[[1]](#footnote-1)

Photo Album Organizer

Raheen Mazgaonkar, 4714-4316, Student, University of Florida

*Abstract*—An average person has hundreds of photos, stored in digital form. These photos are generally organized based on timeline. Apart from this for any further kind of classification, the user would have to manually sort the images [3][4]. This project suggests an application for organizing photos automatically based on the people present in them using face recognition techniques.

*Index Terms*— Face Recognition, PCA, Eigen Faces, Image Organizer.

# Introduction

W

ITH the advent of digital cameras, the ease of taking images has greatly increased. This has led to every individual possessing hundreds of images. With these large numbers of photos, it has become essential to organize these photos for easy retrieval by the user. Photos can be organized based on various categories such as time when the photo was taken, where it was taken, people present in the photo etc. Of these, sorting based on time is one of the earliest and most widely used method of sorting photos. With the presence of GPS in cameras and mobile phones it is also possible to sort images based on the location where they were taken.

In this project, we shall emphasize on organizing photos based on the people present in it. We will use face recognition techniques in order to identify the people present in the image. When the user adds first few images of person, we shall ask for a tag(simply name of the person) for the image. Subsequently, when more images are added they shall be identified and appropriate tags will be given automatically. When user enters a tag, all the images containing those tags will be displayed. Furthermore, the user will also be able create an album of all the people who are given that tag. Additionally, along with the name tag, each user can also be given a relation tag such as family, colleague etc. We can also search and create albums based on these tags.

Thus, face recognition forms the core component of this project. Over time, various techniques for face recognition have been developed. However, all these techniques follow three basic stages: Face detection, Feature Extraction and Face Recognition [1] (Fig 1). Though considered as three separate stages, sometimes they may overlap.

In this project, we shall concentrate only on the feature extraction and face recognition stage. Hence, in order to, make face detection easier all the training and testing images will be solo images with background cropped so that only the face is visible in the image.

Face Recognition techniques can be classified into three types [2]. First one is holistic in which the faces are identified using the entire face and not some local features. Second one is feature-based approach, in which distinctive features of the face such as eyes, mouth, nose, etc. are extracted. The geometric relation between these features is then computed and the image is reduced to a vector of geometric features. These are then fed into a structural classifier for recognition. And lastly, we have the hybrid methods which use both global and local images to recognize faces [1]. In this project, we shall use one of the holistic face recognition algorithms i.e. Principal Component Analysis(PCA) using Eigen Faces.

Face Recognition

Face Detection

Feature Extraction

Image

Result

Fig 1. Stages of face recognition.

# Description

## Project Overview

Unlike traditional face recognition applications, where we require a separate training and testing stage, in this application both are integrated. Fig. 5 gives the flow of the application. For accurate testing, a substantial training set is required. Hence, we do not start our testing and training cycle from the beginning. Instead, a minimum value is set. If the number of images is below this value then the user is required to input tags for the image. Once the number of images crosses this point, whenever an image is added we shall apply PCA to recognize the person. If the image is recognized, then it is stored with the appropriate tag (Fig 2). The user may also add image of a person who is not all already present in the training set. In this case, our algorithm will not be able to identify it. In such cases again, the user will be asked to input tags (Fig 3). Sometimes, if the number of images of a person is very low they are chances that the images are identified incorrectly. In order, to overcome the issue of false acceptance after every identification we shall ask the user to verify the result (Fig 2). Only after the user replies in affirmative, the image will be stored. If we are unable to identify the image, then user will be again be asked to input tag for that image. As the number of images per person increases, accuracy will also increase, and number of user inputs will decrease. Whenever an image is added we shall carry out training process again to increase accuracy.

When the user wants to view images of a person, he just has to enter the person’s name and all the images of that person will appear. Similarly, if the user wants to view images of all the people associated with a relation, he just has to enter the relation and images of everybody belonging to that relation will appear. Furthermore, the user can also create an album of all these images (Fig 4).

