Real Time Application of Face Recognition Concept

Kandla Arora

Abstract*— Face Recognition concept is one of the successful and important applications of image analysis. It's a holistic approach towards the technology and have potential applications in various areas such as Biometrics, Information society, Law enforcement and Surveillance, Smart cards, Access control etc. This paper provides an overview of real time application of Face Recognition concept by generating a matlab code using image acquisition tool box. The basic approach used is Principal Component Analysis using Eigen faces, popularized by the seminal work of Turk and Pentland.

Index Terms*— Eigen faces, Eigenvectors, Face recognition, Principal component analysis (PCA).

I. INTRODUCTION

A facial recognition system is a computer application for automatically identifying or verifying a person from a digital image or a video frame from a video source. One of the ways to do this is by comparing selected facial features from the image and a facial database. The human face plays a major role in conveying identity and emotion. It is typically used in security systems and can be compared to other biometrics such as fingerprint or eye iris recognition systems. A notable advantage of facial recognition over other biometric recognition methods is that it is less cumbersome for end users.

This paper approaches to real time application of face recognition theory and is formulated based on still or video images captured either by digital camera or by a webcam. The faces considered here for comparison are still faces. Here we have developed a Matlab code initializing the webcam of a laptop, capturing the image and comparing it with the database of images present in the laptop.

II. TRADITIONAL TECHNIQUES

Recognition algorithm can be divided into two main traditional approaches,

A. Geometric approach

This is the historical way to recognize people. Geometric features may be generated by segments, perimeters and areas of some figures formed by the points. The featured set is studied to compare the recognition result. Distances in the feature space from a template image to every image in the database were calculated. Following to the FERET protocol, 5 nearest face images were derived and if there were photos of the query person then the result was considered positive. Each image was tested as a query and compared with others. The approach was robust, but it main problem is automatic point location. Some problem arises if image is of bad quality or several points are covered by hair.

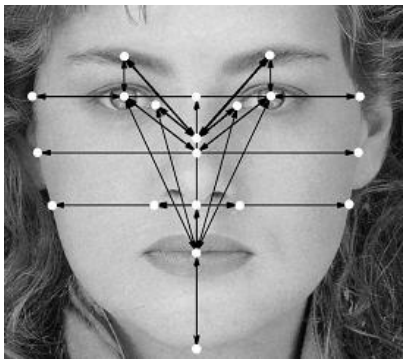

Fig.1. some facial points and distances between them are used in face recognition.

B. Photometric approach

It is a statistical approach that distills an image into values and compares the values with templates to eliminate variances. It relies on the input image in the presence of light and the geometric location of different angles. The photometric transformation is implemented on the source image, does not take into account photometric changes, i.e. changes in the pixel. The main restriction in this approach is that multiple registered images of the same person is required. Since it recognizes the new image by checking that it is spanned in a linear subspace of the multiple gallery images, it cannot handle the new images of a different person which is not included in the gallery set.

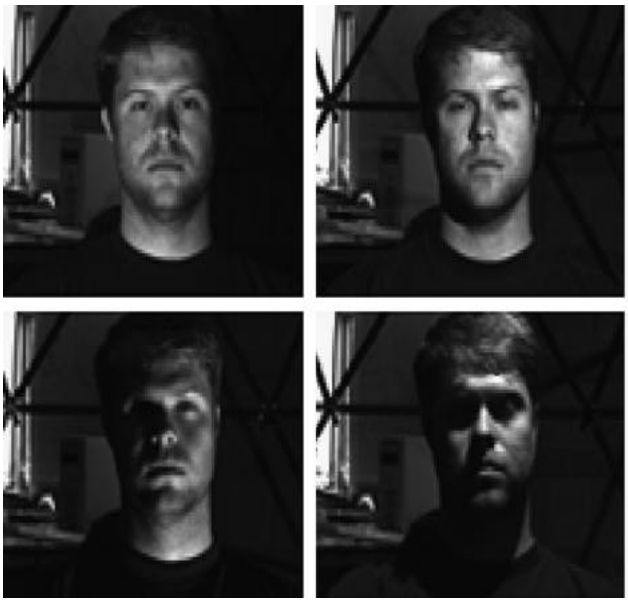

Fig.2. Example of facial images under different lightning conditions.

