**--Lab-7--To perform Hough transform on an image—**

**Code-1) # Detecting Lines with Hough Transform**

import cv2

import numpy as np

# Load image

img = cv2.imread("OCR.jpg")

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

# Edge detection

edges = cv2.Canny(gray, 50, 150, apertureSize=3)

# Hough Transform for lines

lines = cv2.HoughLines(edges, 1, np.pi/180, 100)

# Draw lines

if lines is not None:

for rho, theta in lines[:,0]:

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(img, (x1,y1), (x2,y2), (0,0,255), 2)

cv2.imshow("Hough Lines", img)

cv2.waitKey(0)

cv2.destroyAllWindows()

**Code-2) Detecting Circles with Hough Transform**

import cv2

import numpy as np

# Load image

img = cv2.imread("OCR.jpg")

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

# Blur to reduce noise

gray = cv2.medianBlur(gray, 5)

# Hough Transform for circles

circles = cv2.HoughCircles(

gray, cv2.HOUGH\_GRADIENT, dp=1.2, minDist=30,

param1=50, param2=30, minRadius=10, maxRadius=100

)

# Draw circles

if circles is not None:

circles = np.uint16(np.around(circles))

for x, y, r in circles[0, :]:

cv2.circle(img, (x, y), r, (0, 255, 0), 2)

cv2.circle(img, (x, y), 2, (0, 0, 255), 3)

cv2.imshow("Hough Circles", img)

cv2.waitKey(0)

cv2.destroyAllWindows()