Assignment 3: Digital High-Pass and Low-Pass Filters

Paul Monroy
University of California, Santa Barbara
Santa Barbara, CA 93106
paulmonroy@ucsb.edu

Abstract

In the realm of signal processing, it is necessary to perform operations on images to make them clearer, look for any changes in color, etc. In this report, I will analyze some of the filters used in image processing.

A. Introduction

Digital filters are used for discrete time systems. They are used widely in image processing where operations such as peak detection, edge detection, and moving average filters are implemented. Each filter is made up of vectors or matrices, depending on how smooth you want the output to be. Some of these filters are analogous to continuous time domain operations such as convolution. I will analyze how these 3 filters come to be and how they are used in image processing.

B. Problem 1: Formulating a 3x3 matrix operator as a gradient filter

A gradient is defined as the derivative with respect to each dimension. In continuous time the derivative is defined as

$$\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x)$$

where Δx is defined as a small distance from the point at *x*. However, in discrete time, each data point is separated by a distance of 1, therefore Δx = 1. We end up with

$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$

The simplest discrete gradient filter is the [-1 1] vector gradient that takes two number that are next to each other and finds the difference between them. For example, if we have a vector of numbers

$$f(x) = [10 \;\; 45 \;\; 30 \;\; 42 \;\; 8 \;\; 78]$$

the filter would be implemented as follows:

$$\begin{array}{c}
[\,10(-1) + 45(1) \;\;\; 30 \;\;\; 42 \;\;\; 8 \;\;\; 78] \\
[\,10 \;\;\; 45(-1) + \; 30(1) \;\;\; 42 \;\;\; 8 \;\;\; 78] \\
[\,10 \;\;\; 45 \;\; 30(-1) + 42(1) \;\;\; 8 \;\;\; 78] \\
[\,10 \;\;\; 45 \;\;\; 30 \;\;\; 42(-1) + 8\;(1) \;\;\; 78] \\
[\,10 \;\;\; 45 \;\;\; 30 \;\;\; 42 \;\;\; 8\;(-1) + 78(1)]
\end{array}$$

Which gives us the result

$$f'(x) = [0 \;\;\; 35 \;\;\; -15 \;\;\; 12 \;\;\; -34 \;\;\; 70]$$

Similarly, the second derivative is seen by:

$$f''(x) = \; [0 \;\; 35 \;\; -50 \;\;\; 27 \; -46 \;\;\; 114\,]$$

A symmetric filter would be [-1 0 1].

In image analysis, edge detection is one of the most common operations in image processing for the purpose of image enhancement. The simplest form of edge-detection operator in image processing is the 2 x 2 operator for the 1st-order edge detection. In the horizontal direction, it is in the form

$$C = \begin{bmatrix} -1 & +1 \\ -1 & +1 \end{bmatrix}$$

The 3 x 3 first-order edge detection operator has been widely applied because of the symmetry

$$D = \begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix}$$

Note that this gradient is for the x-direction. They y-direction operator is

$$D = \begin{bmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

The weighted version looks like

$$\begin{bmatrix} -\frac{1}{2} & 0 & +\frac{1}{2} \\ -\frac{1}{2} & 0 & +\frac{1}{2} \\ -\frac{1}{2} & 0 & +\frac{1}{2} \end{bmatrix} \qquad \begin{bmatrix} +\frac{1}{2} & +\frac{1}{2} & +\frac{1}{2} \\ 0 & 0 & 0 \\ -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \end{bmatrix}$$

in the x and y direction, respectively.

We combine these two to get our gradient for both directions by making the y-direction gradient complex

$$\begin{bmatrix} -\frac{1}{2} & 0 & +\frac{1}{2} \\ -\frac{1}{2} & 0 & +\frac{1}{2} \\ -\frac{1}{2} & 0 & +\frac{1}{2} \end{bmatrix} + j \begin{bmatrix} +\frac{1}{2} & +\frac{1}{2} & +\frac{1}{2} \\ 0 & 0 & 0 \\ -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \end{bmatrix}$$

We are now ready to perform operations with these matrices, such as the discrete time Fourier Transform (DTFT). The plot of this is shown in **Figure 1**

