (Majority Voting)	44.0	21.9

Table 5.12 depicts the summary of the recognition results for all PCA- and LDA-based experiments using 5 training images and 5 test images. The two region-division approaches did not improve the performance compared to the whole-face approach using PCA. In the case of LDA, however, both region-division approaches improved over the classic whole-face approach with 99% statistical confidence (See *Table 5.14*).

Table 5.12: Comparison of the recognition rate results for the NEU Face Image Database (NFID) between the whole-face approach and two region-division approaches with majority voting using the PCA and LDA methods with 5 training image and 5 tests image. One of the region-division approaches uses the five regions of the face. The second approach uses the five-regions of the face but also considers the whole face as a sixth region.

Technique	PCA	LDA
Whole Face	65.2	78.5
Five Regions (Majority Voting)	67.3	85.6
Six Regions (Majority Voting)	69.5	88.3

Table 5.13 depicts the summary of the recognition results for all PCA- and LDA-based

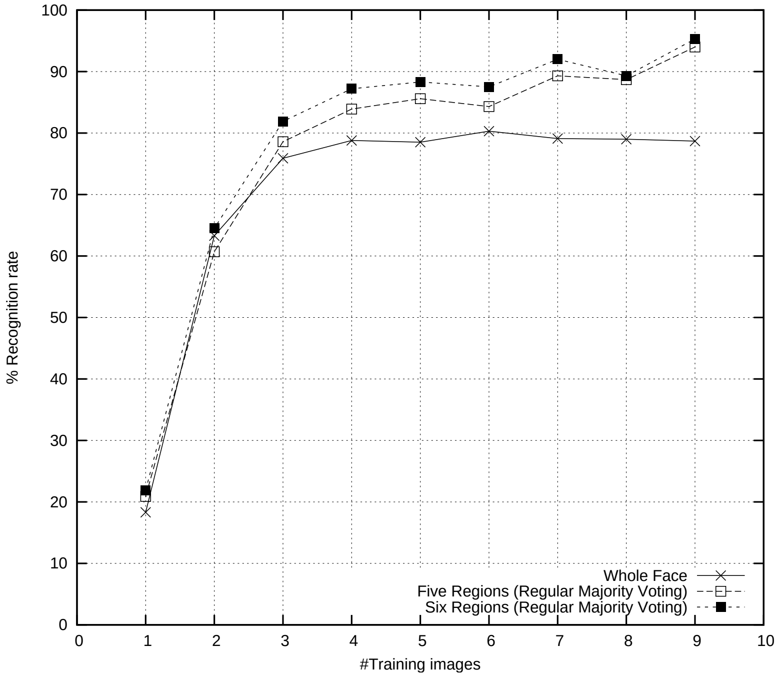


Figure 5.13: LDA recognition rates varying with respect to number of training images using the whole face, five-regions of the face, and five regions of the face combined with the whole face as input (NFID)

experiments using 9 training images and a single test image.

The two region-division approaches did not improve the performance compared to the whole-face approach using PCA. In the case of LDA, however, both region-division approaches improved over the classic whole-face approach with 99% statistical confidence (See *Table 5.14*).

Table 5.13: Comparison of the recognition rate results for the NEU Face Image Database (NFID) between the whole-face approach and two region-division approaches with majority voting using the PCA and LDA methods with 9 training image and 1 test image. One of the region-division approaches uses the five regions of the face. The second approach uses the five-regions of the face but also considers the whole face as a sixth region.

Technique	PCA	LDA
Whole Face	71.3	78.7
Five Regions (Majority Voting)	78.0	94.0
Six Regions (Majority Voting)	81.3	95.3

Table 5.14 presents an overall statistical comparison of the performance using five facial regions with majority voting and using six regions (5 facial regions and the whole face as a sixth region) with majority voting versus the classic whole-face approach. The table presents the *t-test* results for varying number of training and test cases using PCA

Table 5.14: Statistical two-tailed unpaired *t-test* comparison of the recognition results for the NEU Face Image Database (NFID) between the classic whole-face approach and two region-division approaches with majority voting approach using the PCA and LDA feature extraction methods with different sets of training and test sets. One of the region-division approaches uses the five-regions of the face. The second approach uses the five-regions of the face but also considers the whole face as a sixth region. The value in each table cell indicates the confidence level of the t-test based on 10 test averages. Each of the 10 tests run have been generated randomly. Negative values indicate that the classic whole-face approach performed better than the technique in question with the given confidence level reversed. A horizontal dash (—) indicates that the performance difference for the corresponding test was not statistically significant.

	PCA		LDA	
	5 Regions vs Whole Face	6 Regions vs Whole Face	5 Regions vs Whole Face	6 Regions vs Whole Face
1 training, 9 test	-99%	-85%	—	—
5 training, 5 test	—	85%	99%	99%
9 training, 1 test	—	85%	99%	99%

and LDA feature extraction methods.

When only a single training image and 9 test images (per person) were used with PCA, the classic whole-face approach outperformed the five-region approach with 99% confidence. In a similar vein, the whole face performance was numerically (but not necessarily statistically) better than the six-region approach. Hence these results have been expressed as negative values in *Table 5.14*.

In the case of LDA with a single training image and 9 test images per person, there is no statistically significant difference between the performance of the five-region approach and the whole-face approach, and so is the case between the performance of the six-region approach and the whole-face approach.

As the number of training images is increased, we start observing this negative performance being reversed in favor of our region-division approaches except in the case of the five-region approach using PCA. Numerically (but not necessarily statistically), the performance of our region-division approaches was better than the classic whole-face approach using LDA both when 5 training images were used and 9 training images were used.

Unlike in the case of PCA, with LDA, our region-division approaches outperformed the classic whole-face approach with a statistical 99% confidence both when 5 training images were used and 9 training images were used. Therefore, these results indicate that our region-division approach can improve face recognition over the classic whole face approach using LDA.

5.6 Summary

This chapter presented a series of results providing a perspective on the region division with majority voting scheme presented in this thesis and overall results of running the whole-face approach and our approach on the NEU Face Image Database. We used two well-known feature extraction algorithms, namely the PCA and LDA to get a better understanding of the differences between the classic whole-face approach to recognition and the region-division approach reported in this thesis.

