PES University, Bangalore

(Established under Karnataka Act No. 16 of 2013)

MAY 2020: IN SEMESTER ASSESSMENT (ISA) B.TECH. IV SEMESTER

UE18MA251- LINEAR ALGEBRA

MINI PROJECT REPORT

ON

FACE RECOGNITION USING EIGEN FACES

Submitted by

1. Name: G Navyadhara Gana Sai SRN: PES1201800230
2. Name: Siddharth k Rao SRN: PES1201802127
3. Name: Navneeth S Holla SRN: PES1201800262

Branch & Section: CSE_4A

PROJECT EVALUATION

(For Official Use Only)

Sl.No.	Parameter	Max Marks	Marks Awarded
1	Background & Framing of the problem	4	
2	Approach and Solution	4	
3	References	4	
4	Clarity of the concepts & Creativity	4	
5	Choice of examples and understanding of the topic	4	
6	Presentation of the work	5	
	Total	25	

Name of the Course Instructor :

Signature of the Course Instructor :

ABSTRACT

Human body use their brain and eyes to see and sense the world, computer vision aim is the same. It is mainly concerned with analysis and understanding of information from an image or multiple images which involve development of algorithmic and theoretical basis. Linear algebra plays an important role to achieve the aim of computer vision. In this paper, the main focus is on how linear algebra helpful in computer vision. Through the use of linear algebra and these other mathematical models, the field of computer vision has expanded rapidly. Currently computer vision is being used in solving vital problems in a vast array of fields including medical imaging, surveillance, and face and object detection and identification.

I. INTRODUCTION

The aim of this report is to highlight the application of linear algebra in field of computer vision and face recognition, which we have taken as a project. This report shall present the idea of "*Face recognition using eigen faces"* and give basic idea of applications of linear algebra in field of computer vision.

The scope of this work is, to work on a technique to detect/recognize faces by using the method of eigen faces. This is a very well formulated application of linear algebra in the field of computer vision.

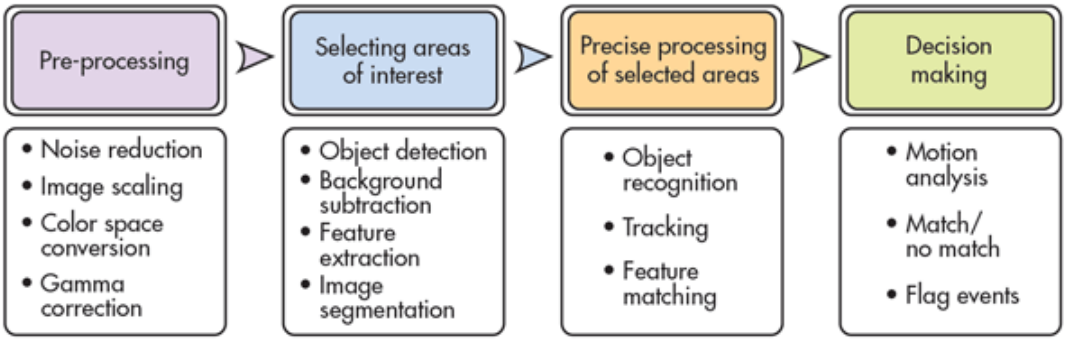
II. REVIEW OF LITERATURE

Computer vision is a relatively new and vastly growing field born out of the study of artificial intelligence. While this ability comes quite easily to humans and animals alike, it is much more difficult to reproduce in computers.

A face recognition algorithm processes the captured image and compares it to the images stored in the database. If a match is found, then the individual is identified. If no match is found, then the individual is reported as unidentified.

It is a field of study in which basically we learn how to reconstruct, understand and interpret the 3D image into the 2D image in terms of structure's properties in the scene. Its main goal is to create a model, replica and most important exceed the human vision with the help of computer hardware and software by using mathematics, science etc.