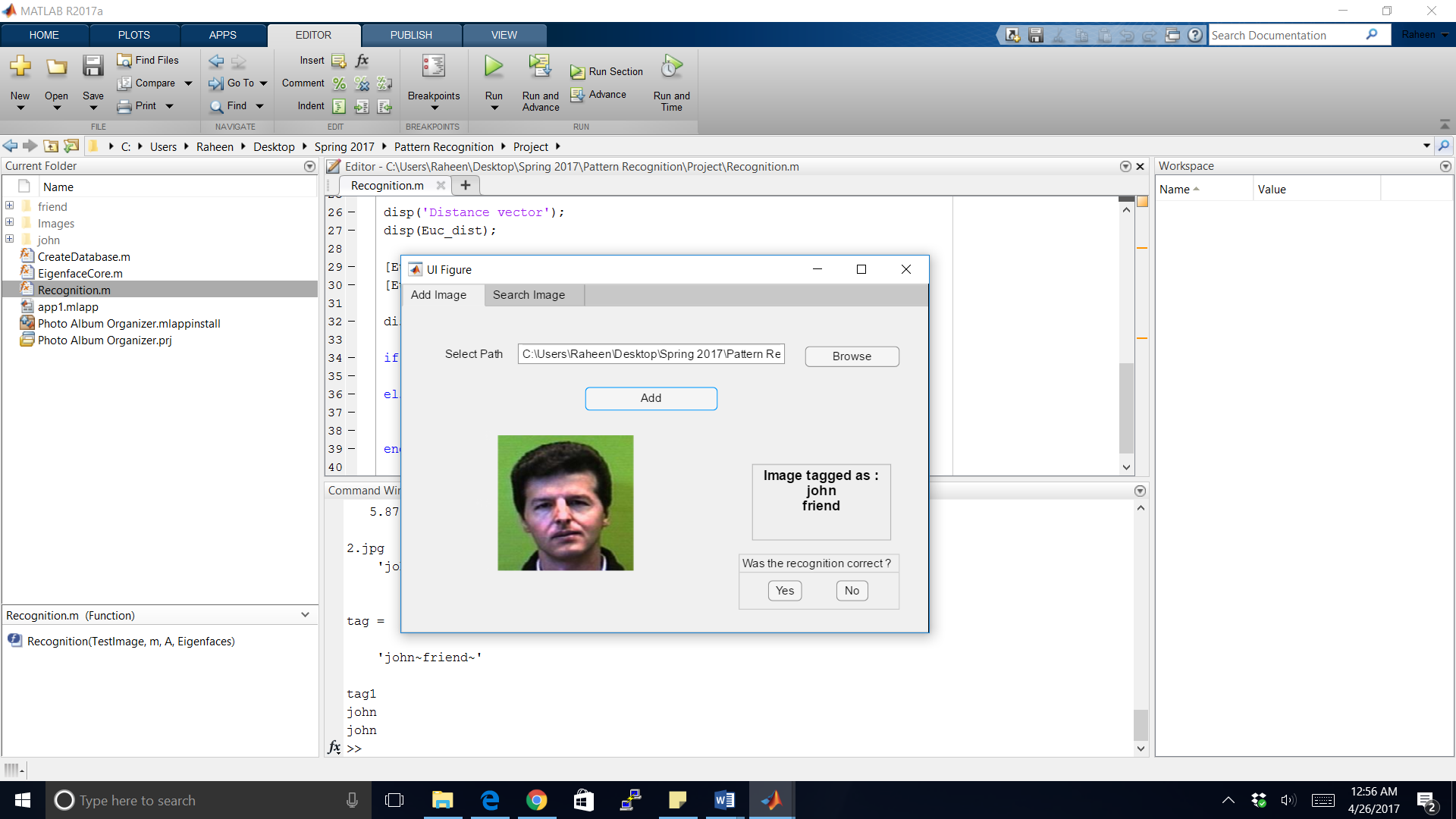


Fig 2. Successful verification

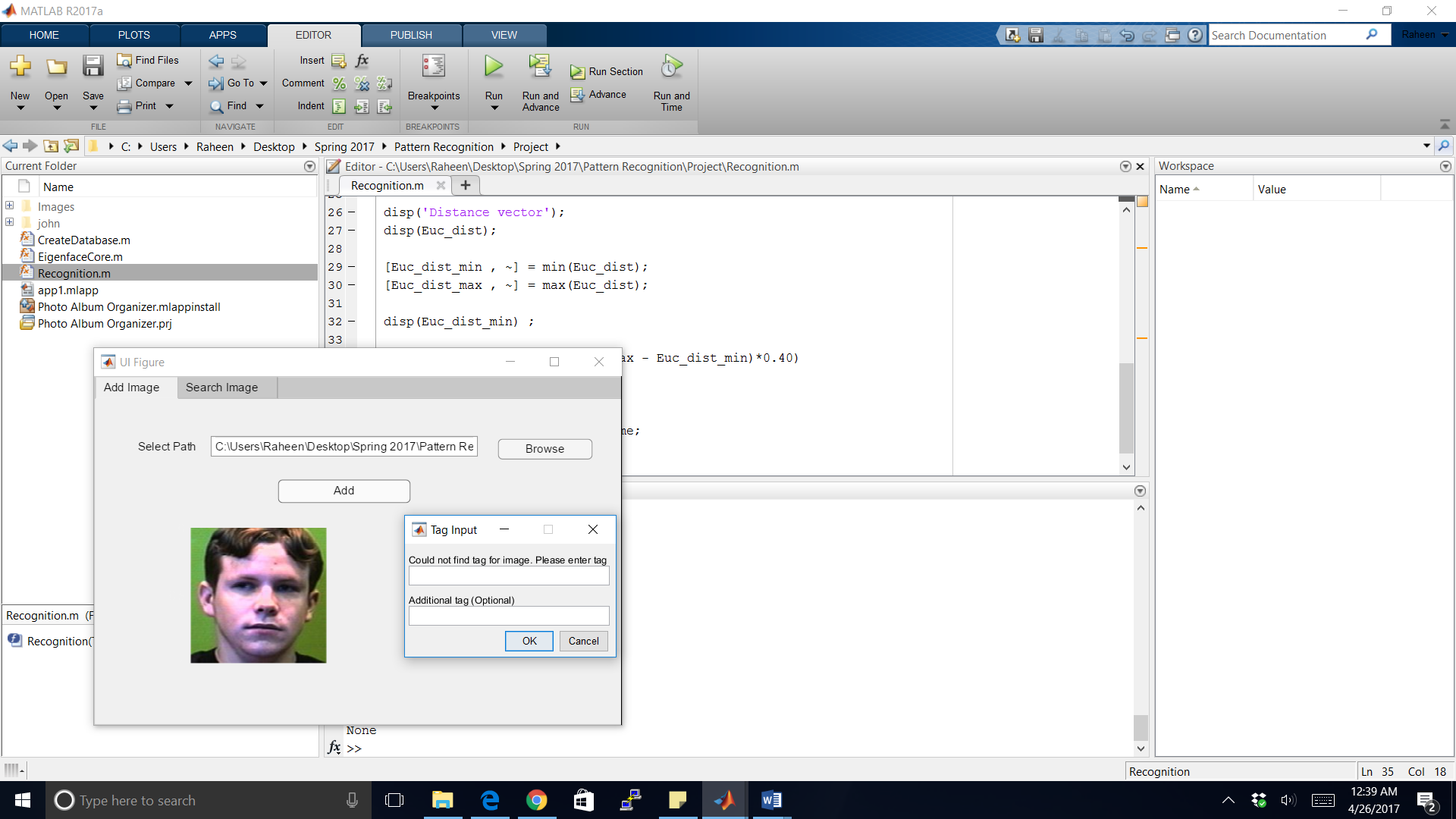


Fig 3: Unsuccessful verification

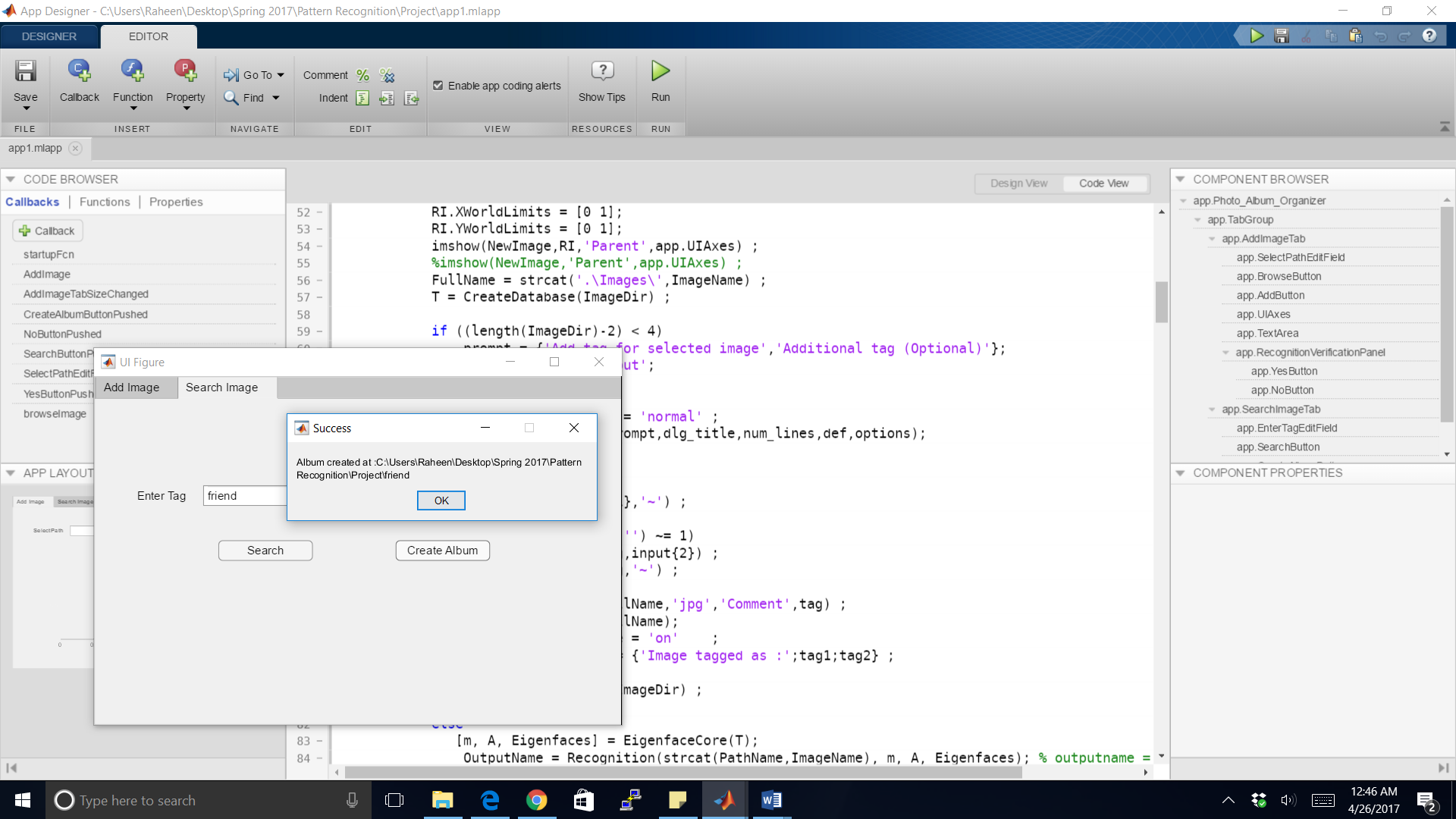


Fig 4: Album generation

## PCA Algorithm

Principal Component Analysis (PCA), is one of most popular and widely used face recognition technique. It was first studied by Sirovich and Kirby [5]. In this project, we shall implement PCA algorithm using eigenfaces. Here, PCA is used to reduce dimensions so that we retain only the information which is necessary. Fig 6 gives overview of the steps involved in PCA [13]. We can elaborate the steps involved in the algorithm as follows: [6][14][15]

1. First, every the NxN two dimensional image is converted to an N2x1 one dimensional vector by simply using row appending method. So, our dataset T is represented as

T = {t1, t2, …….. , tm};

Where,

ti = ith vector and

m = number of images in the dataset.