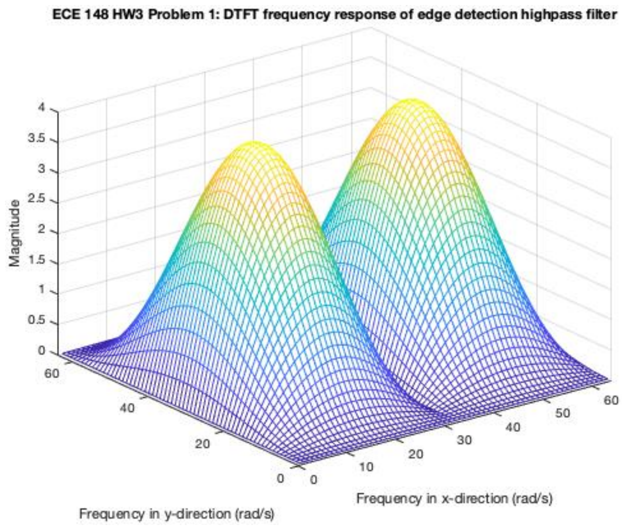

Figure 1

C. Problem 2: Applying the gradient filter to the image for edge detection.

As stated in the problem statement, we now apply the gradient to the image given to us, *FigTreeMove008_Gray.jpg*. Through kernel convolution, we get an image that has its edges detected.

Part 2a asks for the gradient in the x-direction. This is shown in **Figure 2**.

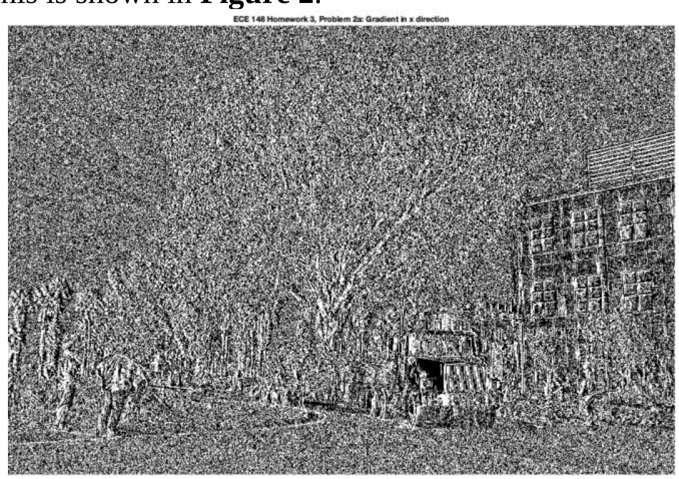

Figure 2

In part 2b we are asked for the edge profile in the vertical direction. This is shown in **Figure 3**.

Figure 3

In part 2c we are asked for the combined horizontal and vertical edge profile. This is shown in **Figure 4.**

Figure 4

Finally, in part 2d we superimpose the combined gradient to get what we see in **Figure 5.** The edges are highlighted in the picture as we expect in edge detection.

Figure 5

D. **Problem 3: Formulating a 3 x 3 matrix operator as the Laplacian filter in the discrete form**

The Laplacian filter can be taken from the second derivative of a function. That is,

$$\frac{\delta^2}{\delta t^2}(\ \cdot) \leftrightarrow \frac{1}{\Delta t^2}(1 - z^{-1})^2$$

$$\begin{aligned} H(z) &= \tfrac{1}{\Delta t^2}(1 - 2z^{-1} + z^{-2}) \\ &= \tfrac{1}{\Delta t^2}2z^{-1}(\tfrac{1}{2}z^1 - 1 + \tfrac{1}{2}z^{-1}) \\ &= -\tfrac{1}{\Delta t^2}2z^{-1}(-\tfrac{1}{2}z^1 + 1 - \tfrac{1}{2}z^{-1}) \\ &= \tfrac{1}{\Delta t^2}2z^{-1}(1 - \tfrac{1}{2}(z^{+1} + z^{-1}) \end{aligned}$$

Converting back to time domain, we have $y[n] = (\ x[n] - \frac{1}{2}(x[n+1] + x[n-1])\ )$.

Where the x[n] term is a center pixel and the and the other two are the average of the left and right pixel in the horizontal direction (or average of top and lower pixel if in the vertical direction). Based on this concept, the 3 x 3 peak detection operator is in the form

$$\frac{1}{8}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

We are now ready to perform a discrete time Fourier transform on this operator. This is seen in **Figure 6**.

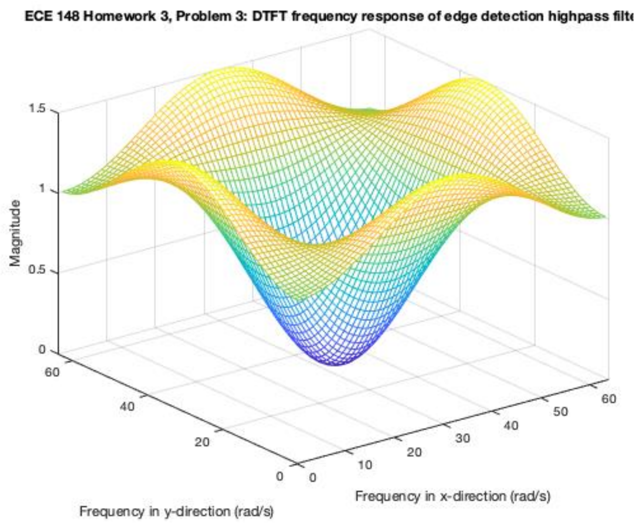

Figure 6

E. Problem 4: Applying the Laplacian filter to the image for peak detection

As stated in the problem statement, we now apply the gradient to the image given to us, *FigTreeMove008_Gray.jpg*. Through kernel convolution, we get an image that has its peaks detected.

