PCA: Using eigenface to perform facial recognition

Name: Liyan Chen

PID:A53221038

Programming Language: Python 2.7

Objective

The objective of this article is to implement the algorithm put forward in M. Turk and A. Pentland, 1991. Eigenfaces for recognition using python, and conduct some numerical experiments to understand how PCA based face recognition works.

Background

Face recognition is an easy task for humans, experiments have shown that even one to three day old babies are able to distinguish between known faces. This motives people to think using computer to recognize face of human. Much of the work in computer recognition in early work focus on using different features such as eyes, mouth and distance and the position of those features to recognize faces. In some way, this works, but it's rather difficult to extend to multiple situations: the change in the head position, the change in the size of the head, and so on. Thus this kind of solution is quiet fragile. M.Turk and A.Pentland based on the Principle Component Analysis, proposed a robust algorithm to recognize face, and introduce a creative view to look at face to be recognized.

Methods/Techniques

The face recognition based on Principle Component Analysis emphasize the face image's global features. To recognize an image as face, we want to extract the relevant information in the image, encode it as efficiently as possible, and compare it with the database we have got.

A facial image of which the size is MXN can be viewed as a point in the high-dimensional image space, and the collection of face image will distributed in the subspace of the high-dimensional image space which we called face space. To present the subspace, we find a basis of the subspace, in which we select the most characteristic features which associated with the largest eigenvalues, then we get the eigenfaces, using those eigenfaces we can reconstruct image which lies in the subspace.

Recognition is performed by projecting a new image into the sub-space spanned by the eigenfaces and then classifying the face by comparing its position in face space with the positions of known individuals.

To perform such task, we have to perform PCA on the covariance matrix of the face dataset:

Assume $\mathrm{X} = \{x_1, x_2, \dots, x_n\}$ is the dataset, in which the x_n is the n^{th} image in the dataset(x_n is flattened as a $\mathrm{MN} * 1$ vector)

1. Compute the mean value of x_n:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

2. Compute the Covariance Matrix:

$$\mathrm{M} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T$$

3. Compute the eigenvalues and the eigenvectors:

$$\mathrm{M}v_i = \lambda v_i$$

   We perform above using the function : numpy.linalg.eigh() in python
4. Order the eigenvectors by sort the corresponding eigenvalues in descending orders.

The eigenvectors we get are full rank, if we want to use only k components, just use eigenvectors[:,k].

But there are still problem involved when we solve the eigenvector problem: the matrix M is of shape: MN X MN, assume M=N=200, then the matrix is like 40000X40000, which is a huge matrix. To solve this problem, we observe that:

$$X^T X v_i = \lambda_i v_i$$

Thus:

$$\mathrm{X}X^T(Xv_i) = \lambda_i(Xv_i)$$

Which means that Xv_i are the eigenvectors of $\mathrm{C} = \mathrm{X}X^T$. Thus we can first find the eigenvectors v_i of X^TX, which is a much smaller matrix compared to X^TX, then left multiply it with X to obtain the eigenvectors of $\mathrm{X}X^T$.

To perform reconstruct task:

1. Project the image to the face space:

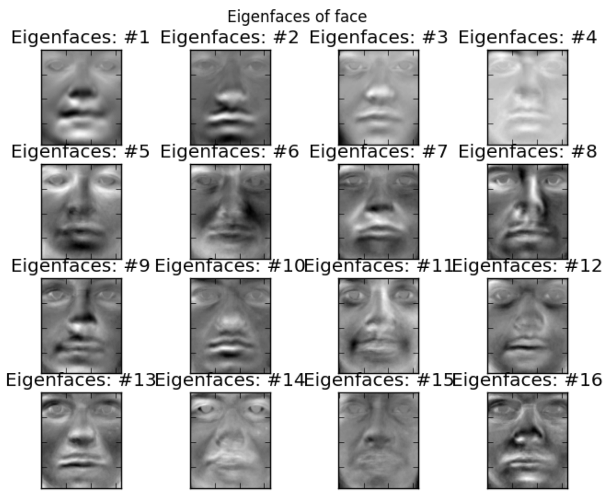
$$\mathrm{y} = W^T(x - \mu)$$

   where W is the eigenvectors matrix we get above.
2. Reconstruct:

$$\mathrm{x} = \mathrm{Wy} + \mu$$

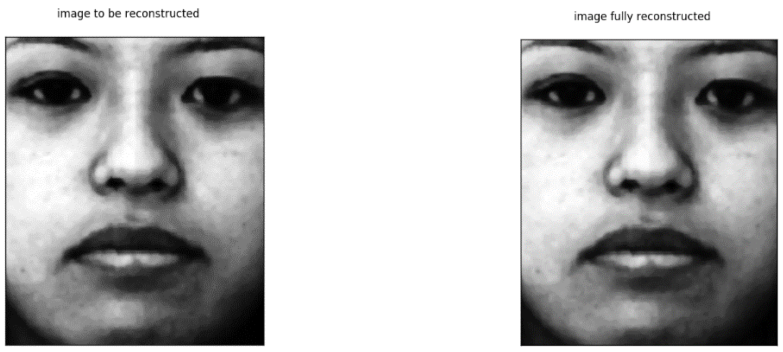
Results&Plots

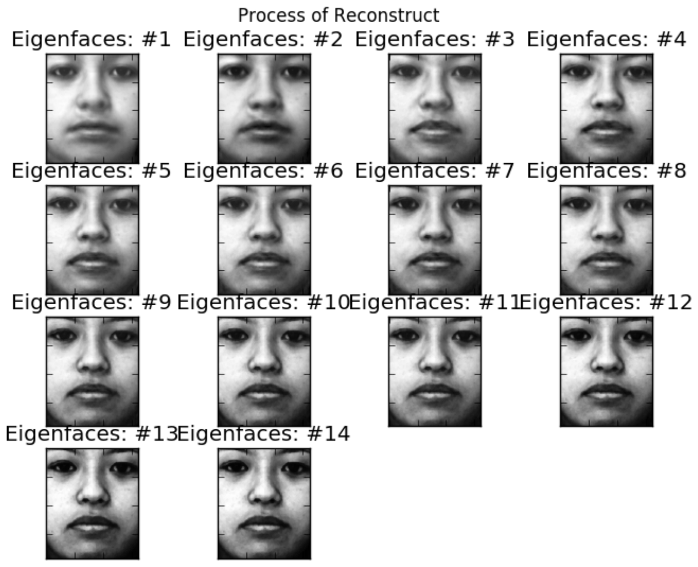
1. Compute the principal components using first 190 people's neutral expression image



We can see that the eigenfaces are just like described, ghost faces. It represent the basis of the face space

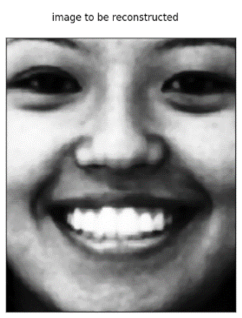
2. Reconstruct one of the 190 people's neutral expression image Using different number of eigenfaces

