Facial Recognition

CS585 Final Project
Qichen Jing, Chennan Ni
12/2/2014

General Topic

"A facial recognition system is a computer application for automatically identifying or verifying a person from a digital image or a video frame from a video source."[8] Generally, there are two approaches for this:

Geometric
Use distinguishing features to identify a person, some of these landmarks includes: distance between the eyes, width of the nose, shape of the cheekbones, length of the jaw line, etc.

Photometric
Use a statistical approach that distills an image into values and compares the values with templates to eliminate variances. Some popular algorithms includes Principal Component Analysis using eigenfaces, Elastic Bunch Graph Matching using the Fisherface algorithm, the Hidden Markov model etc.

Background

Facial Recognition is always an important issue in security field. But it’s not as easy as it looks like in many TV shows.

"In 2002. Boston's Logan Airport ran two separate facial recognition system tests at its security checkpoints using volunteers posing as terrorists over a three-month period and posted disappointing results. Throughout the testing period, the systems correctly identified the volunteers 153 times and failed to identify the volunteers 96 times. As a result of the lackluster success rate of only 61.4 percent, the airport decided to explore other technologies for securing its checkpoints."[9]

New technologies evolve as time goes by. We decide to explore this area, use our knowledge learned in class to implement a few applications, get some experiences in this topic.

Goal

The goal of this project is to implement a system that recognizes a person by matching his or her image to a database of faces: given a set of face images labeled with the person’s identity (the learning set) and an unlabeled set of face images from the same group of people (the test set), identify each person in the test images.

Methods

1. PCA (Principal Component Analysis)

a. Read all images from the training set as a matrix. For example, we have n images and each image have m pixels. Then the matrix is of size n*m. (xi is the matrix of an image)[6]

$$X = \{x_1, x_2, \dots, x_n\}$$

b. Compute the mean of all matrices.

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

c. Normalize each column, minus the mean of each column from each item to compute the the Covariance Matrix S

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

d. Compute the eigenvalues and eigenvectors of S

$$S v_i = \lambda_i v_i, i = 1, 2, \dots, n$$

e. Order the eigenvectors descending by their eigenvalue. The k principal components are the eigenvectors corresponding to the k largest eigenvalues. The k principal components of the observed vector x are then given by

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

$$W = (v_1, v_2, \ldots, v_k)$$

f. According to the information rate we want to guarantee, extract some eigenvalues and corresponding eigen vectors. Decrease the dimensions of your test images by projecting them into the eigen face space.

g. Matching the test images with all training images and return images with distances which are less than the gate.

2. FLD (Fisher's Linear Discriminant)

a. Compute the between-class scatter matrix Sb and within-class scatter matrix Sw, u is the totoal mean, ui is the mean image of class Xi.[5]

$$S_B = \sum_{i=1}^{c} N_i(\mu_i - \mu)(\mu_i - \mu)^T$$

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

b. Look for a projection W, that maximizes the class separability criterion

$$W_{opt} = \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|}$$

c. Solve the general Eigenvalue problem

$$S_B v_i = \lambda_i S_w v_i$$

$$S_W^{-1} S_B v_i = \lambda_i v_i$$

d. To overcome the complication of a singular Sw, use PCA to reduce the dimension of the feature space to N - c, and then applying the standard FLD to reduce the dimension to c - 1.

$$W_{opt}^T = W_{fld}^T W_{pca}^T$$

$$W_{pca} = \arg\max_W \left|W^T S_T W\right|$$

$$W_{fld} = \arg\max_W \frac{\left|W^T W_{pca}^T S_B W_{pca} W\right|}{\left|W^T W_{pca}^T S_W W_{pca} W\right|}$$

3. Sparse representation

The main idea of sparse representation is that we can represent the test sample in an overcomplete dictionary whose base elements are the training samples themselves. If sufficient training samples are available from each class, it will be possible to represent the test samples from the same class. This representation is naturally sparse, involving only a small fraction of the overall training database. And actually, the sparest linear representation of the test sample in terms of the dictionary can be recovered efficiently via l-1 minimization.[11]

Here are basic steps for this algorithm:
a. Given ni training samples of the ith object class, we can construct one matrix to present it.

$$A_i \doteq [v_{i,1}, v_{i,2}, \ldots, v_{i,n_i}] \in \mathbb{R}^{m \times n_i}$$

where vi is the m dimension features extracted from image
b.Combine all matrix for different classes to a larger one

$$A \doteq [A_1, A_2, \ldots, A_k] = [v_{1,1}, v_{1,2}, \ldots, v_{k,n_k}].$$

c.Normalize the columns of A to have l-2 norm.

