Aayushi Pandey, SNR: 2038981

Image Analysis: Take Home Assignment Report

Question 1

Introduction:

In this question, the purpose is to segment the image to identify the coins. The image is very noisy, as the background consists of a prominent texture and the image has a varying range of illumination. There is also a shadow, which makes segmentation even more difficult. Some of the coins touch each other and segmenting them separately is also a challenge.

For segmentation purposes, I pre-processed the image (by smoothing with median filtering/gaussian kernel, opening/closing). To correct for illumination effects, I used top hat filtering. Then I used adaptive thresholding and Otsu thresholding for segmenting and obtaining a binary image. In case of adaptive thresholding, I used only the blue channel of the RGB image, to obtain better results. After applying the thresholding algorithms, I applied some morphological operations to fix the defects (for instance background classified as foreground), to get a good representation of the coins.

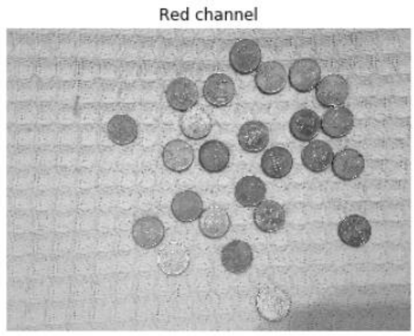
I identified the connected components of the image, and since there were some overlapping coins, I used morphological watersheds to separate them. In the end, I constructed the bounding boxes for the coins, and was successful in detecting all the coins.

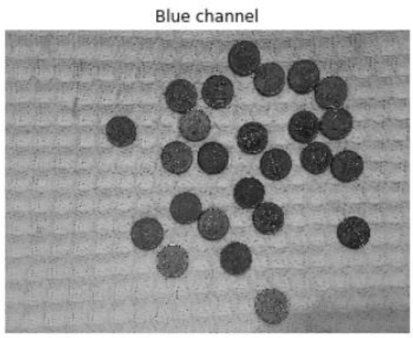
Description of solution: Please run the notebook: Assignment solution_question 1

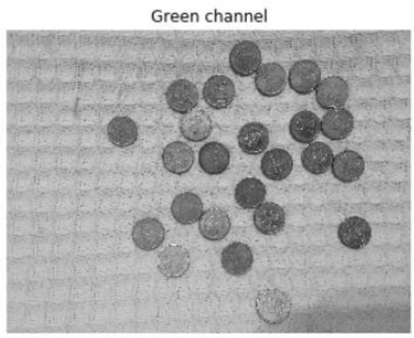
Part 1: I read the image of coins and displayed it. It looks like this:

An image of coins

Part 2: After that I displayed the red, blue and green channels of the image in a subplot. The output looks like this:



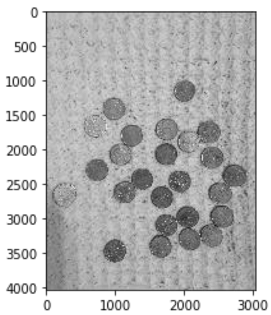




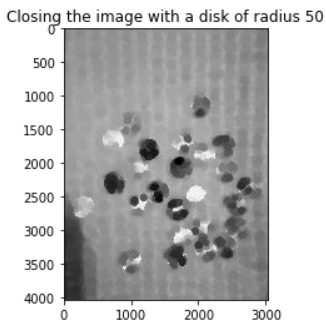
Part 3: After that I use **two approaches to segmentation**:

Method 1: Otsu Thresholding (global) with morphological operations:
Flowchart

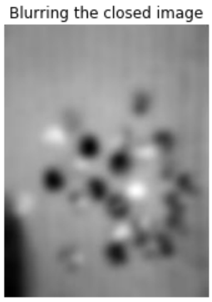
1. Converting the RGB image to grayscale, and converting the datatype to float



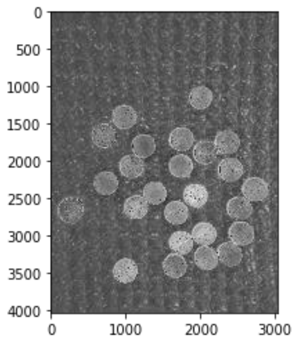
2. A variant of Black Top Hat Filtering: To remove the shadow from image
   a. Closing the image with a disk of radius 50



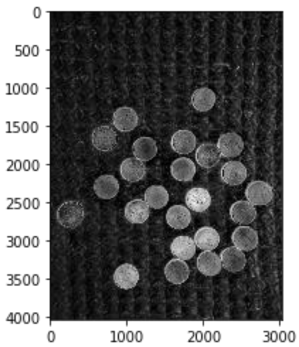
   b. Smoothing the result from a with a gaussian filter (sigma = 70)



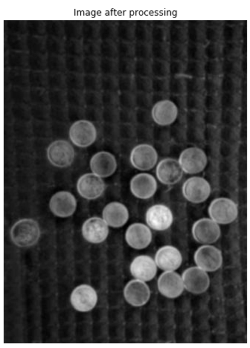
c. Subtracting the original image from the result of b.



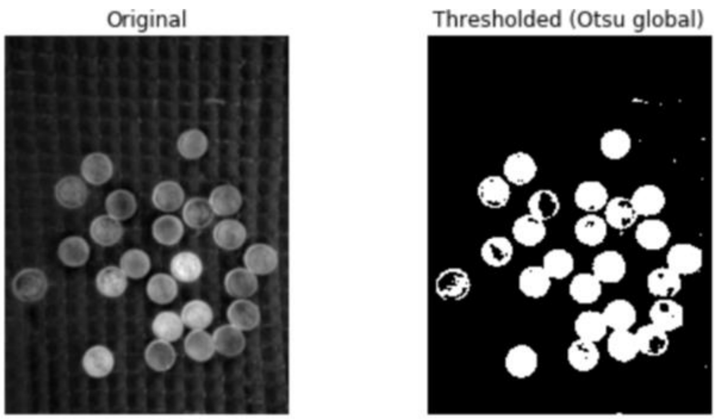
3. Median Filtering of the resultant image from 2, to remove noise (disk of radius 5)



