Lab Instructions - session 7

Hough Transforms

Part 1. Hough Line Transform

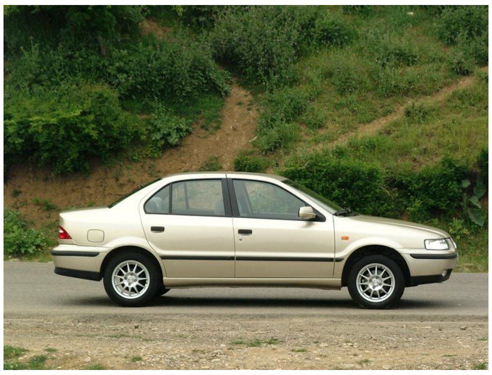
Detect lines in an image using the function cv2.HoughLines
File: **hough_line.py**

```
import numpy as np
import cv2

def draw_line(I,rho,theta):
    "draws a line in image 'I' given 'rho' and 'theta'"
    a = np.cos(theta)
    b = np.sin(theta)
    x0 = a*rho
    y0 = b*rho
    x1 = int(x0 + 1000*(-b))
    y1 = int(y0 + 1000*(a))
    x2 = int(x0 - 1000*(-b))
    y2 = int(y0 - 1000*(a))
    cv2.line(I,(x1,y1),(x2,y2),(0,0,255),1)

I = cv2.imread('highway.jpg')
G = cv2.cvtColor(I,cv2.COLOR_BGR2GRAY) # -> grayscale
E = cv2.Canny(G,100,200) # find the edges

min_votes = 160 # mininum votes to be considered a line
distance_resolution = 1 # 1 pixel: resolution of the parameter "rho"
angle_resolution = np.pi/180 # pi/180 radians: resolution (bin size)
of the parameter "theta"
L = cv2.HoughLines(E,distance_resolution,angle_resolution,min_votes)

# draw the lines
for [[rho,theta]] in L:
    draw_line(I,rho,theta)

cv2.imshow("E",E)
cv2.imshow("I",I)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- What happens by increasing or decreasing the parameter `min_votes`? Why?
- What is the effect of increasing and decreasing the `distance_resolution` and `angle_resolution` parameters? Explain.

Part 2: Hough Circle Transform

The goal is to detect the wheels of the car in the picture using `cv2.HoughCircles`



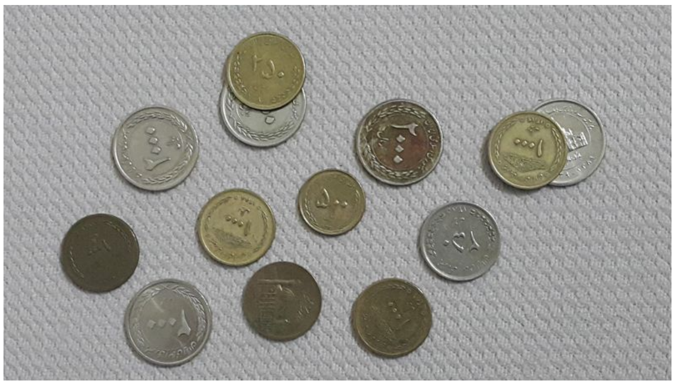
File: **hough_circle.py**

```
import numpy as np
import cv2

I = cv2.imread('samand.jpg')

G = cv2.cvtColor(I,cv2.COLOR_BGR2GRAY) # -> Grayscale
G = cv2.GaussianBlur(G, (3,3), 0);     # Gaussian blur

canny_high_threshold = 200
min_votes = 100 # minimum no. of votes to be considered as a circle
min_centre_distance = 40 # minimum distance between the centres of detected circles
resolution = 1 # resolution of parameters (centre, radius) relative to image resolution
circles = cv2.HoughCircles(G,cv2.HOUGH_GRADIENT,
                           resolution,min_centre_distance,
                           param1=canny_high_threshold,
                           param2=min_votes,minRadius=0,maxRadius=100)

for c in circles[0,:]:
    x = c[0] # x coordinate of the centre
    y = c[1] # y coordinate of the centre
    r = c[2]  # radius

    # draw the circle
    cv2.circle(I,(x,y), r, (0,255,0),2)

    # draw the circle center
    cv2.circle(I,(x,y),2,(0,0,255),2)

cv2.imshow("I",I)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- Change the parameters of `cv2.HoughCircles` and see how each of them affect detection.

Today’s task: count the coins

You need to count the number of coins in the next image:



Write a piece of code to perform this task using hough circle transform. Change the file **task1.py** to perform the task. Play with the parameters until you get the desired results.

File: **task1.py**

```
import numpy as np
import cv2

I = cv2.imread('coins.jpg')
G = cv2.cvtColor(I,cv2.COLOR_BGR2GRAY)
G = cv2.GaussianBlur(G, (5,5), 0);

canny_high_threshold = 160
min_votes = 30 # minimum no. of votes to be considered as a circle
min_centre_distance = 40

circles = np.array([[10,10]])

for c in circles[0,:]:
    x = 100
    y = 100
    r = 40
    cv2.circle(I,(x,y), r, (0,255,0),2)
print circles.shape
n = 100
font = cv2.FONT_HERSHEY_SIMPLEX
cv2.putText(I,'There are %d coins!'%n,(400,40), font, 1,(255,0,0),2)

cv2.imshow("I",I)
cv2.waitKey(0)
```

- What happens by changing different parameters?
- The Hough transform can even detect the partially occluded coins. Why this is the case?

References

- OpenCV-Python Tutorials - Hough Line Transform
- OpenCV-Python Tutorials - Hough Circle Transform