Name: Wei Zhifeng
Matriculation No：G2002825F

EE6403 Distributed Multimedia Systems

Assignment 1

Instructions:

1. Submit only the **softcopy pdf file** through NTULearn EE6403 course site under the Assignment tab by **5 March 2021**.
2. Name your submitted file as FullStudentName_MetriculationNo.pdf (e.g., Tan_Yi_G1234567A.pdf)
3. Write your full name (as in student card) and matriculation no. clearly on the front page.
4. Be concise and to-the-point in your answers. Avoid long and irrelevant answers.
5. Draw boxes or highlight your final numerical answers (e.g., like this 88 or 88) to facilitate marking.
6. This is an individual home assignment. Do not plagiarize.
7. Completed assignment can include printouts of source codes and figures, if applicable.
8. You can use MATLAB or other programming platforms to solve the problems, unless stated otherwise in the questions.
9. For calculation problems, you can choose to write your answers on the papers first, and then scan / convert them into pdf format.

1. Two-dimensional Discrete Cosine Transform (2D-DCT) is a transform employed in the JPEG standard. The 2D-DCT of an $N \times N$ image block is given by:

$$S_{uv} = \alpha(u)\alpha(v)\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} s_{ij} \cos\frac{(2i+1)u\pi}{2N}\cos\frac{(2j+1)v\pi}{2N} \qquad u, v = 0, ..., N-1$$

where

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{N}} & \text{for } k = 0 \\ \sqrt{\frac{2}{N}} & \text{for } k = 1, 2, ..., N-1 \end{cases}$$

Compute **manually** the 2D-DCT of the following image block **A**. Verify your result using MATLAB / other programming platforms.

$$\mathrm{A} = \begin{bmatrix} 20 & 20 & 10 & 10 \\ 20 & 20 & 10 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

1. A is a 4×4 matrix

$$S_{uv} = \alpha(u)\sum_{i=0}^{3}\cos\frac{(2i+1)u\pi}{8}\cdot\alpha(v)\sum_{j=0}^{3}S_{ij}\cos\frac{(2j+1)v\pi}{8}$$

I. $$F_{iv} = \alpha(v)\left\{S_{i0}\cos\frac{v\pi}{8} + S_{i1}\cos\frac{3v\pi}{8} + S_{i2}\cos\frac{5v\pi}{8} + S_{i3}\cos\frac{7v\pi}{8}\right\}$$

$$= \alpha(v)\left\{(S_{i0} + (-1)^v S_{i3})\cos\frac{v\pi}{8} + (S_{i1} + (-1)^v S_{i2})\cos\frac{3v\pi}{8}\right\}$$

$i=0$: $F_{00} = \frac{1}{2}\{(20+10)+(20+10)\} = 30$

$F_{01} = \frac{1}{\sqrt{2}}\{10\cos\frac{\pi}{8} + 10\cos\frac{3\pi}{8}\} = 9.239$

$F_{02} = \frac{1}{\sqrt{2}}\{30\cos\frac{\pi}{4} + 30\cos\frac{3}{4}\pi\} = 0$

$F_{03} = \frac{1}{\sqrt{2}}\{10\cos\frac{3\pi}{8} + 10\cos\frac{9}{8}\pi\} = -3.827$

$i=1$: the same as 1st row.

$i=2; i=3$: $F_{iv} = 0$

$$\therefore F_{iv} = \begin{bmatrix} 30 & 9.239 & 0 & -3.827 \\ 30 & 9.239 & 0 & -3.827 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

II. $$S_{uv} = \alpha(u)\left\{(F_{0v} + (-1)^v F_{3v})\cos\frac{u\pi}{8} + (F_{1v} + (-1)^u F_{2v})\cos\frac{3u\pi}{8}\right\}$$

$V=0$ $S_{00} = \frac{1}{2}\{30+30\} = 30.$

$S_{10} = \frac{1}{\sqrt{2}}\{30\cdot\cos\frac{\pi}{8} + 30\cos\frac{3\pi}{8}\} = 27.716$