d. Solve the l-1 minimization problem:

$$\hat{\boldsymbol{x}}_1 = \arg\min \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad A\boldsymbol{x} = \boldsymbol{y}.$$

or

$$\hat{\boldsymbol{x}}_1 = \arg\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1 \ \text{subject to} \ \|A\boldsymbol{x} - \boldsymbol{y}\|_2 \le \varepsilon$$

e.Computer the residuals

$$r_i(\boldsymbol{y}) = \|\boldsymbol{y} - A\ \delta_i(\hat{\boldsymbol{x}}_1)\|_2$$

f.Output:

$$\text{identity}(\boldsymbol{y}) = \arg\min_i r_i(\boldsymbol{y})$$

One main problem is how to implement the optimization problem, I used the augmented langrange multiplier method from appendix of[12]:

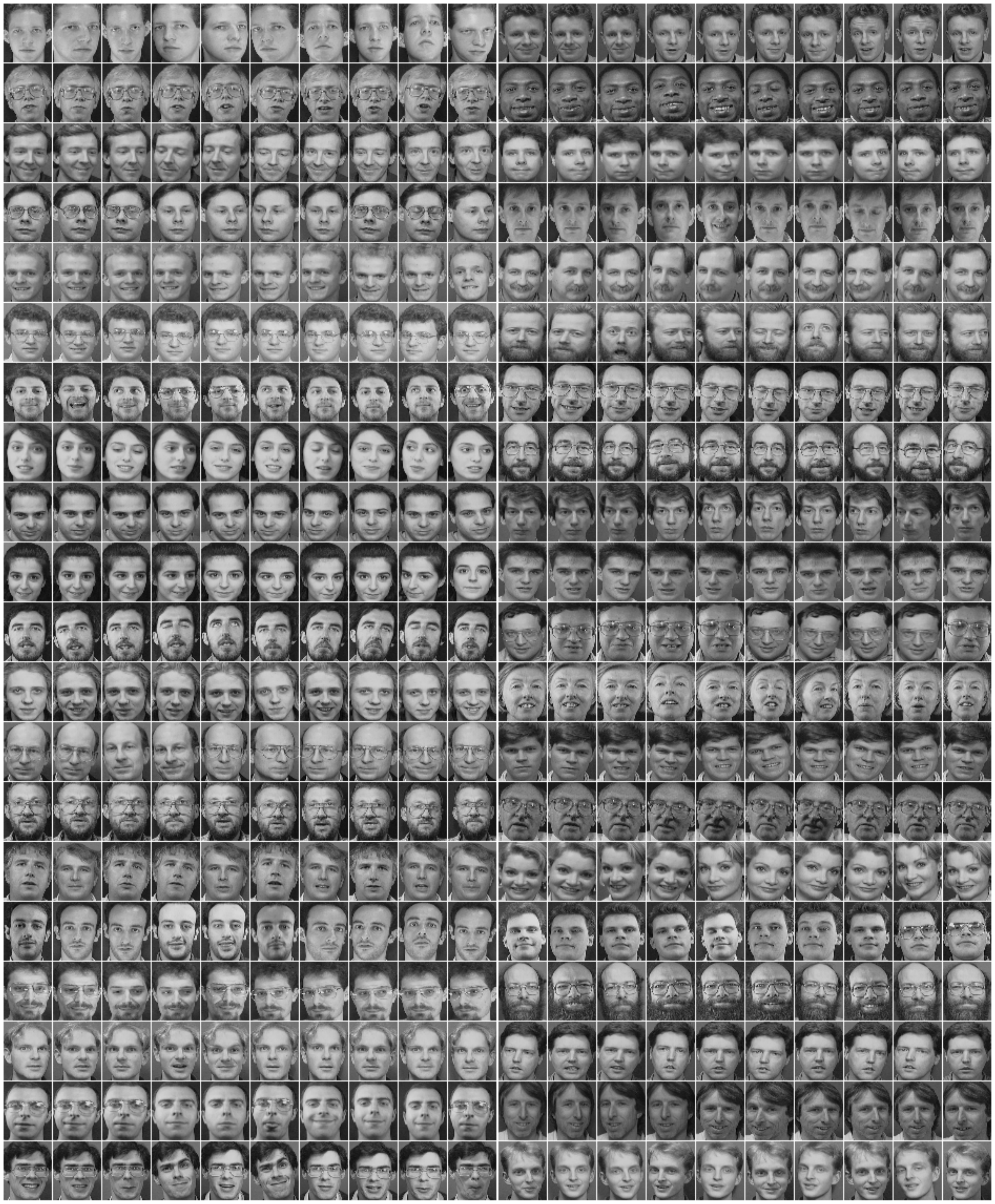
```
1:  Input: $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$, $x_1 = 0$, $e_1 = y$, $\lambda_1 = 0$.
2:  while not converged ($k = 1, 2, \ldots$) do
3:    $e_{k+1} = \mathcal{T}\left(y - Ax_k + \frac{1}{\mu}\lambda_k, \frac{1}{\mu}\right)$;
4:    $t_1 \leftarrow 1$, $z_1 \leftarrow x_k$, $w_1 \leftarrow x_k$;
5:    while not converged ($l = 1, 2, \ldots$) do
6:      $w_{l+1} \leftarrow \mathcal{T}\left(z_l + \frac{1}{\gamma}A^T\left(y - Az_l - e_{k+1} + \frac{1}{\mu}\lambda_k\right), \frac{1}{\mu\gamma}\right)$;
7:      $t_{l+1} \leftarrow \frac{1}{2}\left(1 + \sqrt{1 + 4t_l^2}\right)$;
8:      $z_{l+1} \leftarrow w_{l+1} + \frac{t_l - 1}{t_{l+1}}(w_{l+1} - w_l)$;
9:    end while
10:   $x_{k+1} \leftarrow w_l$,   $\lambda_{k+1} \leftarrow \lambda_k + \mu(y - Ax_{k+1} - e_{k+1})$;
11: end while
12: Output: $x^* \leftarrow x_k, e^* \leftarrow e_k$.
```