The hierarchy of computer vision would be a four-staged process as shown below



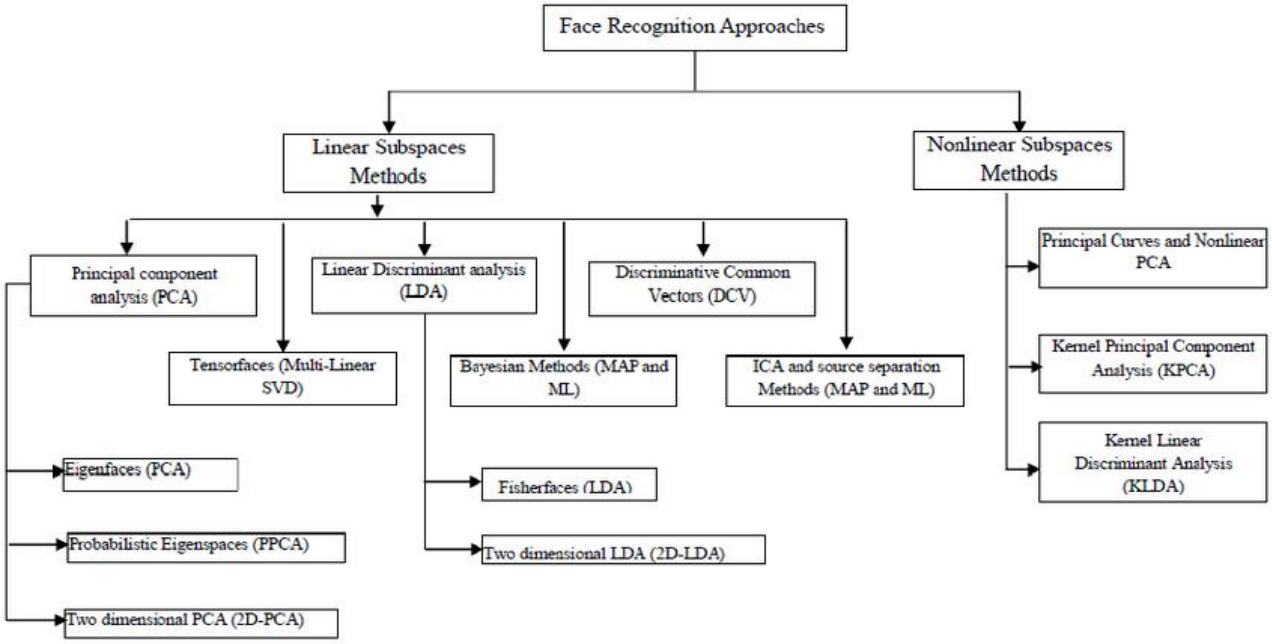
The most common approaches to solving problems in computer vision are statistical and linear models. In statistical models, probability is used to determine what is most likely to be occurring in an image based on the known parameters of the problem and the proximity of parameters to those of predetermined predictable outcomes. Whereas in linear models we take the help of linear algebra in which we take multiple images at different but close to each other from different angles.

Much of the work in computer recognition if faces has focused on detecting individual features such as the eyes, nose and head outline and defining a face model by the position, size, and relationships among these features. Such approaches have been proven difficult to a multiple view and often have been quite fragile and always required a good initial guess to guide them.

Research in human strategies of human face recognition has shown that individual features and their immediate relationships comprise an insufficient representation to account for human face recognition. Nonetheless this method remains the most popular one in computer vision.

Approaches have been made in many different ways to perform the task of face recognition like the one where an automated face recognition has been implemented by characterising face as a set of geometric parameters and performing pattern recognition on the parameters (Kaya & Kobayashi, 1972). And another notable work uses a “smart sensing” approach based on multiresolution template matching (Burt, 1988). This system has been able to recognise people in real-time but has its own flaws like, sensitivity to image size and noise.

There are many ways and approaches to perform the task of face recognition and each one has its own cons and pros. It is our part to choose the best available method.



III. FACE RECOGNITION USING EIGEN VALUES

Much of the previous work on face recognition has ignored the issue of what aspects of the face stimulus are important for identification. This suggested to approach with a coding and decoding of face images, which may give insight into information content of face images, emphasising the local and global features of face.

In general words, we want to extract the relevant information from an image, encode it in the most efficient way possible, and compare it with a database of models encoded similarly. In mathematical way, we wish to find the PCA of the distribution of faces, or the eigen vectors of the covariance matrix, treating each image as a point in space.

The basis of the eigenfaces method is the Principal Component Analysis (PCA). Eigenfaces and PCA have been used by Sirovich and Kirby to represent the face images efficiently. They have started with a group of original face images, and calculated the best vector system for image compression. Then Turk and Pentland applied the Eigenfaces to face recognition problem.

The Principal Component Analysis is a method of projection to a subspace and is widely used in pattern recognition. An objective of PCA is the replacement of correlated vectors of large dimensions with the uncorrelated vectors of smaller dimensions. Another objective is to calculate a basis for the data set. Main advantages of the PCA are its low sensitivity to noise, the reduction of the requirements of the memory and the capacity, and the increase in the efficiency due to the operation in a space of smaller dimensions.

