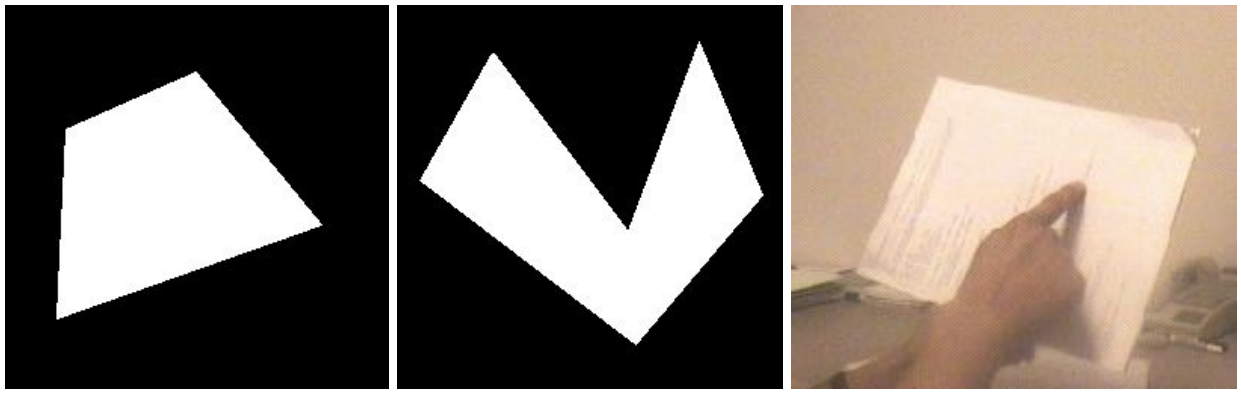
Kyle Qian

Hough Transformation

For this project we worked on implementing the basic ideas of a Hough transformation for line detection. The images we used to test our algorithm are shown below:



These images will be referred to as test, test2, and input from left to right respectively.

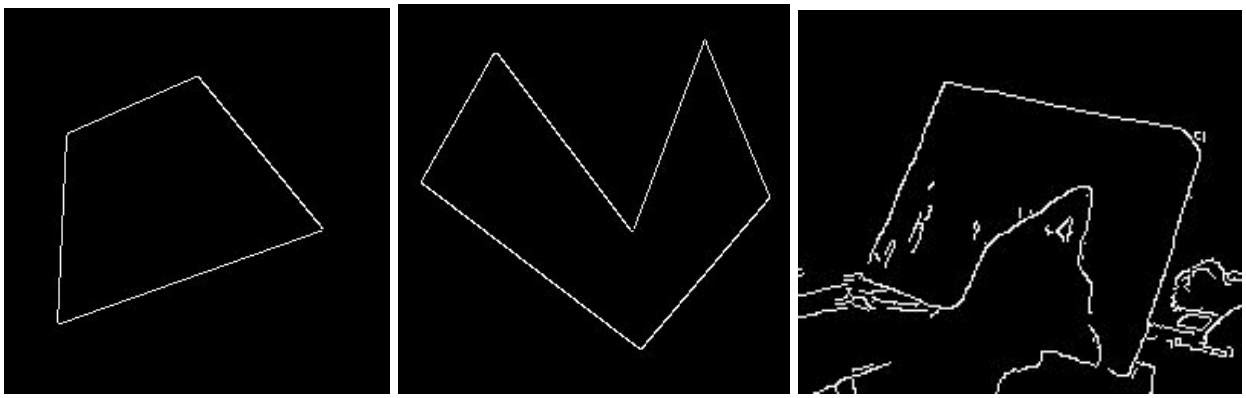
The Algorithm

This algorithm can be broken up into several parts:

1. Edge Detection
    - Perform edge detection on grayscale images (ex. Canny Edge Detection)
2. Convert non-zero points in edge detection image from xy space to parameter space
    - Use polar parameter space, choose how to quantize
    - $p = x * cos\theta + y * sin\theta$
3. Use thresholding, CCL, and non-maxima suppression to identify singular local maxima points
    - Each of these points will represent a different line
4. Convert these points back into lines on the xy plane