where gamma is the eigenvalue of matrix ATA, mu is 2m/l-norm of y.

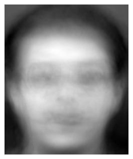
Experiments and Results

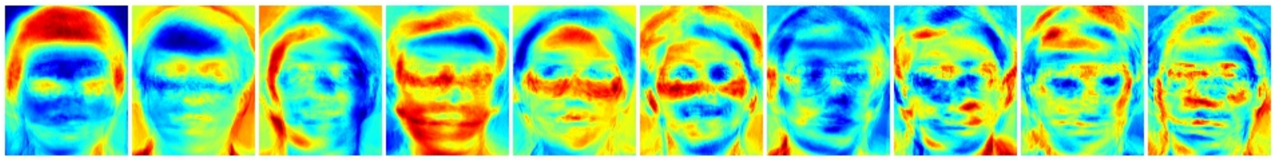
Part 1. PCA

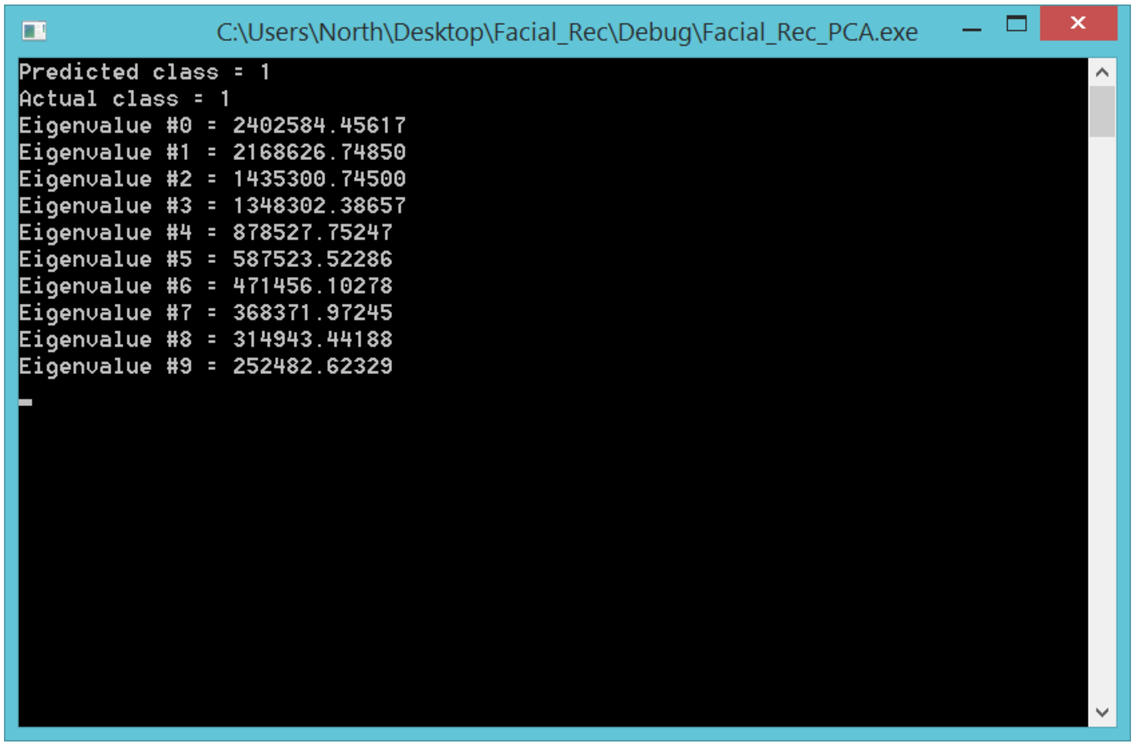
We use the ATT database, cosists of 40*10 images. There are ten different images of each of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement). Below is a preview image of the Database of Faces.[10]



We pick some images out of the batabase, use the rest as training samples, get the mean faces, eigenfaces and predict result.

