Concordia University

Department of Computer Science & Software Engineering

COMP 478/6771 Image Processing

Assignment 2 - Due Date: *October 15, 2024*

Part I: Theoretical questions

1. Image Sharpening Questions:

   a) (**3 points**) Consider the following three Laplacian filters A, B, and C:

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

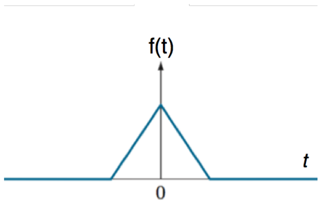
   If we filter a given image *I* by using filters A, B, and C respectively to obtain the sharpened images I_A, I_B, and I_C. What can you say about these filtered images? (Hint: how do their results differ in terms of image sharpness)

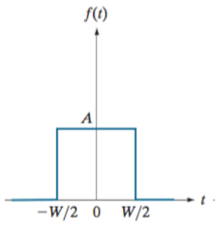
   b) (**7 points**) Write down a 5-by-5 Laplacian-like filter with the center element equal to -12. What general rules should you follow to build such a filter? If we apply this filter to an image *I*, do we get a sharper image compared to those images obtained in part a)? Explain your answer.

2. Fourier Transform Questions:

   a) (**7 points**) Study ***Example 4.1*** in the textbook, then follow the steps in that example to find the Fourier Transform of the function $f(t) = A$ for $0 \le t \le W$ and $f(t) = 0$ otherwise; where both *A* and *W* are constants. Explain the differences between your result and the result in Example 4.1. Consider the case where $A = W = 2$, what is the Fourier Transform of $f(t)$ in this case?

   b) (**4 points**) Use the result of ***Example 4.1*** to find the Fourier Transform of the tent function. The tent function is shown below on the left. Note that the tent function is the convolution of two box functions shown in ***Fig. 4.4(a)*** (Image on the right).





Part II: Programming questions

1. (**22 points**) Download the image, *Doc.tiff* from the assignment folder then carry out the **adaptive thresholding** algorithm to binarize the text in the image without the shadow (remember to convert it to grayscale). Please show your results, discuss about your choice of parameters and filters, and compare your results with the ***adaptthresh()*** function in MATLAB (use "dark foreground" and remember to check out all the relevant options). In scikit-image, a similar function is called ***threshold_adpative()***.

In ***adaptive threshold*** unlike fixed threshold, the threshold value at each pixel location depends on the neighboring pixel intensities. To calculate the threshold *T(x,y)*, the threshold value at pixel location *(x, y)* in the image, we perform the following steps:

1) A *m-by-m* region around the pixel location is selected. *m* is selected by the user.
2) The next step is to calculate the weighted average of the *m-by-m* region (averaging filter). You can choose any averaging filter of your choice and provide your reasons. We will denote the weighted average at location (x,y) by *WA(x, y)*.
3) The next step is to find the Threshold value *T(x, y)* by subtracting a constant parameter, ***c*** from the weighted average value *WA(x, y)* calculated for each pixel in the previous step. The threshold value *T(x, y)* at pixel location *(x, y)* is then calculated using the formula:

$$T(x, y) = WA(x, y) - c$$

Now you will just need to apply the threshold transfer function to obtain the result.

2. (**2 points**) Adding texture to an image is a common task in digital image manipulation. This is often achieved by multiplying an image with a "texture image". Download the image, *paper.tiff* from the assignment folder. Apply this paper texture to the adaptive thresholding result of *Doc.tiff* and show your result.