3.1 INTRODUCTION

Principal Component Analysis (PCA) is a most practicable statistical technique. Its application plays a major role in many scientific fields related to image compression, image analysis and image recognition. It is the most common and familiar approach to identify certain similar patterns in the given dataset. It also helps in converting high dimensional data into low dimensional for easy storage and efficient computation. This method, PCA, derives an orthogonal projection basis vector. These vectors lead to dimensionality reduction and possibly feature selection. Applying PCA approach to set of faces will generate eigenfaces, which are used in face recognition system.

The main reason for applying eigenfaces methodology is actually the outcome of the dilemma, what are the aspects of the face that are to be considered for recognition—whether the face is addressed as a consistent pattern, or the positions of the characteristics features are adequate. It is not sufficient to depend too much on feature representation to support a robust face recognition system. Because, when the image features are occluded or changed by illumination or
degraded with noise, the image gets distorted, this creates problem during analysis. To overcome such problems, eigenface approach is the simplest, easiest and most efficient approach. Eigenface treats the whole face as a uniform pattern for face recognition system.

The main purpose of using PCA is, to find vectors that best account for variation of faces image in the entire space of images. These vectors are called eigenvectors. It is to construct a face space and project the images into face space. In the given image space, eigenfaces are the eigenvectors of the given covariance matrix. In terms of statistics, PCA approach can be used to simplify a dataset. Formally, it is a transform that chooses a new co-ordinate system for the dataset so that the greatest variance by any projection of the dataset comes to lie on the first axis, the second greatest variation on the second axis, and so on. PCA is used for reducing dimensionality in a dataset. The features or characteristics of the dataset that bestow more to its variance can be retained and the later principal components can be eliminated.

The eigenface approach is the initial technique for successful manifestation of automation of the face recognition system. The principal components analysis can be used to derive the low-dimensional representation of faces in the eigenface approach by applying it to a dataset compressing of images representing faces. As mentioned, the system is implemented by projecting face images on to a feature space that spans the significant variations among known face images. These significant features are called eigenfaces. They are
so called because of the embodiment of principal components from respective sets of face images while training the system. However, ears, eyes, nose and mouth, which are individual features of a face, cannot be represented by this approach. This approach just captures the significant points in an image that shows the variations among the set of images in a database and hence permits them to be identified uniquely.

The eigenface approach is a simple and effective algorithm that can be applied on test images unaffected by luminance changes if and only if all the faces in the database are registered under similar lighting conditions. Formally, eigenfaces are the principal components of the distribution of faces or the eigenvectors of the covariance matrix from the set of face images. In order to represent each face exactly, a linear combination of the eigenfaces can be used and reconstructed. For this representation, the eigenvectors corresponding to the largest eigenvalues are used. To account for different lighting conditions, modular eigenface approach can be used. When compared to the standard eigenface method, this method is less sensitive to changes in appearance. The evaluation of eigenvalues and eigenvectors is a unique matrix operation. Eigenvectors are the coordinates that define the direction of the axes, whose lengths are given by the eigenvalues.

The PCA is a constructive statistical technique with many application areas, such as, face recognition, image compression, security access control, criminal identification, law enforcement and is
a common technique for finding patterns in data of high dimension. In communication theory, it is known as the Karhunen-Loève transform. The main idea is to find a set of M orthogonal vectors in the data space that account for maximum possible variance of data. Projecting the data from their original N-dimensional space onto the M-dimensional subspace spanned by these vectors then performs a dimensionality reduction that often retains most of the inherent information in the data. The first principal component is considered to be along the direction with the maximum variance. The second principal component is constrained to lie in the subspace perpendicular to the first. Within that substance, it points in the direction of the maximal variance. Then, the third principal component is considered in the maximum variance direction in the substance perpendicular to the first two, and so on.