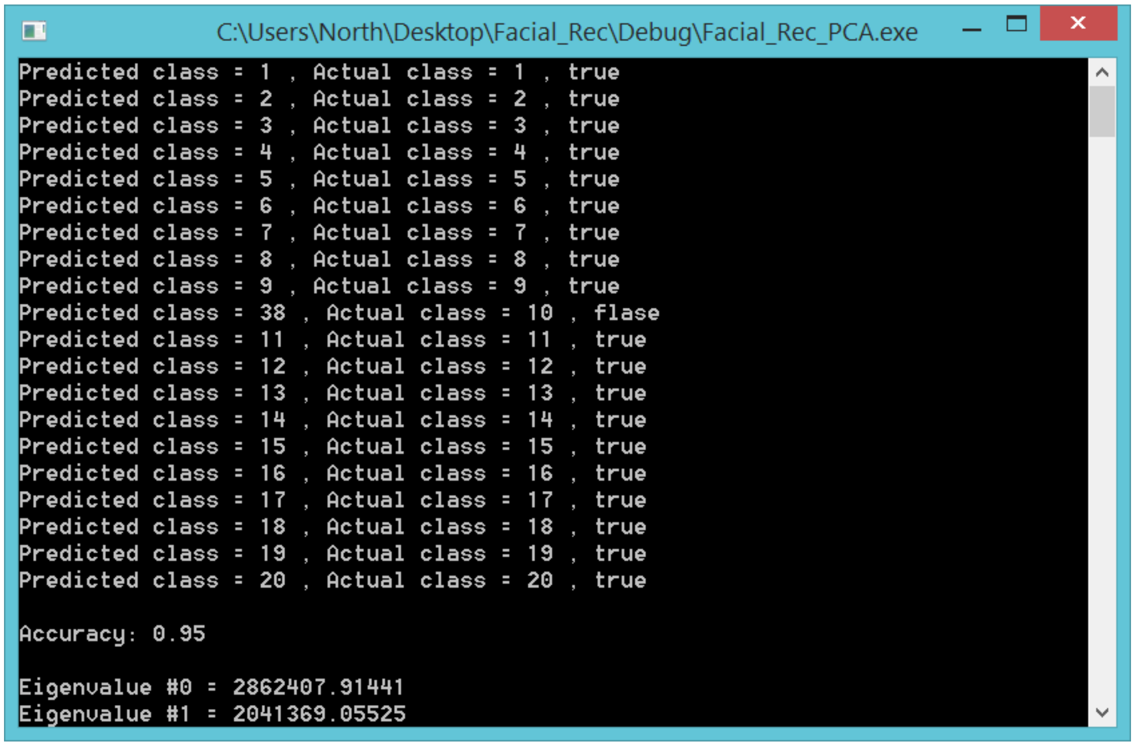






```
Predicted class = 1
Actual class = 1
Eigenvalue #0 = 2402584.45617
Eigenvalue #1 = 2168626.74850
Eigenvalue #2 = 1435300.74500
Eigenvalue #3 = 1348302.38657
Eigenvalue #4 = 878527.75247
Eigenvalue #5 = 587523.52286
Eigenvalue #6 = 471456.10278
Eigenvalue #7 = 368371.97245
Eigenvalue #8 = 314943.44188
Eigenvalue #9 = 252482.62329
```

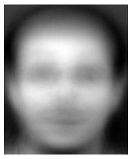
To test the accuracy of this algrithom, I use 20 faces out of 40 faces, the accuracy is 95%.