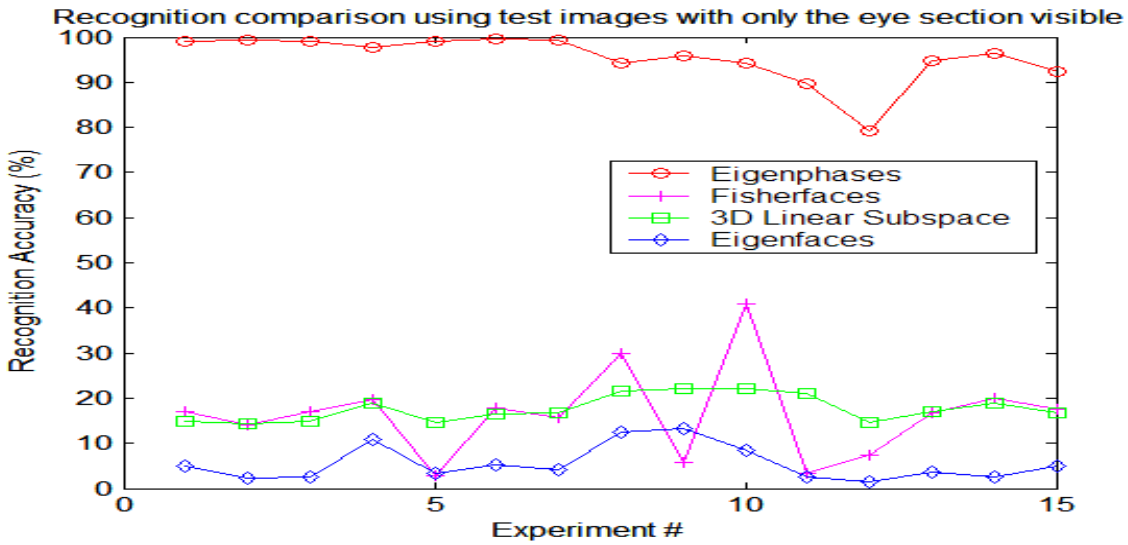
The strategy of the Eigenfaces method consists of extracting the characteristic features on the face and representing the face in question as a linear combination of the so called '*eigenfaces'* obtained from the feature extraction process. The principal components of the faces in the training set are calculated. Recognition is achieved using the projection of the face into the space formed by the eigenfaces.

A comparison on the basis of the Euclidian distance of the eigenvectors of the eigenfaces and the eigenface of the image under question is made. If this distance is small enough, the person is identified. On the other hand, if the distance is too large, the image is regarded as one that belongs to an individual for which the system has to be trained.

This method is found to be highly accurate when compared to other approaches of face recognition which can be seen statistically in the below images.

Comparison of some work related to face recognition

Method	Number of images in the training set	Success rate
Principal Component Analysis	400	79.65%
Principal Component Analysis + Relevant Component Analysis	400	92.34%
Independent Component Analysis	170	tanh function 69.40%
	40	Gauss function 81.35%
Hidden Markov Model	200	84%
Active Shape Model	100	78.12-92.05%
Wavelet Transform	100	80-91%
Support Vector Machines	-	85-92.1%
Neural Networks	-	93.7%
Eigenfaces Method	70	92-100%



APPROACH FOR ALGORITHM

i. Computing Eigen Faces

• Main idea behind eigenfaces

- Suppose Γ is an N^2x1 vector, corresponding to an NxN face image I.

- The idea is to represent Γ (Φ=Γ - mean face) into a low-dimensional space:

$$\hat{\Phi} - mean = w_1 u_1 + w_2 u_2 + \cdots w_K u_K \ (K << N^2)$$

- **Step_1:** Obtain face images I_1, I_2,…….I_M (Trainng images).
- **Step-2:** Represent every image I_i as Γ_i .
- **Step-3:** Compute average face vector Ψ.

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

- **Step-4:** Subtract the mean face.

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

- **Step-5:** Compute the covariance matrix C.

$$C = \frac{1}{M} \sum_{n=1}^{M} \Phi_n \Phi_n^T = AA^T \quad (N^2 \text{x} N^2 \text{ matrix})$$

$$\text{where } A = [\Phi_1 \ \Phi_2 \cdots \Phi_M] \quad (N^2 \text{x} M \text{ matrix})$$

- **Step-6:** Compute the eigen vectors u_i of AA^T

The matrix AA^T is very large --> not practical !!

Step 6.1: consider the matrix $A^T A$ (MxM matrix)

Step 6.2: compute the eigenvectors v_i of $A^T A$

$$A^T A v_i = \mu_i v_i$$

What is the relationship between us_i and v_i?

$A^T A v_i = \mu_i v_i$ => $AA^T A v_i = \mu_i A v_i$ =>

$CAv_i = \mu_i A v_i$ or $Cu_i = \mu_i u_i$ where $u_i = Av_i$

Thus, AA^T and $A^T A$ have the same eigenvalues and their eigenvectors are related as follows: $u_i = Av_i$!!

Note 1: AA^T can have up to N^2 eigenvalues and eigenvectors.

Note 2: $A^T A$ can have up to M eigenvalues and eigenvectors.

Note 3: The M eigenvalues of $A^T A$ (along with their corresponding eigenvectors) correspond to the M *largest* eigenvalues of AA^T (along with their corresponding eigenvectors).

