INTRODUCTION

Line Detection and Circle Detection is implemented by using built-in Hough transform function in MATLAB during the lab.

Line Detection is basically finding all possible lines on which a collection of n edge points.

Circle Detection with Hough Transform is using “voting” in the Hough space to find circles.

Throughout this report, I shortly explain Line and Circle Detection with Hough transform. Then, I will present my results. At the end of each part, I will present a discussion about the detector.

1. Line Detection

Line Detection with Hough transform in MATLAB is implemented during the lab. Hough Transform is basically means that mapping the edge image to the Hough space. At each point of the parameter space is represented by a sinusoid:

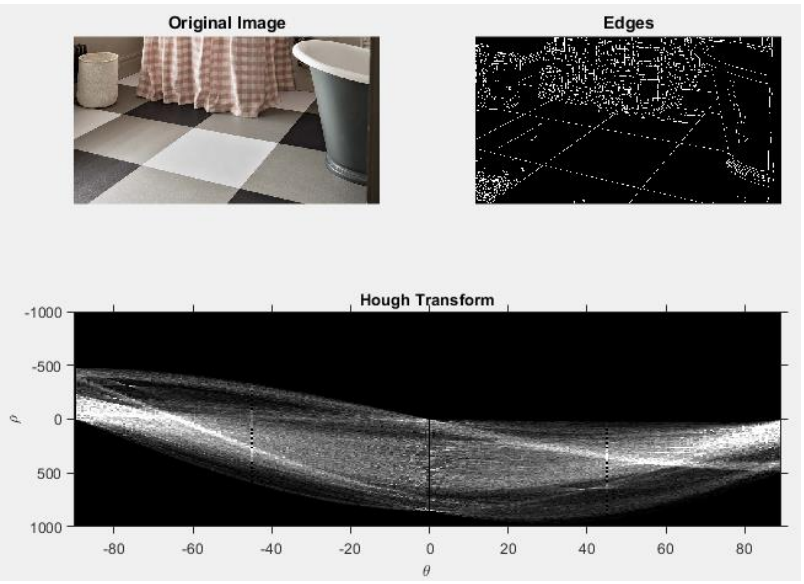
$$\rho = x\cos\theta + y\sin\theta$$

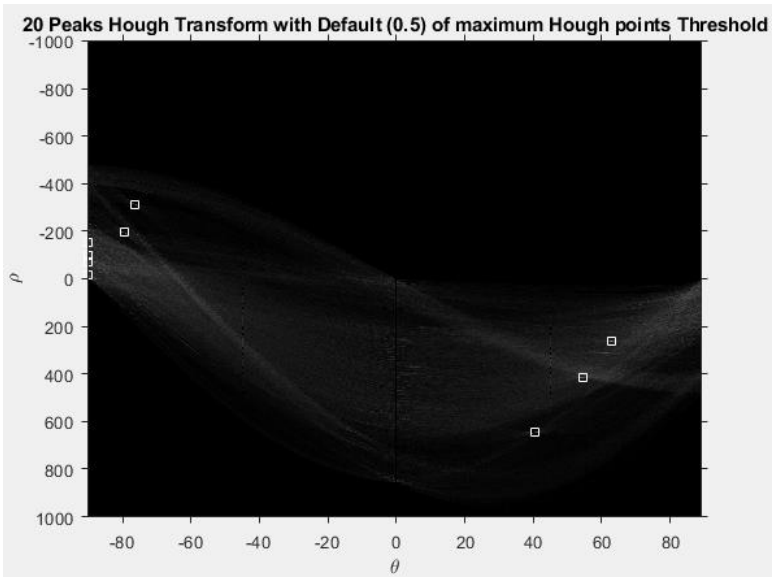
Then, for each point find the ρ and θ pairs and increment the accumulator for these pairs. At the end, peak points of the Hough space can be found by applying thresholding.

