**EE 417 - Computer Vision**

**Post Lab Report**

Smoothing, Sharpening and 1st Derivative Filters

**Name:** Çiğdem Ceyda Düzgeç

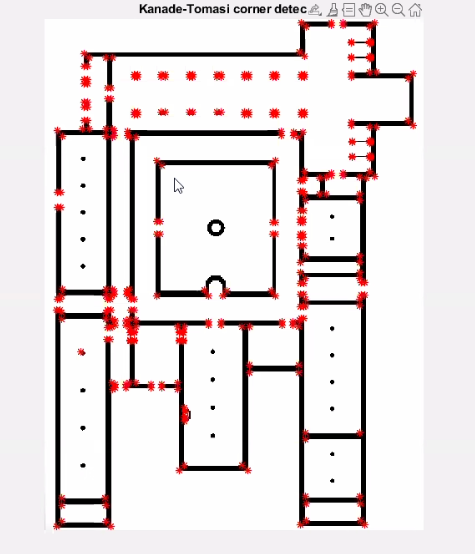
**Student ID:** 23928



**Date: 12.11.2020**

**1.Corner Detection**

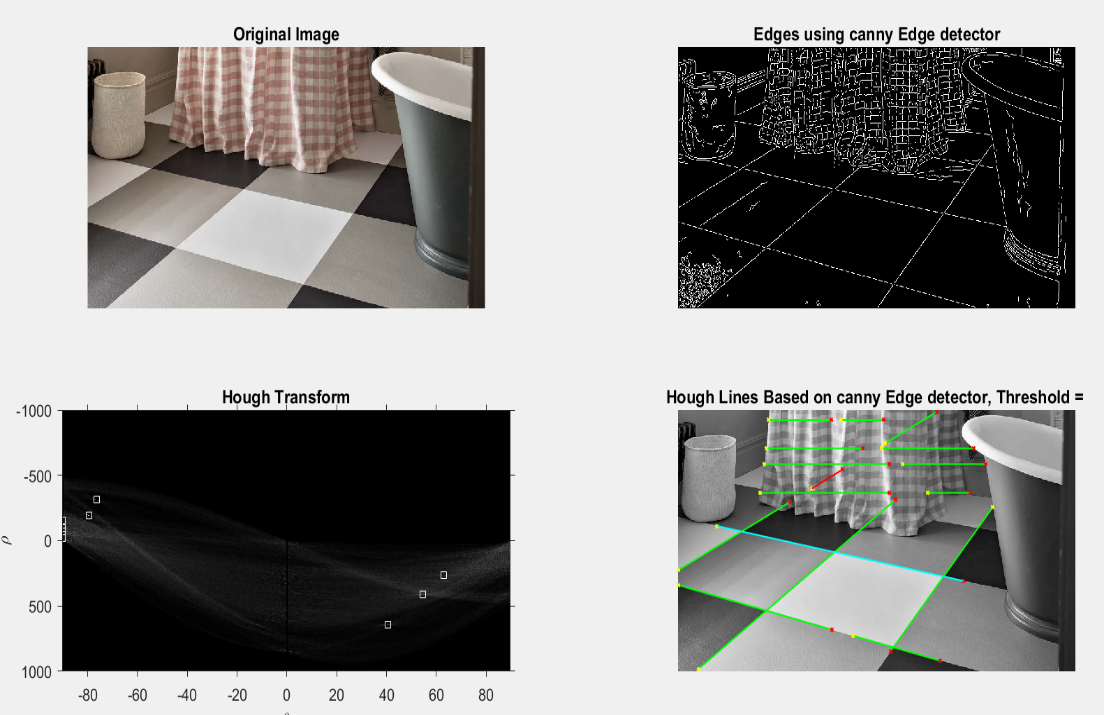
Tomasi Kanade cornerness measure uses minimum of eigenvalues of the image. We first smoothened the image to reduce noise then compute the gradients as Gx and Gy. Then we need to create a H matrix which is [Ix^2 IxIy; IxIy Iy^2]. In order to compute this matrix we created two loops inside of one and another to iterate through the window as we calculated Ix^2, Iy^2, IxIy then we put these values into H matrix. Since Tomasi Kanade uses the minimum of eigenvalues we compared eigenvalues and take the smaller one and process it as corner.