The best tool known for dimensionality-reduction in PCA. This tool helps in reducing a large database to comparative smaller dataset while retaining original data. The inherent complexity of dealing with a large problem in a given time can be minimized using the divided-and-conquer method. Similarly, the dimension of difference information embodied in large covariance matrix is brought down using PCA so as to enhance the subsequent computations in face identification. This technique can be used in eigenface approach to compute the variations in the similarities of the faces in the database and projecting them into a face space. Then, when a test image is the input for recognition, it extracts the prominent features of
the face image and projects them onto the face space to classify and identify the objects. The eigenface approach has been brought out by the researchers almost forty years work ago. The approach was proposed [15] by Turk and Pentland, which was a successful and a real working of automation of human face recognition system. This was a path breaking from the contemporary geometrical representation and recognition methods.

The work embodied in the training is aimed at evolving a model that derives a relationship among the total count of the principal components and the volume of the trained face database. In general, in a PCA algorithm, the principal components are derived from the covariance matrix and these components vary according to the information extracted from the entire image space in its face database. In the present model, the eigenvectors helps in deriving principal components, where the most prominent eigenvalue of every eigenface is treated as components of eigenvector. This process of selecting one principal component from each eigenface is to finally evolve into equality–relation between the total number of the principal components and the number of face images.

A continuous training model is developed with the support of existing system. This new training system called Incremental Principal Component Analysis (IPCA) is designed where a predictor is computed whenever a new face image is appended to the existing face space. The predictor is to be properly and carefully computed with the present existing knowledge under the influence of the new face. The
representative model for the continuous training system using incremental process with one face image at a time is present in the next chapter.

3.2 MODELING PRINCIPAL COMPONENT ANALYSIS (PCA)

A two-dimensional image is defined as a function $f(x,y)$, where $x$ and $y$ are spatial co-ordinates and the amplitude of $f$ at any pair of co-ordinates $(x,y)$ is called the intensity of the image at that point. Face images are represented by intensity values of each pixel. Since every image is in digital format, it can be represented as a matrix of dimension $m \times n$. A digital image is made of grids of pixels with rows and columns. Let the intensity of the pixels at $(x,y)$ location on the grid be represented as $I(x,y)$. Hence the total number of pixels for an image of dimensions $m \times n$ is $mn$. Let this value be represented as $N$.

From here onwards the images are considered as a vector of dimension $N$. For example, if the images have dimension $480 \times 480$ pixels, then the dimension of the image vector will be 230400. So here $N = 230400$. In general, the dimensions of an image is very large, hence image vector dimensionality is also huge. This maps the image to collection of points in a huge space. Since the faces are similar in overall configuration, these images will not be randomly distributed in a huge space and therefore will lie in comparatively low-dimensional space.
Let \( F \) be the set of \( M \) training images of same dimensions \((m \times n) \times M\) expressed as an array.

\[
F = (I_1, I_2, ..., I_M)
\]  

(3.1)

The training set which is a subset of GRIET face database, is presented in Figure 3.1. These \( M \) images are converted in the form of 1-D vectors \( X_i \), \( 1 \leq i \leq M \) of dimension \( N \) \((=m \times n)\), \( X_i \) is an \( N \times 1 \) vector corresponding to the image \( I_i \) in the image space \( F \).

\[
F = (X_1, X_2, ..., X_M)
\]  

(3.2)

![Figure 3.1: Training set of sample face images, a subset of GRIET Face Database.](image)

Face space is a subset of the image space \( F \) and is composed of the vector components as illustrated in Figure 3.2. The pixel represents as a vector component. Thus the whole face is considered a collection of pixels and coded by many vector components, all of which arranged sequentially by concatenating one to the other.
The mean image $\Psi$ is the average information of all images representing the mean value of every pixel in $N$-dimensional vector. Mean image is illustrated in Figure 3.3.

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

(3.3)

Figure 3.2: Basis of Image Space.

Figure 3.3: Mean image of the given $N \times M$ face space.
Now, the mean face is subtracted from each and every image in the dataset to project the variation among the given images. A new image space is created. The mean face subtracted from a particular face is shown in Figure 3.4.

\[ \Phi_i = X_i - \Psi, \quad i = 1, 2, \ldots, M \]  

(3.4)

![Image showing original image, mean image, and difference image.](image)

Figure 3.4: Mean face subtracted from a face image.

