Face Recognition for Different Facial Expressions Using Principal Component analysis

ASHISH SHRIVASTAVA*, SHEETESH SAD#

#Department of Electronics & Communications, CIIT, Indore Dewas Bypass Road, Arandiya P.O.
Jhalaria, Indore-452016(M.P.), India

ABSTRACT -- The face is our primary focus of attention in social intercourse, playing a major role in conveying identity and emotion. We can recognize thousands of faces learned throughout our lifetime and identify familiar faces at a glance even after years of separation. This skill is quite robust, despite large changes in the visual stimulus due to viewing conditions, expression, aging, and distractions such as glasses, beards, changes in hairstyle. Though human faces are complex in shape, face recognition is not difficult for a human brain whereas for a computer this job is not easy. In this paper presents and analyzes the performance of Principle Component Analysis (PCA) based technique for face recognition. We consider recognition of human faces with two facial expressions: single and differential. The images that are captured previously constitute the training set. From these images eigenfaces are calculated. The image that is going to be recognized through our system is mapped to the same eigenspaces. Next I used classification technique namely distance based used to classify the images as recognized or non-recognized. Presently I got result for the single facial expression now I am working for different facial expression.

KEYWORDS: Eigen faces, Principal component analysis, Face recognition

I. INTRODUCTION

The face is our primary focus of attention in social intercourse, playing a major role in conveying identity and emotion. The human ability to recognize faces is remarkable. We can recognize thousands of faces learned throughout our lifetime and identify familiar faces at a glance even after years of separation. This skill is quite robust, despite large changes in the visual stimulus due to viewing conditions, expression, aging, and distractions such as glasses, beards or changes in hairstyle. Though human faces are complex in shape, face recognition is not difficult for a human brain whereas for a computer this job is not easy. The complexity of recognition is prominent and several algorithms are reported in literature that could achieve the recognition with high degree of accuracy. Face recognition system is widely used in different areas [9] that include a) criminal record and identification, b) Robot vision, c) security system, d) human computer interaction, e) image and field processing.

Face recognition system is divided into two categories, **i) appearance based and ii) component based.** For appearance based, we consider the holistic feature or the whole face image as our feature for recognition. On the other hand, in component based face recognition, we consider geometrical relationship of different components of face such as eye, nose, lip etc as the features of a recognition system. **Principal Component Analysis (PCA)** is a fast and efficient technique that is widely used for appearance based face recognition. Principal Component Analysis (PCA) is a fast and efficient technique that is widely used for appearance based face recognition. This technique is also used for dimensionality reduction in different areas that include image processing, signal processing and data mining. This technique is sometimes also called eigenfaces. The eigenfaces approach is chosen for this study considering its capability of recognizing real time images where orientation, illumination and distortion are continuously changing. This work focuses on how the images with real time attributes affect the recognition feature of eigenfaces technique. Our primary objective for this research is to minimize the complexity in calculation for bigger matrices. For example, if we have 120 pictures with the size of (180 × 200), we will have a very big number while calculating the one dimensional vector from 2D matrix (by calculating 180 × 200 × 120) which is a very big number. By using the eigenvectors, we could minimize the use of all the images and reduce them for example 40 pictures which will also bring down our total calculation to (180 × 200 × 40). Though, we are using lesser amount of data, we will still get the same level of accuracy. Besides, we could make the size even smaller by changing the order of matrix multiplication which in turn reduces the principal components, and the end we could work only on (40×120) matrix with the same level of accuracy..