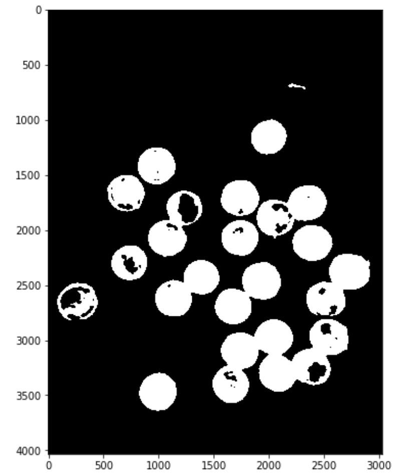
4. Since the background was still noisy, I applied smoothing again (gaussian filter, sigma = 10)



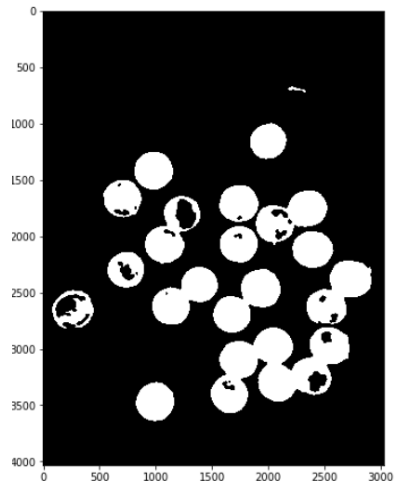
5. After applying global thresholding using Otsu method for computing the threshold, I got the following result:



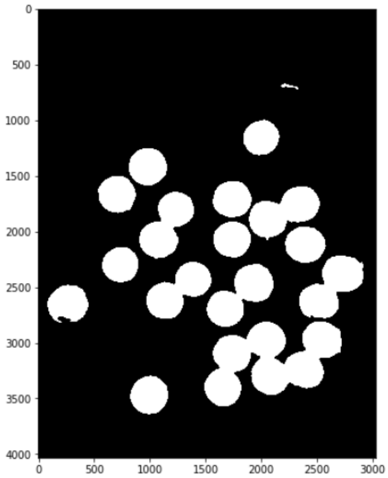
6. Since, the result was not satisfactory, I decided to do further processing using morphological operations, on the image obtained from 6. I applied remove small objects to remove white speckle like structures in the top right corner. The result looks like:



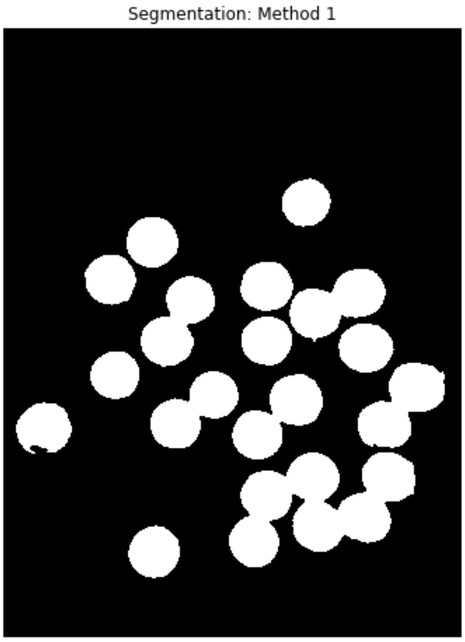
7. Then I applied closing operation, as an attempt to fill minute holes



8. Then, I realized that I can simply fill the holes by using binary fill holes. The result was:



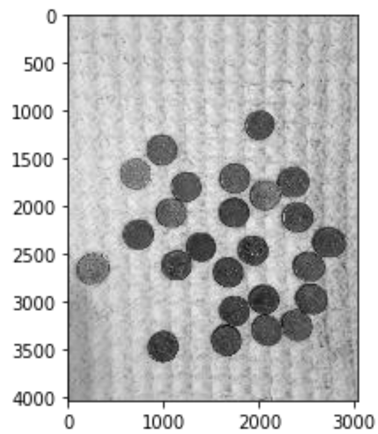
9. Then I removed the small objects again. The final result of segmentation method 1 looks like this:



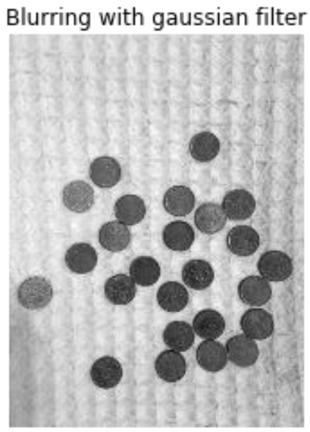
Method 2: Adaptive Thresholding with morphological operations

Flowchart

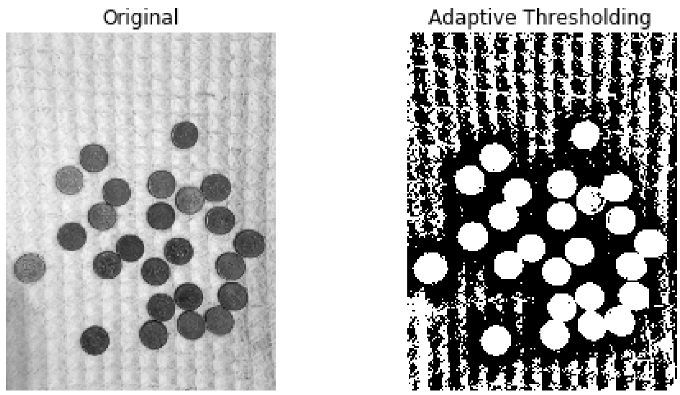
1. Applied median filtering (disk of radius 5) to the blue channel of the image, to remove particle like noise.



