UNIVERSITY OF WISCONSIN-MADISON, COMPUTER SCIENCE DEPARTMENT

CS760 - Assignment 4

Zhicheng Gu
Email: zgu58@wisc.edu
Student ID: 9073696370

October 2, 2016

1 PROBLEM 1

1

I use Otsu's method for threshold calculation and compute the the foreground region. The result is shown as following figure.

Figure 1.1: Problem 1, part 1

The statics of figure is print in the console, which is like:

Image size: 104 x 100

Total number of pixels: 10400

Number of pixels in region: 4655

Length of perimeter: 690

Diameter: 1.103449e+02

2

the distance of every pixel in the image to its closest point on the perimeter is shown in the following figure. The left figure is the distances for pixels outside the interior of the region. The second figure is pixels which lie inside the interior of the segmented region.

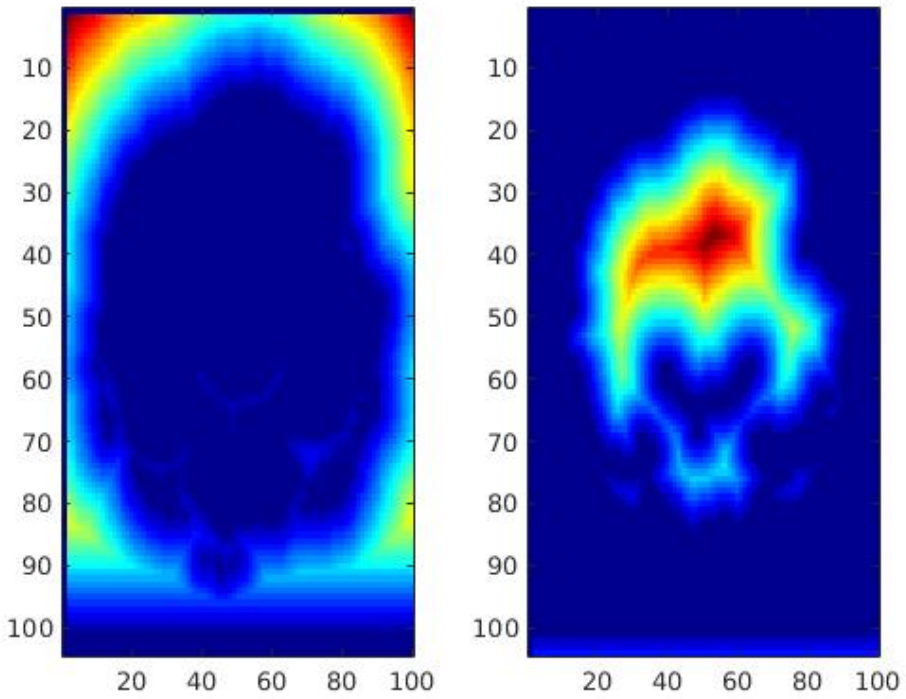

Figure 1.2: Problem 1, part 2

3

The correlation between predictor and area is 0.4643. The correlation between predictor and diameters is 0.4827. From the figure and correlation, we can see that they have positive correlation, but is not too strong.