**2. Line Detection**

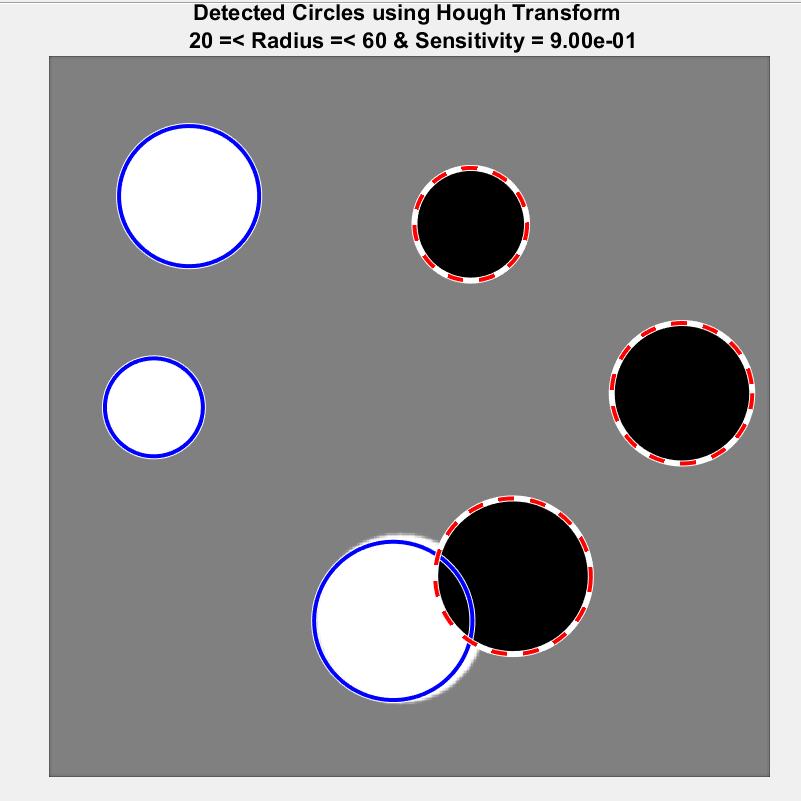
When detecting lines with Hough Transform there are few steps. Firstly, an edge detection algorithm should be used preferably Canny. Then edge points should be mapped to the Hough space and storage in accumulator. Then with the computed accumulator should be interpreted by thresholding.

In the code we first converted image to grayscale then used Canny edge detection. After that we computed H transform matrix with hough function which also gives as rows and columns of the matrix. We used these values later to show to figure with different colors of lines. Then we find hough peak points using the threshold. And used this threshold to compute lines. We selected maximum and minimum lengthvalues. So that we can plot the beginnings and ends of lines. We choose the longest line segment by comparing the length of the line with the min and max values. And after determining the longest lines we showed the figures.



**3. Circle Detection**

For detecting the circles with hough transform we first converted image to greyscale. Then we created a minimum and maximum value for radius for circles and also a sensitivity value. We used these values to find circles with imfindcircles function of MATLAB’s. We find bright and dark circles separately.



**Appendix**

**a.lab4main.m**

%% Q1 Corner Detection

clear all; close all; clc;

tic

img= imread ('Monastry.bmp');

imshow(img);

Corners\_Filtered = lab4ktcorners(img);

toc

%% Q2 Hough Lines

clear all; close all; clc;

tic

img= imread('checker.jpg');

[H,thet,rho] = lab4houghlines(img);

%% Q3 Hough Circles

clear all;close all;clc;

tic

img = imread('circlesBrightDark.png');

[centersBright, radiiBright,centersDark, radiiDark] = lab4houghcircles(img);

toc

**b.lab4ktcorners.m**

function [Corners\_Filtered] = lab4ktcorners(img)

[row, col, ch] = size(img);

if (ch == 3)

img = rgb2gray(img);

end

Corners\_Filtered = zeros( size(img));

% Transform double

img= double(img);

Corners\_Filtered = double(Corners\_Filtered);

