

Face Recognition using Eigenfaces

A Thesis Presented to
The Faculty of the Mathematics Program
California State University Channel Islands

In (Partial) Fulfilment
of the Requirements for the Degree
Masters of Science

by

Mattie Carazett Jones

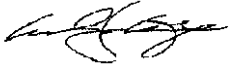
June, 2012

© 2012

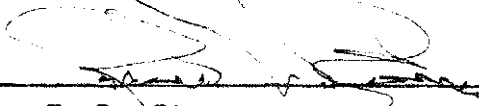
Mattie Carazett Jones

ALL RIGHTS RESERVED

APPROVED FOR THE MATHEMATICS PROGRAM

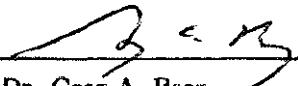


7/13/12
Dr. William Wolfe, Committee Chairperson Date



7/13/12
Dr. Ron Rieger Date

APPROVED FOR THE UNIVERSITY



7-17-12
Dr. Greg A. Berg Date

Non-Exclusive Distribution License

In order for California State University Channel Islands (CSUCI) to reproduce, translate and distribute your submission worldwide through the CSUCI Institutional Repository, your agreement to the following terms is necessary. The author(s) retain any copyright currently on the item as well as the ability to submit the item to publishers or other repositories.

By signing and submitting this license, you (the author(s) or copyright owner) grants to CSUCI the non-exclusive right to reproduce, translate (as defined below), and/or distribute your submission (including the abstract) worldwide in print and electronic format and in any medium, including but not limited to audio or video.

You agree that CSUCI may, without changing the content, translate the submission to any medium or format for the purpose of preservation.

You also agree that CSUCI may keep more than one copy of this submission for purposes of security, back-up and preservation.

You represent that the submission is your original work, and that you have the right to grant the rights contained in this license. You also represent that your submission does not, to the best of your knowledge, infringe upon anyone's copyright. You also represent and warrant that the submission contains no libelous or other unlawful matter and makes no improper invasion of the privacy of any other person.

If the submission contains material for which you do not hold copyright, you represent that you have obtained the unrestricted permission of the copyright owner to grant CSUCI the rights required by this license, and that such third-party owned material is clearly identified and acknowledged within the text or content of the submission. You take full responsibility to obtain permission to use any material that is not your own. This permission must be granted to you before you sign this form.

IF THE SUBMISSION IS BASED UPON WORK THAT HAS BEEN SPONSORED OR SUPPORTED BY AN AGENCY OR ORGANIZATION OTHER THAN CSUCI, YOU REPRESENT THAT YOU HAVE FULFILLED ANY RIGHT OF REVIEW OR OTHER OBLIGATIONS REQUIRED BY SUCH CONTRACT OR AGREEMENT.

The CSUCI Institutional Repository will clearly identify your name(s) as the author(s) or owner(s) of the submission, and will not make any alteration, other than as allowed by this license, to your submission.

Face Recognition using Eigenfaces
Title of Item

Mattie Carazett Jones
Author(s) Name (Print)

Mattie Carazett Jones
Author(s) Signature

6-13-2012
Date

This thesis is dedicated to my sons, Rodrigues Bailey, Antonio and Tevin Jones, and grandfather, Cleofus Bailey (deceased), with loving gratitude and appreciation for their ongoing encouragement and support.

Acknowledgements

I would have not been able to complete this thesis without the advice and encouragement of my long-time friend Frederick Lawrence and my son Rodrigues D. Bailey. After returning to graduate school with apprehension and self-doubt, I embarked on a two-year journey that turned out to be very rewarding and self-gratifying. With sincere appreciation, I would like to thank Dr. Cindy Wyels for her encouragement and undying support. I am also very grateful to Dr. William Wolfe who provided me with indefatigable guidance and Dr. Ronald Gupton who also has continuously provide guidance and support for more than 20 years. I would also like to thank Dr. Brian Sittinger for the many helpful conversations we shared. For support and encouragement throughout my time in graduate school, I would like to offer my special thanks to Antonio Jones, Tevin Jones, and in memory, Cleofus Bailey (deceased). I also would like to thank all my professors, graduate students and close friends, Susan Martinez, Michelle Mueller, Alba Romero.

Abstract

Face Recognition using Eigenfaces

by Mattie Carazett Jones

In this thesis, we researched various methods leading to recognition of a person's face by ways of statistical and mathematical analysis and comparisons of facial features to that of a known person. Many approaches to overcome inherent face recognition challenges have been developed over the years. One of the most accurate and rapid ways to identify faces is to use what is called the eigenface [1] technique, that was created as a linear combination model using the mathematical software called MatLab.

The linear combination model has been recognized for several years and uses the process of Principal Components Analysis (PCA). This system was able to successfully recognize all randomly generated photos (mug-shots) with 97.7 percent accuracy using 21 eigenfaces.

The eigenface technique uses a highly effective combination of linear algebra and statistical analysis (PCA) to generate an identifying set of base faces, the eigenfaces, against which the inputs are tested and matched. Although using a sophisticated statistical model is a means to recognize a person by facial patterns, it is also critically important to acknowledge that the collected data is imperfect and requires some manipulation to be both clean and normalized. The objective is to represent a face as a linear combination of images from our data base. Recently, Random Projection (RP) has emerged as a powerful method for dimensionality reduction. In this paper, I will compare and contrast Random Projection (RP) with PCA using a well known face database. The experimental results illustrate that although PCA represents faces in a low-dimensional subspace, the overall performance is comparable to that of Random Projection, having higher computational requirements and being data dependent.

CONTENTS

1. INTRODUCTION	1
2. PRELIMINARY MATERIAL	3
2.1. Location of Experiment	3
2.2. Background Mathematical skills	4
2.3. Background and Detail Description of Experiment	11
3. METHODOLOGY	14
3.1. Data and Methodology Images	14
3.2. Calculate the Average Face	15
3.3. Generate the Difference Faces	16
3.4. Calculate the Eigenfaces	17
3.5. Project training data into face-space	20
3.6. Evaluate a project test element	24
4. COMPARISON	25
4.1. Eigenfaces to Random projection	25
5. RESULTS	27
5.1. Computer Program Aid	27
6. CONCLUSION and RECOMMENDATION	36
References	37

1. INTRODUCTION

Considerable progress has been made in face recognition research over the last decade, especially with the development of powerful models of face appearance. [2] The most popular appearance-based method is the method of eigenfaces that uses Principal Component Analysis (PCA) to represent faces in a low-dimensional subspace spanned by the eigenvectors of the correlation matrix of the data corresponding to the largest eigenvalues (i.e., directions of maximum variance). Early face recognition algorithms used simple geometric models. Over the past ten to fifteen years, face recognition research and technology has been propelled into the spotlight. This newly discovered interest is primarily the result of major advances within the area of computer vision research and the dramatic success of video surveillance automation. This, coupled with sophisticated printing, dental maps, and eye scans offers a certain intrigue to face recognition. There has also been an increased curiosity in this area primarily due to the huge impact on verification and identification. One of these fascinating methods will be part of an experiment to compare against recently emerging face recognition techniques.