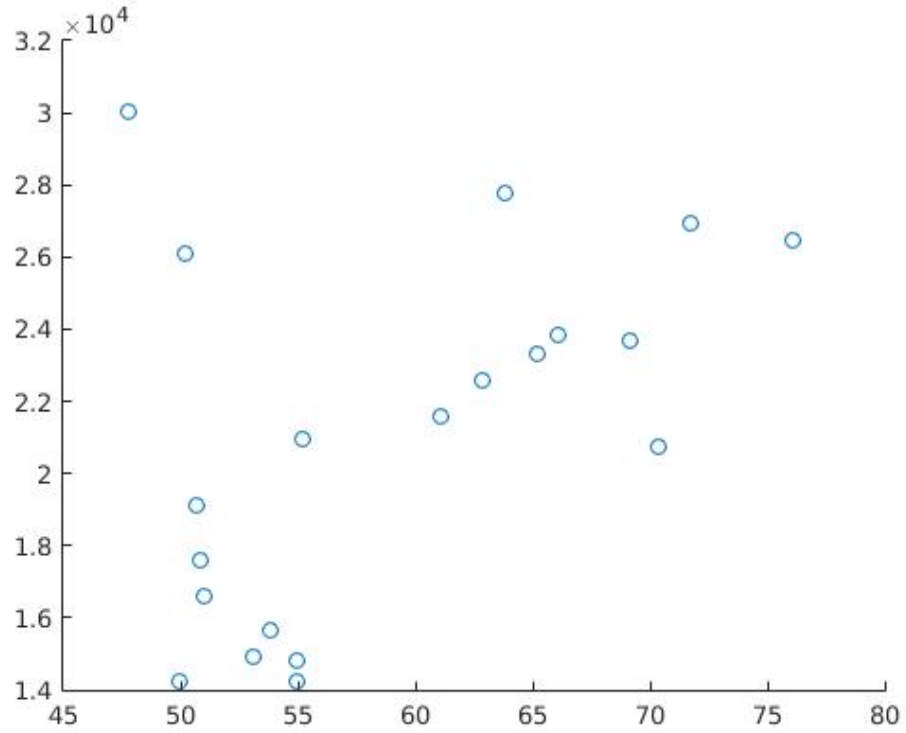

Figure 1.3: Relation between predictor and area

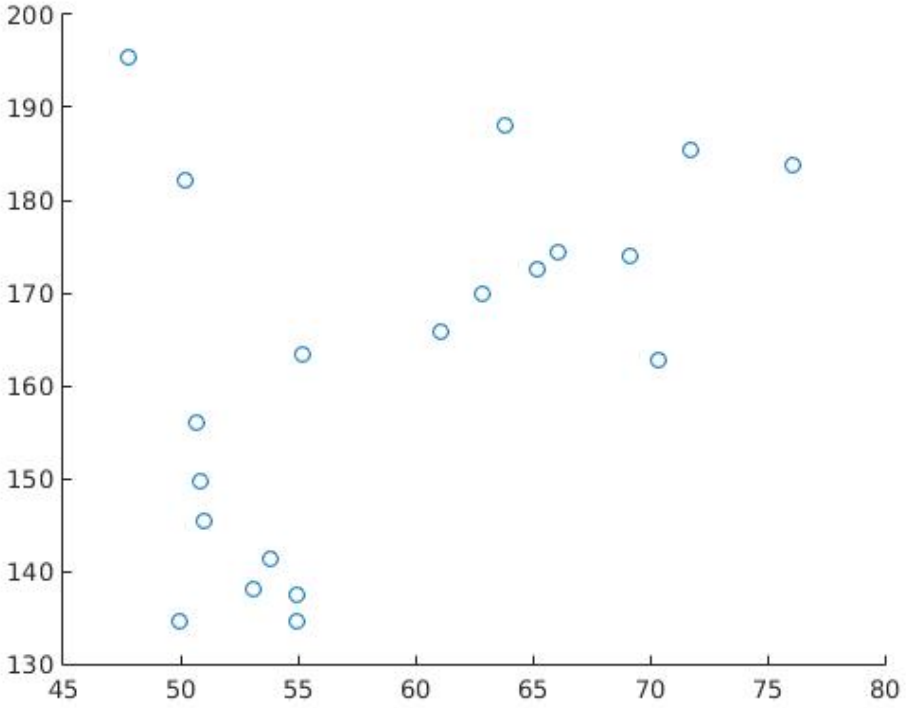

Figure 1.4: Relation between predictor and diameters

2 PROBLEM 2

The idea to find region is as the following. First go through all of the points, once I find a point with value 1, I call a helper function to label all of it's neighbors with a certain value.

The result is shown in following figure.