Step 6.3: compute the M best eigenvectors of AA^T: $u_i = Av_i$

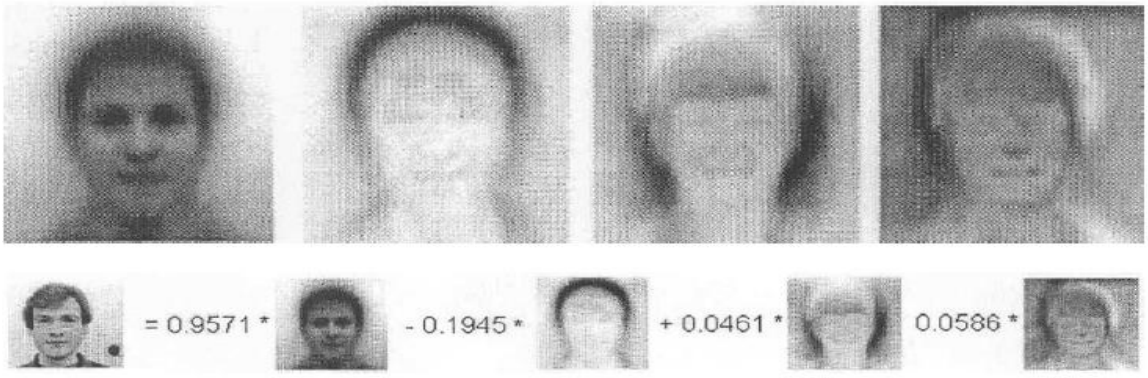
(**important:** normalize u_i such that $\|u_i\| = 1$)

ii. Representing faces onto this basis

- Each face ϕ_i (minus the mean) can be represented as a linear combination of the best K eigen vectors

$$\hat{\Phi}_i - mean = \sum_{j=1}^{K} w_j u_j, \quad (w_j = u_j^T \Phi_i)$$

(we call the u_j's *eigenfaces*)



- Each normalised face ϕ_i is represented in the vector:

$$\Omega_i = \begin{bmatrix} w_1^i \\ w_2^i \\ \cdots \\ w_K^i \end{bmatrix}, \quad i = 1, 2, \ldots, M$$

iii. Recognition of faces

- Given an unknown face image Γ (centered and of the same size like the training faces) follow these steps:

Step 1: normalize Γ: $\Phi = \Gamma - \Psi$

Step 2: project on the eigenspace

$$\hat{\Phi} = \sum_{i=1}^{K} w_i u_i \quad (w_i = u_i^T \Phi)$$

Step 3: represent Φ as: $\Omega = \begin{bmatrix} w_1 \\ w_2 \\ \dots \\ w_K \end{bmatrix}$

Step 4: find $e_r = \min_l \|\Omega - \Omega^l\|$

Step 5: if $e_r < T_r$, then Γ is recognized as face l from the training set.

- The distance e_r is called distance within the face space (difs)

Comment: we can use the common Euclidean distance to compute e_r, however, it has been reported that the *Mahalanobis distance* performs better:

$$\|\Omega - \Omega^k\| = \sum_{i=1}^{K} \frac{1}{\lambda_i} (w_i - w_i^k)^2$$

(variations along all axes are treated as equally significant)

ALGORITHM

- **Training Algorithm**

1. First, we read all the training images.
2. Convert each **[m x n] image matrix** into **[(m x n) x 1] image vector. (**Flatten image matrix to vector).
3. Find average_face_vector, sum (all image vectors)/number(images).
4. Subtract average_face_vector from every image vector.
5. Stack all (image_vectors — average_face_vector) in matrix forming **A = [(m x n) x i]** matrix (where i = number of images).
6. Calculate covariance matrix of above matrix -> **C = A*transpose(A)**.
7. Find **eigenvectors** and **eigenvalues** of above covariance matrix.

8. As size of above matrix is huge its very expensive to compute eigenvectors of such a huge matrix, so we will calculate the **eigenvectors of transpose(A)*A** as the matrix is small of **i x i** dimension.
9. Then we will pre-multiply the eigenvectors of transpose(A)*A with A to get eigenvectors of A*transpose(A). → **(Proof is given in paper)**
10. Choose best k eigenvectors such that **k << i.**
11. Find weights of each image and store it.

- **Testing Algorithm**

1. Normalize the test image -> **I = Test_Image_Vector — Average_Face_Vector.**
2. Find weights of test image using above training process (use test images which have not been used in training process to actually see how accurate your algorithm is recognizing).
3. Calculate error between weights of test image and weights of all images in training set.
4. Choose the image of training set which has minimum error.

There are many ways to calculate error such Euclidean distance, mahala-Nobis distance etc. You can use different distances and see the results and conclude which performs better.