The main idea behind the eigenface technique is to explore the similarities among various images. Separation of average information and the deviation from the mean will be the first step in this approach. Eigenfaces are extracted from the difference image information. From this new image space of \( M \), \( \Phi_i \) images (each with dimension \( N \times 1 \)), the matrix \( A \) is formed with dimension \( N \times M \) by taking each of image vectors \( \Phi_i \) and placing them in each column of matrix \( A \).

\[ A = [\Phi_1, \Phi_2, \Phi_3, \ldots, \Phi_M] \]  

(3.5)
Using matrix $A$, a covariance matrix $C$ is constructed. This can be given by product of matrix $A$ with matrix $A^T$. The dimension of such covariance matrix will be $N \times N$.

$$C = \frac{1}{M} \sum_{i=1}^{M} \Phi_i \Phi_i^T = A \, A^T \tag{3.6}$$

The dimensions of the covariance matrix in naturally very huge, and eigenvector is calculated in such high dimension space.

Without loss of generality of the whole training set, it is possible to reduce the dimensionality of the covariance matrix and denote it with 'L', with reduced dimensionality of $M \times M$ given by,

$$L = A^T A \tag{3.7}$$

The eigenvectors of the covariance matrix $C$ are computed by using the matrix $L$. Then the eigenvector $\rho_i$ and the eigenvalue $\lambda_i$ of $L$ are obtained by solving the characteristic equation of eigenvalue problem  

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

where $I$ is identity matrix.

$$L \cdot \rho_i = \lambda_i \cdot \rho_i \tag{3.8}$$

Substituting the value of $L$ in the Eq. (3.8), we obtain

$$A^T A \cdot \rho_i = \lambda_i \cdot \rho_i \tag{3.9}$$

Multiplying both sides with $A$ in the Eq. (3.9), we obtain

$$A \cdot A^T A \cdot \rho_i = A \cdot \lambda_i \cdot \rho_i \tag{3.10}$$

Since $\lambda_i$ is a scalar quantity, Eq. (3.10) can be rewritten as

$$AA^T \cdot A\rho_i = \lambda_i \cdot A\rho_i \tag{3.11}$$

Substituting $C = AA^T$ in the Eq. (3.11), we obtain

$$C \cdot A\rho_i = \lambda_i \cdot A\rho_i \tag{3.12}$$
Let $\mu_i (= A\rho_i)$ and $\lambda_i$ be $M$ eigenvectors and eigenvalues of $C$, respectively. In practice, a subset $M'$ of face space $M$ is sufficient for face reconstruction, because, the subspace of eigenfaces can be treated as the basis for face space, that is, the original face can be represented as a linear combination of these $M'$ vectors. The remaining $(N - M')$ eigenvectors that are associated with eigenvalues play insignificant role in computation. The eigenfaces computed with the above equation are presented in Figure 3.5.

![Eigenfaces](image1.png)

**Figure 3.5:** Eigenfaces generated from the training image set, which is a subset of the given GRIET Face Database.

The eigenfaces are the vectors that describe faces in the face space. These vectors are perceived as points in the $N$-dimensional space. This can be calculated from the actual summing up of pixels of the faces in the database. Each face is of dimension $110 \times 130$ gray scale mode. This means that the faces to be recognized can be
imagined as points in the N-dimensional space. Since all human face images are mostly similar to one another, all the associated vectors are very close to one another and hence face recognition using these eigenfaces doesn't actually give the needed results. The eigenvalues are ordered, where each one account for the variation among the face images.

### 3.2.1 RECONSTRUCTION OF IMAGE FROM PCA MODEL

The reconstructed image is obtained by multiplying the weight matrix \((\Omega)\) of the unknown image with the eigenvector matrix \((\mu)\) of the covariance matrix \((C)\) and adding the mean face image \((\Psi)\) to it. The trained images are projected into the eigenface space and the weight of each eigenvector is evaluated. The weight \((w_k)\) is simply a product of each image with each of the eigenvectors.

\[
 w_k = \mu_k^T \cdot \Phi_i = \mu_k^T \cdot (X_i - \Psi), \quad k = 1, 2, 3, \ldots, M' \quad (3.13)
\]

where \(\mu_k\) is the \(k^{th}\) eigenvector of the covariance matrix, \(\Phi_i\) is the \(i^{th}\) difference image, \(X_i\) is the \(i^{th}\) image and \(\Psi\) is the mean image.

