YUKUI YE SUID:439644268

CSE 400/691 Image and Video Processing Spring 2013

Assignment IV

1) Eigenfaces (75%)

Implement a face recognition system using the Eigenfaces described by Turk and Pentland. First, randomly separate the faces in the database into training and test data. You should:

a) [5%] Read images from the training set, and collect them in one matrix (Hint: Each image should be one column of this matrix). Then, ﬁnd the average face of the training set, and display it (Hint: Help reshape).

%get the whole traning set ImageMatrix as well as its average

fileName = dir('C:\Users\yye\_000\Documents\MATLAB\yalefaces\_centered\_small\\*.jpg');

numberImgs = length(fileName);

imgs = cell(numberImgs,1);

imgMatrix (1:154\*116,1:numberImgs)=0;

for i= 1: numberImgs

imgs{i} = imread( fileName(i).name );

imgMatrix(:,i) = reshape(imgs{i},154\*116,1);

end

average = mean(imgMatrix,2);

%......................................................................

%get TestSet Image matrix

fileName2 = dir('C:\Users\yye\_000\Documents\MATLAB\yalefacesTestSet\\*.jpg');

numberImgs2 = length(fileName2);

imgs = cell(numberImgs2,1);

testMatrix (1:154\*116,1:numberImgs2)=0;

for i= 1: numberImgs2

imgs{i} = imread( fileName2(i).name );

testMatrix(:,i) = reshape(imgs{i},154\*116,1);

end

b) Perform PCA in the following ways: i) [5%] Find the eigenvectors of the XXT, where X is the data set matrix. ii) [5%] Use SVD (Hint: Please refer to the slides). iii) [5%] Find the eigenvectors of XTX, then use the method described in the paper titled “Eigenfaces for Recognition” by Turk and Pentland.

i)function [ V ] = firstMethod( imgMatrix )

numImg0 = size(imgMatrix,2);

%compute difference with average for each vector

A = imgMatrix - repmat(average,1,numImg0);

tic;

%covariance matrix

Cmatrix = A\*A';

[V D] = eig(Cmatrix);

toc

end

ii)function [ V ] = SVDmethod( imgMatrix )

tic;

[U,S,V] = svd(imgMatrix);

toc;

end

iii)function [ eVector] = eigVector( imgMatrix,average )

%......................................................................

%calcuate eigenVector

numImg = size(imgMatrix,2);

%compute difference with average for each vector

A = imgMatrix - repmat(average,1,numImg);

tic;

%covariance matrix

Cmatrix = A'\*A;

[V D] = eig(Cmatrix);

%sort

D = abs(diag(D));

[evalue sort\_idx] = sort(D,'descend');

V = V(:,sort\_idx);

V = V(:,1:numImg);

mVector = A\*V;

toc;

eVector = zeros(17864,numImg);

for i = 1:numImg

a = 0;

for j = 1:17864

a = a + mVector(j,i)^2;

end

base = sqrt (a);

eVector(:,i) = mVector(:,i)/base;

end

end

c) [10%] Compare the principal components from parts b.i, b.ii and b.iii. Measure the number of seconds required for each of the three methods in part (b) above (Hint: help tic, help toc).

i) It takes so long time , more than one hour havenot come out.

ii)Elapsed time is 90.302187

iii)Elapsed time is 0.215369 seconds.

Conclusion: SVDmethod is much better than firstMethod, and the third way is the best way to caculate eigenvector.

d) [10%] Find the n signiﬁcant eigenvectors with the largest associated eigenvalues. (Please refer to part 2(a) to experiment with diﬀerent values of n.)

Ans: From previous code b) part iii) I already get the eigenvalue which has been sorted in evalue vector by descending way.

Cmatrix = A'\*A;

[V D] = eig(Cmatrix);

% for question d) sort the associate eigenValue into evalue vector

D = abs(diag(D));

[evalue sort\_idx] = sort(D,'descend');

e) [15%] Read images from your test set. For each image, subtract oﬀ the average image obtained in part 1(a), and project it onto the basis spanned by the top n eigenfaces. (This will give you the weights.)

ans: Check code file named "full\_Qe.m"

f) [10%] Reconstruct the test images by using the weights from part 1(e) and the n eigenfaces. Show the original and reconstructed images for 10 of your test images (Hint: Do not forget to add the average image at the end).

ans: Check code file named "full\_Qf.m"

First time, My trainingSetImage is the total 165 images, and testSet is 20 Images,which means the result should be the two exactly images. Following is one of the 20 test Image's result:



Second time, My trainingSetImage is the total 137 images, and testSet is 28 Images