The below image provides a flowchart of the algorithm,

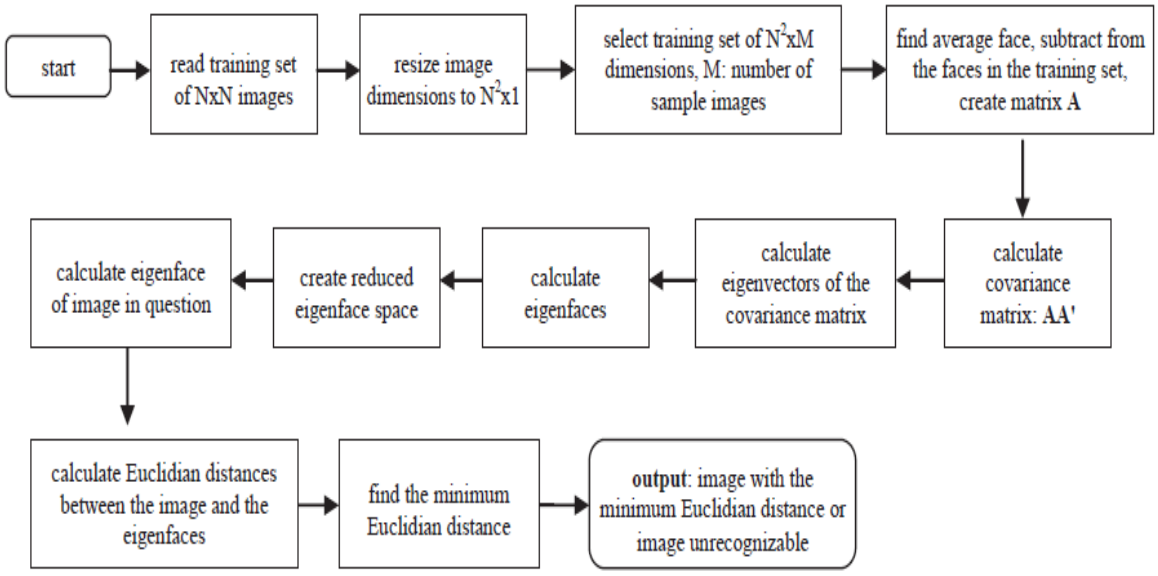


Fig. 1. Flowchart of the algorithm of the Eigenfaces method

IMPLEMENTATION OF ALGORITHM – IN *PYTHON*

Face Recognition Using Eigenfaces

```
import os
import numpy as np
from matplotlib import pyplot as plt
from PIL import Image as im
```

Load Data

```
faces_matrix = np.vstack(neutral)
faces_matrix.shape
```

```
mean_face = np.mean(faces_matrix, axis=0)
mean_face.shape
```

```
plt.imshow(mean_face.reshape(49,58),cmap='gray');
plt.title('Mean Face')

### normalization
faces_norm = faces_matrix - mean_face
faces_norm.shape
```

Compute Covariance

```
# Calculate covariance matrix
face_cov = np.cov(faces_norm.T) #np.cov expects features as rows and observations as columns, so transposed
face_cov.shape
```

SVD

to get eigen-vectors

```
eigen_vecs, eigen_vals, _ = np.linalg.svd(face_cov)
eigen_vecs.shape
```

VISUALISE FIRST 10 PC'S/Eigenfaces ¶

```
fig, axs = plt.subplots(1,3,figsize=(15,5))
for i in np.arange(10):
    ax = plt.subplot(2,5,i+1)
    img = eigen_vecs[:,i].reshape(49,58)
    plt.imshow(img, cmap='gray')
fig.suptitle("First 10 Eigenfaces", fontsize=16)
```

Reconstruction with EigenFaces

```
fig, axs = plt.subplots(2,5,figsize=(15,6))
for k, i in zip([0,1,9,19,39,79,159,199,399,799],np.arange(10)):
    # Reconstruct the first picture '1a.jpg' whose index is 0.
    weight = faces_norm[0,:].dot(eigen_vecs[:,:k]) # Get PC scores of the images
    projected_face = weight.dot(eigen_vecs[:,:k].T) # Reconstruct first face in dataset using k PCs
    ax = plt.subplot(2,5,i+1)
    ax.set_title("k = "+str(k+1))
    plt.imshow(projected_face.reshape(49,58)+mean_face.reshape(49,58),cmap='gray');
fig.suptitle(("Reconstruction with Increasing Eigenfaces"), fontsize=16);
```

IV. RESULTS AND DISCUSSIONS

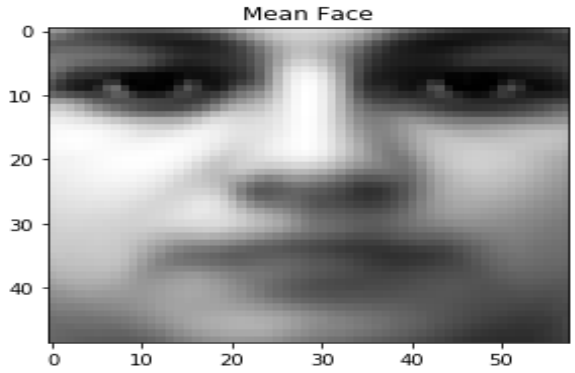
With the above implementation of the face recognition, the results we would get are

