UNIVERSITY OF TEXAS ARLINGTON

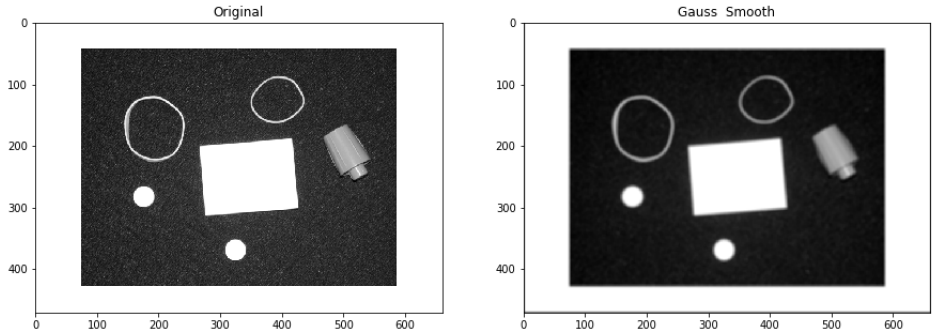
Programming Assignment 2: Using 1st of 2 day quota

Student name: *Avinash Shanker, 1001668570*

Due date: *Oct 20, 2019*

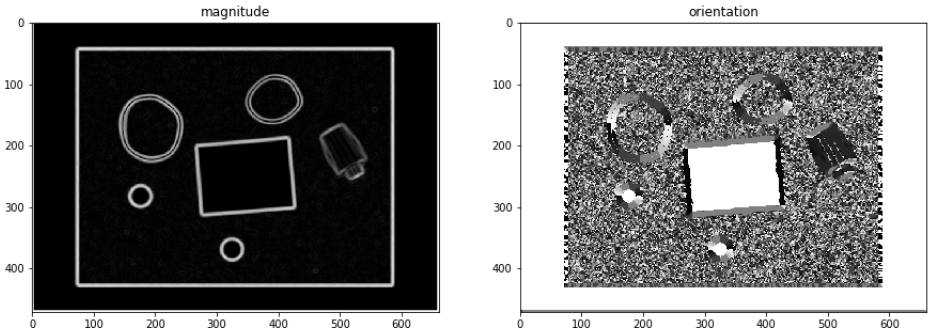
Problem 1-1

- Write a gradient based edge detector, smooth the image with a Gaussian filter and then compute horizontal and vertical derivatives using a derivative filter.
- Give Smoothing as parameter.
- compute the gradient magnitude and orientation.
- Display the gradient magnitude as a image and the orientation using quiver functionality.



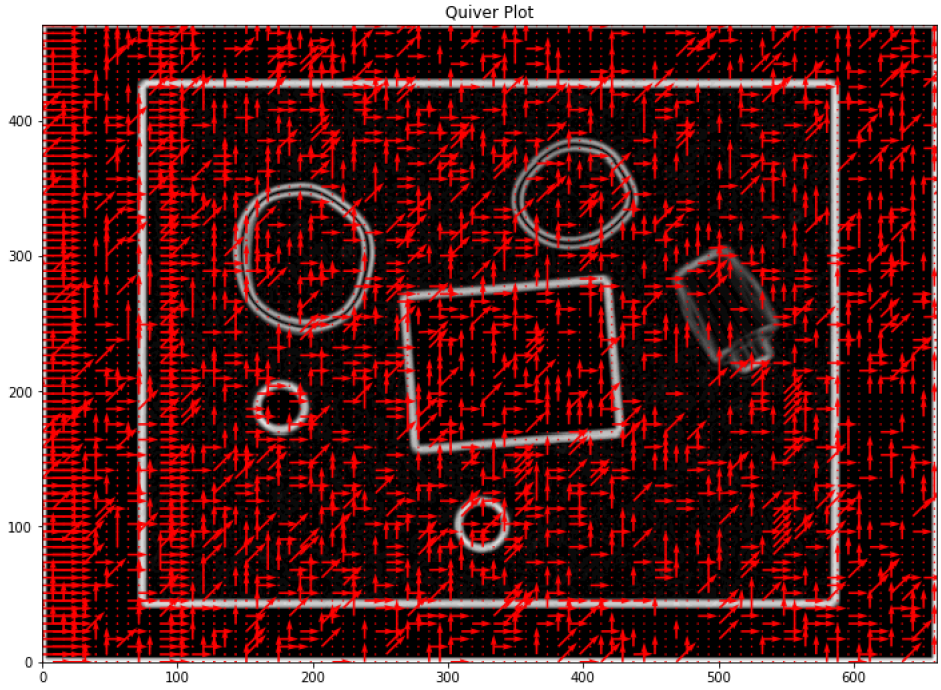
(a) Gaussian Smoothing

Figure 1: Comparing original image with Gauss smoothed setting parameter sigma=3



(a) Magnitude and Orientation

Figure 2: After performing Gaussian smoothing, Magnitude and Orientation of image are computed



(a) Quiver Plot

Figure 3: Gradient magnitude as a image and the orientation using quiver functionality.

