Part A: Line and Edge Detection

Results, observations, explanations, and discussion

1. Compute the edges in this image:

Sobel is used here.

2(a). Hough transform using slope-intercept representation

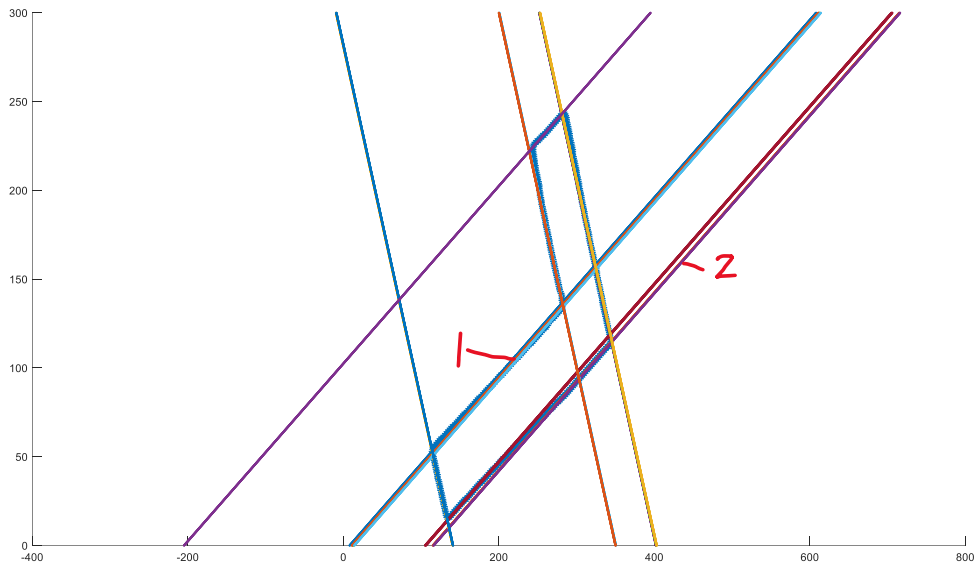

Figure 1: Lines detected by Hough transform using slope intercept representation.

Discussion: For this question, I reused the code from prof's tutorial and adjusted the resolution of a and b. Since a is the slope of the line, I determined the range of slope in the image by manually calculating the gradient value between two points, each from one end of a line. I did the same thing for b (intercept) where I eye-powered the approximate minimum and maximum intercept of the lines.

Observation 1: I had to increase the N value to 17 (which means showing the top 17 lines with the most votes in my_accumulator) to be able to find all 6 lines in the image. Many lines (around 4 lines each) are detected at line 1 and 2. Take line 1 as an example: all the points seem to form one straight line when the image is zoomed out. Upon closer observation, however, we can see that the points are scattered around (refer to the figure below) and

those with similar slope and gradient will form an intersection in the a-b plane, resulting in a line being detected.

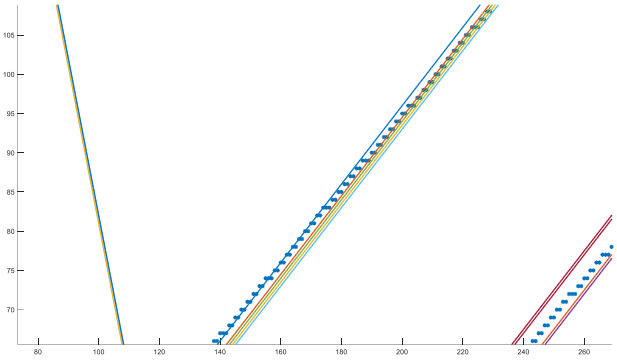

Figure 2: Zoomed in section of edge 1, where "" represents the actual edge and "." represents the line detected by Hough transform.*

Observation 2: Line 4 is only detected when I increased N to 17 (meaning that the line has the 17th most votes among all other lines). This could be due to that the points on that line are very scattered, and it is very short hence there are less points that share the same slope and intercept compared to other lines.

Possible improvement: To solve the issue of detecting multiple lines with similar slope and intercept, we could use quantization which group the lines with similar interception and slope as one line. This could be achieved by using Kmeans. However, due to time constraints, it is not implemented in my solution.