All the weights are converted in the form of a matrix \((\Omega)\) with dimension \(M' \times 1\)

\[
\Omega = [w_1, w_2, w_3, \ldots, w_{M'}]^T \quad (3.14)
\]

The reconstructed image \(\Gamma_f\) is given by
\[ \Gamma_f = \mu . \Omega + \Psi \]

\[ = \left[ \mu_1 \mu_2 \mu_3, ..., \mu_{M'} \right] \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ \vdots \\ w_{M'} \end{bmatrix} + \Psi \]

\[ = \sum_{i=1}^{M'} \mu_i w_i + \Psi \]  \hspace{1cm} (3.15)

where \( \mu = [\mu_1 \ \mu_2 \ \mu_3, ..., \mu_{M'}] \)

By applying the above method the trained sample images as shown in Figure 3.1 are reconstructed from the eigenfaces in Figure 3.5. The reconstructed image is shown in Figure 3.6.

Figure 3.6: Reconstructed images.

As seen from Eq.3.15, the reconstruction of the face image is a simple phenomenon of imparting every eigenface with the mean of the given training set along with the weights. In the reconstruction process all the eigenvalues of the respective eigenvector need not be used. Error estimation is carried out with the number of eigenvalues and the RMS error of the reconstructed image using the equation
\[ \| \Gamma - \Gamma \| \quad (3.16) \]

Observations revealed the fact that the root mean square error increases as the training set samples differ from each other with more variation. This is due to the addition of the mean face image. So, when there is lot of variation among the members of the training set, the mean face image becomes cluttered, which in turn, increases the root mean square error. The test results are presented in Table 3.1 with the root mean square error versus the number of eigenfaces in the training set. It is clear from the observations that the RMS error decreases as the number of eigenfaces increases. This eventually implies that the RMS and the number of eigenfaces in the input set are inversely proportional to each other for the given training set.

**Table 3.1**

**Root Mean Square Error for reconstructed images**

<table>
<thead>
<tr>
<th>Number of Eigenfaces</th>
<th>Root Mean Square Error for reconstructed images</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.3617</td>
</tr>
<tr>
<td>2</td>
<td>0.2076</td>
</tr>
<tr>
<td>3</td>
<td>0.0979</td>
</tr>
<tr>
<td>4</td>
<td>0.0712</td>
</tr>
<tr>
<td>5</td>
<td>0.0506</td>
</tr>
<tr>
<td>6</td>
<td>0.0385</td>
</tr>
<tr>
<td>7</td>
<td>0.0168</td>
</tr>
<tr>
<td>8</td>
<td>0.0064</td>
</tr>
<tr>
<td>9</td>
<td>4.1030e-005</td>
</tr>
<tr>
<td>10</td>
<td>6.0936e-005</td>
</tr>
</tbody>
</table>
Since identification is a pattern recognition task, precise reconstruction of the face image is not necessary. In order to identify, a more minor set of faces with the maximum variation is sufficient for identification. Sirovich and Kirby evaluated [16] a limited version for an ensemble of 115 images. His study revealed that, inorder to describe quite a good set, it is adequate to have about 40 eigenfaces. Turk and Pentland have observed [15] that for a training set of 16 face images, 7 eigenfaces were used to identify a face. Also, Tat Jun Chin and David Suter [96] have come with an inference from their experiment that 8 eigenfaces were enough to account for more than 90% of the variations among a training set of 20 images.

Figure 3.7: Root Mean Square Error for reconstructed images.
The inference drawn from the graph presented in Figure 3.7 reveals that the RMS error for two images is 0.2076, whereas for the other eight images the RMS error is approximately the same. The error graph is plotted by taking an eigenfaces count on the X-axis for 10 eigenfaces and root mean square error on the Y-axis and illustrated in Figure 3.7.