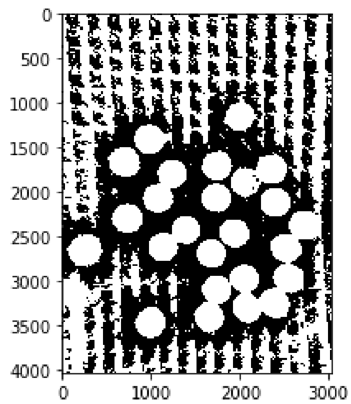
2. To smoothen the image, I applied gaussian filtering (sigma = 3)



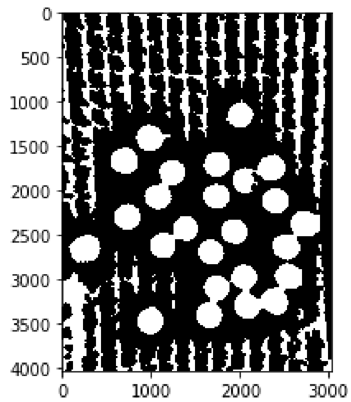
3. I applied adaptive thresholding, to the image obtained from 2. The results are as follows:



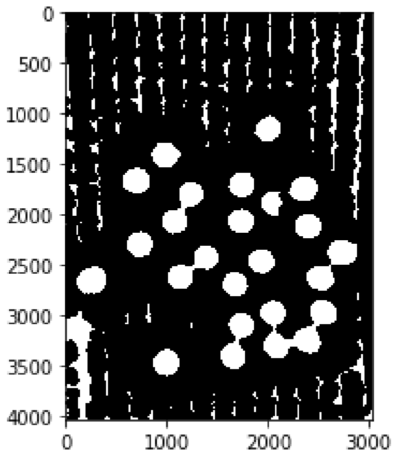
4. I process the image further using morphological operations. I apply binary closing (disk of radius 10) to the image, to fill the minute holes in the coins.



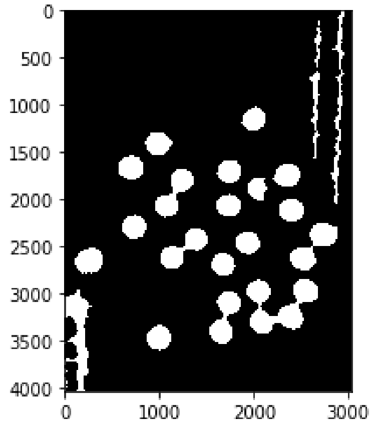
5. I apply binary erosion with a disk of radius of 20 to the image from 4, to remove the white lines.



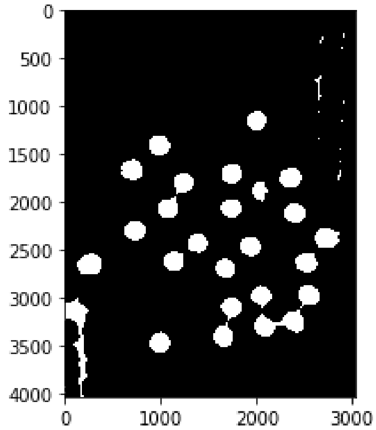
6. I apply binary erosion with a disk of radius 20 again to image from 5, because the white lines still persist.



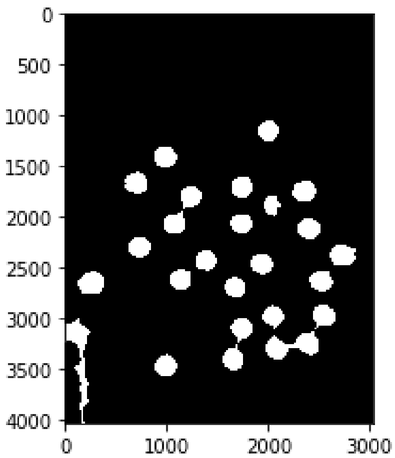
7. I apply remove small objects to remove the distorted line like elements, to image from 6.



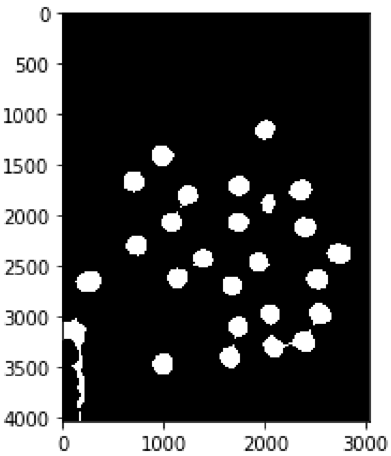
8. I erode the image from 7 further with a disk of radius 20.



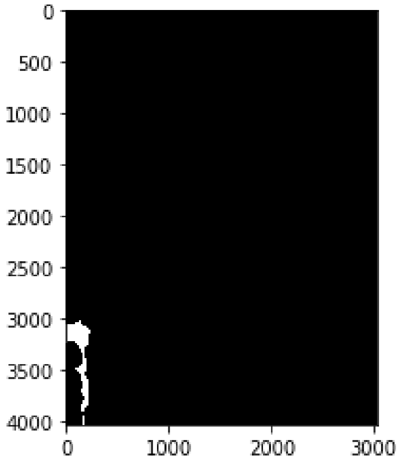
9. I apply remove small objects to remove the distorted line like elements to the image from 8.

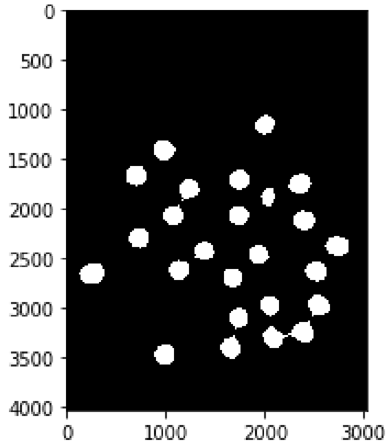


10. I apply binary erosion with a disk of radius 5, to the image from 9, in hope of destructing the shadow that shows as white after thresholding.

