BILKENT UNIVERSITY ELECTRICAL AND ELECTRONICS ENGINEERING DEPARTMENT

---

2019 Summer School
EEE - 321: Signals and Systems
Lab 4 - Off-Lab Assignment
Lab Day/Time/Room: 18.07.2019 - 17:40-19:30 - EE 211
Report Submission Deadline: 19.07.2019 - 17:20

---

Please carefully study this off-lab assignment before coming to the laboratory. There will be a quiz at the beginning of the lab session; this may contain conceptual, analytical, and Matlab-based questions about the lab. Some of the exercises will be performed by hand and others by using Matlab. The report should be prepared as hardcopy and written by hand (except the Matlab based plots and the code segments). Include all the plots, codes, calculations, explanations etc. to your report in a readable format.

**Note:** Note: Along with this pdf file, you will find a .zip file containing a Matlab function and three pictures. Unzip the archive and place all files contained in it under the current directory of Matlab.

# Part 1: Displaying Images

In this lab, you are going to do a bit of image processing. You will perform some simple operations on gray scale (white and black) digital images and see the results.

In this course up to now, we have mainly dealt with one-dimensional signals such as $x(t)$ or $x[n]$. These signals are said to be one-dimensional because they are functions of one independent variable (t or n).

Images, on the other hand are signals of two independent variables. Therefore they are two-dimensional signals. A gray scale digital image can be represented with a 2D discrete function $x[m,n]$ such that $x[m,n]$ denotes the light intensity of the $(m,n)$th pixel of the image (m and n are integers). Since practical images are of finite size (such as $512 \times 512$, $1920 \times 1080$ etc.), we usually have $x[m,n] = 0$ for $m \in [0, M_y - 1]$ and $n \in [0, M_x - 1]$ where the image size is $M_y \times M_x$.

Recall that we store 1D signals in 1D arrays in Matlab. Since images are 2D signals, they are stored in matrices in Matlab. A $M_y \times M_x$ image is stored in a $M_y \times M_x$ matrix inMatlab.

In this first part, you will practice reading images in Matlab and displaying them. You are not going to put any image in your report for this part. It is enough if you just answer the questions below.

Place the m-file named ReadMyImage.m and the image named Part1.bmp under the current directory of Matlab.

Issue the command

**A=ReadMyImage('Part1.bmp');**

You will see that a 2D matrix of type double will be created and stored in the workspace. This matrix contains the image. Find out the size of this matrix, i.e. what are $M_y$ and $M_x$ for this image?

Now examine the ReadMyImage function line by line, and explain the operation of each line. What is the required change in this function to save the image as RGB and the type of uint8?

Next, issue the command

**figure; imshow(A,[ ]);**

A new figure window will open and the image will be displayed in it. What is the importance of [ ] in this code? When it is needed and when it is not?

During the rest of this assignment, you will use these commands to read and display various images.

For this part, just make sure that you run the functions above and see the image without any problem. (For this purpose, check the Matlab command window so that you do not get any warning while displaying the images.)

# Part 2: Basics for 2D Signals and Systems

Recall that 1D discrete time (DT) systems map a 1D input signal $x[n]$ to another 1D signal $y[n]$. (We say discrete 'time' since the independent variable usually denotes time.)

Similarly, a 2D discrete space (DS) system maps a 2D input signal x[m,n] to a 2D output signal y[m,n]. (Now we say discrete 'space' since the independent variables usually denote the space coordinates.) Recall that a 1D DT system is called linear time invariant (LTI) if it satisfies the following two properties:

- $a_1 x_1[n] + a_2 x_2[n]$ produces $a_1 y_1[n] + a_2 y_2[n]$ for all $x_1[n]$, $x_2[n]$, $a_1$, $a_2$.
- $x[n - n_0]$ produces $y[n - n_0]$ for all $x[n]$ and $n_0$.

Similarly, a 2D DS system is called linear space invariant (LSI) if it satisfies:

- $a_1 x_1[m,n] + a_2 x_2[m,n]$ produces $a_1 y_1[m,n] + a_2 y_2[m,n]$ for all $x_1[m,n]$, $x_2[m,n]$, $a_1$, $a_2$.
- $x[m - m_0, n - n_0]$ produces $y[m - m_0, n - n_0]$ for all $x[m,n]$, $m_0$ and $n_0$.

Now let us develop the input output representation of 2D DS LSI systems by forming an analogy with 1D DT LTI systems. Recall that a 1D discrete impulse signal is defined as

$$\delta[n] = \begin{cases} 1 & \text{if } n = 0 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

While developing the input-output relation for a 1D DT LTI system, we first wrote the input signal x[n] as a superposition of shifted impulse signals as:

$$x[n] = \sum_{k=-\infty}^{\infty} x[n]\delta[n-k] \tag{2}$$

where we interpreted x[k] as the coefficient of the impulse shifted by k units (that is, x[k] is the coefficient of $\delta[n-k]$). Then, we named the response that the system gives to $\delta[n]$ as $h[n]$ (impulse response). Using the time invariance property of the system, we recognized that the response of the system to $\delta[n-k]$ should be $h[n-k]$. Then, using the linearity property of the system, we wrote the input-output relation of the 1D DT LTI system as:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] \tag{3}$$

We named the above operation as the convolution of x[n] and h[n] and used the short hand notation

$$y[n] = x[n] * h[n] \tag{4}$$

to denote it.

