**EXPERIMENT-**

**AIM:** Write a program for line finding.

**SOFTWARE:** Python(spyder)

**THEORY:**

The Hough transform is a technique which can be used to isolate features of a particular shape within an image. Because it requires that the desired features be specified in some parametric form, the classical Hough transform is most commonly used for the detection of regular curves such as lines, circles, ellipses, etc. A generalized Hough transform can be employed in applications where a simple analytic description of a feature(s) is not possible. Due to the computational complexity of the generalized Hough algorithm, we restrict the main focus of this discussion to the classical Hough transform. Despite its domain restrictions, the classical Hough transform (hereafter referred to without the classical prefix) retains many applications, as most manufactured parts (and many anatomical parts investigated in medical imagery) contain feature boundaries which can be described by regular curves. The main advantage of the Hough transform technique is that it is tolerant of gaps in feature boundary descriptions and is relatively unaffected by image noise. The Hough technique is particularly useful for computing a global description of a feature (where the number of solution classes need not be known a priori), given local measurements. The motivating idea behind the Hough technique for line detection is that each input measurement) indicates its contribution to a globally consistent

**PROGRAM:**

import cv2

import numpy as np

import math

image1 = cv2.imread(r"C:\Users\HP\Pictures\Picture7.jpg")

gray = cv2.cvtColor(image1, cv2.COLOR\_BGR2GRAY)

# Apply Canny edge detection

canimg = cv2.Canny(gray, 50, 200)

# Use Hough Line Transform to find lines in the image

lines = cv2.HoughLines(canimg, 1, np.pi/180, 100, np.array([]))

if lines is not None:

for line in lines:

rho, theta = line[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(image1, (x1, y1), (x2, y2), (0, 0, 255), 2)

cv2.imshow('Lines Detected', image1)

cv2.imshow("Canny Detection", canimg)

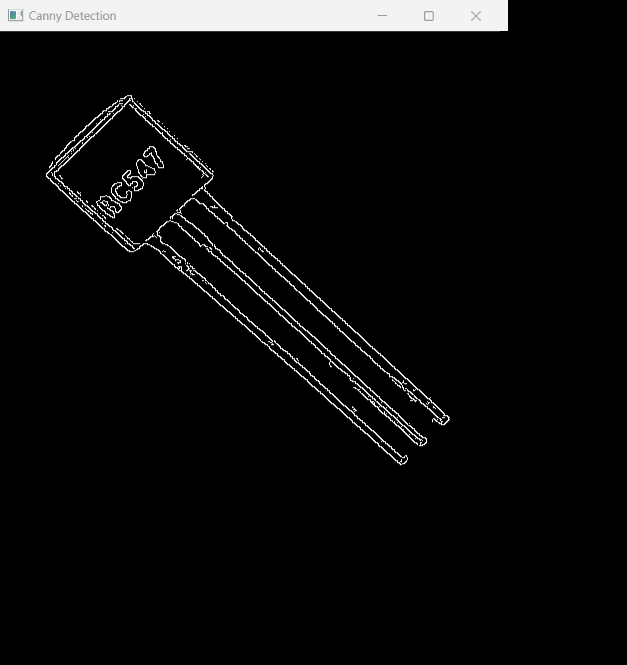
cv2.waitKey(0)

cv2.destroyAllWindows()

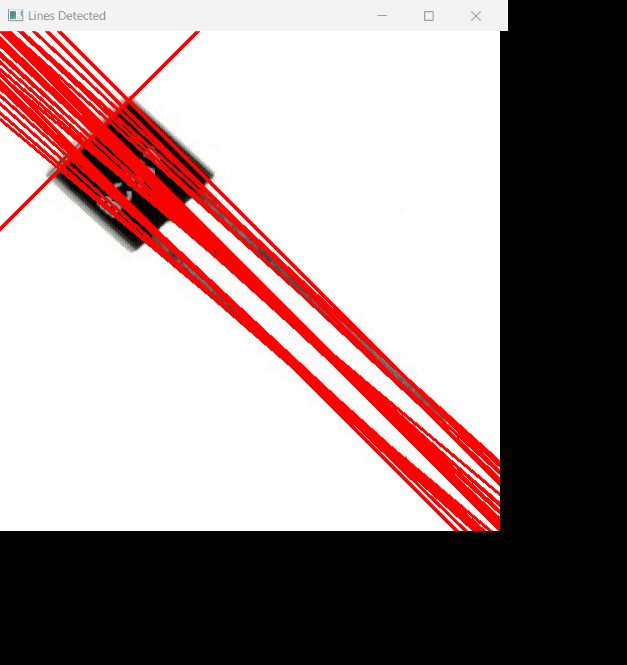
else:

print("No lines detected in the image.")

**output:**

****

**Line detection**

****

**CONCLUSION:** In this practical we find the lines. By using Hough Transform.