Introduction

Background

Eigenfaces is a crucial concept concerning the computer vision of humanity face recognition. According to the illustration of Matthew and Alex \cite{1}, the eigenfaces actual are a set of eigenvectors which used to represent images with the corresponding appropriate weight values. This mechanism is extremely similar with utilizing the Cartesian Coordination System to describe points in the three-dimension space. The latter utilizes the x, y, and z axes to determine the position of a point and the former utilizes the eigenfaces to determine an image. In this process, a set of eigenfaces can be called a set of describe basis for recognizing images.

Introduction

In order to understand preliminarily the basic principle of human face recognition by computer and the role of the eigenfaces is played in this process, this experiment is divided into six sections and the software \emph{matlab} is utilized. Before the lab, there some requisite operations are necessary to be manipulated. To download the file named \emph{data\\_for\\_labC.mat} into the \emph{Matlab} and then to inspect the variables\emph{eigenfaces\_blk} and \emph{employees\_DB}. The former contains 101 parameters and the first 100 of them are the eigenfaces due to their face-like appearance if they are displayed as images. The last parameter of the variable \emph{eigenfaces\_blk} is actually redundant in this lab, which will be explained next in the following paper. The variable \emph{employees\_DB} is a struct that contain the ID numbers of the 100 employees and the corresponding weights describing the faces of these employees. The task of each section is listed as below.

1. Task1: To verify that these eigenface in the variable \emph{eigenfaces\_blk} are orthogonal.
2. Task2: With the given egienfaces, the weights of the input image is found by projecting it on the entire collection of the egienfaces (the training faces). Since the input image projecting on each eigenface will generate a weight value, there will gain a set of weights that is the same size with the number of eigenfaces. There needs to write a \emph{Matlab} function to gain the weights of a face. The declaration of the function should be like:

$Function [weight\\_of\\_face, mean of\_face]=get\\_face\\_weights(im,eigenfaces\_blk)$

Where \emph{mean\\_of\\_face} is the average intensity of the image \emph{im}. Using this function to obtain the weights of the image \emph{find\\_id.jpg}. Then to plot the weights and comment whether the plot convey any information of the employee.

1. Task3: To write a \emph{Matlab} function the generate the image of the corresponding employee by using the weights of the image \emph{find\\_id.jpg} gained from the last task. The declaration of this function should be like:

$function [im]=generate\\_face\\_from\\_weights(weights\\_of\\_face, eigenfaces\_blk, mean\\_of\\_face)$

Finally, to comment the why the 100 eigenfaces are enough to describe an image of a face which has 450\times300 pixels.

1. Taks4: To write a \emph{Matlab} function to find the ID number of an employee by his/her images to achieve the aim of face recognition. The main idea of this task is divided into two steps. The first is to gain the weights of the input image by using the function \emph{ get\\_face\\_weights}. The second step is to compare the Euclidean distance between the weights of the input image and the weights of different employees in the database. When the distance is smallest, the corresponding weights in the database and the weights of the input image can be regarded as belonging to same one face image. The declaration of the function should be like:

$function [ID]=get\\_employees\\_ID\\_from\\_DB(im, employees\_DB,…)$

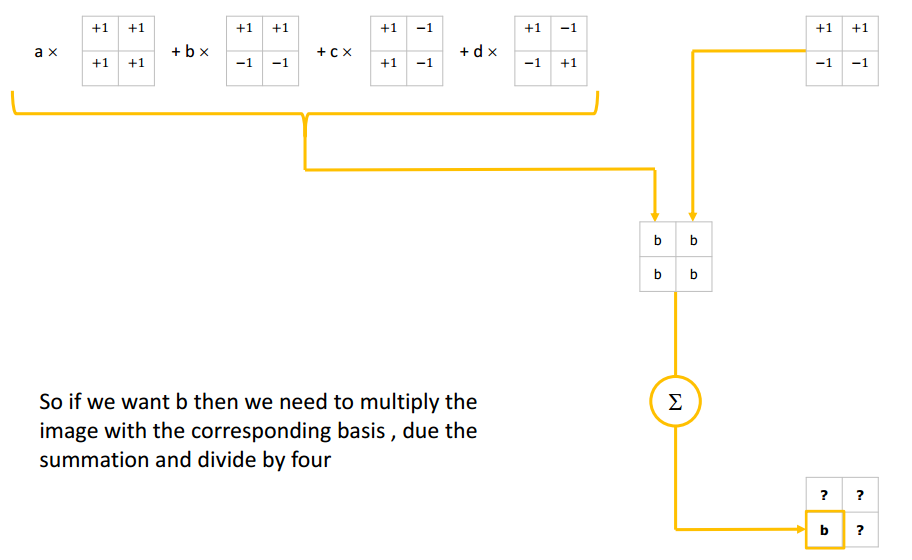
Then to utilize this function to gain the ID number of the image \emph{find\\_id\\_jpg}.

1. Task5: To access the robustness of the face recognition by using the function write in the previous tasks. As for the same person, the different face images can be gained since the different conditions and a set of weights only used to describe a specific image. Hence, the input image may be different with the images described by the weights in the database. Whether the function \emph{get\\_employees\\_ID\\_from\\_DB} still can recognize the images and find the ID number of the employees are needed to be checked. In this task, the two type of noises, \emph{AWGN} and \emph{salt \&pepper}are added on the original image \emph{find\\_id.jpg}. The recognition results of five standard deviations of the \emph{AWGN} are necessary to be compared and that of five noise densities of the \emph{salt \&pepper} noise are also needed to be tested.
2. To discuss the advantages and the disadvantages of having a larger eigenfaces space and summarize the finding concerning the robustness of the eigenface-based image recognition algorithm.

Methodology \& Results

Task1:

In order to verify that the eigenfaces are orthogonal, the knowledge of image transform can be inspired to apply in this task. The coefficients of the transform bases which are orthogonal are gained according the algorithm displayed as below.



In there, the eigenfaces are the bases for describing images and the weights are the coefficients in above figure. Hence, the value of the weights can be found by the following equations:

face(j)=\sum\_{i=1}^101 employees\_DB(j).weights(i).\* eignfaces\_blk(:,:,i)

where the variable $j$ is the number of the employees and $i$ is the number of 101 values of weights. \textbf{If the eigenfaces are orthogonal like in the above figure, the following equation will be satisfied}.

employees\_DB(j).weights(i)=mean{face(j).\* eignfaces\_blk(:,:,i)}

Hence, this equation can be utilized to verify the orthogonalization of these eigenfaces. The \emph{Matlab} code is attached as below.

clear;

load data\_for\_labC.mat;

for j=1:1:100

a=zeros(450,300);

for i=1:1:101 %--to find the image data 'a' of every employee

a=a+employees\_DB(j).weights(i).\*eignfaces\_blk(:,:,i);

end

%since these egienface are orthogonal

b=a.\*eignfaces\_blk(:,:,1);

%actually, c(j) is the first value of the jth weights in the database

c(j)=round(mean(mean(b)),4);

d(j)=round(employees\_DB(j).weights(1),4);

end

result=isequal(c,d)

In the equation \ref{}, the value of $i$ is picked as 1, hence , the matrix \emph{c} will hold the same values with first values of all weights in the database. In order to compare conveniently, the first value of all weights in the database are extracted and put into the matrix \emph{d}. If the matrixes \emph{c} and \emph{d} are equal, the equation ref{} will hold.

Running the \emph{Matlab} code, the command window is displayed as below:



Since the function \emph{isequal} returns to 1, which indicates the matrixes \emph{c} and emph{d} are equal. Then the equation \ref{} is correct and the eigenfaces are orthogonal.

