In the name of God

School of Electrical and Computer Engineering

Signals and Systems

Computer Exercise 3

Delivery deadline: 27 June

Professor: Dr. Rabiee

Z Transform / Convolution

Table of Contents

Basic s of Z Transform in MATLAB

LTI systems in the form of differential equations is shown as follows:

$$\sum_{k=0}^{N} b_k y[n-k] = \sum_{k=0}^{M} a_k x[n-k]$$

With the help of Z transform, we get the following representation:

$$H(z) = \frac{\sum_{k=0}^{M} a_k z^{-k}}{\sum_{k=0}^{N} b_k z^{-k}}$$

$H(z)$ is called the system transfer function. In MATLAB, to create a transfer function, we must have to extend the transfer function:

$$Y(z) = \frac{a_0 + a_1 z^{-1} + \ldots + a_M z^{-M}}{b_0 + b_1 z^{-1} + \ldots + b_N z^{-N}} X(z)$$

From now on, the coefficients z^{-i} in the numerator are shown as $a = (a_0, a_1, \ldots, a_M)$, and at the denominator as $b = (b_0, b_1, \ldots, b_N)$.

In the following, we will examine the practical functions of working with Z Transform in MATLAB:

1) **zplan :** This function plots the representation of zero and the pole of the transfer function in the z -domain along with the unit circle as a reference. Each pole is denoted by 'x' and each zero is denoted by 'o' , which is defined as follows:

$$zplane(a, b) \quad , \quad zplane(z, p)$$

(* If the two inputs of the function are defined as columns, the first input is considered as zeros and the second input as poles, otherwise the two inputs are the same coefficients as a and b)

a. Draw the zero and pole representation of transfer function $H(z) = \frac{2+2z^{-1}+z^{-2}}{1-0.8z^{-1}}$.

2) **impz :** This function obtains and plots the system impulse response, which is defined as follows :

$$[h, t] = impz(a, b)$$

(* If we do not specify h and t , the impulse response will be drawn.)

b. Compute the impulse response of the following system and draw with the stem command :

$$H(z) = \frac{1}{1-0.9z^{-1}}$$

3) **freqz :** This function plots the system frequency response using a differential equation or system transfer function :

$$[H, \omega] = freqz(a, b, N)$$

Where H is the frequency response of the system and ω is the normalized anular frequency, which is obtained as follows:

$$\omega = \frac{\Omega}{F_s} \text{ , } \Omega = 2\pi F$$

For example, consider the continuous sinosoidal signal of Figure 1 at a frequency of 10 Hz, from which 15 data are sampled per second ($F_s = 15$). So we will have:

$$\Omega = 20\pi \, rad/sec \rightarrow \omega = \frac{20\pi}{15} = \frac{4\pi}{3} rad/sample = \frac{2}{3} cycle/sample$$

Therefore, the above result indicates that in each two *<u>periods</u>* *of signal alternation, there are 3 discrete data.*

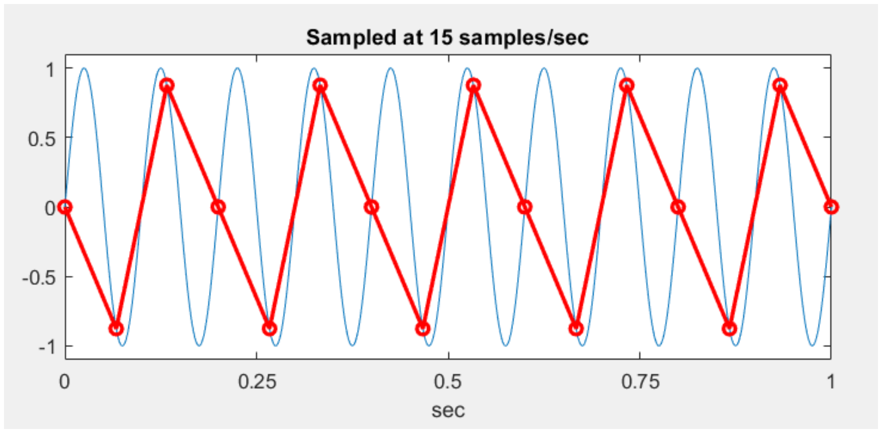

figure 1

(* If you do not specify a value for N , the default value is 512.)