There has been a plethora of papers written on face recognition and various methods associated with this topic. Most of the research favors

a feature-based and connectionist approach to recognition. This approach considers various features of the face and compares them to the same features on other faces. Some of these features include the eyes, ears, nose, and mouth. In addition, most of the recognition approaches use the position size and relationship of these facial features to perform the comparisons. There are three different proposed approaches to face recognition research.

The first method deals with facial characteristics which can be used in recognizing individual faces. The second method incorporates feature vectors extracted from profile silhouettes. The third method uses feature vectors to extract from a frontal view of the face. The most relevant information to best describe a face is derived from the entire face image.[5] This is based on the Karhunen-Loeve expansion (PCA) in pattern recognition. M. Kirby and L. Sirovich have shown that a particular face could be economically represented in terms of a best coordinate system and created the term eigenface.[5] These are the eigenfunctions of the averaged covariance, or normalized correlation, of the ensemble of faces.[5] Later, M. Turk and A. Pentland proposed a face recognition method based on the eigenfaces approach. Measuring and analyzing facial features is used to recognize a person's face by comparing facial structures to that of a known person. Many approaches that overcame face recognition challenges have been devised over the years, however, one of the most accurate and fastest ways to

identify faces is to use what is called the eigenface technique. The eigenface technique uses a highly effective combination of linear algebra and statistical analysis (PCA) to generate an identifying set of base faces, the eigenfaces, against which the inputs are tested, compared and ultimately matched. Although using a sophisticated statistical model to recognize a person by facial patterns is important to identify that the collected data is clean and normalized, the objective is to represent a face as a linear combination of a set of base face images. Matlab was used to create a linear combination model. This paper will discuss the implementation of this algorithm and attempt a critique of whether or not it is a viable solution for a current real-time application. The eigenface face recognition system can be divided into four main segments: (1) Creation of the eigenface basis (Initialization), (2) Projection, (3) Detection and (4) Recognition of a new face. The system follows the following general process sequence (See Top of Page 4):

2. PRELIMINARY MATERIAL

2.1. Location of Experiment. This research experiment took place at California State University at Channel Islands (CSUCI), located in the city of Camarillo, CA, 53 miles North of Los Angeles and 43 miles South of Santa Barbara, California.

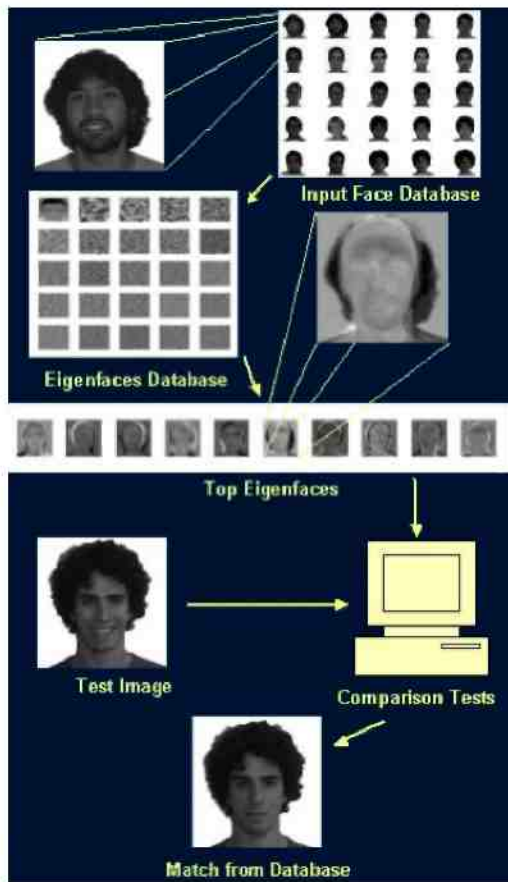


FIGURE 1. A graphical representation of the eigenface face recognition system process sequence.

2.2. **Background Mathematical skills.** First, it is essential to have at least a basic background in the mathematical skills required to understand the Process of Principal Components Analysis which will be addressed later. It is less important to remember the exact mechanics of a mathematical technique than it is to understand the reason why such a technique may be used and what the result of the operation tells us about the data.

- Statistics is centered around the premise that the large set of data, and what is to be analyzed in that set, is presented in terms of the relationships between the individual points in that data set.

- Standard Deviation (most common measure) or Variance is a measure of how spread out the data is.

- Image

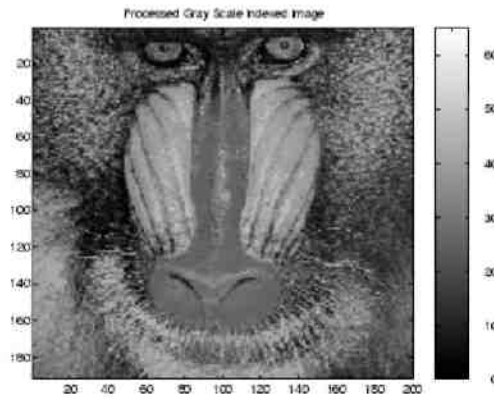


FIGURE 2. The values 0 corresponds to black and 255 to white.