1. Next, we carry out normalization of the vectors. Normalization is nothing but removing the common features from each image so that only the features unique to each image are retained. To normalize the face vector, we calculate the average face and then subtract the average face from every image.

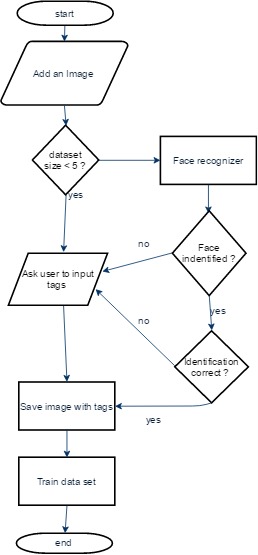


Fig 5: Flowchart of application

Normalized face can be given by

Where,

Φi represents normalized face vector and

µ represents average face

1. Next, we need to calculate the eigen vectors associated with these face vectors. To calculate eigen vectors we first need to calculate the covariance matrix which is given by C = AAT where A is simply the matrix obtained by combining m vectors. So, dimensions of A and AT will be N2 x m. Hence, C will become N2 x N2 in size which is very huge. Computing eigen vectors from here will be impractical. Hence, we need to map eigen vectors from higher dimensionality to lower dimensionality. To do this we calculate covariance matrix as C = ATA. In