The Hong Kong University of Science and Technology
Department of Computer Science and Engineering
COMP4421 (Fall 2019)

Assignment 1

Total = 100 marks
Due: 11:55pm, Oct. 11, 2019
Assignments must be submitted via Canvas
Late Policy: 10% reduction; only one day late is allowed, i.e., 11:55pm, Oct. 12.

Overview

This assignment consists of two sections: written section and programming section. Both programming and written parts should be submitted via the Canvas system. If you would like to finish the written assignment with hand writing, you may scan and upload it. Please submit your answers of the written part in PDF format.

In this programming section, you will use the MATLAB (or Octave) to practice spatial filtering and filtering in frequency domain. A set of M-files can be obtained from the course website. The routine found in the `main.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 programming section.

The written assignment and programming assignment should be zipped (as .zip file) together and renamed as "{your student ID}_assignment_1.zip". Please submitted the zip file via the Canvas system on or before the due date.

1. Written assignment specifics (20%)

One of the effective methods for edge sharpening and contrast enhancement is to apply high-frequency emphasis followed by the histogram equalization.

a) Show that whether or not it matters which process (high-frequency emphasis or histogram equalization) is applied first.
b) If the order does matter, briefly explain why using one of the other method first.

2. Programming assignment specifics (80%)

2.1 Linear Spatial Filtering (20%)

You need to complete the function in “filter_spa.m” and define the input variables in “main.m”. The function “filter_spa.m” takes a grayscale image of type uint8 along with a filter as input and returns an image convoluted with the given filter. Note that you are required to implement the spatial filtering by yourself. The Matlab built-in functions such as “conv2”, “imfilter”, “fspecial”, “medfilt2” or other third-party libraries are not allowed. The function “filter_spa.m” should be able to:

2.1.1 Smooth an image with averaging filters with different sizes, for example, 3×3, 5×5, 11×11 and so on.

2.1.2 Compute x-gradient, y-gradient of an image with Prewitt operator.

2.1.3 Sharpen an image with a 3×3 Laplacian filter. As for the Laplacian filters, you can choose any one of the four variants, as shown in the Reference section (section 3).

Bonus (10%) The naïve approach to convolute an image with a filter mask is to create two loops to retrieve each pixel and do the convolution. Please try to use at most one loop to replace the original two-loop image retrieval. You can take the LBP code (lbp.m) in Canvas as a reference for one-loop image retrieval.

2.2 Non-linear Spatial Filtering (20%)

You need to complete the function in “add_gaussian_noise.m”, “add_salt_pepper_noise.m” and “medfilt2d.m”. Note that the Matlab built-in functions like noise generator and median calculation are not allowed in this question.

2.2.1 The function “add_gaussian_noise.m” takes an input image, mean and standard deviation of the Gaussian distribution as input. This function “add_gaussian_noise.m” adds Gaussian noise to the input image with input mean and standard variance.

2.2.2 The function “add_salt_pepper_noise.m” add salt-and-pepper noise to the input image with the probabilities of the two noise components. We assume that the probabilities for salt (light dot) and pepper (dark dot) are the same.

2.2.3 The function “medfilt2d.m” takes a noise image and an integer “size” as input, and returns image filtered by median filter. The input “size” defines the window size for the median filter. For example, the size of 3×3 median filter is 3. For image with gaussian noise, please show your noisy image with gaussian noise (mean = 0 and standard variance = 0.03) and filtering in figures. For image with salt-and-pepper noise, please show your noisy image with noise probabilities 0.3 and filtering result in figures.

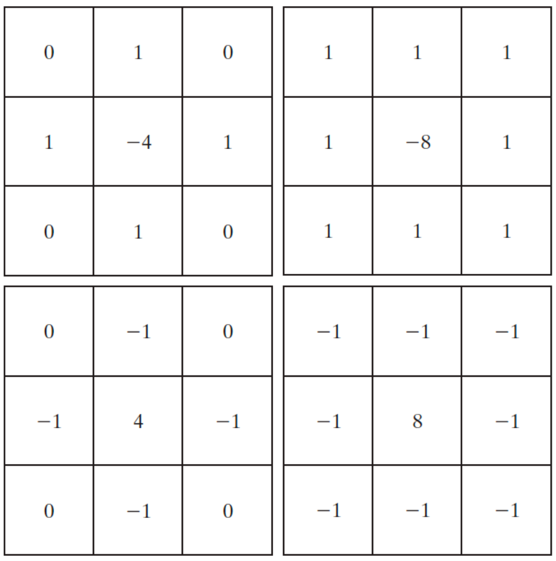
2.3 Filtering in the Frequency Domain (20%)

You need to complete the function "lowpass_filter.m". The function "lowpass_filter.m" takes an input image and the cutoff of the filter D_0 as input, and return an image filtered with ideal lowpass filter. The data type of the output image should be uint8. You are allowed to use functions like "fft2", "ifft2" and "fftshift" to complete this task.

2.4 High-Frequency Emphasis (20%)

You need to complete the "high_freq_emphasis.m" file. The function emphasizes the high frequency components of an image with ideal high pass filter in the frequency domain. You also need to complete to the lowpass filter in "lowpass_filter.m" as a step of implementing the high frequency emphasis one. Different values of a and b will be applied to the filter (see main.m task 4). Please search for the optimal combination for the input "a" and "b". The output image type should be uint8.

3. Reference



0	1	0
1	−4	1
0	1	0

1	1	1
1	−8	1
1	1	1

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

a b
c d

FIGURE 3.37
(a) Filter mask used to implement Eq. (3.6-6).
(b) Mask used to implement an extension of this equation that includes the diagonal terms.
(c) and (d) Two other implementations of the Laplacian found frequently in practice.