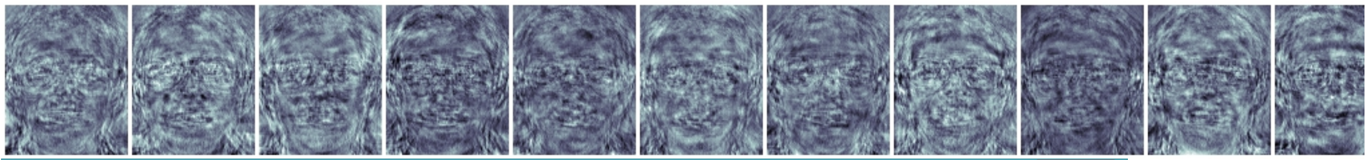


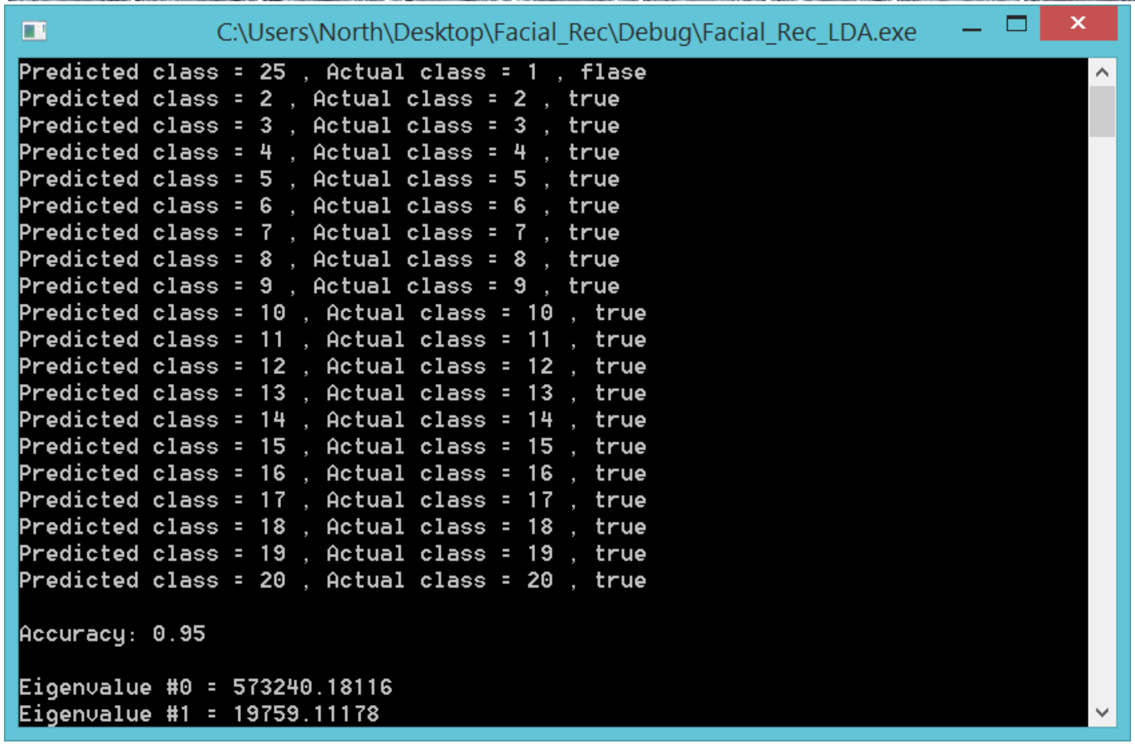
```
Predicted class = 1 , Actual class = 1 , true
Predicted class = 2 , Actual class = 2 , true
Predicted class = 3 , Actual class = 3 , true
Predicted class = 4 , Actual class = 4 , true
Predicted class = 5 , Actual class = 5 , true
Predicted class = 6 , Actual class = 6 , true
Predicted class = 7 , Actual class = 7 , true
Predicted class = 8 , Actual class = 8 , true
Predicted class = 9 , Actual class = 9 , true
Predicted class = 38 , Actual class = 10 , flase
Predicted class = 11 , Actual class = 11 , true
Predicted class = 12 , Actual class = 12 , true
Predicted class = 13 , Actual class = 13 , true
Predicted class = 14 , Actual class = 14 , true
Predicted class = 15 , Actual class = 15 , true
Predicted class = 16 , Actual class = 16 , true
Predicted class = 17 , Actual class = 17 , true
Predicted class = 18 , Actual class = 18 , true
Predicted class = 19 , Actual class = 19 , true
Predicted class = 20 , Actual class = 20 , true

Accuracy: 0.95

Eigenvalue #0 = 2862407.91441
Eigenvalue #1 = 2041369.05525
```

Part 2. FLD

We use the same database in this part as the first part. Get the mean faces, eigenfaces and results. I use 20 faces out of 40 faces, the accuracy is also 95%.







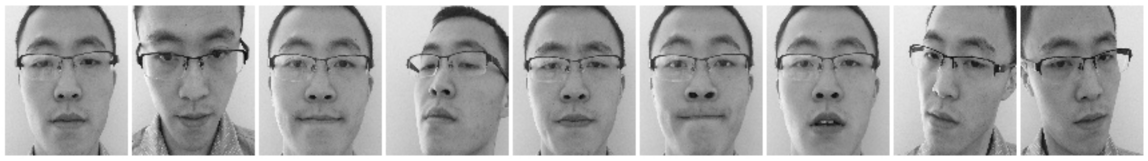
```
Predicted class = 25 , Actual class = 1 , flase
Predicted class = 2 , Actual class = 2 , true
Predicted class = 3 , Actual class = 3 , true
Predicted class = 4 , Actual class = 4 , true
Predicted class = 5 , Actual class = 5 , true
Predicted class = 6 , Actual class = 6 , true
Predicted class = 7 , Actual class = 7 , true
Predicted class = 8 , Actual class = 8 , true
Predicted class = 9 , Actual class = 9 , true
Predicted class = 10 , Actual class = 10 , true
Predicted class = 11 , Actual class = 11 , true
Predicted class = 12 , Actual class = 12 , true
Predicted class = 13 , Actual class = 13 , true
Predicted class = 14 , Actual class = 14 , true
Predicted class = 15 , Actual class = 15 , true
Predicted class = 16 , Actual class = 16 , true
Predicted class = 17 , Actual class = 17 , true
Predicted class = 18 , Actual class = 18 , true
Predicted class = 19 , Actual class = 19 , true
Predicted class = 20 , Actual class = 20 , true

Accuracy: 0.95

Eigenvalue #0 = 573240.18116
Eigenvalue #1 = 19759.11178
```