CHAPTER 6

CONCLUSIONS

Our work focused on applying PCA and LDA feature extraction algorithms using the ORL (Olivetti Research Laboratories) database and our own NEU Face Image Database (NFID), which we prepared in order to demonstrate the performance of the region-division approach applied to face recognition in this thesis. The idea behind creating a new database was to make the recognition task more challenging by making the face images harder to recognize by adding some obstacles such as a hat, glasses or sunglasses than the images available in the ORL database.

With this challenging database and from the results in *Chapter 5*, we can conclude that the performance of the classic approach of using PCA and LDA with the face as a whole was not good enough. We demonstrated the weakness of this method with the NEU Face Image Database we created.

Since, for some examples, considering the whole face failed when the face is occluded or modified in an image, we hypothesized that dividing the face into multiple regions and using the best matching parts as a guide for recognition would perform better. In our approach, each face image is divided into five regions, namely the forehead region, eye region, nose region, mouth region, and chin region. So we created five new databases from the original NEU Face Image Database. Then the processed values of these 5 databases were combined using majority voting, as described in *Chapter 3*.

The results reported in *Chapter 5* show that, for the NEU Face Image Database, the two region-division approaches with majority voting improved, with a statistical 99% confidence, the recognition rate with 5 training images and 5 test images per person compared to the the classic approach with the LDA feature extraction method.

The recognition rates with PCA for five regions increased from 65.2% to 67.3%, and the recognition rate with LDA increased from 78.5% to 85.6%, and for six regions with PCA the recognition rate increased to 69.5% and for LDA to 88.3%. and with 9 training images and 1 test image per person, for the five regions approach with PCA the recognition rate increased from 71.3% to 78.0% and the recognition rate with LDA increased from 78.7% to 94.0%, and, for the six-region approach with PCA, it increased to 81.3%, and, for LDA, the recognition rate increased to 95.3%. From these results, we can also conclude that the LDA algorithm performed better than PCA algorithm on our challenging face image database. On the other hand, the region-division approaches reported in this thesis did not perform any better than the whole-face approach both under PCA and LDA for the ORL database.

6.1 Future Work

Our NEU Face Image Database consists of only 15 persons with 10 face images per person. This database is limited, and it can be enlarged and improved to contain more persons and possibly more images per person. Another improvement to our work may involve applying some other algorithms for face recognition and compare them with both PCA and LDA algorithms such as Support Vector Machines (SVM) and Neural Networks. These performance of these algorithms can be evaluated using our region-division approach, and the results can be compared with the obtained results in this thesis. Moreover, during the preparation of the NEU databases, the five facial regions where cropped and prepared manually. The cropping and resizing of the regions can be done automatically as in [Kotani et al., 2006].

REFERENCES

Peter N. Belhumeur, Joao P. Hespanha, and David J. Kriegman. Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997. ◇ pp: 3, 38

W. W. Bledsoe. The Model Method in Facial Recognition. Technical Report Technical Report PRI:15, Panoramic Research, Inc., Palo Alto, CA, August 1966. ◇ pp: 5

R. Bunschoten. A fully vectorized function that computes the euclidean distance matrix between two sets of vectors. Accessed on June 01, 2008, 1999. `<http://www.mathworks.com/matlabcentral/files/71/distance.m>`. ◇ pp: 79

P.J. Burt. Smart sensing within a pyramid vision machine. *Proceedings of the IEEE*, 76(8): 1006–1015, August 1988. ◇ pp: 6

Giuseppe Cardillo. Student's t-test for unpaired or paired samples. Accessed on June 01, 2008, 2006. `<http://www.mathworks.com/matlabcentral/files/12699/testt.m>`. ◇ pp: 79

R. Chellappa, C.L. Wilson, and S. Sirohey. Human and Machine Recognition of Faces: A Survey. *Proceedings of the IEEE*, 83(5):705–741, May 1995. ◇ pp: 2

Li-Fen Chen, Hong-Yuan Mark Liao, Ming-Tat Ko, Ja-Chen Lin, and Gwo-Jong Yu. A New LDA-based Face Recognition System which can Solve the Small Sample Size Problem. *Pattern Recognition*, 33(10):1713–1726, October 2000. ◇ pp: 28

Yong-Qing Cheng, Yong-Ming Zhuang, and Jing-Yu Yang. Optimal fisher discriminant analysis using the rank decomposition. *Pattern Recognition*, 25(1):101–111, 1992. ◇ pp: 28

Alaa Eleyan. Face recognition using eigenfaces and backpropagation neural networks. Master's thesis, Near East University, 2004. ⋄ pp: 9

Alaa Eleyan and Hasan Demirel. PCA and LDA Based Face Recognition Using Feedforward Neural Network Classifier. In Bilge Günsel, Anil K. Jain, A. Murat Tekalp, and Bülent Sankur, editors, *Proceedings of the International Workshop on Multimedia Content Representation, Classification and Security (MRCS 2006)*, volume 4105 of *Lecture Notes in Computer Science*, pages 199–206, Istanbul, Turkey, September 2006. Springer. ⋄ pp: 28

G.D. Finlayson, J. Dueck, B.V. Funt, and M.S. Drew. Color Eigenfaces. In *Proceedings of the IEEE Third International Workshop on Image and Signal Processing*, pages 607–610, November 1996. ⋄ pp: 3, 17

R. A. Fisher. The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, 7:179–188, 1936. ⋄ pp: 3

Ralph Gross, Jianbo Shi, and Jeffrey Cohn. Quo vadis face recognition? In *Third Workshop on Empirical Evaluation Methods in Computer Vision*, pages 119–132, Kauai, Hawaii, USA, December 2001. ⋄ pp: 11

B. Hemery, J. Mahier, M. Pasquet, and C. Rosenberger. Face authentication for banking. In *Proceedings of The First International Conference on Advances in Computer-Human Interaction (ACHI 2008)*, pages 137–142, feb 2008. ⋄ pp: 14

Rui Huang, Qingshan Liu, Hanqing Lu, and Songde Ma. Solving the Small Sample Size Problem of LDA. In *16th International Conference on Pattern Recognition (ICPR'02)*, volume 3, pages 29–32, Los Alamitos, CA, USA, 2002. IEEE Computer Society. ⋄ pp: 27, 28

Anil K. Jain, Robert P. W. Duin, and Jianchang Mao. Statistical Pattern Recognition: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, January 2000. ⋄ pp: 6