Implementation Results

- Edge Detection

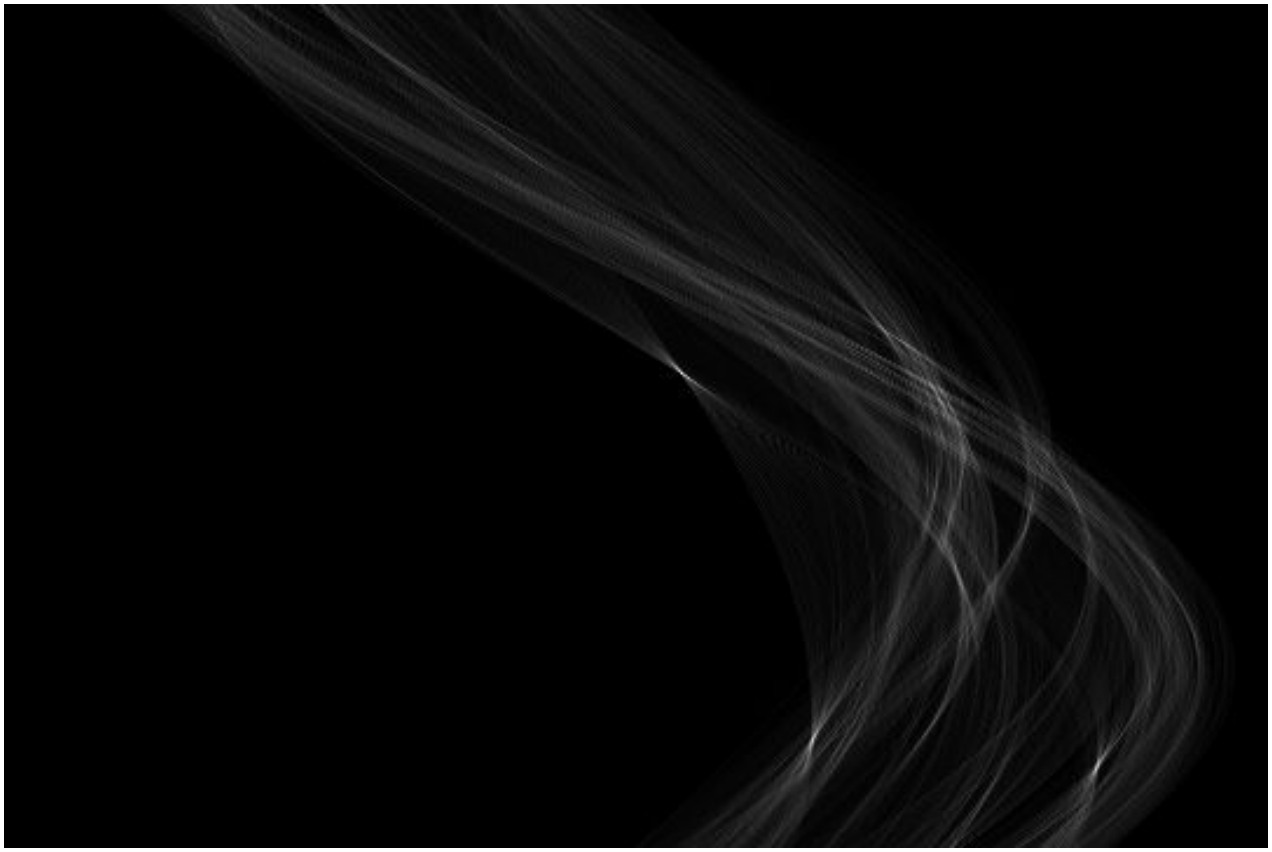


Here are our testing images— test, test2, and input— after using Canny Edge detection.

- Convert from xy space to parameter space

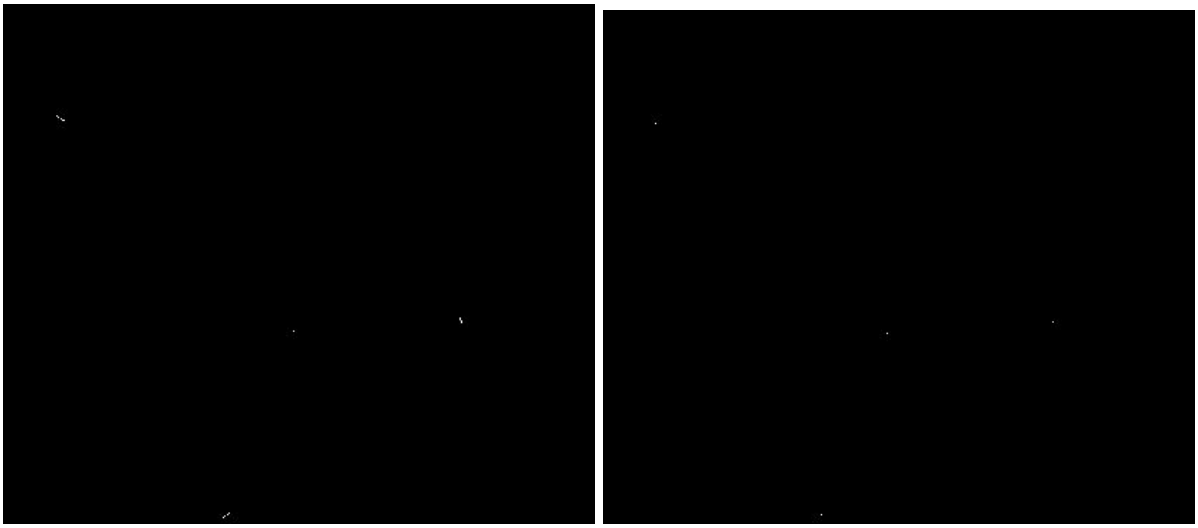
Here is the result for test.bmp. As we can see there are approximately 4 bright spots that represent the 4 lines that should be generated for test.bmp. (One is harder to see, it's at the bottom)

Here is the result for test2.bmp. As we can see there are approximately 6 bright spots that represent the 6 lines that should be generated for test2.bmp.