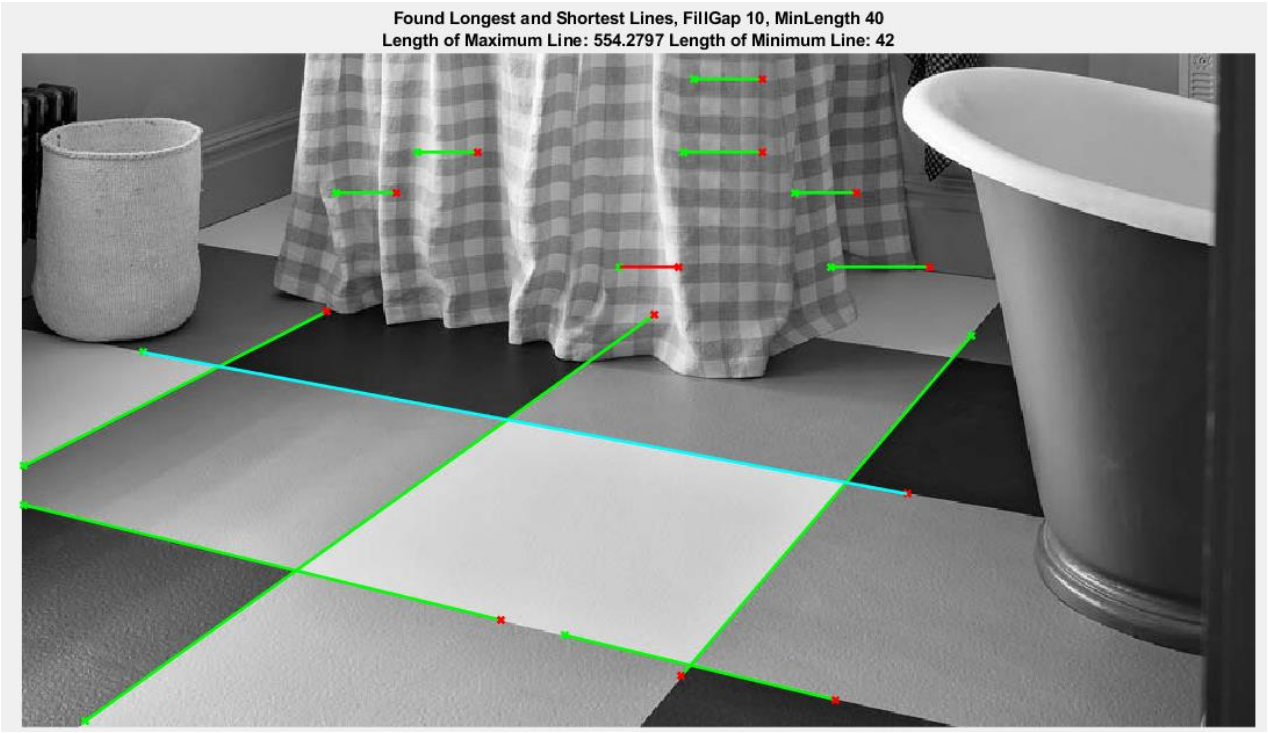
1.1 Line Detection INLAB and POSTLAB

Firstly, we need to obtain edge image for our input image. Therefore, I used ‘edge’ built-in function with Canny method. Then, we can apply ‘hough’ built-in function. This function returns Hough Transform matrix where rows and columns correspond to ρ and θ values respectively. Then, we can use the ‘houghpeaks’ with user defined number of peaks parameter and proper threshold value. This function will return coordinates of these peaks. At the end, we can use these points to detect lines in our image.