$S_{20} = \frac{1}{\sqrt{2}}\{30\cos\frac{\pi}{4} + 30\cos\frac{3\pi}{4}\} = 0$

$S_{30} = \frac{1}{\sqrt{2}}\{30\cos\frac{3\pi}{8} + 30\cos\frac{9}{8}\pi\} = -11.481$

$$\begin{bmatrix} 30 \\ 27.716 \\ 0 \\ -11.481 \end{bmatrix}$$

$V=1$ $S_{01} = \frac{1}{2}\{9.239 + 9.239\} = 9.239$

$S_{11} = 8.536$

$S_{12} = 0$

$S_{13} = -3.536$

$$\begin{bmatrix} 9.239 \\ 8.536 \\ 0 \\ -3.536 \end{bmatrix}$$

$V=2$: $\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

$V=3$: $S_{30} = \frac{1}{2}\{-3.827 - 3.827\} = -3.827$

$S_{31} = -3.536$

$S_{32} = 0$

$S_{33} = 1.465$

$$\therefore S_{uv} = \begin{bmatrix} 30 & -9.239 & 0 & -3.827 \\ 27.716 & 8.536 & 0 & -3.536 \\ 0 & 0 & 0 & 0 \\ -11.481 & -3.536 & 0 & 1.465 \end{bmatrix}$$

matlab:

$$\begin{bmatrix} 30 & 9.2388 & 0 & -3.8268 \\ 27.7164 & 8.5355 & 0 & -3.5355 \\ 0 & 0 & 0 & 0 \\ -11.4805 & -3.5355 & 0 & 1.4645 \end{bmatrix}$$

2. Karhunen-Loeve Transform (KLT) is used in an image compression scheme. A 6×6 image is given as follows:

$$\mathrm{B} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 2 \\ 1 & 1 & 2 & 2 & 1 & 2 \\ 1 & 1 & 2 & 4 & 4 & 4 \\ 1 & 2 & 2 & 3 & 4 & 4 \\ 1 & 2 & 3 & 3 & 4 & 4 \end{bmatrix}$$

(a) Partition the image into 9 2×2 blocks (subimages), and order them lexicographically to form 9 4×1 column vector (**x**). Calculate the mean vector, and show that its covariance matrix is given by:

$$\mathbf{C} = \begin{bmatrix} 1.6944 & 1.7361 & 1.4028 & 1.4167 \\ 1.7361 & 2.0278 & 1.6944 & 1.6667 \\ 1.4028 & 1.6944 & 2.1111 & 1.9167 \\ 1.4167 & 1.6667 & 1.9167 & 2.2500 \end{bmatrix}$$

(b) Determine the corresponding eigenvectors and eigenvalues for the covariance matrix **C** using MATLAB / other programming platforms.

(c) A 2×2 image block after zero-mean centering is given by:

$$\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$$

Calculate the total squared error of the reconstructed image if the image block is approximated using two dominant principal components.

(d) **(Advanced)**
Write a program (e.g., MATLAB) to perform image compression using KLT on a common grayscale image such as Lena. Plot the reconstruction error of the image against the number of retained principal components (coefficients). Your answers should also discuss issues such as the chosen block size, the suitable number of principal components used for compression, and any other related issues.

3. Choose an emerging media application/issue and briefly discuss its importance and impact. You should keep your answer concise (preferably in point form) and **less than half A4 page length**. Note that this is an open-ended question, and you can choose any application/issue that you feel is important. However, you need to explain and justify its relevance/importance clearly. You should not copy or plagiarize answers from some sources, but rather research, understand, and explain in your own words.