Part 3. Facial Recognition in Camera

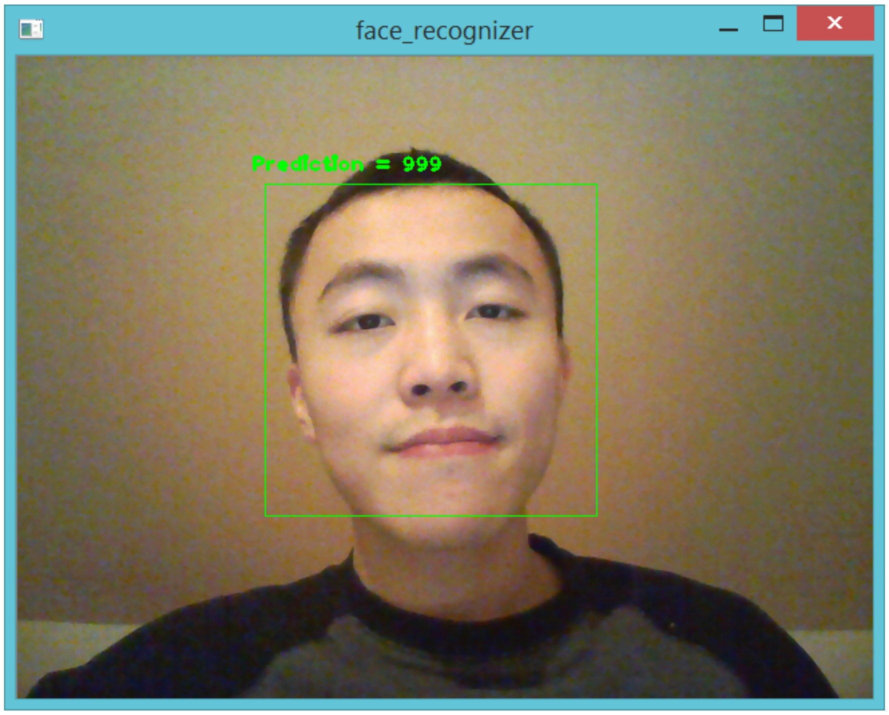
Performing face recognition in surveillance videos is one of the most common applicaitons. I use `CascadeClassifier` in face detection and `Fisherfaces` method for face recognition.

The input selfie images looks like:

Then I crop the image to have better alignment of faces in image.



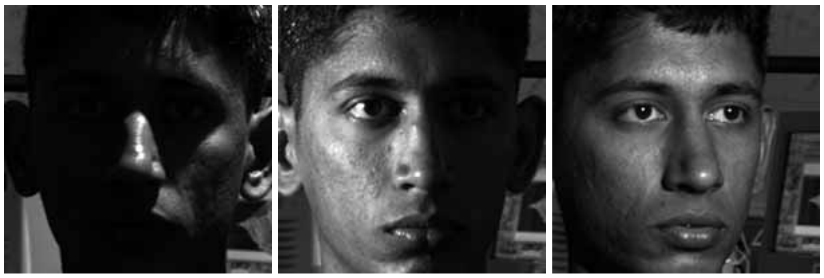
I use the cropped images and att database images as training set. Then open the camera, the system can automatically pick out faces from camera and give perdiction. (999 is the label of my faces)