1.2 Results

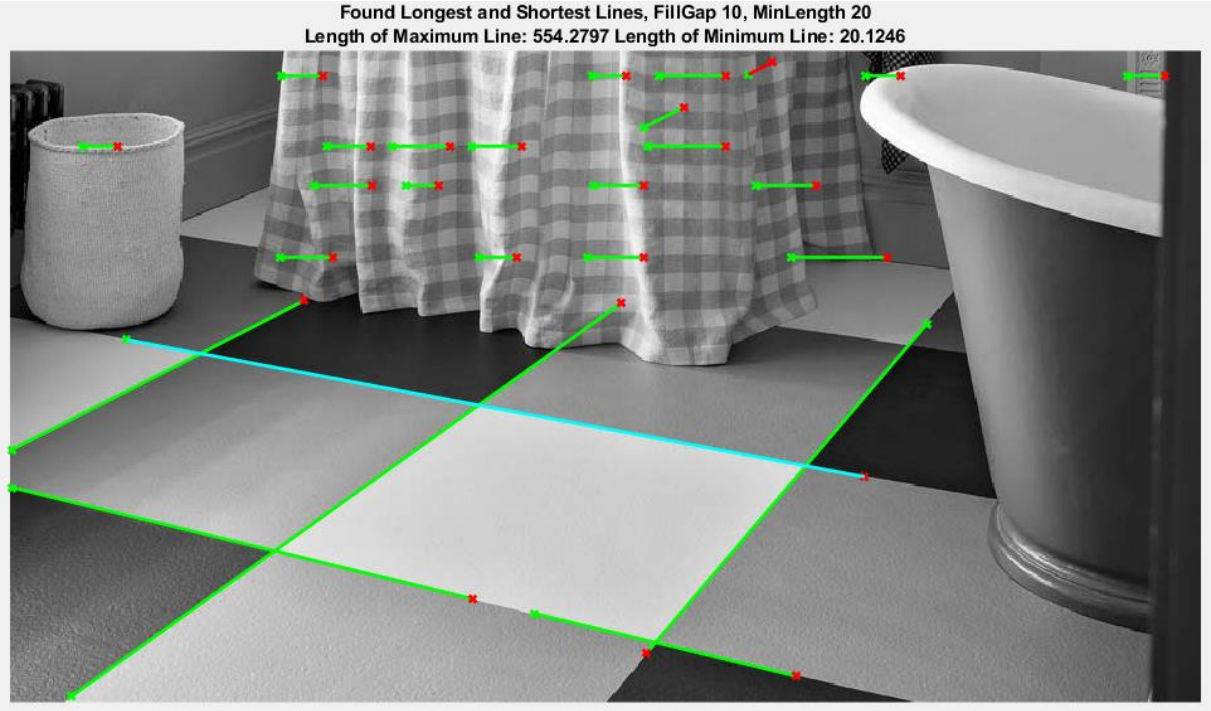




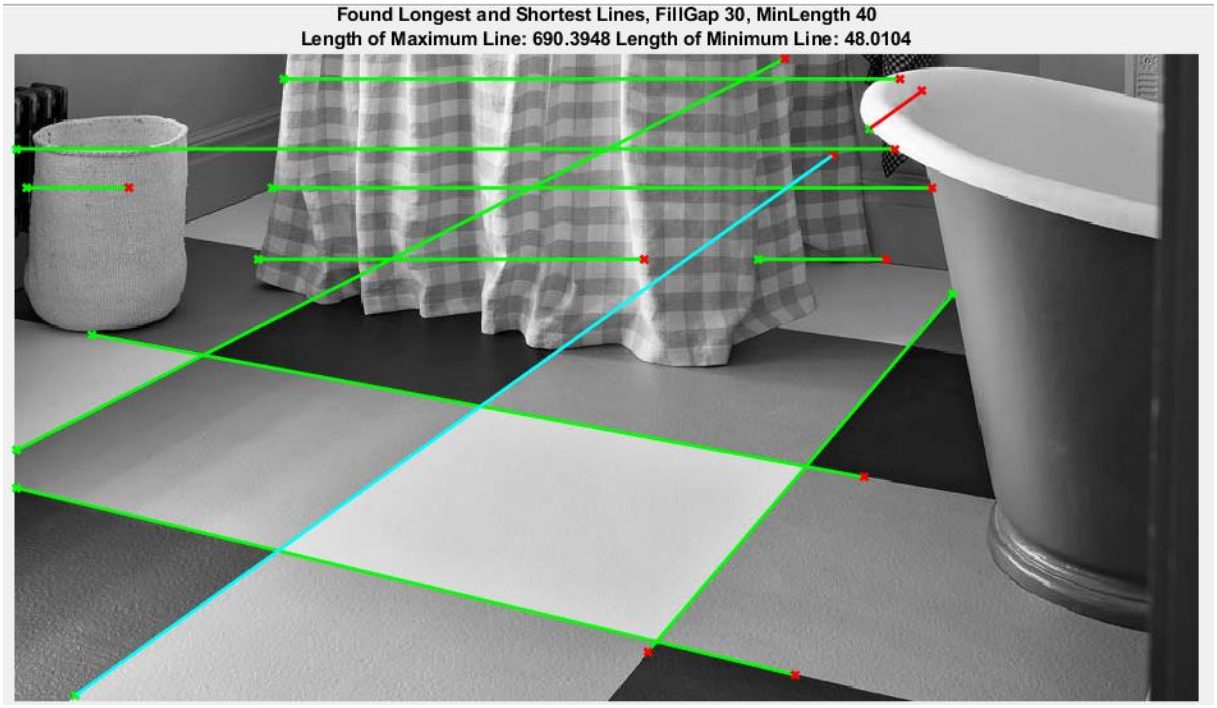


Length of the Maximum Line and Length of the Minimum line can be seen in the title of the image.