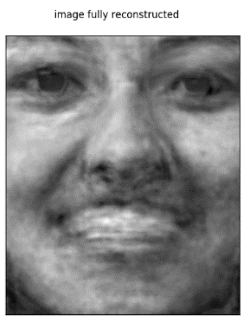


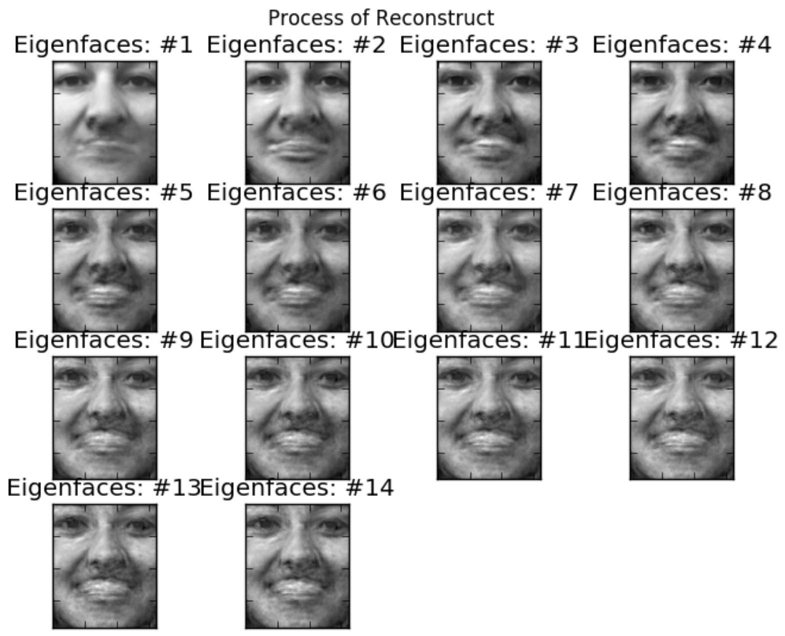


From above picture, we can see that as the number of PCS increase, the face reconstructed is more precise, and when we use all the PCS to reconstruct the face, the reconstructed face are just like the original one. Since the original face is in the face space we constructed, that we can reconstruct it precisely.

3. reconstruct one of 190 people's smiling expression using different number of PCs：

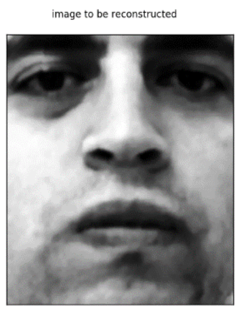


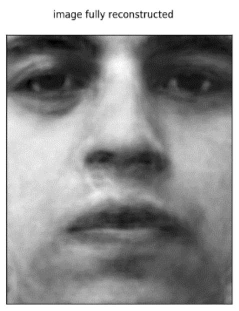


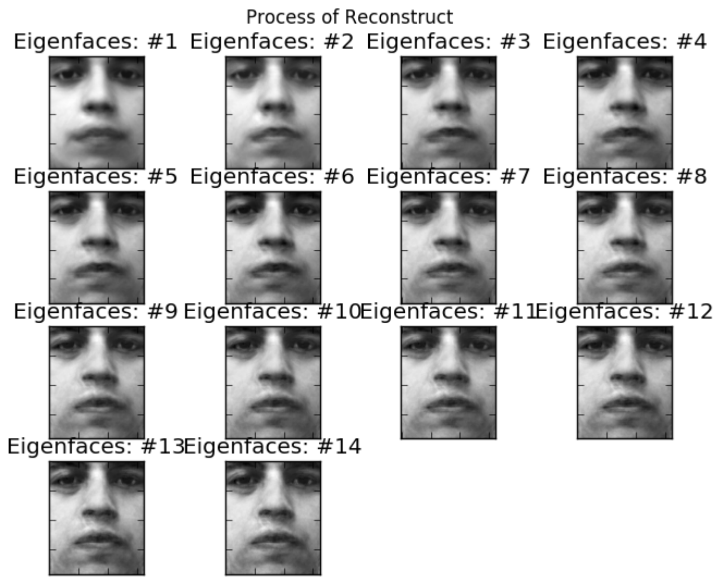


From above result we can see that, using the neutral face space’s eigenfaces cannot reconstruct the smiling face precisely.

4. reconstruct one of the other people's neutral expression image using different number of PCs

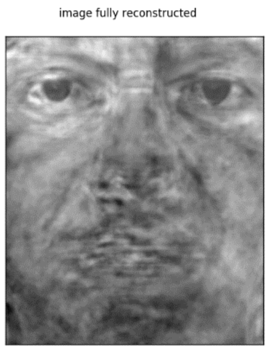


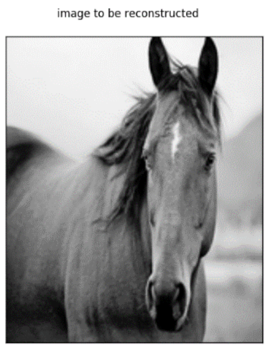




From above figure we can find that as the number of PCS increase, we can get a more vivid face than using only a part of the all PCs, and we can see that the face we constructed using 190 PCs is similar to our target image, which indicate that eigenfaces works when constructing unknown faces.

5. Use other non-human image to reconstruct it using all PCS

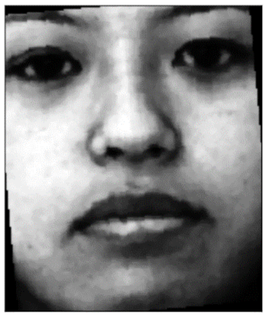




We can see from above figure that an not face-like object cannot be reconstructed.