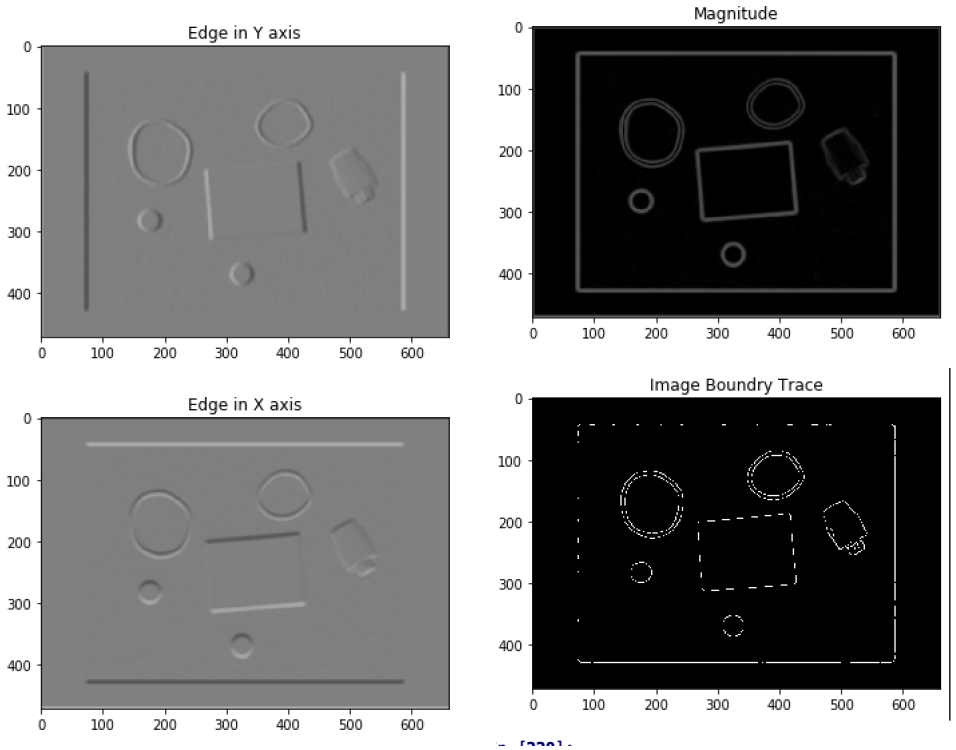
Answer.

- Fig1, Fig2, Fig3 A gradient based edge detector is written, as first step the input image is smoothed with a Gaussian filter to remove any rough edges.
- Gaussian smoothing parameter sigma is inputted form the user(eg:1,2,3,5)
- Then a gradient functions if defined to obtain the magnitude and orientation of the image.
- A quiver plot displays velocity vectors as arrows with components (u,v) at the points (x,y).

- The magnitude is displayed as a image and the orientation is superimposed on it as a quiver plot to show the direction in which the pixel is oriented.
- to ensure that visibility of the plot is clear, the pixels have been silced at a 1:4 alternation to plot the arrows.

Problem 1-2

- Extract the outer boundary of each object contained in one image.
- Apply any technique you can come up with to trace the boundary edge pixels.
- Plot the final boundary you are able to detect and trace for each object.
- Discuss the success, failure, and possible ways for improvement in the report.



