**COMP 408/508, Programming Assignment #3**

**Eigenface vs Fisherface**

In this assignment, I implemented eigenfaces and fisherfaces techniques for face recognition and compare results of both technique.

Firstly, I implemented Eigenfaces part of the assignement since fisherfaces uses Eigenfaces in order to reduce dimension.

The supplied data set has 165 different face images of 15 people. There are 11 different images which are taken under different configurations for each person. During my experiments, I choose one configuration and keep all images taken under this configuration as test images (15 different images). Then I used rest of the dataset as training. As a result, I trained my face recognition system by using 150 different images and test by using 15 different images belongs different people.

1. Face Recognition Using Eigenfaces
2. Methodology:

The problem with image representation is its high dimensionality representation. For instance the images in the supplied dataset has 243x320 pixels which lies in 77760 dimensional space. Even though images lies in high dimensions, all of the dimensions is not useful in order to represent the image. In order to remove unnecessary information, Principal Component Analysis (PCA) is offered. The idea behind the PCA is that a high-dimensional dataset is often described by correlated variables and therefore only a few meaningful dimensions account for most of the information. The PCA method finds the directions with the greatest variance in the data, called principal components.

Eigenfaces techniques uses PCA in order to reduce dimensions of face images. Eigenfaces corresponds to eigenvectors of the covariance matrix of a set of images. Every image can be expressed as a linear combinations of eigenfaces. Eigenface coefficients represents images as a low dimensional feature vector.

Firstly, I read images into 15x11 array cell. Each cell contains image intensity values, vector representation of image intensities (1x17760) and class information. After train my system, I added coefficient information to each cell which is reduced feature representation of images.

After reading images, I implemented the steps from lecture notes which are explained below:

b. Training

1. I calculated mean of image intensity vectors (X) and subtracted it from X vector Then I calculated covariance matrix as shown below:



2. I find eigenvalues and eigenvectors of covariance matrix. However size of covariance matrix is 17760x17760 which is not feasible to solve. Hence I implemented the trick of to covariance matrix size of 150x150 because M × N matrix with M > N can only have N − 1 non-zero eigenvalues. So it’s possible to take the eigenvalue decomposition of size NxN instead:

=

and get the original eigenvectors of with a left multiplication of the data matrix:

( ) = ( ).

3. After finding eigenvectors and eigenvalues, I sorted eigenvectors descending by their eigenvalue and choose k principal components of the eigenvectors corresponding to the k largest eigenvalues. In this assignment, I used k as 14 as explained in assignment.

4. Then I computed the eigenvectors of as follows:

where and are the eigenvectors of and .

5. Since the resulting eigenvectors are orthogonal, to get orthonormal eigenvectors they need to be normalized to unit length. Thus, I normalized eigenvectors.

6. Finally, I projected all training images into PCA subspace by calculating coefficients as shown below:

y =

Now, we can represent each image by 14 dimensional space which are coefficients of their projections to newly PCA subspace.

c. Testing

1. I projected all the testing images into PCA subspace in order to represent them with low dimensional feature vector.

2. Then, I found the nearest neighbor between the projected training images and the testing images. I used Mahalonobis distance in order to improve performance as shown below:



d. Results

Average Recognition accuracy of my eigenfaces system is 0.787879 when M = 14. This result is as expected because eigenface technique is sensitive to rotational variations, lightining and background. Since the images are scaled, eigenface technique is not affected by scale and translation.

3 example of my test image and the best matched image and its classification is shown below:

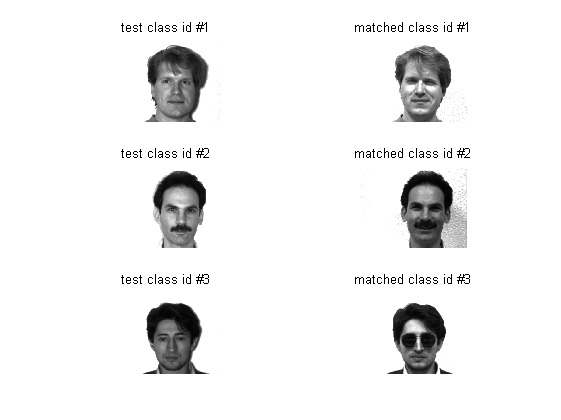


Figure.1 Test image and its best matched image

I showed eigenfaces of one image and its original one below:



Figure.2 Original Image



Figure 3. Eigenfaces of Figure 1.