- It is important to know that Variance and Standard Deviation only operate in one dimension. Many data sets have more than one dimension, and the aim of the statistical analysis of these data sets is usually to identify if there is any relationship between the dimensions. Therefore, it is useful to have a measure of findings showing how much the dimensions vary from the mean with respect to each other. Note: the covariance is measured between multiple dimensions. For example, looking at a data set (x, y, z) ,

you could measure the covariance between x and y , x and z , as well as, y and z dimensions. Furthermore, if you measure the covariance between x and x , y and y ; and, z and z dimensions, the results would be the variance of the x , y , and z dimensions. The formula for variance is very similar to the formula for covariance (See below):

$$s^2 = \frac{\sum_{i=1}^n (X_i - \bar{X})^2}{(n-1)}, s^2 \text{ is the usual symbol for variance of a sample}$$

$$var(X) = \frac{\sum_{i=1}^n (X_i - \bar{X})(X_i - \bar{X})}{(n-1)}, \text{expanded the square term to show both parts}$$

$$cov(X, Y) = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}, \text{notice the second set of brackets, the } X\text{'s}$$

are replaced with Y 's (where X and Y are random variables)

Basically, these formulas state that for each data item, multiply the difference between the value x and the mean of x , by the difference between the y value and the mean of y . Then, add all these up and divide by $(n-1)$. Let us next examine two dimensional data from a class of Intermediate Algebra students. Specifically, how many hours in total they spent studying for a chapter test and the grade they received. In our example, the first dimension is H , the hours studied and the second dimension is G or the grade each student received. Now let us calculate the covariance between the number of study hours (H) and the grade each student received (G) as represented in the following table:

$$\text{cov}(H, G) =$$

Hours	Grade	$X - \bar{X}$	$Y - \bar{Y}$	$(X - \bar{X})(Y - \bar{Y})$
5	60	-8	-14	11.2
7	90	1.2	16	19.2
2	40	-3.8	-34	129.20
5	80	-8	6	-4.8
10	100	4.2	26	109.2
29	370			264
5.8	74			66

Table A. Covariance Table Hours vs Grade

Examining the results of the above table, the value is positive as indicated by both dimensions increasing together and shows that the hours increase while the grades increase as well. Conversely, if the value is negative, then one dimension increases while the other decreases. In short, the hours of study increases while the grade decreases. Note: A zero indicated in the two dimension example are independent of each other. This method will be illustrated later because of the information associated with our images is difficult to visualize at this point.

- Covariance Matrix enables us to interpret the covariance of multiple dimensional data, N . The definition of the covariance matrix for a set of data with N dimensions is as follows:

$C^{(n \times n)} = Cov(Dim_i, Dim_j)$, where $C^{(n \times n)}$ is a matrix with n rows and n columns and Dim_x is the x_{th} dimension.

In standard English, a N -dimensional data set forms a square matrix of n rows and columns. Each entry in the matrix is the result of calculating the covariance (scatter) between two separate dimensions.

$$C = \begin{bmatrix} cov(x, x) & cov(x, y) & cov(x, z) \\ cov(y, x) & cov(y, y) & cov(y, z) \\ cov(z, x) & cov(z, y) & cov(z, z) \end{bmatrix}$$

This represents a covariance matrix for an imaginary three-dimensional data set, using the usual dimensions x, y , and z . Notice down the main diagonal, the covariance value is between one of the dimensions and itself. Also, since $cov(x, y) = cov(y, x)$, the matrix is symmetrical about the main diagonal.

The correlation matrix of N random variables, x_1, \dots, x_n is the $n \times n$ matrix with (i, j) -th entry equal to $corr(x_i, x_j)$. If the measures of correlation used are product-moment coefficients, then the correlation matrix is the same as the covariance matrix of the standardized random variables $x_i/\sigma(x_j)$ for $i = 1, \dots, n$. This applies to both the matrix of population correlations (in which case σ is the population standard deviation), and to

the matrix of sample correlations (in which case σ denotes the sample standard deviation). Consequently, each is necessarily a positive semi-definite matrix.

The correlation matrix is symmetric because the correlation between x_i and x_j is the same as the correlation between x_j and x_i . [14]

Definition. Let A be a square matrix. A non-zero vector C is called an **eigenvector** of A if and only if there exists a number (real or complex) λ such that

$$AC = \lambda C$$

If such a number λ exists, it is called an **eigenvalue** of A . The vector C is called eigenvector associated to the eigenvalue λ . [13]

An Eigenvector is a non-null vector whose direction is unchanged by that transformation. [13] Eigenvectors are a special case of matrix multiplication. Observe that by multiplying two matrices can only be done if they are of compatible sizes. Now, let's consider a non-eigenvector:

$$\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} x \begin{bmatrix} 1 \\ 3 \end{bmatrix} = \begin{bmatrix} 11 \\ 5 \end{bmatrix}, \text{ note the resulting vector is not a multiple of the original vector.}$$

In the next example, the resulting vector is exactly 4 times the original vector:

$$\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} x \begin{bmatrix} 6 \\ 4 \end{bmatrix} = \begin{bmatrix} 24 \\ 16 \end{bmatrix} = 4x \begin{bmatrix} 6 \\ 4 \end{bmatrix}$$

In particular, because the vector $\begin{bmatrix} 6 \\ 4 \end{bmatrix}$ is a vector in two-dimensional space and represents an arrow pointing from the origin $(0,0)$ to the point $(6,4)$. The other matrix, 2×2 , is a transformation matrix. We know that by multiplying on the left of a vector, 2×1 , the resulting matrix is a vector that is transformed from its original position. Eigenvectors are derived from this transformation approach. Basically, if there was a vector on the line represented by $y = x$, its reaction would be itself. This vector and all multiples of it (length does not matter) would be an eigenvector of that transformation matrix. Moreover, eigenvectors can only be found for square matrices. However, not every square matrix has eigenvectors. Another property of eigenvectors is that given an $N \times N$ matrix that has eigenvectors, there are N of them. For example, with a 4×4 matrix, there are 4 eigenvectors. Also, an eigenvector property can be seen when you scale a vector by some amount (making it longer) but do not change its direction. Lastly, all the eigenvectors of a symmetric matrix are perpendicular. In other words, it is orthogonal no matter how many dimensions you have. Basically, the data will be expressed in terms of perpendicular eigenvectors (instead of x and

y axes). Because the length of a vector does not affect whether it is an eigenvector or not (whereas the direction does), we are going to scale the eigenvector to make it have a length of 1. Because of this, all eigenvectors will have the same length. For small matrices (no bigger than 3×3), the above process is fairly easy. For large matrices, because of its complexity, this process will be done within a MatLab program.

- An eigenvector's eigenvalue is the scale factor that has been multiplied as in the previous example. In the example, the value was 4 which is the eigenvalue associated with that eigenvector. In simple terms, no matter what multiple of the eigenvector we took before we multiplied it by the square matrix, we would always get 4 times the scaled vector as the result.

