The Hong Kong University of Science and Technology
Department of Computer Science and Engineering
MSBD5010 (Fall 2020)

Assignment 1

Total = 100 marks
Due: 11:55pm, Oct. 11, 2020
Assignments must be submitted via Canvas
Late Policy: 10% reduction; only one day late is allowed, i.e., 11:55pm, October 12.

Overview

This assignment consists of two sections: programming section and written section. Both programming and written parts should be submitted via the Canvas system on or before the due date. If you would like to finish the written assignment with hand writing, you may scan and upload it as a PDF file.

In this programming assignment, you will use MATLAB to compute gradient magnitude of the image, to program an additive (zero mean) uniform random noise generator, periodic noise generator, arithmetic mean filter, Alpha-trimmed filter, and Wiener filter. A set of M-files can be obtained from the course website. The routine found in the msbd5010_assign1.m file performs a series of image processing and display operations on a pre-defined grayscale image. You are asked to complete the missing implementations of functions in the given skeleton code.

Programming assignment specifics (80%)

Part 1: Histogram equalization (15%)

The function *histogram_eq.m* involves a series of operations on the histograms of images.

1.1 Given a gray level image *G*, compute and display its histogram. Note: You are not allowed to use the built-in histogram function.

1.2 Perform histogram equalization on *G*, then compute and display the histogram of the result image.

1.3 Given a color image *C* in RGB mode, perform histogram equalization on the R, G, B channels of C separately. Rebuild a RGB image from these histogram-equalized channels.

1.4 Compute the histograms on R, G, B channels separately and then calculate an average histogram from these three histograms. Use this average histogram as the basis to obtain a single-valued histogram equalization intensity transformation function. Apply this function to the R, G, B channels individually. Rebuild an RGB image from these processed channels.

Part 2: Create a gradient magnitude image from a grayscale image. (5%)

You need to complete the function in the file "grad_mag_image.m". The function "grad_mag_image" takes a grayscale image of type uint8 as input and returns a gradient magnitude image of the same data type. $mag(\nabla f)$ at z_5 can be calculated with the following equation:

$$mag(\nabla f) \approx |(z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)| + |(z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)|$$

z1	z2	z3
z4	z5	z6
z7	z8	z9

Hint: you need to define the gradient mask in "grad_mag_image.m".

Part 3.1: An additive uniform random noise generator. (5%)

You need to complete the implementation of an additive uniform random noise generator in the gen_uniform_noise.m file. The routine gen_uniform_noise takes four variables as input: size of an image along the X-axis and Y-axis, and the lower bound and upper bound. You should generate an image of **additive uniform random noise** with the given lower bound and upper bound. The data type of the output image should be double.

Hint: You can find a MATLAB internal function to generate uniform random numbers.

Part 3.2: Arithmetic mean filter. (5%)

You need to complete the implementation of the routine in the arithmetic_mean_filter.m file. The routine takes a noisy image (in grayscale format of type uint8) as input and attempts to remove the noise with an arithmetic mean filter. A 5×5 window is used to filter the noisy image. The output image type should be uint8.

Part 3.3: Alpha-trimmed mean filter. (15%)

You need to complete the atrimmed_mean_filter.m file. The routine takes a noisy image (in grayscale format of type uint8) as input and attempts to

remove the noise with an alpha-trimmed mean filter. Different values of d will be applied to the filter. A 3x3 window is used to filter the noisy image. The output image type should be uint8.

Part 4: High-boost filter (10%)

High-boost filtering is a process that has been used for many years by the printing and publishing industry to sharpen images consists of subtracting an unsharp (smoothed) version of an image from the original image. The process is as follows,

a. Blur the original image $f(x, y)$, denote it as $\overline{f}(x, y)$.
b. Subtract the blur image $\overline{f}(x, y)$ from the original image (the result difference is usually called the mask), denote it as $g_{mask}(x, y)$
c. Add the mask to the original.

The high-boost filtering can be represented by the following equation,

$$g(x, y) = f(x, y) + k * g_{mask}(x, y),$$

Where $k > 1$.

Perform high-boost filtering on the given image. The averaging part of the process should be done using the filter in Part 3.2. Choose a k as you see fit.

