Tim Herrmann
EECS 332
MP5
Canny Edge Detection

Programming Language:

- Python 2.7.6 with the following dependencies
  - matplotlib
  - numpy
  - cv2
  - pylab

Input:

Image – grayscale or color

Output:

A grayscale image showing only the edges.

Method:

Step 1: Read the image into a grayscale format
Step 2: Apply a Gaussian Smoothing. Sigma can be set as a global variable at the top of the program.
Step 3: Calculate Image Gradient. I performed this calculation using the Sobel technique in both the x and y directions and taking the combined magnitude. Theta is also calculated in this step by taking the inverse tangent of the two directional values.
Step 4: Select High and Low Thresholds. The high threshold is determined by creating a histogram with a bin count equal to the maximum magnitude counted in step 3. This ensures each bin has only one magnitude value. Using the cumulative sum, the threshold is determined as the value at which the cumulative sum is greater than the given percentage. The percentage can be changed as a global variable at the top of the program. The Low Threshold is set to .5*T-high.

> *Note*: in order to gain more granularity with threshold values for sterile images such ast test1.bmp. I removed all 0 magnitude values from the cumulative distribution before calculating the threshold. This is a great help for images where the majority of the pixels have 0 gradient*

Step 5: Suppressing Nonmaxima. Using the quantization method and a LUT, the local maxima are calculated. First, the theta value is used to determine which pixels to consider. Then the current pixels are compared to its two neighbors along that direction. If the pixel is greater than or equal to both, it is kept as part of the "ridge". If not, it is set to 0.
Step 6: Low Thresholding. Any remaining values in the magnitude array that are less than the low threshold are set to 0 prior to the edge linking step.
Step 7: Edge Linking. This function iterates through all of the pixels. When it finds a pixel that is considered a "strong edge", meaning its magnitude is greater than T_high, it calls a function "checkpixel" that marks the pixel as a final edge and checks all of its neighboring pixels for values greater than T_low. If it finds any, it recusrsively calls itself, repeating the process of marking that pixel as a final edge and checking its neighbors. The end result is a binary array identifying all the pixels marked final edges.
Step 8: The output from Step 7 is converted to an image and saved.

Examples and Analysis:

Full Example of All Steps for Lena.bmp (sigma = 5, threshold = 80)

Original Image

Gaussian Smoothing

Visualization of Gradient Magnitude

Non-Maxima Suppressed Gradient

Edge Connection and Thresholding

Parameter Variations

Sigma = 5

Threshold = 70%

Threshold = 80%

Threshold = 90%

Lower thresholds result in more edges, but also significantly more noise. Higher thresholds limit the noise, but miss many of the edges. 80% seems to be a good compromise for this image and sigma value.

Threshold = 80%

Sigma = 1

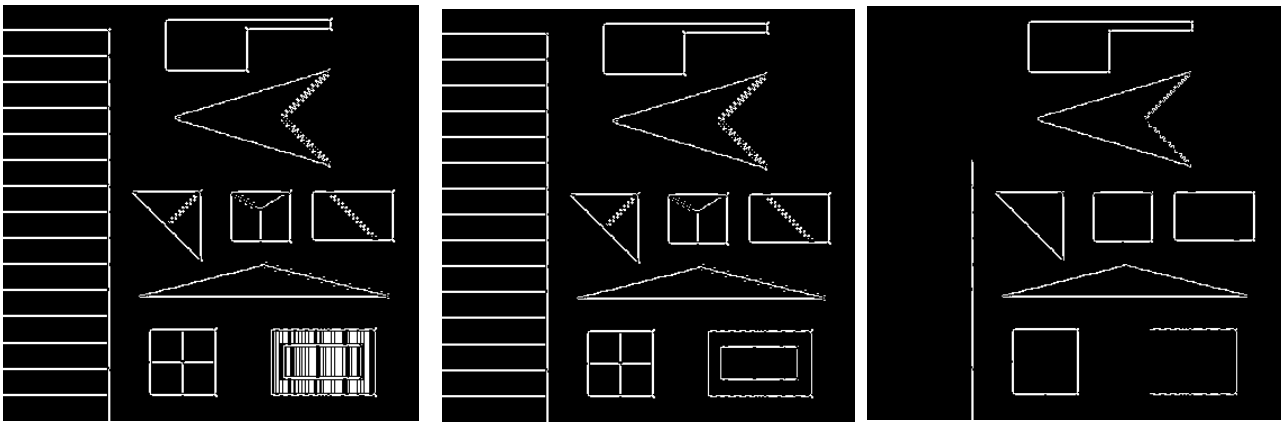
Sigma = 2

Sigma = 5

Although not as noticeable as changing the threshold, you can see small differences based on sigma values. One noticeable spot is the top of the hat where lower sigma has a bit more noise.

Test1.bmp Thresholding

Threshold = 10 | Threshold = 25 | Threshold = 80



Thresholding for a sterile image such as test1.bmp is more much more difficult as seen above. Changing the threshold slightly can make a big difference.

Additional Images