- The output for the mean/average face

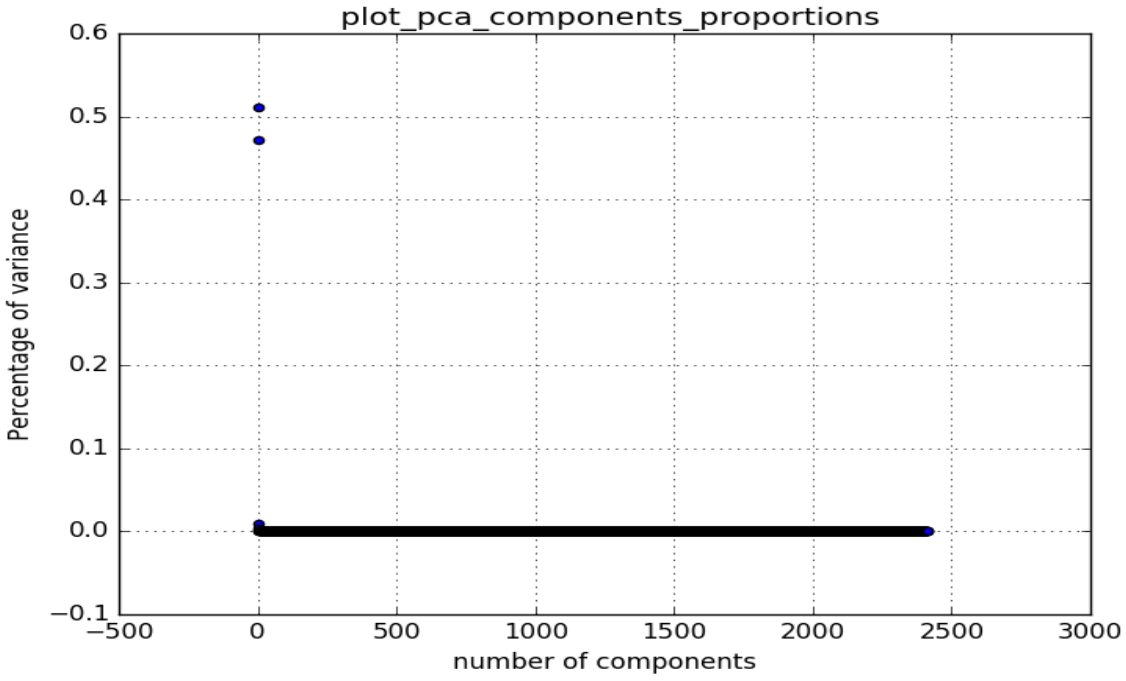
```
In [7]: plt.imshow(mean_face.reshape(49,58),cmap='gray');
        plt.title('Mean Face')

        ### normalization
        faces_norm = faces_matrix - mean_face
        faces_norm.shape

Out[7]: (200, 2842)
```



- The plot for the variance for no: components

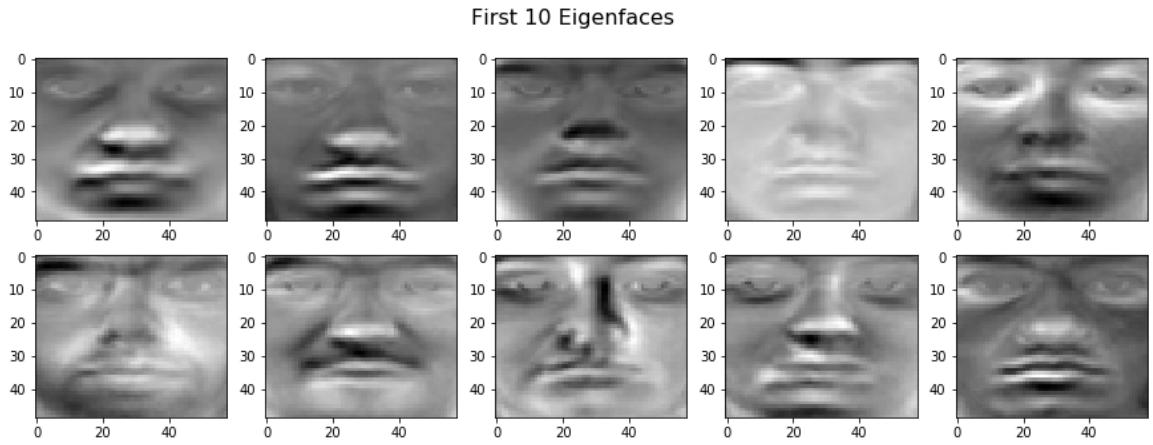


- The first 10 EigenFaces

Visualize first 10 PCs / eigenfaces

```
fig, axs = plt.subplots(1,3,figsize=(15,5))
for i in np.arange(10):
    ax = plt.subplot(2,5,i+1)
    img = eigen_vecs[:,i].reshape(49,58)
    plt.imshow(img, cmap='gray')
fig.suptitle("First 10 Eigenfaces", fontsize=16)
```

