CMSC5711 Image Processing and Computer Vision
Assignment 2
(Total 100 points)

Part I

1. (10 points) Given $I = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}$ and $h = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$, you are required to implement the following tasks. Compute $I * h$, where $*$ is the convolution operator. Show the steps of your calculation. (Hints: Try to use correlation to implement the above convolution. You should first find the flipped version of h and correlate it with I. In this question, I and h are partially overlapped.)

2. (15 points) Suppose there is an image $I = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 255 & 255 & 0 & 0 \\ 0 & 255 & 255 & 0 \\ 0 & 0 & 255 & 255 \end{pmatrix}$, please compute the corresponding edge image of I if the Prewitt mask is used and the threshold is set to be 200. Please show us the detailed steps. (Hints: Prewitt has the following properties: $x_{win} = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix}$, $y_{win} = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}$.)

3. (15 points) You are required to find the equation of a circle passing through the following edge points: $(x, y, \theta) = (1, 1, -90^o)$, $(3, 3, 0^o)$, $(1, 5, 90^o)$ where θ means the angle between the horizontal axis and the line perpendicular to the tangent of the edge. Create a (r, x_c, y_c) table for these edges and use Hough Transform to solve this problem.

Part II

4. (60 points) It's a fundamental task in computer vision applications to detect basic geometric shapes. In this exercise, you are required to write programs to detect basic shapes, including lines, circles, rectangles as well as triangles. Edge detection and Hough Transform approaches are the most frequently used techniques for feature extraction and object detection. In this assignment, you are suggested to apply these two techniques into shape detection. Through this exercise, you are expected to:

   i). Get familiar with MATLAB, especially matrix operations.

ii). Learn to deal with images in MATLAB, such as read an image, display an image, and transform an RGB image into a gray image.

iii). learn to write a function in MATLAB and call this function.

iv). Enhance your understanding about edge detection and Hough Transform.

Background: Follow the steps shown below to learn how to use these MATLAB functions. If you are familiar with them, you can skip this part.

i). Study and get familiar with basic image operations, such as *imread(), rgb2gray(), imshow()* and etc. You can make use of the MATLAB Help file or refer to the reference website provided by Mathworks.

ii). Study basic matrix operations, including *size(), zeros()* and get to know how to access elements of a matrix and how to assign values to elements of a matrix.

iii). Learn to build a function in *.m* files and call it from other *.m* scripts or functions.

Implementation: Suppose you are given an image in which some basic shapes are drawn, then you goal is to detect these basic shapes and classify them based on their geometric properties. There are totally three types of shapes, including circles, rectangles and triangles. These shape are non-overlapping in the image. Finally, you need to render these detected shapes with different colors according to their classes. You may rely on edge detection techniques and Hough transform approaches to implement this task. One sample image and the corresponding expected result are shown in Figure 1 and Figure 2. Detailed steps are shown below:

i). Detect edges of the given image using the Sobel, Prewitt or Canny method (refer to the function of *edge()*).

ii). Apply the Circle Hough Transform method to detect circles in the image (refer to the function of *imfindcircles()*). After detection of circles, you can render them with a predefined color for circles.

iii). Find boundaries in the edge image (refer to the function of *bwboundaries()*). In this way, you can extract several subregions using what returned by *bwboundaries()* and analyze each individual subregion respectively. (Refer the function of *[B, label] = bwboundaries(edg)*. You can use the output B to analyze individual boundaries in the following step while the output *label* is helpful when you try to render the image.)

iv). Use Line Hough Transform to detect lines in each boundary (refer to the function of *hough()* and *houghpeaks()*). If the number of lines detected in some boundary is 3, it means that the shape inside this boundary is a triangle. Otherwise if the number of lines is 4, it indicates that this shape is a rectangle. By counting the number of lines detected in each boundary, you are able to judge what kind of shape this boundary belongs to.

v). Render the image using the information above.