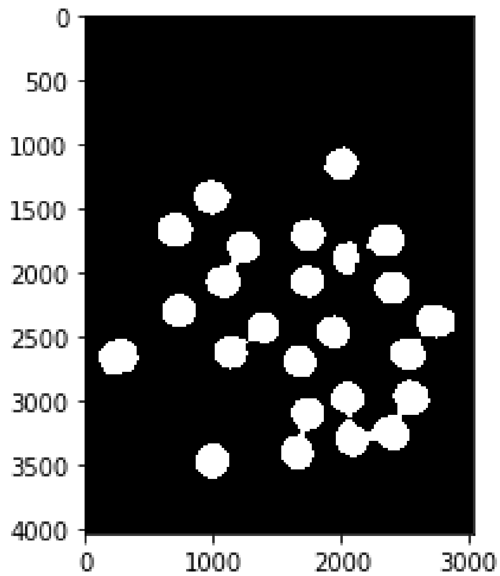


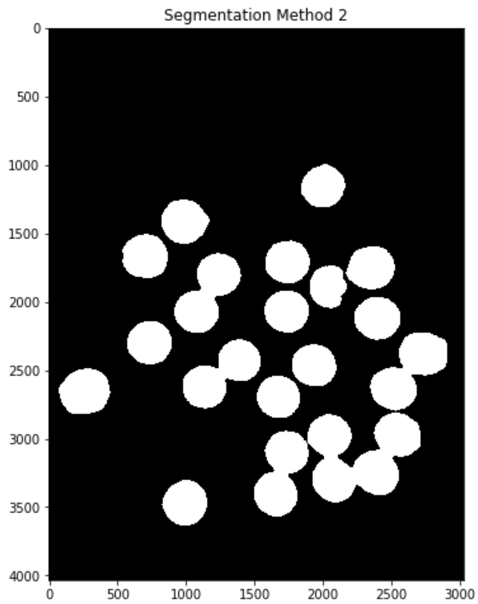
11. I use remove small objects to the image, but I remove the coins. Then I subtract this image from image in 10, to obtain just the coins.



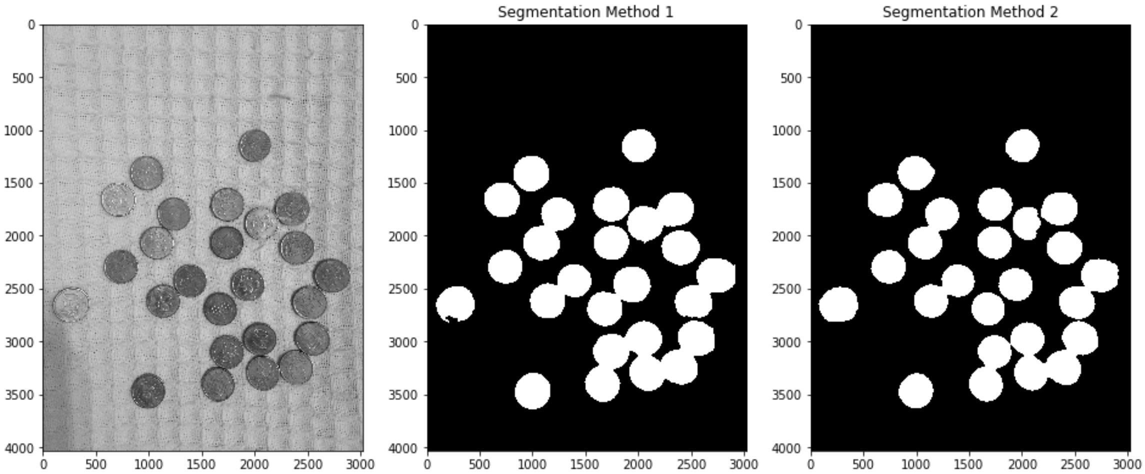


12. Then I dilate the image twice with a disk of radius 30, so that the coins retain their original shape





Comparison:



Now comparing the results from method 1 and method 2. I have been successful in not classifying the background elements as foreground.

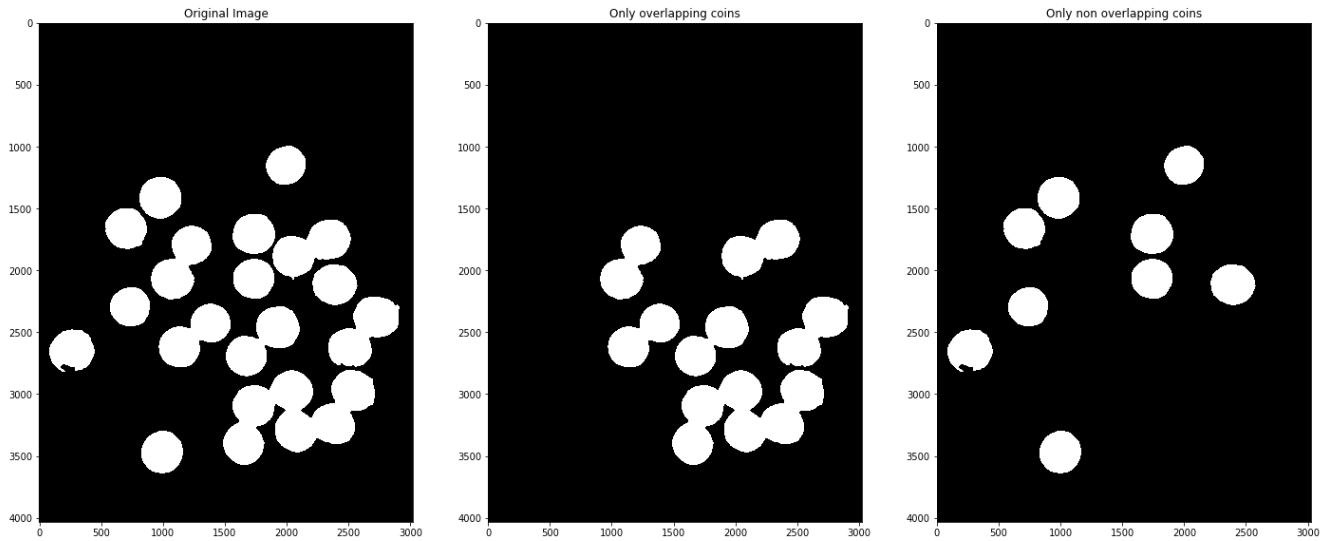
In segmentation method 1, the shape of the coins is distorted slightly, but the connectivity remains intact.

In segmentation method 2 also, the shape of coins is distorted slightly, but the connectivity is affected. Due to too much erosion, one coin is separated from the coin it was touching.