Mohamed S. Kamel, Helen C. Shen, Andrew K. C. Wong, and Radu I. Campeanu. System for the recognition of human faces. *IBM Systems Journal*, 32(2):307–320, April 1993. ⋄ pp: 6

M. Kirby and L. Sirovich. Application of the Karhunen-Loeve Procedure for the Characterization of Human Faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):103–108, 1990. ⋄ pp: 17

Koji Kotani, Qiu Chen, Feifei Lee, and Tadahiro Ohmi. Region-division vector quantization histogram method for human face recognition. *Intelligent Automation and Soft Computing*, 12(3):257–268, 2006. ⋄ pp: 6, 73

S.K. Nayar, H. Murase, and S.A. Nene. Parametric Appearance Representation. In *Early Visual Learning*, pages 131–160. Oxford University Press, February 1996. ⋄ pp: 3, 17

A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, Seattle, WA, USA, June 1994. ⋄ pp: 17, 38

Kenneth Rose. Deterministic Annealing for Clustering, Compression, Classification, Regression, and Related Optimization Problems. *Proceedings of the IEEE*, 86(11):2210–2239, November 1998. ⋄ pp: 6

K. Sirlantzis, S. Hoque, and M. C. Fairhurst. Diversity in multiple classifier ensembles based on binary feature quantisation with application to face recognition. *Applied Soft Computing*, 8(1):437–445, 2008. ⋄ pp: 14

Daniel L. Swets and Juyang Weng. Using Discriminant Eigenfeatures for Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):831–836, 1996. ⋄ pp: 3

Koichi Tanaka, Mustuo Sano, Shuichi Ohara, and Masashi Okudaira. A parametric template method and its application to robust matching. In *2000 Conference on Computer Vision and Pattern Recognition (CVPR 2000)*, pages 1620–1627, Hilton Head, SC, USA, June 2000. ⋄ pp: 6

Q. Tian, M. Barbero, Z.H. Gu, , and S.H. Lee. Image Classification by the Foley-Sammon Transform. *Optical Engineering*, 25(7):834–840, 1986. ⋄ pp: 28

Antonio Trujillo-Ortiz. Power estimation of a performed student's t test about mean(s). Accessed on June 01, 2008, 2003. <http://www.mathworks.com/matlabcentral/files/2907/powerStudent.m>. ◇ pp: 79

M.A. Turk and A.P. Pentland. Face recognition using eigenfaces. In *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 586–591, Maui, Hawaii, USA, June 1991. ◇ pp: 3, 17, 38

ORL Website. Database of faces (formerly ”the orl database of faces”). Accessed on June 01, 2008, 1994. <http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>. ◇ pp: 42

Yanjun Yan and Lisa Ann Osadciw. Fusion for component based face recognition. In *Proceedings of CISS 07 (CDROM)*, Johns Hopkins University, Baltimore, Maryland, USA, March 2007. ◇ pp: 14

Hong Yang and Yiding Wang. A lbp-based face recognition method with hamming distance constraint. *Image and Graphics, 2007. ICIG 2007. Fourth International Conference on*, pages 645–649, aug 2007. ◇ pp: 14

Jian Yang and Chengjun Liu. A general discriminant model for color face recognition. In *Proceedings of the IEEE 11th International Conference on Computer Vision (ICCV 2007)*, pages 1–6, oct 2007. ◇ pp: 14

W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Computing Surveys*, 35(4):399–458, 2003. ◇ pp: 2

APPENDICES

APPENDIX A

Matlab Source Code

This appendix contains the Matlab source code that implements the ideas presented in this thesis. For conducting student's t-tests, we used a function named `testt()` that is publicly available [Cardillo, 2006]. This function depends on the `powerStudent()` function [Trujillo-Ortiz, 2003]. We also use a function called `distance()` that computes the Euclidean distance matrix between two sets of vectors [Bunschoten, 1999] –this function has been renamed to `euclideanDistance()` in our source code.

All program listings in this appendix have been reformatted from their original in order to fit their text within the available text area.

A.1 computeAverageImage.m

```
function avgImage = computeAverageImage( trainsets, trainSetSize,
                    numRuns, curRun, numPersons, numImagesPerPerson,
                    dirPath, imageXdim, imageYdim )
   % Keeps track of which row to pick. The row format is such that
   % each test case is listed one after the other for each person.
   % Therefore, there are a total of N x M rows where N is the number
   % of persons and M is the value of numRuns
   %
   trainRow = curRun;
   I = zeros( imageXdim, imageYdim );
   for i=1:numPersons
      %fprintf(1, '\n>>> Averaging [Run %d, Person %d]\n', curRun, i );
      curTrainVector = trainsets( trainRow, 1:trainSetSize );
      for j=1:trainSetSize
         % Specific index of the training image for the current person
         %
         trainval = curTrainVector( j );
         % Name of the current training image
         %
         facefile = [ dirPath 'face' int2str( trainval +
                      ((i - 1) * numImagesPerPerson) ) '.bmp'];

         % Read the input face image and add it to the total
         %
         [X, MAP] = imread( facefile );
         X=ind2gray( X, MAP );
         I = I + double( X );
      end
      trainRow = trainRow + numRuns;
   end
   % Compute the average face from all training images in the current
   % training set
   avgImage = I / (numPersons * trainSetSize);
end
```

A.2 computeTrainDifference.m

```
function A = computeTrainDifference( trainsets, curRun, numRuns,
             numPersons, trainSetSize, numImagesPerPerson, avgImage,
             imageXdim, imageYdim, dirPath )
   % Computes the difference between the average training image and
   % each of the training images for the current run and then compute
   % the covariance matrix from this difference
   %
   vectorIndex=curRun;
   for personID=1:numPersons
      curTrainVector = trainsets( vectorIndex, 1:trainSetSize );
      for curTrainIndex=1:trainSetSize
         % Generate the name of the current training image
         %
         trainImageIndex = curTrainVector( curTrainIndex );
         facefile = [ dirPath 'face' int2str( trainImageIndex +
                      ((personID - 1) * numImagesPerPerson) ) '.bmp'];
         [X, MAP] = imread( facefile );
         X = ind2gray( X, MAP );
         diff = double( X ) - avgImage;
         newDiffVect = reshape( diff, 1, (imageXdim * imageYdim) );
         % Create a column matrix
         diffTransposed = transpose( newDiffVect );
         ai = ((personID - 1) * trainSetSize) + curTrainIndex;
         A( :, ai ) = diffTransposed; % Covariance matrix holder
      end
      vectorIndex = vectorIndex + numRuns;
   end
end
```