c. Draw the magnitude and phase of the frequency response of the following system:

$$H(z) = \frac{1-0.9^8 z^{-8}}{1-0.9z^{-1}}$$

4) **filter :** By extending the transfer function $H(z)$, We come to the following criterion:

$$X(z)\left(a_0 + a_1 z^{-1} + \ldots + a_N z^{-N}\right) = Y(z)(b_0 + b_1 z^{-1} + \ldots + bz^{-M})$$

$$Z^{-1} \rightarrow$$

$$x[n] \; * \left(a_0, a_1, \ldots, a_N\right) = y[n] * \left(b_0, b_1, \ldots, b_M\right) \rightarrow x[n] * a = y[n] * b$$

By use of filter function in MATLAB, we approximate the output $y[n]$with values a, b , $x[n]$ that are written as follows:

$$y[n] = filter(a, b, x)$$

d. Filter a delta dirac signal with a length of 100 with the transfer function $H(z) = \frac{1}{1-0.9z^{-1}}$.

5) **tf2zp** : This function returns the zeros, poles and coefficients[1] of a transfer function with coefficients a and b:

$$[z, p, k] = tf2zp(a, b)$$

e. Find the coefficient, zero and poles of a system with a transfer function $H(z) = \frac{z}{z^2-1.4144z+1}$.

6) **zp2tf :** This function works exactly the opposite of the tf2zp function and finds the coefficients a, b by having the location of zero and the pole and the coefficient of the transfer function.

f. Suppose a transfer function with poles $p = [0.5\ ,\ 0.45\pm0.5j]$ and zeros $z = [-1, j, -j]$ hase a gain of $k = 1$ .Display the zero and pole of the transfer function as well as the magnitude and phase of the frequency response.

7) **residuez :** Consider the conversion function defined at the beginning of this section, which can be broken down into fractional parts as follows:

$$H(z) = \sum_{k=0}^{M-N} c_k z^{-k} + \sum_{k=1}^{N} \frac{A_k}{1-p_k z^{-1}}$$

The above view can be obtained with the help of MATLAB as follows:

$$[A, p, C] = residuez(a, b) \quad , \quad [a, b] = residuez(A, p, C)$$

Consider the transfer function, for example $H(z) = \frac{z^{-1}}{3-4z^{-1}+z^{-2}}$. MATLAB output for the trio $[A, p, C]$ is as follows:

$$A = [0.5, -0.5]\ ,\ p = [1, 0.33],\ C = []$$

Therefore, the division of the function into partial fractions is as follows:

[1] Gain

$$H(z) = \frac{0.5}{1-z^{-1}} - \frac{0.5}{1-\frac{1}{3}z^{-1}}$$

(* The three outputs can also be given to the residuez function and the coefficients a and b extracted.)

g. Consider the following differential equation:

$$y[n] = x[n-1] - 1.2x[n-2] + x[n-3] + 1.3y[n-1] - 1.04y[n-2] + 0.222y[n-3]$$

★ Obtain the separated form of the system transfer function.
★ Draw the magnitude and phase of the system frequency response.
★ Find the zero and the poles of the transfer function.
★ Obtain and draw the system impulse response.

Removing disturbing frequencies using Z Transform

In this section, you will review the tools that you practiced in the previous section. There is a song in the exercise file that contains an annoying sound. Your goal is to eliminate the annoying sound as much as possible so that the sound of the song can be heard clearly.