Hence, for the subsequent parts I will be using the binary image obtained from segmentation method 1.

Part 4: Separating overlapping and non-overlapping coins:

Using remove small objects, because single coins are smaller objects as compared to joint coins.



Part 5: Obtaining connected components:

I use the label function, to obtain the connected components of an image. Then I overlay these labels on the original RGB image, to separate the connected components from each other. The result looks like this:

Connected components of the image

Part 6:

I tried to find the boundary of the image by simply eroding and subtracting the eroded image from original image. But this approach wasn't useful, because some coins overlap with each other, and in that case, they don't get a proper boundary.

Original image

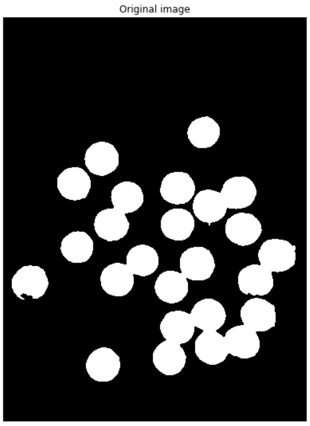
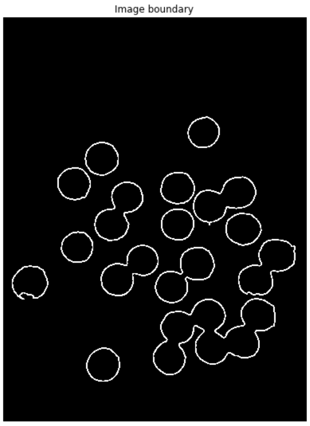
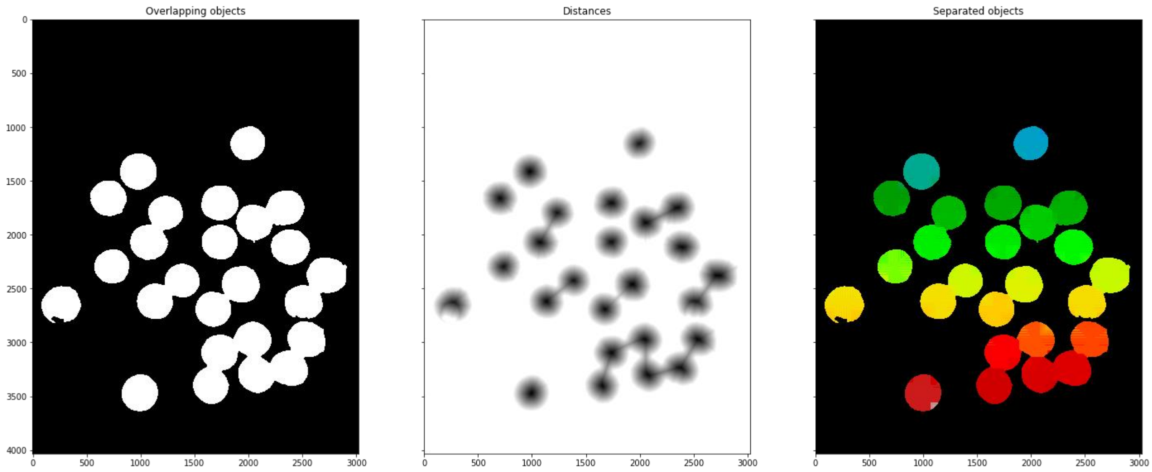


Image boundary

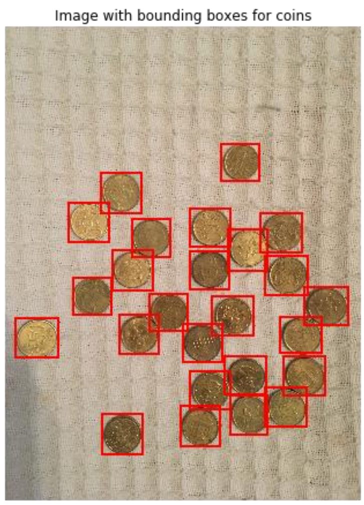


The connected coins don't get a boundary of their own

So, I used morphological watershed instead, and with region props I found the bounding boxes.



The resultant image with bounding boxes is:



I am able to detect all the coins. There are 25 coins in the image.

<u>Conclusion:</u>

I am successful in detecting all the coins. I used various standard image analysis techniques. However, the segmented image has a slightly distorted shape. More techniques can be used to approach this problem, for instance say, template matching/super pixels or even machine learning, and even a much better result can be obtained.

<u>References:</u>

1. Worksheets supplied during the course
2. https://scikit-image.org/docs/dev/auto_examples/segmentation/plot_label.html

<u>Special Instructions:</u>

Please note that the code for this question takes about 15 minutes to run, due to the part where I close the image with a very large structuring element for black top hat filtering. Please be patient.

Question 2:

Introduction:

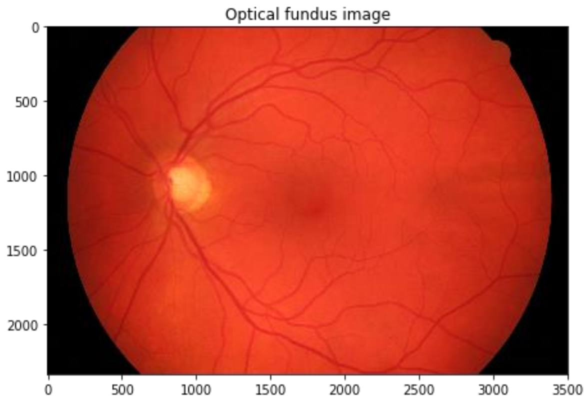
This question concerns segmenting the image to obtain the vessels. A manually annotated image is provided as ground truth. The image has different levels of illumination, and this makes segmentation challenging. Obtaining the detail becomes very difficult, because some vessels are very fine, and they are not very distinguished from the background. I apply various pre-processing techniques (median filtering, gaussian filtering, opening/closing by reconstruction etc.) for noise removal and top hat filtering to treat for illumination differences. I use global thresholding and adaptive thresholding for segmentation, and use Jaccard score as a metric to compare them to the ground truth. The highest Jaccard score is obtained was 0.718. I try ridge filters as well, but since I was not very successful in removing noise, so I couldn't get any good results.

