Assignment for Module Two

Here you will implement a complete Canny algorithm for edge detection. This will be achieved by putting together the parts that have been discussed in the lectures.

Start by getting your Part One code all together. You need to follow the slides and the video that explains the code (starts around 10:30 in video2 for Part One, of Module 2: Canny Edge Detector Part One). This involves changing the marrh.c code and blending it with the sobel.c code. You will know that you are done with this Part One when you get output similar to what was produced as Sobel's magnitude image. You must test your code on the chess picture stored in garb34.pgm available at

http://www.cs.ucf.edu/courses/cap4453/inputpics/ .

Use a value of 1.0 for sigma.

Your output should look like the pic called cannymag.pgm available at
http://www.cs.ucf.edu/courses/cap4453/outputpics/

Next, to do Part Two, you need to add in the code for peaks that is described in the lecture video (starts around 4:43 in Video3 for Part Two, of Module 2: Canny Edge Detector Part Two). This will end up producing for you an output image that should look like cannypeaks.pgm available at

http://www.cs.ucf.edu/courses/cap4453/outputpics/

Make sure that for Part Two, you use array names that are consistent with the names used in Part One.

There are many possible sources of error at this step, so you must be very careful to implement things correctly. Keep improving your resulting image (peaks), until your image looks like cannypeaks.pgm (as in the paragraph above). There are many illusory images that will look somewhat like cannypeaks.pgm, these are not acceptable; so you must try hard to get the correct answer.

Next, you will implement the code for Part Three. You will need to manually enter two thresholds (HI and LO; select HI close to 100, and LO close to 35), and implement the code shown in the lecture video (look at the start of video2 for Canny Part Three, of Module 2: Canny Edge Detector Part Three). Your Final Edges output will look something like the picture cannyfinal.pgm available at

http://www.cs.ucf.edu/courses/cap4453/outputpics/

Lastly, you will implement Part Four, but will squeeze it in before Part Three in the code, so that you do not need to have inputted HI and LO (instead, you will now input a value for Percent). The code for this is described in the lecture video (starts at about 8:00 for video1 of Canny Part Four, of Module 2: Canny Edge Detector Part Four). The output will look the same as for Part Three, the only difference is that: now, you should not have to input HI and LO, instead use Percent (give it a value of about 0.5 percent, i.e., i.e., 0.005). Good Luck, just email me if you need help.

Your canny program will have to be able to read in any picture file (.pgm), and will produce three output pictures (magnitude.pgm, peaks.pgm, and Final Edges.pgm). Be prepared to run this program on garb34.pgm, and to show the grader in/during a conference that it runs correctly.