EE 417 Computer Vision – Post Lab #5

Line and Circle Detection Using Hough Transform

Osman Burak Çevik

osmanburak@sabanciuniv.edu

26399

Introduction

Lab 5 aims to detect lines and circles by using Hough transform with different parameters. Time performance is also investigated.

Line Detection

The function first reads an image and converts it to a black- white edge image with Canny operator. Then ‘hough’ function is used to obtain Hough transform of the black-white image. Hough points with thedeclared threshold are selected by ‘houghpeaks’ function. Peak points and ‘houghlines’ function are used to find the lines in the image. Founded lines are plotted. Green colored lines are all the colores except longest and shortest lines. Longest line is colored with cyan and shortest line is colored with red.

Result for checker.jpg, threshold=5 for houghpeaks function:

Graphical user interface, diagram

Description automatically generated

Elapsed time: 2.985570 seconds.

Result for checker.jpg, threshold=1 for houghpeaks function:

A picture containing text, screenshot, indoor

Description automatically generated

Elapsed time: 3.009663 seconds.

Result for checker.jpg, threshold=10 for houghpeaks function:

Graphical user interface, diagram

Description automatically generated

Elapsed time: 1.459686 seconds.

Size of checker.jpg: 880x481

Result for ex1.jpeg(276x183), threshold=1 for houghpeaks function:

Graphical user interface, application

Description automatically generated

Elapsed time: 1.245038 seconds.

Result for ex1.jpeg, threshold=5 for houghpeaks function:

Graphical user interface, application

Description automatically generated

Elapsed time: 2.916529 seconds.

Result for ex1.jpeg, threshold=10 for houghpeaks function:

Graphical user interface, application

Description automatically generated

Elapsed time: 2.553860 seconds.

Result for ex1.jpeg, threshold=200 for houghpeaks function:

Graphical user interface, application

Description automatically generated

Elapsed time: 2.518686 seconds.

Circle Detection

The function first reads an image that contains several different size circles and convert it to a black-white image. Then detects all the circles with radius r such that 20<=r<=60 pixels by using Hough transform to implement it ‘imfindcircles’ function is used. ‘Sensitivity’ factor is changed to test the performance of circle detection. ‘ObjectPolarity’ parameter is changed to detect bright and dark cirles seperately.

Result for coin.png(860x672), objectPolarity=bright and sensitivity=0.9 for imfindcircles function:

A group of coins

Description automatically generated with medium confidence

Elapsed time: 0.813499 seconds.

Result for coin.png, objectPolarity=bright and sensitivity=0.5 for imfindcircles function:

A pile of coins

Description automatically generated with medium confidence

Elapsed time: 0.877797 seconds.

Result for coin.png, objectPolarity=bright and sensitivity=0.05 for imfindcircles function:

A group of coins

Description automatically generated with medium confidence

Elapsed time: 0.664899 seconds.

Result for coin.png, objectPolarity=dark and sensitivity=0.9 for imfindcircles function:

A group of coins

Description automatically generated with low confidence

Elapsed time: 0.639085 seconds.

Result for coin.png, objectPolarity=dark&bright and sensitivity=0.9 for imfindcircles function:

A group of coins

Description automatically generated with medium confidence

Elapsed time: 1.247980 seconds.

Discussion

Line Detection: I’ve implemented the function with different sized images and the function

works properly for all.

Increasing threshold detects more lines. However there are non-line structures detected as well. With increasing threshold values number of the non-line structures increases.

Non-linear structures are detcted due to noise. Images contain noise. LoG can be used to reduce the number of non-line structures.

I’ve couldn’t find what the time performance is related. The time elapsed is different for the same paramteres if utilized twice etc.

Circle Detection: I’ve implemented the function with different sized images and the function

works properly for all.

The sensitivity can have values between 0 and 1. Increasing the sensitivity

helps detecting close circles.

I’ve couldn’t find what the time performance is related. The time elapsed is different for the same paramteres if utilized twice etc.

APPENDIX

close all; clear all; clc;

tic

I = imread('ex1.jpeg');

imshow(I)

I=rgb2gray(I);

BW = edge(I,'canny');

imshow(BW);

[H,theta,rho] = hough(BW);

figure

imshow(imadjust(rescale(H)),[],...

'XData',theta,...

'YData',rho,...

'InitialMagnification','fit');

xlabel('\theta (degrees)')

ylabel('\rho')

axis on

axis normal

hold on

colormap(gca,hot)

P = houghpeaks(H,200,'threshold',ceil(0.3\*max(H(:))));

x = theta(P(:,2));

y = rho(P(:,1));

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

lines = houghlines(BW,theta,rho,P,'FillGap',5,'MinLength',7);

figure, imshow(I), hold on

max\_len = 0;

for k = 1:length(lines)

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

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

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

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

len = norm(lines(k).point1 - lines(k).point2);

if ( len > max\_len)

max\_len = len;

xy\_long = xy;

end

end

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

toc

HOUGH LINES

tic

A = imread('coin.png');

A=rgb2gray(A);

imshow(A)

Rmin = 20;

Rmax = 60;

[centersBright, radiiBright] = imfindcircles(A,[Rmin Rmax],'ObjectPolarity','dark','Sensitivity',0.9);

[centersDark, radiiDark] = imfindcircles(A,[Rmin Rmax],'ObjectPolarity','bright','Sensitivity',0.9);

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

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

toc

HOUGH CIRCLES