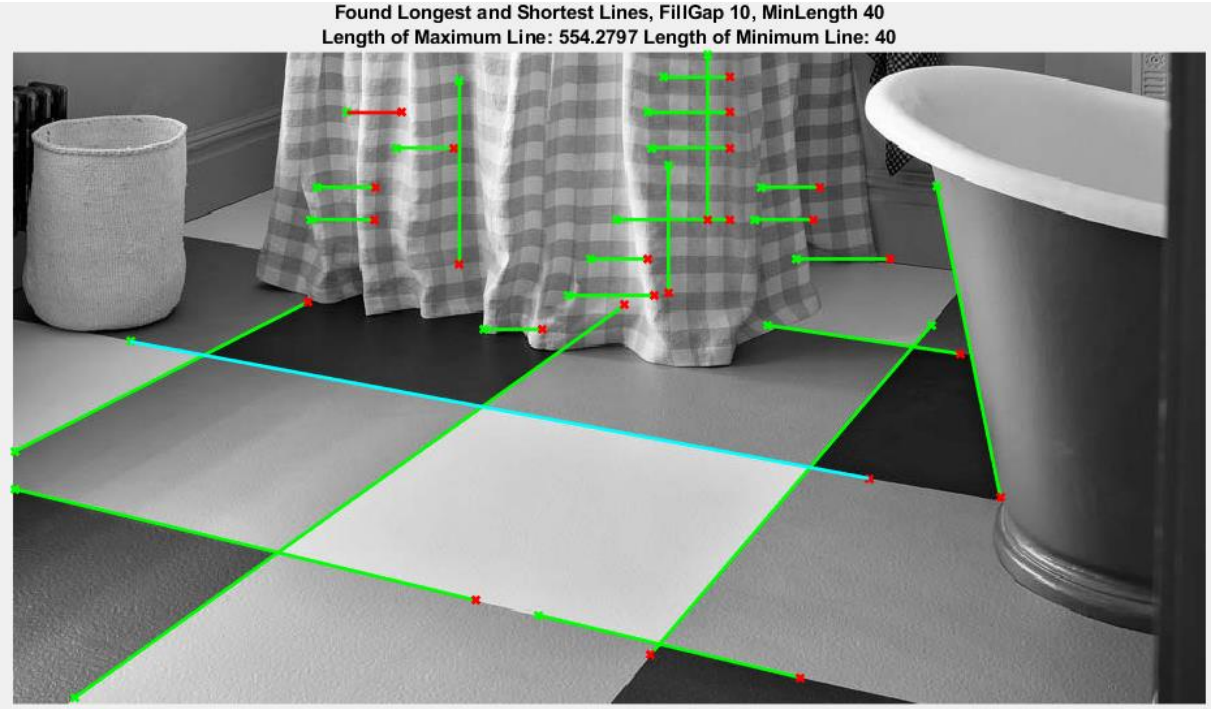
Changing MinLength parameter:



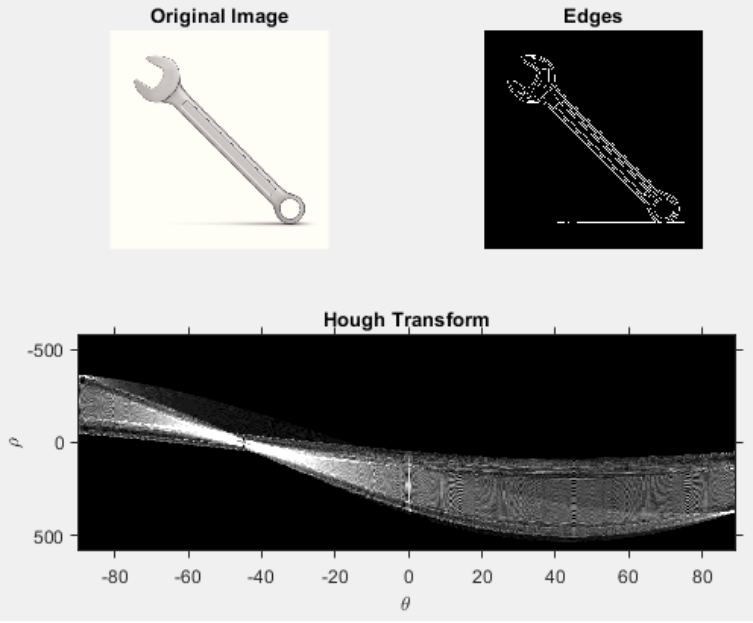
Changing FillGap parameter:

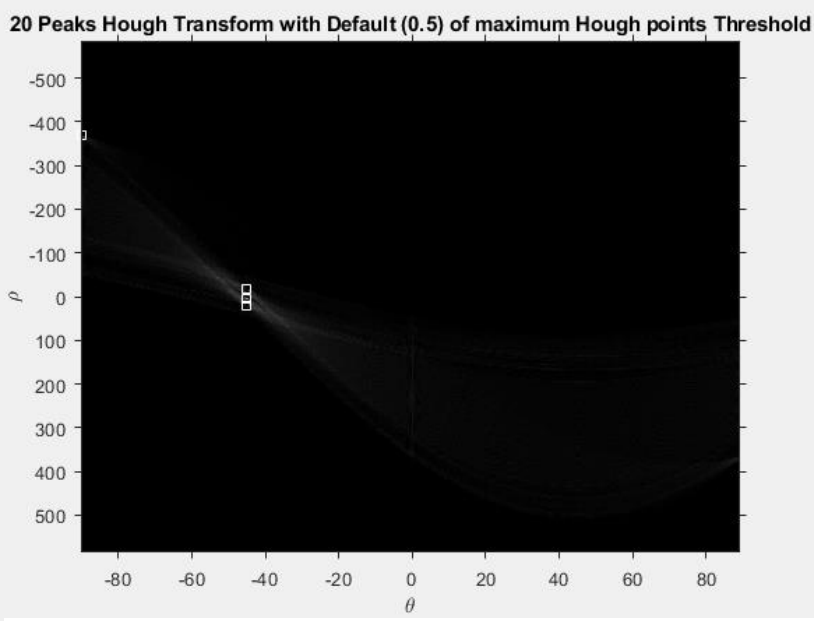


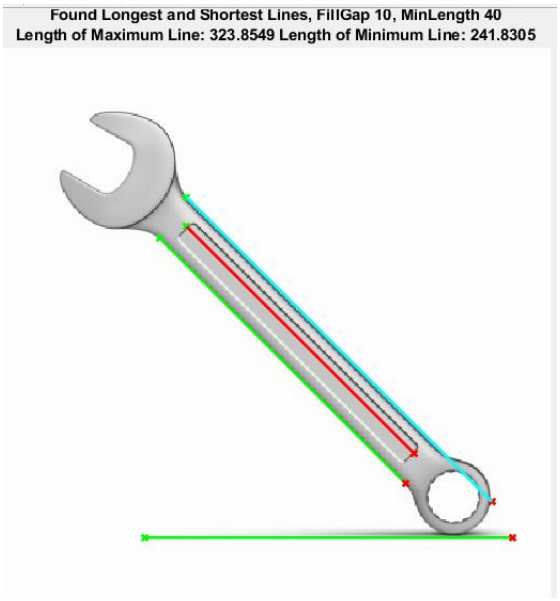
Changing Threshold Parameter (Threshold = ceil(0.3*max(H(:)))):



Different Image:







1.3 DISCUSSION

We can observe that some lines are succesfully detected by Hough Transform. All lines are indicated with green, longest line is indicated with cyan and the shortest line is indicated with red. Corresponding double lengths can be seen in the title of the images.

We observe that if we play with minLength parameter of the 'houghlines' the number of lines in the output image is changing. If we pick too low minLength parameter, we can end up with wrong classified lines in the output. On the other hand, if we pick too large minLength parameter, then we may miss the real lines in the image. For instance, detector could not catch lines in the high part of the wrench image as these lines have lower length than the parameter.

We also observe that if we choose low FillGap parameter, we may miss the opportunity of merging two lines which are connected in the real image. However, if we choose too high for FillGap parameter, we may end up merging distinct lines with each other which is wrong approach.

In addition to these, we should also pick the threshold value properly for choosing peaks to not miss or wrong classification of peaks in the Hough Space.

Therefore, all of these parameters (minLength, FillGap, Threhsold) should be choosen wisely.