2(b). Normal Intersection

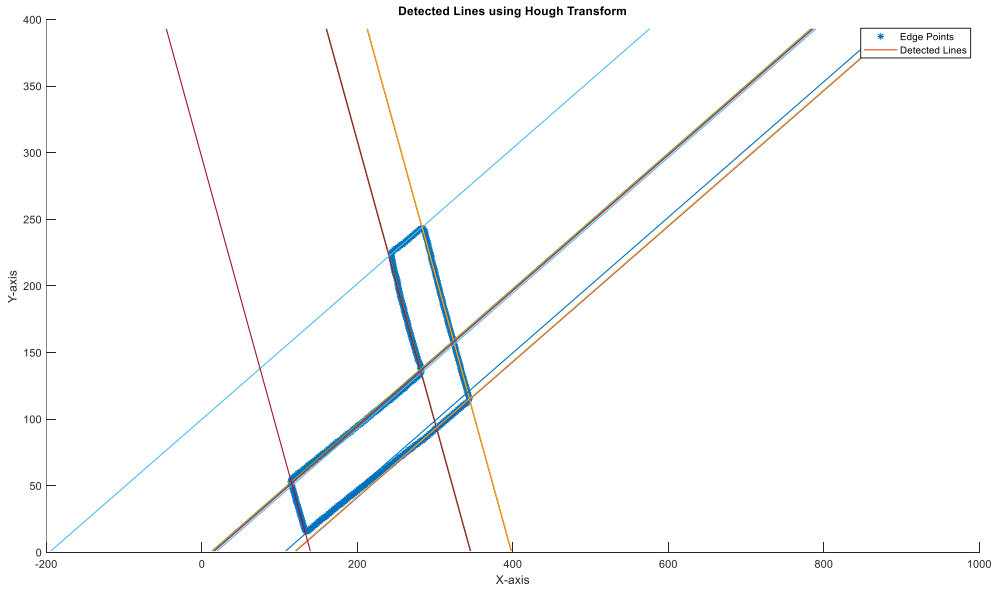

Figure 3: Lines detected by Hough transform using slope intercept representation.

Results: Lowest N count for all lines to be detected is 13, which is slightly lower than that of slope-intercept representation.

3(a)

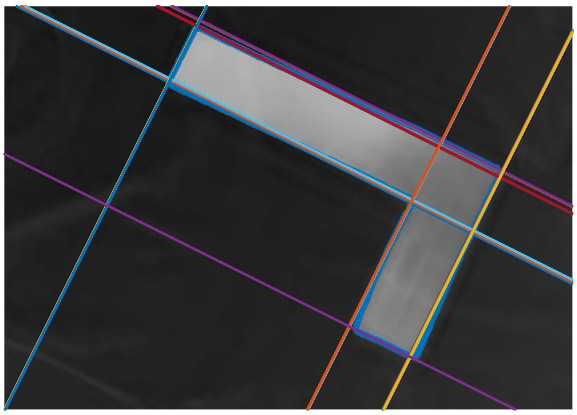

Figure 4: Lines detected by Hough transform on the original image.

(b)

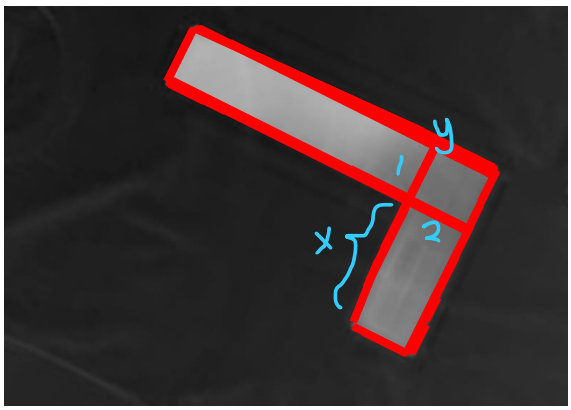

Figure 5: Lines detected by Hough transform on the original image, lines are trimmed to only show on the boundary of the segment

Algorithm: For every intersection, a unique key is created in the format of "a_(a value)_b_(b value)". The points (x-y plane) that form intersection in a-b plane will be hashed to the corresponding key as a 2D array. To plot the line on the boundary of image, iterate through the ab values with top votes in accumulator and plot the line using the points stored in the keys.

Room for improvement: Undesired lines are found at 1 and 2. It is suspected that there is a point at y that has the same a and b value as the points in x, which misled the algorithm to classify point y as a part of line x. Same explanation applies to 2. Due to time constraints, I am unable to solve this issue.

Part B. Segment Boundaries

1.

