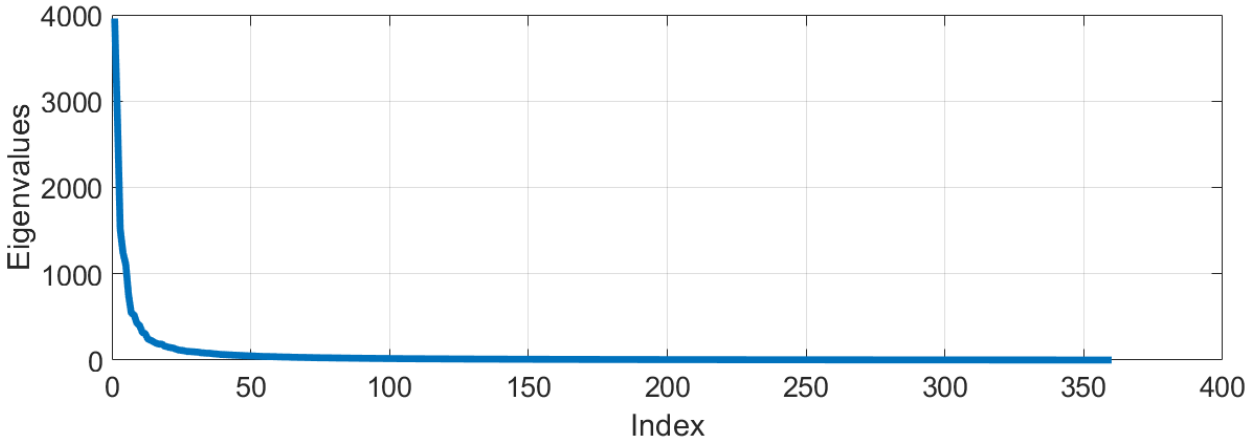
Important Note: Submit all your codes as a matlab script ("lab10.m") to SUCourse **until the end of the lab**.

Face Detection:

In this section of the lab, you will write a program to detect if a given image is a face or not by making use of the eigenfaces algorithm. You will be given an image dataset consisting of 10 images (56×46) taken from different views of 40 people. You can use 9 images per person to build your model and the rest for testing its performance. Note that, in the given dataset the images are already flattened and concatenated therefore it has 2576 (56×46) columns and 400 rows. A sample of the dataset is shown in the figure below:

You will first need to process your image dataset to compute the eigenfaces and then choose only the most effective K of them corresponding to the largest eigenvalues. To choose K properly, you can look at the at the decay of the eigenvalues after plotting them as shown below.

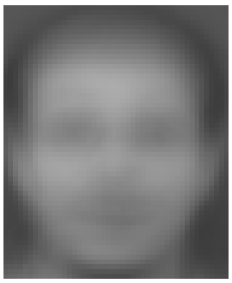


After selecting your K eigenvectors, show the first 12 computed eigenfaces as follows:

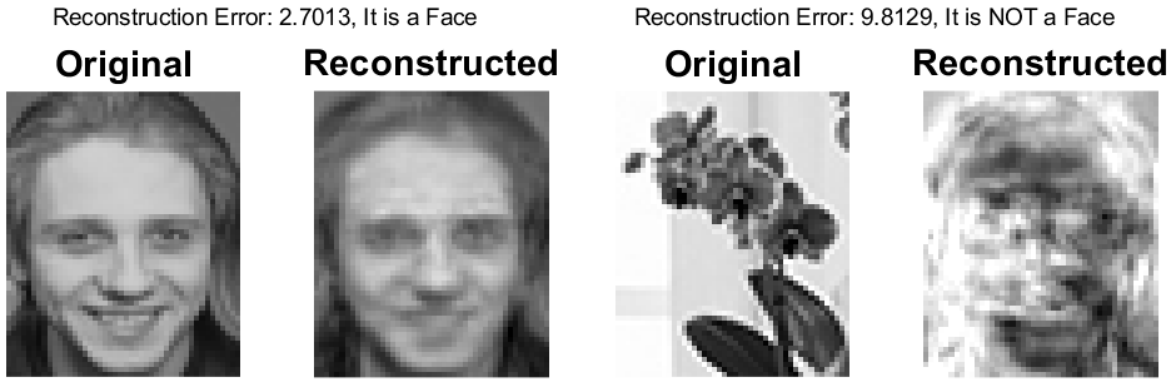
Then given a new image, decide if it is a face or not by making use of the equation below and specifying a proper threshold value.