II. MATHEMATICAL APPROACH

Our face recognition system consists of several steps. Each of the steps is described in detail in below:

Initialization and Finding Principal Components

At first we take images. These images are nothing but the matrix which has pixel intensity at different rows and columns. This image could be viewed as a vector also. If an image has height, h and width, w, then we could formulate this image as w vectors, where each vector has h dimensions. The rows of the images are placed one after another like the Figure1 below:

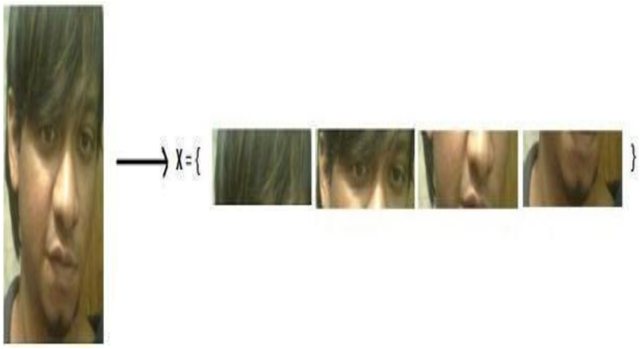


Fig 1: Formation of the face's vector from face's images

The vector which is *represents* our image and this image has a certain space so this is called image space. If we have N images, we have image space dimension as N □ w□ h . In this image space all images are represented by w by h pixels. These images under same image space look like each other. They all have two eyes, a nose, a mouth etc located at the same image space.

Now we will build the face space from the image space. The main task of building a face space is to describe the face images. The basis vector of this space is called principal component. The dimension of the face space will be M □ w□ h . In the face space all pixel is not relevant and each pixel depends on the neighbors. So the dimension of face space is less than the dimension of the image space. We could find the principle components of the faces by finding the **eigenvectors** of the **covariance matrix** of the set of face images. This eigenvectors are basically a set of features which characterize to the maximum variations between face images. Each of this images that comes from the image space contribute more of less to the eigenfaces. So we can display eigenvector as a sort of ghostly faces which we call **eigenfaces**. Actually eigenfaces do not exist in real world. We could not say we can build or create eigenface of a particular image face which is in the image space. Eigen face actually is an imaginary face which is a combination of all the images with in a particular image space.

We present the mathematical formulation of eigenfaces below.

1. We obtain N training images $I1, I2, \ldots. IN$.Each of these images have dimension w □ h . Convert these images into vector space by concatenation. After the concatenation a matrix is converted to a vector.
2. Represent each image I_i with its corresponding Vector λ_i

$$\begin{bmatrix} B_{11} & B_{12} & \cdot & B_{1h} \\ \vdots & \vdots & \vdots & \vdots \\ B_{w1} & B_{w2} & \vdots & B_{wh} \end{bmatrix} \xrightarrow[concatenation]{} \begin{bmatrix} B_{11} \\ \vdots \\ B_{1h} \\ \vdots \\ B_{wh} \end{bmatrix} \triangleq \lambda_i$$

3. Calculate the mean face vector □ by the following equation

$$\omega = \frac{1}{N}\sum_{i=1}^{N} \lambda_i$$

Subtract the mean face, from each face vector, to get a set of vector μ_i.

λ_i

$$\mu_i = \lambda_i - \varpi$$

The purpose of subtracting the mean image from each image vector is to keep only the distinguishing features from each face by removing the common information.
Find the covariance matrix C by the following equation:

$C = A^T A$ where, $A = [\mu_1, \mu_2, \ldots\ldots \mu_N]$

Find the eigenvalues and eigenvectors for the covariance matrix C. Sort the eigenvectors according to the eigenvalues. Take the first *M* eigenvectors that have higher eigenvalues. Now each eigenvector will have *N×1* dimension. Let us name those eigenvectors as η_i for *i=1, 2…M*. *Projection of new face to eigenfaces*

When a new image is encountered, calculate the set of weights based on the new or input image and the *M* eigenfaces by projecting the input image onto each of the eigenfaces. The mathematical formulation is given below:

Let us consider the new image as I_{new}

Find out the *M* eigenface components, Ψ_l, by projecting the new image

$$\Psi_l = \gamma_l^T (I_{new} - \varpi) \quad \text{for } l=1,2,\ldots M.$$

Where,

for *l=1, 2,….M.*

$$\gamma_l = \sum_{k=1}^{N} \eta_{lk}\, \mu_k$$

Create a new feature vector, $\Omega_{......}$ *for the new image by concatenating eigenface components,* □*1*

$$\Omega_{new} = [\Psi_1, \Psi_2, \ldots\ldots \Psi_N]$$

Face Recognition by classification algorithms

The last step of the face recognition system is to identify the new face to be recognized or not recognized. If the face is recognized the system will tell the person's name for whom the face has been recognized. In the other word, if we have *N* persons in the image database, we say that there are *N* classes where each individual person representing a class.Comparison is done by the Euclidian distance between two features, □*new* and □*i* , if the distance is less than some predefined threshold, *t*, we say that the image is recognized. The class of the new image will be one that has the least Euclidian distance with the new image, providing this distance is less than the threshold.