2. Circle Detection

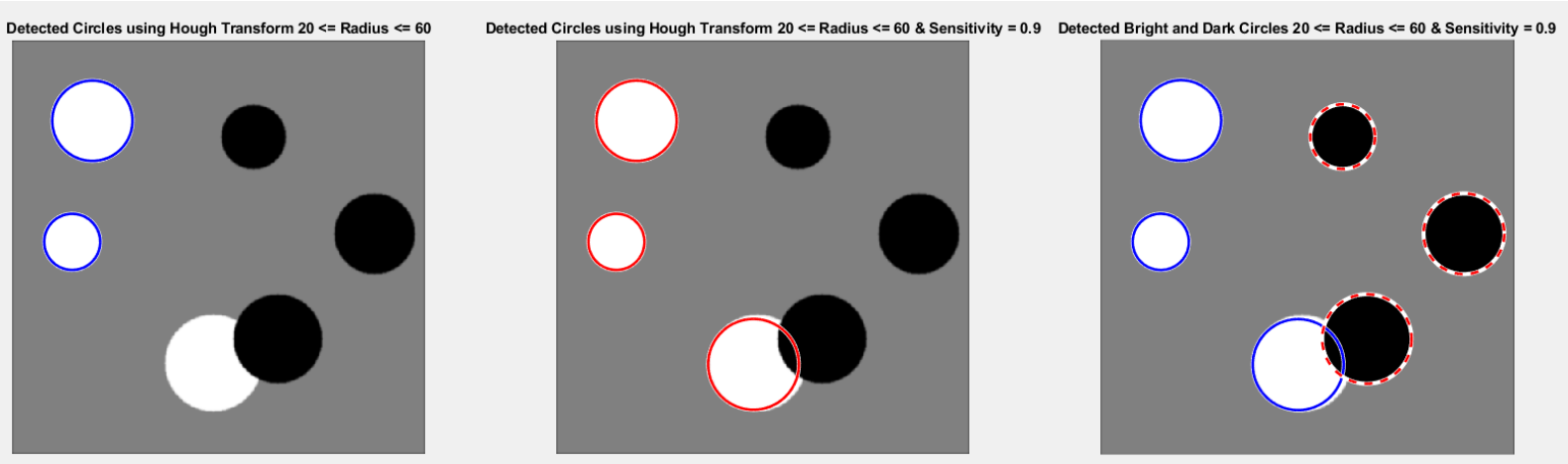
Circle Detection with Hough transform in MATLAB is implemented during the lab. Hough transform is using voting scheme to describe a line or curve. According to these votes, Hough transform chooses the representation of corresponding set of points.

2.1 Circle Detection INLAB and POSTLAB

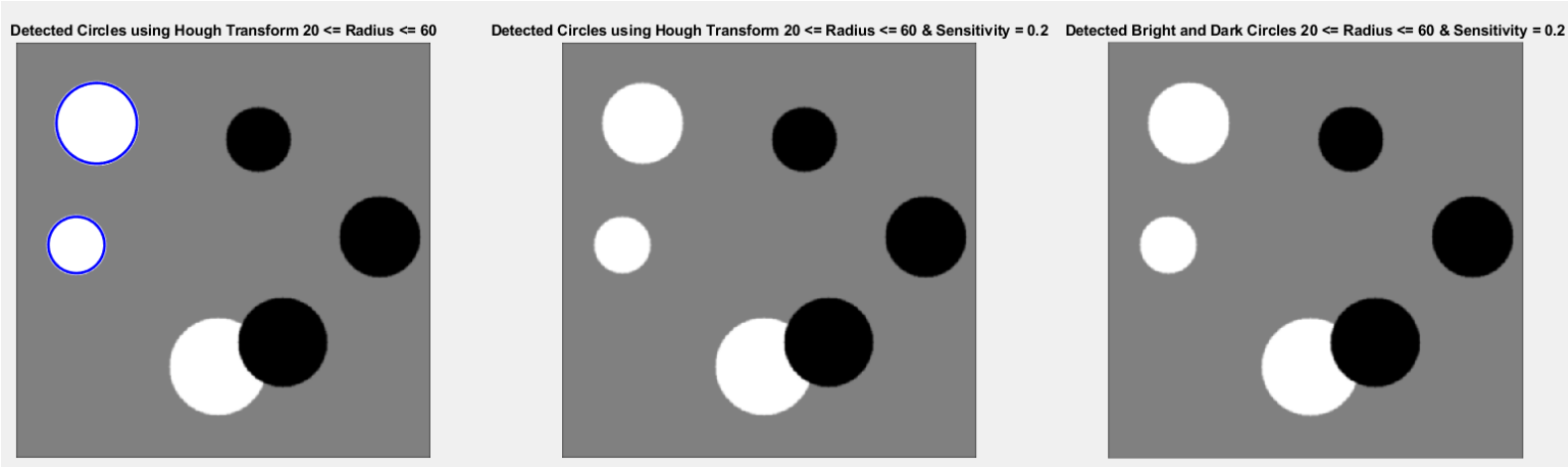
We used imfindcircles built-in function of MATLAB to detect coordinates of circles' centers. This function returns the coordinates of centers which have radius in the predetermined given range. Then, we can use viscircles to show these circles on the our image.