6. Rotate the image and Observe the difference

image to be reconstructed, rotated degree=5



Rotated image fully reconstructed

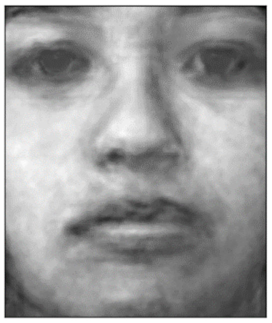
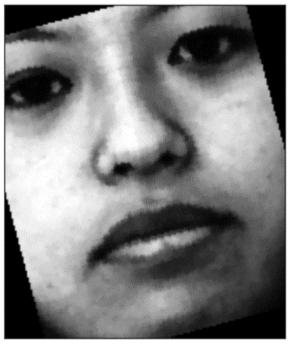


image to be reconstructed, rotated degree=15



Rotated image fully reconstructed

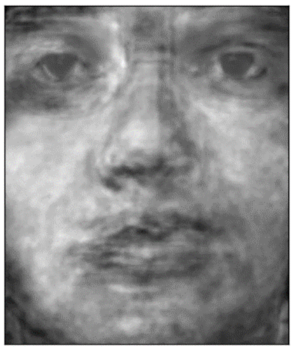
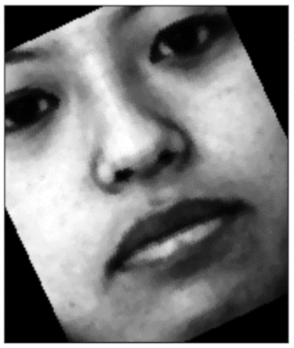


image to be reconstructed, rotated degree=25



Rotated image fully reconstructed

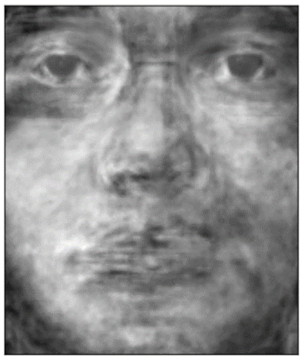
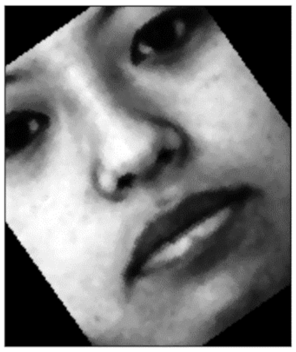
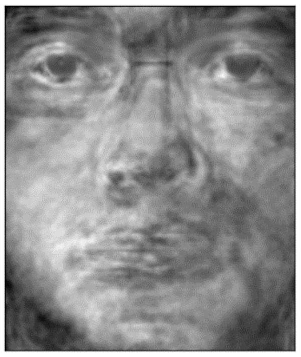
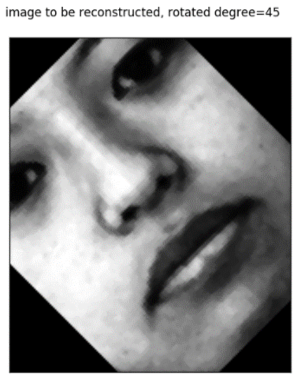


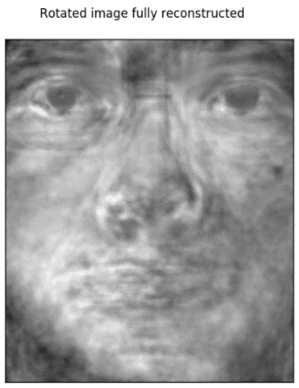
image to be reconstructed, rotated degree=35



