Fast Prototyping Exercise 1

Sabiha Barlaskar
Student ID: 917932501

1. Acquired an initial set of 35 images, Tl, T2, T3, . . . ,Tm.
2. Vectorised the images
3. Calculated the average of the images
   psi = sum(T1+T2+….+Tn)/|M|

The average face psi looks as follows:

Figure 1.Average image psi

4. In order to get the unique features of each image, I subtracted the average face vector from each face vector Tau to get a set of vectors phi. This removed the common features from each face.
phi = T - psi

5. Using A = [phi1, phi2, phi3,…,phin], the covariance matrix was calculated as

C = AA’

6. Calculated the eigen vectors of C and sorted the eigen vectors by the eigen values in descending order to find out the k best images with the highest eigen vectors. This gave a set of eigen faces as the result which is as follows:

Figure2. Eigen Faces

7. Projected faces into it's face space. A new face image is transformed into its eigenface components by

yj = top_k_eig_vec' * phi;

8. Projected a set of training faces into it's eigen space vector, where the average is again removed from the set of faces in order to get a set of unique features.

9. Projected a test face into the face space
yz = top_k_eig_vec' * phi_test

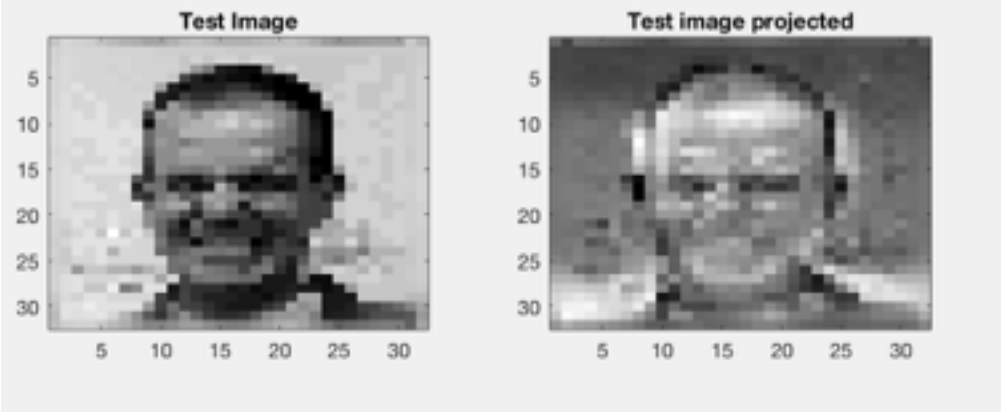

Figure 3 Test image and it's projection onto face space defined by the eigenfaces

10.Calculated the Euclidean distance between the projected test image and the trained image and found out the k best images that match the test image.

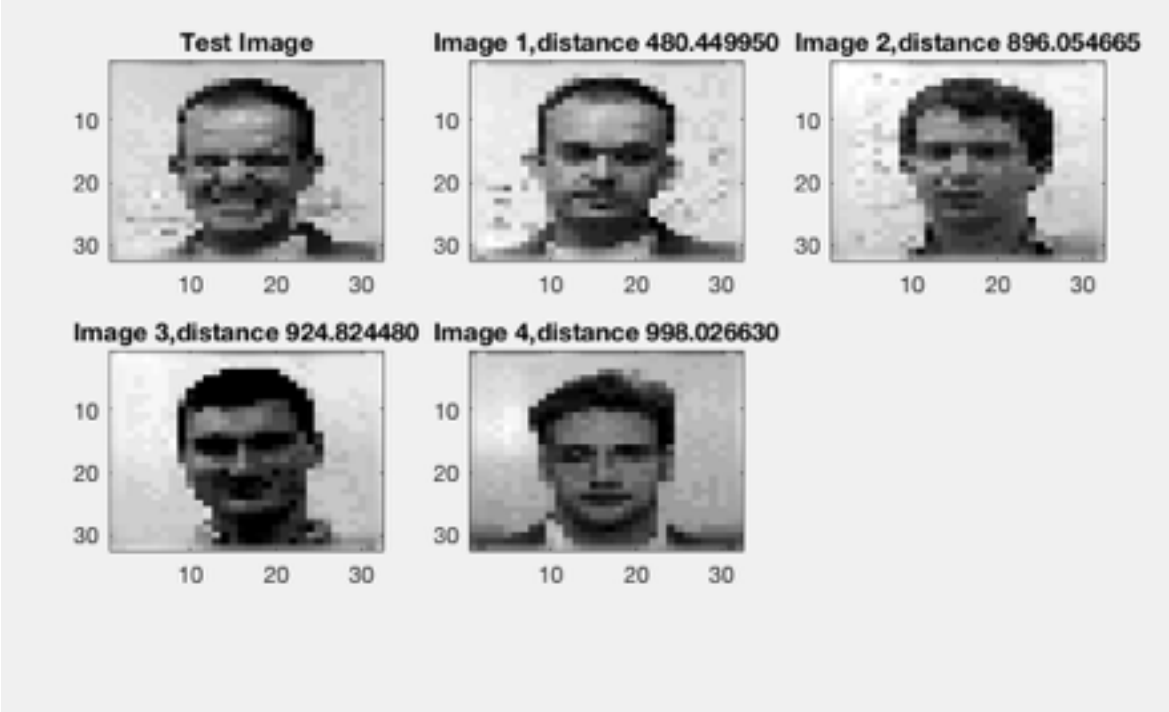

Figure 4.Nearest Neighbor classification yz with yi by picking the index i that best match to yz according to Euclidean distance.

