NAMES AND ROLL #:

WAJAHAT ALI WASSAN ------ 2K18/IT/129

GAJAN KHAN URF M-AYAZ ------ 2K18/IT/43

SUBJECT: COMPUTER VISION AND IMAGE PROCESSING

ASSIGNMENT TO:

SIR Dr.SANDER ALI KHOWAJA

GITHUB LINK : SIR IT IS RAR FILE UPLOADED , IF YOU CLICK ON FILE IT WILL BE DOWNLOADED TO YOUR COMPUTER.

https://github.com/ayazgajan/cvassignment.git

Problem:

- Find edges of image.
- Find HT line in image.
- Save images in disk.

Solution:

- I have used canny edge detector to detect edges in picture.
- I have used HoughLines function of CV2 library to get HT lines in image.
- I have images in disk using imsave function of skimage library.

Modification:

- This code is collected from slides & combined multiple codes to complete this project.
- I have changed the threshold 1 value & threshold 2 value to get, true edges of picture.
- Then I have changed the Iteration value to 200 to 80 because my image was low resolution.
- I have saved the results in disk to get understanding of what program is doing without executing it.

Algorithm:

- Import required packages.
- Read image & change its color scale to grey.
- Get edges of image using canny edge detector.
- Get HT lines of image by Iterating over whole picture by a loop.
- Show results in individual windows.
- Save results in disk.

Conclusion:

- I have got true edges of picture.
- I have got HT lines of picture.
- I have saved the images & results in disk.

Code:

```
import cv2

import numpy as np

from skimage import io


# handling images

myImage = cv2.imread('img.png')

grayImage = cv2.cvtColor(myImage, cv2.COLOR_RGB2GRAY)
```

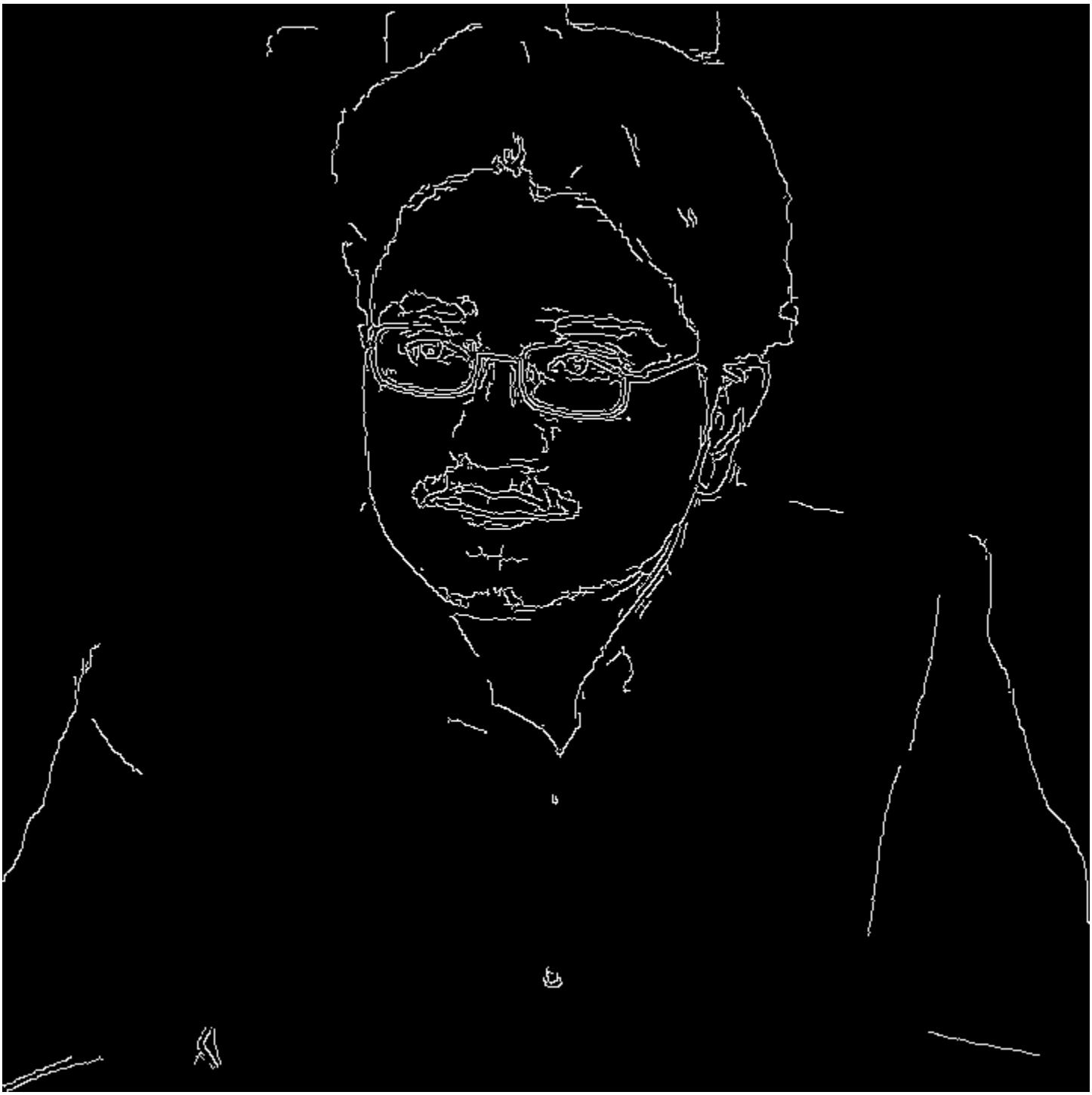
```
# find edges using canny edge detector
cannyEdges = cv2.Canny(grayImage, 50, 100)
# find lines in image
lines = cv2.HoughLines(cannyEdges, 1, np.pi / 180, 80)
for line in lines:
    rho, theta = line[0]
    a = np.cos(theta)
    b = np.sin(theta)
    x0 = a * rho
    y0 = b * rho

    x1 = int(x0 + 1000 * (-b))
    y1 = int(y0 + 1000 * (a))
    x2 = int(x0 - 1000 * (-b))
    y2 = int(y0 - 1000 * (a))
    cv2.line(myImage, (x1, y1), (x2, y2), (0, 0, 255,), 2)

# show result
cv2.imshow('Canny Edges', cannyEdges)
cv2.imshow('HT lines', myImage)

# save result in disk for better understandability with out running code
io.imsave('result/Canny edges.png', cannyEdges)
io.imsave('result/HT lines.png', myImage)

# waiting for key
k = cv2.waitKey(0)
if cv2.waitKey(0) & 0xff == 27:
    cv2.destroyAllWindows()
```



Results: