****

**Course Information**

Course Title: Digital Image Processing

Section: 1

Course Instructor: Dr. Ahmed Wasif Reza

Associate Professor

Department of Computer Science & Engineering

**Lab-05**

**Student’s Information**

**Name:** Mujahidul Islam

**ID:** 2019-2-60-072

**Department:** Computer Science & Engineering

**Date of Submission: 24 November 2022**

**Question 1**

Apply Fourier transform to transform an image from spatial domain to frequency domain. Apply inverse Fourier transform to transform the image from frequency domain to spatial domain.

Code:

img = imread('MRI.png');

img2 = double(mat2gray(img));

F = fftshift(fft2(img2));

I = log(1+abs(F));

imginv=abs(ifft2(F));

subplot(2,2,1);imshow(img); title('Original image');

subplot(2,2,2);imshow(I); title('Fourier transform of image');

subplot(2,2,3);bar(I); title('Frequency bar graph');

subplot(2,2,4);imshow(imginv); title('Inverse Fourier transform of image');

Output:

Graphical user interface

Description automatically generated

**Question 2**

Compress the image using Fourier Transform.

Code:

a=imread('MRI (1).png');

[row col] = size(a);

A=fft2(a);

count\_pic=2;

subplot(2, 2, 1);imshow(a); title('Original image');

for thresh=0.1\*[0.001 0.005 0.006]\*max(max(abs(A)))

ind=abs(A)>thresh;

count=row\*col-sum(sum(ind));

Alow=A.\*ind;

per=100-count/(row\*col)\*100;

Blow=uint8(ifft2(Alow));

subplot(2,2,count\_pic);

imshow(Blow);

count\_pic=count\_pic+1;

title([num2str(per) '% of fft basis'])

end

Output:

Graphical user interface

Description automatically generated

**Question 3**

Apply discrete cosine transform to transform an image from spatial domain to frequency domain. Apply inverse discrete cosine transform to transform the image from frequency domain to spatial domain.

Code:

image = imread('MRI.png');

image = im2gray(image);

compress = dct2(image);

imshow(log(abs(compress)),[])

compress(abs(compress) < 10) = 0;

K = idct2(compress);

K = rescale(K);

figure,montage({image,K});title('Original Image After DCT ');

inverseDct=abs(idct2(image))

figure,imshow(inverseDct);title('Inverse DCT Trasnform');

Output:

Graphical user interface

Description automatically generated

**Question 4**

Compress the image using Discrete Cosine Transform.

Code:

a=imread('MRI.png');

image = im2gray(a);

compress = dct2(image);

compress(abs(compress) < 10) = 0;

K = idct2(compress);

K = rescale(K);

subplot(221);imshow(a); title('Original image');

subplot(222);imshow(K); title('DCT Compress image');

Output:

Graphical user interface, text

Description automatically generated

**Question 5**

Compress the image using Haar Transform.

Code:

image\_name = 'MRI.png';

delta = 0.0001;

disp(delta)

if (delta>1 || delta<0)

error('harr\_wt: Delta must be a value between 0 and 1');

end

H1=[0.5 0 0 0 0.5 0 0 0;0.5 0 0 0 -0.5 0 0 0;0 0.5 0 0 0 0.5 0 0 ;0 0.5 0 0 0 -0.5 0 0 ;0 0 0.5 0 0 0 0.5 0;0 0 0.5 0 0 0 -0.5 0;0 0 0 0.5 0 0 0 0.5;0 0 0 0.5 0 0 0 -0.5;];

H2=[0.5 0 0.5 0 0 0 0 0;0.5 0 -0.5 0 0 0 0 0;0 0.5 0 0.5 0 0 0 0;0 0.5 0 -0.5 0 0 0 0;0 0 0 0 1 0 0 0;0 0 0 0 0 1 0 0;0 0 0 0 0 0 1 0;0 0 0 0 0 0 0 1;];

H3=[0.5 0.5 0 0 0 0 0 0;0.5 -0.5 0 0 0 0 0 0;0 0 1 0 0 0 0 0;0 0 0 1 0 0 0 0;0 0 0 0 1 0 0 0;0 0 0 0 0 1 0 0;0 0 0 0 0 0 1 0;0 0 0 0 0 0 0 1;];

H1o = (H1.\*(2^0.5));

H2o = (H2.\*(2^0.5));

H3o = (H3.\*(2^0.5));

Ho=normc(H1o\*H2o\*H3o);

H = H1\*H2\*H3;

x=double(imread(image\_name));

len=length(size(x));

if len~=2

error('harr\_wt: Input image must be a grey image, use "haar\_wt\_rgb" function to compress RGB Images');

end

yo = zeros(size(x));

y = zeros(size(x));

[r,c]=size(x);

for i=0:8:r-8

for j=0:8:c-8

p=i+1;

q=j+1;

yo(p:p+7,q:q+7)=(Ho')\*x(p:p+7,q:q+7)\*Ho;

y(p:p+7,q:q+7)=(H')\*x(p:p+7,q:q+7)\*H;

end

end

figure;

imshow(x/255);

n1=nnz(y);

zo=yo;

m=max(max(yo));

yo=yo/m;

yo(abs(yo)<delta)=0;

yo=yo\*m;

z=y;

y=y/m;

y(abs(y)<delta)=0;

y=y\*m;

n2=nnz(y);

for i=0:8:r-8

for j=0:8:c-8

p=i+1;

q=j+1;

zo(p:p+7,q:q+7)=Ho\*yo(p:p+7,q:q+7)\*Ho';

z(p:p+7,q:q+7)=inv(H')\*y(p:p+7,q:q+7)\*inv(H);

end

end

figure;

subplot(121);

imshow(x/255);

title("original image");

subplot(122)

imshow(z/255);

title("compressed image");

imwrite(x/255,'orginal.tif');

imwrite(z/255,'compressed.tif');

compression\_ratio = n2/n1;

Output:

Graphical user interface

Description automatically generated