III. SYSTEM DESCRIPTION

We have developed our system by using MATLAB 2008a (version7.6).because we found that the performance of MATLAB 2008a (version 7.6) is better than other programming language. Besides, MATLAB is a high-level language for technical computing development environment for managing code, files, and data interactive tools for iterative exploration, design, and problem solving. It also supports Mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, and numerical integration 2-D and 3-D graphics functions for visualizing data. For our system we have used certain method and techniques that are offered explicitly by MATLAB. These are as follows:

A*) Image reading*: MATLAB can easily read an image and convert the image in a certain matrix. Later on we can use the image matrix for our related work. MATLAB can read an image of 8 bit up to 32 bit.

B) *Image conversion from RGB to GRAYSCALE*: MATLAB can convert an image from RGB to GRAYSCALE. This computational task can be done by MATLAB command. If RGB image is 32 bit it represent RED for 8 bit, GREEN for 8 bit, BLUE for 8 bit.

C) *Image resize*: MATLAB command can be used to resize a certain image in to any size that MATLAB allow.

D) *Convert MATRIX to 1 dimensional VECTOR*: We can use certain technique to convert a matrix to 1 dimensional vector which helps us to compute the desired output.

E)*Matrix transpose*: In MATLAB we can easily transpose a certain matrix. In our system, at first of the system take the images. These images are captured by webcam or other image capturing source. We calculate the mean image and eigenfaces.

The entire sequence of training and testing is sequential and can be broadly classified as consisting of following three steps:

1. Database Preparation
2. Training
3. Testing

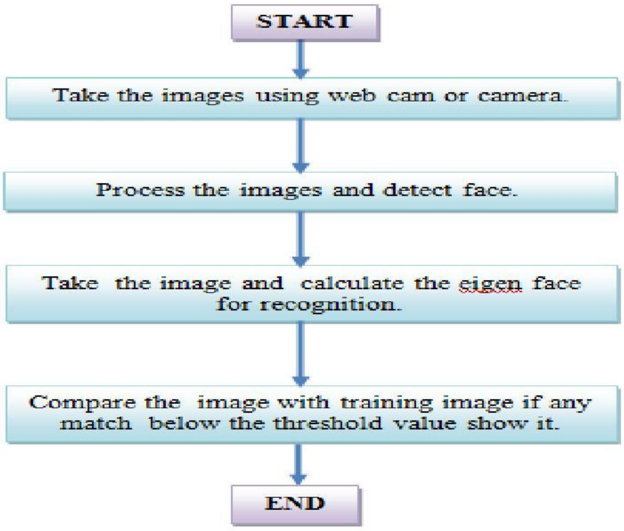

START
Take the images using web cam or camera.
Process the images and detect face.
Take the image and calculate the eigen face for recognition.
Compare the image with training image if any match below the threshold value show it.
END