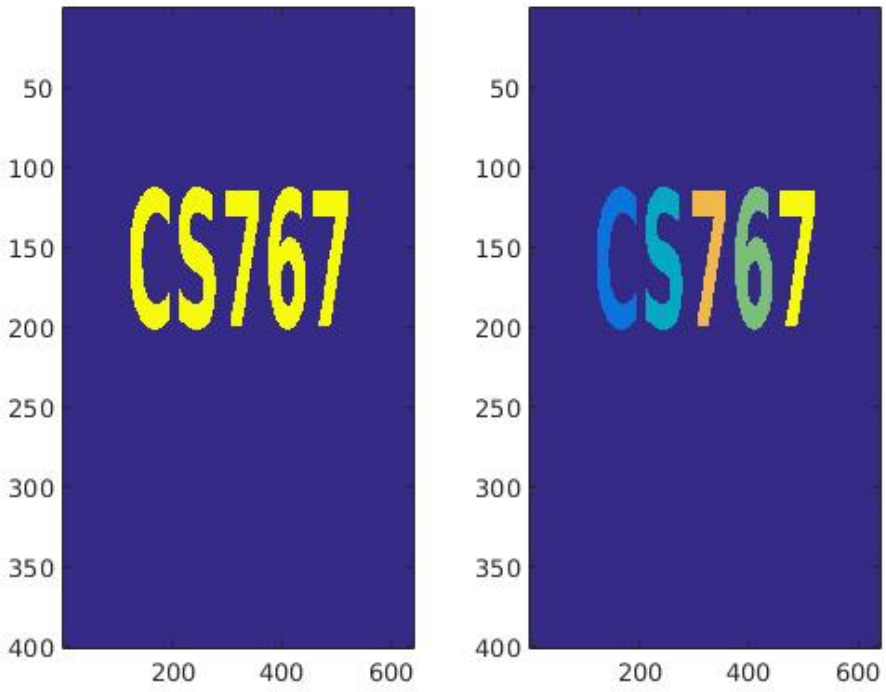

Figure 2.1: Problem 2

3 PROBLEM 3

1

The result is shown in following figure.

2

imview() is unavailable in my matlab, I use ginput() instead.

After I select a point on the figure, i go through all it's neighbor in 5*5 range. It's any of it's neighbor is edge, then recursive do this step.

Below are some success case and some failure cases. The success cases all have clear edges while the failure cases all have broken edges. Hence, I believe the key to good boundary tracing is you get a good edges.