We have two important parameters for the imfindcircles function. Firstly, the sensitivity parameter is used as a kind of threshold. If it is increased, the number of circles in the output increase as well as the false positive rate. Secondly, the ObjectPolarity parameter is specified to catch bright or dark circles according to background of the image.

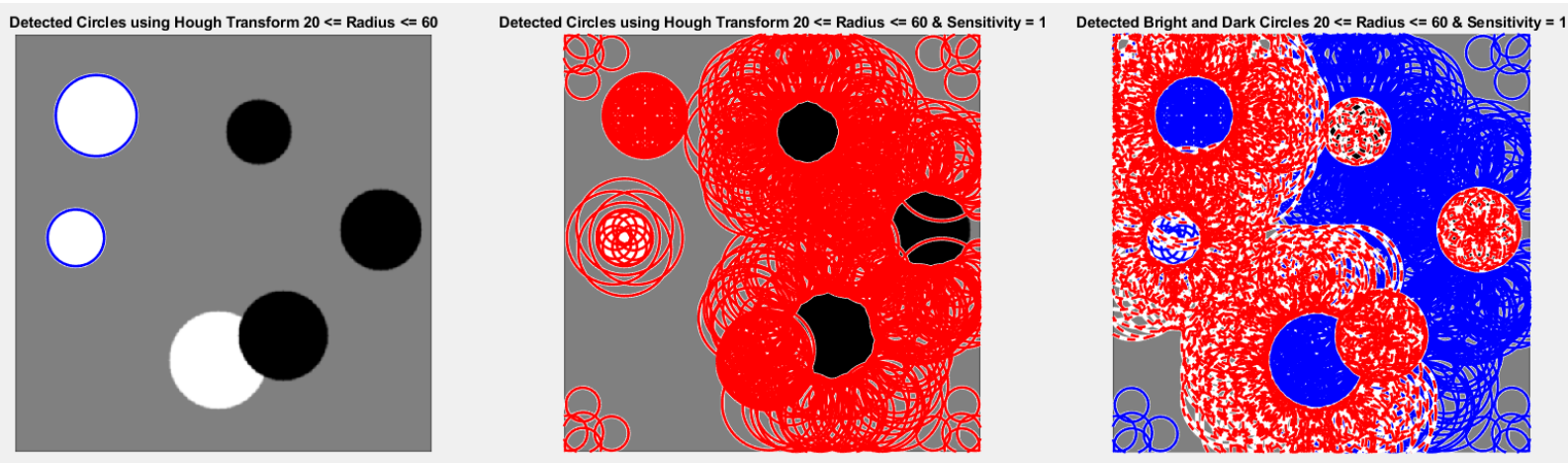
2.2 Results



Sensitivity = 0.2:



Sensitivity = 1:



Different Image:

2.3 DISCUSSION

We can observe that we are able to catch circles with Hough Transform. Sensitivity parameter is playing crucial role in this detector. If we choose too low value, we missed the real circles in the images. However, if we choose too high value for sensitivity, we ended up with too many false positive circles. Therefore, we need to find optimal sensitivity for the detector.

We can observe that when we may specify the circles based on their brightness level. For instance, when we choose dark as the object polarity, we spesifically showed the darker circles in the third image with dashed lines.

We can also observe that even circles in the first image or balls in the second image do not have all circle part the input image, Hough transform is able to catch these circles. What I mean is that even some part of the circle is getting blocked by another object in the image, we are able to catch these circles.

