

Full-name: Phan Thi My Dung

Lecturer: Lee Chil Woo

Class: Advance in Pattern Recognition

FACE RECOGNITION USING EIGENFACES

1. Introduction

The application is a face recognition system. The scheme is based on an information theory approach that decomposes face images into a small set of characteristic feature images called “eigenfaces”, which may be thought of as the principal components of the initial training set of face images (“face-space”) and then classifying the face by comparing its position in face space with the position of known individuals.

This approach to face recognition involves the following initialization operations:

- i. Acquire an initial set of face images (the training set)
- ii. Calculate the eigenfaces from the training set, keeping only the M images that correspond to the highest eigenvalues. These M images define the *face space*. As new faces are experienced, the eigenfaces can be updated or recalculated.
- iii. Calculate the corresponding distribution in M-dimensional weight space for each known individual, by projecting their face images onto the *face space*.

The following steps are used to recognize new face images:

- i. Calculate a set of weights based on the input image and the M eigenfaces by projecting the input image onto each of the eigenfaces.
- ii. Determine if the image is a face at all by checking to see if the image is sufficiently close to face space
- iii. If it is a face, classify the weight pattern as either a known or unknown person.

2. Calculate eigenfaces

Let a face image $I(x,y)$ be a two-dimensional $N \times N$ array of (8-bit) intensity values. An image may also be considered as a vector of dimension N^2 , so that a typical image of size 256×256 becomes a vector of dimension 65,536. The main idea of principal component analysis is to find the vectors that best account for the distribution of face images within the entire image space. These vectors define the subspace of face images, which we call “eigenfaces”. Each vector is of length N^2 , describes an $N \times N$ image, and is a *linear combination of the original face images*.

Let the training set of face image be $\Gamma_1, \Gamma_2, \dots, \Gamma_M$.

The average face: $\Psi = \frac{1}{M} \sum_{n=1}^M \Gamma_n$

Each face differs from the average: $\Phi_i = \Gamma_i - \Psi$

The vectors u_k and scalars λ_k are the eigenvectors and eigenvalues of the covariance matrix of the face images Φ_i

$$C = \frac{1}{M} \sum_{n=1}^M \phi_n^T \phi_n = AA^T \quad \text{where } A = [\Phi_1, \Phi_2, \dots, \Phi_M].$$

Following the analysis in the paper, we construct the M by M matrix

$L = A^T A$, where $L_{mn} = \Phi_m^T \Phi_n$, and find the M eigenvectors, v_l of L. These vectors determine linear combinations of the M training set faces to form the eigenfaces u_l

$$u_l = \sum_{k=1}^M v_{lk} \Phi_k, \quad l = 1, \dots, M$$

3. Recognize a face image

The new face image Γ is projected into facespace by: $\omega_k = \mathbf{u}_k^T (\Gamma - \Psi)$

The weights form a vector $\Omega^T = [\omega_1, \omega_2, \dots, \omega_M]$.

The euclidean distance: $\epsilon_k^2 = \|\Omega - \Omega_k\|^2$ measure the distance between the new image and face class k. A face is classify belong to class k when the minimum ϵ_k is below some chosen threshold θ_ϵ . Otherwise the face is classified as unknown.

Distance between image and face space

$$\epsilon^2 = \|\Phi - \Phi_f\|^2$$

There are four possibilities:

- $\epsilon_k < \theta_{\epsilon k}$ and $\epsilon < \theta_\epsilon$: *known person*
- $\epsilon_k < \theta_{\epsilon k}$ and $\epsilon \geq \theta_\epsilon$: *unknown person*
- $\epsilon_k \geq \theta_{\epsilon k}$ and $\epsilon < \theta_\epsilon$: *not a human face*
- $\epsilon_k \geq \theta_{\epsilon k}$ and $\epsilon \geq \theta_\epsilon$: *not a human face*

4. Implementation and Experimental result

- Training Data: the figure 1 shows train face images, number of images $M = 12$, size of image is 160 by 160