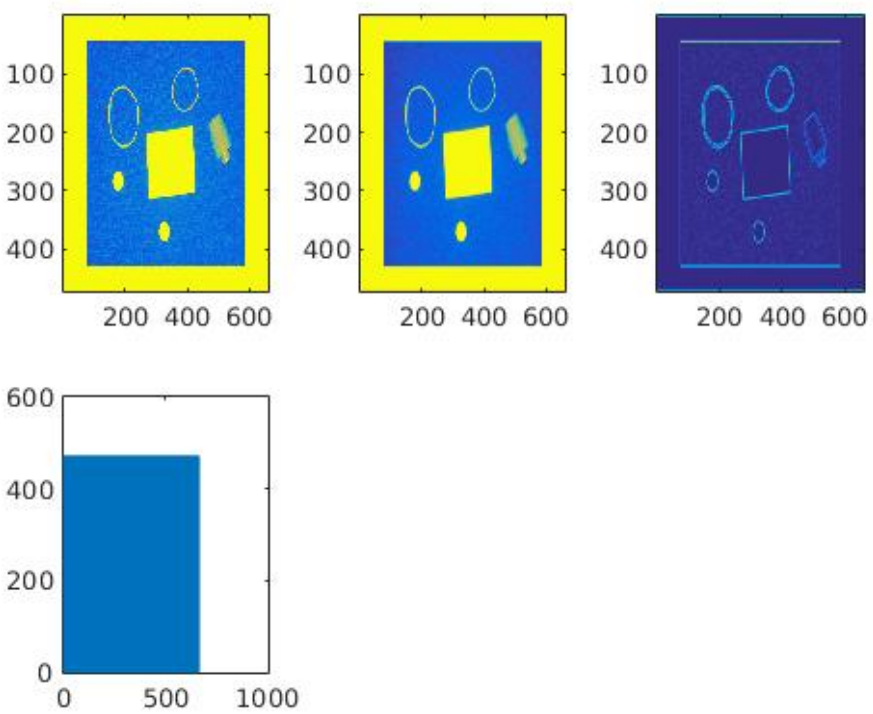

Figure 3.1: Problem 3, part 1

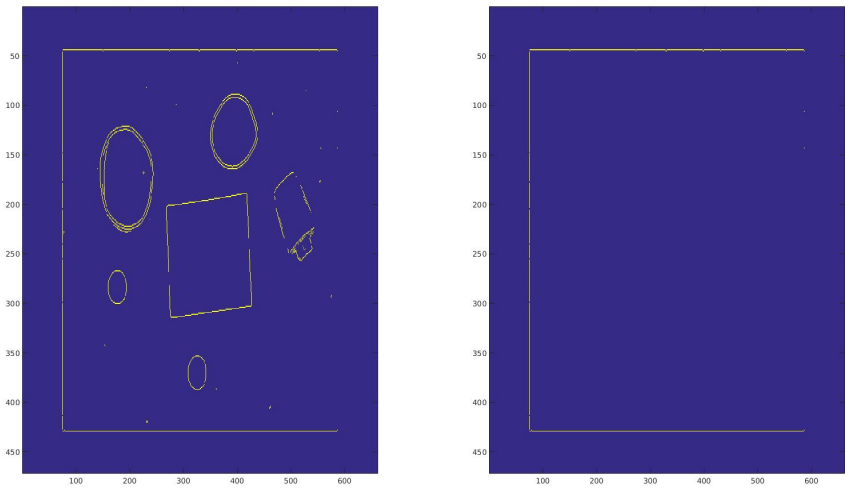

Figure 3.2: Success example

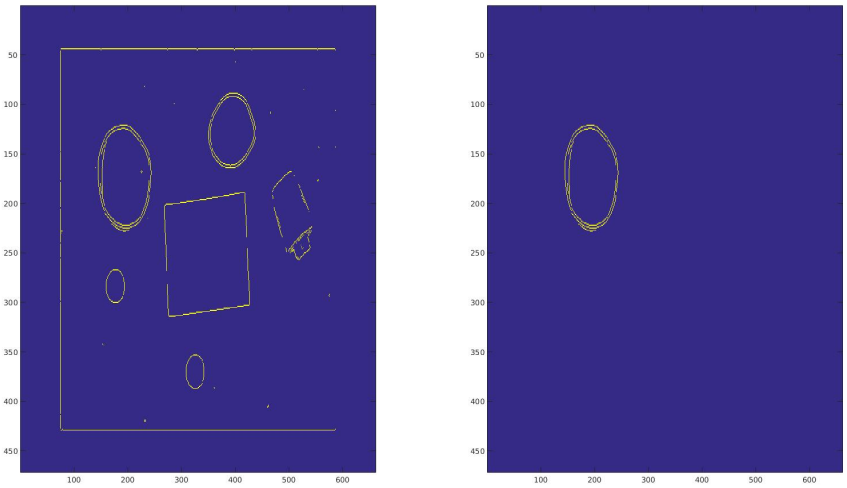

Figure 3.3: Success example

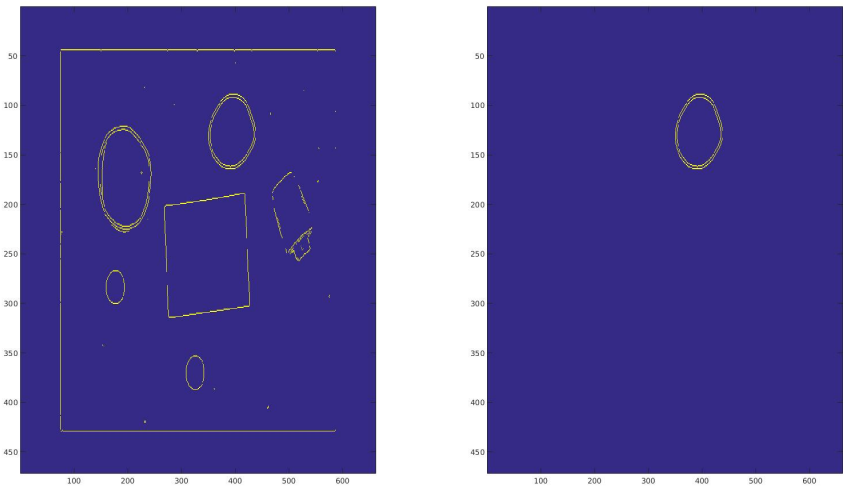

Figure 3.4: Success example

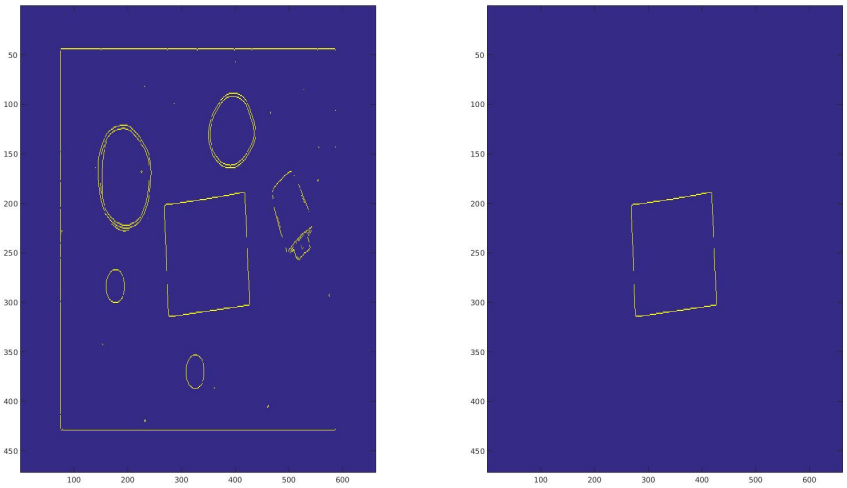

Figure 3.5: Success example

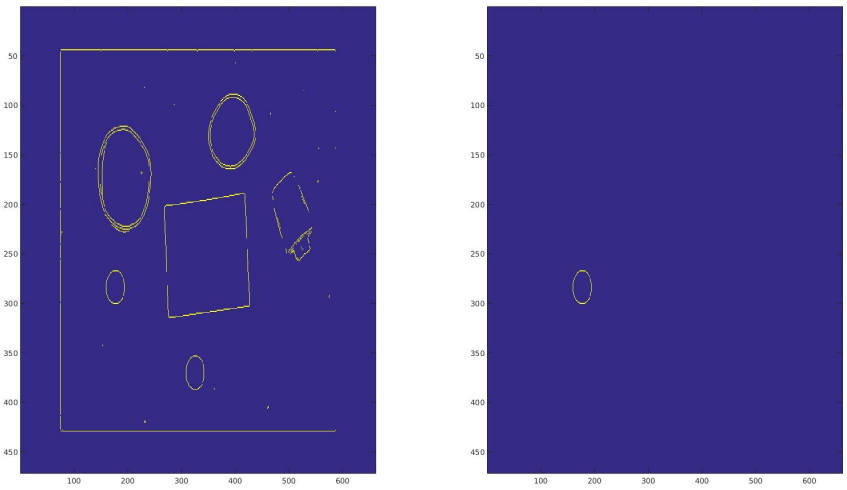