$$\underbrace{||x - (\bar{x} + a_1 v_1 + a_2 v_2 + \cdots + a_k v_k)||}_{\text{Reconstruction Error}} < \text{Threshold}$$

where, x is a new image in the vector form (row.col $\times$ 1), $\bar{x}$ is the vector form of the mean image, v_i is the computed eigenvectors and $a_i = (x - \bar{x})^T v_i$. Your calculated $\bar{x}$ should look as follows:

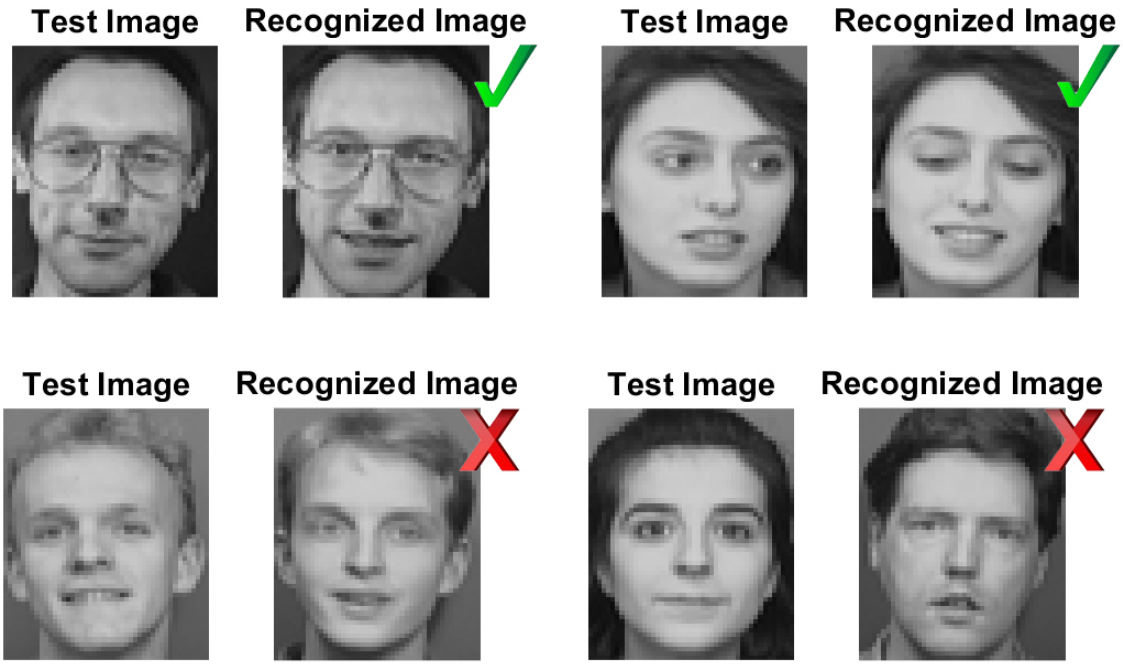


Finally, your resulting images should look as follows:



Face Recognition:

Once you are able to detect faces, build a database of descriptors composed of the calculated a_i for each face image in your training set. Then given a new face, calculate its descriptor and compare it with the ones in your dataset using Euclidean distance metric to recognize it. Your results should look as follows;



Useful codes for this lab:

```
% Reshape a flattened image to show it
Image=uint8(reshape(Flattened_Image, [row,col]));

% Calculate K Eigenvalues and Eigenvectors
[V, D] = eigs(A^TA,K)

% Calculate Vectorwise Norm of a Matrix
N=vecnorm(A)

% Display multiple image frames as rectangular montage
figure; montage(D, 'Size', [r c]);
% where D is a matrix with the size of row×col×1×numOfImages
```

Post Lab

Provide resulting images for each step. Explain the procedure that you follow. Test the algorithm with different images in the provided dataset. Moreover, take some images of your own face at different angles and test them with the algorithm. Additionally use Mahalanobis distance metric instead of Euclidean distance and compare their perfromances. Discuss and comment on your results.

Deadline for post lab report submission to SUCourse: **2 January 2019, 23:55**.