(a) Gradient in X and Y axis (b) Image Magnitude and Boundary detection

Figure 4: Boundary detection, no inbuilt function used. All functions are written

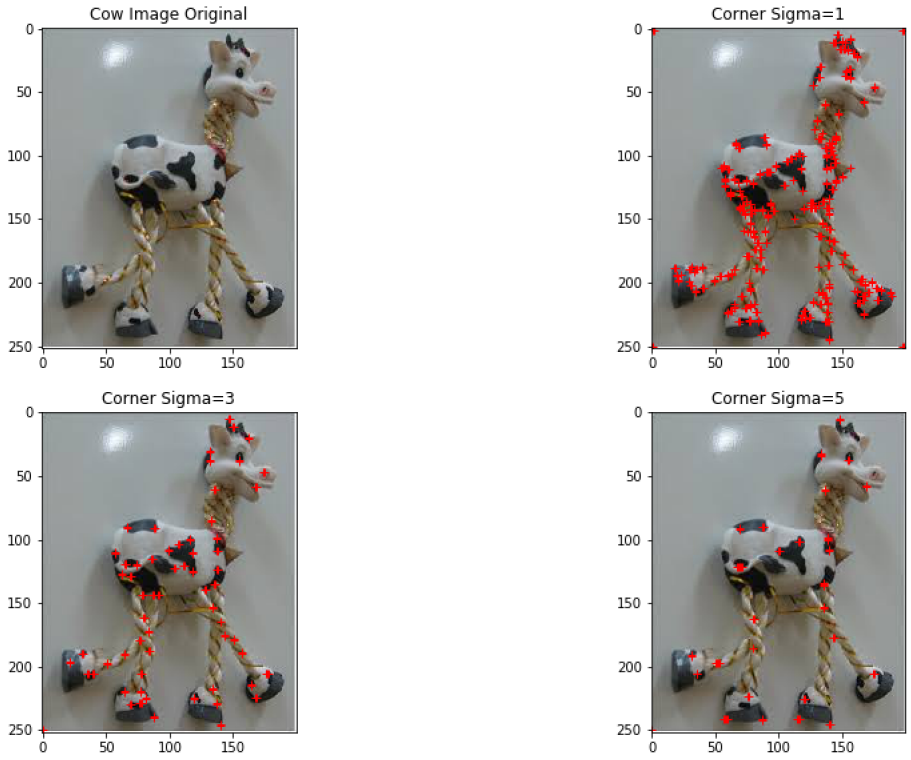
Answer.

- Figure 4b shows the boundary detected.
- To obtain the boundary of an image, as first step I have applied Gaussian blurring to smooth the image.
- I have then used sobel filter to obtain magnitude and orientation.

- Then, I have applied Non-max Suppression to remove redundant or duplicate edges. This process is to show a single line instead of multiple blurred pixels.
- To do this we compare the magnitude of the gradient between one pixel and it's neighboring pixels along the same direction and select the pixel whose magnitude is the largest.
- I then apply thresholding to all edges and define them either week (some low number say 50) or strong (white – 255).
- Then I, find out whether any given pixels neighbors has value equal to 255, if yes then change the value of the pixel to 255, otherwise discard the pixel by setting the value to 0.
- We can from the top left to right order 4 times and then return the final output.
- Failure points of my method is that it couldn't draw a complete boundary. As some neighbouring pixels are set to zero comparing the neighbouring pixels. This leads to some connected pixels not being detected.
- This can be fixed by optimising non max suppression and threshold algorithm.

