Programming Assignment Report

Ran Tian 49070588

The main contend of Sobel and Roberts edge detection are the same except for the kernel matrix, So I just analyze the case for Sobel, same logic also apply for Roberts.

**Part 1:**

**Step1:** Get the gray scale matrix representation for image, use rgb2gray() in Matlab.

**Step2:** Convolve the edge operators(kernel\_x,kernel\_y) with gray scale image, we end up with getting delta\_x and delta\_y.

**Step3:** Given delta\_x and delta\_y, we compute the magnitude and direction for each element, Notice for direction matrix we can directly calling Matlab build-in function atan2().

**Step4:** Normalization Step, since we deal with image in gray scale, the value for each pixel is in range(0,255). I choose to lineally map the original range to (0,255). To be more detailed:

For magnitude matrix, firstly we found the maximum value and compute the ratio(max\_value/255), then for each element, the normalized value is just (original/ratio).

For direction matrix, since there are negative values, we firstly find the minimum value and maximum value, then the gap= max\_value-min\_value. For each element, the normalized value is (original – min\_value)\*255/ratio.

**Step5:** Thresholding, here I choose the thresholding value **manually**, which means firstly I look into the histogram of image pixels and then set up the threshold value. For my code, the magnitude threshold value is **20**, and the direction threshold value is **165**(or the ratio is 0.65).

Then we get two images after part 1: magnitude &direction:

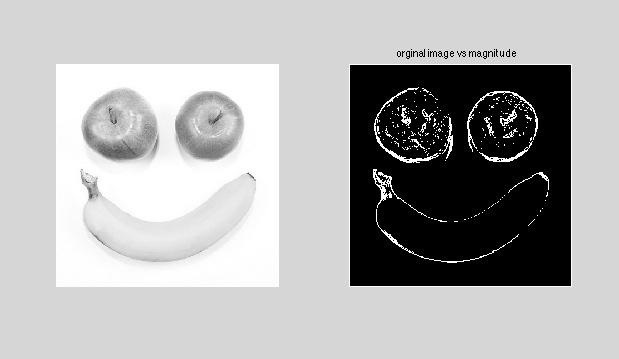


Figure 1 Magnitude Image



Figure 2 Direction Image

From the figures above, the result of magnitude is better than the direction matrix.

**Part 2:**

**Step1:** Expansion: we will fill in the gap for the magnitude matrix from part 1.

The way here I use is to count the 8 neighbors of the element 0, if count >= 3,we make it to 255.

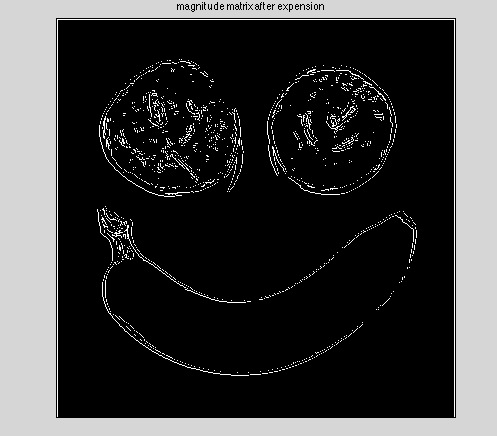


Figure 3 Magnitude Matrix after Extension

**Step2:** Thinning: Here I do 4 passes from the extension matrix from last step, from left->right, right->left, top->down, down->top, for each pass, I will remove the element if any of 3 conditions satisfy:

1: Number of 255’s for its neighbors is less than 3

2: The element don’t have N,S,E,W neighbors depends on which pass.

3: The diagonal neighbors of the element are not 255 depend on which pass.

Based on these 3 criteria above, the thinning image looks like:

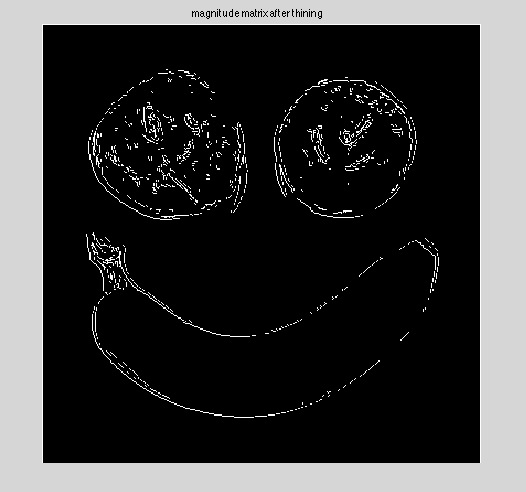


Figure 4 Magnitude Matrix after Thinning

Here is the Matlab build-in Sobel edge detection result:

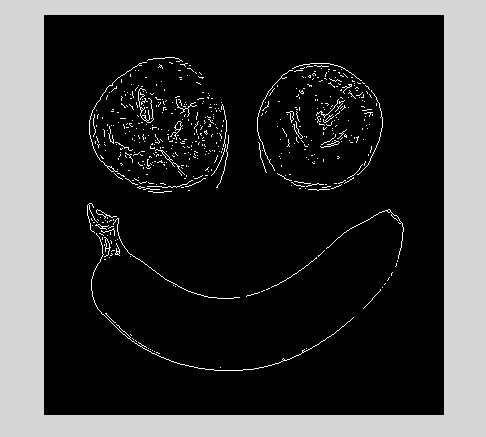


Figure 5: Matlab Build-in Sobel Operator

Compare with the Matlab build-in operator, our result lose some of edge information. My guess for this difference is on threshold and thinning algorithm, Here I set threshold manually and only pass the matrix in 4 directions for thinning, which may not good enough. But in general, my sobel edge detection shows the functionality.