A.3 generateTestAndTrainCases.m

```
function [testsets, trainsets] = generateTestAndTrainCases(
         numPersons, numRuns, testSetSize, numImagesPerPerson )
   %rand( 'seed', 1000 )
   testsets=[];
   trainsets=[];
   imageIDs=randperm( numImagesPerPerson );
   combinations=nchoosek( imageIDs, testSetSize );
   num_combinations = length( combinations );
   for i=1:numPersons
      curtestset=[];
      curtrainset=[];
      for j=1:numRuns
         pick=randomval( num_combinations );
         curtest=[];
         for col=1:testSetSize
            curtest=[ curtest combinations( pick, col )];
         end
         curtrain=setdiff( imageIDs, curtest );
         curtestset=[ curtestset; curtest ];
         curtrainset=[ curtrainset; curtrain ];
      end
      testsets=[testsets; curtestset];
      trainsets=[trainsets; curtrainset];
   end
end
```

A.4 mainmain.m

```
% Main program for running all tests
%
modeName='UNKNOWN_Mode'
if (mode == 1) modeName='PCA';
elseif (mode == 2) modeName='LDA';
end

methodName='UNKNOWN_Method'
if (method == 1) methodName='Weighted Majority Voting';
elseif (method == 2) methodName='Regular Majority Voting';
end

faceType='';
if (regionSelection(1) == 1) faceType=[faceType 'forehead ']; end
if (regionSelection(2) == 1) faceType=[faceType 'eye ']; end
if (regionSelection(3) == 1) faceType=[faceType 'nose ']; end
if (regionSelection(4) == 1) faceType=[faceType 'mouth ']; end
if (regionSelection(5) == 1) faceType=[faceType 'chin ']; end
if (regionSelection(6) == 1) faceType=[faceType 'full ']; end

outfilename = ['results.' databaseName '.' modeName '.' methodName '.'
               faceType '.' fileinfo '.txt'];
outfilename = deblank( outfilename );
resultout = fopen ( outfilename, 'wt' );

fprintf( resultout, 'LaTeX>%% %s Experiments (%s, %s, regions=[%s])\n',
         modeName, databaseName, methodName, faceType );

for numTest=(numImagesPerPerson - 1):-1:1
   performance=[];
   decisions=[];
   % Generate the test and training sets that will be used in all
   % testing so that our comparisons are standardized
   %
   [testsets, trainsets]=generateTestAndTrainCases( numPersons,
                         numRuns, numTest, numImagesPerPerson );
   numTrain = (numImagesPerPerson - numTest);
   [performance decisions]=regiondivision( testsets, trainsets, mode,
                          method, databaseName, regionSelection,
                          numPersons, numRuns, numTest,
                          numImagesPerPerson, numBestValues, mainPath );
   averagePerformance = mean( performance );
   fprintf( resultout, 'LaTeX> %3d & %3d & %4.1f & \n', numTrain,
                      numTest, averagePerformance );
```

```
    fprintf( resultout, 'Matlab>performance=[%s]\n',
             sprintf( '%3.2f ', performance ) )
    fprintf( resultout, 'Matlab>decisions=[%s]\n',
             sprintf( '%d ', decisions ) )
end
```

A.5 projectTestImages.m

```
function y = projectTestImages( testsets, curRun, numRuns, numPersons,
                                trainSetSize, numImagesPerPerson,
                                avgImage, eigenFaces, imageXdim,
                                imageYdim, imageDirPath )
   vectorIndex = curRun;
   trainMatrixSize = (numPersons * trainSetSize);
   testSetSize = numImagesPerPerson - trainSetSize;
   for personID=1:numPersons
      % Pick the test set image indices for the current person
      %
      curTestVector = [testsets( vectorIndex, 1:testSetSize )];
      for curTestIndex=1:testSetSize
         % Index of the training image for the current person
         %
         testImageIndex = curTestVector( curTestIndex );
         facefile = [ imageDirPath 'face' int2str( testImageIndex +
                      ((personID - 1) * numImagesPerPerson) ) '.bmp'];
         [AI,Mp2] = imread( facefile );
         AI = ind2gray( AI, Mp2 );
         DI = double( AI ) - avgImage;
         gamma = reshape( DI, (imageXdim * imageYdim), 1 );
         omega = zeros( trainMatrixSize, 1 );
         for k=trainMatrixSize:-1:1
             W = transpose( eigenFaces( :, k ) ) * gamma;
             omega( (trainMatrixSize - k) + 1 ) = W;
         end
         yi = ((personID - 1) * testSetSize) + curTestIndex;
         y( :, yi ) = omega(:);
      end
      vectorIndex = vectorIndex + numRuns;
   end
end
```

A.6 projectTrainImages.m

```
function x = projectTrainingImages( trainsets, curRun, numRuns,
                                    numPersons, trainSetSize,
                                    numImagesPerPerson, avgImage,
                                    eigenFaces, imageXdim, imageYdim,
                                    imageDirPath )
   vectorIndex = curRun;
   trainMatrixSize = (numPersons * trainSetSize);
   for personID=1:numPersons
      % Pick the training set image indices for the current person
      %
      curTrainVector = [trainsets( vectorIndex, 1:trainSetSize )];
      for curTrainIndex=1:trainSetSize
         % Index of the training image for the current person
         %
         trainImageIndex = curTrainVector( curTrainIndex );
         facefile = [ imageDirPath 'face' int2str( trainImageIndex +
                      ((personID - 1) * numImagesPerPerson) ) '.bmp'];
         [AI, Mp2] = imread( facefile );
         AI = ind2gray( AI, Mp2 );          % AI: Arbitrary Image
         DI = double( AI ) - avgImage;      % DI: Difference Image
         gamma = reshape( DI, (imageXdim * imageYdim), 1 );
         omega = zeros( trainMatrixSize, 1 );
         for k=trainMatrixSize:-1:1
             W = transpose( eigenFaces( :, k ) ) * gamma;
             omega( (trainMatrixSize - k) + 1 ) = W;
         end
         xi = ((personID - 1) * trainSetSize) + curTrainIndex;
         x( :, xi ) = omega( : );
      end
      vectorIndex = vectorIndex + numRuns;
   end
end
```