Rotated image fully reconstructed

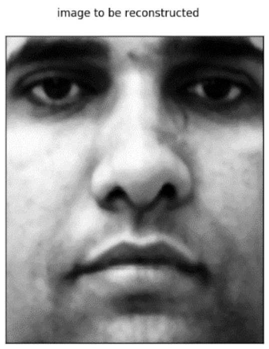


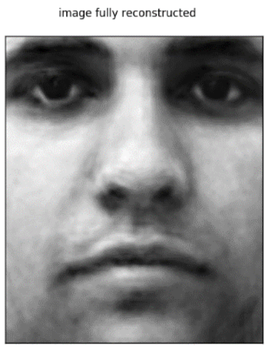


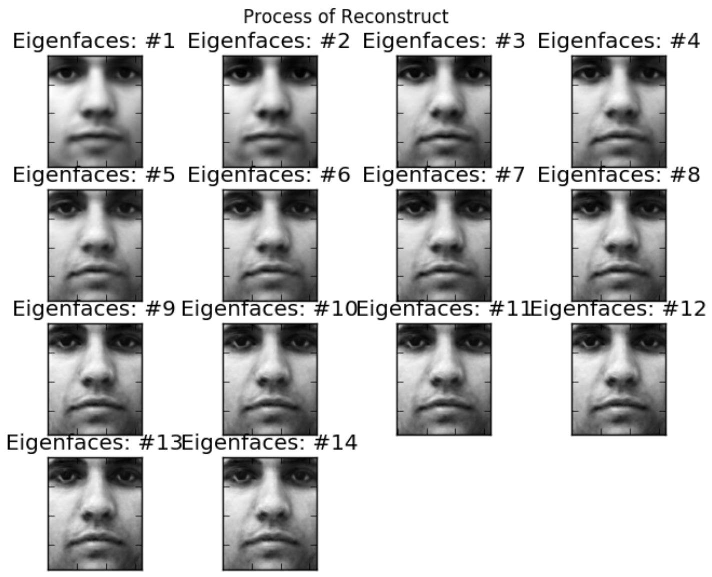


Above picture are images reconstructed when we input the rotated image, we can see that as the degree rotated increase, the performance of reconstruction drops, and when degree> 45, the reconstructed face is nearly failed, which means that rotation's effect on image reconstruction cannot be ignored.

7. Adding noise to one of the 10 people's neutral expression image and reconstruct







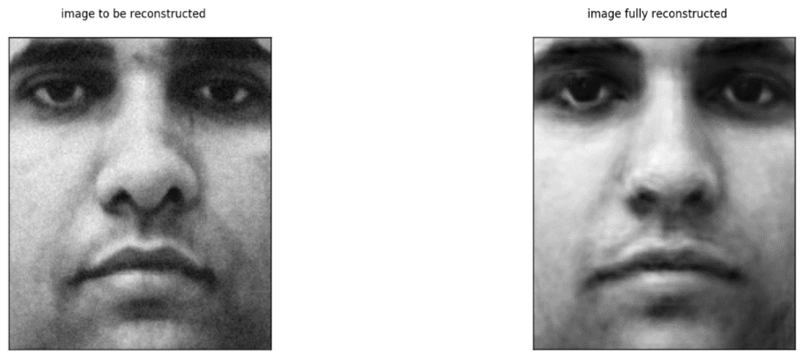
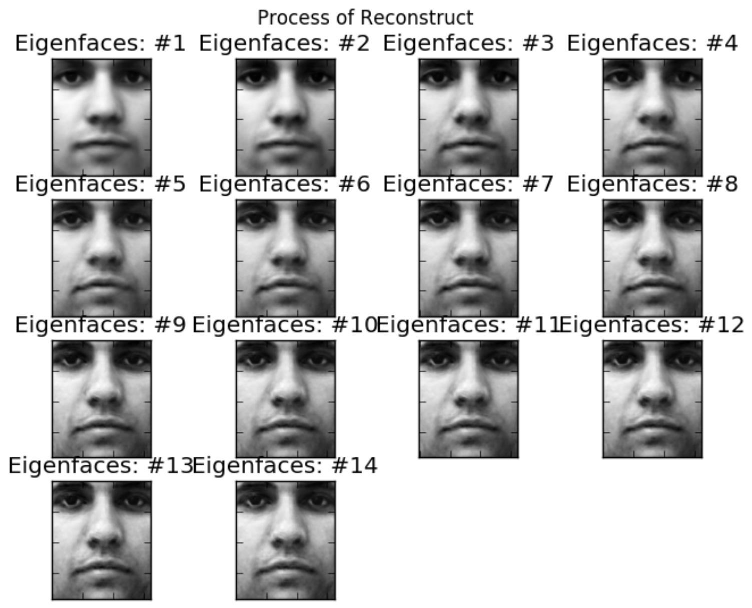


image to be reconstructed
image fully reconstructed


Process of Reconstruct
Eigenfaces: #1
Eigenfaces: #2
Eigenfaces: #3
Eigenfaces: #4
Eigenfaces: #5
Eigenfaces: #6
Eigenfaces: #7
Eigenfaces: #8
Eigenfaces: #9
Eigenfaces: #10
Eigenfaces: #11
Eigenfaces: #12
Eigenfaces: #13
Eigenfaces: #14