Manuscript received on November, 2012

Kandla Arora, Electronics and Communication department, Mahamaya Technical University, Agra, India.

III. POPULAR TECHNIQUES

According to the new technical era, some advancement has taken place and some techniques of facial recognition have achieved popularity. Some of the popular face recognition algorithms are as follows,

A. Principal Component Analysis (PCA) using Eigenfaces.

PCA is the simplest of the true eigenvector based multivariate analysis. Mathematically, it is an orthogonal linear transformation that transforms the data to a new coordinate system. The use of Eigenfaces is commonly called as Principal Component Analysis. With PCA, the image must be used of same size and they are normalized to line-up the eyes and mouth of the subjects within the image. Using PCA, dimension of data using data compression basics is reduced and precisely decompose the face structure into orthogonal and uncorrelated components know as Eigenfaces. The face image can be represented as a weighted sum or feature vector of the Eigenfaces which can be stored in a 1-D array.

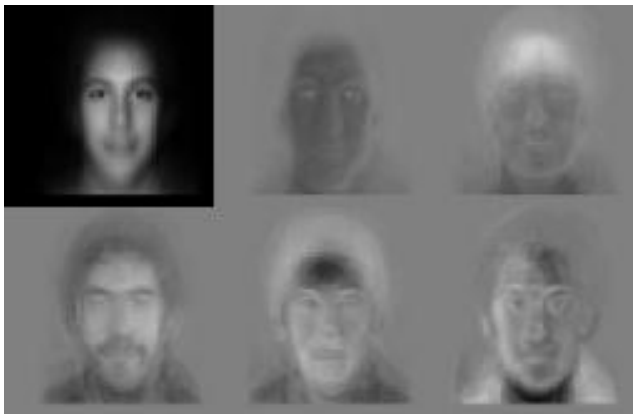

Fig.3. Standard Eigenfaces

B. Linear Discriminant Analysis

LDA is one of the most popular linear projection techniques for feature extraction. It finds the set of the most Discriminant projection vectors which can map high-dimensional samples onto a low-dimensional space. Using the set of projection vectors determined by LDA as the projection axes, all projected samples will form the maximum between-class scatter and the minimum within-class scatter simultaneously in the projective feature space. In Figure 4, where each block represents a class, there are large variances between classes, but little variance within classes.

Fig.4. Examples of four classes using LDA.

C. Elastic Bunch Graph Matching using the Fisherface Algorithm

EGM as one of the dynamic link architectures uses not only face-shape but also the gray information of image, and the Fisherface algorithm as a class-specific method is robust about variations such as lighting direction and facial expression. In the proposed face recognition adopting the above two methods, the linear projection per node of an image graph reduces the dimensionality of labelled graph vector and provides a feature space to be used effectively for the classification. In comparison with the conventional method, the proposed approach could obtain satisfactory results from the perspectives of recognition rates and speeds. It relies on the concept that real face images have many non-linear characteristics such as variations in illumination, pose and expression. It uses the Gabor wavelet transform. Figure 5, the Gabor jet forms a node on the elastic grid which describes the behaviour of the image around a given pixel. To detect the shapes and to extract the features of the image, the Gabor filter is convolved with the given image.

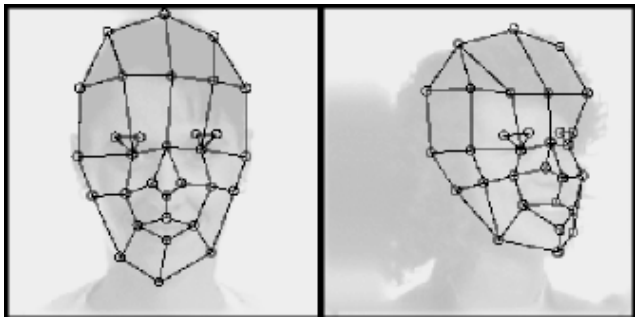

Fig.5, Elastic Bunch Map Graphing.

