CS 6643 **Fall 2021**
Project 1: Canny's Edge Detector **E. K. Wong**

Total number of points = 100. You can work on the project by yourself or work in a team of two. You can discuss with your classmates on how to do the project but everyone or every team is expected to do their own coding and turn in their own report, source code and output image results.

Project Description: Implement the *Canny's Edge Detector* as described in the lecture slides. The Canny's Edge Detector consists of four steps: *Gaussian smoothing, gradient operation, non-maxima suppression* and *thresholding.* The input to your program is a grayscale image of size $N \times M$ (rows × cols.) Use the 7 × 7 Gaussian mask below (page 2) for smoothing the input image. Use the center of the mask as the reference center. If part of the Gaussian mask goes outside of the image border, let the output image be undefined at the location of the reference center. Note that the entries in the Gaussian mask do not sum to 1. After performing convolution (or cross-correlation), you need to perform normalization by dividing the results by the sum of the entries (= 140 for the given mask) at each pixel location. Instead of using the Robert's operator, use the Prewitt's operator to compute horizontal and vertical gradients. If part of the 3 × 3 mask of the operator goes outside of the image border or lies in the undefined region of the image after Gaussian filtering, let the output value be undefined. For the third step, follow the procedure in the lecture slides for non-maxima suppression. At locations with undefined gradient values and when the center pixel has a neighbor with undefined gradient value, let the output be zero (i.e., no edge.) For the fourth step, use simple thresholding and produce three binary edge maps by using three thresholds chosen at the 25th, 50th and 75th percentiles of the gradient magnitudes after non-maxima suppression (exclude pixels with zero gradient magnitude when determining the percentiles.)

S - Task 1

E - Task 2

E Task 3

S Task 4

Design your program in a modular fashion and create separate program functions for the four steps: *Gaussian smoothing, gradient operation, non-maxima suppression* and *thresholding.* Generate the following *normalized* image results with an output range of [0,255]. For image locations with undefined values as described above, replace with value 0. The output images should be of the same size as the original input image. (1) Normalized image result after Gaussian smoothing. (2) Normalized horizontal and vertical gradient responses (two separate images.) To generate normalized gradient responses, take the absolute value of the results first and then normalize. (3) Normalized gradient magnitude image. (4) Normalized gradient magnitude image after non-maxima suppression. (5) Binary edge maps using simple thresholding for thresholds chosen at the 25th, 50th and 75th percentiles.

You can use Python, C++/C, Java or Matlab to implement your program. If you would like to use another language, send me an email first. You are not allowed to use any built-in library functions to perform any of the steps that you are required to implement, including the convolution (or cross-correlation) operation. The only library functions you are allowed to use are: *reading* and *writing* image files, *matrix* and *vector multiplications*, computation of *percentile values*, and other *commonly used math operations*.

Testing your program: Two grayscale test images of size $N \times M$ in bitmap (*.bmp*) format will be provided on *Brightspace* for you to test your program.

Hand in the following on Brightspace by the due date:

(a) Your source code file. Put comments in your source code to make it easier for someone else to read your program. Points will be taken off if you do not have comments in the source code.
(b) The output image files (in *.bmp* format) generated by your program for (1) to (5) above. There should be a total of 16 output image files for the two test images.
(c) A PDF file that contains instructions on how to run your program. If your program requires compilation, instructions on how to compile your program should also be provided. Also, copy

and paste the output images and your source code onto the PDF file (to make it easier for us to grade your project.) This is in addition to the source code file and output image files that you need to submit separately (as described in (a) and (b) above.)

7×7 Gaussian mask

1	1	2	2	2	1	1
1	2	2	4	2	2	1
2	2	4	8	4	2	2
2	4	8	16	8	4	2
2	2	4	8	4	2	2
1	2	2	4	2	2	1
1	1	2	2	2	1	1