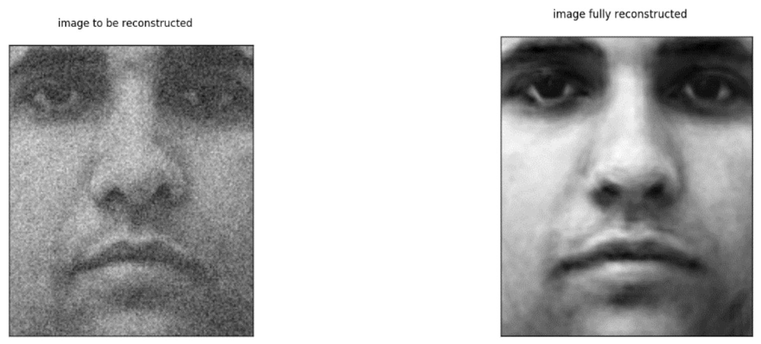
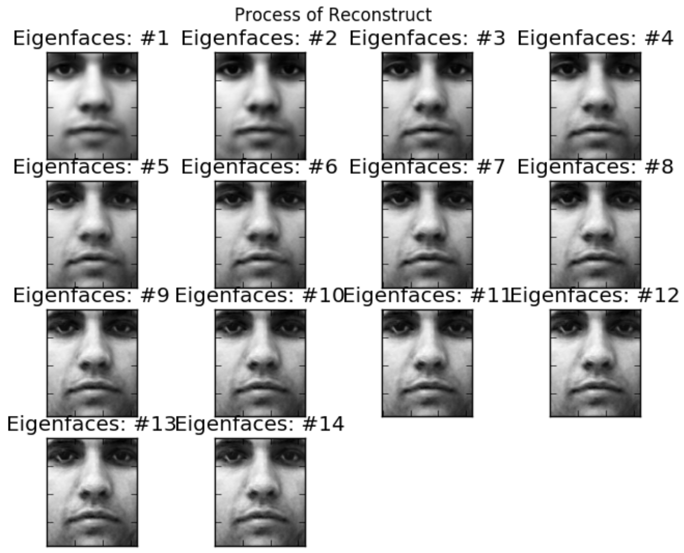


image to be reconstructed
image fully reconstructed


Process of Reconstruct
Eigenfaces: #1
Eigenfaces: #2
Eigenfaces: #3
Eigenfaces: #4
Eigenfaces: #5
Eigenfaces: #6
Eigenfaces: #7
Eigenfaces: #8
Eigenfaces: #9
Eigenfaces: #10
Eigenfaces: #11
Eigenfaces: #12
Eigenfaces: #13
Eigenfaces: #14