1. First listen to the sound of the mentioned song and describe what you hear.

2. Obtain the Discrete song data ($x[n]$) And the sampling frequency. Then draw $x[n]$ in terms of n .

3. Using the knowledge gained from previous computer exersise about DTFT , measure and draw the magnitude and phase of Fourier coefficients once in terms of frequency F (Up to Nyquist frequency) and once per ω.

4. Use the diagrams obtained in the previous section to find the disturbing frequencies in Hz and $rad/sample$(name the frequencies in $rad/sample$ $\omega_1$the ω_2).

5. Next, with the help of Z transform, you are going to implement a transfer function that attenuates the disturbing frequencies in your song. [2]The general form of your transfer function is as follows:

$$H(z) = \frac{(z-z_1)(z-z_2)(z-z_3)(z-z_4)}{(z-p_1)(z-p_2)(z-p_3)(z-p_4)}$$

[2] Notch Filter

The positions of the zeros and poles of the transfer function are shown as follows:

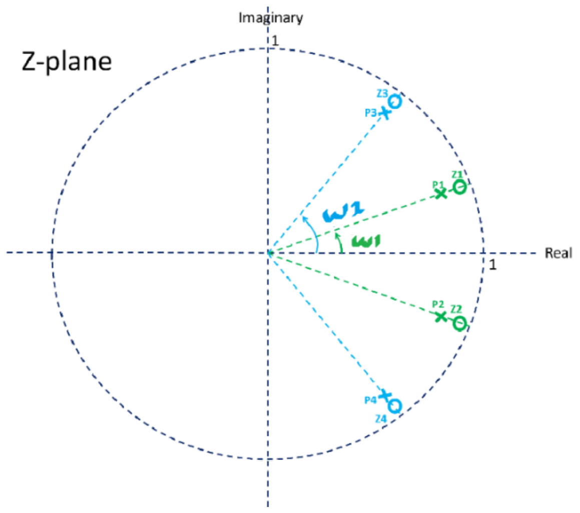

Figure 1 : Zero position and pole of the transfer function $H(z)$

$$Zeros: \{z_1 = r_1 e^{j\omega_1},\ z_2 = r_1 e^{-j\omega_1}\ z_3 = r_1 e^{j\omega_2},\ z_4 = r_1 e^{-j\omega_2}\}\ ,$$

$$Poles: \{p_1 = r_2 e^{j\omega_1},\ p_2 = r_2 e^{-j\omega_1}\ p_3 = r_2 e^{j\omega_2},\ p_4 = r_2 e^{-j\omega_2}\}$$

$Consider\ that\ r_1 = 0.99,\ r_2 = 0.96.$

❶ What do you think is the reason for the above zero and pole reperesentation?

To answer this question, plot the frequency response graph in terms of ω. for this, first you need to calculate the coefficients for z^i in numerator and denominator:

$$H(z) = \frac{a_0 z^4 + a_1 z^3 + a_2 z^2 + a_3 z^2 + a_4 z^1 + a_5}{b_0 z^4 + b_1 z^3 + b_2 z^2 + b_3 z^2 + b_4 z^1 + b_5}$$

Calculate the Coefficients in the numerator $a = (a_0, a_1, a_2, a_3, a_4, a_5)$ And at the denominator $b = (b_0, b_1, b_2, b_3, b_4, b_5)$ using the convolution function in MATLAB.

Draw the zero and the poles representation of the above transfer function and compare with Figure 1.

You can use one of the following two commands to draw a frequency response:

$$[H, \omega] = freqz(Num, Den, N) \qquad [H,\ \omega] = freqz(Num, Den, N,\ 'whole')$$

❷ What do you think is the difference between the frequency response in these two methods?

❸ Draw the magnitude and phase diagram of the frequency response in terms of ω in both normal and decibel comparisons .

(* The value of N is arbitrary and the larger it is, the more accurate the results will be.)

❹ The values assigned to radius r_1 , r_2, have a direct effect on the shape of the filter created.