2. (a)

$$x = \left[\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \\ 2 \\ 4 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \\ 4 \\ 4 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 3 \\ 3 \\ 3 \end{bmatrix}, \begin{bmatrix} 4 \\ 4 \\ 4 \\ 4 \end{bmatrix} \right]$$

$$\bar{x} = \begin{bmatrix} 1.22 \\ 1.56 \\ 1.89 \\ 2.33 \end{bmatrix}$$; Caculate the covariance matric by matlab

$$C = \begin{bmatrix} 1.6944 & 1.7361 & 1.4028 & 1.4167 \\ 1.7361 & 2.0278 & 1.6944 & 1.6667 \\ 1.4028 & 1.6944 & 2.1111 & 1.9167 \\ 1.4167 & 1.6667 & 1.9167 & 2.2500 \end{bmatrix}$$

(b) $\lambda_1 = 6.9621$ $V_1 = [0.4464, 0.5109, 0.5147, 0.5242]$

$\lambda_2 = 0.7577$ $V_2 = [0.5811, 0.4541, -0.4013, -0.5433]$

$\lambda_3 = 0.2634$ $V_3 = [-0.1715, 0.0837, 0.7319, -0.6541]$

$\lambda_4 = 0.1001$ $V_4 = [0.6585, -0.7251, 0.1958, -0.0464]$

(c) given block $x = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

Using 2 dominant principal components : V_1, V_2

$$\hat{x} = \sum_{i=1}^{2} C_i V_i$$

$C_i = V_i^T x$ $\therefore$ $C_1 = V_1^T \cdot x = 1.5498$

$C_2 = V_2^T \cdot x = -0.4905$

$\hat{x} = \begin{bmatrix} 0.4068 \\ 0.5690 \\ 0.9945 \\ 1.0790 \end{bmatrix}$

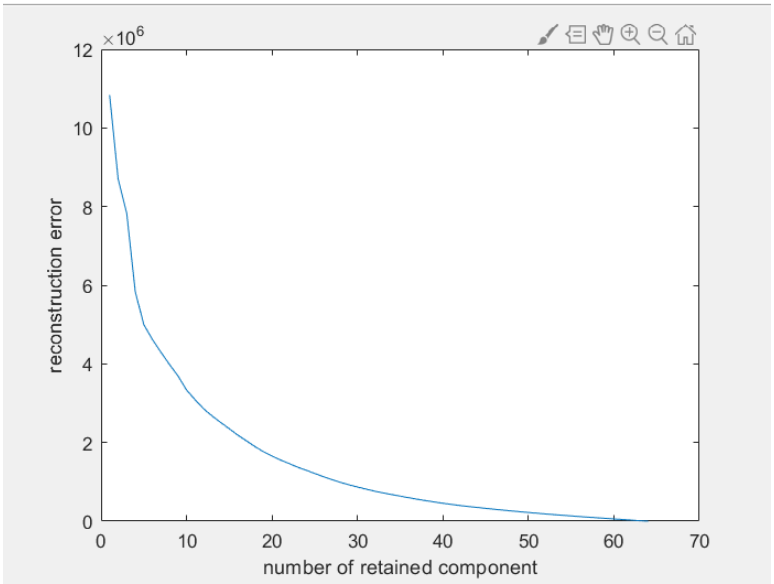
$\|x - \hat{x}\|^2 = 0.3575$

(d) using matlab to perform image compression:

Origin picture

Origin image: greyscale image-Lena
Size:512*512
Partition block:8*8

There will be 64 eigenvectors and 64 eigenvalue; the reconstruction error of the image against the number of retained principal components is shown below; when the retain component is more than 20 ,the image keep most of the information and doesn't have obvious distortion .



When the retained component became lesser, the reconstructed image had distortion.as shown below:

Retain 64 components

Retain 40 components

Retain 20 components

Retain 5 components

Retain 3 components

Retain 1 components

Discussion:

i. Suitable number of component:

The eigenvalue is shown below:

128792.933432640	7519.76115380978	2819.20222644235	1752.49453395914
1263.32190261401	644.984235854247	482.287814600074	422.665962250945
378.431727698935	336.608894976706	244.944346623354	204.594972999419
156.067775611174	141.242416466733	127.875452528290	114.176568294778
100.512888359471	92.3062863799178	84.9421453064514	67.0299031453966
60.6699697525434	51.2065381073888	48.1148808154198	44.4806571846890
43.7058879581519	41.9276170185849	39.9190256053626	33.8367378970478
32.4597646397793	27.6833324520258	25.8053532825357	25.5517339370349
21.6688599292538	21.0400320617971	19.5008992411366	19.3573513826830
18.9173287513911	17.1778277500287	16.6747958233711	15.2906039190978
14.5369088564235	13.8678003605868	11.8879431552308	11.4981987093097
11.1644443161957	10.9054403824091	10.3485564092491	10.1922735577670
9.82480961354050	9.47846149500899	9.06565818339703	8.94875686769245
8.65834279773830	8.03186146210701	7.92323979744253	7.84232802979282
7.35817884196669	7.00575508743135	6.90037978724045	6.63028747178071
6.41087372087252	6.32212308056162	5.87438823423504	5.34161512963573

The larger eigenvalue it is, the more information the corresponding eigenvector has. Therefore, if we drop some of the eigenvector with smaller eigenvalue, the image will not have explicit distortion.

And if we drop more eigenvector, the compression rate will be higher (the image should not have distortion).

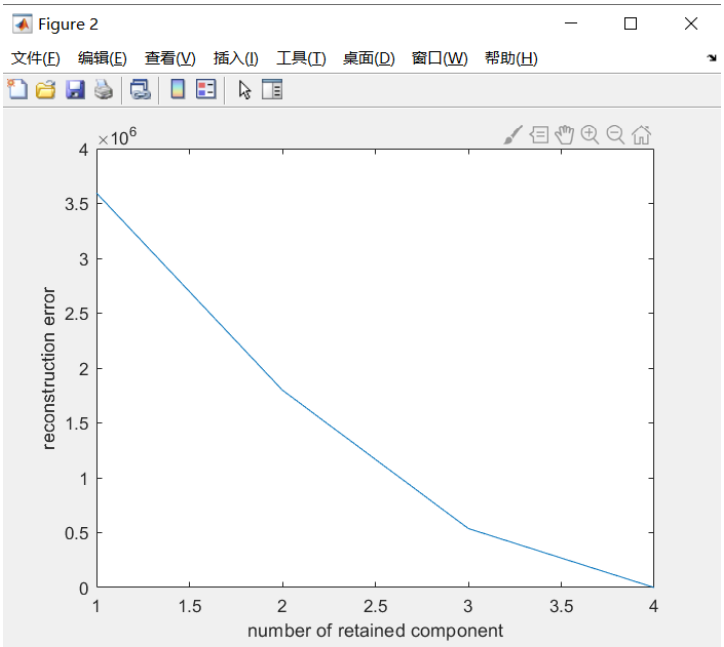
For this picture, we can even keep 12 to 24 retained principle components.

ii. Block size

The chosen block size is 8*8. Cause JPEG partitions the image to 8*8 block, so I also use 8*8 block size.

if the block size is too small, the number of eigenvectors and eigenvalue is small. So the component we can drop is less, what's more, the largest eigenvector contains most of the information and the compression rate will be low. And maybe a larger block size could have more detail.

Take 2*2block for example. The reconstruction error of image (partition block is 2*2) is shown below.



The first component keeps the most information of the image, even if we keep the first component, the image doesn't have obvious distortion. So the compression rate might not be high.

Retain 1 components

3.Discussion about an emerging media

Media is an intermediary of conveying information, such as image, video, audio, smell, touch and so on. Among these, streaming media is an emerging media, which has many advantage and impacts on our daily life. Streaming media is media that continuously received by and presented to end-user while being delivered by a provider.

Advantage and Importance:

- Real-time: you can look through the information at the same time when you are receiving data. You don't have to download or transmit the entire file before, it is convenient and time-saving.
- High-quality: you can change the transmission format depends on network condition. Due to the faster network, we can enjoy high-quality video and image through streaming media. If the network condition changes, you can change the transmission format but not stop receiving data.
- Distributed storage: The data can be stored in different server and transmit to different user. people can share data and information anywhere.
- General-format: nearly all file format can be transmitted through this media.