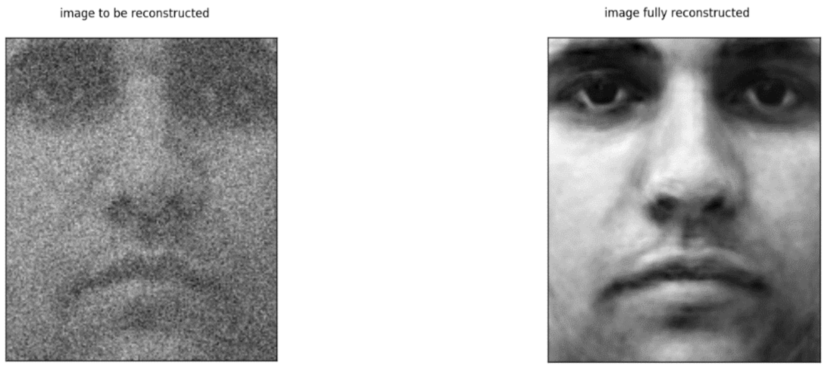
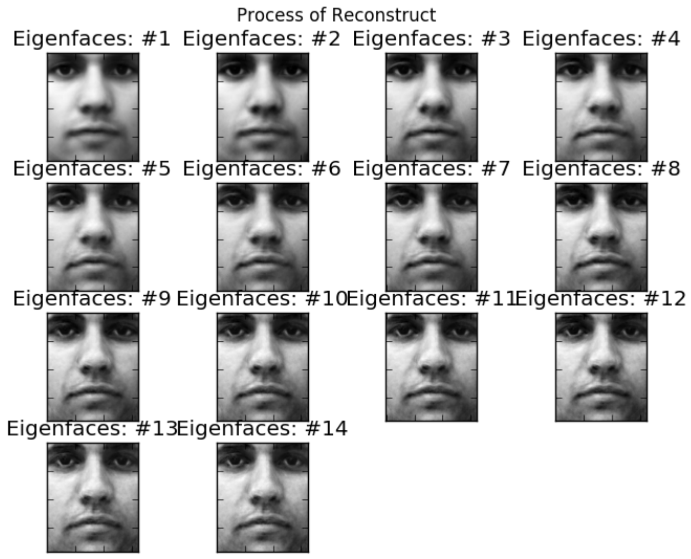
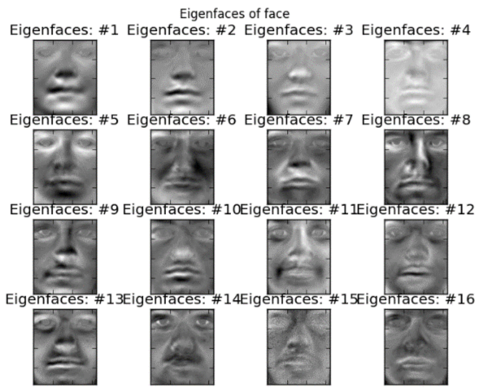


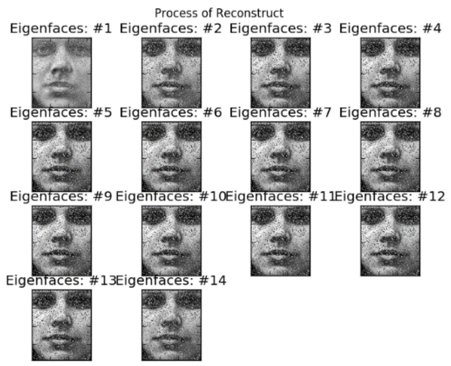
image to be reconstructed
image fully reconstructed

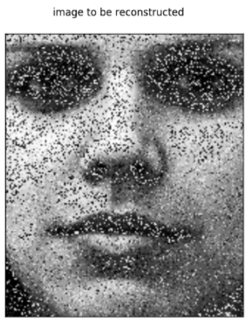

Process of Reconstruct
Eigenfaces: #1
Eigenfaces: #2
Eigenfaces: #3
Eigenfaces: #4
Eigenfaces: #5
Eigenfaces: #6
Eigenfaces: #7
Eigenfaces: #8
Eigenfaces: #9
Eigenfaces: #10
Eigenfaces: #11
Eigenfaces: #12
Eigenfaces: #13
Eigenfaces: #14

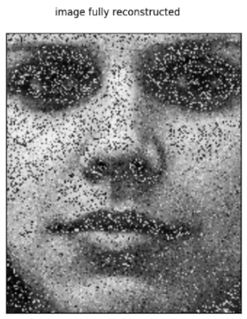
As we can see from the above result, no matter how much noise we add on the input image, we can get a pretty good reconstructed image, which implies that noise on the input damage has little influence on the reconstruction.

8. Reconstruct with contaminated pic









From above picture we can find that reconstructed image is also with noise, which is simply to explain, since the face space are contaminated(which means that the eigenface associated with the input image is with noise, which we can see from the eigenface draw above), so the reconstructed image will also come with noise.

Discussion

From the result we can see that using eigenface to reconstruct face and perform facial recognition is quiet a robust way, since as long as the face space you constructed is not contaminated, the result will not be disturbed by the input noise, and the performance of recognizing unknown face is pretty good.

The weakness of this method is that when the data you used to train the eigenface are contaminated, the result will be effected, and also the method is not perform well when the input face is rotated .

DataSet

http://fei.edu.br/~cet/facedatabase.html
frontalimages_spatiallynormalized_cropped_equalized_part1 (~ 4MB)
frontalimages_spatiallynormalized_cropped_equalized_part2 (~ 4MB)

Python code:

See Code.zip.