CSCI 1320 Computer Science I: Engineering Applications – Fall 2017
Instructor: Zagrodzki
Lab 6
Due Friday, October 13 by 11:55pm

Image processing

For Lab 6 and Assignment 7, you will create a simple image-processing program, with functions similar to those found in Adobe Photoshop or The Gimp. Most of the functions you implement for this assignment will take an input image, process the image, and produce an output image.

Note: it would be possible to achieve some the functionality required in this assignment by using built-in Matlab functions, especially from a couple of specialized toolboxes. You are NOT allowed to use these image-specific built-in functions. You will need to code your own implementation of these functions.

Provided files:
A shell script *lab6_hwk7.m* is provided to get you started. Also, a collection of images is provided for testing. You can use your own images as well.

What You Have to Do for the Lab
Implement a menu driven program, similar with the one in the textbook at pages 209-215 (3rd edition). Each button should trigger a call to a function, just like the textbook example. To start with, your program already has the following three buttons:

1. Load Image – loads one image file. In order to select one image, the file needs to be in the same folder as the main script. The loaded image will become the current image and can be passed as an input to the other functions.

2. Display Image – displays the current image.

3. Exit Program – closes the menu and terminates the script.

For Lab 6, you need to implement the functions `makeBright_L` and `makeBright_NL`, and modify the menu to add a new button for each one of these two functions.

Task 1 Brighten: individually modifies the RGB channels of an image, brightening or darkening it. **Use Loops!**
`function [ outImg ] = makeBright_L( inImg, brightness )`
This function brightens each pixel and writes out the new image to `outImg`. The value of the input parameter `brightness` is the amount by which the image

should be brightened, so it can be any integer value in the range [-255 255]. Positive values will brighten the image, while negative values will darken the image. You must use loops to modify each individual pixel values.

Task 2 Brighten2: individually modifies the RGB channels of an image, brightening or darkening it. **NO Loops!**
`function [ outImg ] = makeBright_NL( inImg, brightness )`
This function will achieve the same thing as `makeBright_L`, but without the use of loops.

Task 3. Add menu buttons for the `makeBright_L` and `makeBright_NL` functionality. Choosing one of these two menu items should result in:

a. Asking the user to input a value between -255 and 255 for the `brightness` parameter, followed by a call to the *makeBright_L / NL* function, with the current image and the value of `brightness` as inputs.
b. Displaying the original image and the resulting image, side by side (use subplots)
c. Save the resulting image.

NOTE 1: Every one of the functions you need to develop for Lab 6 and Assignment 7 should return a **uint8 array** called `outImg` , which should have all element values in the range [0,255].

NOTE 2: For some of the functionality of this program, we will ask you to implement the same tool both using the programming method (loops) and using the Matlab method (vectorization). When testing the function version that uses loops, we recommend you choose as your current image one of the "small" versions provided: *Lena1_small.jpg* vs *Lena1.jpg*. Otherwise, the function call will take a noticeable longer amount of time to return.

Submitting the assignment:
Make sure each script or function file is well commented and it includes a block comment with your name, course number, assignment number and instructor name. Zip all the .m files and the image files together and submit the resulting *.zip* file through Moodle as Lab 6 by Friday, October 13 by 11:55pm.

Note: we will not accept any submissions that are not archive files. If you submit a pdf, you will not get any credit for your work