Here is the result for input.bmp. There are a few clear areas where we can see a local maxima should exist, but because of input.bmp's greater complexity we have a much more complicated parameter space.

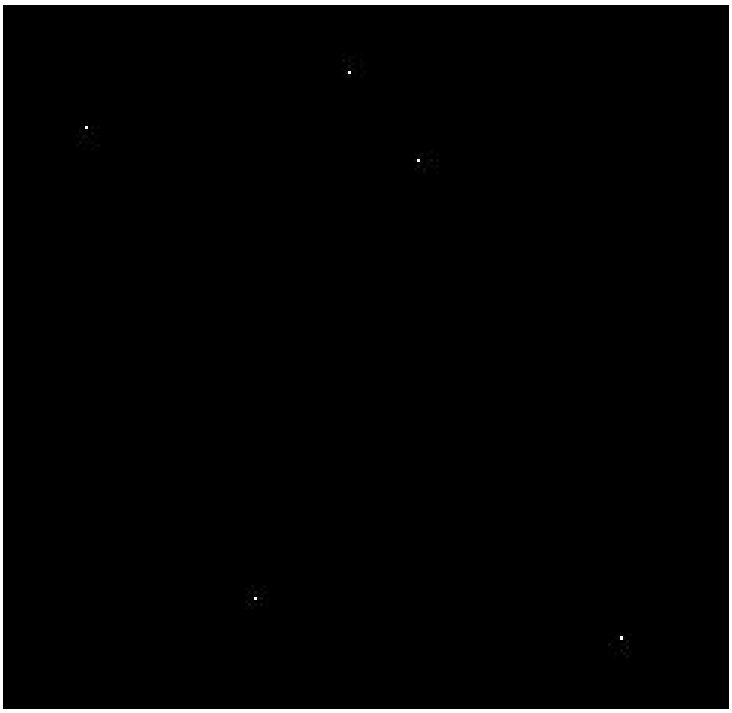
- Thresholding, CCL, and Non Maxima suppression



Here is the thresholding and maxima suppression images for the image test.bmp, left and right respectively. They are cropped to fit together better. We arbitrarily choose a threshold of 'votes' needed to be included in the threshold image, then perform CCL and non maxima suppression

on each labeled component. As we can see, the small regions of white pixels in the threshold image become singular pixels of white in the maxima suppressed region. For this image, we get 4 local maximas, which will correspond to 4 lines in the original image.

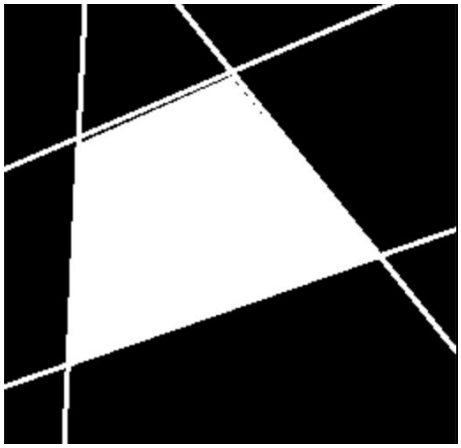

Above are the thresholding and maxima suppression images for the image test2.bmp, left and right respectively. The same process for test.bmp was used here, but we now have 6 local maxima points which will correspond to 6 lines in the original image.

Above are the thresholding and maxima suppression images for the image input.bmp, left and right respectively. The same process for test.bmp was used here, but we now have 5 local

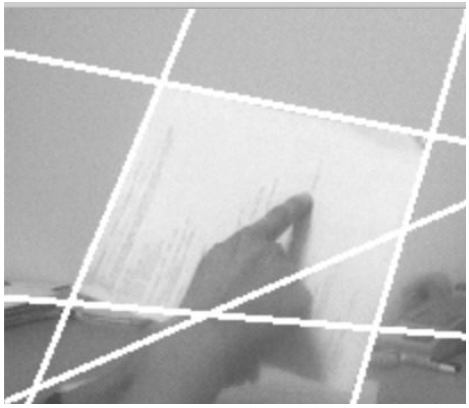
maxima points once again. This will correspond to 4 lines in the original image.

- Line Detection



Here is our result for test.bmp. As we can see, there are four lines that each correspond to a local maxima that we found earlier.

Here is our result for test2.bmp. As we can see, there are six lines that each correspond to a local maxima that we found earlier.



Here is our result for input.bmp. As we can see, there are five lines that each correspond to a local maxima that we found earlier.

Kyle Qian

Analysis

Changing the amount of quantization for θ and ρ produce different results. A greater quantization of either will give more accurate lines, as they will give a greater precision for each of our xy space to parameter space calculations; However, in doing so we also greatly increase the computation run time.