The relative measures of distance from face Space are
(a) 480.44, (b) 896.05,
(c) 924.82 (d) 998.02.

Image in the top left is the test image and the image in the top middle is the trained image taken from the set of images FA that best matches the test image.

The code for the image recognition is given below:

```
%Read the files from the path%
path = dir('/Users/sabihabarlaskar/Documents/MATLAB/FaceRecognition_Data/ALL/
small_*.TIF');

%Initialize the matrix with ones%
tau = ones(1024,35);
for i =1:35
   filename = strcat('/Users/sabihabarlaskar/Documents/MATLAB/
FaceRecognition_Data/ALL/',path(i).name);
   %Read the images %
   I = double(imread(filename));
   %Vectorize the images into a column vector of size N^2 X 1 and store it
   %in a matrix to form an image space
   V_I = I(:);
   tau(:,i) = V_I;
end

%calculating the average face vector
psi = mean(tau,2);
%figure;
%imshow(uint8(reshape(psi,32,32)));
size(psi);

%Subtracting the average face vector from each face vector Tau to get a set of
%vectors phi. The purpose of subtracting the mean image from each image vector is to
%be left with only the unique features
%from each face and removing common features
phi = tau - repmat(psi,1,35);
%A is a matrix that contains the phi of all the images
A = phi;
size(A);
C = A * A';
[eigvec,eigval] = eig(C);
eigval = diag(eigval);
[sortedeigval, eig_indices] = sort(abs(eigval),'descend');
Sorted_eig = eigvec(:,eig_indices);
size(Sorted_eig);
```

```
k = 19;
figure('NumberTitle','off','Name','Eigen faces')
top_k_eig_vec = Sorted_eig(:,1:k);
for i = 1:19
   colormap('gray');
   subplot(5,4,i);
   imagesc(reshape(top_k_eig_vec(:, i), 32, 32));
end

yj = top_k_eig_vec' * phi;
path1 = dir('/Users/sabihabarlaskar/Documents/MATLAB/FaceRecognition_Data/FA/
small_*.TIF');


tau_train = ones(1024,12);
for i =1:12
   filename1 = strcat('/Users/sabihabarlaskar/Documents/MATLAB/
FaceRecognition_Data/FA/',path1(i).name);
   %Read the images %
   I_train = double(imread(filename1));
   %Vectorize the images into a column vector of size N^2 X 1 and store it
   %in a matrix to form an image space
   V_I_train = I_train(:);
   tau_train(:,i) = V_I_train;
end

phi_train = tau_train - repmat(psi,1,12);
yj_train = top_k_eig_vec' * phi_train;


I_test = double(imread('/Users/sabihabarlaskar/Documents/MATLAB/
FaceRecognition_Data/FB/small_00100FB0.tif'));
I_t = imread('/Users/sabihabarlaskar/Documents/MATLAB/FaceRecognition_Data/FB/
small_00100FB0.tif');

%Initialize the matrix with ones%
tau_test = I_test(:);
size(tau_test);
phi_test = tau_test - psi;
yz = top_k_eig_vec' * phi_test;
```

```
yz_u = top_k_eig_vec * yz;
figure;
imagesc(reshape(yz_u,32,32));
colormap('gray');

%Calculate the Euclidean distance
dist = sqrt(sum((yj - yz).^2)');
[sorted_dist,order] = sort(dist);
theta = 900;
figure('NumberTitle','off','Name','ALL images’);
%Displaying the k images with a distance d with the test image
for i=1:10
   subplot(3,3,1)
   imagesc(I_t);
   colormap('gray');
   title('Test Image projected');

   if sorted_dist(i) < theta
      subplot(3,3,i+1)
      imagesc(reshape(tau(:,order(i)),32,32));
      colormap('gray');
      title(sprintf('Image %d,distance %f',i, sorted_dist(i)));
   end
end



dist_train = sqrt(sum((yj_train - yz).^2)');
[sorted_dist_train,order] = sort(dist_train);
theta = 900;
figure('NumberTitle','off','Name','FA images');
for i=1:10
   subplot(3,3,1)
   imagesc(I_t);
   colormap('gray');
   title('Test Image projected');

   if sorted_dist(i) < theta
      subplot(3,3,i+1)
      imagesc(reshape(tau_train(:,order(i)),32,32));
      colormap('gray');
```

```
        title(sprintf('Image %d,distance %f',i, sorted_dist_train(i)));
    end
end
```