Figure 3.6: Success example

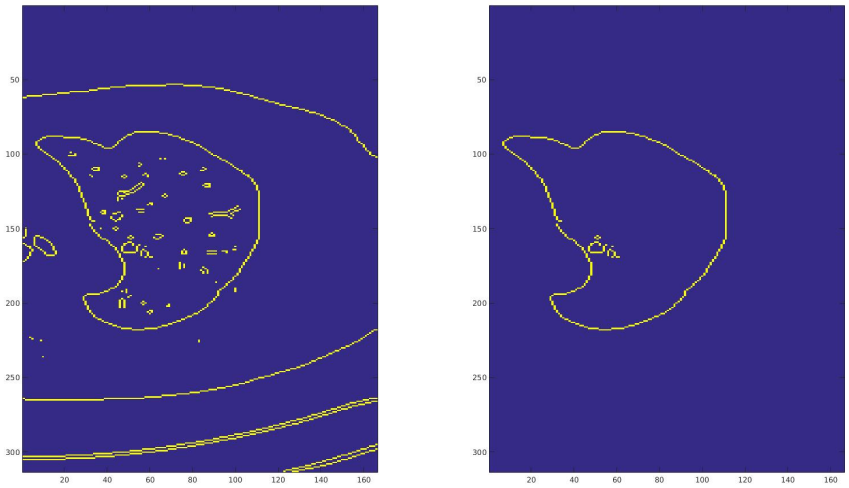

Figure 3.7: Success example

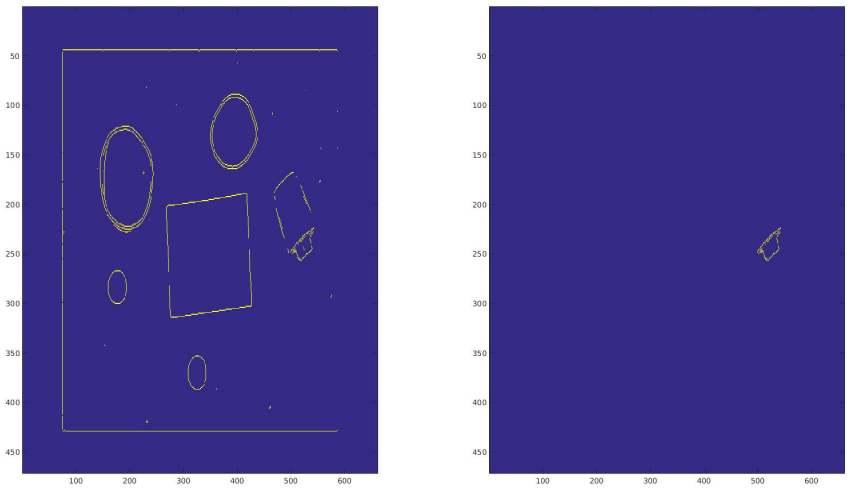

Figure 3.8: Not very successful

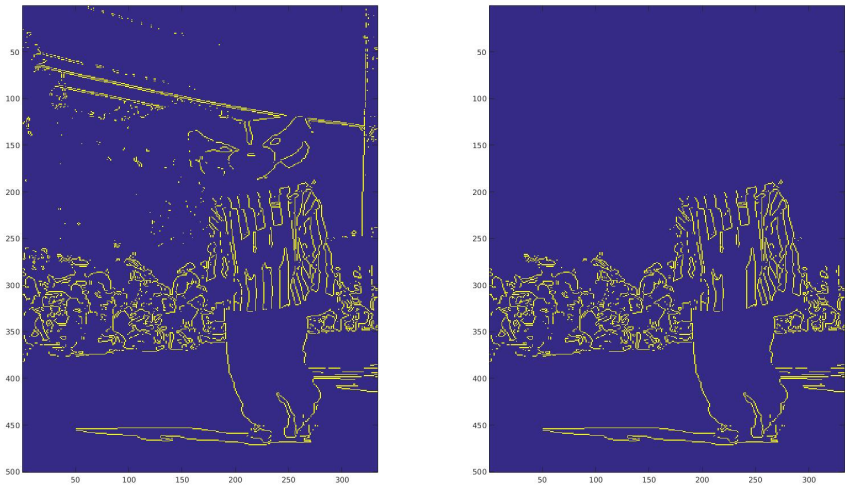

Figure 3.9: Not very successful

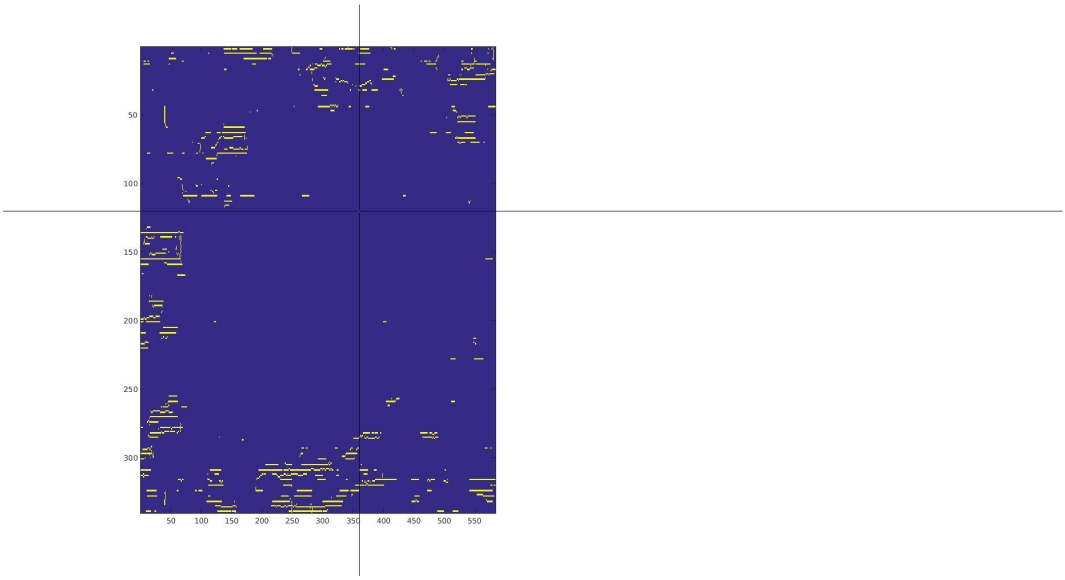

Figure 3.10: Totally fail example

4 PROBLEM 4

1

I am using dynamic programming method in this paper to speed up the computation of mask feature. Specifically, when we need to compute the sum of area D in the following figure, we have

$$\begin{aligned} D &= (A+B+C+D) - (A+B) - (A+C) + A \\ &= sum(4) - sum(2) - sum(3) + sum(1) \end{aligned} \tag{4.1}$$

where sum(n) means the total sum from top left corner to point 4. Hence, we only need to compute every sum(n), and then we can get any area D by linear time.