Different eigeenfaces above shown above seem to accentuate different features of the face. They focuses on different parts of the image because eigenfaces focuses on variatians in the image.

I also reconstructed the image in Figure 1. by using eigenfaces shown in Figure 2. Reconstruction is accomplished by multiplying each coefficient calculated during training part by eigenfaces vectors. The results are shown below:

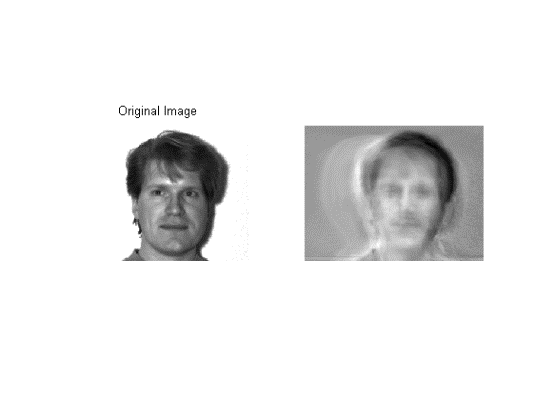


Figure 4. Reconstructed Image (M = 14)

Then I increased the eigen factor size to 100 and 140, then reconstructed image again.

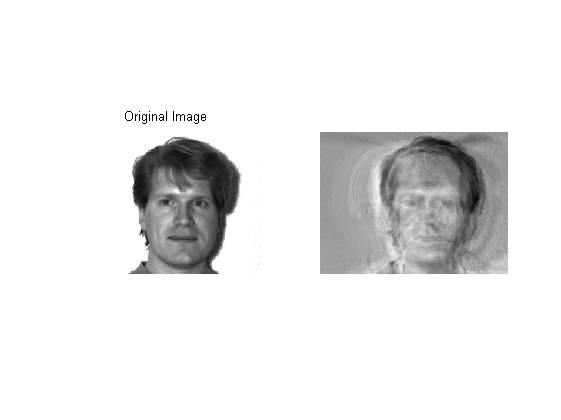
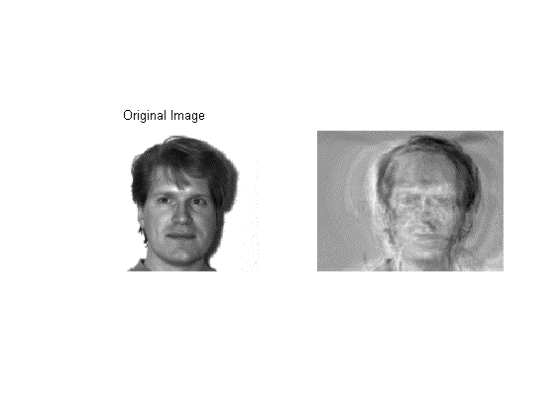
 

Figure 5. Reconstructed Image (M=100) Figure 6. Reconstructed Image (M=140)

When we compare these images, we can easily see that there not much difference between two of them because after some threshold more eigenfaces do not include additional information to reconstructed image. Thus it is rational to reduce dimensionality of feature vector.

1. Face Recognition Using FisherFaces
2. Methodology:

The PCA finds a linear combination of features that maximizes the total variance in data. While this is clearly a powerful way to represent data, it doesn’t consider any classes and so a lot of discriminative information may be lost when throwing components away. Imagine a situation where the variance is generated by an external source, let it be the light. The components identified by a PCA do not necessarily contain any discriminative information at all, so the projected samples are smeared together and a classification becomes impossible.

