Assignment #5

Due date: May 24 (Th.)

Submission

E-mail a zip file including the source codes, a report and test images to TA. **You must submit function m-files per problems like Appendix at the end of this document. If you don't follow the code structure in Appendix, the score will be deducted.** The filename should be named as student idn_name.zip (e.g. 20183000_kdhong.zip).
TA's e-mail address is **ltk010203@kaist.ac.kr**.

Due date: **05/24 23:59**. (Refer to the delay policy in the web site)

Test images in the web site:
monarch_gray_512x512.raw
object_gray_360x285.raw
texture1_gray_256x256.raw
texture2_gray_256x256.raw
hough_gray_256x256.raw

Notice

All the programming assignments are based on MATLAB. (Do not use any function in MATLAB. But **you can use basic functions available in C++ standard library** like round, ceil, floor, rand, bitshift, sqrt, sum, exponential, log, trigonometric and abs etc. functions **and vectorization**.) 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 with 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.

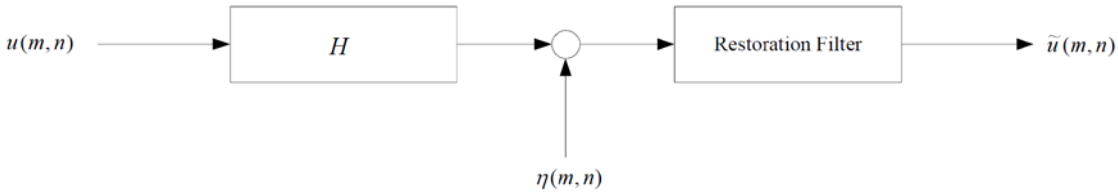
Scoring policy: implementation (60), processing time (10), and report (30)

The criteria of the scoring deduction applied to common problems:

1. Using MATLAB functions
    - Related to 'Language Fundamental' except to basic functions available in C++ standard library: -2 points per problem.
      'Language Fundamental':
      https://kr.mathworks.com/help/matlab/language-fundamentals.html?lang=en
    - the other MATLAB functions: -50% per problem.
2. Inexecutable code: -50% per problem.

1. Image Restoration

Blur a given gray mornarch image and then add the random Gaussian noise $\eta(m, n)$ to it so that the SNR may be 14 dB. Then, restore the blurred image (see the figure below.). Here, filter H is a LPF corresponding to the Hamming window in the frequency domain. For given $u(m, n)$, H, and the variance of $\eta(m, n)$, implement following restoration filters. Then, provide the corresponding output images and their PSNRs.



1.1. Pseudo-Inverse

Perform the restoration using a pseudo-inverse filter. Examine the PSNR value by using 'psnr_p_inverse' as the parameter name of the PSNR.

1.2. Wiener Filter

Perform the restoration using the Wiener filter. Examine the PSNR value by using 'psnr_wiener' as the parameter name of the PSNR.

1.3. Constrained Least Square Restoration

By using the Laplacian operator as a roughness measure, restore the input image. Observe the change of the restored image depending on the variation of λ, and analyze it. (Use 'lamda' as the parameter name of λ.) Also describe the meaning of the optimal λ.
Examine the PSNR value by using 'psnr_const' as the parameter name of the PSNR.

1.4. Draw the power spectra of original image and noise. Describe the frequency characteristics of the pseudo-inverse and Wiener filter.

1.5. Compare and analyze the image characteristics resulting from 1, 2 and 3.

2. Image Analysis

2.1. Transform features

For the images of texture1_gray_256x256.raw and texture2_gray_256x256.raw, perform the Fourier transform and apply angular-slits of 45° and 135° respectively. Determine the ratio of the energy in each slit to the total energy. Display the inverse Fourier transformed results of each angular-slit-applied data. Discuss the results briefly.
The ratios of the energy in the slit of 45° to the total energy are named as `t1_45` and `t2_45` for texture 1 and 2, respectively. For the slit of 135°, the ratios are named as `t1_135` and `t2_135` for texture 1 and 2, respectively.

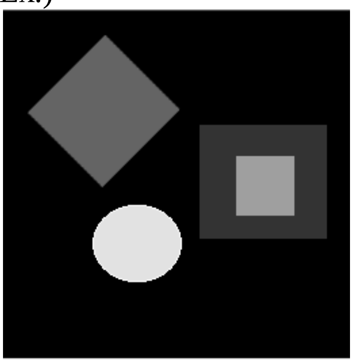
2.2. Threshold-based segmentation

For a given image of object_gray_360x285.raw, extract the object by using a threshold-based segmentation scheme. Use the threshold value determined by the Ostu's method.

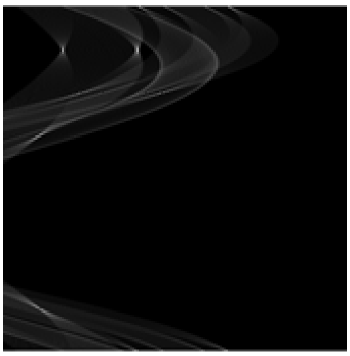
2.3. General Hough Transform

Perform the line Hough transform for the given image, hough_gray_256x256.raw. Display the parameter space (ρ, θ) using the histogram equalization and display the input image with detected straight lines. You can refer to an example given below.

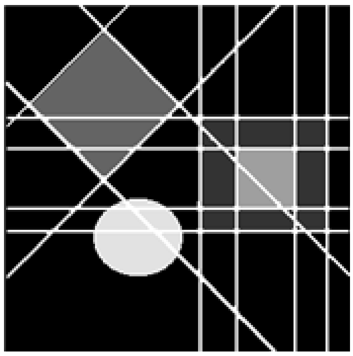
Ex.)



Input image



Parameter space image



Detected straight lines

Results

1.1
- Pseudo-inverse filtered image

1.2
- Wiener filtered image

1.3
- Images according to λ

1.4
- Power spectrum of original image
- Power spectrum of Gaussian noise image
- Power spectrum of pseudo-inversed image
- Power spectrum of Wiener filtered image

2.1
- 45° slit shape in frequency domain
- 135° slit shape in frequency domain
- texture 1 image (spatial domain, frequency domain)
- texture 1 image applied 45° slit (frequency domain)
- Inverse texture 1 image after applying 45° slit (spatial domain)
- texture 1 image applied 135° slit (frequency domain)
- Inverse texture 1 image after applying 135° slit (spatial domain)
- texture 2 image (spatial domain, frequency domain)
- texture 2 image applied 45° slit (frequency domain)
- Inverse texture 2 image after applying 45° slit (spatial domain)
- texture 2 image applied 135° slit (frequency domain)
- Inverse texture 2 image after applying 135° slit (spatial domain)

2.2
- Extracted object image by using a threshold-based segmentation scheme.

2.3
- Input image
- Parameter space image
- Detected straight lines image

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_512x256.raw'),'rb');
   gray_image = fread(file,fliplr([256,512]),'*uint8')';
   fclose(file);

   file = fopen(fullfile(imgdir,'\Color_Baby_512x256.raw'),'rb');
   color_image = fread(file,fliplr([256,512*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
```