APPENDIX FOR CODES

MAIN

```
checker = imread("checker.jpg");
circles = imread('circlesBrightDark.png');
wrench = imread("wrench.jpg");
balls = imread("balls.jpg");

% % Line Detection
lab5houghlines(checker);
% lab5houghlines(wrench);


% % Circile Detection
lab5houghcircles(circles, 0.9);
% lab5houghcircles(circles, 0.2);
% lab5houghcircles(circles, 1);
% lab5houghcircles(balls, 0.9);
```

HOUGHLINES

```
function lab5houghlines(img)
% Ensuring grey scale image
new_img = img;
[row,col,ch] = size(img);
if(ch == 3)
    img = rgb2gray(img);
end

BW = edge(img,'canny');
[H,T,R] = hough(BW,'RhoResolution',0.5,'Theta',-90:0.5:89);

figure()
subplot(2,2,1);
imshow(new_img);
title('Original Image');

subplot(2,2,2);
imshow(BW);
title('Edges');

subplot(2,2,[3,4]);
imshow(imadjust(rescale(H)),'XData',T,'YData',R,...
      'InitialMagnification','fit');
title('Hough Transform');
xlabel('\theta'), ylabel('\rho');
axis on, axis normal, hold on;
```

```
peaks = 20;
P  = houghpeaks(H,peaks);
figure()
imshow(H, [] ,'XData',T,'YData',R,...
      'InitialMagnification','fit');
xlabel('\theta'), ylabel('\rho');
axis on, axis normal, hold on;
plot(T(P(:,2)),R(P(:,1)),'s','color','white');
title(peaks + " Peaks Hough Transform with Default (0.5) of maximum Hough points Threshold")
% Find lines and plot them
fillgap = 10;
minlength = 40;
lines = houghlines(BW,T,R,P,'FillGap',fillgap,'MinLength',minlength);
figure()
imshow(img);
hold on
max_len = 0;
min_len = 9999999;
for k = 1:length(lines)
 xy = [lines(k).point1; lines(k).point2];
 plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green');
 % plot beginnings and ends of lines
 plot(xy(1,1),xy(1,2),'x','LineWidth',2,'Color','green');
 plot(xy(2,1),xy(2,2),'x','LineWidth',2,'Color','red');
 % determine the endpoints of the longest and shortest line segment
 len = norm(lines(k).point1 - lines(k).point2);
```

```
 if ( len > max_len)
   max_len = len;
   xy_long = xy;
 end
 if (len < min_len)
    min_len = len;
    xymin_long = xy;
 end
end

% highlight the longest line segment
plot(xy_long(:,1),xy_long(:,2),'LineWidth',2,'Color','cyan');

% highlight the shortest line segment
plot(xymin_long(:,1),xymin_long(:,2),'LineWidth',2,'Color','red');
title({"Found Longest and Shortest Lines, FillGap "+ fillgap + ", MinLength "+minlength; "Length of Maximum Line: "+max_l
```

HOUGHCIRCLES

```
function lab5houghcircles(img, sensitivity)
% Ensuring grey scale image
new_img = img;
[row,col,ch] = size(img);
if(ch == 3)
    img = rgb2gray(img);
end
Rmin = 20;
Rmax = 60;
[centers, radii, metric] = imfindcircles(img,[Rmin, Rmax]);
figure()
subplot(1,3,1)
imshow(img)
viscircles(centers, radii, 'Color','b');
title("Detected Circles using Hough Transform "+Rmin + " <= Radius <= " + Rmax)
subplot(1,3,2)
imshow(img)
[centers, radii, metric] = imfindcircles(img,[Rmin, Rmax], "Sensitivity", sensitivity);
viscircles(centers, radii, 'Color','r');
title("Detected Circles using Hough Transform "+Rmin + " <= Radius <= " + Rmax + " & Sensitivity = "+ sensitivity)
subplot(1,3,3)
imshow(img)
[centers, radii, metric] = imfindcircles(img,[Rmin, Rmax], "Sensitivity", sensitivity);
viscircles(centers, radii, 'Color','b');
[centers, radii, metric] = imfindcircles(img,[Rmin, Rmax], "Sensitivity", sensitivity, "ObjectPolarity", "Dark");
viscircles(centers, radii, 'Color','r',  "LineStyle", "--");
title("Detected Bright and Dark Circles "+Rmin + " <= Radius <= " + Rmax + " & Sensitivity = "+ sensitivity)
```