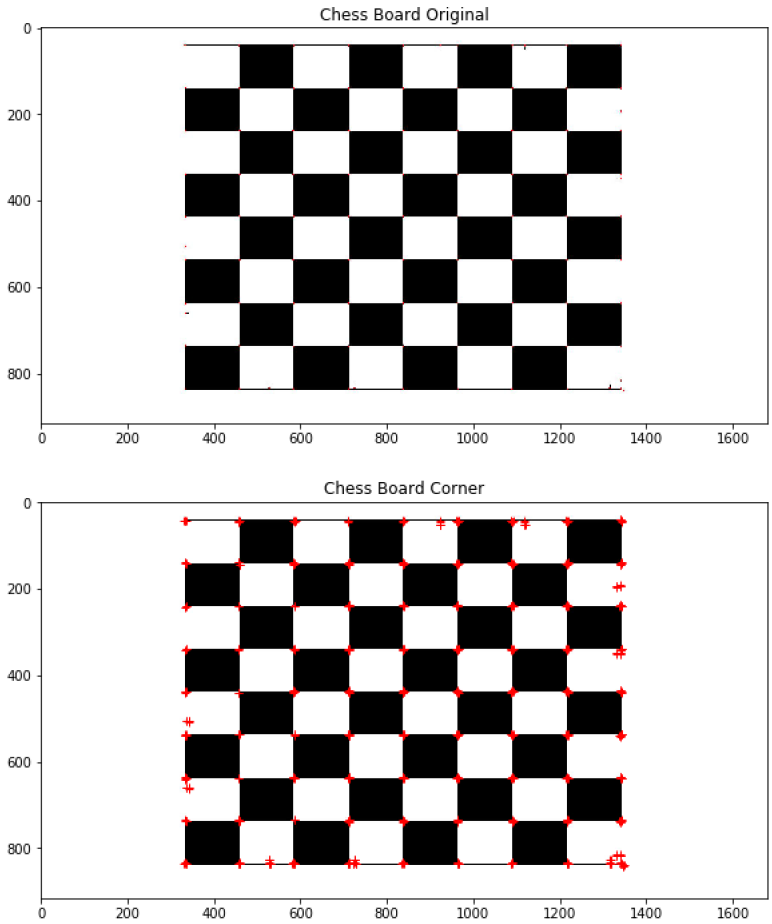
Problem 2

- Implement your own Harris corner detector.
- Your implementation should take an 2D image I as an input and compute corner responses for each pixel. Take the points of local maxima (within a radius r) of the response.



(a) Harris corner detection on cow, the number of corners detected decreases as the sigma value of Gaussian blur increases

Figure 5: Harris Filter



(a) Harris corner detection on a chess board, all the edges are detected correctly

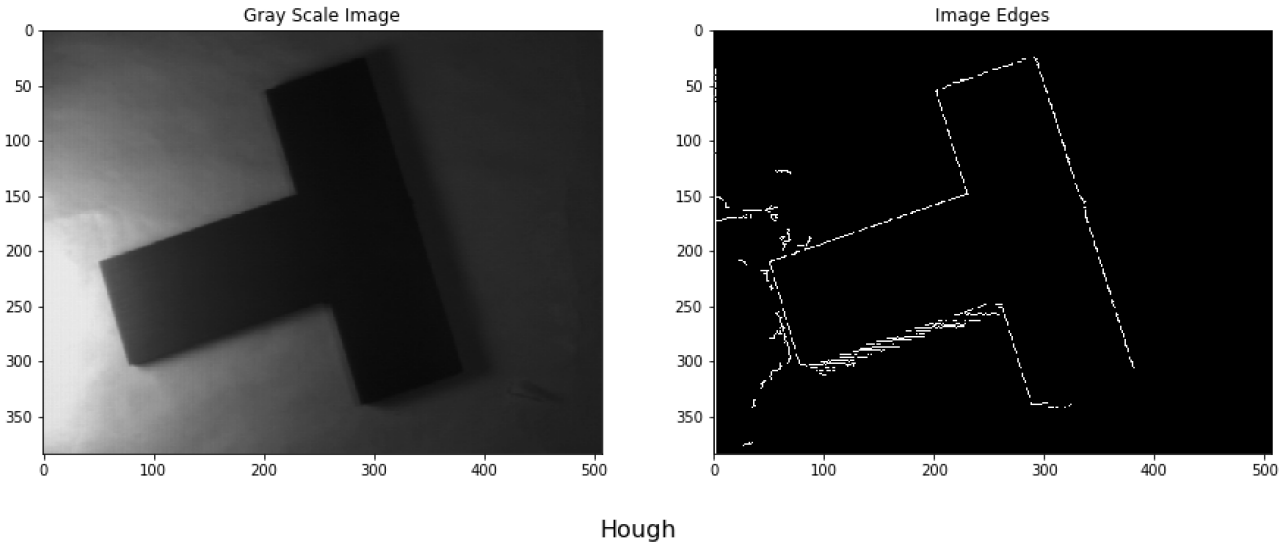
Figure 6: Harris Filter

Answer.

- Fig5 and Fig6 A corner can be interpreted as the junction of two edges, where an edge is a sudden change in image brightness.
- In figure5 Cow image, Harris corner detector algorithm is implemented. The image is first passed through Gaussian filter before performing Harris detection.
- It can be noticed that, the number of corners detected is decreasing as, the value of sigma in Gauss blur is increased.
- It is optimum to have sigma value between 1 and 3 for correct corner extraction.
- Points of local maxima of the response is taken