A.7 randomval.m

```
% randval = randomval( x )
%
% Generates a random integer value between 1 and x, inclusive
%
% Examples:
%
% >> randomval( 5 )
% ans = 5
% >> randomval( 5 )
% ans = 4
% >> randomval( 5 )
% ans = 3
% >> randomval( 5 )
% ans = 1
% >> randomval( 5 )
% ans = 5
% >> randomval( 5 )
% ans = 3
%
function randval=randomval( x )
   randval = 1 + rem( floor( rand() * x ), x );
end
```

A.8 **regiondivision.m**

```
% Note that regular full face or individual region tests can be done
% using the weighted majority voting mode with a single region
%
function [performance allDecisions]=regiondivision( testsets,
         trainsets, mode, method, databaseName, regionSelection,
         numPersons, numRuns, testSetSize, numImagesPerPerson,
         numBestValues, mainPath )

if (method == 1)
   fprintf( '--- Region Division with Weighted Majority Voting ...\n' );
elseif (method == 2)
   fprintf( '--- Region Division with Regular Majority Voting ...\n' );
else
   fprintf( '*** Error: Unsupported method %d! Exiting ...\n', method );
   exit( 1 );
end

if (mode == 1)
   fprintf( '--- Region Division with PCA ...\n' );
elseif (mode == 2)
   fprintf( '--- Region Division with LDA ...\n' );
else
   fprintf( '*** Error: Unsupported mode %d! Exiting ...\n', method );
   exit( 1 );
end

performance=[];

isUseForehead=regionSelection( 1 );
isUseEye=regionSelection( 2 );
isUseNose=regionSelection( 3 );
isUseMouth=regionSelection( 4 );
isUseChin=regionSelection( 5 );
isUseWholeFace=regionSelection( 6 );

trainSetSize = (numImagesPerPerson - testSetSize);

allDecisions=[];

for curRun=1:numRuns;
   if (isUseForehead)
      imageDirPath=strcat( mainPath, databaseName, '\',
                           'forehead', '\' );
      if (mode == 1)
         fprintf( '--- Performing PCA (Region Division) on FOREHEAD
                       region ...\n' );
         F=pca_RegionDivisionTest( testsets, trainsets, numPersons,
                                   numRuns, curRun, testSetSize,
                                   numImagesPerPerson, imageDirPath );
      elseif (mode == 2)
         fprintf( '--- Performing LDA (Region Division) on FOREHEAD
                       region ...\n' );
         F=lda_RegionDivisionTest( testsets, trainsets, numPersons,
                                   numRuns, curRun, testSetSize,
                                   numImagesPerPerson, imageDirPath );
      end
   end
   if (isUseEye)
      imageDirPath=strcat( mainPath, databaseName, '\', 'eye', '\' );
      if (mode == 1)
         fprintf( '--- Performing PCA (Region Division) on EYE
                       region ...\n' );
         E=pca_RegionDivisionTest( testsets, trainsets, numPersons,
                                   numRuns, curRun, testSetSize,
                                   numImagesPerPerson, imageDirPath );
      elseif (mode == 2)
         fprintf( '--- Performing LDA (Region Division) on EYE
                       region ...\n' );
         E=lda_RegionDivisionTest( testsets, trainsets, numPersons,
                                   numRuns, curRun, testSetSize,
                                   numImagesPerPerson, imageDirPath );
      end
   end

   if (isUseNose)
      imageDirPath=strcat( mainPath, databaseName, '\', 'nose', '\' );
      if (mode == 1)
         fprintf( '--- Performing PCA (Region Division) on NOSE
                       region ...\n' );
         N=pca_RegionDivisionTest( testsets, trainsets, numPersons,
                                   numRuns, curRun, testSetSize,
                                   numImagesPerPerson, imageDirPath );
      elseif (mode == 2)
         fprintf( '--- Performing LDA (Region Division) on NOSE
                       region ...\n' );
         N=lda_RegionDivisionTest( testsets, trainsets, numPersons,
                                   numRuns, curRun, testSetSize,
                                   numImagesPerPerson, imageDirPath );
      end
   end
```

```
if (isUseMouth)
   imageDirPath=strcat( mainPath, databaseName, ’\’, ’mouth’, ’\’ );
   if (mode == 1)
      fprintf( ’--- Performing PCA (Region Division) on MOUTH
                    region ...\n’ );
      M=pca_RegionDivisionTest( testsets, trainsets, numPersons,
                                numRuns, curRun, testSetSize,
                                numImagesPerPerson, imageDirPath );
   elseif (mode == 2)
      fprintf( ’--- Performing LDA (Region Division) on MOUTH
                    region ...\n’ );
      M=lda_RegionDivisionTest( testsets, trainsets, numPersons,
                                numRuns, curRun, testSetSize,
                                numImagesPerPerson, imageDirPath );
   end
end

if (isUseChin)
   imageDirPath=strcat( mainPath, databaseName, ’\’, ’chin’, ’\’ );
   if (mode == 1)
      fprintf( ’--- Performing PCA (Region Division) on CHIN
                    region ...\n’ );
      C=pca_RegionDivisionTest( testsets, trainsets, numPersons,
                                numRuns, curRun, testSetSize,
                                numImagesPerPerson, imageDirPath );
   elseif (mode == 2)
      fprintf( ’--- Performing LDA (Region Division) on CHIN
                    region ...\n’ );
      C=lda_RegionDivisionTest( testsets, trainsets, numPersons,
                                numRuns, curRun, testSetSize,
                                numImagesPerPerson, imageDirPath );
   end
end

if (isUseWholeFace)
   imageDirPath=strcat( mainPath, databaseName, ’\’, ’full’, ’\’ );
   if (mode == 1)
      fprintf( ’--- Performing PCA (Region Division) on the
                    WHOLE FACE ...\n’ );
      W=pca_RegionDivisionTest( testsets, trainsets, numPersons,
                                numRuns, curRun, testSetSize,
                                numImagesPerPerson, imageDirPath );
   elseif (mode == 2)
      fprintf( ’--- Performing LDA (Region Division) on the
                    WHOLE FACE ...\n’ );
      W=lda_RegionDivisionTest( testsets, trainsets, numPersons,
                                numRuns, curRun, testSetSize,
                                numImagesPerPerson, imageDirPath );
   end
end

numCorrectDecisions=0;
decisionsForCurrentRun=[];

for i=1:(testSetSize*numPersons)
   bestDistanceValues=[];
   bestDistanceIndices=[];
   % Keep track of how many regions have been included in testing
   %
   numRegions=0;

   if (isUseForehead)
      numRegions = numRegions + 1;
      [bestVals bestIdxs]=sort(F(i,:));
      bestDistanceValues(numRegions,:)=bestVals(1:numBestValues);
      bestDistanceIndices(numRegions,:)=bestIdxs(1:numBestValues);
   end

   if (isUseEye)
      numRegions = numRegions + 1;
      [bestVals bestIdxs]=sort(E(i,:));
      bestDistanceIndices(numRegions,:)=bestIdxs(1:numBestValues);
      bestDistanceValues(numRegions,:)=bestVals(1:numBestValues);
   end

   if (isUseNose)
      numRegions = numRegions + 1;
      [bestVals bestIdxs]=sort(N(i,:));
      bestDistanceValues(numRegions,:)=bestVals(1:numBestValues);
      bestDistanceIndices(numRegions,:)=bestIdxs(1:numBestValues);
   end

   if (isUseMouth)
      numRegions = numRegions + 1;
      [bestVals bestIdxs]=sort(M(i,:));
      bestDistanceIndices(numRegions,:)=bestIdxs(1:numBestValues);
      bestDistanceValues(numRegions,:)=bestVals(1:numBestValues);
   end

   if (isUseChin)
      numRegions = numRegions + 1;
      [bestVals bestIdxs]=sort(C(i,:));
      bestDistanceValues(numRegions,:)=bestVals(1:numBestValues);
      bestDistanceIndices(numRegions,:)=bestIdxs(1:numBestValues);
```

```
end

if (isUseWholeFace)
   numRegions = numRegions + 1;
   [bestVals bestIdxs]=sort(W(i,:));
   bestDistanceValues(numRegions,:)=bestVals(1:numBestValues);
   bestDistanceIndices(numRegions,:)=bestIdxs(1:numBestValues);
end

if (method == 1)
   % Do Weighted Majority Voting
   % First, we pick the distance values in the first column,
   % which are the smallest and the corresponding indices from
   % the index matrix
   %
   bestOfBestValues = bestDistanceValues( :, 1 );
   bestOfBestIndices = bestDistanceIndices( :, 1 );
   weightContributions = min( bestOfBestValues ) ./ bestOfBestValues;
   weightIndices = ceil( bestOfBestIndices / trainSetSize );
   % Weights for all persons in the database
   personWeights = zeros( numPersons, 1 );
   for j=1:numRegions
      personID = weightIndices( j );
      personWeights( personID ) = personWeights( personID ) +
                                  weightContributions( j );

   end
   [maxPersonWeight maxPersonID] = max( personWeights );
   if (maxPersonID == ceil( i / testSetSize ))
      numCorrectDecisions = numCorrectDecisions + 1;
      decisionsForCurrentRun = [decisionsForCurrentRun 1];
   else
      decisionsForCurrentRun = [decisionsForCurrentRun 0];
   end
elseif (method == 2)
   % Do Regular Majority Voting
   %
   personIDMatrix = ceil( bestDistanceIndices / trainSetSize );
   % Weight for each person
   personWeights = zeros( numPersons, 1 );
   for j=1:numRegions
      curRegion_personID = personIDMatrix( j, : );
      for k=1:numBestValues
         curID = curRegion_personID( k );
         personWeights( curID ) = 1 + personWeights( curID );
      end
   end
```

```
         [maxPersonWeight maxPersonID] = max( personWeights );
         % Compare the person ID corresponding to the matrix location
         % that has the highest weight contribution with the person ID
         % of the current test subject. Note that we do not have the
         % information about which particular face image we are
         % dealing. We only know the ID of the test subject because of
         % the order of the test case, meaning the first 3 test
         % subjects would belong to Person 1, the next 3 would belong
         % to Person 2 and so on. If the particular face image needs
         % to be known, we will need to modify this program so that
         % that information is carried along with other argument
         % values
         %
         if (maxPersonID == ceil( i / testSetSize ))
            numCorrectDecisions = numCorrectDecisions + 1;
            decisionsForCurrentRun = [decisionsForCurrentRun 1];
         else
            decisionsForCurrentRun = [decisionsForCurrentRun 0];
         end
      end
   end
   fprintf( ’--- Number of correct decisions/#Tests: %d/%d\n’,
            numCorrectDecisions, (testSetSize*numPersons) );
   fprintf( ’--- Training set size=%d, number of persons=%d\n’,
            trainSetSize, numPersons );
   P = ( numCorrectDecisions / (testSetSize*numPersons) ) * 100;
   if (method == 1)
      disp( sprintf( ’>>> Performance of Region-Division with Weighted
                          Majority Voting using PCA = %3.2f’, P ))
   elseif (method == 2)
      disp( sprintf( ’>>> Performance of Region-Division with Majority
                          Voting using PCA = %3.2f’, P ))
   end
   performance = [performance P]
   allDecisions = [allDecisions decisionsForCurrentRun];
end
```