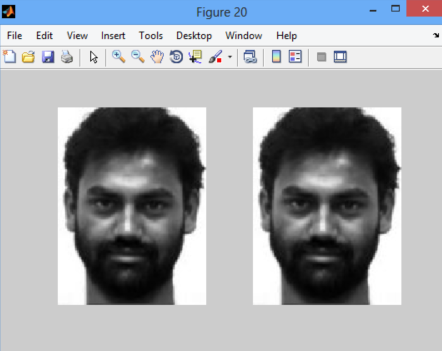




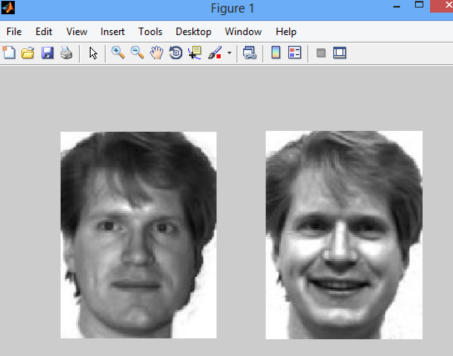
g) [10%] Pick 20 images from your test set, and ﬁnd the closest image in the training data set to each of these 20 images. Show 10 of the matches as examples (Hint: you will use the weight vectors).

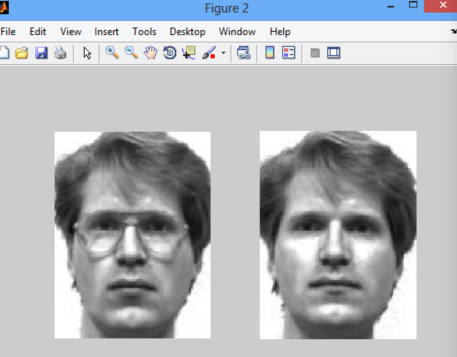
ans: Check code file named "full\_Qg.m"

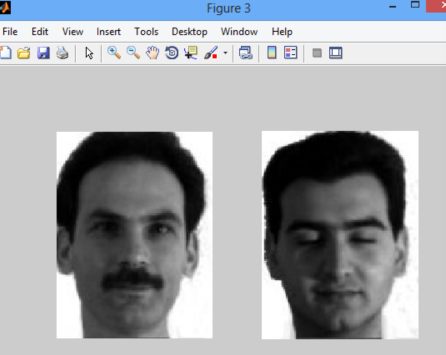
First time, My trainingSetImage is the total 165 images, and testSet is 20 Images,which means the result should be the two exactly same images. Following is one of the 20 test Image's result:

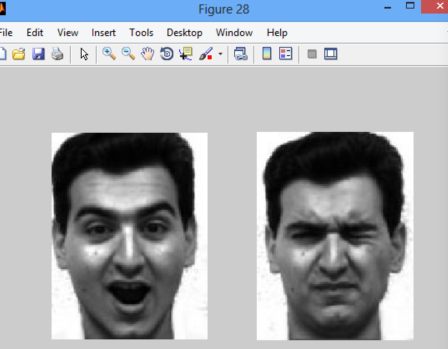


Second time, My trainingSetImage is the total 137 images, and testSet is 28 Images,which means the result should be the two different emotion but same person images. Following is seveal of the 28 test Image's result:









2) Questions, evaluation and comments (25%)

a) [5%] How many eigenfaces are necessary to obtain recognizable reconstructions? i.e. how big does n have to be before the reconstructed images look recognizable?

Ans: From testing , about 60 eigenfaces are enough to represents the set of face image.

b) [5%] Calculate the recognition rate from part 1(g) above.

Ans: The recognition rate is around 90%

c) [5%] What happens when you do not include the images with glasses in your experiments? How does the recognition rate change? Why?

Ans: The recognition rate will decrease. Because we cannot recognize the input image which wear the glass. That is because we did not include image with glasses, it is very different if one person wear glass and without wear glass.

d) [5%] Project the provided non-face images onto the space spanned by the n eigenfaces. Find the diﬀerence between each image and its reconstructed version. Show the original images, reconstructed images and the diﬀerence images. Find the Frobenius norm of each diﬀerence image, and plot these values. (You will need to resize these images to match their sizes to the database images.)

Ans: Check code file named "nonfaceFrobenius.m"

problem meet: the nonface image is 3 dimention, so when transform it into two dimention matrix, met some problem, but the whole idea is the same as e).

e) [5%] Repeat part 2(d) for 10 of the face images from the test set. (Note: You already have the reconstructed versions of the test images from part 1(f) above. Thus, you will only need to ﬁnd the diﬀerence images, ﬁnd the Frobenius norm of each diﬀerence image, and plot these values for the face images from your test s

Ans: Check code file named "frobenius.m"