Part 5.1: Periodic noise (5%)

You need to complete the implementation of an additive periodic noise generator in the *gen_periodic_noise.m* file. One of the periodic noise model with the size of $M \times N$ in 2D is the sine wave, which is given by $r(x, y) = A_x \sin(2\pi u_0(x + B_x)/M) + A_y \sin(2\pi v_0(y + B_y)/N))$,where A_x and A_y are amplitudes, u_0 and v_0 are the frequencies along x-axis and y-axis. The noise generator will be invoked in the following sub-tasks.

Part 5.2: Denoising in frequency domain (10%)

Let $A_x = A_y = 10$, $u_0 = \frac{2M}{\pi}$, $v_0 = \frac{2N}{\pi}$, $B_x = 0$, $B_y = 0$, add the above periodic noise to the given image. You are required to implement the function *band_reject_filter.m* which performs bandreject filtering on a gray level image, returning a gray scale image. You can choose to implement either the Butterworth bandreject filter or the Gaussian bandreject filter, which are given as follows,

$$H_{Butterworth}(u, v) = \frac{1}{1 + \left[\frac{D(u, v)W}{D^2(u, v) - D_0^2}\right]^{2n}}$$

$$H_{Guassian}(u, v) = 1 - e^{-\frac{1}{2}\left[\frac{D^2(u,v) - D_0^2}{D(u,v)W}\right]^2}$$

where $D(u,v) = \sqrt{(u-u_0)^2 + (v-v_0)^2}$ is the distance from the origin of the centered frequency rectangle, W is the band width and D_0 is its radial center. **Please determine the suitable band width and its radial center for the periodic noise given by the above parameters.**

Part 6: Wiener filtering with the power spectra of noise and un-degraded image. (10%)

You need to complete the Wiener filter, suppose we know the power spectra of the noise S_η and the un-degraded image S_f. You need to complete the implementation in the wiener_filter.m file. The data type of the output image should be uint8. You are not allowed to use the internal MATLAB function "wiener2" for this task.

Sample run of the programming assignment

The main routine of the assignment including displaying the final results is well written in the msbd5010_assign1.m file. After completing all the functions, you are supposed to get three figures on the screen when you run the command below in the MATLAB environment.

```
>> msbd5010_assign1;
```

Written assignment specifics (20%)

The Fourier spectrum (Fourier Representation) in Fig.(b) corresponds to the original image Fig.(a) and the Fourier spectrum in Fig.(d) was obtained after the image Fig.(a) was padded with zeros (Shown in the Fig.(c)).

a) Explain the significant increase in signal strength along the vertical and horizontal axes of Fig.(d) compared with Fig.(b).
b) Explain the significant increase in signal strength in the low frequency region of Fig.(d) compared with Fig.(b).

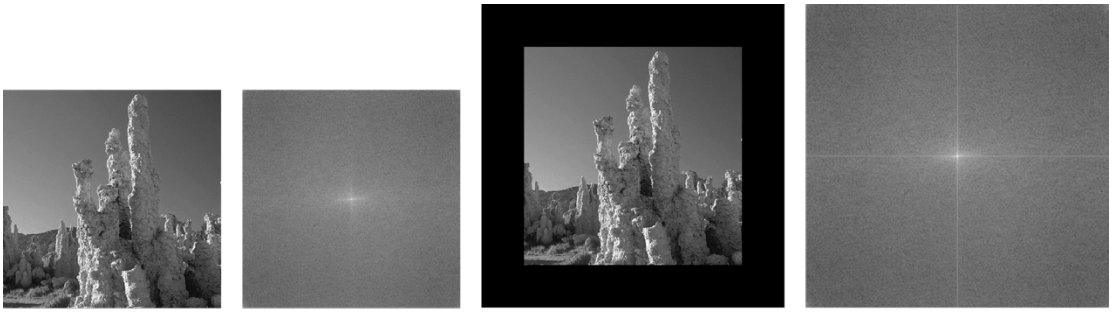

Figure: Fourier Transform

Assignment Submission and Marking

Your submitted all the .m files (including the provided main function) along with the written assignment. You must compress all your files with the following filename format: [your 8-digit student ID]_assign1.zip, e.g. 09654321_assign1.zip, into one file.

If your assignment compressed file has been submitted multiple times before the due date (including late submission date), the new version will replace the old version in marking. Your assignment will be marked using MATLAB R2019a under the Windows environment.