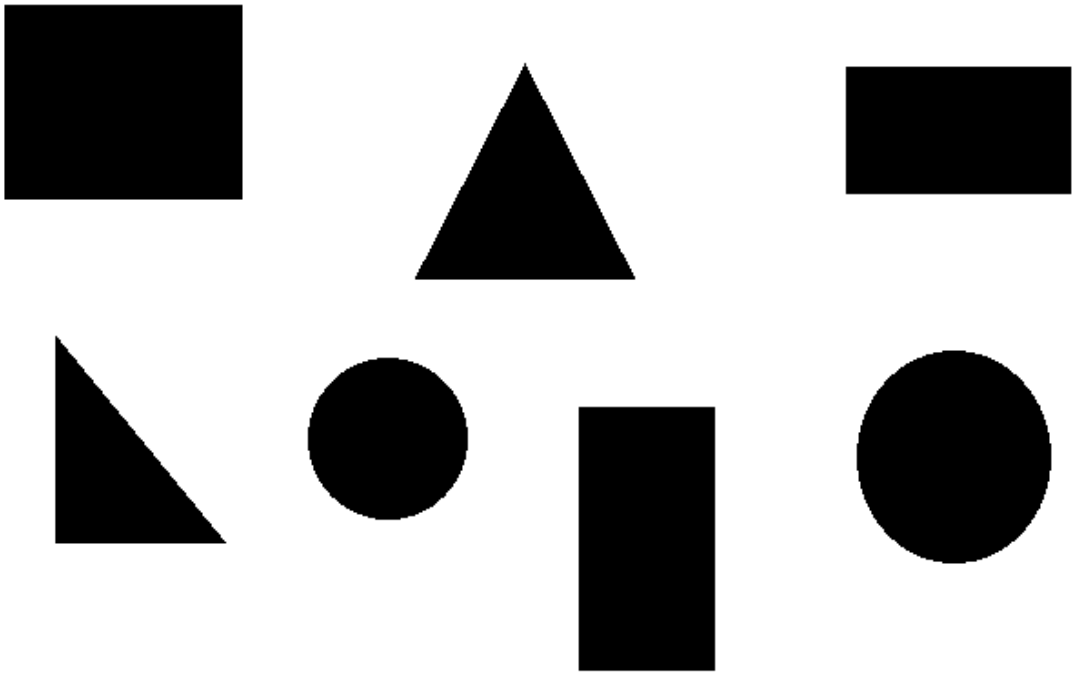

Figure 1: original image

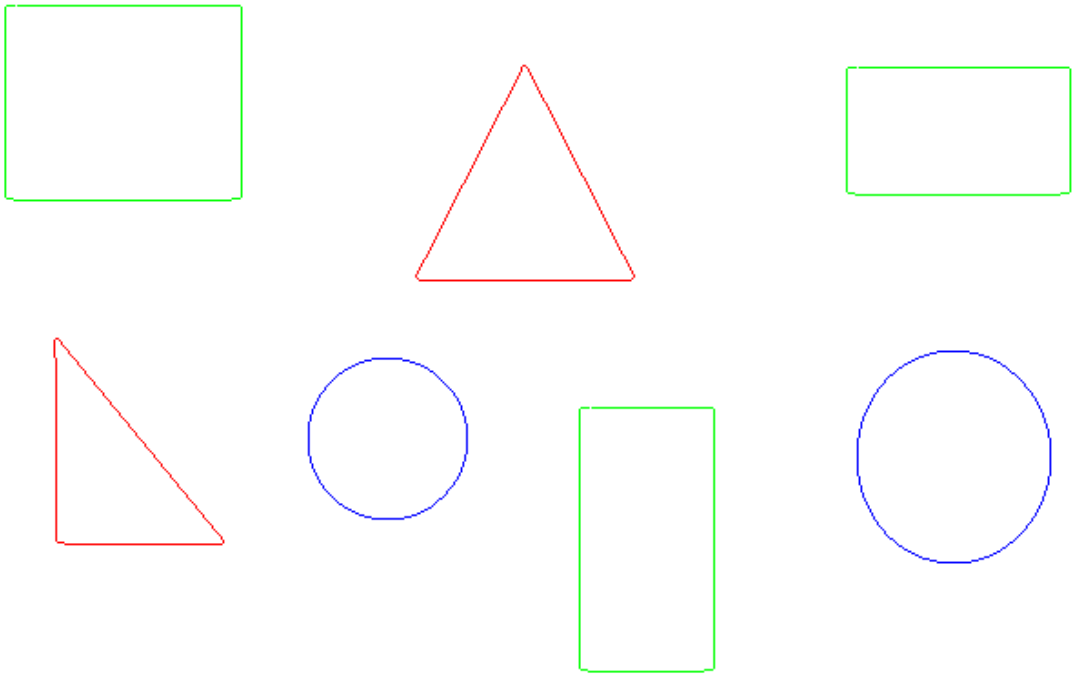

Figure 2: rendered image

Unfortunately, the function of *imfindcircles()* for detecting circles is only available in the image processing toolbox of the latest version of MATLAB. Thus for some of you, the step ii). above may not be applicable. In this case, there are two solutions to overcome this problem. One solution is that you can try to write a function for circle detection using

Hough Transform by yourself. This is highly recommended for students and you will be awarded with higher marks for that. Another way is to avoid detecting circles. Since circles can be regarded as a polygon with infinite sides, when applying Line Hough Transform to analyze the boundaries and you can find that the number of sides returned by a circle is much larger than that returned by a rectangle or triangle. That is to say, you can apply Line Hough Transform to detect circle instead of using Circle Hough Transform. In this way, the above step ii). can be removed, but you need to do more analysis after you get the boundary of each object and judge what kind of shape it belongs to.

Expected Results: Synthesize three more images as Figure 1 by yourself and run your shape detection program on your synthesized images. Submit your results on the Figure 1 together with those on your synthesized images to us.

Requirement and Submission

You should submit a brief report to *cmsc5711.2017@gmail.com* before the deadline listed on the website. Your report should include: (i) Written answers on part I; (ii) Programming results of detailed steps on part II ; Pack all the things in a *.zip* package. Both the email title and *.zip* file title should be *Assignment_2_your student ID*. You could discuss with each other about the implementation, but note that plagiarism is strictly forbidden in this course.