% initialize o corners threshold window size

corners = [];

T = 400000;

k= 1;

%smoothing

Simg= imgaussfilt(img);

%Find Gradients

[Gx,Gy] = imgradientxy(Simg);

for i=k+1:1:row-k

for j = k+1:1:col-k

WindowX = Gx(i-k:i+k,j-k:j+k);

WindowY = Gy(i-k:i+k,j-k:j+k);

xx = sum(sum(WindowX.\*WindowX));

yy = sum(sum(WindowY.\*WindowY));

xy = sum(sum(WindowX.\*WindowY));

end

end

H = [xx xy ; xy yy];

[e1, e2] = eig(H);

if (min(e1, e2)>T)

corners = [corners;i, j];

end

%visualization

img = uint8(img);

figure;

imshow(img)

hold on;

plot(corners(:,2), corners(:,1), 'r\*', 'MarkerSize',7,'LineWidth',1);

title('Kanade-Tomasi corners detection');

end

**c. lab4houghlines.m**

function [H,T,R] = lab4houghlines(I)

[row,col,ch] = size(I);

img = I;

if (ch==3)

img = rgb2gray(img);

end

% Edge Detection: the selection of the edge detector is very significant!

% Hint use edge built-in function

img\_edges = edge(img,'Canny');

% Hough Transform + Display

% [H,T,R] = ...

[H,T,R] = hough(img\_edges,'RhoResolution',0.5,'Theta',-90:0.5:89.5);

figure('Name','Hough Transform','NumberTitle','off');

subplot(2,2,1)

imshow(I);

title('Original Image');

subplot(2,2,2)

imshow(img\_edges);

title('Edges using canny Edge detector');

subplot(2,2,3)

imshow(H,[],'XData',T,'YData',R,'InitialMagnification','fit');

title('Hough Transform');

xlabel('\theta'), ylabel('\rho');

axis on, axis normal, hold on;

% Hough Peaks Points using Threshold + Display

Thresh = ceil(0.5\*max(H(:)));

P = houghpeaks(H,20,'threshold',Thresh);

x = T(P(:,2)); y = R(P(:,1));

plot(x,y,'s','color','white');

% Hough Lines + Display

lines = houghlines(img\_edges,T,R,P,'FillGap',20,'MinLength',70);

% Plotting All Lines and Highlighting Longest and Shortest Lines

subplot(2,2,4);

imshow(img),title('Hough Lines Based on canny Edge detector, Threshold = '), hold on

max\_len = 40;

min\_len = 2000;

for k = 1:length(lines)

xy = [lines(k).point1; lines(k).point2];

plot(xy(:,1),xy(:,2),'LineWidth',1,'Color','green');

% Plot beginnings and ends of lines

plot(xy(1,1),xy(1,2),'x','LineWidth',1,'Color','yellow','Markersize',3);

plot(xy(2,1),xy(2,2),'x','LineWidth',1,'Color','red','Markersize',3);

% Determine the endpoints of the longest 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;

xy\_short = xy;

end

end

plot(xy\_long(:,1),xy\_long(:,2),'LineWidth',1,'Color','cyan');

plot(xy\_short(:,1),xy\_short(:,2),'LineWidth',1,'Color','red');

end

**d. lab4houghcircles.m**

function [centersBright, radiiBright,centersDark, radiiDark] = lab4houghcircles(I)

[row,col,ch] = size(I);

img = I;

if (ch==3)

img = rgb2gray(img);

end

Rmin = 20;Rmax = 60;Sens = 0.9;

[centersBright, radiiBright] = imfindcircles(I,[Rmin Rmax],'ObjectPolarity','bright','Sensitivity',Sens);

[centersDark, radiiDark] = imfindcircles(I,[Rmin Rmax],'ObjectPolarity','dark','Sensitivity',Sens);

imshow(I),title(sprintf('Detected Circles using Hough Transform \n %d =< Radius =< %d & Sensitivity = %.2d', Rmin, Rmax, Sens)), hold on

viscircles(centersBright, radiiBright,'Color','b');

viscircles(centersDark, radiiDark,'LineStyle','--');

end