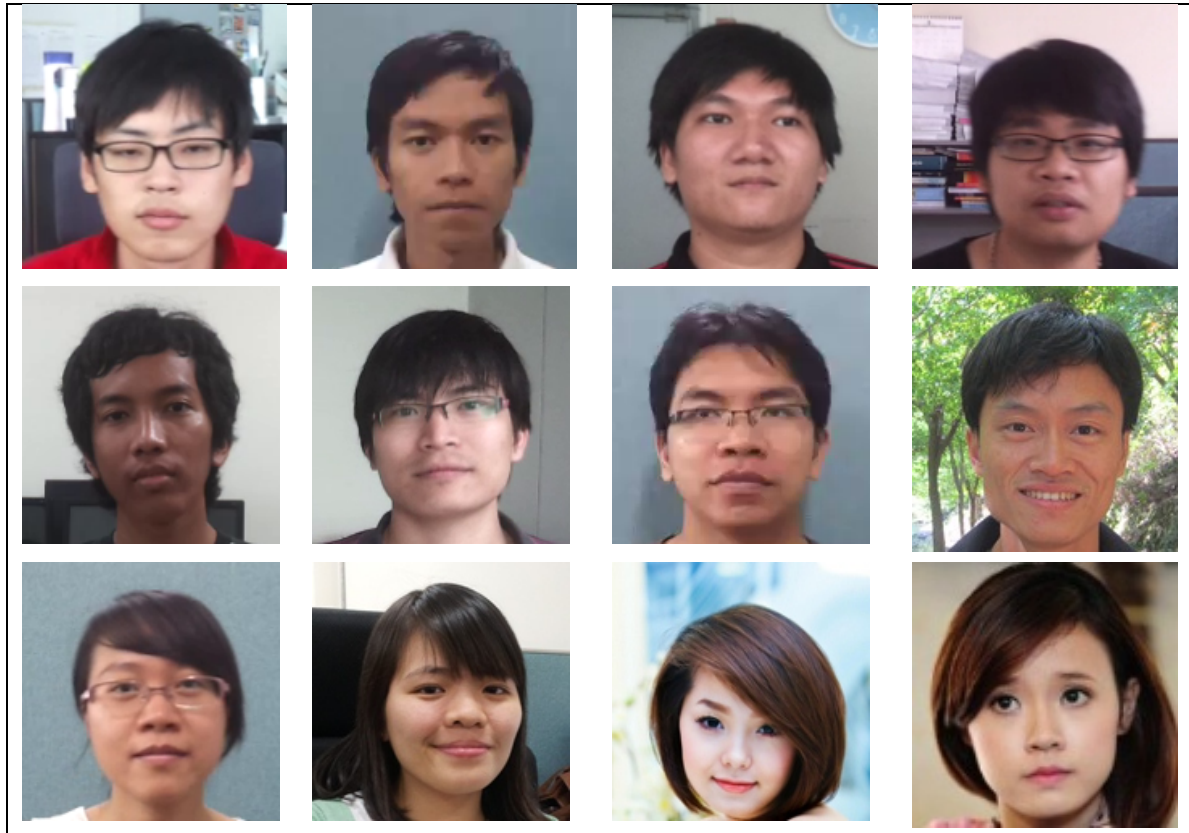


Figure 1: Training faces

4.1. Training phase

- o Chose $M' = 12$ because M is small (default $M' = 20$ if $M < M' \Rightarrow M' = M$)
- o Calculate mean face

```
% calculate mean face
for i = 1:m
    f = strcat('Train/',listFiles(i).name);
    org_img = imread(f);
    img = rgb2gray(org_img);
    [W,H] = size(img);
    % convert matrix of image to vector
    im(:,i) = double(reshape(img,[],1));
    si = si + im(:,i);
end
n = size(im(:,1),1);

% average face
si = double(si/m);
mean_face = uint8(reshape(si,W,H));
```

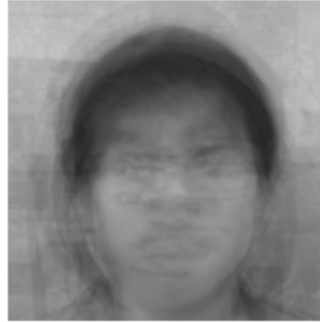


Figure 2: Mean face

- o Calculate eigenfaces

```
% calculate L matrix to get eigenvector
for i = 1:m
    % calculate face differs from average
    pi(:,i) = im(:,i) - si;
    pi_i = double(pi(:,i));
end
for i = 1:m
    for j = 1:m
        L(i,j) = pi(:,i)'*pi(:,j);
    end
end
[eVectors,eValues] = eigs(L,m_eigens);
% eigenfaces
for i=1:m_eigens
    eigenFaces(:,i) = zeros(n,1);
    for j=1:m_eigens
        eigenFaces(:,i) = eigenFaces(:,i) + eVectors(j,i)*pi(:,j);
    end
end
end
```

- o Calculate weight corresponding for each class (every image will represent for each class)

```
% calculate weight
omega_k = zeros(m_eigens,m_eigens);
for i= 1:m
    for j = 1:m_eigens
        omega_k(j,i) = eigenFaces(:,j)'*(im(:,i) - si);
    end
end
end
```