this case, size of the covariance matrix becomes m x m. Finding eigen vectors from here becomes a more feasible task. We shall calculate m eigen vectors from here. We can always map eigen vectors from higher dimensionality to lower dimensionality using the formula:

Where

represents the eigen vector in higher dimension and represents the eigen vector in lower dimension

1. Next, from the m vectors found in the previous step, we need to select k vectors where k<=m such that these k vectors represent all the faces present in the data set.
2. Finally, we shall represent each image as a linear combination of these k images. Thus, each image can be associated with a vector Ω such that,

Where

is the weight of the jth vector in the ith image.

This is the last step involved in the training process.

1. For testing an image, we shall follow steps 1-5 of the image to get its Ω value. Then we shall find the distance di of this value from all the Ωi values obtained in step 5.
2. Next, we shall find the minimum di. If this value is less than the threshold value (which is generally found through experiments), then the face is matched with the one whose vector is represented by Ωi . Otherwise we say that the face cannot be identified.

# Evaluation

## Setting Threshold Value

The PCA algorithm finds images from the dataset which has minimum distance from the input image. However, there is no threshold set on how much minimum distance ensures accurate verification of the image. Generally, this threshold​ is depends​ on the needs of the application. If the application is resistant to false positives the threshold can be on the higher side otherwise, is should be on the lower side. Mostly, when training size is set, the threshold can be found by testing with various images and using the minimum distance to decide the threshold. If the application is not at all resist to false positives we can use the highest minimum distance otherwise​ we can use the average.

