



جامعة القادسية

كلية علوم الحاسوب وتكنولوجيا المعلومات

قسم الحاسوب

Face recognition by using eigenface algorithm

بحث مقدم الى مجلس كلية علوم الحاسوب وتكنولوجيا المعلومات كجزء من متطلبات
نيل شهادة البكالوريوس

أعداد

محمد نجاح لفته

زهراء خضر عبيس

رحمه قاسم ياسين

أشراف

م.م. سلام علاوي حسين

شكر و تقدير

"كن عالما .. فإن لم تستطع فكن متعلما ، فإن لم تستطع فأحب العلماء، فإن لم تستطع فلا تبغضهم"
بعد رحلة بحث و جهد و اجتهاد تكلفت بإنجاز هذا البحث ،
نحمد الله عز وجل على نعمه التي من بها علينا فهو
العلي القدير .

كما لا يسعنا إلا أن نخص بأسمى عبارات الشكر و
التقدير الاستاذ "**علي حاكم**" و الدكتور "**علي محسن**"
لما قدماه لنا من جهد و نصح و معرفة طيلة انجاز هذا
البحث .

كما نتقدم بالشكر الجزيل لكل من أسهم في تقديم يد
العون لإنجاز هذا البحث ، كما لا ننسى أن نتقدم بأرقى
و أثنى عبارات الشكر و العرفان الى استاذنا الغالي
"**سلام علاوي**". الذي كان عوننا لنا في بحثنا هذا ونورا
يضيء الظلمة التي كانت تقف أحيانا في طريقنا. إلى
من زرعوا التفاؤل في دربنا وقدموا لنا المساعدات
والتسهيلات والمعلومات ، فلهم منا كل الشكر،

الإهداء

بسم الله الرحمن الرحيم

(قل اعملوا فسيرى الله عملكم ورسوله والمؤمنون)

صدق الله العظيم

إلهي لا يطيب الليل إلا بشكرك ولا يطيب النهار إلى
بطاعتك .. ولا تطيب اللحظات إلا بذكرك .. ولا تطيب
الآخرة إلا بعفوك .. ولا تطيب الجنة إلا برويتك "الله جل
جلاله"

إلى من بلغ الرسالة وأدى الأمانة .. ونصح الأمة .. إلى
نبي الرحمة ونور العالمين "سيدنا محمد صلى الله عليه
وسلم"

إلى من كلفه الله بالهبة والوقار .. إلى من علمني
العطاء بدون انتظار .. إلى من أحمل أسمه بكل افتخار ..
أرجو من الله أن يمد في عمرك لترى ثماراً قد حان
قطافها بعد طول انتظار وستبقى كلماتك نجوم أهدي بها
اليوم وفي الغد وإلى الأبد .. **والدي العزيز**

إلى ملاكي في الحياة .. إلى معنى الحب وإلى معنى
الحنان و التفاني .. إلى بسمه الحياة وسر الوجود إلى

من كان دعائها سر ناجحي وحنانها بلسم جراحي إلى
أعلى الحباب.. **أمي الحبيبة**

إلى من رافقتني منذ أن حملنا حقائب صغيرة ومعك
سرت الدرب خطوة بخطوة وما زلن يرافقني حتى
الآن.. إلى شمعة متقدة تنير ظلمة حياتي.. **اخواتي**

إلى رفيق دربي في هذه الحياة ، معك أكون أنا و بدونك
أكون مثل أي شيء ، إلى من أرى التفاؤل بعينه
والسعادة في ضحكته .. في نهاية مشواري أريد أن
أشرك على مواقفك النبيلة إلى من تطلعت لنجاحي
بنظرات الأمل.. **أخي**

إلى الإخوة و الأخوات ، إلى من تحلو بالإخاء وتميزوا
بالوفاء والعطاء إلى ينابيع الصدق الصافي إلى من
معهم سعدت ، وبرفقتهم في دروب الحياة الحلوة
والحزينة سرت إلى من كانوا معي على طريق النجاح
والخير إلى من عرفت كيف أجدهم وعلموني أن لا
أضيعهم .. **أصدقائي**

Contents

Abstract.....	
1.1 Introduction: -.....	8
1.2 General Difficulties	2
1.3 Eigenfaces:	2
1.4 Loading the images:.....	4
1.5 Training:.....	5
1.5.1 Classification:	6
2.1 Principal Components Analysis (PCA):	8
2.2 Linear Discriminant Analysis (LDA):	10
2.3 Dataset Arrays:	13
3.1 For execute the program we have several steps: -	15
3.2 When we need to training the system	22
References:.....	27

Abstract:

Face recognition is the process of identification of a person by their facial images. This technique makes it possible to use the facial image of a person to authenticate him into a secure system. Face is the main part of human being to be distinguished from one another. Face recognition system mainly takes an image as an input and compares this image with a number of images stored in database to identify whether the input image is in that database or not. There are many techniques used for face recognition. In this project, we have discussed two techniques: Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). Both of these techniques are linear. PCA applies linear projection to the original image space to achieve dimensionality reduction. LDA applies linear projection from the image space to a low dimensional space by maximizing between class scatter and minimizing the within class scatter. These methods will be discussed here based on accuracy and percentage of correct recognition.

Chapter one

1.1 Introduction: -

The face recognition problem can be formulated as follows:

Given an input face image and a database of face images of known individuals, how can we verify or determine the identity of the person in the input image.

Why Use the Face for Recognition?

Biometric-based techniques have emerged as the most promising option for recognizing individuals in recent years since, instead of authenticating people and granting them access to physical and virtual domains based on passwords, PINs, smart cards, plastic cards, tokens, keys and so forth, these methods examine an individual's physiological and/or behavioral characteristics in order to determine and/or ascertain his identity [3]. Passwords and PINs are hard to remember and can be stolen or guessed; cards, tokens, keys and the like can be misplaced, forgotten, purloined or duplicated; magnetic cards can become corrupted and unreadable. However, an individual's biological traits cannot be misplaced, forgotten, stolen or forged. Biometric-based technologies include identification based on physiological characteristics (such as face, fingerprints, finger geometry, hand geometry, hand veins, palm, iris, retina, ear and voice) and behavioral traits (such as gait, signature and keystroke dynamics) [3]

1.2 General Difficulties

Face recognition is a specific and hard case of object recognition. The difficulty of this problem stems from the fact that in their most common form (i.e., the frontal view) faces appear to be roughly alike and the differences between them are quite subtle. Consequently, frontal face images form a very dense cluster in image space which makes it virtually impossible for traditional pattern recognition techniques to accurately discriminate among them with a high degree of success

1.3 Eigenfaces:

Eigenfaces is a well-studied method of face recognition based on principal component analysis (PCA), popularized by the seminal work of Turk & Puntland. Although the approach has now largely been superseded, it is still often used as a benchmark to compare the performance of other algorithms against, and serves as a good introduction to subspace-based approaches to face **recognition**[1]. PCA is a method of transforming a number of correlated variables into a smaller number of uncorrelated variables. Similar to how Fourier analysis is used to decompose a signal into a set of additive orthogonal sinusoids of varying frequencies, PCA decomposes a signal (or image) into a set of additive orthogonal basis vectors or *eigenvectors*. The main difference is that, while Fourier analysis uses a fixed set of basis functions, the PCA basis vectors are learnt from the data set via unsupervised training. PCA can be applied to the task of face recognition by converting the pixels of an image into a number of eigenface feature vectors, which can then be compared to measure the similarity of two face images.[2]

Eigenfaces are mostly used to:

- a.** Extract the relevant facial information, which may or may not be directly related to human intuition of face features such as the eyes, nose, and lips. One way to do so is to capture the statistical variation between face images.
- b.** Represent face images efficiently. To reduce the computation and space complexity, each face image can be represented using a small number of dimensions.

Typical structures of face recognition system consist of three major steps, acquisition of face data, extracting face feature and recognition of face. These steps are elaborated as follow:

A. Acquisition of Face Data:

Acquisition and Processing of Face Data is first step in the face recognition system. In this step face images are collected from different sources. The sources may be camera or readily available face image database on the website. The collected face images should have the pose, illumination and expression etc. variation in order to check the performance of the face recognition system under these conditions. Processing of face database require sometimes otherwise causes serious effect on the performance of face recognition systems due changes in the illumination condition, background, lighting conditions, camera distance, and thus the size and orientation of the head. Therefore, input image is normalized and some image transformation methods apply on the input image.

