Assignment #4

Due date: April 24 (Tue.)

Submission

E-mail a zip file including the source codes, a report and test images to TA. An execution m-file should be included in source codes. **You should follow the structure of the execution m-file which is given in the Appendix at the end of this document.**
Name the filename as student-idn_name.zip (e.g. 20183000_kdhong.zip).
TA's e-mail address is **ltk010203@kaist.ac.kr**

Due date: **04/24 23:59**. (Refer to the delay policy in the web site)

Test images in the web site:
Color_monarch_512x512.raw   Gray_monarch_512x512.raw

Notice

All the programming assignments are based on MATLAB. (Do not use any image processing function in MATLAB **except allowing functions.** However, **you can use basic functions available in the C standard library,** such as round, ceil, floor, rand, bitshift, sqrt, sum, exponential, log, trigonometric, and abs ones. **You can also use vectorization in your program**.) All source codes for submission should include comments.
Describe your work and analyze the corresponding results in the report. A proper length of the report is 10 pages of A4 size including figures. Report exceeding the recommended length will get a penalty. The report should include the followings.

1. Simple theoretical backgrounds & programming strategies
2. Result images
3. Analysis of the results

If a **copy version** is found, the score will be **zero** point without any exception.

Scoring policy: implementation (60), processing time (10), and report (30)

Available MATLAB functions in this assignment

conv2, psnr

1. # Image Transform

(for Gray_monarch_512x512.raw)

1. DFT

a. Apply the DFT using the fast algorithm (so-called FFT) to the given image and **display the magnitude in log scale and the phase of DFT coefficients,** respectively. All processes should be finished within 3 seconds.
b. Apply the inverse FFT to the transformed result and examine the **reconstructed image.** All processes should be finished within 3 seconds.
c. Apply the inverse FFT to only the magnitude of the transformed result and examine the **reconstructed image.**
d. Apply the inverse FFT to only the phase of the transformed result and examine the **reconstructed image.**
e. Discuss the results of c and d.

2. DCT

a. Perform the DCT for the given image and **display the magnitude in log scale. Perform the inverse DCT and verify the reconstructed image**. Each process should be finished within 15 seconds.
b. Divide the given image into 16x16 blocks and **apply 16x16 block DCT to each block**, and then **display the DCT coefficients in log scale**. **Perform the inverse DCT and verify the reconstructed image**. Each process should be finished within 3 seconds.
c. Compare the transformed results of a and b.

3. Discrete Hadamard transform (DHT)

Apply the DHT to the given image and **display magnitude in log scale**. **Perform the inverse DHT and verify the reconstructed image**.
Examine the reconstructed image and explain the efficiency of energy compaction.

4. Compression Efficiency of DFT, DCT, and DHT

Explain which transform can provide a better performance from the viewpoint of compression rate if we try to compress the transformed images.

5. Wavelet Transform

a. Perform 3-level wavelet transform for the given image.
b. Perform the inverse wavelet transform and examine the reconstructed image.
c. Describe the characteristics at each band.

2. Image Enhancement

(for Gray_monarch_512x512.raw and Color_monarch_512x512.raw)

1. Point operations

a. Histogram equalization

Implement the histogram equalization program and apply it to the **given gray image.**

In addition, implement histogram equalization programs for the RGB, HSI, and CMY coordinates, respectively, and apply them to the **given color image.** Compare the applied results obtained from three different coordinates.

b. Histogram modification

Apply the histogram modification to the given gray image by using the exponential distribution. ($v_{min} = 0$)

(Name input parameter α as 'alpha' at the beginning of the code.)

2. Spatial Domain Filtering

(for Gray_monarch_512x512.raw)

a. Gamma correction

Implement a Gamma correction algorithm.

User can select the value of γ.

(Hint: Intensity range of output image is bounded by [0, 255].)

(Name input parameter γ as 'gamma' at the beginning of the code.)

b. Spatial filtering
Add a salt & pepper noise and a Gaussian noise to **a gray image**. And apply the Mean filtering, median filtering, and directional smoothing to the noisy images with kernel sizes of 3 and 7, respectively.
Compare the PSNRs of the noisy images with those of the filtered images.
(Name output parameters of PSNR values as in the bottom table.)