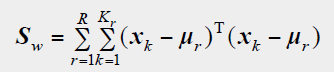
Linear Discriminant Analysis solves this problem. LDA maximizes the ratio of between-classes to within-classes scatter. The idea is simple: same classes should cluster tightly together, while different classes are as far away as possible from each other.

The step of LDA is explained below:

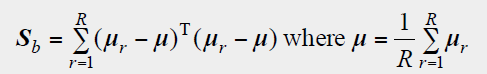
1. Training:

1. Firstly I reduced the dimensions of feature vectors to 135 for inverse operation in LDA by applying eigen faces technique. I used my trainPCA function for this purpose only setting M value to 135.

2. Find mean of feature vector for each class. Then I calculated within-class scatter matrix as shown below:



3. Find mean of classes in order to calculate between-class scatter matrix as shown below:



4. Then I solved eigenvalue and eigenvector problem of matrix.

5. After finding eigenvalue and eigenvector (fisherfaces), I sorted eigenvectors by their eigenvalues and selected 135 principal compnents.

6. Finally, I projected all training images into LDA subspace by calculating coefficients as shown below:

y =

c. Testing

1. I projected all the testing images into LDA subspace in order to represent them with low dimensional feature vector.

2. Then, I found the nearest neighbor between the projected training images and the testing images. I used Mahalonobis distance in order to improve performance as shown below:



d. Results

Average Recognition accuracy of my eigenfaces system is 0.818182 when M = 14. The accuracy rate of fisher face is better than eigenface because it take into consideration of with in class and between class information.

3 example of my test image and the best matched image and its classification is shown below:

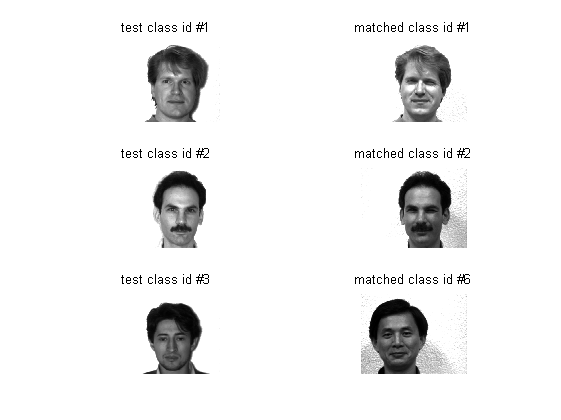


Figure.7 Test image and its best matched image

The Fisherfaces method learns a class-specific transformation matrix, so they do not capture illumination as obviously as the Eigenfaces method. The Discriminant Analysis instead finds the facial features to discriminate between the persons. It’s important to mention, that the performance of the Fisherfaces heavily depends on the input data as well. Practically said: if you learn the Fisherfaces for well-illuminated pictures only and you try to recognize faces in bad-illuminated scenes, then method is likely to find the wrong components (just because those features may not be predominant on bad illuminated images). This is somewhat logical, since the method had no chance to learn the illumination.

I showed fisherfaces of one image and its original one below:



Figure.8 Original Image

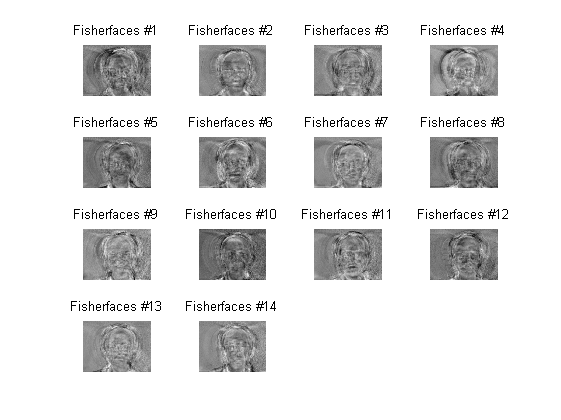


Figure 9. Fisherfaces of Figure 8.