A.9 pcacomp.m

```
% PCA Performance Comparison
%
diary pca-statistics.txt

% Syntax: TESTT(X1,X2,TST,ALPHA,TAIL)
%
fprintf( 1, '-----------------------------------------------\n' );
fprintf( 1, 'PCA Comparison Results (1 training, 9 test)\n' );
fprintf( 1, '-----------------------------------------------\n' );

clear all;

conflevel85=0.15;
conflevel95=0.05;
conflevel99=0.01;

% PCA Experiments (neudb, Weighted Majority Voting, regions=[full ])
% 1 &   9 & 49.4 &
%
fullFaceAverage=[58.52 57.04 40.74 54.07 37.04 57.04 40.74
                 57.04 44.44 47.41 ];
fullFaceDecisions=[1 1 1 0 1 ... 1 0 0 1 1 1 0 1 0 1 1 1 1 0 1 ];

% PCA Experiments (neudb, Regular Majority Voting, regions=[forehead
% eye nose mouth chin ])
%
fiveRegionAverage=[45.93 45.19 43.70 37.78 29.63 42.96 37.04 41.48
39.26 35.56 ];
fiveRegionDecisions=[0 1 0 0 0 ... 0 0 0 0 0 1 0 1 1 0 1 0 0 ];

% PCA Experiments (neudb, Regular Majority Voting, regions=[forehead
% eye nose mouth chin full ])
%
sixRegionAverage=[48.15 44.44 48.89 41.48 41.48 41.48 34.81 43.70
                  45.19 50.37 ];
sixRegionDecisions=[0 0 1 0 ... 0 0 0 0 0 0 ];

fprintf( 1, '\n\n>>> Comparison of (fiveRegionAverage,
                                   fullFaceAverage)\n\n' )

testt( fiveRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel99 )
```

```
fprintf( 1, '\n\n>>> Comparison of (fiveRegionDecisions,
                                    fullFaceDecisions)\n\n' )

testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel99 )

fprintf( 1, '\n\n>>> Comparison of (sixRegionAverage,
                                    fullFaceAverage)\n\n' )

testt( sixRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel99 )

fprintf( 1, '\n\n>>> Comparison of (sixRegionDecisions,
                                    fullFaceDecisions)\n\n' )

testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel99 )

fprintf( 1, '------------------------------------------------\n' );
fprintf( 1, 'PCA Comparison Results (5 training, 5 test)\n' );
fprintf( 1, '------------------------------------------------\n' );

clear all;

conflevel85=0.15;
conflevel95=0.05;
conflevel99=0.01;

% PCA Experiments (neudb, Weighted Majority Voting, regions=[full ])
% 5 &   5 & 65.2 &
%
fullFaceAverage=[69.33 58.67 64.00 65.33 66.67 68.00 61.33 65.33
                 68.00 65.33 ];
fullFaceDecisions=[1 1 0 0 ... 1 0 1 1 1 1 1 0 0 1 ];

% PCA Experiments (neudb, Regular Majority Voting, regions=[forehead
% eye nose mouth chin ])
%
fiveRegionAverage=[65.33 72.00 66.67 73.33 66.67 70.67 65.33 64.00
                   58.67 70.67 ];
fiveRegionDecisions=[1 1 1 0 ... 1 0 1 1 1 1 0 0 1 0 1 ];

% PCA Experiments (neudb, Regular Majority Voting, regions=[forehead
% eye nose mouth chin full ])
```

```
%
sixRegionAverage=[74.67 78.67 70.67 62.67 66.67 73.33 62.67 65.33
                  65.33 74.67 ];
sixRegionDecisions=[1 0 0 0 1 ... 1 0 1 1 1 0 1 1 0 0 1 0 0 ];

fprintf( 1, '\n\n>>> Comparison of (fiveRegionAverage,
                                   fullFaceAverage)\n\n' )

testt( fiveRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel99 )

fprintf( 1, '\n\n>>> Comparison of (fiveRegionDecisions,
                                   fullFaceDecisions)\n\n' )

testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel99 )

fprintf( 1, '\n\n>>> Comparison of (sixRegionAverage,
                                   fullFaceAverage)\n\n' )

testt( sixRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel99 )

fprintf( 1, '\n\n>>> Comparison of (sixRegionDecisions,
                                   fullFaceDecisions)\n\n' )

testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel99 )

fprintf( 1, '-------------------------------------------------\n' );
fprintf( 1, 'PCA Comparison Results (9 training, 0 test)\n' );
fprintf( 1, '-------------------------------------------------\n' );

clear all;

conflevel85=0.15;
conflevel95=0.05;
conflevel99=0.01;

fullFaceAverage=[60.00 86.67 86.67 80.00 53.33 73.33 73.33
                 66.67 53.33 80.00 ];
fullFaceDecisions=[1 1 0 1 ... 1 0 1 0 1 1 1 0 1 1 ];
```

```
fiveRegionAverage=[60.00 93.33 80.00 93.33 66.67 86.67 80.00
                   66.67 73.33 80.00 ];
fiveRegionDecisions=[1 1 0 1 ... 1 0 1 1 1 1 1 1 0 ];

sixRegionAverage=[73.33 93.33 66.67 80.00 80.00 100.00 80.00
                  66.67 86.67 86.67 ];
sixRegionDecisions=[1 0 1 1 ... 1 1 1 1 1 1 ];

fprintf( 1, ’\n\n>>> Comparison of (fiveRegionAverage,
                                   fullFaceAverage)\n\n’ )

testt( fiveRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel99 )

fprintf( 1, ’\n\n>>> Comparison of (fiveRegionDecisions,
                                   fullFaceDecisions)\n\n’ )

testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel99 )

fprintf( 1, ’\n\n>>> Comparison of (sixRegionAverage,
                                   fullFaceAverage)\n\n’ )

testt( sixRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel99 )

fprintf( 1, ’\n\n>>> Comparison of (sixRegionDecisions,
                                   fullFaceDecisions)\n\n’ )

testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel99 )

diary off
```

