ECE 435, Medical Image Processing
Assignment 1

Spring 2020

A digital image can be seen as a matrix (or an array of matrices) of numbers with different data types such as *double, uint8, logical*, among others. By default, MATLAB saves all numeric variables as double-precision floating-point values and unless memory space is a limitation, this default format does not need to be changed when performing mathematical operations. Figure 1 shows the supported data types available in MATLAB[1].

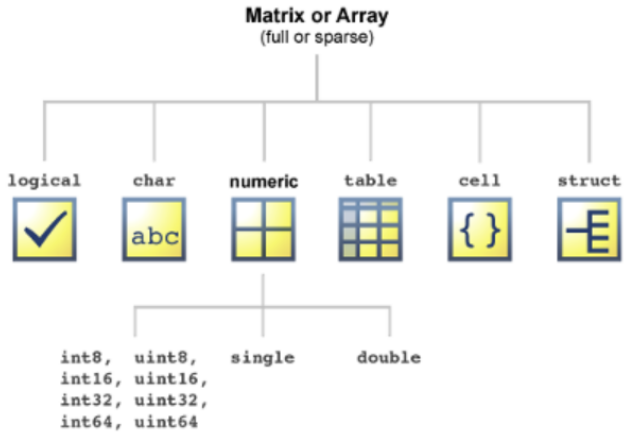

Figure 1: Supported data structures and types in MATLAB.

However, if the type of variable is changed manually, you should make sure that the target variable in the mathematical operations is in the correct format. In image processing, it is important to make sure that matrices have the right data type when applying different functions or mathematical operations. In general, there are three image types: indexed , grayscale (Intensity) and RGB (Truecolor) Images. Before proceding with this assignment, please carefully read the following MATLAB help page:

[1] `https://www.mathworks.com/help/matlab/matlab_prog/fundamental-matlab-classes.html`

https://www.mathworks.com/help/matlab/creating_plots/image-types.html

1 Loading, Manipulating and Storing Images

DICOM (Digital Imaging and Communications in Medicine) is a standard for handling, storing, printing, and transmitting information in medical imaging. DICOM files are 4-D files with the '.dcm' format that encompass a sequence of images received from an imaging device. The dimensions of these files are [rows, columns, samples, slices] where *samples* represents the number of color channels per image (each image is represented by an unit of rows $\times$ columns), and *slices* comprises the number of images. MATLAB provides a wide range of functions for loading, manipulating and storing DICOM files.

This section of the assignment involves the loading, manipulation (i.e., addition of noise) and visualisation of such images.

Problem 1: Gaussian Noise (2 points)

(a) Write the function 'AddNoise.m' to be called using the following form:

$$NoisyImg = AddNoise\ (DicomAddress, NoiseStats)$$

This function receives the DICOM file's address (string) and Gaussian noise parameters, mean and standard deviation ($NoiseStats = [Mean, Std]$) as inputs. It then adds Gaussian noise to the first slice of the DICOM file and plots both the original and noisy slices in the same plot (DO NOT use MATLAB's *imnoise* function). Save the original and noisy slices in the same folder as the original image under the names 'OrgImg.png' and 'GaussianNoise.png'.

Note: Replace all the negative values in the slice by zero before doing any processing. Before saving the images in '.png' format, normalize the image values in the [0,1] range.

(b) Is there any way to save your noisy image with *double* values in the '.png' format and then load it in MATLAB without loss of information?

(c) Use this function to load the image 'RIDER_Lung_CT.dcm' in the assignment folder and report the results for two sets of arbitrary noise parameters.

(d) How many slices are embedded in 'RIDER_Lung_CT.dcm'? What are their dimensions?

Note: In 'imshow', the display range of a grayscale image is specified as a two-element vector of the form [*low high*]. The *imshow* function displays all the value less than or equal to *low* as black, and all the value higher than or equal

to *high* as white. Values between *low* and *high* are displayed as intermediate shades of gray, using the default number of gray levels based on the image data type (see 'DisplayRange' of 'imshow' in MATLAB help). However, if you specify an empty matrix ($imshow(image, [])$), then *imshow* uses a display range of [min(I(:)) max(I(:))]. In other words, the minimum value in I is black, and the maximum value is white. Always use ([]) for better mapping of image values to graylevels when visualizing your images.

Problem 2: Other types of noise and MSE (1 point)

(a) Write the function 'AddNLNoiseNoise.m' to be called using the following form:

$$[NoisyImg, MSE] = AddNLNoise\ (DicomAddress, NoiseType, NoiseChr)$$

This function receives the DICOM file's address, noise type (string), and its statistics information as inputs, and adds noise to the first slice of a file. When writing this function, use MATLAB's *imnoise* to add either 'salt & pepper' or 'speckle' noise types. In the output of the function, also report the Mean Square Error (MSE) between the original image and the noisy one. MSE is a standard measure to calculate the distance between two sets of observations (in our case, images). For two images I_1 and I_2, MSE is defined as follows:

$$MSE = \frac{\sum_{M,N}[I_1(m,n) - I_2(m,n)]^2}{M \times N}, \tag{1}$$

where M and N are the number of rows and columns in the input images. Finally, save the noisy image under the name '*NoiseType*.png' (*NoiseType* should be read from the function's input and reflected in the output, for example 'Salt&pepper.png' or 'Speckle.png') and show the original and noisy images in one plot.