Impacts:

- This media changes our concept of data transmission, we don't have to process and look through the data after it is entirely downloaded. We can gain data anytime and anywhere, we don't have to bring many hardcopy with us.
- This media has impacts on conventional storage media, cause this new media is convenient and cheap and has higher storage capacity.
- This media requires faster network and larger bandwidth, which push the development of other technologies.

Appendix

```
%Question 1
%A is the input image blockA
A=[20 20 10 10;
    20 20 10 10;
    0 0 0 0;
    0 0 0 0];
%use dct2 to perform the 2D-DCT transform of this
image
B=dct2(A);

%Question 2
%matric is the input 6*6 image B
matric=[0 0 0 0 0 0;
    0 0 1 1 1 2;
    1 1 2 2 1 2;
    1 1 2 4 4 4;
    1 2 2 3 4 4;
    1 2 3 3 4 4];

%2.(a)X is to store the partitioned 4*1 block
X=[];
%partition the image into 2*2 block and order them to
form 9 4*1 column vector
for i=1:2:size(matric)
    for j=1:2:size(matric)
        c=reshape(matric(i:i+1,j:j+1)',4,1);
        X=[X;c'];
    end
end
%calculate the mean vector of X
X_mean=mean(X);
%perform zero-mean centering and perform cov(x) to
compute the covariance matrix
x=(X-X_mean);
C=cov(x);

%2.(b)calculate the eigenvalue and eigenvector for
the covariance matrix C
%V is eigenvector ; D is eigenvalue
[V,D]=eig(C);
```

```
%2.(c) perform reconstruction on an 2*2 block
block=[0 1 1 1 ];

%factor=V'*block
retained_comp=2;
%drop some eigenvector and keep the retained
components
new_V=[zeros(size(V,1),size(V,2)-
retained_comp)';V(:,size(V,2)-
retained_comp+1:size(V,2))'];
%caculate factor ci; the value in corresponding
direction after compression
factor=block*new_V;
%reconstruct block
block_estimate=factor*new_V';
%calculate squared error using 2 dominant principal
component
error=sse(block-block_estimate);


%Question 2.(d)
%read source image -- lena
source_img=imread('source_img.bmp');

%X is to store partitoined block erroe is to store
the reconstruction error
%in each block
X=[];
error=[];

%an array to keep retain_component;
retain=(64:-1:1);
%retain=(4:-1:1)
len=size(source_img,1);

%partition the image to different block size (8*8 or
2*2)
for i=1:8:len
%for i=1:2:len
    for j=1:8:len
   % for j=1:2:len
       c=reshape(source_img(i:i+7,j:j+7)',64,1);
   %     c=reshape(source_img(i:i+1,j:j+1)',4,1);
        X=[X;c'];
```

```
    end
end

%perform zero-mean centering and caculate covariance
matrix
X=double(X);
X_mean=mean(X);
x=(X'-X_mean')';
C=cov(x);

%caculate eigenvalue and eigen vector
[V,D]=eig(C);

%keep different retained component ;
for i=1:length(retain)
    retained_comp=retain(i);
    new_V=[zeros(size(V,2)-
retained_comp,size(V,1));V(:,size(V,2)-
retained_comp+1:size(V,2))'];
    factor=X*new_V; %calculate ci factor
    X_estimate=factor*new_V';%compression
    tem_img=[];
    row=[];

    %reconstruct the image from X to tem_img
   for k=1:1:4096
   %  for k=1:1:65536
        block=reshape(X_estimate(k,:),8,8)';
   %     block=reshape(X_estimate(k,:),2,2)';
        if mod(k,64)==0
     %   if mod(k,256)==0
            %»»ÐÐ
            row=[row block];
            tem_img=[tem_img;row];
            row=[];
        else
            row=[row block];
        end
   end
   %show the reconstruct image
    img=uint8(tem_img);
    imshow(img);
    retained_comp
    %calculate the reconstruction error
```

```
    %error=[error,sse(img-source_img)];
    t=sum(sum((source_img-img).^2))
    error=[error,t];
end
figure(2);
%draw the reconstruction error
plot(retain,error);
xlabel('number of retained component');
ylabel('reconstruction error');
```