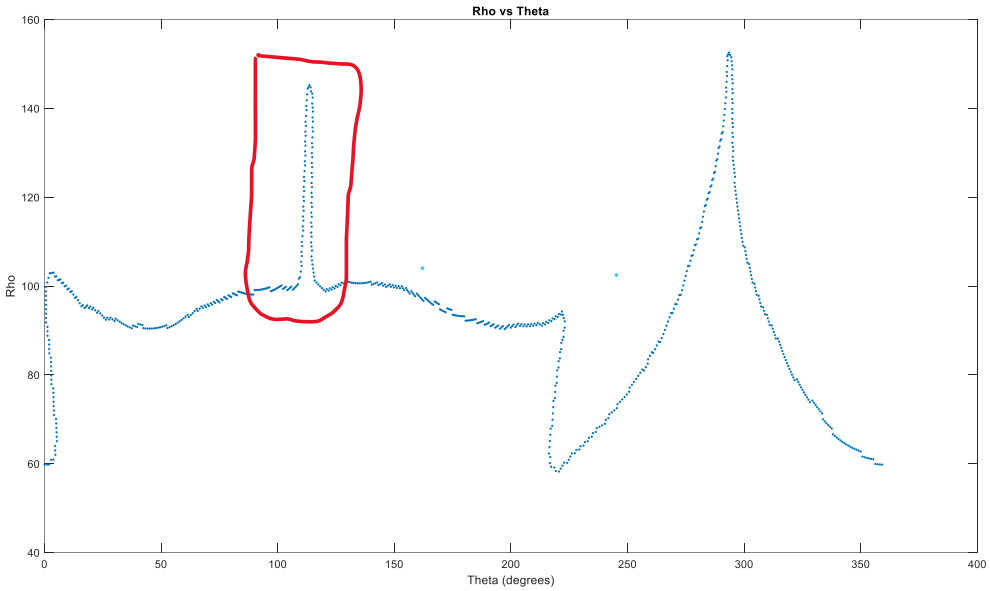

Figure 6: Rho vs theta

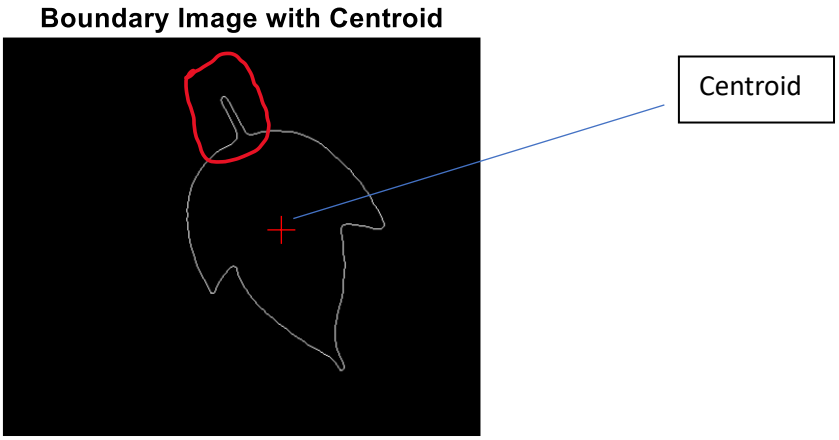

Figure 7: Centroid of K.bmp

The red cross in the figure represents the centroid of the leaf segment. We can observe that the rho-theta plot matches the segment. For example, in the plot, the part in the red box corresponds to the stem of the leaf (in red box also). Rho goes in counterclockwise direction around the centroid.

2. Chain code

```
Ans:
Columns 1 through 29
```

```
     0     0     0     0     0     0     0     0     0     0     7     0
0     0     7     0     0     7     0     0     7     0     7     0     0
7     0     7     0
```

Columns 30 through 58

```
     7     0     7     0     7     0     7     7     0     7     7     7
7     0     7     7     7     7     7     7     7     7     7     7     0
7     7     7     7
```

Columns 59 through 87

```
     7     7     7     7     7     7     7     0     7     7     6     7
7     6     7     7     7     7     7     7     7     7     6     7     7
6     7     6     7
```

Columns 88 through 116

```
     7     7     7     6     7     7     6     7     6     7     6     6
7     6     7     7     7     6     7     7     6     7     6     7     6
7     6     7     6
```

Columns 117 through 145

```
     6     7     6     7     6     6     6     5     5     5     4     4
4     4     4     4     4     4     3     4     4     3     4     4     4
3     4     4     4
```

Columns 146 through 174

```
     4     4     4     3     4     4     4     4     4     4     3     4
4     4     4     4     5     4     4     5     6     5     6     6     6
6     6     6     7
```

Columns 175 through 203

```
     6     6     6     6     6     7     6     6     6     6     6     6
6     6     6     6     6     6     6     6     7     6     6     6     6
6     7     6     6
```

Columns 204 through 232

```
     6     6     6     6     5     6     6     6     6     6     6     6
6     6     5     6     6     6     6     6     6     6     6     6     6
6     6     6     6
```

Columns 233 through 261

```
     5     6     6     6     6     5     6     6     6     5     6     6
5     6     6     6     5     6     6     6     5     6     6     6     6
6     5     6     6
```