2.3. Background and Detail Description of Experiment. This approach uses a general, two-dimensional pattern of the face for recognition. The research of Turk and Pentland [1] suggests that an algorithm treats the face recognition problem as a two-dimensional problem, which assumes that most faces are seen under similar conditions. In this approach, images of known faces are compared to those of unknown faces and whether the face matches one of the known faces, is a new face or not a face at all. To conduct analysis and comparison tests we used the AT&T database which

consists of a total of 400 faces. This particular face database consists of 10 sample images from each of the 40 subjects. The 20 images within the test set will be alternated after 10 recognition trails to establish how robust our program is. Rather than store all the image information of the entire set of known faces, only the eigenfaces will be stored. These are the eigenvectors of the set of faces. The eigenfaces/eigenvectors combined make up a face space and each new face from the test set is projected onto this facespace. The eigenface technique is a powerful solution to the face recognition dilemma. It is the most intuitive way to classify a face. The eigenface technique uses more information by classifying faces based on general facial patterns. These patterns include, but are not limited to, the special features of the face. By using more information, eigenface analysis is naturally more effective than feature-based face recognition. Eigenfaces are fundamentally nothing more than basis vectors for real faces. As will be seen later, it is important to discuss tests which will attempt to see whether this algorithm breaks down. Moreover, face recognition reliability is complicated as a result of varying degrees of obscured known faces, illumination, translation of images and resolution changes. All these factors and conditions will be tested. The below table summaries can be used as a quick reference for the terms and symbols being used in the calculations from this point:

Symbol	Meaning
S	The number of sample images in our face database.
e	The number of eigenfaces to be generated. $e \leq S$. Tweaking this number an generated differences in performance
$I_1 \dots \tilde{I}_s$	The sample images as column vectors. The size of each vector is $(x*y)$ times 1,, x pixels width, y pixels length
A	The average image found from each sample image. (Same size as the sample images).
$D_1 \dots \tilde{D}_s$	The difference vectors between each sample image and the average image. (Column vectors, same size)
V	The matrix of α vectors
$\lambda_1 \dots \lambda_e$	These are the eigenvalues of the $V^T V$ and $V V^T$. The value of e depends on the number of eigenfaces desired....(4)
$X_1 \dots X_e$	These are the eigenvectors that correspond to the eigenvalues. ($e \times 1$)
$m_1 \dots m_e$	The eigenfaces/vectors that are generated from the sample images. The eigenvectors of $V V^T$ matrix. And each corresponds to the eigenvalues $\phi_1 \dots \phi_e$. (Column vectors, same size)
$M = [m_1 \dots m_e]$	The matrix of eigenfaces. When projecting and recovering images to and from the facespace, the calculations are easier.
$\tilde{M} = \{\tilde{m}_1 \dots \tilde{m}_e\}$	The weighted vectors after the projection into facespace. Each \tilde{m}_e is a positive or negative scalar such that each is multiplied by its corresponding eigenface and all are added together which produces the original image.

TABLE 1. All Symbols and its associated meanings [12]

3. METHODOLOGY

3.1. **Data and Methodology Images.** First, we needed to collect a set of face images and therefore obtained 400 face images from the AT& T database. These face images become our database of known faces. We will later determine whether or not an unknown face matches any of these known faces. All face images must be the same size in pixels and, for our purposes, they must be gray-scale, with values ranging from 0 to 255. Each face image is converted into a vector $I_{1,S}$ of length P , $P = \text{imagewidth} * \text{imageheight}$. The most useful face sets have multiple images per person. This sharply increases accuracy, due to the increased information available on each known individual. We will call our collection of faces the training set (faces).

PCA will be used to construct a low-dimensional linear subspace that best explains the variation in the set of face images. All images are gray-scaled faces and if not, MatLab converts them to gray scale for our analysis. They are in PGM file format, which we used MatLab to read the files in and do the conversion.

Image 1 Pixel 1	Image 2 Pixel 1	Image 3 Pixel 1	Image S Pixel 1
Image 1 Pixel 2					Image S Pixel 2
Image 1 Pixel 3					Image S Pixel 3
.....				
Image 1 Pixel P					Image S Pixel P

Table 2. Picture to Matrix conversion [15]

Note: For example, if each image is an 256 x 256 pixels; then P is 65536.

Here we are working with two dimensions because the matrix can become too large and complex very quickly. Each sample image will be referred to as \tilde{I}_n , where n indicates that we are dealing with n^{th} sample image ($1 \leq n \leq S$). Each image in the sample set will be represented as a linear combination of the eigenfaces. The number of possible eigenfaces is equal to the number of face images in the sample set. The method used will make it possible to approximate the faces using only the best eigenfaces or basically, the eigenfaces that account for the most variation.

3.2. Calculate the Average Face. Before finding the eigenfaces, we need to calculate the average face, A , as follows:

$$A = \frac{1}{S} \sum_{t=1}^S \tilde{I}_t$$

Here S is the number of faces in our face database. This is done by merging all the columns into a single column and thus, adding all images columns pixel-by-pixel or row-by-row and then divide by S total images. The below matrix contains the mean pixels:

Mean Image Pixel 1
Mean Image Pixel 2
Mean Image Pixel 3
.....
Mean Image Pixel P

Table 3. The mean pixels per column [15]

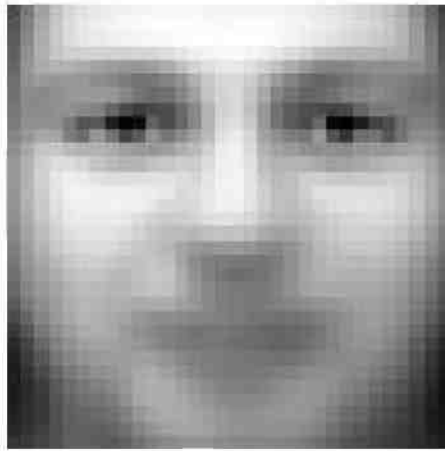


FIGURE 3. An image of an average face from the AT&T face database.

3.3. **Generate the Difference Faces.** The next step is to calculate the difference faces by subtracting the average, in other words, the mean face from each sample image.

$$\tilde{D}_n = \tilde{I}_n - A, 1 \leq n \leq S$$

Where D_n represents the n^{th} with the mean subtracted from it and \tilde{I}_n the eigenvectors. Moreover, this produces a data set whose mean is zero. Later

in the process, we use these differences to compute a covariance matrix V for our data set. Recall from above, the covariance between two sets of data reveals how much the sets correlate. In other words, the intensity relationships between the two sets can be analyzed using the covariance.

