Lab Assignment #4
Due *Tuesday*, November 13 by 11:59pm
(upload to CatCourses)

Title: Edge detection

In this lab, you will implement a simple edge detection scheme. You will use the magnitude of the gradient to determine whether a pixel belongs to an edge or not. The gradient will be implemented using linear spatial filtering.

a) Write the function `spatial_filter` that takes two inputs: a grayscale image (a matrix, not a filename) and a filter (also represented as a matrix). This function applies the filter to the image as described in section 3.4.1 of the in the 3rd edition of the text or section 3.4 of the 4th edition. Specifically, it should implement the equation at the bottom of page 145 of the 3rd edition or equation 3.4 of the 4th edition. Your function should output an image (a matrix, not a file) **of type double** with the same dimensions as the input image. You can assume that the filter is odd sized. Use zero padding to extend the image when the filter goes beyond the boundaries of the image. Write a script to test your `spatial_filter` function (you do not need to submit this test script). A good initial test is that filtering an image with a spatial filter which has all zeros except for a one in the middle should return the same image (and thus is the identity filter).

b) Write the function `gradient_magnitude` that takes one input: a grayscale image (a matrix, not a filename). This function should return an image (a matrix, not a file) **of type double** with the same dimensions as the input. This output contains the magnitude of the gradient. You should compute the gradient using the Sobel masks shown in figure 10.14 (both editions) using your `spatial_filter` function.

c) Write the function `find_edges` which takes two inputs: a grayscale image (a matrix, not a filename) and a threshold value (a scalar). This function should return an image **of type uint8** with the same dimensions as the input. Your function should detect edges by applying a threshold to the magnitude of the gradient as computed using your `gradient_magnitude` function. The output should have value 255 where edges are detected and 0 elsewhere.

In summary, `find_edges` calls `gradient_magnitude` which makes calls to `spatial_filtering`.

Apply your `find_edges` function to the watertower.tif image available at the course website (on the schedule page). Experiment with different threshold values to find a value which gives a reasonable edge image.

Submit:

- The code for your `spatial_filter`, `gradient_magnitude`, and `find_edges` functions.

- Your best edge image and the threshold value used to compute it (you don't need to find the absolute best image, just something that looks like my result below).

watertower.tif
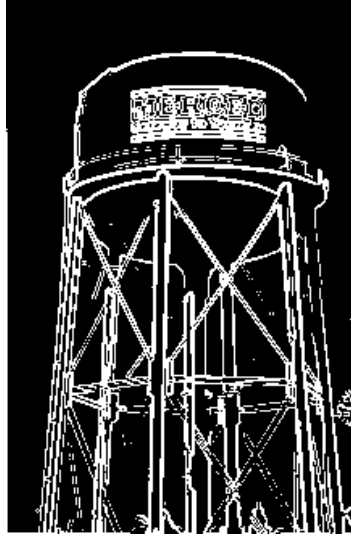
My result.

In your report, discuss:

- The effects of different threshold values.
- Compare your results with those of using the built-in Matlab implementation of the Canny edge detector. Do a help on "edge" in Matlab to determine how to perform Canny edge detection. Explain the difference between your results and those of the Canny edge detector based on what you learned about it in lecture and from the text.