Project 1
CSC 9010 - Computer Vision
Grading: 160 points
Due Date: Feb. 5, 2015

Description: As the digital world is rapidly expanding, image processing is becoming more prevalent and important. For example, Instagram was acquired for $725 million dollars using only the basic techniques that you will be performing in the project. You will be programming several image processing algorithms to enhance images using Matlab. This lab is meant to reinforce the basic data structures, variables, and methods that you have learned thus far.

There are many built in commands in Matlab that perform these operations for you. However, you will not be using these built-in commands, but rather writing the algorithms from scratch. I will provide the built in commands so that you may check your answers, and so that these Matlab commands are reinforced. You must have the image processing toolbox from Matlab installed.

Part 1 - Grayscale (20 points)

(a) Original image. (b) Grayscale result.

Figure 1: CSC 9010 Computer Vision - grayscale sample result for Part 1.

Grayscale is an image processing technique that takes a 3 channel RGB image and turns it into a single channel of gray values. The color image can be represented in vector form as $< R, G, B >$. The range of values that a pixel may take is 0-255. To properly convert a color image into a grayscale image, one must use the following formula:

$$Y = 0.2989R + 0.5870G + 0.1140B \tag{1}$$

where R,G,B represent the intensity values of the red, green, and blue channel respectively, and Y represents the grayscale value. Use matlab to obtain the 3 channels into separate variables and perform matrix-scalar multiplication to get the final result.

Built-in Matlab Command - You can test your result using the built-in matlab command,

I = rgb2gray(RGB)

converts the truecolor image RGB to the grayscale intensity image I. The rgb2gray function converts RGB images to grayscale by eliminating the hue and saturation information while retaining the luminance.

Part 2 - Contrast stretch (20 points)

(a) Original image. (b) Contrast result.

Figure 2: CSC 9010 Computer Vision - Contrast stretch with the values $c = 0$ and $d = 180$

Contrast stretching (often called normalization) is a simple image enhancement technique that attempts to improve the contrast in an image by "stretching" the range of intensity values it contains to span a desired range of values, e.g. the the full range of pixel values that the image type concerned allows. Contrast stretching takes the following form,

$$< \hat{R}, \hat{G}, \hat{B} >= (< R, G, B > -c)\frac{(b-a)}{(d-c)} + a \tag{2}$$

where a and b are the lower and upper limits of the image, respectively. In our case, these limits will be normally be set to 0 and 255. c and d represent the mapped range such that a maps to c and b maps to d. Prompt the user to provide the values for c and d.
Built-in Matlab Command - You may test your result using the built-in matlab command,

J = imadjust(I,[low_in; high_in],[low_out; high_out])

maps the values in I to new values in J such that values between low_in and high_in map to values between low_out and high_out.

Part 3 - Gaussian blur (20 points)

A Gaussian blur (also known as Gaussian smoothing) is the result of blurring an image by a Gaussian function. It is a widely used effect in graphics software, typically to reduce image noise and reduce detail. Mathematically, applying a Gaussian blur to an image is the same as convolving the image with a Gaussian function. The Gaussian function is defined as,

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3}$$

You will create a 5x5 Gaussian kernel and convolve the kernel with the image to produce a blurred result. You can assume that $\sigma = 1$ for all cases.
Built-in Matlab Command - You can test your result using the built-in matlab command,

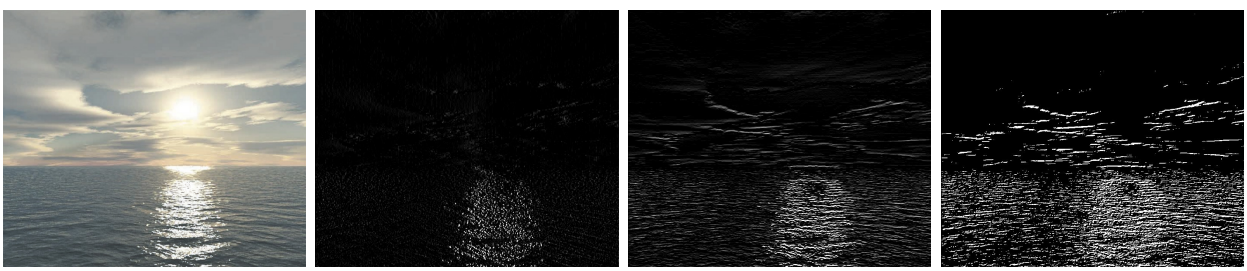
(a) Original image. (b) Blurred result.

Figure 3: CSC 9010 Computer Vision - Gaussian blur with $\sigma = 1$ and 5x5 kernel.

h = fspecial('gaussian', hsize, sigma); blurred = imfilter(I,h);

filters the multidimensional array I with the multidimensional filter h. The array I can be logical or a nonsparse numeric array of any class and dimension. The result blurred has the same size and class as I.

Part 4 - Sobel edge detection (20 points)



(a) Original image. (b) G_x. (c) G_y. (d) G_{thresh}.

Figure 4: CSC 9010 Computer Vision - Edge detection using the Sobel filter. G_{thresh} is computed at threshold = 50.

The Sobel operator, sometimes called Sobel Filter, is used in image processing and computer vision, particularly within edge detection algorithms, and creates an image which emphasizes edges and transitions. The Sobel filter is defined as the following 3x3 matrix,

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}, G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \tag{4}$$

You may implement the Sobel edge operator either through a straight correlation of the 3x3 matrix or you may use seperable filters to accomplish this task (which are slightly more efficient). Prompt the user to provide a threshold for the final binary image. Note, your threshold image may

look different than the built in matlab thresholding method, which is ok.

Built-in Matlab Command - You can test your result using the built-in matlab command,

BW = edge(I,'sobel',thresh)
specifies the sensitivity threshold for the Sobel method. edge ignores all edges that are not stronger than thresh.

Part 5 - Color transfer (40 points)

(a) Original image. (b) Target image. (c) Color transfer result.

Figure 5: CSC 9010 Computer Vision - Color transfer result borrowing the colors from the target image and applying it to the source image.

The article, "Color Transfer between Images" describes a method for a more general form of color correction that borrows one image's color characteristics from another image. You will be converting the image to LAB space which minimizes correlation between channels for many natural scenes. This space is based on data-driven human perception research that assumes the human visual system is ideally suited for processing natural scenes. Follow the procedure carefully in that article. There is no built-in matlab command for this.

Part 6 - PDF write up (20 points) Create a 2 page write up that summarizes the interesting parts of your program and includes results. Include any problems or insights that you encountered.

Deliverables: Submit on Blackboard.