A.10 ldacomp.m

```
% Syntax: TESTT(X1,X2,TST,ALPHA,TAIL)
%
% Please note that long output lines have been clipped in order to
% make all content fit the available space
%
diary lda-statistics.txt

fprintf( 1, '-------------------------------------------------\n' );
fprintf( 1, 'LDA Comparison Results (1 training, 9 test)\n' );
fprintf( 1, '-------------------------------------------------\n' );

clear all;

conflevel85=0.15;
conflevel95=0.05;
conflevel99=0.01;

% LDA Experiments (neudb, Weighted Majority Voting, regions=[full ])
%
fullFaceAverage=[11.85 14.81 14.07 25.93 42.96 8.89 10.37 37.78
                 7.41 8.89 ];
fullFaceDecisions=[0 0 0 0 0 ... 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 ];

% LDA Experiments (neudb, Regular Majority Voting, regions=[forehead
% eye nose mouth chin ])
%
fiveRegionAverage=[19.26 11.85 15.56 28.15 23.70 32.59 16.30
                   24.44 17.04 20.00 ];
fiveRegionDecisions=[0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 ];

% LDA Experiments (neudb, Regular Majority Voting, regions=[forehead
% eye nose mouth chin full ])
%
sixRegionAverage=[13.33 16.30 20.74 25.93 27.41 36.30 10.37 27.41
                  18.52 22.96 ];
sixRegionDecisions=[0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 ];

fprintf( 1, '\n\n>>> Comparison of (fiveRegionAverage,
                                   fullFaceAverage)\n\n' )

testt( fiveRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel99 )
```

```
fprintf( 1, ’\n\n>>> Comparison of (fiveRegionDecisions,
                                   fullFaceDecisions)\n\n’ )

testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel99 )

fprintf( 1, ’\n\n>>> Comparison of (sixRegionAverage,
                                   fullFaceAverage)\n\n’ )

testt( sixRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel99 )

fprintf( 1, ’\n\n>>> Comparison of (sixRegionDecisions,
                                   fullFaceDecisions)\n\n’ )

testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel99 )

fprintf( 1, ’------------------------------------------------\n’ );
fprintf( 1, ’LDA Comparison Results (5 training, 5 test)\n’ );
fprintf( 1, ’------------------------------------------------\n’ );

clear all;

conflevel85=0.15;
conflevel95=0.05;
conflevel99=0.01;

% LDA Experiments (neudb, Weighted Majority Voting, regions=[full ])
% 5 &   5 & 78.5 &
fullFaceAverage=[81.33 76.00 77.33 76.00 80.00 68.00 84.00 76.00
                 81.33 85.33 ];
fullFaceDecisions=[1 1 0 0 0 ... 1 0 1 1 0 1 0 0 1 ];

% LDA Experiments (neudb, Regular Majority Voting, regions=[forehead
% eye nose mouth chin ]) 5 &   5 & 85.6 &
fiveRegionAverage=[88.00 86.67 78.67 82.67 82.67 93.33 82.67 86.67
                   85.33 89.33 ];
fiveRegionDecisions=[1 0 1 1 1 ... 1 1 1 0 1 1 1 1 1 1 0 1 ];


% LDA Experiments (neudb, Regular Majority Voting, regions=[forehead
% eye nose mouth chin full ]) 5 &   5 & 88.3 &
sixRegionAverage=[92.00 92.00 84.00 86.67 82.67 93.33 86.67 86.67
```

```
                  86.67 92.00 ];
sixRegionDecisions=[1 1 1 1 ... 1 0 1 1 1 1 1 1 1 0 1 ];

fprintf( 1, '\n\n>>> Comparison of (fiveRegionAverage,
                                     fullFaceAverage)\n\n' )

testt( fiveRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel99 )

fprintf( 1, '\n\n>>> Comparison of (fiveRegionDecisions,
                                    fullFaceDecisions)\n\n' )

testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel99 )

fprintf( 1, '\n\n>>> Comparison of (sixRegionAverage,
                                    fullFaceAverage)\n\n' )

testt( sixRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel99 )

fprintf( 1, '\n\n>>> Comparison of (sixRegionDecisions,
                                    fullFaceDecisions)\n\n' )

testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel99 )

fprintf( 1, '------------------------------------------------\n' );
fprintf( 1, 'LDA Comparison Results (9 training, 0 test)\n' );
fprintf( 1, '------------------------------------------------\n' );

clear all;

conflevel85=0.15;
conflevel95=0.05;
conflevel99=0.01;

fullFaceAverage=[73.33 80.00 73.33 86.67 73.33 100.00 80.00
                 66.67 73.33 80.00 ];
fullFaceDecisions=[1 1 1 0 ... 1 1 0 1 1 1 1 0 1 1 ];

fiveRegionAverage=[80.00 100.00 100.00 100.00 86.67 100.00 93.33
                   93.33 93.33 93.33 ];
```

```
fiveRegionDecisions=[1 1 0 1 1 ... 0 1 1 1 1 1 1 1 1 ];

sixRegionAverage=[80.00 100.00 100.00 100.00 100.00 93.33 93.33
                  93.33 100.00 93.33 ];
sixRegionDecisions=[1 1 0 1 ... 1 1 1 1 1 1 1 1 ];

fprintf( 1, '\n\n>>> Comparison of (fiveRegionAverage,
                     fullFaceAverage)\n\n' )

testt( fiveRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( fiveRegionAverage, fullFaceAverage, 0, conflevel99 )

fprintf( 1, '\n\n>>> Comparison of (fiveRegionDecisions,
                                   fullFaceDecisions)\n\n' )

testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( fiveRegionDecisions, fullFaceDecisions, 0, conflevel99 )

fprintf( 1, '\n\n>>> Comparison of (sixRegionAverage,
                                   fullFaceAverage)\n\n' )

testt( sixRegionAverage, fullFaceAverage, 0, conflevel85 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel95 )
testt( sixRegionAverage, fullFaceAverage, 0, conflevel99 )

fprintf( 1, '\n\n>>> Comparison of (sixRegionDecisions,
                                   fullFaceDecisions)\n\n' )

testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel85 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel95 )
testt( sixRegionDecisions, fullFaceDecisions, 0, conflevel99 )

diary off
```