3.4. Calculate the Eigenfaces. The eigenfaces that we are looking for are simply the eigenvectors of V . However, since V is of dimension P (the number of pixels in our images), solving for the eigenfaces, gets very complicated very quickly. In fact, eigenface face recognition would not be possible if we had to do this. This is where the fascination behind the eigenface system happens. Based on the statistical technique discussed above, known as Principal Component Analysis (PCA), we can reduce the number of eigenvectors for our covariance matrix from P (the number of pixels in our image) to S (the number of images in our dataset). This is extremely important. We can solve for these eigenvectors by taking the eigenvectors of a new $S \times S$ matrix. Because of the following linear algebra, by using the exact procedure above, we can calculate the covariance between two data sets. Therefore, the covariance matrix C is defined by VV^T , where $V = [\tilde{D}_1, \tilde{D}_2, \tilde{D}_3, \dots, \tilde{D}_S]$, that is the columns of the V matrix are formed by the different faces \tilde{D}_n . and from linear algebra, $C = VV^T$. The dimensions of V are $((x, y) \times S)$, where x and y are the size of the sample

images and S is the number of sample images. Also, the dimensions of V^T are $S \times (x, y)$, and again, where x and y are the size of the sample images and S is the number of sample images. Shown below, the eigenvectors can be found by considering the linear combinations of the eigenvectors of

$$V_{((x,y)xS)} V_{(Sx(x,y))}^T - V_{(Sx(x,y))}^T V_{((x,y)xS)},$$

recalling linear algebra, the right-side dimensions are $(S \times S)$, the dimensions of the new matrix.

To find the eigenvectors of the new matrix, m_e , these vectors are the eigenfaces generated from the sample images, S . We will define them as follows:

$$m_e = \frac{\sum_{t=1}^S D_t \tilde{X}_{te}}{\sqrt{\lambda_e}}$$

Where m_e is an eigenvector of VV^T , the new matrix, $M = |m_1, m_2, \dots, m_e|$. As previously stated, we have summed up all the different faces, multiplying each by the first value of the e_{th} eigenvector of the $V^T V$ (which is X_{th}). The numerator is then divided by the square root of the e_{th} eigenvalue, which corresponds to the $V^T V$ matrix. By using principle component analysis (later, PCA which will be discussed in detail) on the set of large vectors, the result is a set of M orthonormal vectors and their associated eigenvalues λ_n , which best describes the spread of the data in the P -dimensional space. Note: Only $M - e$ eigenfaces are actually needed to produce a complete basis for the facespace (sample), where e is the number of unique individuals in

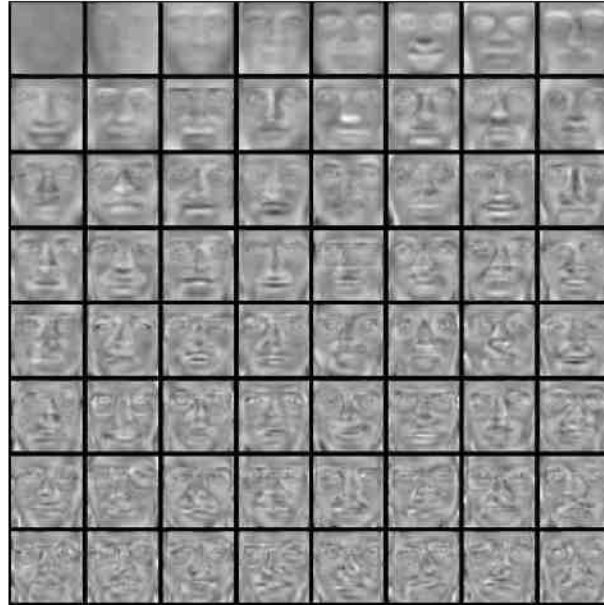


FIGURE 4. The top 64 eigenfaces of the AT&T face database.

the set of known faces. Ultimately, we can get an acceptable reconstruction of the image using only a few eigenfaces (M'), where M' usually ranges anywhere from $.1M$ to $.2M$. These correspond to the vectors with the highest eigenvalues and represent the most variance within face space. Also, these eigenfaces provide a small yet powerful basis for facespace.

Note that the first eigenvector accounts for 50% of the variance in the data set, while the first 20 eigenvectors together account for just over 85%. Also, the first 30 eigenvectors account for 90%. During investigation, increasing the number of eigenvectors generally increases recognition accuracy

but also increases computational costs. However, using too many principal components does not necessarily always lead to higher accuracy since we eventually reach a point of diminishing returns. The ideal number of eigenvectors to retain will depend on the application and the data set but in general, a size that captures around 90% of the variance is usually a reasonable trade-off. Now that the basis vectors for the facespace have been constructed, projecting the faces into facespace is next.

3.5. Project training data into face-space. To project each training image \tilde{I} onto subspace spanned by principal components is represented as follows:

$$\tilde{m}_1, \tilde{m}_2, \tilde{m}_3, \dots, \tilde{m}_e = m_1^T(\tilde{I}_1 - A), \dots, m_e^T(\tilde{I}_S - A)$$

Basically, this means the vector of weights is found by multiplying the transpose of M , where M^T is calculated by letting each eigenface form a column of the matrix. This is accomplished by a vector that is found by subtracting the average face image A , a column vector, from a sample or test image, \tilde{I}_S , which is a column vector as well. Using only a weighted sum of these eigenfaces, it is possible to reconstruct each face in the dataset. In summary, any human face can be considered to be a combination of these standard faces or the standardized face ingredients. As above, one's face may be composed of the average face plus 10% from eigenface, 1, 55%

from eigenface, 2, and even -3% from eigenface 3. Remarkably, it does not take many combined eigenfaces to achieve a fair approximation of most faces. Because the basis faces were found through PCA, it is possible to reconstruct an original face image from the known eigenfaces weights (see figure) on the face space (see below).

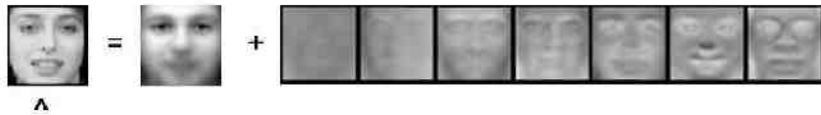


FIGURE 5. The reconstruction of an original face.

Let \tilde{I} be the recovered face:

$$\tilde{I} = \left(\sum_{n=1}^c m_n \tilde{m}_n \right) | A$$

Basically this illustrates adding the weighted eigenfaces m_n together and then reintroducing the average face vector. Note \tilde{I} is still in column vector form after this reconstruction and therefore must be converted to a normal image. Now to identify a new image, the process becomes a pattern recognition task. The new image is not one of the 400 images in the sample space. Here we will measure the differences between the new images and the original images. This will be done by using the new axes derived from the PCA analysis (eigenfaces). Here the PCA analysis gives us the original images in terms of the differences and similarities between them.