Columns 262 through 290

```
     6     6     5     6     6     6     6     6     6     6     6     6
6     6     6     6     6     6     6     7     6     6     6     6     6
7     6     6     7
```

```
  Columns 291 through 319

     6     7     6     6     7     6     7     6     7     6     7     6
6     7     6     6     6     5     5     4     3     3     3     2     3
3     2     3     3

  Columns 320 through 348

     3     3     3     3     3     4     3     4     3     4     3     3
4     3     3     4     3     4     3     4     3     4     3     4     3
4     3     3     3

  Columns 349 through 377

     3     4     3     3     4     3     4     3     3     4     3     3
4     3     3     3     3     3     3     4     3     3     4     3     3
4     3     3     3

  Columns 378 through 406

     3     3     3     3     4     3     3     3     4     3     3     2
3     3     4     3     3     3     3     3     3     2     3     3     3
3     3     2     3

  Columns 407 through 435

     3     3     2     3     2     3     3     3     2     3     2     3
2     3     2     3     2     3     2     3     2     2     2     3     2
2     2     3     4

  Columns 436 through 464

     3     4     5     4     5     5     5     5     6     5     5     5
5     6     5     5     6     5     5     6     5     6     5     6     5
6     6     5     6

  Columns 465 through 493

     5     4     4     3     3     2     3     2     3     2     3     2
3     2     3     2     3     3     2     3     2     3     2     3     2
3     2     3     2

  Columns 494 through 522

     3     2     2     3     2     2     3     2     2     2     3     2
2     3     2     2     3     2     2     3     2     2     2     3     2
2     2     2     3

  Columns 523 through 551

     2     2     2     2     2     2     2     3     2     2     2     2
2     2     2     2     3     2     2     2     1     2     2     2     2
3     2     2     2

  Columns 552 through 580

     1     2     1     2     2     2     2     2     2     1     2     2
2     2     2     1     2     2     2     1     2     2     2     1     2
2     1     2     1
```

```
  Columns 581 through 609

     2     1     2     1     2     1     1     2     1     2     1     1
2     1     1     1     2     1     1     1     1     1     1     1     1
1     1     1     1

  Columns 610 through 638

     1     0     1     1     0     1     0     1     0     1     0     1
0     1     0     1     1     1     1     1     1     2     2     2     2
2     2     3     2

  Columns 639 through 667

     2     3     2     3     2     3     2     3     2     3     2     3
2     3     2     3     2     3     2     3     2     3     3     2     3
2     3     2     2

  Columns 668 through 696

     3     2     1     2     1     0     0     7     7     7     6     7
7     6     7     6     7     6     7     6     7     6     7     7     6
7     6     7     6

  Columns 697 through 725

     7     6     7     6     7     6     7     7     6     7     6     7
6     7     7     0     7     0     0     0     0     1     0     0     0
1     0     0     1

  Columns 726 through 747

     0     0     0     0     0     0     0     0     1     0     0     0
0     0     0     0     0     0     0     0     7     0
```

Part C: Segmentation Features

1.

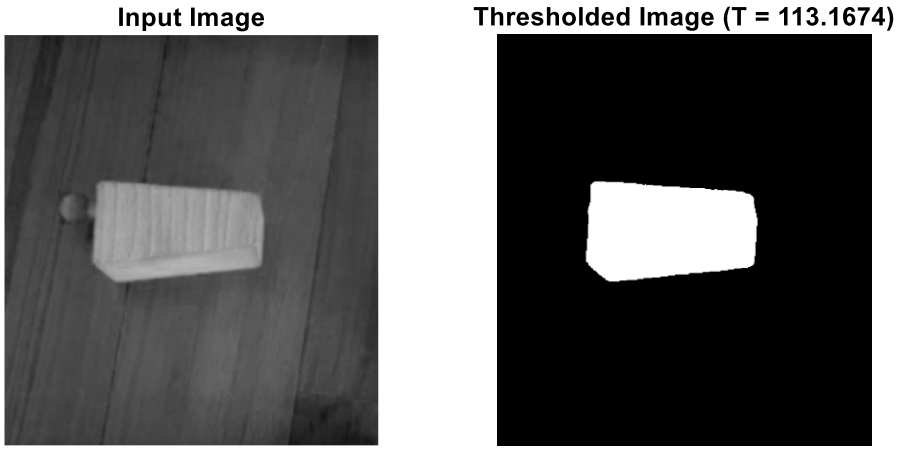

Figure 8: Input image and thresholded image (using intermeans)

From the result, we can observe that intermeans method works very well for this image as it is able to segment out the main object accurately. There are several features in this image which are important for intermeans method to work well:

a. When there's a significant contrast or difference in brightness or intensity between the main object and the background. The method calculates a threshold that separates these regions accurately, resulting in a clean segmentation of the main object.
b. When the original image does not contain significant noise or artifacts, the thresholding process becomes more reliable. Noise in the image can interfere with the determination of an accurate threshold level, potentially affecting the segmentation quality.

2. Features
   Perimeter: 567.0810

   Area: 20016

   Compactness: 1.2785

   Centroid: (200.3693, 230.1944)