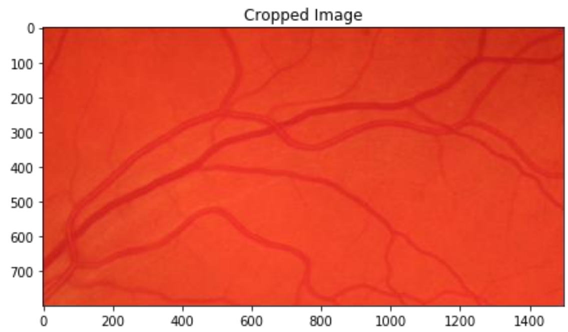
In the second part of the question, I use the manually annotated image and remove vessels less than 8 pixels wide by opening with a square structuring element of width 8. Then I applied thinning, and computed the medial axis transform, and used the distances obtained to display the local width. Then finally I used the probabilistic Hough transform, to detect the lines, because I could obtain line segments, and hence the length of the vascular network. I determined the orientation, by using the end points of the line segments and applying tan inverse operation.

Description of solution: Please run the notebook: Assignment solution_question 2

2.1:

Part 1: The cropped image looks like this:

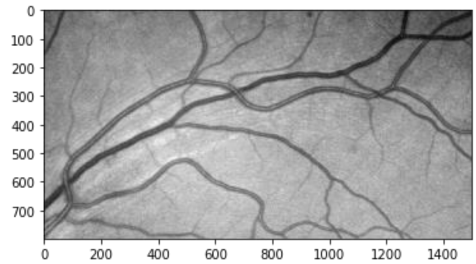




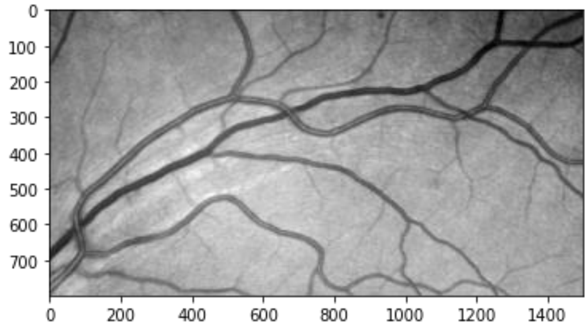
Part 2: Segmenting the vessels in the cropped image

Steps:

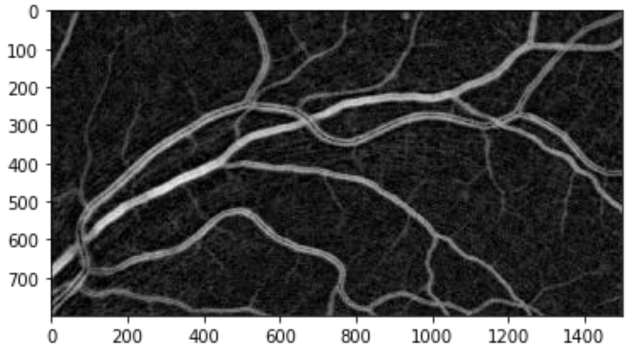
1. Converting the image to grayscale and changing data type to float



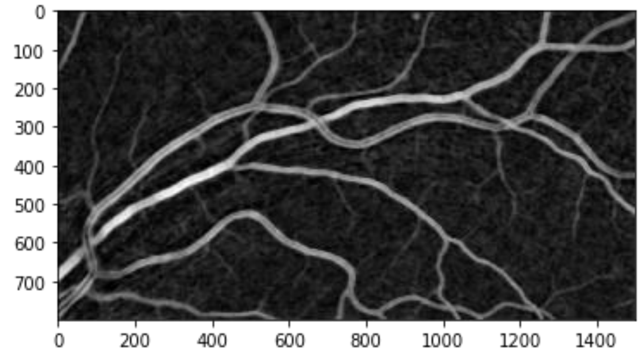
2. Denoising the image using median filtering with disk of radius 3



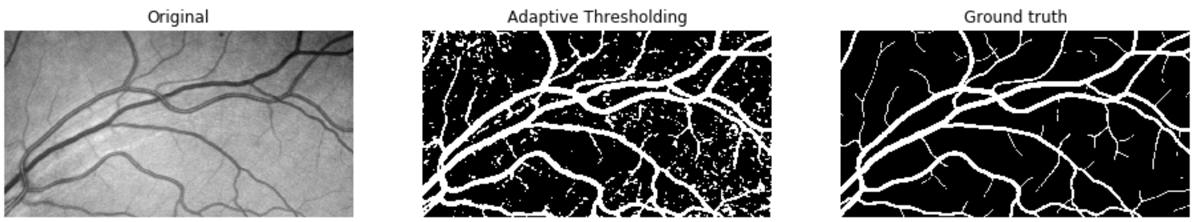
3. To deal with different illumination in different regions of the image, I apply black top hat filtering (closing with a disk of radius 25 minus the original image)



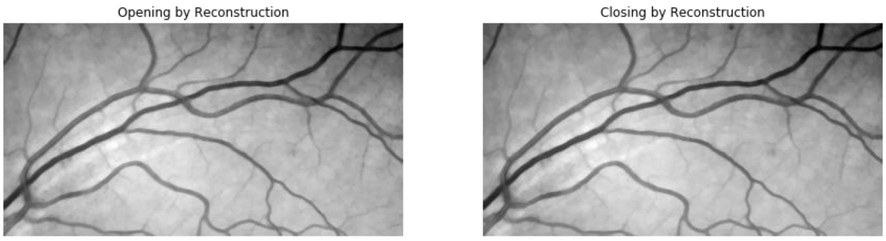
4. I denoise the result from 3, by applying median filtering to it with a disk of radius 5



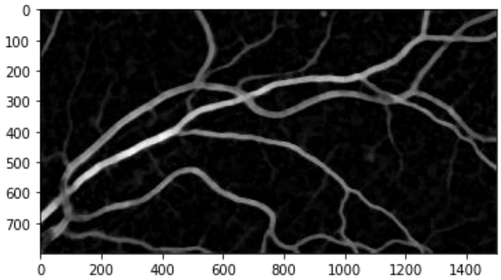
5. I apply Otsu thresholding to the image from 4, and obtain the following result. The Jaccard score is **0.718. This is the highest Jaccard score I obtained** in all the operations I tried.
6. I try some alternative techniques to threshold. I use adaptive thresholding, but the Jaccard score is only about 0.57.