Principal Components Analysis (PCA) is a way of identifying patterns in data as well as distinguishing certain characteristics of the data (i.e. similarities and differences). The Principal Component can be thought of as the set of features which together characterize the variations of intensity in respect to sample points of different face images. Each image location contributes more or less to each principal component, so that the latter can be displayed as what may be referred to as a ghostly face called an eigenface. Some of the assumptions and limitations of PCA are as follows:

- Limited to re-expressing the data as a linear combination of its basis vectors.
- Simple and non-parametric, meaning it is very generic and does not depend on the input.
- Principal components are orthogonal.
- Mean and variances are sufficient.
- In general, PCA is used to describe a large dimensional space with a relatively small set of vectors. To restate in another way, the data can be compressed by reducing the number of dimensions without much loss of information. PCA is a popular technique for finding patterns in data of high dimension, as well as expressing the data in such a way as to highlight their similarities and differences. It is

commonly used in face and tumor recognition, image compression, as well as a range of other applications. PCA is applicable to face recognition because face images are usually very similar to each other (relative to images of non-faces) and clearly share the same general pattern and structure. PCA tells us that since we have only M images, we have only M non-trivial eigenvectors.

- Basically, PCA analysis has identified the statistical patterns in the data. The eigenfaces spans a S' dimensional subspace of the original S^2 image space. The S significant eigenvectors of the M matrix are chosen as those with the largest associated eigenvalues.

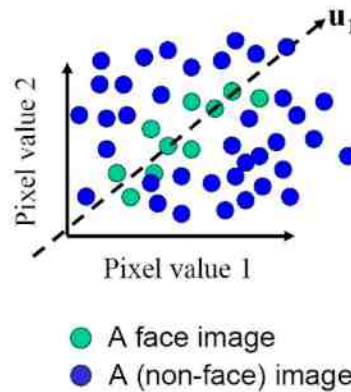


FIGURE 6. Face space containing face images and its feature vector

A face image can be approximately reconstructed on the face space by using its feature vector and the eigenfaces. The face image under consideration

is rebuilt by adding each eigenface with a contribution of M_i to the average of the sample set images. The degree of the fit or the rebuild error ratio can be expressed by means of the Euclidean distance between the original and the reconstructed face image:

$$\frac{\|\psi' - \psi\|}{\psi}, \text{ rebuild error ratio}$$

The rebuild error ratio increases as the training set members differ dramatically from each other. Next, the members differ from each other where the average face image becomes more diluted and thus increases the rebuild error ratio.

3.6. Evaluate a project test element. Lets take a new image from our training set and transform it into eigenface components (projected onto facespaces) by the above process $\tilde{m}_1, \dots, \tilde{m}_e$ for $e = 1, \dots, S'$. The weights form the below vector:

$$\psi^T = [\tilde{m}_1 \tilde{m}_2 \dots \tilde{m}_{S'}]$$

This is the contribution of each eigenface in representing the new face image and treating the eigenfaces as a basis set for the face images. This vector is then used in the standard pattern recognition algorithm to find which of a number of predetermined faces, if any, best describes the face. The algorithm proposed by Turk and Pentland [1] makes a distinction between a face class and a face image. A face class consists of the collection of

face images belonging to an individual. Because a face class contains more than one image ψ_c most reliable recognition can be calculated by averaging the results of the eigenface representation over a small number of face images (as few as one) for each individual. Basically, this is calculated as the average of the weights obtained when projecting each image of a class ψ_c . Classification is performed by comparing the vectors of S , sample set, with ψ^T , the new face image. This comparison is based on the Euclidean distance between the two members/images to be smaller than the user-defined threshold ϵ_e .

$$\frac{\|\psi - \psi_c\|}{\|\psi_c\|} \leq \epsilon_e$$

If the comparison falls within the user defined threshold, then the new face is classified as known. If not within the threshold, it is classified as unknown and can be added to face library with its corresponding vector for later use. This process makes the face recognition process a learning system capable of recognizing new face images.

4. COMPARISON

4.1. Eigenfaces to Random projection. PCA has been very popular with face recognition, especially with the development of the method of eigenfaces. In this section, we evaluate the PCA eigenface for face recognition by conducting a number of experiments using different scenarios and

several data sets. As stated above we considered the AT& T popular face database in this study. In each experiment we divided the database under experimentation into two subsets: training, test, and a test image from the test set. The training set provided a comparison/base set. We performed a number of experiments using two different procedures for evaluating recognition (1) nearest match and (2) most efficient method.

Randomly selecting a test image, the nearest match approach performs recognitions by finding the nearest face from the training set. Recognition accuracy is computed as the ratio of the faces recognized correctly from the training set over the total number of faces in the training set. Also, to account for instabilities in the performance of random projection (RP) due to the random nature of the projection matrix, the recognition performance reported for RP is the average over 200 different RPs (6 trails).[16]

Given a test face, the efficient method retrieves the faster of the two techniques, PCA and RP using the training set. We compute the time it takes for each test face, that is, the ratio of the time it takes for the face images retrieved from the training set that belong to the same person as in the test set, over the total time it takes for the total number of images stored in the training set for that person. Recognition accuracy is computed by averaging the recall rates for the images from the test set. In the case of RP, we report again the average of over 100 different RPs to account

for instabilities in the performance of RP due to the random nature of the projection matrix.

We have evaluated and compared PCA with RP by varying the number of dimension (eigenface) as well as the number of images per subject in the training set. Also, we experimented with varying the identity of the subjects in the training and test sets. It should be noted that this has an affect only on the performance of PCA and hence, RP is independent of the test set. Now, we explain our experiments and results in more detail.

5. RESULTS

5.1. Computer Program Aid. Matlab is used to create and analyze a very simple implementation of eigenfaces face recognition. Recall from above that PCA is the transformation of a number of correlated variables into a smaller number of uncorrelated variables. PCA can best be compared to how Fourier analysis is used to decompose a signal into a set of additive orthogonal sinusoids of varying frequencies. Basically, PCA decomposes an image (signal) into a set of additive orthogonal basis vectors or eigenvectors. One major difference is that Fourier analysis uses a fixed set of basis functions. The PCA basis vectors are learned from the data set via unsupervised training. In this research, PCA is applied to the task of face recognition by converting the pixels of an image into a number of

eigenface feature vectors, which is then compared to measure the similarity of other face images within the database.

Recall that the faces are obtained from the AT&T database. The first step is to load the training images into Matlab. Several requirements are noted as follows:

- Greyscale images with a consistent resolution - All images are black and white pictures. If not, each color image is converted to greyscale.
- Aligned based on facial features - Faces are frontal and well-aligned on facial features such as the eyes, nose and mouth.
- Recall that each image is converted into a column vector and then the images are loaded into a matrix of size $(p \times s)$, where p is the number of pixels in each image and s is the total number of images.