Figure 4: Basic flow diagram of the face recognition system

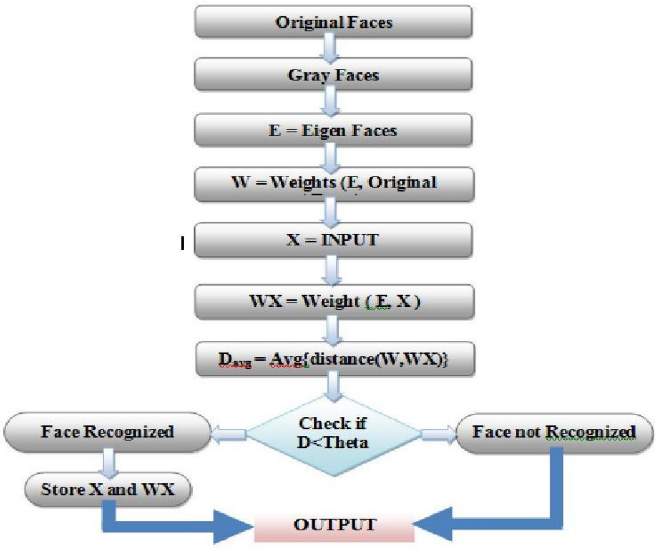
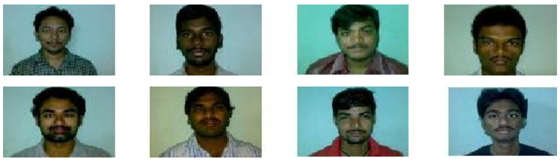


Figure 5: Detail flow of execution in the system

III. RESULTS & DISCUSSION

At first I take 8 images in this processing. These images are nothing but the matrix which has pixel intensity at different rows and columns.



Each image has its own intensity values at different rows and columns. This image could be viewed as a vector also. If an image has height, *h* and width, *w*, then we could formulate this image as *w* vectors, where each vector has *h* dimensions. The rows of the images are placed one after another like the Figure below:

$$= \begin{bmatrix} a_1 \\ a_2 \\ \cdot \\ \cdot \\ a_N \end{bmatrix}$$

$$= \begin{bmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ b_N \end{bmatrix}$$

$$= \begin{bmatrix} c_1 \\ c_2 \\ \cdot \\ \cdot \\ c_N \end{bmatrix}$$

$$= \begin{bmatrix} d_1 \\ d_2 \\ \cdot \\ \cdot \\ d_N \end{bmatrix}$$

$$= \begin{bmatrix} e_1 \\ e_2 \\ \cdot \\ \cdot \\ e_N \end{bmatrix}$$

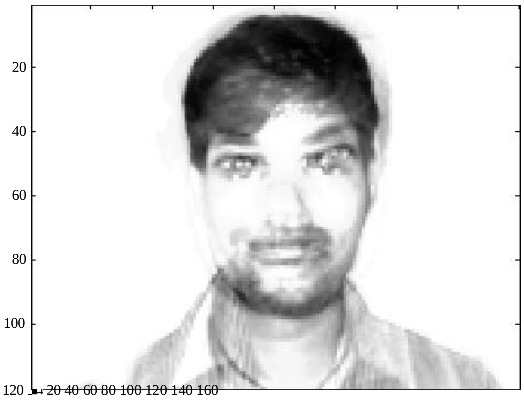
$$= \begin{bmatrix} f_1 \\ f_2 \\ \cdot \\ \cdot \\ f_N \end{bmatrix}$$

$$= \begin{bmatrix} g_1 \\ g_2 \\ \cdot \\ \cdot \\ g_N \end{bmatrix}$$

$$= \begin{bmatrix} h_1 \\ h_2 \\ \cdot \\ \cdot \\ h_N \end{bmatrix}$$

Now I calculate the mean face vector $\bar{m}$ by the following equation

$$\bar{m} = \frac{1}{M}\begin{bmatrix} a_1 + b_1 + c_1 + \ldots\ldots h_1 \\ a_2 + b_2 + c_2 + \ldots\ldots h_2 \\ . \quad . \quad . \quad . \\ . \quad . \quad . \quad . \\ a_N + b_N + c_N + \ldots\ldots h_N \end{bmatrix}, \text{ Where M=8.}$$



Mean Face

Subtract the mean face from each face vector, the purpose of subtracting the mean image from each image vector is to keep only the distinguishing features from each face by removing the common information.