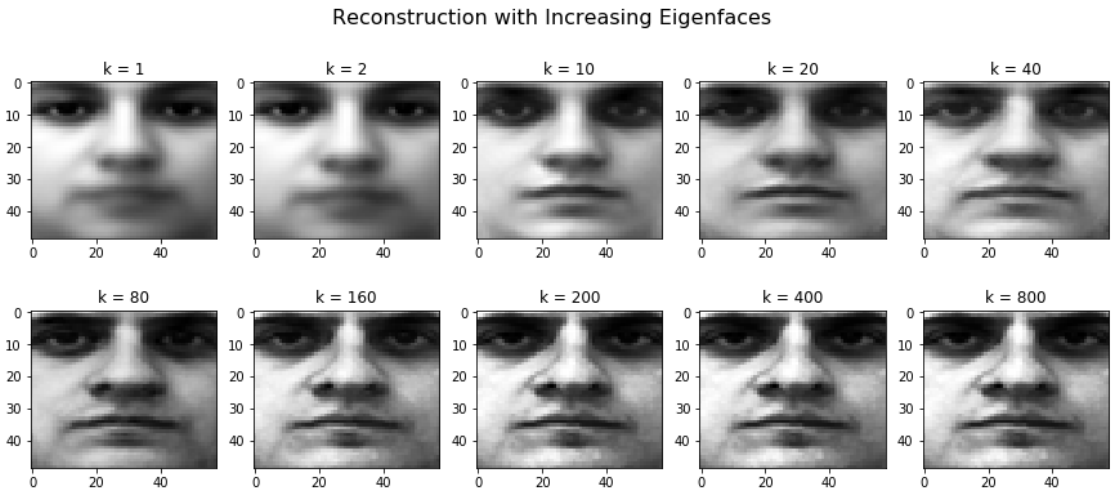
```
Text(0.5,0.98,'First 10 Eigenfaces')
```



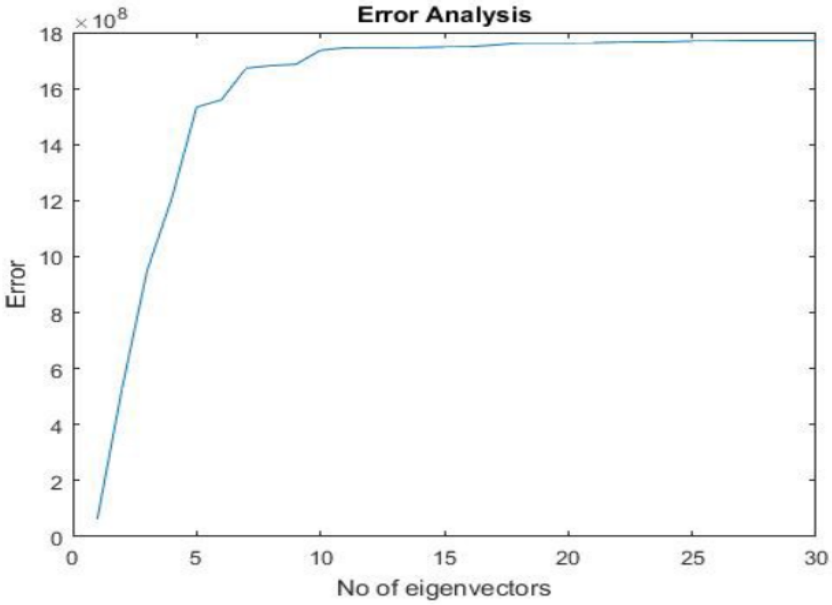
- Reconstruction of Face with eigenfaces with variance up to 99%

Reconstruction with Eigenfaces

```
fig, axs = plt.subplots(2,5,figsize=(15,6))
for k, i in zip([0,1,9,19,39,79,159,199,399,799],np.arange(10)):
    # Reconstruct the first picture '1a.jpg' whose index is 0.
    weight = faces_norm[0,:].dot(eigen_vecs[:,:k]) # Get PC scores of the images
    projected_face = weight.dot(eigen_vecs[:,:k].T) # Reconstruct first face in dataset using k PCs
    ax = plt.subplot(2,5,i+1)
    ax.set_title("k = "+str(k+1))
    plt.imshow(projected_face.reshape(49,58)+mean_face.reshape(49,58),cmap='gray');
fig.suptitle(("Reconstruction with Increasing Eigenfaces"), fontsize=16);
```