- ➔ Suppose the values of both radius are close to 1 once (r_1 , $r_2 > 0.9$), but the distance between the two radius ($r_1 - r_2$) is 0.01 once and 0.07 another time.
- ➔ Consider the distance between the two radius is constant(for example 0.03) but set the values of the two radius once in the interval $[0.7, 0.8]$ and again in the interval [0.9, 1] .

In each case, plot the magnitude of the frequency response and compare the effect of the distance between the two radius and their values in the shape of the created filter.

6. Now, using the filter function introduced in the previous section, filter the input song and then draw the frequency response of the filtered song. Has your filter been able to reduce the impact of disturbing frequencies? Listen to the filtered song. Is the annoying sound still heard?
7. In order to be able to build a stronger filter than before to reduce the impact of disturbing frequencies, we need to increase the degree of the introduced transfer function:

$$H_{new}(z) = \frac{(z-z_1)^n(z-z_2)^n(z-z_3)^n(z-z_4)^n}{(z-p_1)^n(z-p_2)^n(z-p_3)^n(z-p_4)^n}$$

Which n shows the order of zeros and the poles of the new transfer function.

For example, the integrated model of the new transfer function is the cascaded version of the original transfer function for 5 times ($n = 5$) which is depicted in Figure 2 :

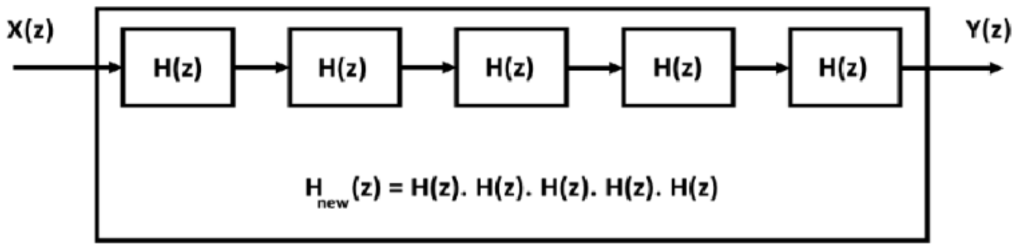

figure 2

➔ Now, draw the frequency response of the new filter with the same zeros and poles as the original transfer function. What improvement do you find in your filter compared to the original one ?

➔ As before, filter the input song this time using the new transfer function and plot the frequency response of the filtered song and compare with section 6 . Is the sound of the new filtered song clear?

Edge detection with convolution

In this question, we want to calculate the local variance of [3]an image to retrieve its edges, and in general see what information can be extracted from the image using the kernel type. In general, local information in an image means extracting a property of a limited number of pixels from an image. The good thing about this is that we turn an image into smaller frames and try to extract information about that image from within each frame, just like our brains do when we see an image.

Depending on which image feature we want to focus on, we need to look at the image with a different lens. The type of lens you choose is called the kernel. Usually kernels are square in size $n * n$. The kernel size can vary depending on the application and the number of pixels in the original image. Here we have the kernel of size $3 * 3$.

Now consider a custom kernel, for example $kernel = [4\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -4\ ]$. By selecting a window of size $3 * 3$ From the pixels in the main image, the information about them is calculated by multiplying the corresponding kernel internally in the selected window. For example, in Figure 3, the orange square pixels in the left corner of the image are multiplied by the corresponding kernel, and the result is stored in the corresponding array of a new matrix.