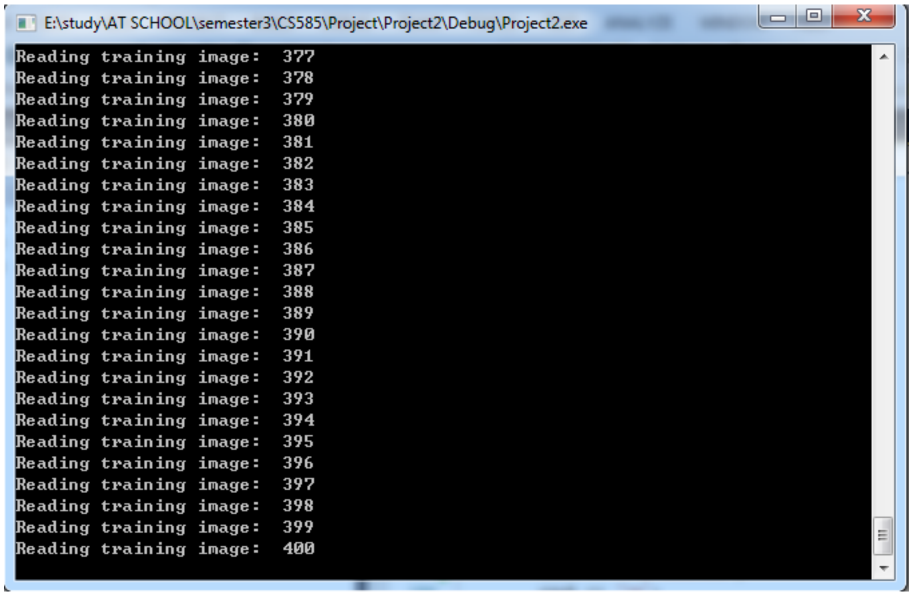


Part 4. Sparse representation

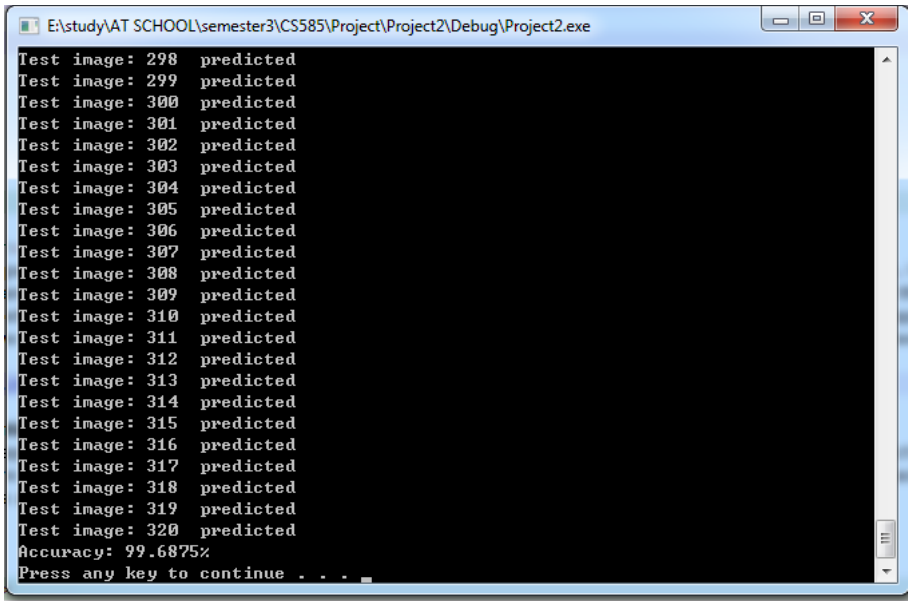
The database for sparse representation is: Yale B face database. I choose ten subjects with 352 images for each. And the images are of various illumination, expression.
Here is an example of it:

After using face detection method in opencv, we get face images of various size(All of them are around about 250*250) as our training set and testing set. We down sample all images to 12*10.
Examples:


Firstly we choose 400 training faces as dictionary, 40 for each subjects. And test on 320 images, 32 for each.



```
E:\study\AT SCHOOL\semester3\CS585\Project\Project2\Debug\Project2.exe
Reading training image:  377
Reading training image:  378
Reading training image:  379
Reading training image:  380
Reading training image:  381
Reading training image:  382
Reading training image:  383
Reading training image:  384
Reading training image:  385
Reading training image:  386
Reading training image:  387
Reading training image:  388
Reading training image:  389
Reading training image:  390
Reading training image:  391
Reading training image:  392
Reading training image:  393
Reading training image:  394
Reading training image:  395
Reading training image:  396
Reading training image:  397
Reading training image:  398
Reading training image:  399
Reading training image:  400
```



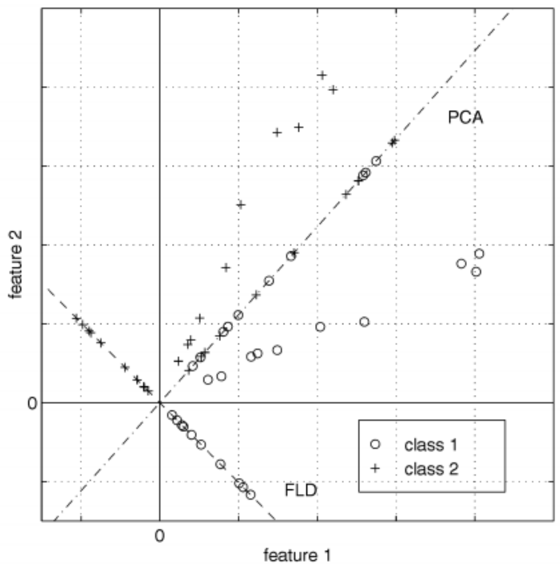
```
E:\study\AT SCHOOL\semester3\CS585\Project\Project2\Debug\Project2.exe
Test image: 298  predicted
Test image: 299  predicted
Test image: 300  predicted
Test image: 301  predicted
Test image: 302  predicted
Test image: 303  predicted
Test image: 304  predicted
Test image: 305  predicted
Test image: 306  predicted
Test image: 307  predicted
Test image: 308  predicted
Test image: 309  predicted
Test image: 310  predicted
Test image: 311  predicted
Test image: 312  predicted
Test image: 313  predicted
Test image: 314  predicted
Test image: 315  predicted
Test image: 316  predicted
Test image: 317  predicted
Test image: 318  predicted
Test image: 319  predicted
Test image: 320  predicted
Accuracy: 99.6875%
Press any key to continue . . .
```

Then we randomly choose half of all images as training images and the rest part as testing set.(1760 as basis and 1760 images for testing). And final accuracy is 91.478%

Discussion

PCA & FLD

The results of both experiments of PCA and FLD are good, but our sample size is not big enough to represent all the cases. For PCA, it doesn't consider any classes and so a lot of discriminative information may be lost when throwing components away. The training samples we use in the experiments are under admittedly idealized conditions, "the variation within class lies in a linear subspace of the image space". For example, the lighting conditions are very similar in all images. That's one of the reasons why this method has a 95% accuracy. To build a more reliable method for reducing the dimensionality of the feature space, we use FLD. FLD method learns a class-specific transformation matrix, it should have better recognition rates than the PCA method because it does not capture illumination as obviously as the previous method. Below is a comparison of principal component analysis (PCA) and Fisher's linear discriminant (FLD) for a two class problem where data for each class lies near a linear subspace from Peter's paper[7].