D. The Hidden Markov model

In HMM-based face recognition system, in which a scanning strategy is employed to simulate a human-like saccadic sequence, computed on the basis of the concept of saliency. The approach converts a face image into an attention based "scanpath," that is, a sequence composed of two types of information: Where information, the coordinates of the salient region in the face, and What information, local features detected in there. At the core of the scanning mechanism is the calculation of saliency. This calculation should be cheap enough that it can be applied to the whole image without significantly increasing time and space requirements, and it should be informative. With this approach, a cheap and parallel search for salient features will drive a serial and detailed analysis.

IV. PROPOSED TECHNIQUE

There are several techniques behind face recognition, we would be focusing on real time application of Principal Component Analysis using Eigen faces. In this method the difference between a given face image and a mean image is weighted, which is obtained by averaging a predefined set of faces. The training set is a group of face images from which the mean face is calculated. Face recognition takes place by linearly projecting the image to a low dimensional image space and weighting the difference with respect to a set of eigenvectors.

If the difference (weight) is bellow certain threshold, the image is recognized as a known face; otherwise, the face can be classified as an unknown face or not a face at all. This is done with the help of matlab code code initializing the webcam of a laptop, capturing the image and comparing it with the database of images (training images) present in the laptop. To perform face recognition, the similarity score is calculated between an input 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). There are various steps in performing this face recognition, we will be discussing step by step, however before that we will be discussing about Eigen faces used in this paper.

A. Brief note on Eigenfaces

Eigenfaces are a set of eigenvectors used in the computer vision problem of human face recognition. Specifically, the Eigenfaces are the principal components of a distribution of faces, or equivalently, the eigenvectors of the covariance matrix of the set of face images, where an image with N x N

Pixels is considered a point (or vector) in N2-dimensional space. The idea of using principal components to represent human faces was developed by Sirovich and Kirby [1] (Sirovich and Kirby 1987) and used by Turk and Pentland [2] (Turk and Pentland 1991) for face detection and recognition.

Mathematically, it is simply finding the principal components of the distribution of faces, or the eigenvectors of the covariance matrix of the set of face images, treating an image as a point or a vector in a very high dimensional space. The eigenvectors are ordered, each one accounting for a different amount of the variations among the face images. These eigenvectors can be imagined as a set of features that together

Characterize the variation between face images. Each image locations contribute more or less to each eigenvector, so that we can display the eigenvector as a sort if "shadowy" face which we call an eigenface.

B. Approach followed for facial recognition using Eigenfaces

The eigenfaces approach for face recognition involves the following two operations:

1. Initialization process:

The steps involve in initialization process are

- Acquire a set of training images.
- Calculate the Eigenfaces from the training set, keeping only the best *M* images with the highest eigenvalues. These *M* images define the "face space". As new faces are experienced, the Eigenfaces can be updated.
- Calculate the corresponding distribution in *M*-dimensional weight space for each known individual (training image), by projecting their face images onto the face.

2. Recognition process:

Having initialized the system, next process involves is recognition process and the steps involves in this process are,

- Given an image to be recognized, calculate a set of weights of the *M* Eigenfaces by projecting the it onto each of the Eigenfaces.
- Determine if the image is a face at all by checking to see if the image is sufficiently close to the face space.
- If it is a face, classify the weight pattern as either a known person or as unknown.
- (Optional) Update the Eigenfaces and/or weight patterns.
- (Optional) Calculate the characteristic weight pattern of the new face image, and incorporate into the known faces.