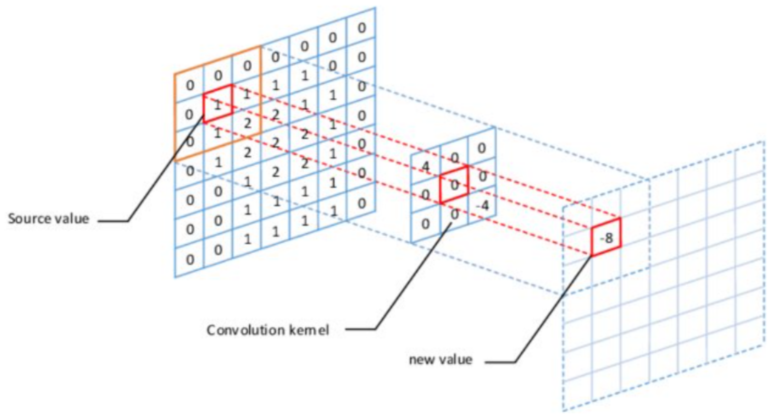

Figure 3

Now, in order to form all the new matrix elements resulting from kernel convolution in the main image, each time we shift the selected window by one row or one column to cover all the pixels of the original image. For example, in Figure 4, we started the calculations from the upper left window, and each time we shifted the window one direction to the right, and calculated the result of the internal multiplication until we reached the upper right window.

[3] Local Variance

* Also, a row and a zero column are added to the sides of the image so that the matrix resulting from the input image convolving in the kernel is the same size as the input image . For example in Figure 4, if we do not add zero rows and columns, the size of the output matrix will be $3 * 3$ instead of $5 * 5$. Why ?

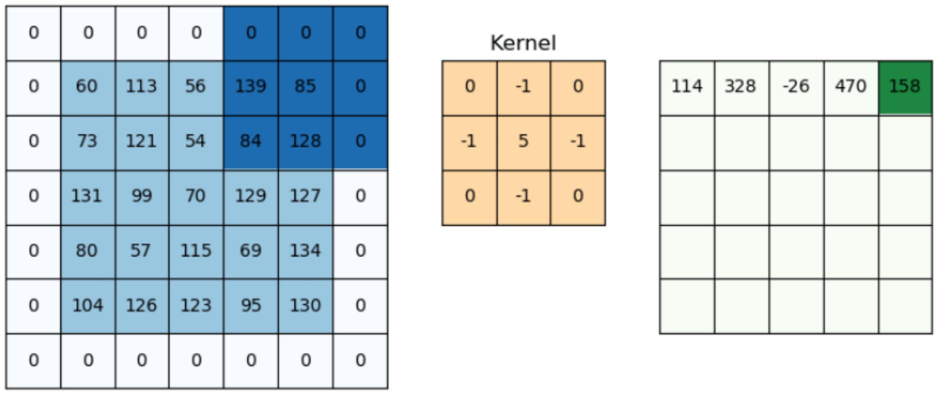

Figure 4

1) Write a function to output destination image by taking the input image and a custom kernel of size $3 * 3$ (The destination image must be the same size as the input image). Compare the result with the output of the ready-made MATLAB function:

$$Convolved_{image} = conv2(Image, kernel,\ mode = 'same')$$

(* Consider the input image as a random matrix)

Now that you are familiar with two-dimensional convolution and kernel, let's choose the kernel type. In order to be able to distinguish the edges in an image, we need to obtain the local variance matrix of the image, which is calculated as follows:

$$Lvar(image) = Lmean\left(image^2\right) - Lmean(image)^2$$

Therefore, to calculate the local variance, we need to calculate the local average an image. To calculate the local average of an image, it is enough to calculate the average of the pixels inside a window. Therefore, the selected kernel type must be in the form

$kernel = \left[\frac{1}{9}\ \frac{1}{9}\ \frac{1}{9}, \frac{1}{9}\ \frac{1}{9}\ \frac{1}{9}, \frac{1}{9}\ \frac{1}{9}\ \frac{1}{9}\right].$

2) First read the ' Amesterdam.jpg ' image in the computer exercise folder and then draw the black and white image [4] as shown in Figure 5 using the imagesc command (* Convert the values in the black and white image to ' double ').

