EEE 391
Basics of Signals and Systems
Fall 2021-2022
Matlab Assignment 1

Deadline: 22 November 2021 by 23.55 on Moodle

There are several types of signals such as audio, video, speech, image, radar and so on. In this assignment you will work on image signals and get hands on experience with MATLAB by implementing some image processing techniques.

Question 1: Some Basic Image Manipulations

You have a sample image provided on Moodle. Download and read it by using the Matlab command: image_matrix = imread('dog.jpg');

(i) Display the size of the "image_matrix". Colorful images are 3D matrices and each pixel is represented by three channels which correspond to red, green and blue (RGB) colors. Any other color is simply a mixture of these three with different ratios. Set red and green channel values of image_matrix to zero and display the image by using "imshow" function. Repeat the same procedure by firstly setting red & blue then blue & green channels to zero and display the output images. Compare the three images you obtained and explain the reasons of differences.

(ii) The given image is a colorful one but we want to convert it to grayscale. You could use "rgb2gray" function to convert an image to grayscale. However, we want you to implement this function to get the intuition behind it. Grayscale images are 2D images and each pixel has a value in between zero and one which represents the brightness/darkness. Grayscale pixel values are weighted average of RGB colors. Since human eyes are more sensitive to green light and less sensitive to blue light, green channel has a greater weight and blue channel has a smaller weight. Commonly used weights are as in the following formula:

$$Gray = (0.299 \times R + 0.587 \times G + 0.114 \times B) \div 512 \tag{1}$$

Use the above formula to convert the image to grayscale. Display the grayscale image and the original image side by side to show that the conversion is successful.

(iii) Upside down version of an image can be obtained by replacing the rows of the image in reverse order. Also, 90 degrees rotated version of an image can be obtained by replacing rows and columns. You can do this by first taking transpose of the image and then replacing the rows in reverse order. Obtain the upside down and 90 degrees rotated versions of the image. Display the normal, upside down and rotated versions side by side.

(iv) We think that the current size of the image is a little large and it would be better to crop it a little bit. You should crop the image such that the remaining image should contain the pixels between 25-775 in x direction and 50-750 in y direction. Display the cropped image and save it in your workspace. Use this version for the next questions.

(v) Pixel values for grayscale image are in between 0 and 1. 1 represents the brightest and 0 represents the darkest color. Replace the pixel values lower than 0.2 by 0 and display the output image. What do you see? What is the effect of this operation?

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Figure 1: Vertical (left) and Horizontal (right) edge detection filters.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Figure 2: Sharpening filter.

Question 2: Convolution and Edge Detection Filter

Hereafter, we want to implement more complicated signal processing operations. These operations are characterised by filter matrices and output images are obtained by applying the chosen filter on the input image by using convolution operation. For a 2D input image X, 2D output image Y and 2D filter H 2D convolution operation is given by the following formula:

$$Y[i,j] = \sum_u \sum_v X[u,v]H[i-u, j-v] \tag{2}$$

Firstly, you will implment an edge detection filter which is shown in figure 1. Since the input is a 2D image, you need to implement 2D convolution between the filter and image. Actually, Matlab provides a built-in function "conv2" which can be used for 2D convolution. However, implementing an operation from scratch is the best way to learn it. Therefore, you should not use this built-in function but implement it by yourself. You can do this by using nested loops. However, it is better to refrain from using loops and use matrix and vector operations as much as possible. This will also make your life easier and you will learn to benefit from the power of Matlab. At the end, compare the outputs of your implementation with the outputs of built-in convolution function.

(i) Use vertical edge detection filter and perform convolution. Display the output image. Compare your result with the output of built-in "conv2" function. Comment on the effect of this filter.

(ii) Use horizontal edge detection filter and perform convolution. Display the output image. Compare your result with the output of built-in "conv2" function. Comment on the effect of this filter.

(iii) Now, merge the output of vertical and horizontal filters and display the result. Compare the outputs for three cases (vertical, horizontal, merged) and comment on the differences.

Question 3: Sharpening Filter

(i) By enhancing the differences in neighbor pixel values, edges can be highlighted and details can be made more significant. You will use sharpening filter (shared in figure 2) for this purpose. Implement convolution operation with this filter and comment on your results by comparing changes in the details of the image.

(ii) Instant changes in a signal correspond to high frequency components. Therefore, images having sharp transitions between neighbor pixel values carry high frequency components. Object edges are highlighted and details can be recognised more easily in such images. By considering that, what kind of a filter is sharpening filter do you think (High pass, low pass, band pass)?

(iii) Now, use "freqz2(kernel, N, N)" function to plot frequency response of sharpening filter. By looking at the plot, what can you say about the type of this filter?

$$\begin{bmatrix} 0.33 & 0.33 & 0.33 \end{bmatrix}$$

Figure 3: Sample moving average filter for M = 3.

Question 4: Noisy Image and Moving Average Filter

(i) During the transmission of a signal, some external factors can change the content and add some noise on it. In this part, you will simulate a noisy signal. Create a matrix which has the same size with the input image and fill it with random numbers. Generate the random numbers by using Gaussian Random Number Generator. Use mean value of 0 and standard deviation 0.5. Multiply the generated matrix by 0.2 to scale it down. Add the noise matrix to input image and display the noisy image.

(ii) Some filters can be used to denoise the received signal. M-point moving average filter is one of them. It basically replaces the each pixel value by the average of (M-1) ÷ 2 left neighbor pixels and (M-1) ÷ 2 right neighbor pixels. For instance, for M = 3:

$$y[n] = \frac{1}{3}(x[m-1] + x[m] + x[m+1]) \tag{3}$$

Corresponding filter can be seen in figure3. Apply m-point average filter to the noisy image for M = 7, 21 and 41. Display and compare the results and comment on the effect of the filters. Do they help reducing the noise? Is there an undesirable effect on the image?