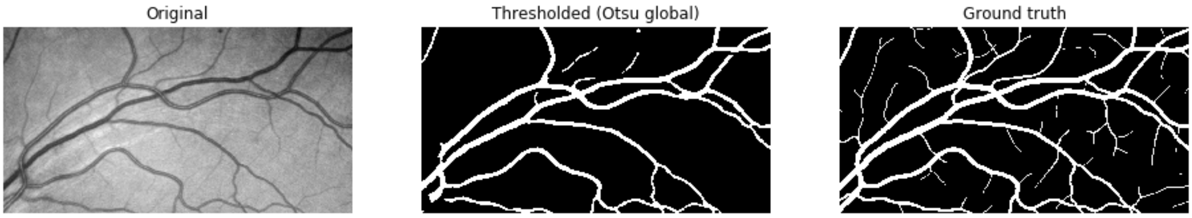
7. After that, I use image from 1, and perform morphological reconstruction on it (to denoise it better)



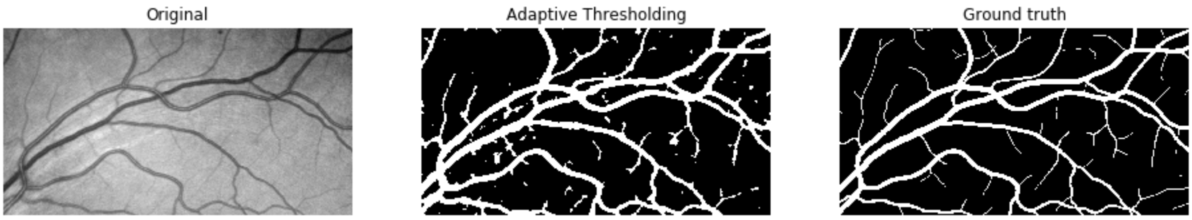
8. Then I apply black top hat filtering to image from 7 (using disk of radius 25), to correct for illumination effects.



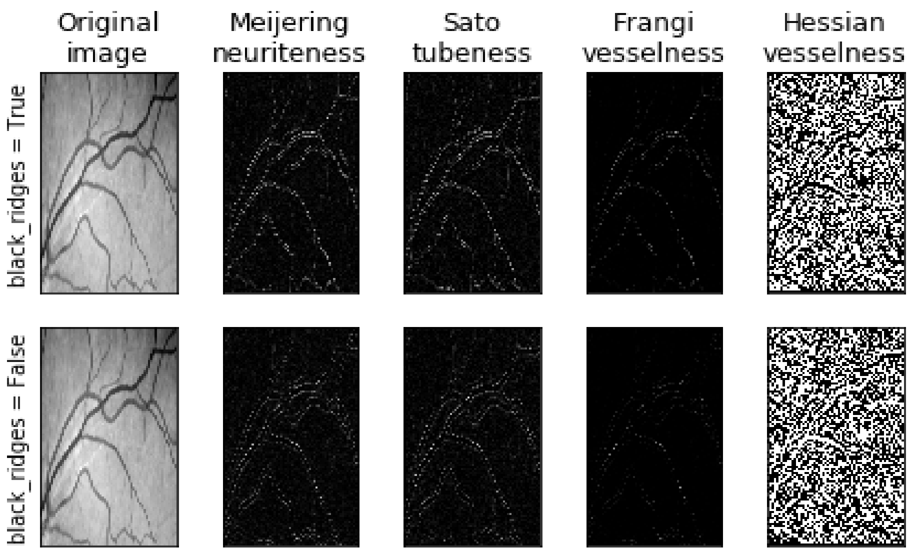
9. I applied global thresholding to image from 8, the Jaccard score is 0.703



10. I apply adaptive thresholding to image from 8, the Jaccard score is 0.651



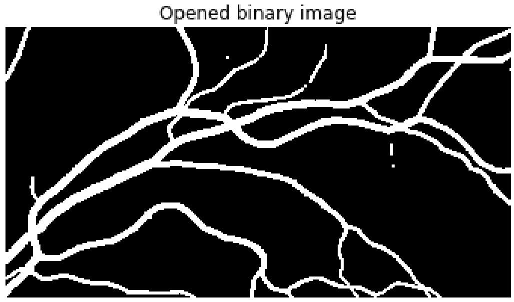
11. After that, I try applying ridge filters to the denoised image from 4. But the results are not satisfactory. The reason can be the highly noisy nature of the image, even after cleaning.



2.2:

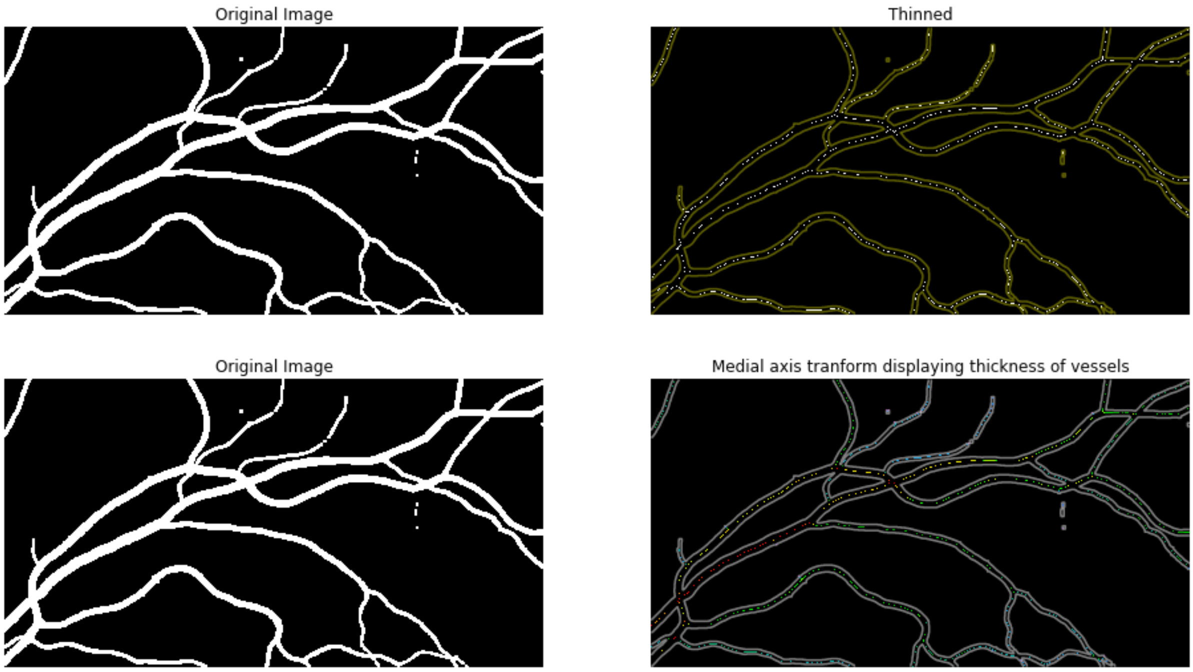
Part 1: Remove small vessels (less than 8 pixels wide) with morphological opening.

I used the manually annotated binary image provided to us, and used a square structuring element of width 8, to remove the vessels less than 8 pixels wide. This is the result:



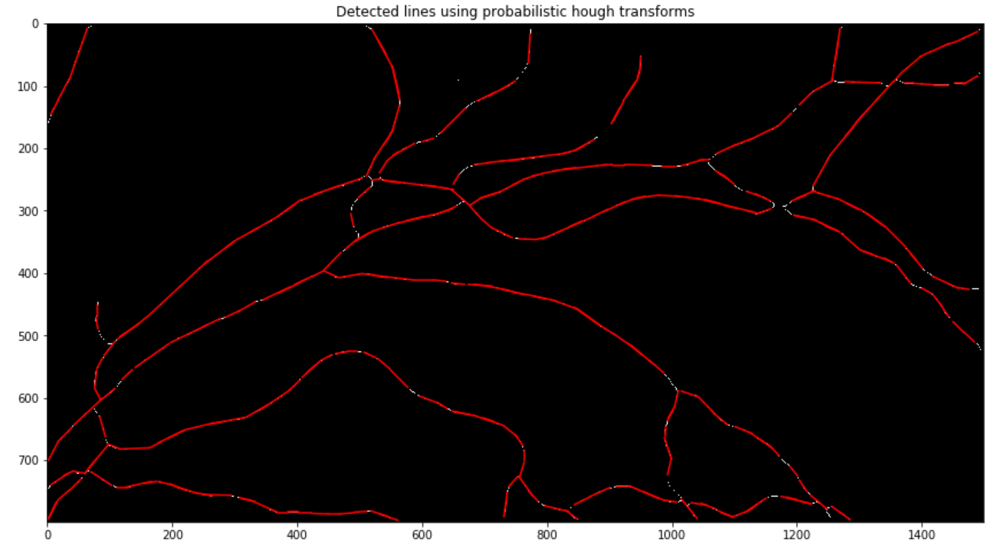
Part 2: Morphological thinning, medial axis transform and width computation

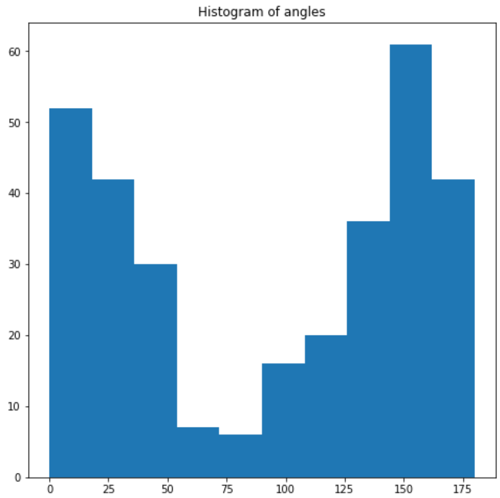
I thinned the image to compute the one-pixel wide centerline. Then I applied the medial axis transform and performed the local width computation of the vascular network. The results are displayed with the contour of the image.

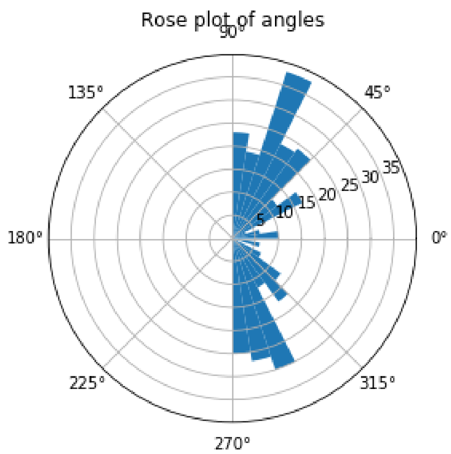


Part 3: Length and orientation of the vascular network

I applied probabilistic hough transform to the thinned image. I was able to detect line segments in the vascular network. And after obtaining the lines, I computed their orientation (in terms of angle)







Conclusion:

For the first part of the question, I got a Jaccard score of around 72% as my best result. Better denoising, and other techniques for instance CNN can boost the accuracy even further. The kernel ridge method doesn't give good results due to highly noisy nature of the image. Using the manually annotated image, I can obtain the characteristics like length and orientation of vascular network in part 2.

References:

1. Worksheets supplied during the course
2. https://scikit-image.org/docs/dev/auto_examples/edges/plot_ridge_filter.html
3. https://scikit-image.org/docs/0.10.x/auto_examples/plot_medial_transform.html
4. https://scikitlearn.org/stable/modules/generated/sklearn.metrics.jaccard_score.html#sklearn.metrics.jaccard_score