[4] Gray

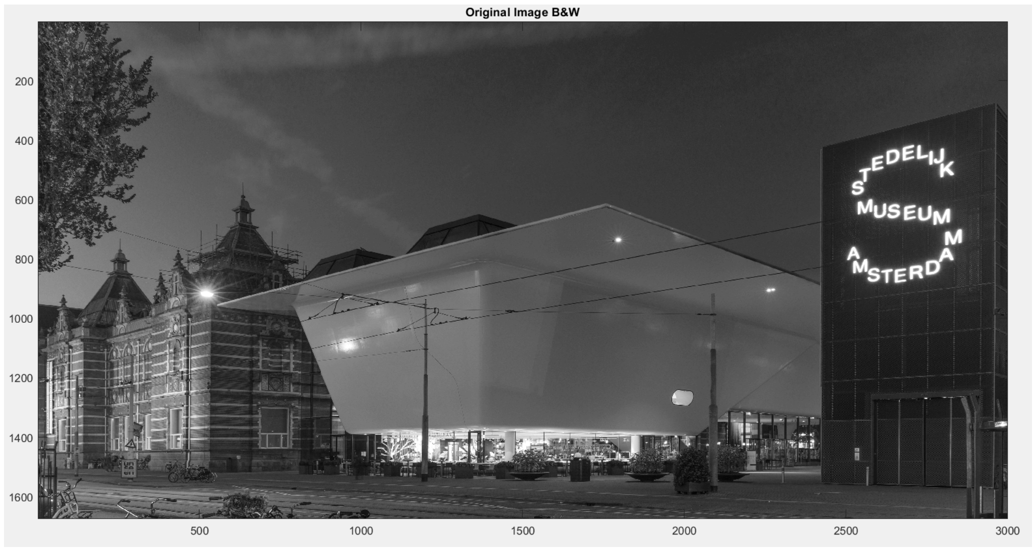

Figure 5

3) Now, with the help of the defined kernel and using the $conv2$MATLAB function, get the local variance matrix of the above black and white image.
4) To distinguish between edges and other parts of the image, you need to convert the local variance matrix to a binary matrix. To do this, set the elements of the matrix whose value is greater than the average of the total elements of the matrix to 0 and otherwise to 1, and then draw the resulting binary matrix as shown in Figure 6.

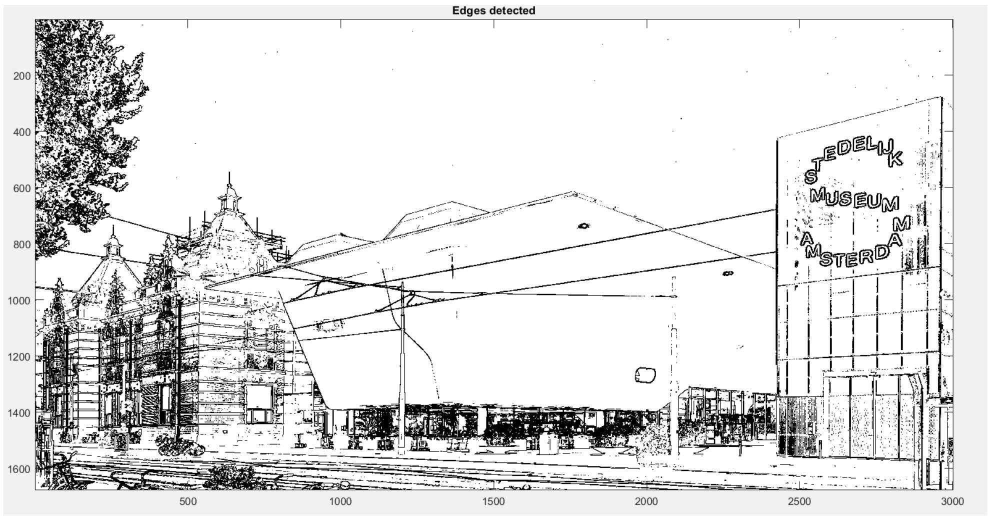

Figure 6

5) Now suppose you perform all the above steps for the two kernels $kernel_1 = [1\ 0\ -1,\ 1\ 0\ -1,\ 1\ 0\ -1]$, $kernel_2 = [1\ 1\ 1,\ 0\ 0\ 0,\ -1\ -1\ -1]$, respectively, and plot the resulting binary matrices as shown in Figures 7 and 8, respectively.

