Topic 3: Edge Detection

Unless otherwise noted, when code is referenced, it refers to the file “edgedet.m”, which is available on the SuMAc website.

1. Look at the section of the code with heading “Local Rates of Change”. The first line of code in this section provides us with the domain for x, specified by 0 and 10 as well with how close together we will evaluate the function (specified by 0.1).

   a. Using the default values for x, try changing the value used for *c* so that there are only one or two consecutive white pixels in Figure 7. In other words, find the “edges” for this function.

   Let us now explore what happens if we change how close together the x values are. That is, change the 0.1 in “x=0:0.1:10” to the following values:
   0.01, 0.05, 0.2, 0.5, 1, 2, 2.5, and 5.

   b. Repeat part a for each of these values. Are you able to keep both “edges” for each of these values while subject to the constraint on the number of consecutive white pixels? For that matter, are you able to find both “edges”? NOTE: You may need to comment out the “axis equal off” command for Figures 5, 6, and 7.

   c. Based on what you saw in part b, does it seem like there is some relationship between how close the x values are and the value for the threshold *c*? If so, what does it appear to be?

2. Repeat problem 1, but change the function to “y=abs(tan(x));”

3. Are the results similar for problems 1 and 2? What is different in nature about the two functions used to calculate *y*?

4. Repeat problem 1, but add noise to *y* using the command “normrnd(0, 0.5)”.

5. Repeat problem 2, but add noise to *y* using the command “normrnd(0, 0.5)”.

6. How does the noise affect the results in problems 6 and 7?

7. Refer to the code in the section headed by “Edge Detection”. Use both edge detection methods for the following images.

   a. fabric.png
   b. blobs.png
   c. AT3_1m4_01.tif
   d. office_1.jpg
   e. office_6.jpg
   f. saturn.png
   g. testpat1.png

   For each image, which edge detector do you like better? Do either seem particularly useful?

   NOTE: The images can be read into matlab using the command “imread”. Some of these are RGB images and will need to be converted into grayscale using the command “rgb2gray”.

8. Refer to the sections of the code headed by “Image Histograms” and “Image Thresholding”. For each of the images in problem 7, do the following:

   a. Compute the image histogram and examine it.
   b. If the image histogram has at least two separate peaks like we discussed earlier, use the thresholding procedure from the code.
   c. If you decide to threshold the image, perform both methods of edge detection on this new image. Which method do you think works better? Did the thresholding help the edge detection?

   NOTE: Again, some images will need to be converted into grayscale.

9. Repeat problem 7, but add (i) Salt & Pepper noise and (ii) Gaussian noise to the images. How do