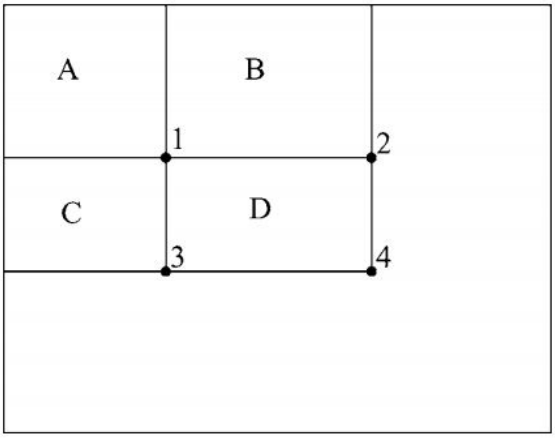

Figure 4.1: Four array references

The time complexity will be O(length * width). If we compute the mask sum of each points by a for loop, the time complexity will be O(length * width * m * n), where m and n are length and width of mask. I speed up my function getFeatureHist from a few minutes to less than one second by using this dynamic programming method.

2

The result is shown in following figure. We can see that after mask detection, the place where left side and right side have very different density will get higher value. The brighter places are typically the edges or features.

3

The histograms for different mask sizes are shown in the following figures. The mask size (m and n) are 5, 10, 15 ,20, 40. From the figures we can see that the small mask (5 * 5) gives out too much noise, while the large mask (40 * 40) gives useless result.

The shape of histograms of small mask result is shape, which means the more points are near 0. The shape of histograms of small mask result is more evenly distributed, but the feature figure is not very good.

In general, I think we need to choose reasonable mask size (10 to 20) to make results readable and feature distributed more evenly.