A.11 setuppath.m

```
%
% Set up the path to the Matlab directory that contains all software
%
cd 'C:\Documents and Settings\Administrator\Desktop\software';
addpath 'C:\Documents and Settings\Administrator\Desktop\software';
mainPath='C:\Documents and Settings\Administrator\Desktop\software\'
```

A.12 run_neu_lda_rmv_regiondivision.m

An example script that runs 10 randomly generated experiments on the NFID database using LDA and majority voting using five regions of the face.

```
close all;
clear all;

setuppath

regionSelection=[1 1 1 1 1 0];
numPersons=15
numImagesPerPerson=10
databaseName='neudb';
numRuns=10
numBestValues=4
mode=2 % LDA
method=2 % Regular Majority Voting
fileinfo='np15_nr10_nbv4'

mainmain

% End of script
```

A.13 run_orl_pca_wmv_full.m

An example script that runs 10 randomly generated experiments on the ORL database using PCA using only the whole face.

```
close all;
clear all;

setuppath

regionSelection=[0 0 0 0 0 1];
numPersons=40
numImagesPerPerson=10
databaseName=’orldb’;
numRuns=10
numBestValues=1
mode=1 % PCA
method=1 % weighted Majority Voting
fileinfo=’np40_nr10_nbv1’

mainmain

% End of script
```

■