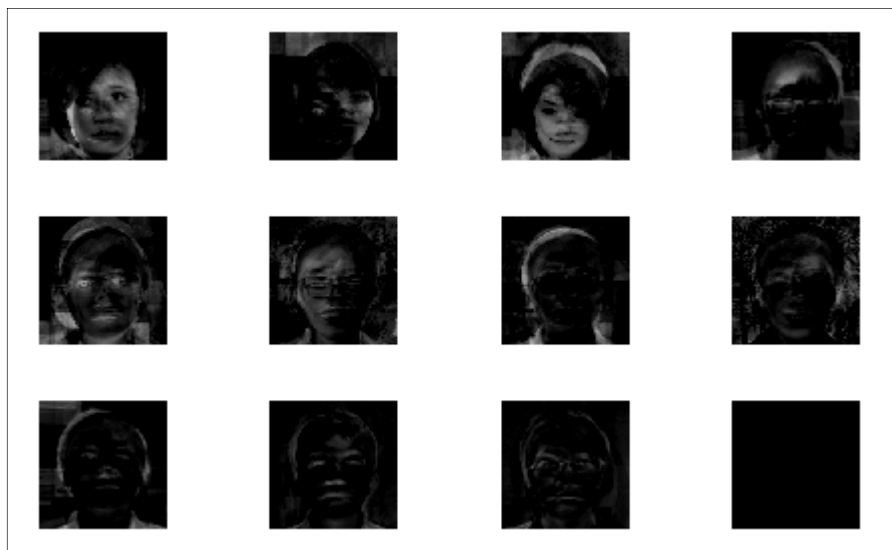


Figure 3: Eigenfaces

4.2. Recognition phase

- $\Theta_{\epsilon k} = 25000000$, $\Theta_{\epsilon} = 16900$ from the run time experimentation.
- The θ_{ϵ} is threshold for checking whether the face is close to face-space
- The θ is threshold for checking whether the face is close to face-class
- The test data includes 8 face images of known people whose face images are in training data, 2 face images of unknown people whose face images are not in training data and 2 images of apples which are not human face

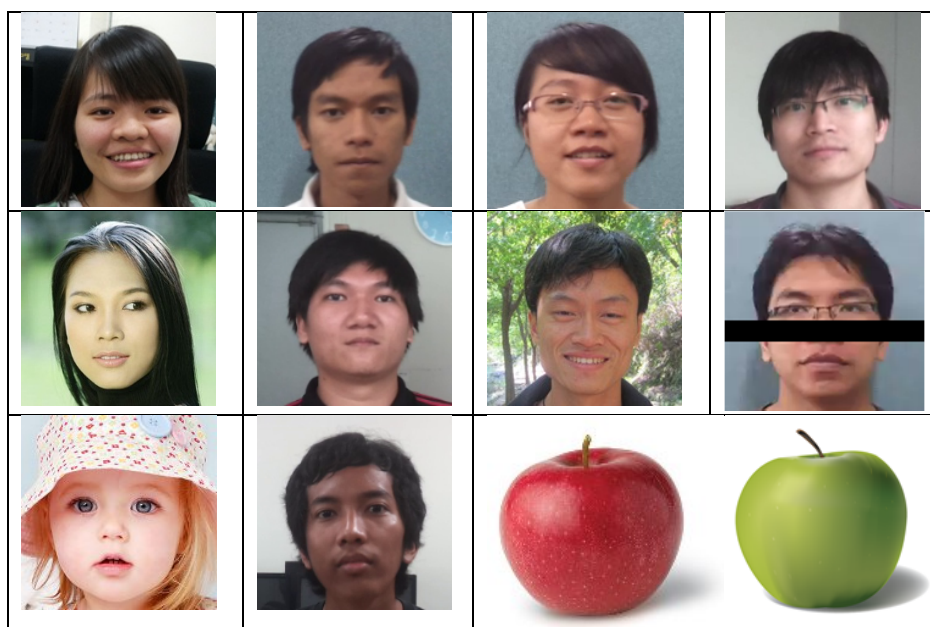


Figure 4: Testing faces








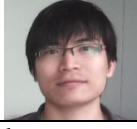

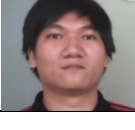
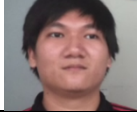
```

% project into face space
m_eigens = size(eigenFaces,2);
omega = zeros(m_eigens,1);
pi = img - si;
for i=1:m_eigens
    omega(i) = eigenFaces(:,i)'\*pi;
end

pi_f = zeros(size(img,1)*size(img,2),1);
for i=1:m_eigens
    pi_f = pi_f + omega(i)*eigenFaces(:,i);
end
% calculate distance to face space
esiplon = norm(pi-pi_f);

% check img close to face class
for i=1:m_eigens
    esp_k(i) = norm(omega - omega_k(:,i));
end
[m_esiplon_k,index] = min(esp_k);

```

Input Face	Distance to face space	Distance to face class	Result
	1.58462e+12	3.8356e+07	
	4.10679e+11	14554707.8695	
	4.66791e+11	1.96818e+07	
	7.36295e+11	3.12586e+06	
	9.52175e+11	6.00668e+07	Unknown person
	2.82568e+11	1.37865e+07	










	4.24406e+11	2.35789e+07	
	3.71205e+11	2.14141e+07	
	15680.5539	88241838.2103	Unknown person
	1.22187e+12	1.82295e+07	
	1.68289e+12	-	Not human face
	1.69452e+12	-	Not human face

Figure 5: Result of testing for 14 images

- Passed testcase: 11/12
- Failed testcase: 1/12

REFERENCES

“Eigenfaces for recognition,” and “Face recognition using eigenfaces”
M. Turk and A. Pentland, 1991