Now, define the two dimensional discrete impulse signal as

$$\delta[m,n] = \begin{cases} 1 & \text{if } m = 0, n = 0 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

and going through the same steps, derive the input-output relation of a 2D DS LSI system, or derive the formula for 2D convolution of x[m,n] and h[m,n] that we denote as $x[m,n] * h[m,n]$. In other words, derive the 2D version of Equation 3. Show-explain your steps as is done above for the 1D case. Include your work to your report. You should obtain the following result:

$$y[m,n] = \sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} x[k,l]h[m-k,n-l] \tag{6}$$

$$= x[m,n] * h[m,n] \tag{7}$$

$$= \sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} h[k,l]x[m-k,n-l] \tag{8}$$

# Part 3: 2D FIR Filter

Recall that a 1D DT LTI system is called FIR if the impulse response h[n] contains a finite number of nonzero values such that $h[n] = 0$ for $n \neq [0, M_h - 1]$ where $M_h$ is a positive integer. Similarly, a 2D DS LSI system is called FIR if the impulse response h[m,n] contains a finite number of nonzero values such that $h[m,n] = 0$ for $m \neq [0, M_y - 1]$ and $n \neq [0, M_x - 1]$ where $M_y$ and $M_x$ are positive integers.

In this part, you will write a Matlab function that computes the output when a finite sized input image x[m,n] (of size $M_y \times M_x$) is input to a 2D FIR DS LSI system whose impulse response h[m,n] is of size $H_y \times H_x$. From another perspective, your code will compute the 2D convolution of two finite-size 2D signals x[m,n] and h[m,n]. We assume that

- x[m,n] can only be nonzero within $0 \leq m \leq M_y - 1$ and $0 \leq n \leq M_x - 1$
- h[m,n] can only be nonzero within $0 \leq m \leq H_y - 1$ and $0 \leq n \leq H_x - 1$

Under these conditions the 2D discrete convolution reduces to:

$$y[m,n] = y[n] = \sum_{k=0}^{H_y-1} \sum_{l=0}^{H_x-1} h[k,l]x[m-k,n-l] \tag{9}$$

Based on the above equation, show the range of nonzero values of y[m,n]. That is, find four numbers $D_1$, $D_2$, $D_3$ and $D_4$ such that y[m,n] is always zero when $m < D_1$, $m > D_2$, $n < D_3$ and $n > D_4$. Determine these numbers in terms of $M_x$, $M_y$, $H_x$ and $H_y$. Include your work to your report.

Next, write aMatlab function of the following form
**function [y]=DSLSI2D(h,x)** where

- **h**, a matrix of size $H_y \times H_x$ denotes the impulse response of the system, such that **h**(1,1) = h[0,0], **h**(1,2) = h[0,1],...,**h**(k,l) = h[k-1,l-1].
- **x**, a matrix of size $M_x \times M_y$ denotes the input signal, such that **x**(1,1) = x[0,0], **x**(1,2) = x[0,1],...,**x**(k,l) = x[k-1,l-1].
- **y**, a matrix of size $N_y \times N_x$ denotes the output signal, such that **y**(1,1) = y[0,0], **y**(1,2) = y[0,1],...,**y**(k,l) = y[k-1,l-1].

Basically, this function implements Equation 9.

Note: Do not use any built in command of Matlab, and your code should work efficiently, as usual.

Check your function: If you take

$$x = \begin{bmatrix} 1 & 0 & 2 \\ -1 & 3 & 1 \\ -2 & 4 & 0 \end{bmatrix} \tag{10}$$

and

$$h = \begin{bmatrix} 1 & -1 \\ 0 & 2 \end{bmatrix} \tag{11}$$

you should get

$$y = \begin{bmatrix} 1 & -1 & 2 & -2 \\ -1 & 6 & -2 & 3 \\ -2 & 4 & 2 & 2 \\ 0 & -4 & 8 & 0 \end{bmatrix} \tag{12}$$

Include your code to your report.

# Part 4: Image Denoising

Now it is time to see some practical image processing examples.

Read the picture named Part4.jpg in a matrix and display it. The picture that you see is highly corrupted by noise. Noisy images may occur due to several reasons. For example, during the capturing the image by a CCD camera, each CCD pixel may insert noise. Or, if the image is transmitted through a noisy communication channel, the received image can be corrupted by noise. In this part, you will try to rescue this image from the noise without disturbing the image itself as much as possible.

Suppose we know that the noise on the picture has high frequency content (just as many other types of noise that we encounter in electronics engineering). We also know that typical daily life images taken by a typical camera have low-frequency content. Therefore, why should not we apply a low pass filter to the noisy image and try to eliminate the noise?