Next, the program completes the below steps required by the training face recognition for calculating the mean of the input face images as follows:

- Subtract the mean from the input images to obtain the mean-shifted images
- Calculate the eigenvectors and eigenvalues of the mean-shifted images (correlation matrix)
- Order the eigenvectors by their corresponding eigenvalues, in decreasing order

- Retain only the eigenvectors with the largest eigenvalues (the principal components)
- Project the mean-shifted images into the eigenspace using the retained eigenvectors

As previously stated, once the face images have been projected into the eigenspace, the similarity between any pair of face images can be calculated by finding the Euclidean distance between their corresponding feature vectors and; such that the smaller the distance between the feature vectors, the more similar the faces. Next, we define a simple similarity score based on the inverse Euclidean distance. Now, in order to perform face recognition, the similarity score is calculated between a test face image and each of the training images. The matched face is the one with the highest similarity, and the magnitude of the similarity score indicates the confidence of the match (with a unit value indicating an exact match). Moreover, to detect cases where no matching face exists in the training set; a minimum threshold for the similarity score is established to ignore any matches below this minimum score.

The most significant computer program challenges faced included:

- Making the program user friendly
- Syntax errors(the right command to use for desired results)

- Compatibility
- Importing images
- Normalizing each image (e.g. size)
- Variety of images

Needless to say, several books and websites were used to build the program from the ground up. The link to the files is located below:

<http://student.csuci.edu/~mattie.jones260/ComputerGraphicspage.html> or

<http://www.adrive.com/public/62f0ed30d2aaf279e13f00e434c263b172431905030cdfc3267cc7945afbb3ee.html>

- Save file to main C drive. If renaming is applicable, please follow these steps:
 - Go to thesis/face recognition folder
 - Open *load_databas_Vfile*
 - Change the letter drive in the code
- Repeat *load_databas_V2Load_databas_T1Load_databas_T1*
 - Rename files
 - Move to folders files

Now, we test PCA and RP using the nearest match approach. Given a test image from the test set; we built a low-dimensional space using the images from the training set. We used the same procedure for RP to project

a test image from the test set in a random, low-dimensional, space using the following algorithm:

- (1) Compute the average face:

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

where, M is the number of face images in the training set.

- (2) Subtract the mean face from each other face:

$$\Phi_i = \Gamma_i - \Psi$$

- (3) Generate the random matrix using the algorithm[7]

- (4) Project the normalized face images in the random subspace:

$$\varpi_i = R\Phi_i$$

Each face in the training set is then represented by the coefficients of projection ϖ_i . [7] Note during recognition, the input face is normalized the same way and projected in the same random space. Hence, the face is recognized by comparing its projections to the projections of the training images, using a similarity-based procedure (standard deviation).

This algorithm is similar to the eigenface approach except that the projection matrix is computed randomly instead of applying PCA on the sample covariance matrix of the data. There were no problems that came up during the study. The only change that should have been done is to add more detail by providing different facial views. Also, it would have been

an improvement to have a larger data set to work with. In conclusion, this study confirms that measuring and analysing facial features is used to recognizing a person's face by comparing facial structure to that of a known person. With clean and normalized data, this type of analysis was accomplished with a linear combination statistical model. Solving problems and interpreting data is getting more sophisticated and user-friendly with time. MatLab is an analytical tool that gives meaning to numbers and creates useful models to be used again and again. For each experiment, we varied the number of dimensions and reported the recognition accuracy using the nearest and most efficient procedure.

We have performed two types of experiments: (a) the test set contains images of the same subjects as in the training set, and (b) the test and training sets contain different subjects. Note that the second scenario is more realistic when we do not have a representative set of images for the training set. Because of the nature of eigenfaces, we would need to use images of other subjects to create a representative test set and compute the eigenspace.

The below charts show the results assuming subjects with the same identity both in the training and test set while figures show the case of subject

having different identity. In both cases, the training set consists of 9 images from each subject (totalling 360) and the test set is comprised of 40 randomly selected images from the training set.

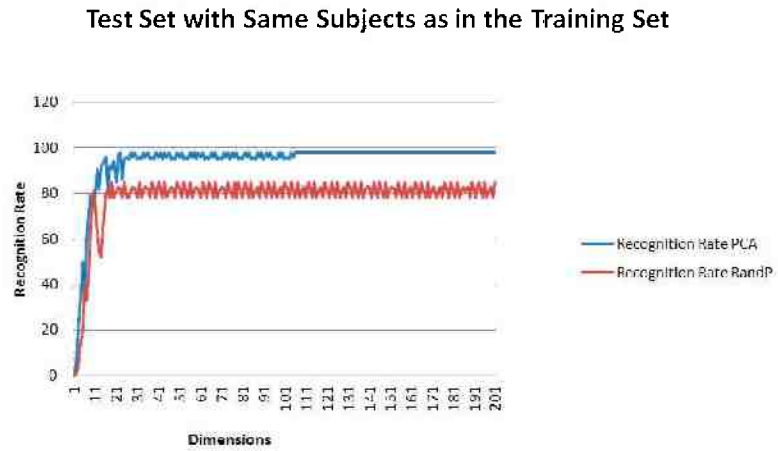


FIGURE 7. Note: The higher dimensions/eigenfaces perform better using the PCA method.

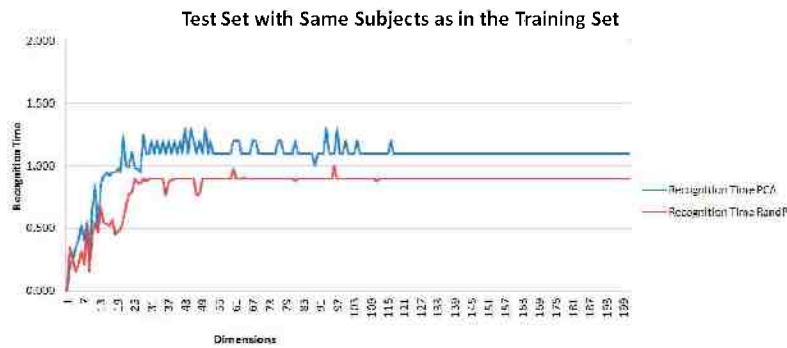


FIGURE 8. Note: The higher dimensions/eigenfaces perform better using the PCA method.