PSNR	**Kernel 3**		**Kernel 7**	
	Salt & pepper noise	**Gaussian noise**	**Salt & pepper noise**	**Gaussian noise**
Mean filtering	`SP_3_mean`	`Gau_3_mean`	`SP_7_mean`	`Gau_7_mean`
Median filtering	`SP_3_med`	`Gau_3_med`	`SP_7_med`	`Gau_7_med`
Directional smoothing	`SP_3_dir`	`Gau_3_dir`	`SP_7_dir`	`Gau_7_dir`

3. Transform Domain Filtering
(for Gray_monarch_512x512.raw)

Add a Gaussian noise to the given gray image. And apply LPF and HPF in the DFT domain and the DCT domain, respectively. Analyze the inverse transformed images obtained from each domain.

Results

1.1 a. DFT: Magnitude, phase
1.1 b. DFT: Reconstructed image
1.1 c. The inverse FFT to only the magnitude of the transformed result.
1.1 d. The inverse FFT to only the phase of the transformed result.

1.2 a. DCT: magnitude, reconstructed image
1.2 b. DCT 16x16: the DCT coefficients, reconstructed image

1.3 DHT: magnitude, reconstructed image

1.5 a. DWT: 3 level DWT
1.5 b. DWT: reconstructed image

2.1 a. Histogram equalization
- Gray,
- RGB,
- HSI,
- CMY

2.1 b. Result image after applying input parameter α.

2.2 a. Result image after applying input parameter γ.

2.2 b.
- Salt & pepper noise, kernel 3, mean filtering
- Salt & pepper noise, kernel 7, mean filtering
- Gaussian noise, kernel 3, mean filtering
- Gaussian noise, kernel 7, mean filtering

- Salt & pepper noise, kernel 3, median filtering
- Salt & pepper noise, kernel 7, median filtering
- Gaussian noise, kernel 3, median filtering
- Gaussian noise, kernel 7, median filtering

- Salt & pepper noise, kernel 3, directional filtering
- Salt & pepper noise, kernel 7, directional filtering
- Gaussian noise, kernel 3, directional filtering
- Gaussian noise, kernel 7, directional filtering

3.

- No noise, LPF, DFT
- No noise, HPF, DFT
- Gaussian noise, LPF, DFT
- Gaussian noise, HPF, DFT

- No noise, LPF, DCT
- No noise, HPF, DCT
- Gaussian noise, LPF, DCT
- Gaussian noise, HPF, DCT

Appendix_Methods for reading the input and for displaying the output image in MATLAB **(Important)**

- Please **place a code for reading the input at the beginning of the execution code and place a code for displaying the output image at the end of the execution code,** as shown in the example below.
- Display output images **for each MATLAB figure**, when executing the execution m-file, as shown in the example below.

'Problem_1.m'

```
Clear variables

function Problem_1()
   % Template for EE535 Digial Image Processing
   % Insert the code in the designated area below

   %% Loading directory for image files
   imgdir = uigetdir('Image Directory');
   file = fopen(fullfile(imgdir,'\Gray_Baby_512x512.raw'),'rb');
   gray_image = fread(file,fliplr([512,512]),'*uint8')';
   fclose(file);

   file = fopen(fullfile(imgdir,'\Color_Baby_512x512.raw'),'rb');
   color_image = fread(file,fliplr([512,512*3]),'*uint8')';
   fclose(file);

   %%
   %%---------------------Insert code below ----------------------%%
   output_image = InnerFunction(gray_image); % Sample code - delete
this

   %% Displaying figures (Edit this part as needed)
   figure; imshow(output_image,[]); title('Problem_1.1');
   figure; imshow(output_image2,[]); title('Problem_1.2');
   ...

   %%---------------------------------------------------------------%%

end

function OUTPUT = InnerFunction(INPUT)
%%---------------------------------------------------------------%%
                          %function code%
%%---------------------------------------------------------------%%
end
```