B. Extracting Face Feature:

Feature extraction process can be defined as the process of extracting relevant information from a face image. In feature extraction, a mathematical representation of original image called a biometric template or biometric reference is generated, which is stored in the database and will form the basis (vector) of any recognition task. Later these extracted features used in recognition. A greyscale pixel is considered as initial feature.

C. Recognition of Face:

Once the features are extracted and selected, the next step is to classify the image. Appearance-based face recognition algorithms use a wide variety of classification methods Such as PCA, LDA. In classification the similarity between faces from the same individual and different individuals after all the face images in database are represented with relevant features. Sometimes feature extraction & recognition process done simultaneously.[6]

Advantages of PCA:

1. Smaller representation of database because we only store the training images in the form of their projections on the reduced basis.
2. Noise is reduced because we choose the maximum variation basis and hence features like background with small variation are automatically ignored.

1.4 Loading the images:

The first step is to load the training images. You can obtain faces from a variety of publicly available face databases, The main requirements are that the faces images must be:

- Greyscale images with a consistent resolution. If using colour images, convert them to greyscale first with `rgb2gray`.
- Cropped to only show the face. If the images include background, the face recognition will not work properly, as the background will be incorporated into the classifier.
- Aligned based on facial features. Because PCA is translation variant, the faces must be frontal and well aligned on facial features such as the eyes, nose and mouth. Most face databases have ground truth available so you

don't need to label these features by hand. The Image Processing Toolbox provides some handy functions for image registration.[5]

Each image is converted into a column vector and then the images are loaded into a matrix of size $n \times m$, where n is the number of pixels in each image and m is the total number of images. The following code reads in all of the PNG images from the directory specified by `input_dir` and scales all of the images to the size specified by `image_dims`:

1.5 Training:

Training the face detector requires the following steps (**the steps to perform PCA**):

1. Calculate the mean of the input face images.
2. Subtract the mean from the input images to obtain the mean-shifted images.
3. Calculate the eigenvectors and eigenvalues of the mean-shifted images.
4. Order the eigenvectors by their corresponding eigenvalues, in decreasing order.
5. Retain only the eigenvectors with the largest eigenvalues (the *principal components*).
6. Project the mean-shifted images into the eigenspace using the retained eigenvectors.

The code is shown below:

```
% steps 1 and 2: find the mean image and the mean-shifted input images
mean_face = mean(images, 2);
shifted_images = images - repmat(mean_face, 1, num_images);
% steps 3 and 4: calculate the ordered eigenvectors and eigenvalues
[evec, score, evals] = princomp(shifted_images');
% step 5: only retain the top 'num_eigenfaces' eigenvectors (i.e. the principal
components)
```

```
num_eigenfaces = 20;
evecs = evecs(:, 1:num_eigenfaces);
% step 6: project the images into the subspace to generate the feature vectors
features = evecs' * shifted_images;
```

Steps 1 and 2 allow us to obtain zero-mean face images. Calculating the eigenvectors and eigenvalues in steps 3 and 4 can be achieved using the `princomp` function. This function also takes care of mean-shifting the input, so you do not need to perform this manually before calling the function. However, I have still performed the mean-shifting in steps 1 and 2 since it is required for step 6, and the eigenvalues are still calculated as they will be used later to investigate the eigenvectors. The output from step 4 is a matrix of eigenvectors. Since the `princomp` function already sorts the eigenvectors by their eigenvalues, step 5 is accomplished simply by truncating the number of columns in the eigenvector matrix. Here we will truncate it to 20 principal components, which is set by the variable `num_eigenfaces`; this number was selected somewhat arbitrarily, but I will show you later how you can perform some analysis to make a more educated choice for this value. Step 6 is achieved by projecting the mean-shifted input images into the subspace defined by our truncated set of eigenvectors. For each input image, this projection will generate a feature vector of `num_eigenfaces` elements.

1.5.1 Classification:

Once the face images have been projected into the Eigen space, the similarity between any pair of face images can be calculated by finding the Euclidean distance between their corresponding feature vectors and; the smaller the distance between the feature vectors, the more similar the faces. We can define a simple similarity score based on the inverse Euclidean distance:

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).[7]

Given an input image input image with the same dimensions image_dims as your training images, the following code will calculate the similarity score to each training image and display the best match:

Closing remarks:

The Eigenfaces approach is now largely superseded in the face recognition literature. However, it serves as a good introduction to the many other similar subspace-base face recognition algorithms. Usually these algorithms differ in the objective function that is used to select the subspace projection. Some subspace-based techniques that are quite popular include:

- **Independent component analysis (ICA)** – selects subspace projections that maximize the statistical independence of the dimensions
- **Fisher's linear discriminant (FLD)** – selects subspace projections that maximize the ratio of between-class to within-class scatter.
- **Non-negative matrix factorization (NMF)** – selects subspace projections that generate non-negative basis vectors[9]

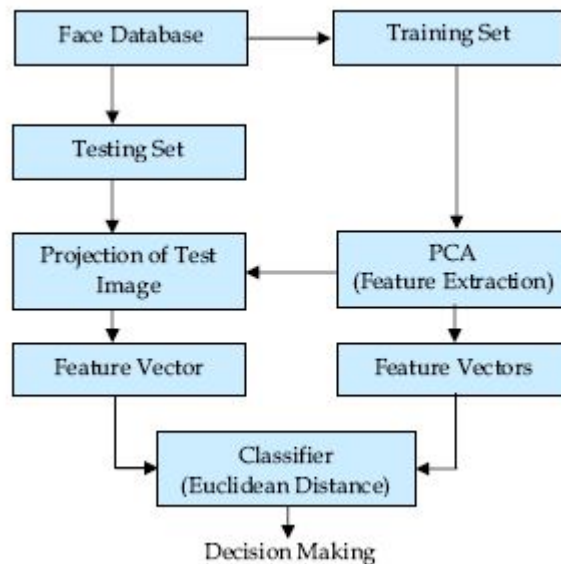
Chapter two

2.1 Principal Components Analysis (PCA):

Real-world data sets usually exhibit relationships among their variables. These relationships are often linear, or at least approximately so, making them amenable to common analysis techniques. One such technique is principal component analysis ("PCA"), which rotates the original data to new coordinates, making the data as "flat" as possible.

Given a table of two or more variables, PCA generates a new table with the same number of variables, called the ***principal components***. Each principal component is a linear transformation of the entire original data set. The coefficients of the principal components are calculated so that the first principal component contains the maximum variance (which we may tentatively think of as the "maximum information"). The second principal component is calculated to have the second most variance, and, importantly, is uncorrelated (in a linear sense) with the first principal component. Further principal components, if there are any, exhibit decreasing variance and are uncorrelated with all other principal components.[1]

PCA is completely reversible (the original data may be recovered exactly from the principal components), making it a versatile tool, useful for data reduction, noise rejection, visualization and data compression among other things. This article walks through the specific mechanics of calculating the principal components of a data set in MATLAB, using either the MATLAB Statistics Toolbox, or just the base MATLAB product.



PCA approach for face Recognition

- For the most part, PCA really is as wonderful as it seems. There are a few caveats, however:

1. PCA doesn't always work well, in terms of compressing the variance. Sometimes variables just aren't related in a way which is easily exploited by PCA. This means that all or nearly all of the principal components will be needed to capture the multivariate variance in the data, making the use of PCA moot.
2. Variance may not be what we want condensed into a few variables. For example, if we are using PCA to reduce data for predictive model construction, then it is not necessarily the case that the first principal components yield a better model than the last principal components (though it often works out more or less that way).
3. PCA is built from components, such as the sample covariance, which are not statistically robust. This means that PCA may be thrown off by outliers and other data pathologies. How seriously this affects the result is specific to the data and application.
4. Though PCA can cram much of the variance in a data set into fewer variables, it still requires all of the variables to generate the principal components of future observations. Note that this is true, regardless of how many principal components are retained for the application. PCA is not a subset selection procedure, and this may have important logistical implications.[4]

2.2 Linear Discriminant Analysis (LDA):

Linear discriminant analysis (LDA) is one of the oldest mechanical classification systems, dating back to statistical pioneer Ronald Fisher, whose original 1936 paper on the subject, *The Use of Multiple Measurements in Taxonomic Problems*.