The observations that can be made from the above table and graph is that, the first 2 eigenfaces provide the principal components with prominent information and the next eigenfaces contain insignificant information for the identification process. This is almost 20% of the entire training set, which implies that such a small number of eigenfaces is enough for face recognition. In a database of 200 face images, just 40 images of the training set would suffice for face recognition. From this, the root mean square error can be reduced to be around 2% of the training image set. The representation of graph in Figure 3.7 is a solid example of the heuristic implementation of the above propositions. This implies that for the increasing number of eigenfaces, the root mean square error falls proportionately with a representative example in Figure 3.7. But the maximum variation is observed to be significant in the first two faces when compared with other faces in the reconstructed face space of the present evaluation.
3.2.2 RECOGNITION PROCEDURE FROM PCA MODEL

The weight vector defined in Eq. 3.14 is used for recognition of a face. To recognize a new face image (\( \Gamma \)), its weight (\( w_i \)) is evaluated first by multiplying the eigenvector (\( \mu_i \)) of the covariance matrix (\( C \)) with difference image (\( \Gamma - \Psi \))

\[
w_i = \mu_i^T (\Gamma - \Psi)
\]  

(3.17)

Now the weight matrix (\( \Omega \)) of the unknown image becomes

\[
\Omega = [w_1, w_2, w_3, ..., w_M]^T
\]  

(3.18)

In a face database of \( M \) images, for the recognition program to be evaluated, half of the images are considered as training images and the other half as testing images. Training images are the images that are fed to the system prior to the test procedure. The acquired knowledge in terms of eigenvalues is a prerequisite for any test procedure. The training images are projected into the face space and their weights are calculated. Then, the test image is also projected into the same face space and its weight is also calculated. Then the Euclidean Distance \( \varepsilon_k \) between weight matrices of unknown image (\( \Omega \)) and each face class (\( \Omega_k \)) is defined by

\[
\varepsilon_k^2 = \| \Omega - \Omega_k \|^2 \quad k = 1, 2, ..., N_c
\]  

(3.19)

where \( N_c \) is the number of face classes. In order to distinguish between face images, the Euclidean Distance (\( \varepsilon \)) between the original unknown image (\( \Gamma \)) and the reconstructed image (\( \Gamma_f \)) is computed.

\[
\varepsilon^2 = \| \Gamma - \Gamma_f \|^2
\]  

(3.20)
Calculating the square root of the sum of the squares of the differences between corresponding two data points derives Euclidean Distance between that points. In order to classify an unknown image, the distance between the image data and the class in multi-feature space is used, which is called the minimum distance classifier. One of the minimum distance classifiers is Euclidean Distance, which is appropriate for calculating the distance between one-dimensional vectors. It is theoretically identical to the similarity index. This distance is compared with a threshold value. Threshold value estimates the maximum allowable distance from a face class. It also measures the distance from the face space.

The procedure adopted in the evaluation of weight is presented in the following way. A sample with four training images and three testing images each with resolution of 40 is considered. The following are the eigenfaces of the four training images that are considered for testing the face recognition algorithm. The weights of four training images are presented in Table 3.2.

**Table 3.2**

<table>
<thead>
<tr>
<th>w₁</th>
<th>w₂</th>
<th>w₃</th>
<th>w₄</th>
</tr>
</thead>
<tbody>
<tr>
<td>25.1328</td>
<td>-1.9839</td>
<td>9.1212</td>
<td>-4.2500e-004</td>
</tr>
</tbody>
</table>
Three images are considered for testing $\Gamma_1$, $\Gamma_2$, $\Gamma_3$, each with resolution 40 and their projected weights are as shown in Table 3.3.

<table>
<thead>
<tr>
<th>Weights $\rightarrow$</th>
<th>$w_1$</th>
<th>$w_2$</th>
<th>$w_3$</th>
<th>$w_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Testing Image ↓</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Gamma_1$</td>
<td>-0.3148</td>
<td>-1.9839</td>
<td>-16.6457</td>
<td>-2.7000e-005</td>
</tr>
<tr>
<td>$\Gamma_2$</td>
<td>-8.1322</td>
<td>-12.7221</td>
<td>9.1212</td>
<td>-5.3500e-004</td>
</tr>
<tr>
<td>$\Gamma_3$</td>
<td>-6.6785</td>
<td>-9.7633</td>
<td>18.3256</td>
<td>0.0656</td>
</tr>
</tbody>
</table>