V. APPLICATION OF PCA IN FACIAL RECOGNITION

Assume a face image I(x,y) be a two-dimensional M by N array of intensity values, or a vector of dimension MxN. For simplicity the face images are assumed to be of size NxN resulting in a point in N2 dimensional space. An ensemble of images, then, maps to a collection of points in this huge space.

The steps involved are as follows,

1. Obtain a set with M face images, in our example M=3. Each image is transformed into a vector of size N and placed into the set.

$$S = \{ \Gamma_1, \Gamma_2, \Gamma_3, \ldots\ldots\ldots, \Gamma_M \}$$

Fig.6. The training images set for the analysis

2. Obtain the mean image **Ψ**,

$$\Psi = \frac{1}{M} \sum_{n=1}^{M} \Gamma_n$$

Fig.7. The mean image of the training set

3. Find the difference **Φ** between the input image and the mean image

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

4. Make a set of M orthonormal vectors, $\mathbf{u_n}$, which best describes the distribution of the data. The kth vector, $\mathbf{u_n}$, is chosen such that

$$\lambda_k = \frac{1}{M}\sum_{n=1}^{M}\left(u_k^T \Phi_n\right)^2$$

Is a maximum, subject to

$$u_l^T u_k = \delta_{lk} = \begin{cases} 1 & \text{if } l = k \\ 0 & \text{otherwise} \end{cases}$$

Here, $\mathbf{u_k}$ and $\boldsymbol{\lambda_k}$ are the eigenvectors and eigenvalues of the covariance matrix C

5. The covariance matrix C is obtained in the following way

$$C = \frac{1}{M}\sum_{n=1}^{M}\Phi_n\Phi_n^T$$

$$= AA^T$$

$$A = \{\Phi_1, \Phi_2, \Phi_3, \ldots\ldots, \Phi_n\}$$

6. A^T

$$L_{mn} = \Phi_m^T \Phi_n$$

7. Once we have found the eigenvectors, $\mathbf{v_l}$, $\mathbf{u_l}$

$$u_l = \sum_{k=1}^{M} v_{lk}\Phi_k \qquad l = 1, \ldots\ldots, M$$

Fig.8. Eigenfaces of the corresponding training images set

VI. RECOGNITION PROCEDURE

The recognition procedure involves the following steps which are as follows,

1. A new face is transformed into its Eigenface components. We compare our input image with our mean image and multiply their difference with each eigenvector of the L matrix. Each value would represent a weight and would be saved on a vector Ω.

$$\omega_k = u_k^T\left(\Gamma - \Psi\right)$$

$$\Omega^T = \left[\omega_1, \omega_2, \ldots\ldots, \omega_M\right]$$

2. Now we determine which face class provides the best description for the input image. This is done by minimizing the Euclidean distance

$$\varepsilon_k = \left\|\Omega - \Omega_k\right\|^2$$

3. The input face is considered to belong to a class if $\boldsymbol{\varepsilon_k}$ is below an established threshold $\boldsymbol{\theta_\varepsilon}$, i.e. $\boldsymbol{\varepsilon_k} < \boldsymbol{\theta_\varepsilon}$ Then the face image is considered to be a known face.
4. If the difference is above the given threshold, but bellow a second threshold, the image can be determined as a unknown face.
5. If the input image is above these two thresholds, the image is determined NOT to be a face.

VII. IMPLEMENTATION IN MATLAB

The above discussed methods have been implemented in Matlab. The algorithm which we have generated and implemented in face recognition procedure involves following steps which are as follows,