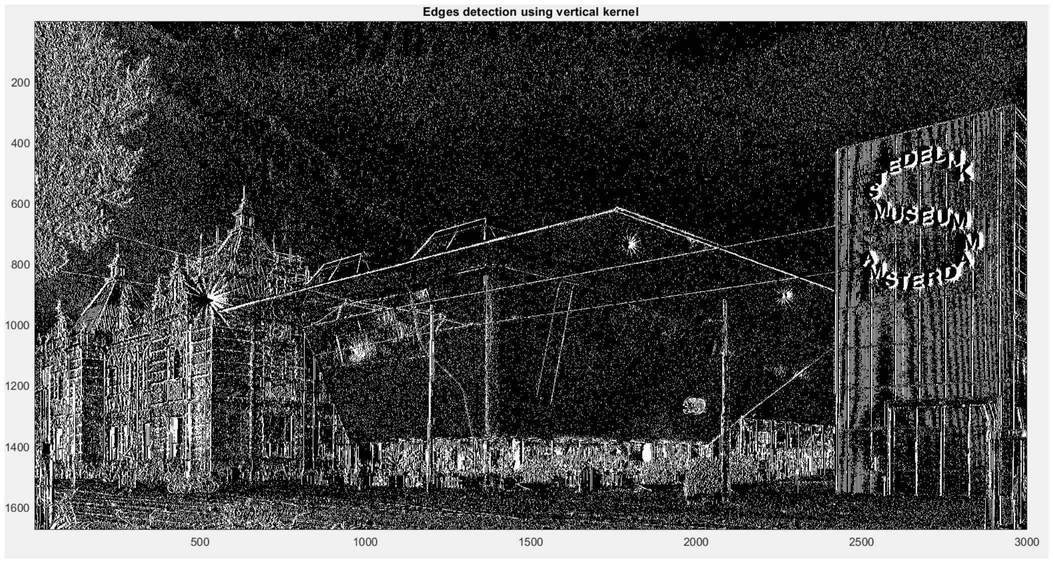

Figure 7: Binary image of the first kernel

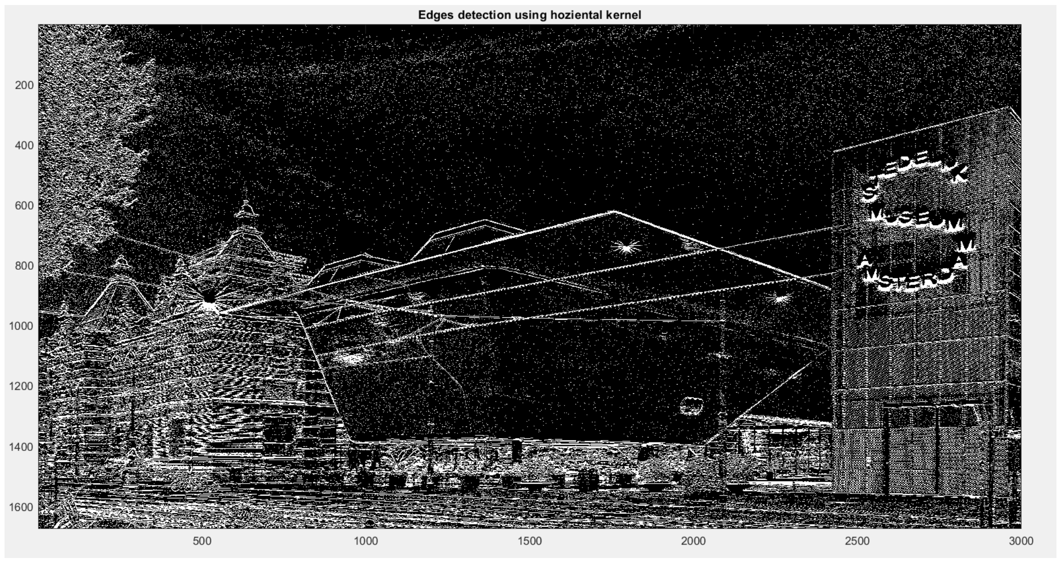

Figure 8: Binary image of the second kernel

6) In your opinion, what are the special features of the two defined kernels? What feature of the black and white image corresponding to each kernel does the black and white image highlight?
7) This time, consider a new kernel from the sum of the two kernels above:
   $kernel = [2\ 1\ 0,\ 1\ 0\ \ -1\ ,0\ \ -1\ \ -2\ ]\ .$
   What is the difference between the binary image obtained by this kernel and the two binary images obtained in Section 6?
   What do you conclude?

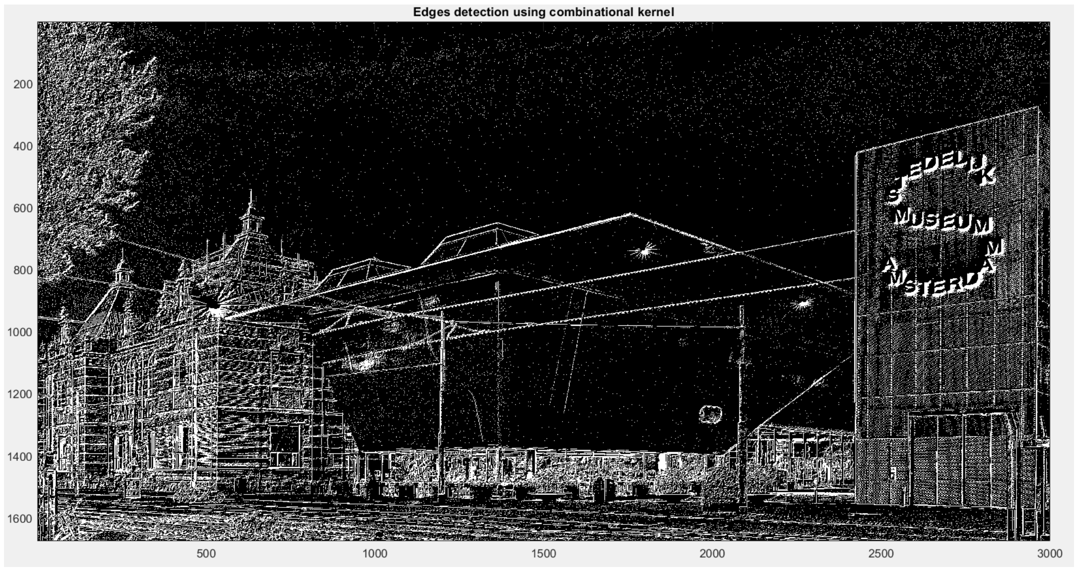

Figure 9 : Binary image obtained from the new kernel

Bonus

Suppose you have a biassed dice with an odds of 1 or 2 equal $\frac{1}{12}$, 3 or 4 equal $\frac{1}{6}$, and 5 or 6 equal.$\frac{1}{4}$ Be. Using the Z-Transform , design an algorithm that can calculate the probability that the sum of k dice is equal to n . Then implement your algorithm in MATLAB and calculate the required probability for n = 3 and k = 2 .

Final Points

1. Finally, place the delivery code for each question in a .m file. Also, in each .m file, separate the different parts of each question by cells.
2. The evaluation of your work is determined by having a proper report, so be sure to include the output of the graphs and the answers to the different parts in your report. Exercises without a report will not be awarded a grade.
3. The Computer exercise aims to help you learn new concepts, so you will get a low score if there is an unjustifiable similarity in your reports or codes.
4. If you have any questions or concerns about the exercise, you can contact me via email (sh.vassef@ut.ac.ir) or in the telegram group.

Good luck.