One period of the magnitude response of 2D discrete Fourier transform of an ideal low pass filter in the interval $\omega_x, \omega_y \in [-\pi, \pi)$ has the following form

$$H(e^{j\omega_x}, e^{j\omega_y}) = \begin{cases} 1 & \text{if } 0 \leq |\omega_x| \leq B \text{ and } 0 \leq |\omega_y| \leq B \\ 0 & \text{otherwise,} \end{cases} \tag{13}$$

where B is a free parameter that determines the bandwidth of the filter. By assuming that the phase response of this filter is always 0, compute the 2D invese discrete Fourier transform of this function, which is the impulse response of the ideal low pass filter, and write it to your report.

The impulse response that you found is symmetric with respect to origin and it extends to infinity. However, the Matlab function that you wrote in the previous part assumes that the impulse response of the filter is nonzero when $0 \leq m \leq H_y - 1$ and $0 \leq n \leq H_x - 1$. In order to use this filter in your Matlab code, firstly, apply a space shift to the impulse response that you found so that it becomes symmetric with respect to $n = \lceil (H_x - 1)/2 \rceil$ and $m = \lceil (H_y - 1)/2 \rceil$. What is the effect of this space shift in the Fourier domain? Then assume that the impulse response of the filter is 0 outside the interval $0 \leq m \leq H_y - 1$ and $0 \leq n \leq H_x - 1$.

Write the final form of the impulse response of the filter in your report together with the Matlab code which generates this impulse response, h[m,n], as a 2D matrix. Next, you will investigate denoising performance for different bandwidth and filter size values.

- Assume that B is $a\pi/10$, where a is the second last digit of your ID number (take it 5 if it is 0), and $H_x = H_y$. For different values of $H_x$, convolve the image matrix and h and examine the output images. Which $H_x$ value gives the best performance in terms of the image denoising? What is the drawback of choosing large or small value for $H_x$? Put only the output image which shows the best performance to your report indicating $H_x$ value for this image.

- Let D denote your ID number, and let D6 denote your ID number in modulo 6. That is $D6 = D\,mod\,6$. Take $H_x = H_y$ to be D6. If $D6 = 0$,take $H_x = H_y = 3$. For different values of B, convolve the image and h and examine the output images. Which B value gives the best performance in terms of the image denoising? What is the drawback of choosing large or small value for B? Put only the output image which shows the best performance to your report indicating B value for this image.

- Now try different combinations of B and $H_x$. By examining the output images, discuss the subject from different views, such as, denoising performance, excessive amount of smoothing, border effects ets. Put only the output image which shows the best performance to your report indicating B and $H_x$ value for this image.

As you see, it is possible to greatly clarify the information in a corrupted signal using only the simple concepts of convolution, LSI systems, etc. In this part, we tried to remove the noise from a corrupted image. Such problems are called image denoising problems. Many algorithms have been developed by many researchers that try to clear the images from different types of noise while giving minimum damage to the original image.

# Part 5: Edge Detection

Another widely studied interesting problem of image processing is the detection of edges in an image, which is called the edge detection problem. In a typical image, edges are the set of points in the close vicinity of which the pixel values change abruptly. Since changes are sudden and great, edges are inherently associated with high frequencies. Therefore, we can make use of high pass filters to detect edges. Read the picture named Part5.bmp in a matrix and display it.

Consider a 2D FIR DS LSI system whose impulse response h1[m,n] is equal to:

$$h1[m,n] = \begin{cases} 1 & \text{if } m = 1, n = -1 \\ 1 & \text{if } m = 0, n = -1 \\ 1 & \text{if } m = -1, n = -1 \\ -1 & \text{if } m = 1, n = 1 \\ -1 & \text{if } m = 0, n = -1 \\ -1 & \text{if } m = -1, n = -1 \\ 0 & \text{otherwise.} \end{cases} \tag{14}$$

Prepare h1[m,n] and process your image with this filter. Let y1[m,n] denote the resulting image. Display this image and include it to your report. Also, display the image which is defined as $s1[m,n] = y1[m,n]^2$. Include this image to your report as well. Which parts of the original image are emphasized? Include your answer to your report. Now let

$$h2[m,n] = \begin{cases} 1 & \text{if } m = -1, n = 1 \\ 1 & \text{if } m = -1, n = 0 \\ 1 & \text{if } m = -1, n = -1 \\ -1 & \text{if } m = 1, n = 1 \\ -1 & \text{if } m = 1, n = 0 \\ -1 & \text{if } m = 1, n = -1 \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Again, prepare h2[m,n] and process your image with this filter. Let y2[m,n] denote the resulting image. Display this image and include it to your report. Also, display the image which is defined as $s2[m,n] = y2[m,n]^2$. Include this image to your report as well. Which parts of the original image are emphasized now? Comment on the difference with the image you obtained with h1[m,n]. Include your answer and comments to your report.