$$\bar{a}_m = \begin{bmatrix} a_1 - m_1 \\ a_2 - m_2 \\ \ldots \\ \ldots \\ a_N - m_N \end{bmatrix} \quad \bar{b}_m = \begin{bmatrix} b_1 - m_1 \\ b_2 - m_2 \\ \ldots \\ \ldots \\ b_N - m_N \end{bmatrix} \quad \bar{c}_m = \begin{bmatrix} c_1 - m_1 \\ c_2 - m_2 \\ \ldots \\ \ldots \\ c_N - m_N \end{bmatrix}$$

$$\bar{d}_m = \begin{bmatrix} d_1 - m_1 \\ d_2 - m_2 \\ \ldots \\ \ldots \\ d_N - m_N \end{bmatrix} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \bar{h}_m = \begin{bmatrix} h_1 - m_1 \\ h_2 - m_2 \\ \ldots \\ \ldots \\ h_N - m_N \end{bmatrix}$$

Now I build a vector which is having order $N \times h$.

$$A = \begin{bmatrix} \bar{a}_m & \bar{b}_m & \bar{c}_m & \bar{d}_m & \bar{e}_m & \bar{f}_m & \bar{g}_m & \bar{h}_m \end{bmatrix}$$

Find the covariance matrix C by the following equation:

$$C = AA^T$$

Find the eigenvalues and eigenvectors for the covariance matrix C. Sort the eigenvectors according to the eigenvalues. Take the first M eigenvectors that have higher eigenvalues. When a new image is encountered, calculate the set of weights based on the new or input image and the M eigenfaces by projecting the input image onto each of the eigenfaces. The last step of the face recognition system is to identify the new face to be recognized or not recognized. Comparison is done by the Euclidian distance between two features, New and Training images, if the distance is less than some predefined threshold, t, we say that the image is recognized. The class of the new image will be one that has the least Euclidian distance with the new image, providing this distance is less than the threshold.

IV. CONCLUSIONS

We conclude with the following remarks. Training set and test images need to be taken in good, comfortable illumination settings and need to be frontal faces with minimal head tilt. Number of images in the training set is a significant factor. It impacts on defining the correct threshold value for accepting true matches and rejecting false matches. If we will take larger threshold then false recognition rate will increase. In this I showed all the result for different facial expressions.

ACKNOWLEDGMENTS

I would like to express my special thanks of gratitude to my guide who gave me the golden opportunity to do this wonderful project on the topic Face Recognition System for Different facial expressions using PCA, which also helped me in doing a lot of Research. I am making this project not only for marks but to also increase my knowledge.

REFERENCES

[1]. Matthew A. Turk and Alex P.Pentland. "*Eigenfaces for ecognisatoin*". Journal of cognitive neurosciences, Volume 3, Number 1, Nov 27, 2002. http://www.face-rec.org/algorithms/PCA/jcn.pdf

[2].Chi-Fa Chen, Yu-Shan Tseng and Chia-Yen Chen "Combination of PCA and Wavelet

Transforms for Face Recognition on 2.5D Images", Palmerston North, November 2003, Department of Electrical Engineering, I-Shou University, Kaohsiung, Taiwan. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=& arnumber=4700005

[3]. Ilkeratalay, "Face Recognition using Eigen faces", Istanbul technical university Jan 1996. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=& arnumber=4798953

[4]. Belongie, S., J. Malik, and C. Fuh. Matching shapes. In 8th International Conference on Computer Vision, volume 1, pages 454-463. IEEE Computer Society Press, July 2001.

[5]. Gonzalez, R.C., Richard E. Woods. Digital image processing, 2nd edition. Pearson Education, 2002.

[6]. Gonzalez, R.C. Digital Image Processing by MATLAB, 3rd edition Pearson Education, 2004.

[7]. MATLAB User guide, Retrieved on 28th. June, 2010. http://www.mathworks.com/access/helpdesk/hel p/techdoc/.

[8]. http://www.biometricscatalog.org/NSTCSubcom mittee/Documents/Face%20Recognition.pdf

[9]. Alex Pentland, Baback Moghaddam, and Thad Starner, "View-Based and Modular Eigen spaces for Face Recognition", *IEEE Conf. on Computer Vision and Pattern Recognition*, MIT Media Laboratory Tech. Report No. 245 1994

[10]. Stan Z.Li & Anil K. Jain "Handbook of Face Recognition Springer publications".