A. Matlab code

```
% code to open the webcam of laptop and to capture image
Clear all
Close all
clc
button=input('would you like to open the camera?(y \ n)!','s');
if strcmp(button,'y')
disp(button)
vid=videoinput('winvideo',1,'YUY2_320x240');
% code to convert image to RGB format
set(vid,'ReturnedColorSpace','rgb');
preview(vid)
start(vid);
im=getdata(vid,1);
figure,imshow(im)
imwrite(im,'C:\Users\kandla\Desktop\Project\1.jpg');
%  number of images on your training set.
M=3;
%Chosen std and mean.
um=100;
ustd=80;
%read and show images (jpg);
S=[];
figure(1);
for i=1:M
str=strcat('C:\Users\kandla\Desktop\Project\',int2str(i),'.jpg');
   eval('img=imread(str);');
   subplot(ceil(sqrt(M)),ceil(sqrt(M)),i)
   imshow(img)
   if i==3
      title('Training set','fontsize',18)
   end
   drawnow;
   [irow icol]=size(img);
   img(irow,icol) ;
   temp=reshape(img,irow*icol,1);
   S=[S temp];
End
%Here we change the mean and std of all images. We normalize all images.
for i=1:size(S,2)
   temp=double(S(:,i));
   m=mean(temp);
   st=std(temp);
   S(:,i)=(temp-m)*ustd/st+um;
End
%show normalized images
figure(2);
for i=1:M
str=strcat('C:\Users\kandla\Desktop\Project\photo\',int2str(i)
,'.jpg');
   img=reshape(S(:,i),icol,irow);
   img=img';
   eval('imwrite(img,str)');
```

```
subplot(ceil(sqrt(M)),ceil(sqrt(M)),i)
   imshow(img)
   drawnow;
   if i==3
      title('Normalized Training Set','fontsize',18)
   end
end
%mean image;
m=mean(S,2);
 %obtains the mean of each row instead of each column
tmimg=uint8(m);
img=reshape(tmimg,icol,irow);
img=img';
figure(3);
imshow(img);
title('Mean Image','fontsize',18)
 % Change image for manipulation
dbx=[];
for i=1:M
   temp=double(S(:,i));
   dbx=[dbx temp];
end
%Covariance matrix C=A'A, L=AA'
 A=dbx';
L=A*A';
% vv are the eigenvector for L
% dd are the eigenvalue for both L=dbx'*dbx and
C=dbx*dbx';
[vv dd]=eig(L);
% Sort and eliminate those whose eigenvalue is zero
 v=[];
d=[];
for i=1:size(vv,2)
   if(dd(i,i)>1e-4)
      v=[v vv(:,i)];
      d=[d dd(i,i)];
   end
end
 %sort,  will return an ascending sequence
 [B index]=sort(d);
ind=zeros(size(index));
dtemp=zeros(size(index));
vtemp=zeros(size(v));
len=length(index);
for i=1:len
   dtemp(i)=B(len+1-i);
   ind(i)=len+1-index(i);
   vtemp(:,ind(i))=v(:,i);
end
d=dtemp;
v=vtemp;
 %Normalization of eigenvectors
 for i=1:size(v,2)
   kk=v(:,i);
   temp=sqrt(sum(kk.^2));
   v(:,i)=v(:,i)./temp;
end
%Eigenvectors of C matrix
 u=[];
for i=1:size(v,2)
   temp=sqrt(d(i));
   u=[u (dbx*v(:,i))./temp];
end
%Normalization of eigenvectors
for i=1:size(u,2)
  kk=u(:,i);
  temp=sqrt(sum(kk.^2));
  u(:,i)=u(:,i)./temp;
end
 % show eigenfaces;
 figure(4);
for i=1:size(u,2)
   img=reshape(u(:,i),icol,irow);
   img=img';
   img=histeq(img,255);
   subplot(ceil(sqrt(M)),ceil(sqrt(M)),i)
   imshow(img)
   drawnow;
   if i==3
      title('Eigenfaces','fontsize',18)
   end
end
% Find the weight of each face in the training set.
 omega = [];
for h=1:size(dbx,2)
   WW=[];
   for i=1:size(u,2)
      t = u(:,i)';
      WeightOfImage = dot(t,dbx(:,h)');
      WW = [WW; WeightOfImage];
   end
   omega = [omega WW];
end
% Note: the input image must have a bmp or jpg extension.
% code to read input image that have been captured by the
% webcam as discussed above
  InputImage=
imread(strcat('C:\Users\kandla\Desktop\Project\photo\1.jpg')
);
figure(5)
subplot(1,2,1)
imshow(InputImage);
colormap('gray');title('Input image','fontsize',18)
InImage=reshape(double(InputImage)',irow*icol,1);
temp=InImage;
me=mean(temp);
st=std(temp);
temp=(temp-me)*ustd/st+um;
NormImage = temp;
Difference = temp-m;
NormImage = Difference;
p = [];
aa=size(u,2);
for i = 1:aa
   pare = dot(NormImage,u(:,i));
   p = [p; pare];
end
ReshapedImage = m + u(:,1:aa)*p;
 %m is the mean image, u is the eigenvector
ReshapedImage = reshape(ReshapedImage,icol,irow);
ReshapedImage = ReshapedImage';
%show the reconstructed image. ???????
subplot(1,2,2)
imagesc(ReshapedImage); colormap('gray');
```

```
title('Reconstructed image','fontsize',18)
InImWeight = [];
for i=1:size(u,2)
   t = u(:,i)';
   WeightOfInputImage = dot(t,Difference');
   InImWeight = [InImWeight; WeightOfInputImage];
end
ll = 1:M;
figure(6);
subplot(1,2,1)
stem(ll,InImWeight)
title('Weight of Input Face','fontsize',14)
% Find Euclidean distance ??Euclidean??
e=[];
for i=1:size(omega,2)
   q = omega(:,i);
   DiffWeight = InImWeight-q;
   mag = norm(DiffWeight);
   e = [e mag];
end
kk = 1:size(e,2);
subplot(1,2,2)
stem(kk,e)
title('Eucledian distance of input image','fontsize',14)
MaximumValue=max(e);
MinimumValue=min(e);
avg=mean(e);
display(['average=',num2str(avg)]);
if (avg<80000)
display(' u succeeded');
else
display(' face authentication failed');
end
elseif strcmp(button,'n')
    disp(button)
    disp('gudbye')
 end
```

