CS540/440 – Digital Image Processing
Assignment 1 – Matlab Warm-up Exercises
Due: 16:00 p.m. Wednesday, January 31, 2018

General Assignment Instructions:

1. Be sure to place semicolons wherever appropriate, to suppress unnecessary console output, such as when loading images into memory, or operating on them.
2. Please include comments (e.g., **your name and assignment number**) at the top of each m-file. **In your main function, place a message "-----Finish Solving Problem X-----" followed by a pause command (i.e., wait for a key to be pressed before continuing) at the end of each solution, where X is the question number (i.e., 1, 2, 3, etc.).**
3. **You should submit your zipped m-files via the Blackboard system. Please do not send any image**!
4. Discussion of the assignment is encouraged, but **you may not share code**.

Problems:

1. **[1 point]**

Load the image ***peppers.bmp*** into a variable ***A***.
Display the loaded image ***A*** on figure 1 with the message "RGB Original Image" as the figure title.
{*Think: What is the data type of A? What is the size of A?*}
Matlab hints: imread, figure, imshow, title, disp, pause

2. **[5 points]**

Convert image ***A*** into a grayscale image and store it as ***B***.

Transpose image ***B*** as ***TB***.

Vertically flip image ***B*** as ***VB*** so that the left half of ***B*** becomes the right half of ***VB*** and the right half of ***B*** becomes the left half of ***VB***.

Flip columns of image ***B*** in the left/right direction as ***FB***.

For example,

B = 64 2 3 61 ;
9 55 54 12
17 47 46 20
40 26 27 37

VB = 3 61 64 2 ;
54 12 9 55
46 20 17 47
27 37 40 26

FB = 61 3 2 64
12 54 55 9
20 46 47 17
37 27 26 40

Display images ***B***, ***TB***, ***VB,*** and ***FB*** on figure 2 with ***B*** located at the upper left, ***TB*** located at the upper right, ***VB*** located at the lower left, and ***FB*** located at the lower right. Label each image with its corresponding matrix name (e.g., ***B***, ***TB***, ***VB***, and ***FB***).

Display the maximum, minimum, mean, and median intensity value of ***B*** on the Matlab console.

Matlab hints: rgb2gray, transpose (or '), subplot, max, min, mean, median, fliplr, flipud, flipdim

3. **[4 points]**
Normalize image ***B*** to ***C***, whose data type is **double** and whose values fall in the range of [0, 1]. Display image ***C*** on figure 3 with the message "Normalized Grayscale Image" as the figure title. (Note: Image ***C*** should appear the same as the image ***B***.)

Raise each pixel in the upper quarter rows of image ***C*** to the power of 0.5 and raise each pixel in the lower quarter rows of image ***C*** to the power of 1.5. Keep the middle two quarter rows of image ***C*** unchanged. Store the result as an image (matrix) ***D***. Display images ***D*** on figure 4 with the message "Processed Grayscale Image" as the figure title. **Make sure that no loops are used to accomplish the task.**

On the Matlab console, explain the effects after applying the above two operations.

Save image ***D*** in jpg format to a file called "X_D.jpg" where X should be your first name. Open it using a standard image viewing program to verify that the image is saved properly.

Matlab Hint: double, /, ./, ^, .^, imwrite, display, disp, :

4. **[5 points]**
Perform binary thresholding on the original normalized grayscale image ***C***. A threshold 0.3 is chosen and all values in ***C*** greater than or equal to the threshold are set to 1, otherwise set to 0. Find **two efficient solutions** to obtain the thresholded binary image and save it in ***bw1*** and ***bw2***. **Both solutions should not use any loop**

structure, should not call Matlab built-in function im2bw, and should be distinct in nature.

Use the Matlab built-in function *im2bw* to do the same task and save its thresholded binary image in ***bw3***.

Compare your results ***bw1*** and ***bw2*** with the Matlab's result ***bw3***. If they are equal, display the message "My two methods worked"; otherwise, display the message "One of my two methods or both did not work". Of course, the first message should be displayed when running the program.

Display ***bw1, bw2,*** and ***bw3*** side-by-side on figure 5 and label the three images with "my first method", "my second method", and "Matlab method", respectively.

Matlab Hint: find, >=, zeros, ones, &, &&

5. **[5 points]**
Write a Matlab function **BlurImage** to replace all 16 pixels in each non-overlapping 4×4 block of any input image with their average intensity value. The prototype of **BlurImage** function is as follows:

function [blurredIm] = BlurImage(oriIm); where oriIm is the original image and blurredIm is the blurred image.

Call this function in your main script to blur image ***A***, and save the blurred image to variable ***BA***.
Call this function in your main script to blur image ***B***, and save the blurred image to variable ***BB***.

Display images ***A***, ***B***, ***BA*** and ***BB*** in the raster-scan order (left to right and top to bottom) with the appropriate title on figure 6.

6. Close all figures and clear all variables.
Matlab Hint: close, clear