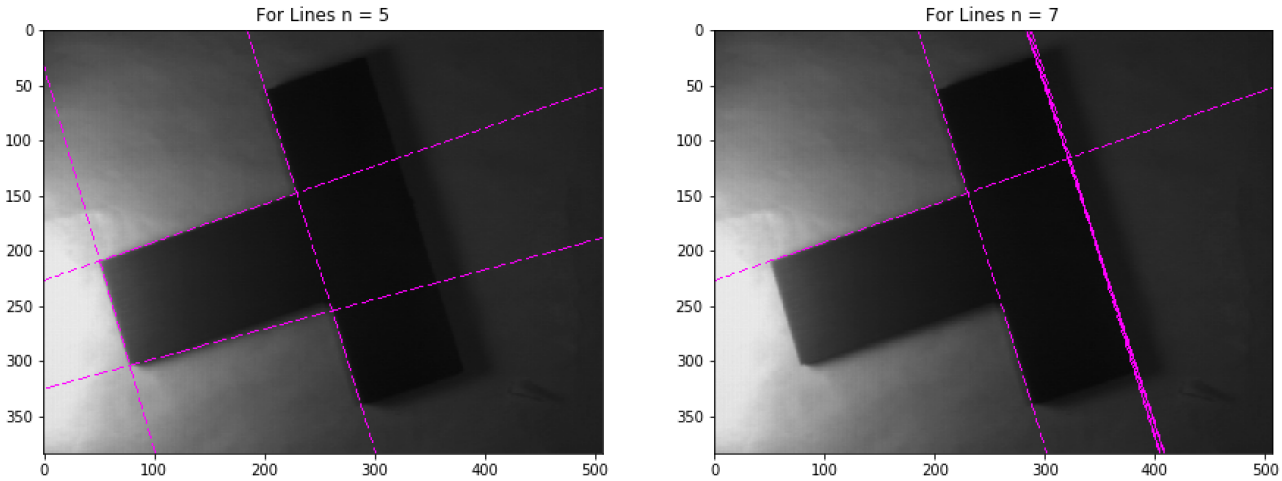
Problem 3-1

- Automatically identify straight lines in an image: lines = myHoughLine (imBW, n), where imBW is a binary image and n is the desired number of lines (in order of voting importance).



(a) Left is BW image and right is Canny filter applied for edges

Figure 7: Hough Transformation



(a) Left image is where N=5 and Right image is N=7. But, has multiple lines overlapping

Figure 8: Hough Transformation Output which identifies straight lines in order of voting

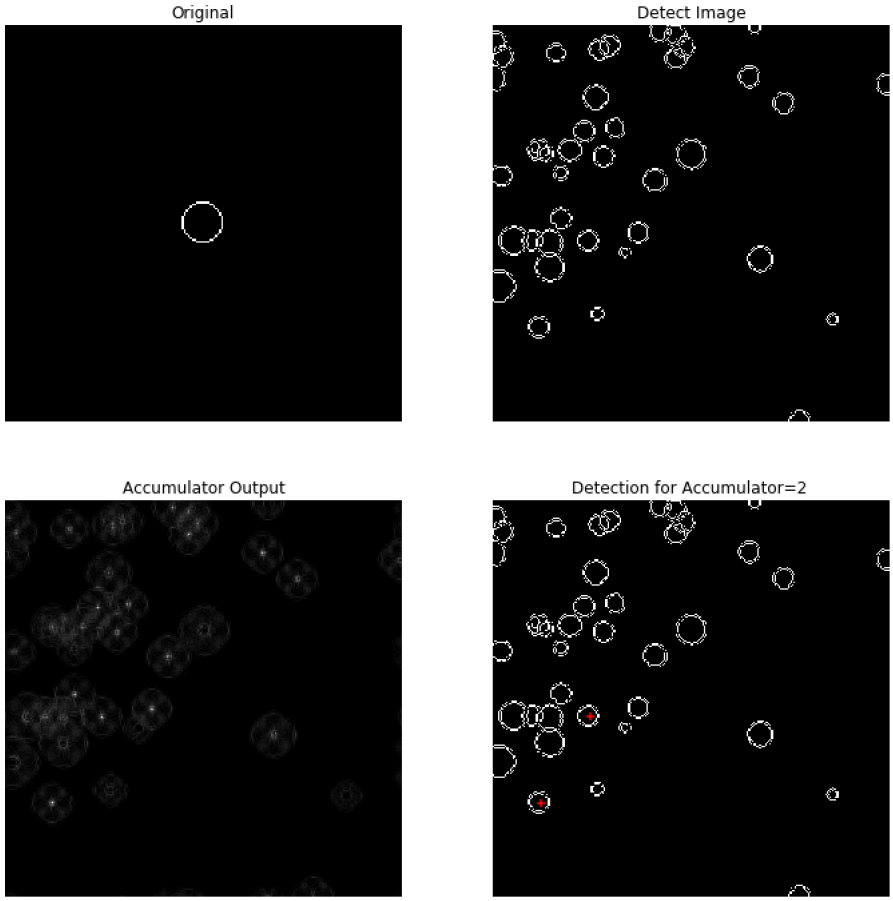
Answer.