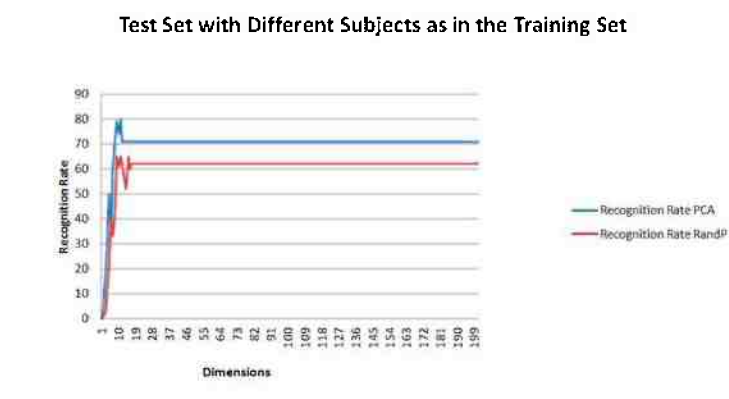


FIGURE 9. Note: The higher dimensions/eigenfaces perform better using the PCA method.

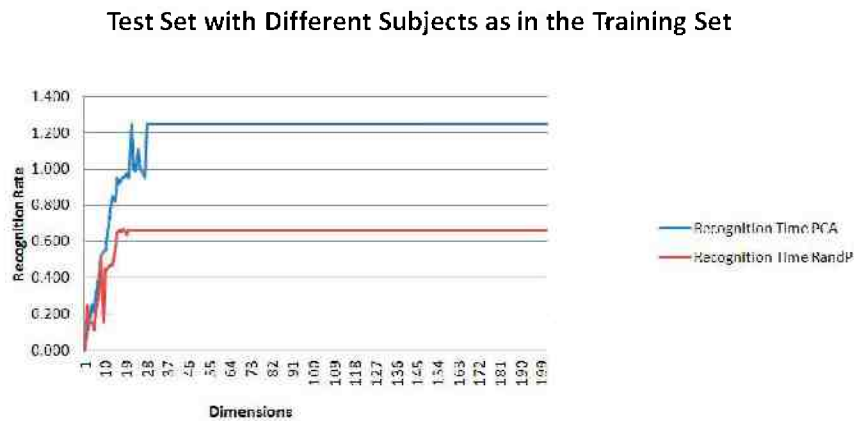


FIGURE 10. Note: The higher dimensions/eigenfaces perform better using the PCA method.

The images in the test set were changed and replaced for each run of the simulation. While in the second case, we built the test set from distinctly different images not in the training set. In part of the experiment, we varied the number of images per subject in the training set to test the sensitivity

of each method. We tested 4 images from each subject and then one image from each subject. The results shown below are consistent with previous studies in several ways:

PCA in general performs better when the identity of the subjects in the training set is the same to the identity of the subjects in the training set.[7]

Comparing PCA with RP, our results show that RP compares favorably with PCA for moderate or higher number of dimensions. The difference in performance becomes smaller and smaller as the number of dimensions increases.

It is obvious that by using the same subjects in the training set as in the test set are more interesting. Because RP is data independent and PCA is data dependent, it seems to do slightly better than PCA for higher dimensions. Overall, both seem to have very close performance and are affected by the number of images per subject in the training set. Basically, as the number of images per subject in the training set decreases; the performance also decreases. Moreover, using different subjects in the training set from the test set, shows results that are quite consistent with known research. RP seems to be doing better than PCA using much higher dimensions.

In this set of experiments, we test PCA and RP using how long it takes each algorithm to render a known match. The accuracy of recognition was considered in the nearest match calculations. When the subjects in the

training and test sets have the same identity were examined, both procedure types of experiments: nearest and speed methods; were consistent with the research, depending on the number of dimensions depicting the efficiency of the method. RP performed consistently better because of the complexity of the algorithm.

6. CONCLUSION AND RECOMMENDATION

We have analyzed the PCA method of face recognition as well as presented an experimental study to evaluate PCA for face recognition with a more state of the art methodology, random projection. Our results indicate that PCA compares favorably to RP, especially when using efficiency as a experiment and nearest match, which uses the ratio discussed earlier.

Our results also show that RP is much faster to computer analysis and the data more independent which might be important factors to consider in a face recognition application. For future research, we plan to investigate more systematically the use of ensembles of RPs for face recognition. Also we plan to investigate much larger ensembles as well as more powerful combination rules such as those presented in various articles and research.[7] Another interest would be to perform comparisons with more powerful dimensionality reduction techniques such as LDA and ICSEA.

REFERENCES

- [1] M.A. TURK AND A.P. PENTLAND, Eigenfaces for Recognition, *Journal of Cognitive Neurosciences*, Volume 3, Number 1, Nov 27, 2002
- [2] DIMITRI PISSARENKO, Eigenface-based facial recognition, Dec 1, 2002
- [3] L.I. SMITH, A Tutorial on principal component analyses, Feb 2002
- [4] M.A. TURK AND A.P. PENTLAND, Face recognition using eigenfaces, *Proc. CVPR*, pp 586-591. IEEE, June 1991
- [5] ILKER ATALAY Face recognition using Eigenfaces, January 1996
- [6] P.N. BELHUMEUR, J.P. HESPAÑA, AND D.J. KRIEGMAN, Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection, *European Conf. Computer Vision* 1996, pp. 45-58
- [7] NAVIN GOEL, GEORGE BEBIS, AND ARA NEFIAN, Face Recognition Experiment with Random Projection, January 1982.
- [8] ATT FACE DATABASE Eigenfaces for Recognition, *Journal of Cognitive Neurosciences*, Volume 3, Number 1, Nov 27, 2002
- [9] EIGENFACES, <http://en.wikipedia.org/wiki/Eigenface>
- [10] ATON, HOWARD, Elementary linear algebra, 6 th ed., Feb 1992
- [11] W. ZHANG, R. CHIELLAPPA, P.J. PHILLIPS, AND A. ROSENFELD, Face recognition: A literature survey, *ACM Computing Surveys*, pp 35(4):399-458. December 2003
- [12] CHRISTOPHER BURNS AND RYAN MORLOK A Comparison of Face Recognition Methods, May 2003
- [13] EIGENVALUES, http://en.wikipedia.org/wiki/eigenvalue_vectors
- [14] CORRELATION, http://en.wikipedia.org/wiki/correlation_matrix
- [15] IMAGES, <http://en.wikipedia.org/wiki/images>

- [16] E. BRIGHAM AND H. MANINILA, Random projection in dimensionality reduction application to image and text data, *International Conference on Knowledge Discovery and Data Mining*, pp. 245-250, 2001.
- [17] D. ACHLIOPTAS Database friendly random projection, *ACM Symposium on the Principles of Database Systems*, pp. 274-281, 2001
- [18] S. DASGUPTA Experiments with random projection, *Uncertainty in Artificial Intelligence*, 2000