Task 2:

The function \emph{get\\_face\\_weights} is written by applying the equation \ref{} as below:

function [weights\_of\_face,mean\_of\_face]=get\_face\_weights(im,eigenfaces\_blk)

weights\_of\_face=zeros(1,101);

im=double(im);

for i=1:1:101

mean\_of\_face=mean(mean(im));

weights\_of\_face(i)=mean(mean(im.\*eigenfaces\_blk(:,:,i)));

end

The \emph{Matlab} code of this task is attached as below:

clear;

im=imread('find\_id.jpg');

load data\_for\_labC.mat;

eigenfaces\_blk=eignfaces\_blk;

[weights\_of\_face,mean\_of\_face]=get\_face\_weights(im,eigenfaces\_blk);

figure;

plot(weights\_of\_face);

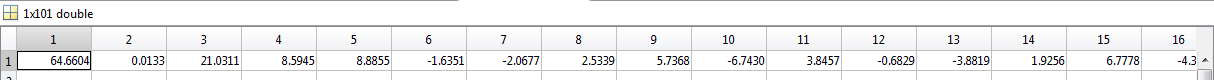
title('The weights of the input image');

ylabel('Weights values');

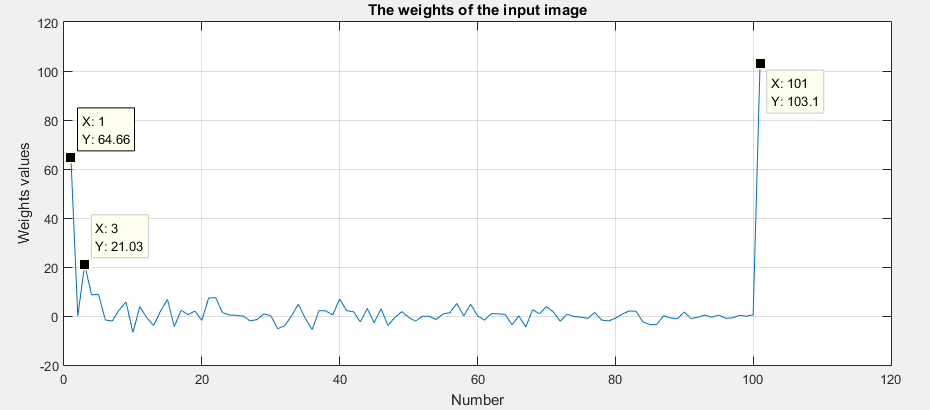
xlabel('Number');

grid on;

Running the code and the weights of the image \emph{find\\_id.jpg} is found in the \emph{Workspace} of the \emph{Matlab}. The partial screenshot is attached as below ( since the limited width of the paper).



The weights of the image \emph{find\\_id.jpg} is depicted as below.



From the figure \ref{}, the weights values corresponding to the 1th, 3th and 101th eigenfaces are larger than the other weights values of corresponding eigenfaces, which indicates the input image contains more details and information of 1th, 3th and 101th eigenfaces.

Task3:

The \emph{Matlab} function \emph{generate\\_face\\_from\\_weights} is written by using the formula \ref{}.

function[im]=generate\_face\_from\_weights(weights\_of\_face,eigenfaces\_blk, mean\_of\_face)

im=zeros(450,300);

for i=1:1:101

im=im+weights\_of\_face(i).\*eigenfaces\_blk(:,:,i);

end

im=uint8(im);

The code of this task is attached as below.

clear;

im=imread('find\_id.jpg');

load data\_for\_labC.mat;

eigenfaces\_blk=eignfaces\_blk;

[weights\_of\_face,mean\_of\_face]=get\_face\_weights(im,eigenfaces\_blk);

imre= generate\_face\_from\_weights(weights\_of\_face,eigenfaces\_blk, mean\_of\_face);

subplot(121);

imshow(im);

title('The original image');