- Fig7 and Fig8 The canny edge filter is used on original image which is grayscaled. This filter provided edges of the input image.

- Once edges are obtained, Hough transform is applied to detect the lines in the order of voting. For the first test where N=5. The output gives 4 lines. The minimum threshold=25 and maximum threshold=50 for the canny function.
- For the next test, N=7 The minimum threshold=0 and maximum threshold=50 generated upto approx 5 lines which are overlapping.

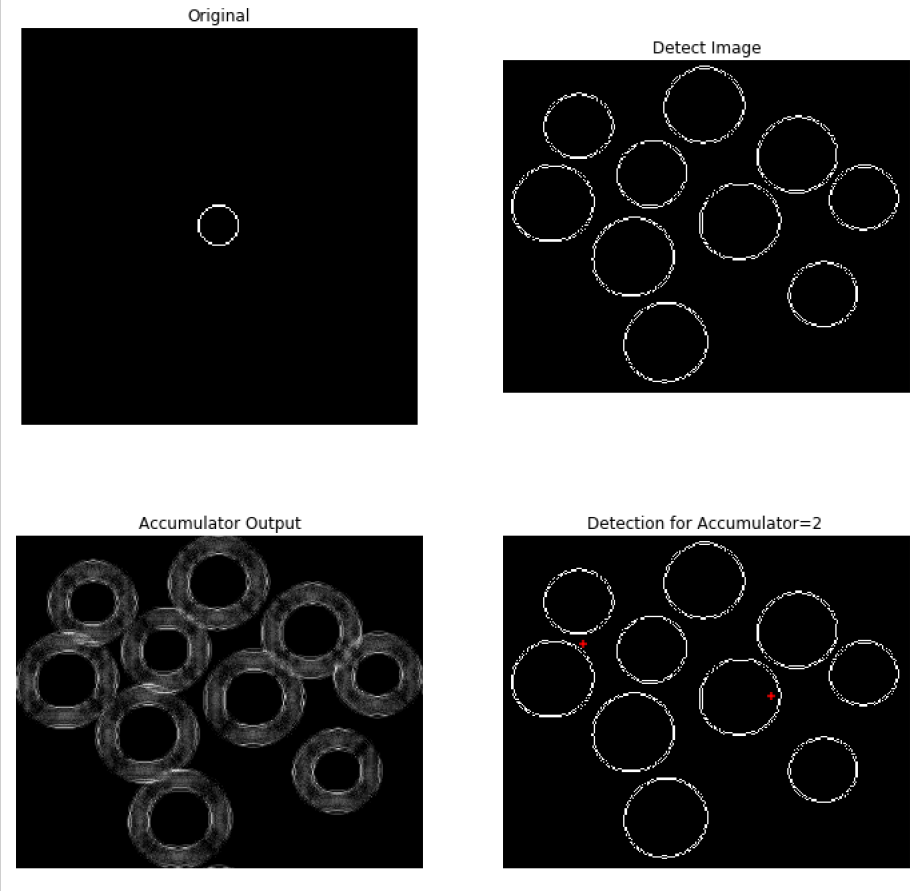
Problem 3-2

- Write two functions, yourcellvar = myHoughCircleTrain(imBW, c, ptlist), and myHoughCircleTest(imBWnew, yourcellvar).
- The binary image imBW supplied to the first function will have a single circular object. Further, c will be the (given) reference point, and for convenience we will also provide you an ordered list of boundary points in ptlist.
- You will identify circular objects in a novel image, imBWnew. Your function should report the reference points for the top two circles identified. You will receive full credit if your reference point is close enough.



(a) Hough Circle Train

Figure 9: Hough Circle Train Top 2 results for given data 1st and 2nd Circle coordinates are (97,218) and (47,305)



(a) Hough Circle For my test image

Figure 10: Hough Circle Train Top 2 results for my test data

Answer.

- Plist and C are used to build a rtable based in the train image.
- utilizing the rtable my code will look the shape provided in the test image.
- My code could find the coordinates of the top two matching circles. Results for given data 1st and 2nd Circle coordinates are (97,218) and (47,305).
- To test my work further u have used another test input which have similar circles.
- myHoughCircleTrain function worked well with a minute . detection error for the seconday test image. But, if the number of results are increased the detection rate is better.