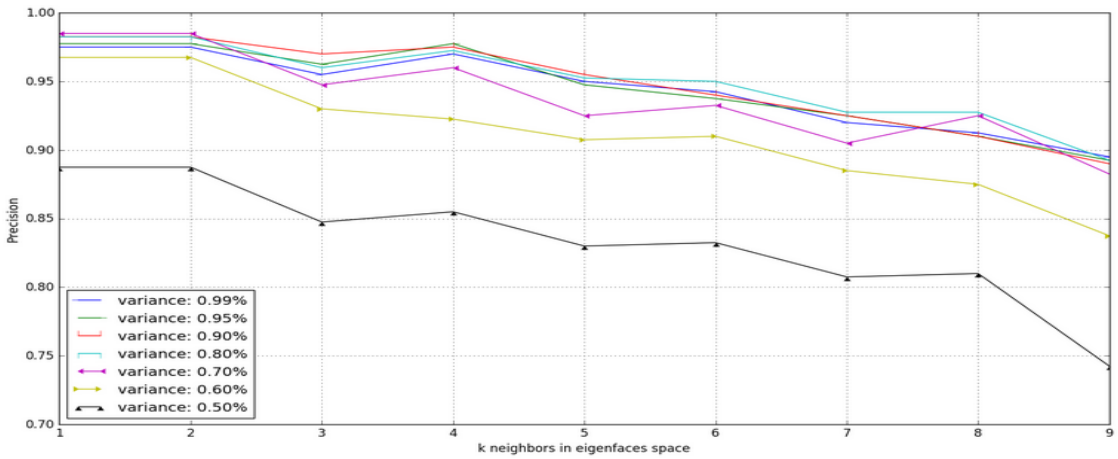
Error Analysis:



Convergence occurs after 10th eigenvector

You can see that error increases as you increase the no of eigenvectors because though error is low in first couple of eigenvectors it is very hard to represent to basis of entire training set and as a result it ends up recognizing wrong image.

Recognition For an image, we projected it into the eigenspace, and the image was considered as the linear combination of the eigenfaces. We only used the top n eigenfaces to represent an image, where the n was determined by how much variance this sub-eigenspace can represent. We investigated 99%, 95%, 90% and 80% percent of variance



At $n \geq 10$ eigenvectors image is recognized correctly and error remains more or less the same. In error analysis when error more or less remains the same and correct image is recognized then we can say that it has hit convergence. Also, as you train with a greater number of images it is likely that convergence point will go up as you will need more eigenvectors to represent basis.

How we can improve accuracy performance of this algorithm?

- We can use training dataset of images of same person with many different kinds of angle images of same person like side face, front face, tilted etc. This improves our chances of recognizing the face drastically
- Also, we can club all image vectors of same person different images and find principal components of that matrix instead of just comparing on single image vector basis.

V. CONCLUSION

The eigenface approach to face recognition was motivated by linear algebra and information theory, leading to idea of basing face recognition theory on a small set of image features that best approximates the set of known face images, without requiring that they correspond to our intuitive notions of facial parts and features. Although it is not an elegant solution to the general recognition problem, eigenface approach is still providing a practical solution that is well fitted to the problem.

Eigen face approach can be made to work at very high accuracy with the lowest rejection rates, which most of the approaches lack. The success rate is calculated as 94.74%. To increase the success rate, the eigenfaces method can be fortified with the use of additional information, such as the face triangle.

From the results, it can be concluded that, for recognition, it is sufficient to take about 10% eigenfaces with the highest eigenvalues. It is also clear that the recognition rate increases with the number of training images per person. It is obvious that if the minimum distance between the test image and other images is zero, the test image entirely matches the image from the training base. If the distance is greater than zero but less than a certain threshold, it is a known person with other facial expression, otherwise it is an unknown person.

Bibliography

Face Recognition Using Eigen Faces

1. M. Turk and A. Pentland, *"Eigenfaces for Recognition*", Journal of Cognitive Neuroscience, vol. 3, no. 1, pp. 71–86, 1991.
2. I, Yazar., H, S, Yavuz., & M, A, Çay. (2009) *Face Recognition Performance Comparisons by Using Tanh and Gauss Functions in the ICA Method*, IATS'09, Karabük, Turkey.
3. Gupta, S., Sahoo, O., Goel, A., & Gupta, R. (2010). A New Optimized Approach to Face Recognition Using Eigenfaces. *Global Journal of Computer Science And Technology*, Retrieved from shorturl.at/eELX8
4. Wills, H (2014, April 17). *A Study of Linear Algebra for Computer Vision*.
5. Anshul, G., Kiran K. (2017, March). *A Study of Linear Algebra for Computer Vision*. Volume 5, Issue 3.
6. Shah, D (2017, April 1). Retrieved from shorturl.at/hoCY9
7. shorturl.at/ptST4
8. Face recognition using eigen faces. *Python-Machine Learning*. Retrieved from shorturl.at/eoARZ
9. Fares, J. (2017) *Face Recognition Machine Vision System Using Eigenface.*