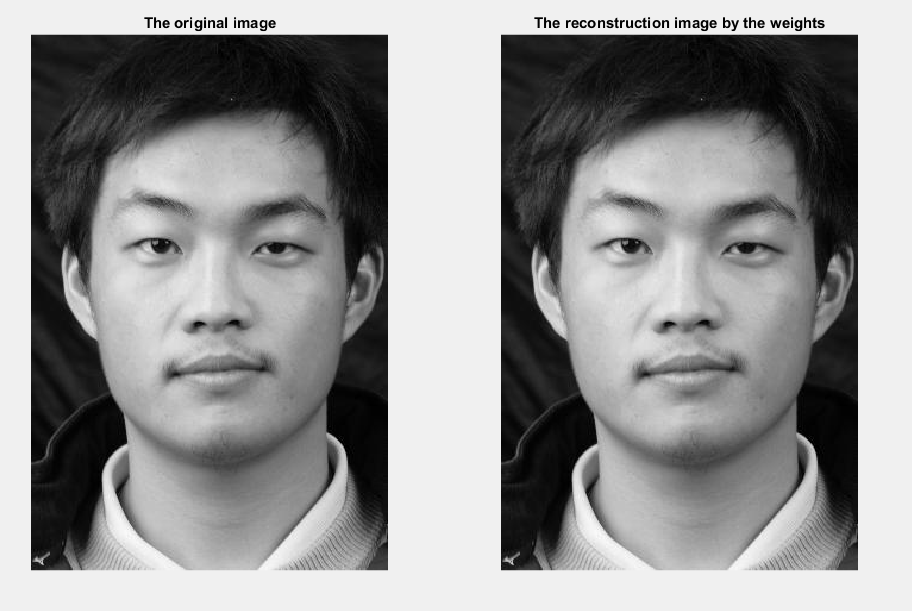
subplot(122);

imshow(imre);

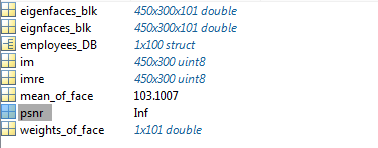
title('The reconstruction image by the weights');

psnr=calculate\_psnr(im,imre);

Running the code and obtain the original input image and reconstruction image as below.



Certainly, the quality of the reconstruction image is measured by using the metric \emph{PSNR}. The \emph{PSNR} value between the original image and the reconstruction image is found in the \emph{Workspace} shown below.



The \emph{PSNR} value is infinite, hence, the quality of the reconstruction image is extremely large.

According the citation \cite{1}, the size of a training image is 450\times300 pixels which interprets the eigenfaces space should be 450\times300 dimensions. Nevertheless, the paper introduces another reduced alternative solution for construct the eigenfaces space. If the number M of training images is smaller than the size N\times N of the training image, $M-1$ eigenfaces is enough for constructing the face images. In this task, the number of the training images is 100 and the size is 450\times300 for all training images. \textbf{Hence, 99 eigenfaces are enough for describe an image. Surely, the 100 eigenfaces is also can describe an image. The drawback of using 100 eigenfaces to describe an image is that the describe bases are not simplest and will produce redundancies.}

Task4:

In this task, the concept of the Euclidean distance is applied. In order to find the weights having smallest difference with the weights of the input image \emph{find\\_id.jpg} in the database, the Euclidean distance is utilized to describe the difference between two weights and the formula is given as below.

Of course, the function \emph{get\\_face\\_weights} written before is utilized at here to gain the weights of the input image. The function \emph{get\\_employees\\_ID\\_from\\_DB} is written as below.

function [ID] = get\_employees\_ID\_from\_DB (im, employees\_DB,eigenfaces\_blk)

[weights\_of\_face,mean\_of\_face]=get\_face\_weights(im,eigenfaces\_blk);

a=norm(weights\_of\_face-employees\_DB(1).weights);%--assuming the nearest distance

for i=1:1:100

b=norm(weights\_of\_face-employees\_DB(i).weights);

c=b;

if (a>=c)%--update the nearest distance

a=c;

j=i;

end

end

ID=employees\_DB(j).id;

The \emph{Matlab} script is written as blow:

clear;

load data\_for\_labC.mat;

im=imread('find\_id.jpg');

eigenfaces\_blk=eignfaces\_blk;

ID= get\_employees\_ID\_from\_DB (im, employees\_DB,eigenfaces\_blk)

Running the code and the result is displayed in the command window exhibited in the Figure \rfe{}.