(b) Report the noisy images and MSE results for two sets of 'NoiseChr' for each 'salt & pepper' and 'speckle' noise (total of 4 results).

2 Image Filtering

When acquiring images with imaging systems, one would typically perform filtering operations to mitigate noise, interference, variation in illumination, poor contrast, among others, before further processing the image. Thus, in virtually all computer vision systems, some sort of pre-processing step is required. Linear and nonlinear filters play important roles in this context. In this section of the assignment we will investigate the difference between convolution and correlation, which are operators used in linear filtering, and apply some filters to reduce the effects of noise in the input images.

When sliding a filter on the image, the size of the output image would be less than that of the input because of a lack of information in the neighborhood of

border pixels (see Figure 2). To have an output that preserves the dimensions of the input, the size of the original image should be enlarged based on the size of the desired filter, by adding some extra rows and columns. This process is called *image padding* and can be automatically done in most of the filtering functions offered by MATLAB.

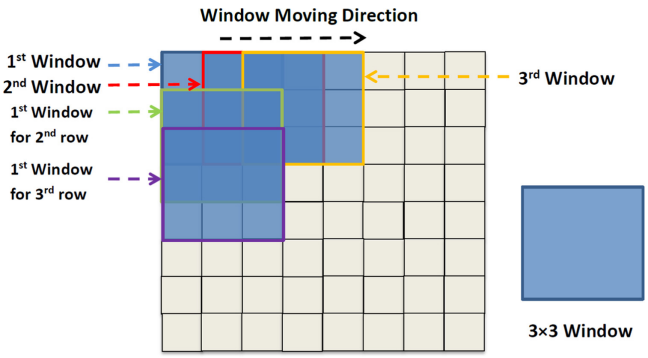

Figure 2: Sliding a 3×3 kernel on an image.

Problem 3: Convolution vs. Correlation (1 point)

(a) Use the 'imfilter' function in MATLAB to filter 'ChestXray.png' with the F_{hor} kernel (use proper padding to preserve the size of the input in the output). Try both 'correlation' and 'convolution' options in the function and report both results. Justify the difference observed between the otuputs.

$$F_{hor} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

(b) Which property of the convolution operation makes it a better option for filtering?

Problem 4: Image Denoising (4 points)

(a) Write the function 'GaussianFiltering.m' to be called using the following form:

$$[I_{out}, MSE\_pre, MSE\_post] = GaussianFiltering\,(OrgImgAdd, NoisyImgAdd, n, \sigma)$$

This fuction filters the gray level input image saved in $OrgImgAdd$ by a $n \times n$ Gaussian filter with standard deviation σ. Show both the input and output images in one plot. Also, report the MSE between the original image and 1) noisy image (MSE_pre), 2) the filtered image (MSE_post).

When writing this function, do not use 'imfilter' or 'imgaussfilt'. Instead, use MATLAB's 'fspecial' function with **two** for-loops. Use proper zero padding to make sure that the output has the same size as the input image.

(b) Is there any difference between convolution and correlation in this case? Why?

(c) Run the 'GaussianFiltering' function on three different noisy images from problems 1 and 2. Report the output images and describe the results.

(d) Write the function 'NLFiltering.m' to be called using the following form:

$$I_{out} = NLFiltering\ (OrgImgAdd, n) \tag{2}$$

This fuction receives the address of an image, 'OrgImgAdd', and filter size n as inputs, and filters the input image with a $n \times n$ median filter. Use one of 'nlfilter' or 'colfilt' functions in MATLAB to avoid for-loops.

(e) Apply 'NLFiltering' to your three noisy images and plot the outputs. Explain the results.

(f) What is the difference between the 'blockproc', 'nlfilter' and 'colfilt' functions in MATLAB?

(g) Although commonly referred as a type of *noise*, speckle is actually an *unwanted modification* of the measured signal. If we model the reflectivity function of the surface under imaging as an array of scatterers, because of the finite resolution of the imaging device, each individual cell will capture only the summation of all the backscatteres coming out of it. Since the backscatteres might have different directions, this summation can be a stronger or weaker version of the expected return signal from that cell. One way to mitigate this phenomenon is to average the sequence of signals obtained in different time steps.

Write the function 'SpeckleDen.m' to be called using the following form:

$$[I_{out}, MSE\_pre, MSE\_post] = SpeckleDen\ (DICOMAddress, n, var\_speckle)$$

This function reads a 'DICOM' file, adds speckle noise with variance 'var_speckle' to every slice in the sequence and then plots the average of the first n slices as a single image (I_{out}). MSE_pre is calculated between the first original slice and its noisy version. MSE_post is calculated between the original slice and the average image, I_{out}. Apply the function to 'RIDER_Lung_CT.dcm' and see if the temporal averaging method can reduce the effect of speckle noise.

(h) What is the main drawback of the averaging method?

Deliverables: When submitting your assignment, use the following format for the name of the folder possessing all the required files:

'FirstName_VNumber_Assignment1.zip' (e.g. 'Claire_V00888888_Assignment1.zip'). The folder should include:

- A '.pdf' file with your written report, which includes answers to all the questions and a description of your results.
- The following MATLAB files and functions:

  AddNoise.m

  AddNLNoise.m

  A MATLAB script for Problem 3 (a).

  GaussianFiltering.m

  NLFiltering.m

  SpeckleDen.m
- The following output images:

  'OrgImg.png'

  'GaussianNoise.png'

  'Speckle.png'

  'Salt&pepper.png'