In problem 4a, we plot the image with peak detection, seen in **Figure 7**.

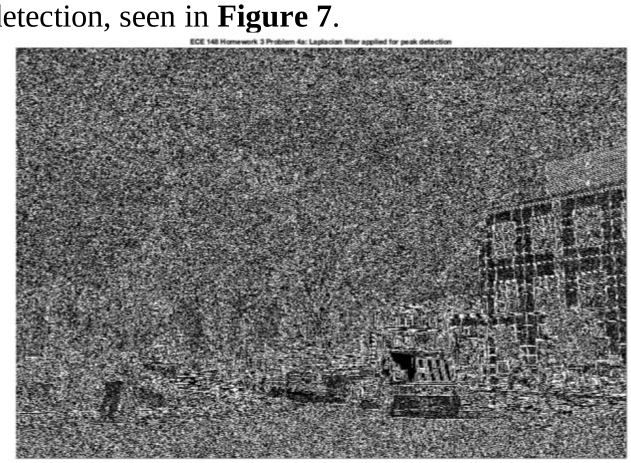

Figure 7

In problem 4b, we superimpose the peak detected image onto the original image. This is seen in **Figure 8**.

Figure 8

F. Problem 5: Formulating a 3 x 3 matrix operator as the moving average filter in the discrete form

A moving average filter is an N-point window along a sequence and the output is the average value of the N pixels within the window. As an example, based on the concept, the matrix representation of the convolution kernel of the 3-point moving average filter is

$$\frac{1}{3}[1 \quad 1 \quad 1]$$

A 2-D version of a moving average filter is

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Performing a discrete time Fourier Transform, we see the result in **Figure 9**.

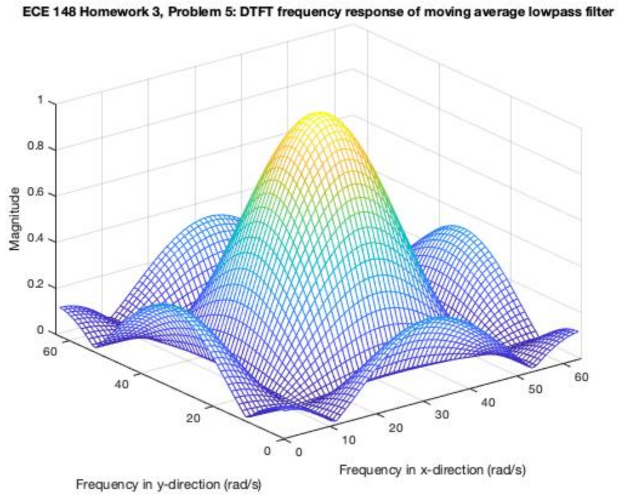

Figure 9

G. Problem 6: Applying the moving average filter to the given image for smoothing.

As stated in the problem statement, we now apply the moving average filter to the image given to us, *FigTreeMove008_Gray.jpg*.

In Problem 6a, we are asked to show the result of this lowpass filter. This is seen in **Figure 10**. It is blank, although it looks like there is no picture there.

ECE 148 Homework 3 Problem 6a: Lowpass filter applied for moving average

Figure 10

In Problem 6b, we superimpose the filter onto the original image. This is seen in **Figure 11**.

Figure 11

H. Summary

After analyzing each of these filters, we see that they all come from different backgrounds and perform their own unique operations, analogous to those of continuous time domain.

The gradient edge detection filter comes from the first derivative of a function. As seen in the set up for Problem 1, we can adjust it for discrete time and 2 dimensions to get a symmetric 3x3 filter.

The Laplacian peak detection filter comes from the second derivative of a function. As seen in the set up for Problem 3, we can adjust it for discrete time and 2 dimensions to get a symmetric 3x3 filter.

The moving average filter is just as the name sounds. If a window has N values, then the filter output has a value equal to the average of the n-values, as seen in Problem 5.

These filters, along with many others, are essential in image processing and show us how important it is to analyze discrete time data.