VIII. RESULT

The results from the above matlab implementation are quite successful. Below mentioned a table showing the success and error rates of face recognition on self created image database in various conditions.

CONDITION	SUCCESS	ERROR
NORMAL	83%	17%
LIGHT VARIATIONS	61%	39%
SIZE VARIATIONS	55%	45%

LIMITATIONS OF THIS APPORACH AND CONCLUSION

As the result shows that the various test conducted in different environment have certain limitations over the size, light and the head orientation. But as the real time application is done this method showed very good classification of faces. Limitation also depends on the quality of webcam used. It should be of high quality and precision so that all of your facial expressions can be easily read out. A noisy image or partially occluded face causes recognition performance to degrade. This recognition system is made for the still images but for the future scope more work can be done on this system.

REFERENCES

1. M.A. Turk and A.P. Pentland. "*Face recognition using Eigenfaces"*. In Proc. of Computer Vision and Pattern Recognition, pages 586-591. IEEE, June 1991b.
2. M.Turk and A. Pentland, "Eigenfaces for Recognition", *Journal of Cognitive Neuroscience*, March 1991.
3. L.I. Smith. "*A tutorial on principal components analysis*"
4. Delac K., Grgic M., Grgic S., "*Independent Comparative Study of PCA, ICA, and LDA on the FERET Data Set*", International Journal of Imaging Systems and Technology, Vol. 15, Issue 5, 2006, pp. 252-260.
5. H. Moon, P.J. Phillips, "*Computational and Performance aspects of PCA-based Face Recognition Algorithms*", Perception, Vol. 30, 2001, pp. 303-321.
6. Matlab tutorials and learning resources http://www.mathworks.in/academia/student_center/tutorials/launchpad.html
7. Matlab image acquisition tool box. http://www.mathworks.in/products/imaq/