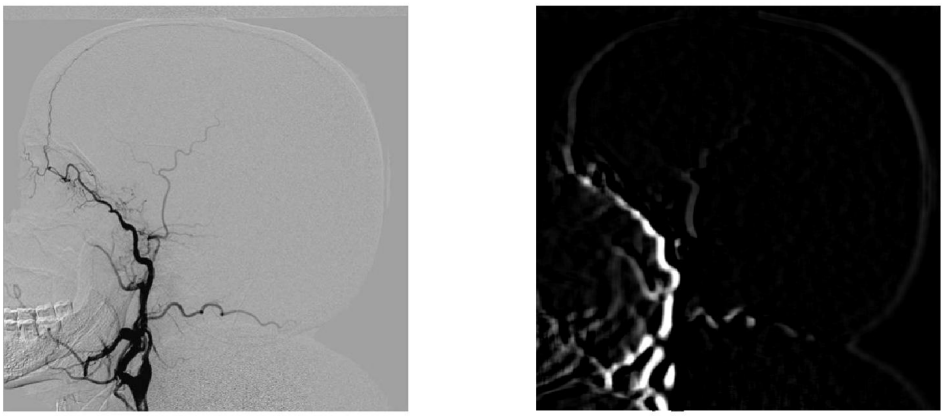

Figure 4.2: Result get by mask 10 * 10

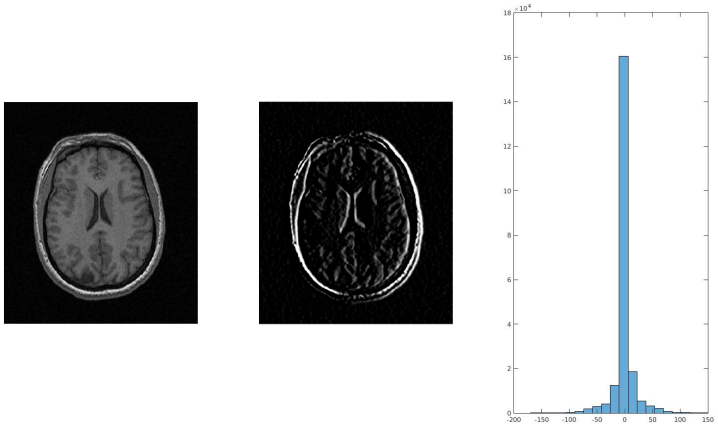

Figure 4.3: mask 5 * 5

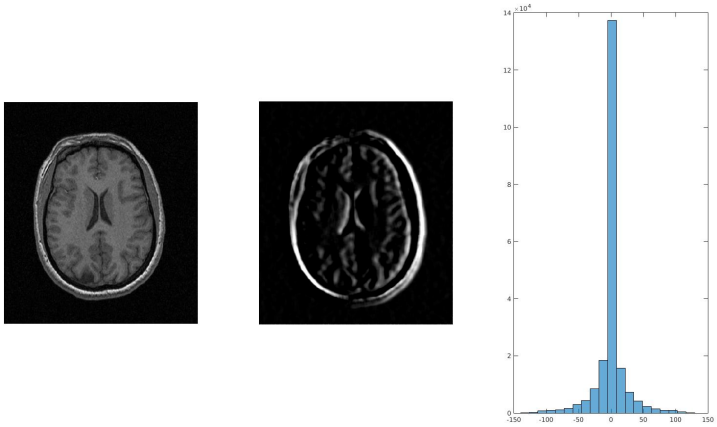

Figure 4.4: mask 10 * 10

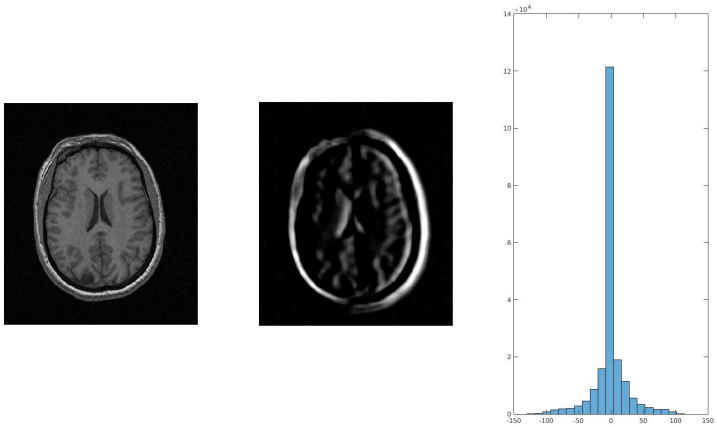

Figure 4.5: mask 15 * 15

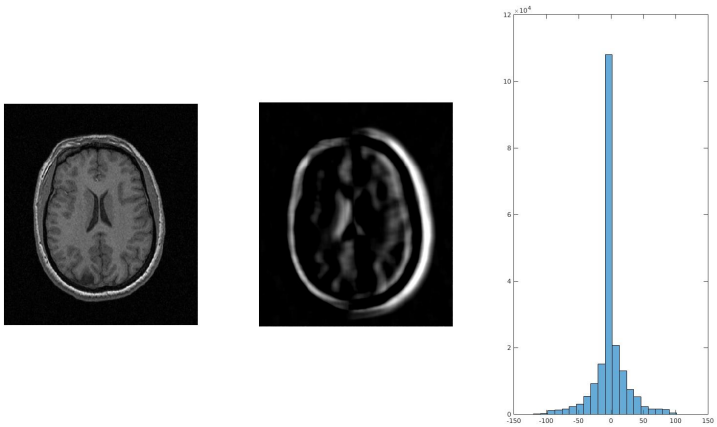

Figure 4.6: mask 20 * 20

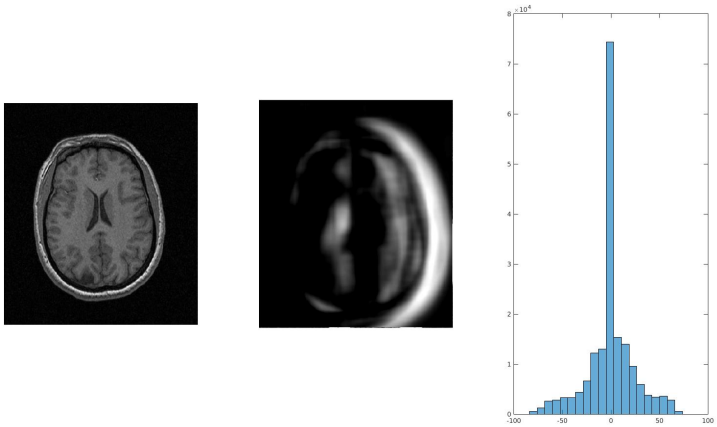

Figure 4.7: mask 40 * 40