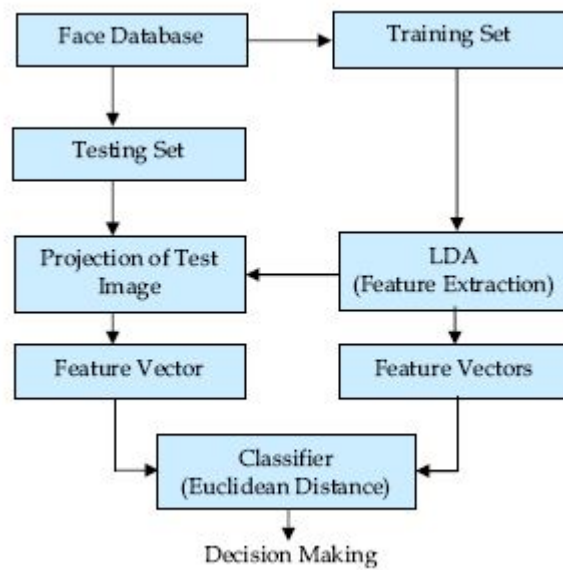
The basic idea of LDA is simple: for each class to be identified, calculate a (different) linear function of the attributes. The class function yielding the highest score represents the predicted class.[8]

There are many linear classification models, and they differ largely in how the coefficients are established. One nice quality of LDA is that, unlike some of the alternatives, it does not require multiple passes over the data for optimization. Also, it naturally handles problems with more than two classes and it can provide probability estimates for each of the candidate classes.

Some analysts attempt to interpret the signs and magnitudes of the coefficients of the linear scores, but this can be tricky, especially when the number of classes is greater than 2.

LDA bears some resemblance to principal components analysis (PCA), in that a number of linear functions are produced (using all raw variables), which are intended, in some sense, to provide data reduction through rearrangement of information. Note, some important differences: First, the objective of LDA is to maximize class discrimination, whereas the objective of PCA is to squeeze variance into as few components as possible.[2]

Second, LDA produces exactly as many linear functions as there are classes, whereas PCA produces, as many linear functions as there are original variables. Last, principal components are always orthogonal to each other ("uncorrelated"), while that is not generally true for LDA's linear scores.



LDA approach for face recognition

There are 5 general steps for performing a linear discriminant analysis.

1. Compute the d -dimensional mean vectors for the different classes from the dataset.
2. Compute the scatter matrices (between-class and within-class scatter matrix). For all samples of all classes the between class scatter matrix S_B and the within-class scatter matrix S_W are defined by:

$$S_B = \sum_{i=1}^c M_i \cdot (x_i - \mu) \cdot (x_i - \mu)^T$$

$$S_W = \sum_{i=1}^c \sum_{x_k \in X_i} (x_k - \mu_i) \cdot (x_k - \mu_i)^T$$

Where M_i is the number of training samples in class i , c is the number of distinct classes, μ_i is the mean vector of samples belonging to class i and X_i represents the set of samples belonging to class i with x_k being the k -th image of that class. S_W represents the scatter of features around the mean of each face class and S_B represents the scatter of features around the overall mean for all face classes.

3. Compute the eigenvectors ($\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_d$) and corresponding eigenvalues ($\lambda_1, \lambda_2, \dots, \lambda_d$) for the scatter matrices. The goal is to maximize SB while minimizing SW, in other words, maximize the ratio $\det[\mathbf{SB}]/\det[\mathbf{SW}]$. This ratio is maximized when the column vectors of the projection matrix ($\mathbf{W}$) are the eigenvectors of $\mathbf{SW}^{-1}\mathbf{SB}$.
4. Sort the eigenvectors by decreasing eigenvalues and choose k eigenvectors with the largest eigenvalues to form a $d \times k$ -dimensional matrix $\mathbf{W}$ (where every column represents an eigenvector).
5. Use this $d \times k$ eigenvector matrix to transform the samples onto the new subspace. This can be summarized by the mathematical equation: $\mathbf{y} = \mathbf{W}^T \times \mathbf{x}$ (where $\mathbf{x}$ is a $d \times 1$ -dimensional vector representing one sample, and $\mathbf{y}$ is the transformed $k \times 1$ -dimensional sample in the new subspace).

Closing thoughts:

Though it employs a simple model structure, LDA has held up reasonably well, sometimes still besting more complex algorithms. When its assumptions are met, the literature records it doing better than logistic regression.

It is very fast to execute and fitted models are extremely portable- even a spreadsheet will support linear models (...or, one supposes, paper and pencil!) LDA is at least worth trying at the beginning of a project, if for no other reason than to establish a lower bound on acceptable performance.

2.3 Dataset Arrays:

Statistics and Machine Learning Toolbox™ has *dataset arrays* for storing variables with heterogeneous data types. For example, you can combine numeric data, logical data, cell arrays of character vectors, and categorical arrays in one dataset array variable.

Within a dataset array, each variable (column) must be one homogeneous data type, but the different variables can be of heterogeneous data types. A dataset array is usually interpreted as a set of variables measured on many units of observation. That is, each row in a dataset array corresponds to an observation, and each column to a variable. In this sense, a dataset array organizes data like a typical spreadsheet.

Dataset arrays are a unique data type, with a corresponding set of valid operations. Even if a dataset array contains only numeric variables, you cannot operate on the dataset array like a numeric variable. The valid operations for dataset arrays are the methods of the dataset class.

Dataset Array Properties:

In addition to storing data in a dataset array, you can store metadata such as:

- Variable and observation names
- Data descriptions
- Units of measurement
- Variable descriptions

This information is stored as dataset array properties. For a dataset array named `ds`, you can view the dataset array metadata by entering `ds.Properties` at the command line. You can access a specific property, such as variable names—property `VarNames`—using `ds.Properties.VarNames`. You can both retrieve and modify property values using this syntax.

Variable and observation names are included in the display of a dataset array. Variable names display across the top row, and observation names, if present, appear in the first column. Note that variable and observation names do not affect the size of a dataset array.

Applications

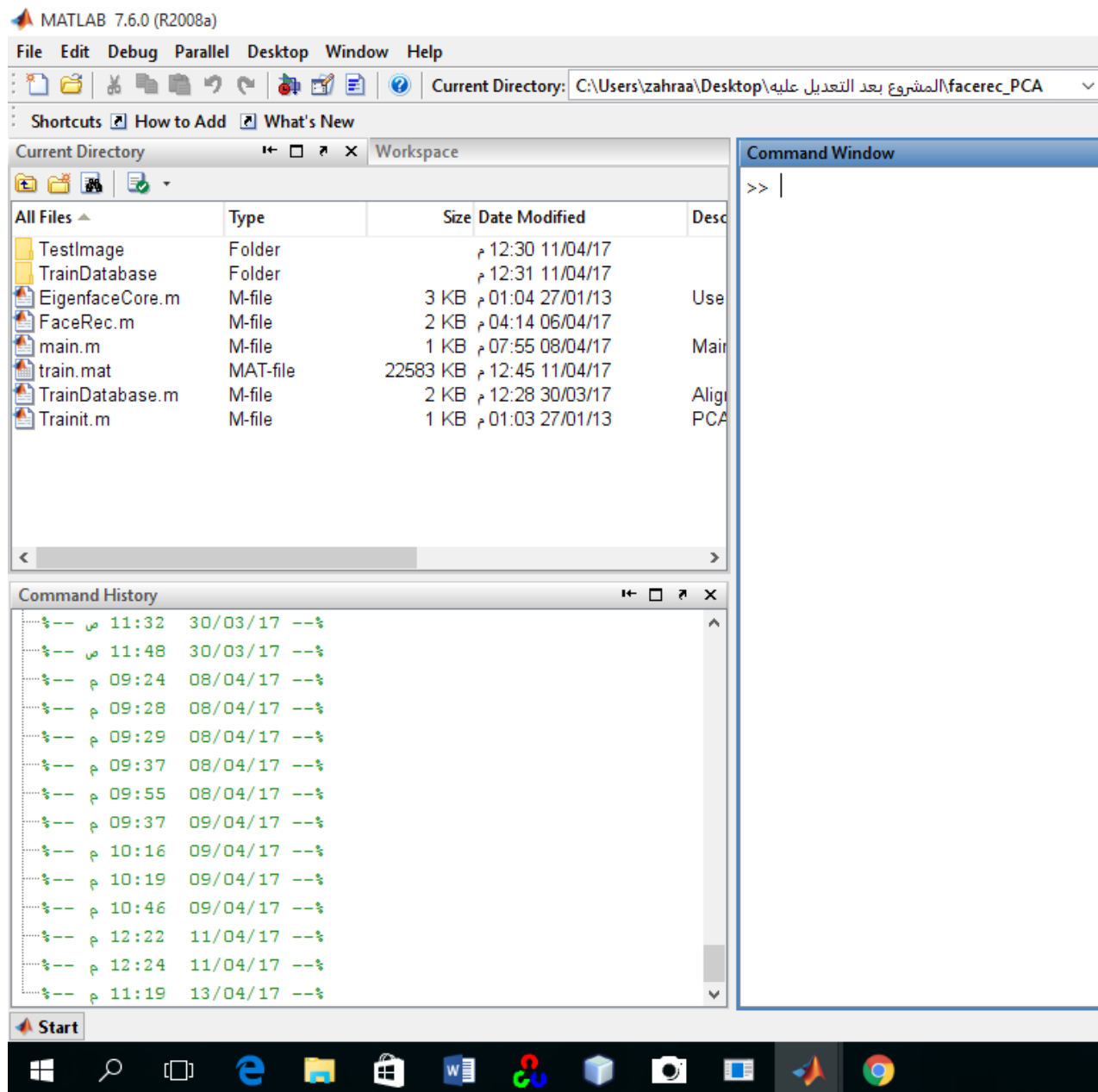
Face recognition is used for two primary tasks:

1. Verification (one-to-one matching): When presented with a face image of an unknown individual along with a claim of identity, ascertaining whether the individual is who he/she claims to be.
2. Identification (one-to-many matching): Given an image of an unknown individual, determining that person's identity by comparing (possibly after encoding) that image with a database of (possibly encoded) images of known individuals. [7, 8].

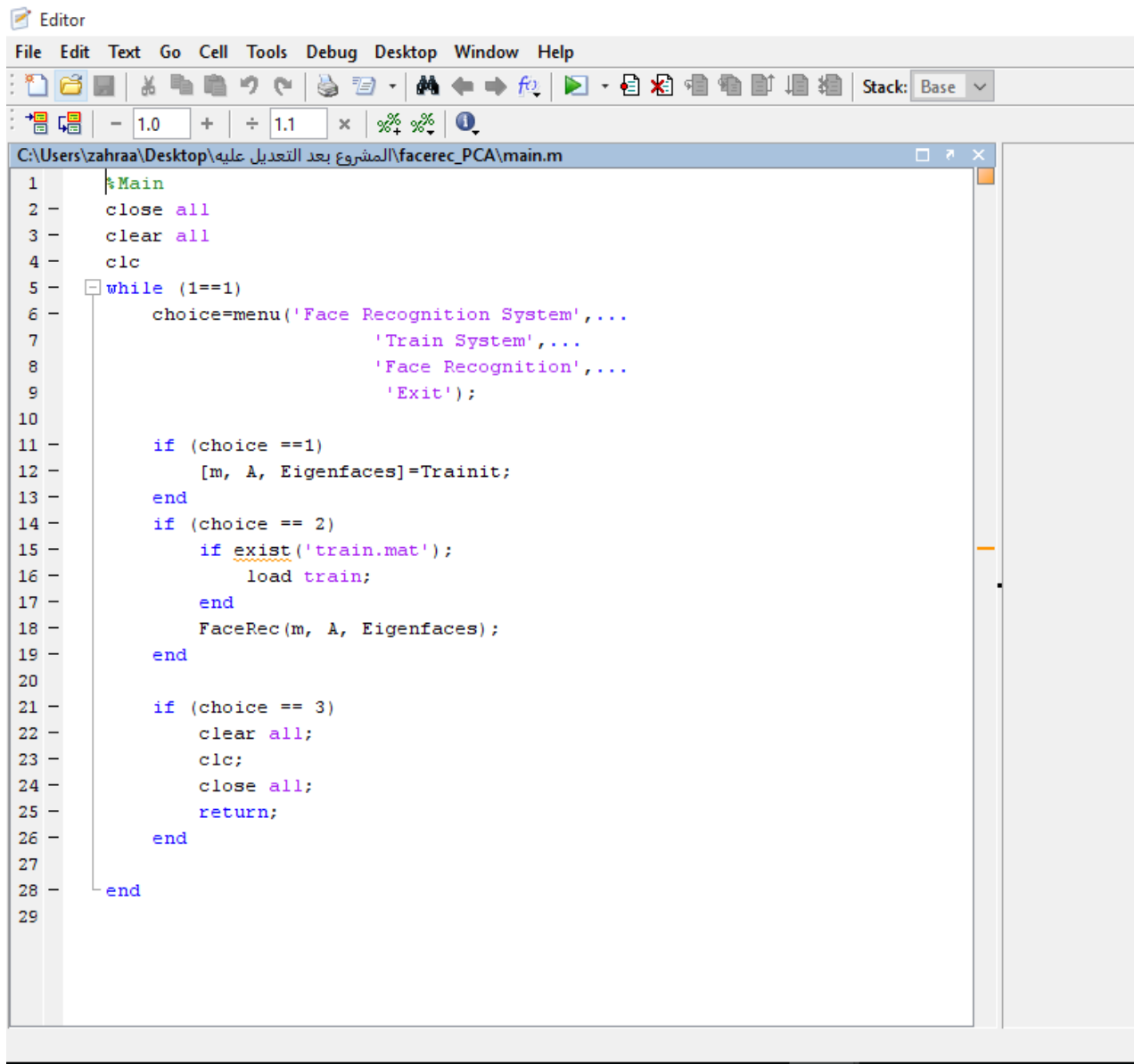
Chapter three

3.1 For execute the program we have several steps: -

- ❖ Open MATLAB version 7.6.0



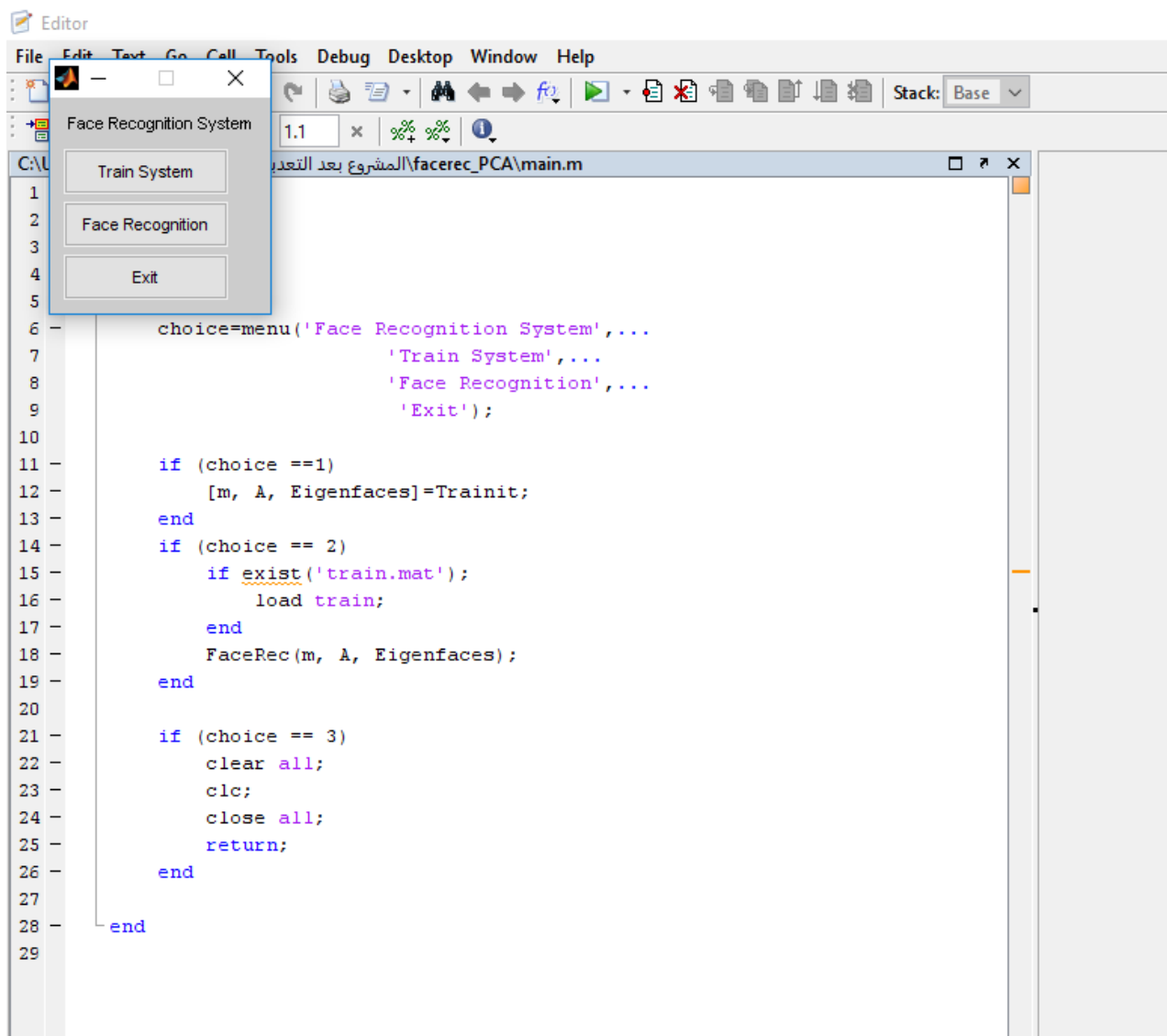
❖ Choose “main” code:



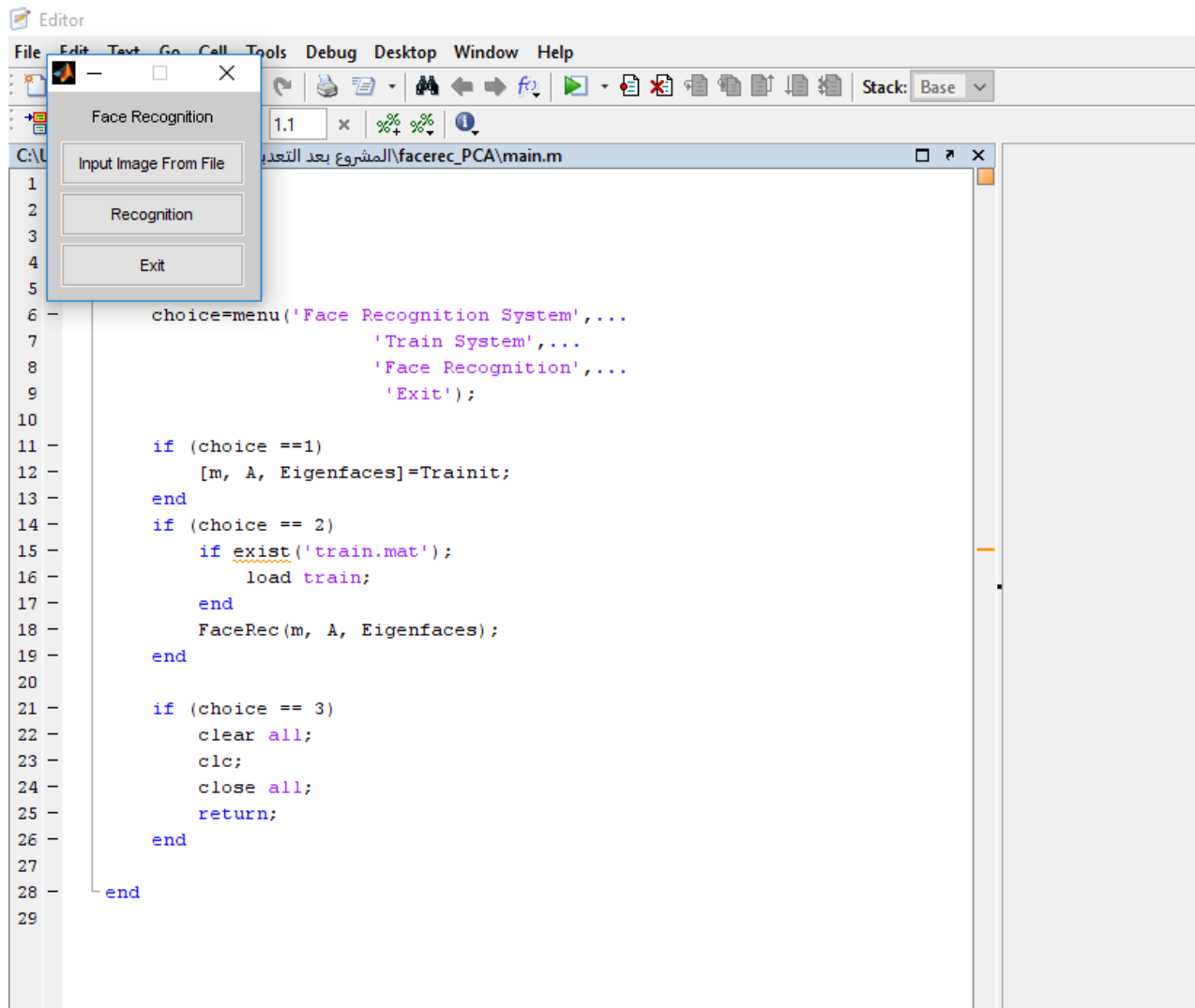
The image shows a MATLAB Editor window with the following code:

```
1 %Main
2 close all
3 clear all
4 clc
5 while (1==1)
6     choice=menu('Face Recognition System',...
7                 'Train System',...
8                 'Face Recognition',...
9                 'Exit');
10
11     if (choice ==1)
12         [m, A, Eigenfaces]=Trainit;
13     end
14     if (choice == 2)
15         if exist('train.mat');
16             load train;
17         end
18         FaceRec(m, A, Eigenfaces);
19     end
20
21     if (choice == 3)
22         clear all;
23         clc;
24         close all;
25         return;
26     end
27
28 end
29
```

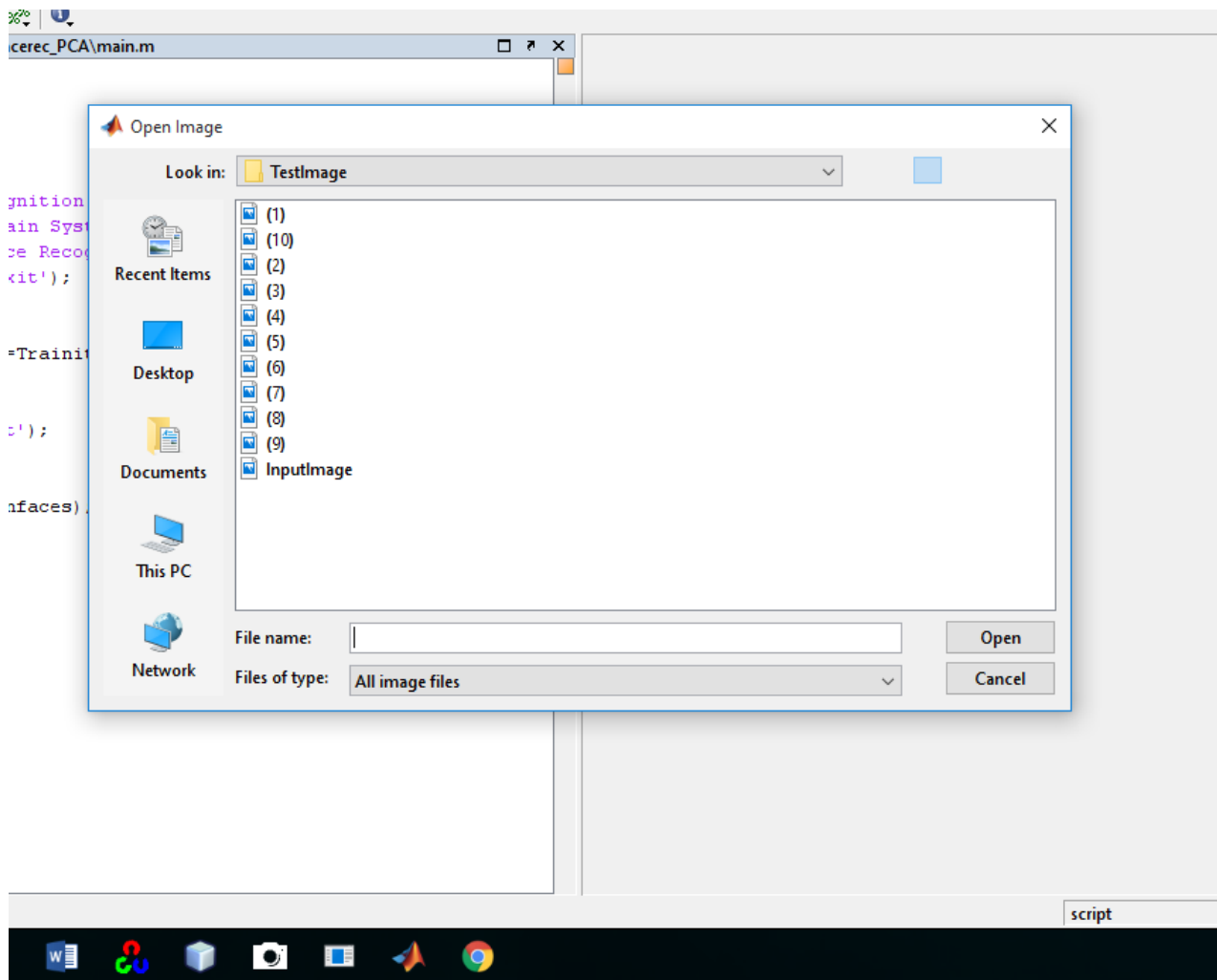
❖ From “Debug” list choose “run” order:



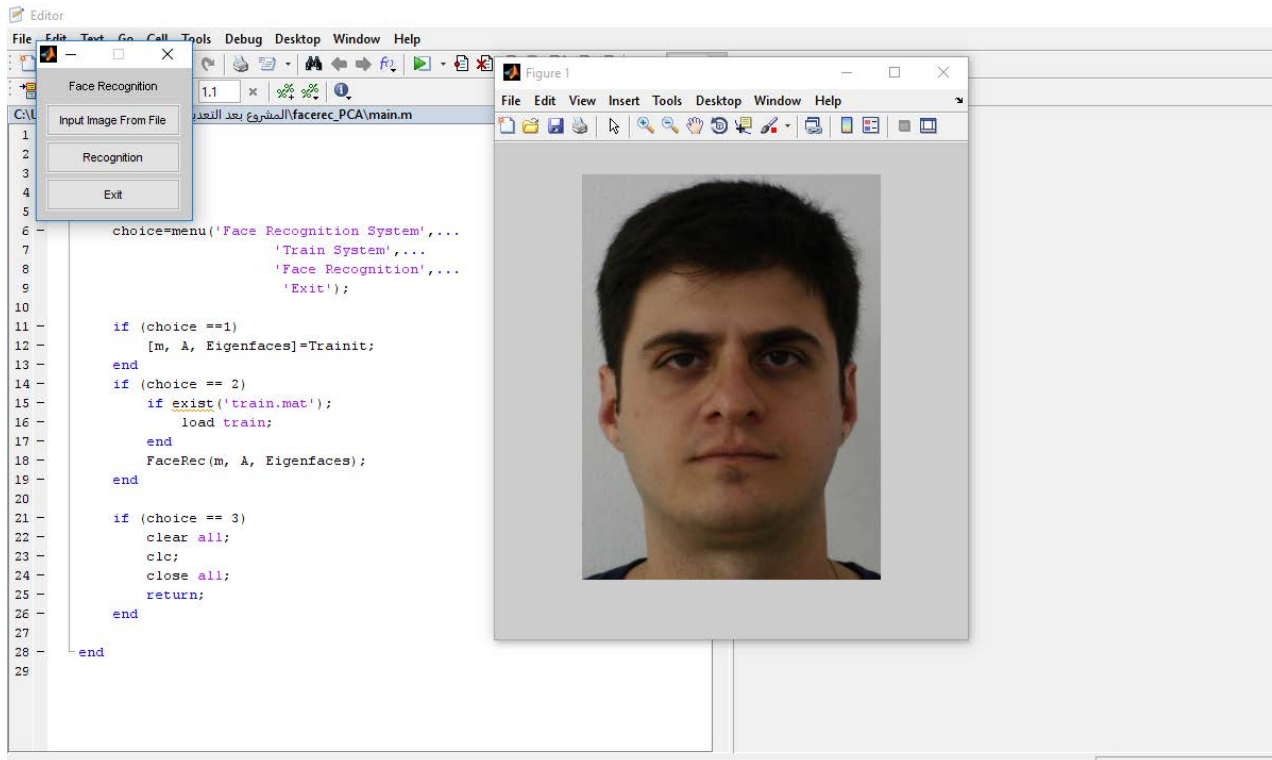
❖ Choose “face recognition” from executable interface:



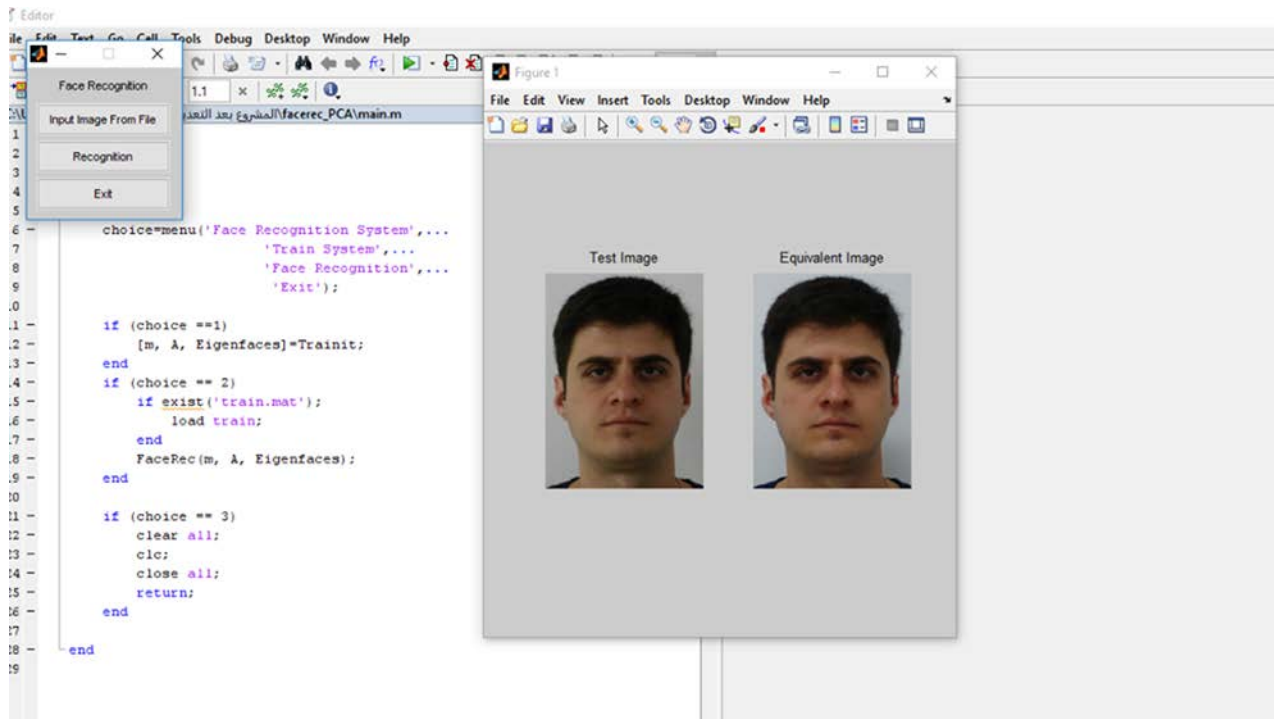
❖ Click on “input image from file” to insert test image:



❖ Choose test image:

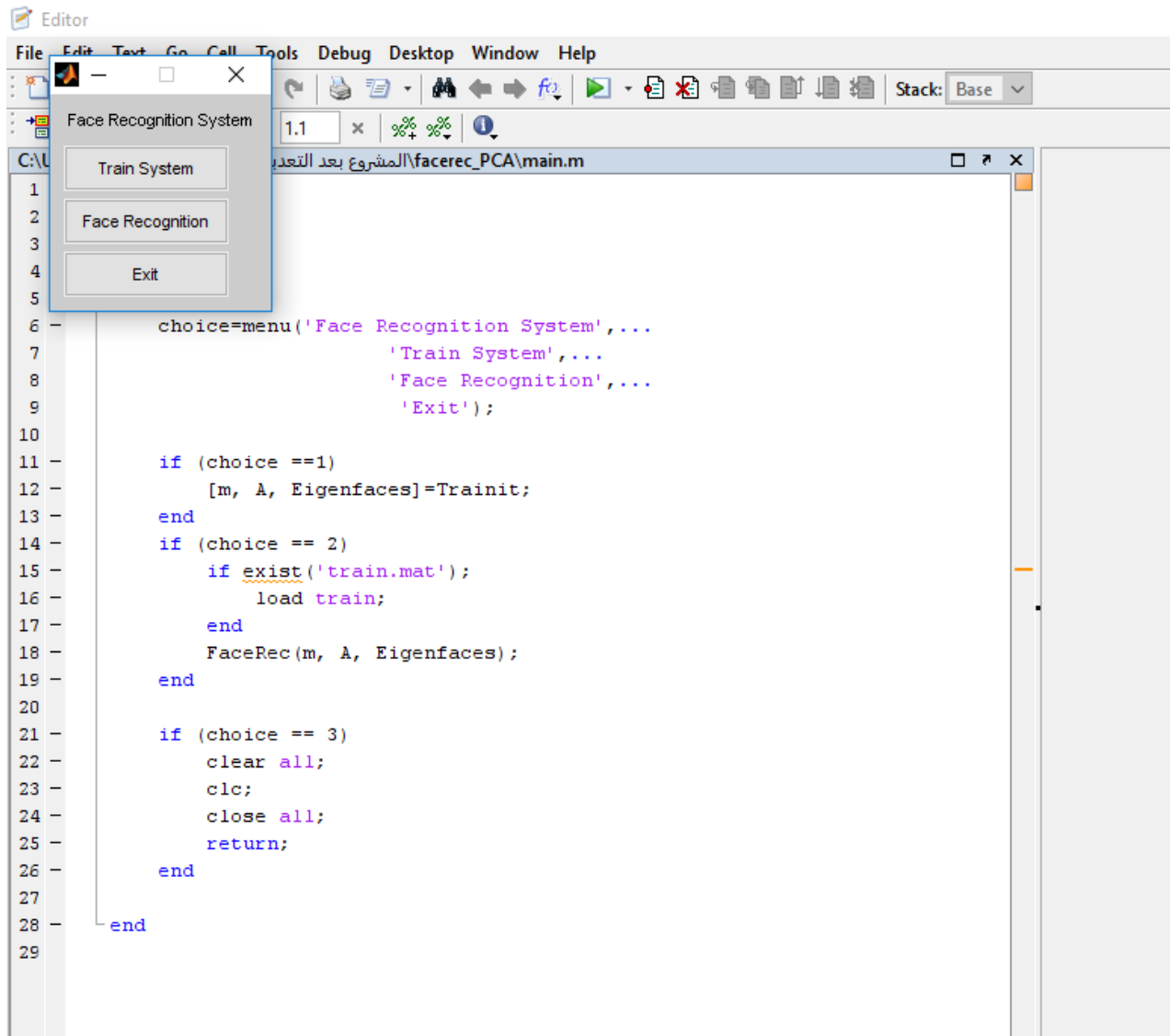


❖ Click on “recognition”:

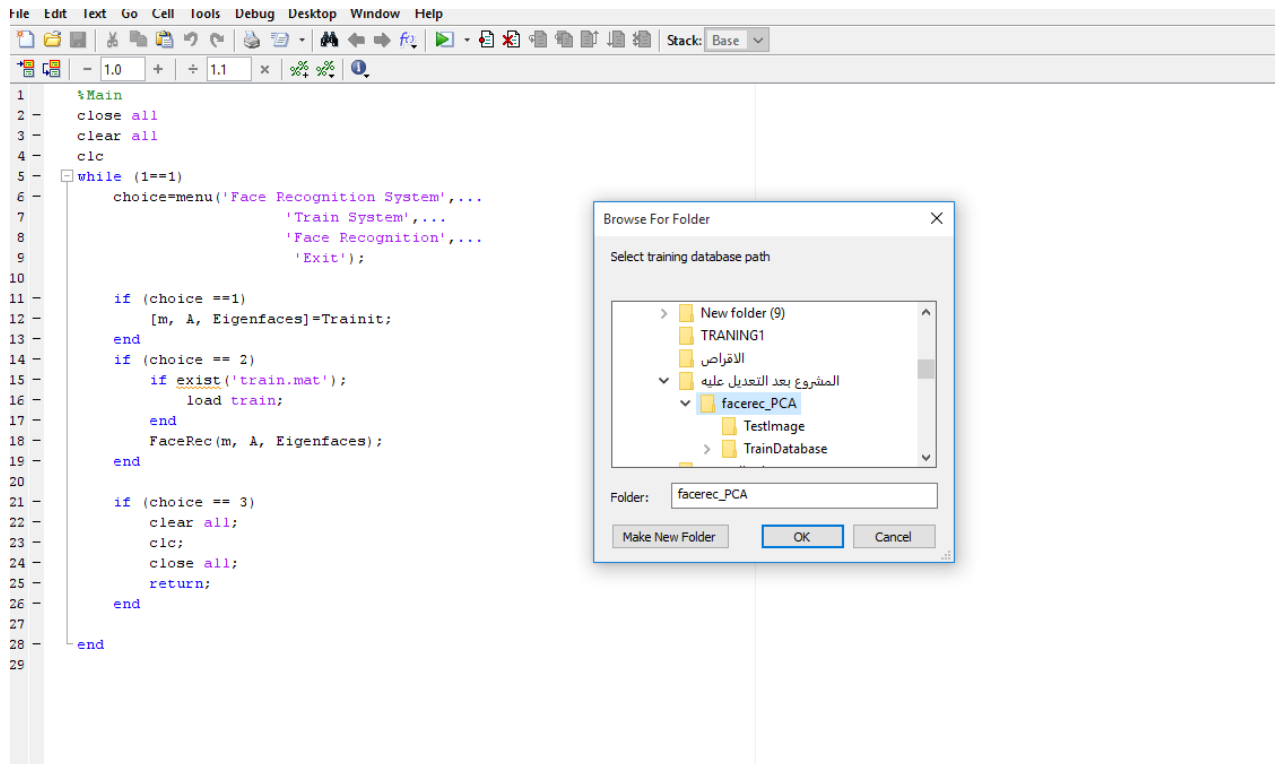


3.2 When we need to training the system

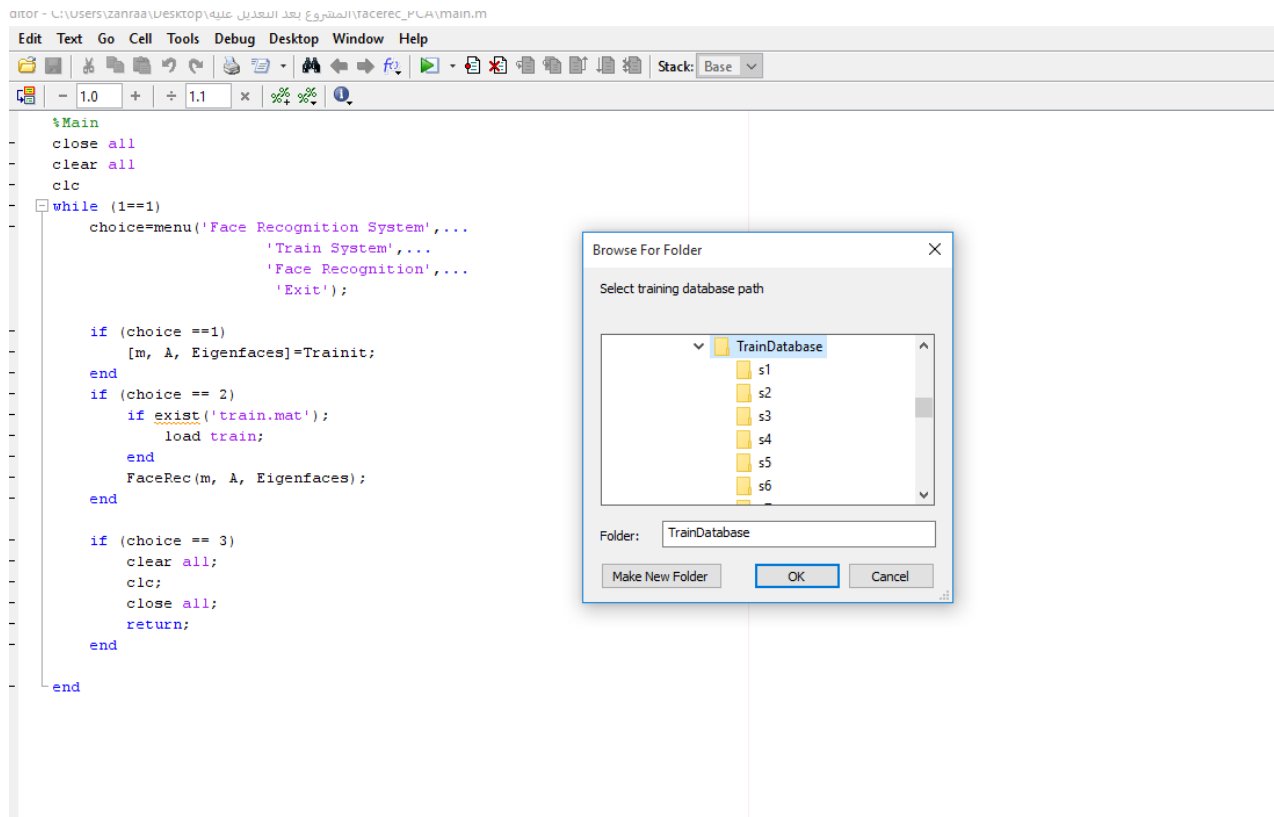
- From “Debug” list choose “run” order:



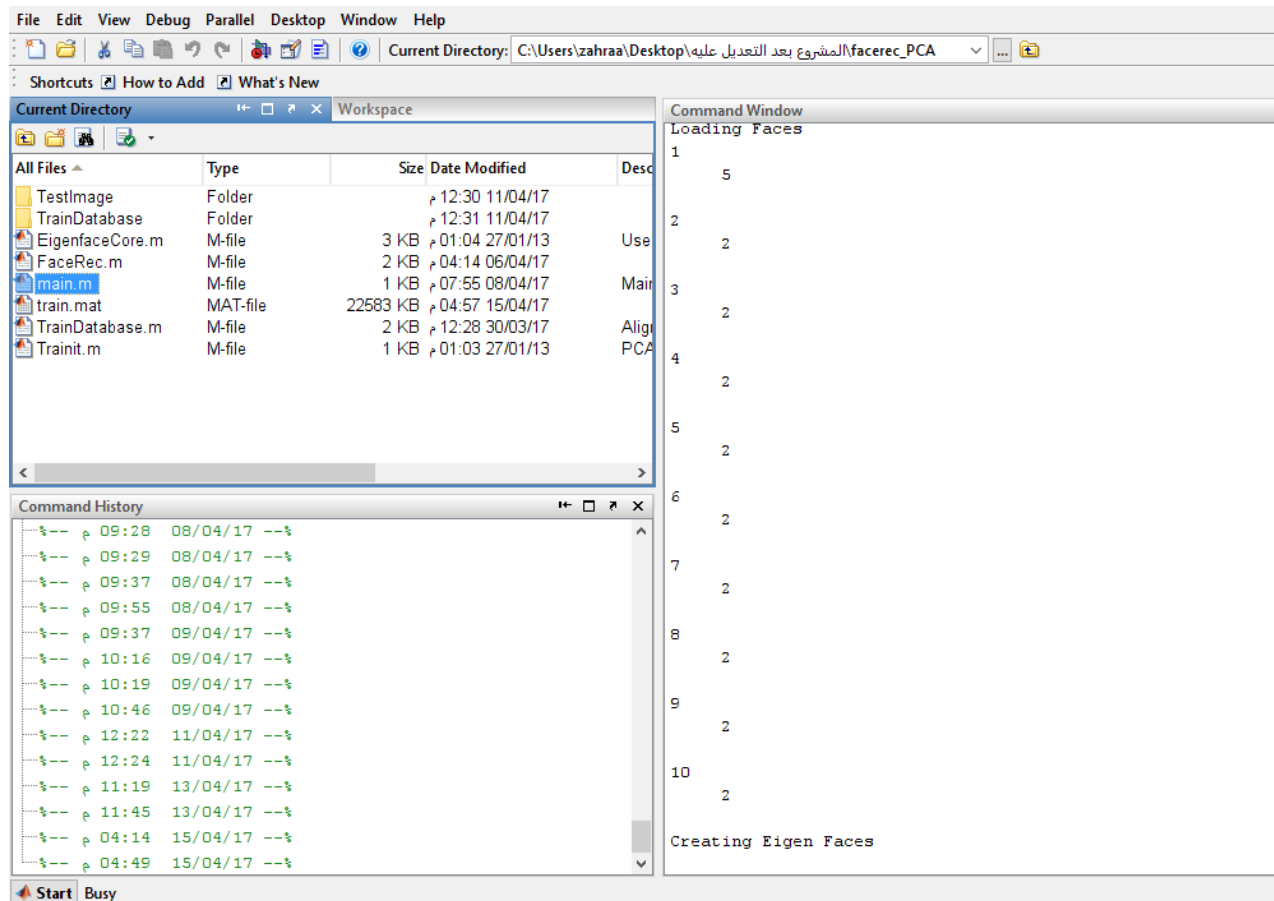
- Choose "train system" from executable interface:



- Click on "train database"



- That is show we the picture on the training system



```

EigenfaceCore.m  X  +
1  function [m, A, Eigenfaces] = EigenfaceCore(T)
2
3  disp('Creating Eigen Faces');
4  %%%%%%%%%%%%%%% Calculating the mean image
5  m = mean(T,2); % Computing the average face image m = (1/P)*sum(Tj's)    (j = 1 : P)
6  Train_Number = size(T,2);
7
8  %%%%%%%%%%%%%%% Calculating the deviation of each image from mean image
9  A = [];
10 for i = 1 : Train_Number
11     temp = double(T(:,i)) - m; % Computing the difference image for each image in the training set Ai = Ti - m
12     A = [A temp]; % Merging all centered images
13 end
14
15 L = A'*A; % L is the surrogate of covariance matrix C=A*A'.
16 [V, D] = eig(L); % Diagonal elements of D are the eigenvalues for both L=A'*A and C=A*A'.
17
18 %%%%%%%%%%%%%%% Sorting and eliminating eigenvalues
19 % All eigenvalues of matrix L are sorted and those who are less than a
20 % specified threshold, are eliminated. So the number of non-zero
21 % eigenvectors may be less than (P-1).
22
23 L_eig_vec = [];
24 for i = 1 : size(V,2)
25     if( D(i,i)>1 )
26         L_eig_vec = [L_eig_vec V(:,i)];
27     end
28 end
29
30 %%%%%%%%%%%%%%% Calculating the eigenvectors of covariance matrix 'C'
31 % Eigenvectors of covariance matrix C (or so-called "Eigenfaces")
32 % can be recovered from L's eigenvectors.
33 Eigenfaces = A * L_eig_vec; % A: centered image vectors
34 save train
35 end

```

In this simplified code Eigenfaces is calculated for training images and this process is done in several steps:

- 1- Calculating the mean image
2. Calculating the deviation of each image from mean image (calculating the picture difference for all images in the training set)
3. Calculate the variance matrix 'L' and calculate the diagonal D elements which are the eigenvalues for both $L = A' * A$ and $C = A * A'$
- 4- Calculation of eigenvectors
- 5 - Calculating the eigenvectors of covariance matrix 'C' Eigenvectors of covariance matrix C (or so-called "Eigenfaces") can be recovered from L's eigenvectors.

References:

- 1- Facial recognition using egnefaces by PCA. (IJRTE), vol.1, no. 1, may 2009.
- 2- B. Karduman, relevant component analysis, M.S thesis yyildiz technical university, turkey,2008.
- 3- F. kahraman, B. Kurt and M. Gokmen face recognition based on active shape model, siu2005, Antalya, turkey,2005.
- 4- LINEAR Discriminant Analysis-A Brief tutorial, vol. 3, p22,2009.
- 5- Eigen faces for face recognition, ece 533 final project report, Min Luo, Department of Biomedical Engineering.
- 6- Summary:" Eigenfaces for Recognition" (M. Turk, A. Pentland), Ed Lawson.
- 7- K. Kim, "Intelligent Immigration Control System by Using Passport Recognition and Face Verification," in International Symposium on Neural Networks. Chongqing, China, 2005, pp.147-156.
- 8- J. N. K. Liu, M. Wang, and B. Feng, "iBotGuard: an Internet-based intelligent robot security system using invariant face recognition against intruder," IEEE Transactions on Systems Man and Cybernetics Part C-Applications and Reviews, Vol.35, pp.97-105, 2005.
- 9- H. Moon, "Biometrics Person Authentication Using Projection-Based Face Recognition System in Verification Scenario," in International Conference on Bioinformatics and its Applications. Hong Kong, China, 2004, pp.207-213.