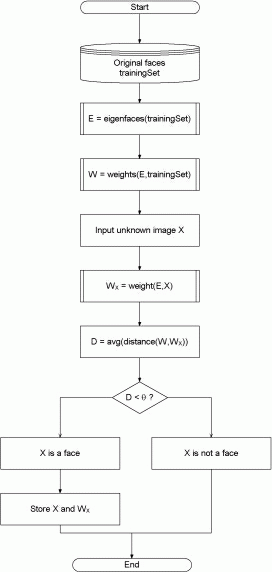


Fig 6: Overview of steps involved in PCA

For this application, however, the training size is not fixed so the approach mentioned above cannot be used. Hence, instead of keeping the threshold as a constant number, we assign it as a function of the minimum​ and maximum euclidean distance between the training and testing images. Based on tests, carried on training sets of various sizes, we reached to the following threshold: (max euclidean distance - min euclidean distance)\* 0.4.

# Conclusion

Face recognition is a very powerful tool and finds way in several applications. One of the methods of Face Recognition, that is, Principal Component Analysis was studied. This algorithm was then used to develop a useful application i.e. Photo Album Organizer which helps users to sort images based on the people present in them. It does this by applying PCA on each image and identifying the people present in them.

References

1. W. Zhao, R. Chellappa, P.J. Phillips, A.Rosenfeld, "Face Recognition: A Literature Survey" in ACM Computing Surveys (CSUR), Volume 35 Issue 4, December 2003, pp 399-458.
2. R. Jafri, H.R. Arabnia, "A Survey of Recognition Techniques" in Journal of Information Processing Systems, Vol. 5, No. 2, June 2009, pp 41-68.
3. L. Zhang, L. Chen, M. Li, H. Zhang “Automated Annotation of Human Faces in Family Albums”, in Multimedia ’03 Proceedings of the eleventh ACM international conference on Multimedia, pp 355-358.
4. Chen L., Hu B., Zhang L., Li M. and Zhang H.J., “Face annotation for family photo album management”, International Journal of Image and Graphics, p.1-14, Vol. 3, No. 1, 2003
5. L. Sirovich, M. Kirby, "Low-Dimensional Procedure for Characterization of Human Faces", J. Optical Soc. Am., vol. 4, pp. 519-524, 1987.
6. K. Delac., M. Grgic.,S. Grgic., “Independent Comparative Study of PCA, ICA, and LDA on the FERET Data Set, in International Journal of Imaging Systems and Technology, Vol. 15, Issue 5, 2006, pp. 252-260.
7. A. Georghiades, P. Belhumeur, and D. Kriegman's paper, "From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose", PAMI, 2001
8. http://www.cs.otago.ac.nz/cosc453/student\_tutorials/principal\_components.pdf
9. http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec\_tutorial.html?highlight=face%20recognition
10. https://medium.com/@ageitgey/machine-learning-is-fun-80ea3ec3c471#.ig3by5j8r
11. http://www.anefian.com/research/face\_reco.htm

[12] http://cswww.essex.ac.uk/mv/allfaces/

[13] http://openbio.sourceforge.net/resources/eigenfaces/

eigenfaces- html/facesOptions.html

[14] http://www.vision.jhu.edu/teaching/vision08/Handouts/

case\_study\_pca1.pdf

[15] S.P. Bahurupu, D.S. Chaudari, "Principal Component Analysis for

Face recognition", International Journal of Engineering and

Advanced Technology (IJEAT), Volume-1, Issue-5, June 2012.

[16] S.K. Bhattacharyya, k. Rahul, "F0ace Recognition by

Linear Discrimininant Analysis", International Journal

of Communication Network Security, Volume-2, Issue-2, 2013

[17] P. Jonathon Phillips. 1999. Support vector machines applied to

Face recognition. In Proceedings of the 1998 conference on Advances

in neural information processing systems II, David A. Cohn (Ed.).

MIT Press, Cambridge, MA, USA, 803-809.

[18] https://www.mathworks.com/matlabcentral/fileexchange/

17032-pca-based-face-recognition-system

1. [↑](#footnote-ref-1)