$w_1$, $w_2$, $w_3$ and $w_4$ are the weights of the training images. $w\Gamma_1$, $w\Gamma_2$ and $w\Gamma_3$ are the weights of testing images. The $(i,j)^{th}$ entry in the table is the Euclidean Distance between weight of the $i^{th}$ testing image and the weight of the $j^{th}$ training image. If $(i,j)^{th}$ entry is 0, it represents $i^{th}$ test image is recognized as $j^{th}$ image. If all the entries corresponding to the test image are non-zero, then the image is an unknown image. Testing image $\Gamma_1$ is recognized as image 2, and the testing image $\Gamma_2$ is recognized as image 3, which the testing image $\Gamma_3$ is unknown image, which is nearer to image with a representative example as presented in Table 3.4.
Table 3.4

Euclidean Distances between weights of training and testing images

<table>
<thead>
<tr>
<th></th>
<th>$w_1$</th>
<th>$w_2$</th>
<th>$w_3$</th>
<th>$w_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_{\Gamma_1}$</td>
<td>22.4336</td>
<td>0</td>
<td>23.8191</td>
<td>0.0005</td>
</tr>
<tr>
<td>$w_{\Gamma_2}$</td>
<td>29.3284</td>
<td>13.8660</td>
<td>0</td>
<td>3.0184e-016</td>
</tr>
<tr>
<td>$w_{\Gamma_3}$</td>
<td>28.1258</td>
<td>7.7132</td>
<td>12.2697</td>
<td>0.0569</td>
</tr>
</tbody>
</table>

The above recognition process is further elaborated by taking the training images as both testing and training images, as depicted in Table 3.5. $w_1$, $w_2$, $w_3$ and $w_4$ are the weights of the training and testing images. $(i,j)^{th}$ entry in the table is the Euclidean Distance between weight of the $i^{th}$ testing image and the weight of the $j^{th}$ training image. If $(i,j)^{th}$ entry is 0, it represents $i^{th}$ testing image is recognized as $j^{th}$ image. In Table 3.5 all the diagonal elements are zeros, which means that the $i^{th}$ image is same as the $j^{th}$ image i.e., recognizing itself.
Table 3.5

Differences between weights of training and testing images

<table>
<thead>
<tr>
<th></th>
<th>$w_1$</th>
<th>$w_2$</th>
<th>$w_3$</th>
<th>$w_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1$</td>
<td>0</td>
<td>24.3663</td>
<td>15.2546</td>
<td>22.1255</td>
</tr>
<tr>
<td>$w_2$</td>
<td>24.3663</td>
<td>0</td>
<td>8.9123</td>
<td>1.6043</td>
</tr>
<tr>
<td>$w_3$</td>
<td>15.2546</td>
<td>8.9123</td>
<td>0</td>
<td>6.1212</td>
</tr>
<tr>
<td>$w_4$</td>
<td>22.1255</td>
<td>1.6043</td>
<td>6.1212</td>
<td>0</td>
</tr>
</tbody>
</table>

Weight vector is a representation of face class. Application of Euclidean Distance measures is to find a match in-between weight of the probe image and the registered weights of the trained images. This is very essential phase of face recognition. Recognition rate is obtained with a simple statistical relation of number of test images with an exact match versus the total number of test images. This relationship is used to evaluate the performance of the algorithm.
3.3 3D FACE GENERATION

In general, all the required and significant poses of a particular individual may not be possible to capture in real-world scenario. Hence, limited number of eigenfaces can be generated for a face space. A framework is proposed where a 2D face image is considered and projected onto 3D space. The framework consist of two parts, the first part deals with 2D-to-3D integrated face reconstruction and the next emphasis on face recognition using the virtual faces with different pose, illumination and expressions (PIE).

In order to reconstruct the face, the only required input to the system is a frontal face image of a subject with normal illumination and neutral expression. Based on 2D alignment, key points are captured and the image is rotated, either clockwise or anti-clockwise through different angles. These key points or feature points are accurate enough to generate an integrated reconstructed faces. Thus virtual faces are generated as per the requirements. Analysis-by-synthesis approach is employed to obtain multiple views of an individual face image. The generated 2D faces from an individual face image when projected in 3D space are projected in chapter 5 along with the performance measured.