\textbf{Hence, the employee’s ID number of the image \emph{find\\_id.jpg} is 96.}

Task5:

In this task, the main idea is to add different noises on the original image \emph{find\\_id.jpg} then to utilizing the function \emph{get\\_employees\\_ID\\_from\\_DB} written in the task 4 to recognize the noisy images. If the ID number is correct, the recognition algorithm still has the robust ability to identify the image. Otherwise, the recognition algorithm will be invalid.

a). For the \emph{AWGN}, the standard deviations are chosen as 5, 10, 60, 100 and 150 to construct five noisy images the \emph{Matlab} is attached as below:

clear;

im=imread('find\_id.jpg');

load data\_for\_labC.mat;

eigenfaces\_blk=eignfaces\_blk;

sigma=[5 20 60 100 150];

for i=1:1:5

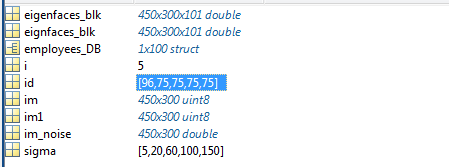
im\_noise=wgn(450,300,sigma(i).^2);

im1=uint8(double(im)+double(im\_noise));

id(i)= get\_employees\_ID\_from\_DB (im1, employees\_DB,eigenfaces\_blk);

end

After running the code, the recognized ID numbers of the five noisy images are found in the matrix \emph{id} in the \emph{Workspace}.



Obviously, the ID number 96 of the first noisy image with the standard deviation 5 can be recognized out correctly and the other noisy image cannot be recognized. The recognized situations is displayed by the Table \ref{}.

Table1

b). For the \emph{salt \&pepper} noise, the noise densities are chosen as 0.01, 0.2, 0.5, 0.7 and 0.9 to form five different noise the \emph{Matlab} is attached as below:

clear;

im=imread('find\_id.jpg');

load data\_for\_labC.mat;

eigenfaces\_blk=eignfaces\_blk;

percentage=[0.01 0.2 0.5 0.7 0.9];

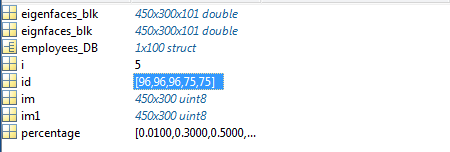
for i=1:1:5

im1=imnoise(im,'salt & pepper',percentage(i));

id(i)= get\_employees\_ID\_from\_DB (im1, employees\_DB,eigenfaces\_blk);

end

After running the code, the ID number of the noisy image can be found in the \emph{Workspace} shown as below.



It is apparent when the noise densities of the \emph{salt\&pepper} are 0.7 and 0.9, the recognition results 75 are incorrect. When the noise densities are 0.01, 0.3 and 0.5, the correct ID number 96 can be found the noisy image can be recognized. The recognized situations is displayed by the Table \ref{}.

Table2

Task6:

If the eigenfaces space is larger, there will be some advantages and disadvantages:

1). Advantage

When the eigenfaces space is become larger-e.g., from 100 to10000, there will need more training images to construct the eigenface space, which will cause that \textbf{more face images can be described and recognized} by the eigenfaces space.

2) Disadvantages

a). Larger eigenfaces space means there needs \textbf{larger memory} to store the more eigenfaces for the computer.

b). Since the size of the weights is same with the number of the eigenfaces, larger eigenfaces space will leads the larger size of weights. Hence, the requirement for the computer cache is higher.

c). The larger eigenfaces space and the larger size of weights will cause the computationally speed to be slower and will cost more time to recognize a face image and the efficiency will become lower.

Findings for the robustness of the eigenface-based image recognition algorithm:

1). As for the \emph{AWGN}, the algorithm actually has a \textbf{ threshold value} to identify whether the compared images belong to one person or not. When the Euclidean distance is smaller than this threshold value, the compared images belong to the same person. When the Euclidean distance is larger than this threshold value, the algorithm will produce an incorrect identify and gain a wrong result. In this task, this threshold value can be represented by the standard deviation of the \emph{AWGN} and is located in the interval of 5 to 20 of the standard deviation.

2). As for the \emph{salt\&pepper} noise, this threshold value can be represented by the noise density of the noise and is located in the interval of 0.5 to 0.7 of the noise density.

@ARTICLE{6793549,   
author={M. Turk and A. Pentland},   
journal={Journal of Cognitive Neuroscience},   
title={Eigenfaces for Recognition},   
year={1991},   
volume={3},   
number={1},   
pages={71-86},   
doi={10.1162/jocn.1991.3.1.71},   
ISSN={0898-929X},   
month={Jan},}