Sparse representation
The result of sparse representation is really good more than 91% on Yale B database and this method is effective to predict the label of image with occlusion. From the paper I refer to, they test on images under random corruption. Even when the corruption rate is 70%, this algorithm could achieve 90.7% accuracy rate. And they also tested on images with only nose, right eye and mouth&chin, achieving 87.3%, 93.7%, 98.3% accuracy respectively. One problem in our experiment is misalignment: the face part is firstly using face detection algorithm, so it causes alignment and we will focus on this problem in the future.

Conclusions

From this project, we learned some basic methods in facial recognition. For PCA, the principal components are projected onto the eigenspace to find the eigenfaces and an unknown face is recognized from the minimum euclidean distance of projection onto all the face classes. For FLD, we try to maximize the difference of between-class scatter matrix and within-class scatter matrix. PCA has a good recognition rate when the faces images are under good illumination conditions. FLD performs better than PCA when the lighting varies. But both methods suffer when the faces have a big angle of inclination from the image plane. Modifications are needed to improve the recognition rate for these methods.

Except for implementing the classical PCA and FLD algorithm, we also learned a lot about the cutting edge sparse representation method which is robust to illumination, occlusion. And it helps us to understand the face recognition problem from the other perspective. One the we need to do in the future is to handle misalignment for it. We can do this by solving the other optimization problem:

$$\hat{\tau} = \arg\min_{\boldsymbol{x},\boldsymbol{e},\tau\in T} \|\boldsymbol{x}\|_1 + \|\boldsymbol{e}\|_1 \quad \text{subj to} \quad \boldsymbol{y}\circ\tau = A\boldsymbol{x} + \boldsymbol{e}.$$

In the future, we decide to try other methods like Skin Texture Analysis, 3D Facial Recognition and deep learning method.

Credits and Bibliography

[1] Duda, Richard O. and Hart, Peter E. and Stork, David G., Pattern Classification (2nd Edition) 2001.
[2] Fisher, R. A. The use of multiple measurements in taxonomic problems. Annals Eugen. 7 (1936), 179–188.
[3] Zhao, W., Chellappa, R., Phillips, P., and Rosenfeld, A. Face recognition: A literature survey. ACM Computing Surveys (CSUR) 35, 4 (2003), 399–458.
[4] "Principal Component Analysis and Linear Discriminant Analysis with GNU Octave". Internet. 2 Dec. 2014. http://www.bytefish.de/blog/pca_lda_with_gnu_octave/
[5] "Fisher Linear Discriminant Analysis". paper online. 2 Dec. 2014. http://www.ics.uci.edu/~welling/classnotes/papers_class/Fisher-LDA.pdf
[6] "Face Recognition with OpenCV". opencv online documentation. 2 Dec. 2014. http://docs.opencv.org/trunk/modules/contrib/doc/facerec/
[7] "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection". Peter N. Belhumeur, Joao~ P. Hespanha, etc. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 19, NO. 7, JULY 1997.
[8] "Facial Recognition" Wikipedia. 3 Dec. 2014. http://en.wikipedia.org/wiki/Facial_recognition_system
[9] "Face Recognition History" Internet. 3 Dec. 2014. https://epic.org/privacy/facerecognition/
[10] "ATT Database Faces" Internet. 3 Dec. 2014. http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
[11]Wright J, Yang A Y, Ganesh A, et al. Robust face recognition via sparse representation[J]. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2009, 31(2): 210-227.
[12]Wagner A, Wright J, Ganesh A, et al. Toward a practical face recognition system: Robust alignment and illumination by sparse representation[J]. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2012, 34(2): 372-386.
[13]Yang A Y, Sastry S S, Ganesh A, et al. Fast ℓ 1-minimization algorithms and an application in robust face recognition: A

review[C]//Image Processing (ICIP), 2010 17th IEEE International Conference on. IEEE, 2010: 1849-1852. [14]Mohammed A A, Minhas R, Jonathan Wu Q M, et al. Human face recognition based on multidimensional PCA and extreme learning machine[J]. Pattern Recognition, 2011, 44(10): 2588-2597. [15]Chen Y C, Patel V M, Phillips P J, et al. Dictionary-based face recognition from video[M]//Computer Vision–ECCV 2012. Springer Berlin Heidelberg, 2012: 766-779.