МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Систем автоматического управления

ОТЧЕТ

по лабораторной работе №7
по дисциплине «Техническое зрение»
Тема: «Преобразование Хафа для поиска прямых и окружностей»

Студент гр. 1492 Старцев Н.А.

Преподаватель Федоркова А.О.

Санкт-Петербург
2024

Цель работы: изучить методы поиска прямых и окружностей с помощью преобразования Хафа

Задание 1:

Для выполнения этой части задания используйте рисунок 7_1.

Исправьте это изображение так, чтобы линии таблицы исчезли, а числа остались.

Задание 2:

Для выполнения этого задания используйте обработанное операцией размыкания изображение из работы 4. На нём найдите самый длинный отрезок и самую большую окружность.

Задание 3:

Для выполнения этого задания возьмите видео из задания 3 к лабораторной 5. Ваша задача - найти линию на этом видео с помощью преобразования Хафа. Можно найти несколько линий, которые проходят по границам полосы разметки.

Задание 4:

Для выполнения этого задания используйте картинки 7_5_1-7_5_6.

Найдите любые три круглых дорожных знака.

.

Примеры работы программы:
Задание 1

		5	3					
8							2	
	7			1		5		
4					5	3		
	1			7				6
		3	2				8	
	6		5					9
		4					3	
					9	7		

Рисунок 1 – удаление линий с рисунка

Задание 2.

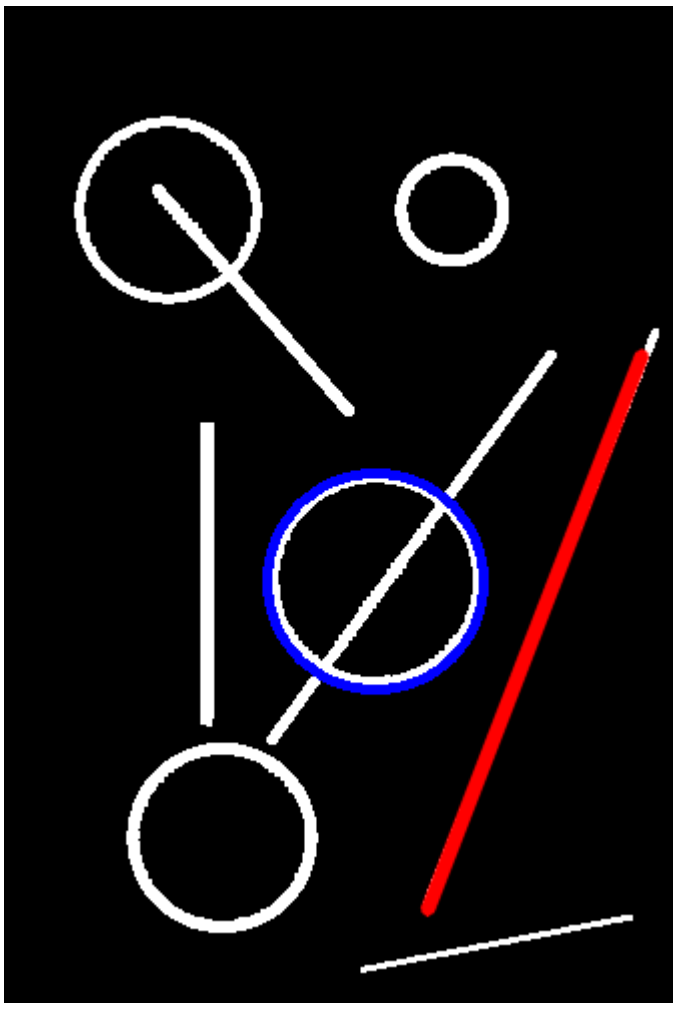

Рисунок 2 – определение самой большой окружности и самой длинной линии

Задание 3

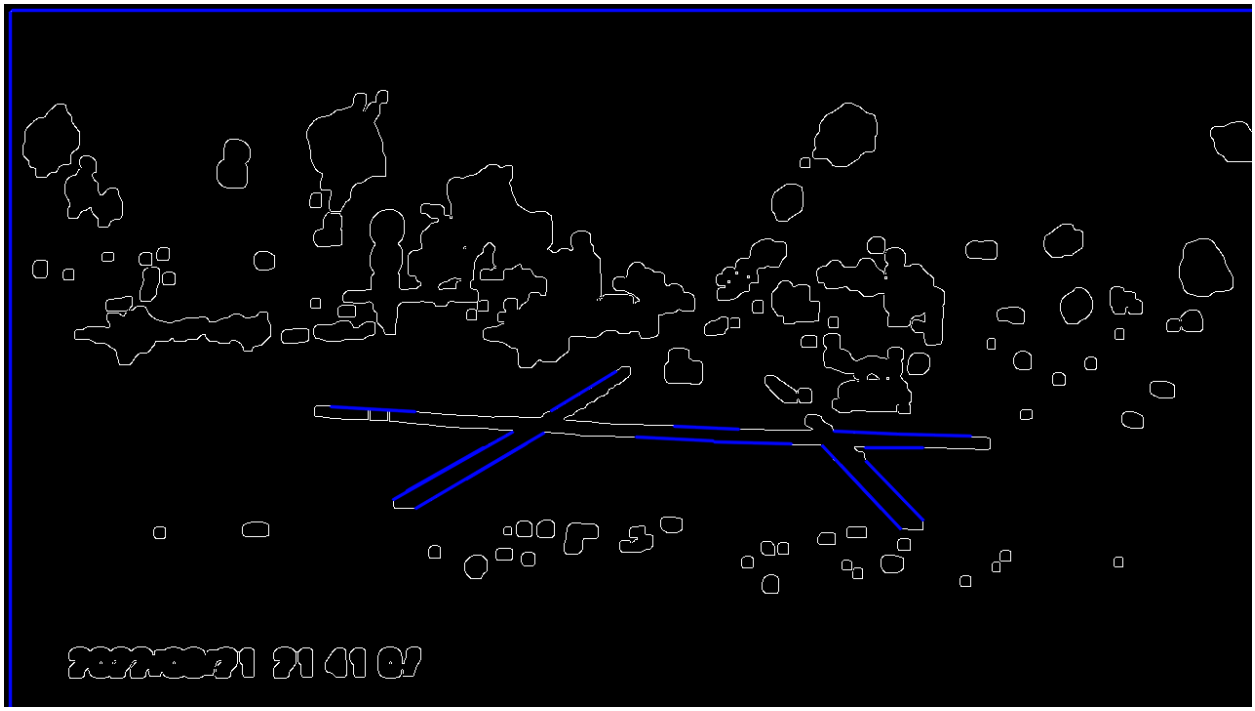

Рисунок 3 – результат работы программы, нахождение линий разметки

Задание 4

Рисунок 4 – Определение круглых дорожных знаков.

Вывод.
В ходе лабораторной работы были изучены методы поиска линий, отрезков и окружностей

Результаты работы

Код программы:

Задание 1

```
import cv2
import numpy
import math

def nothing(x):
    pass


pathImg = "./lab7/7_1.jpg"
winName = "Test Window"

cv2.namedWindow(winName, cv2.WINDOW_AUTOSIZE)
img = cv2.imread(pathImg, flags=cv2.IMREAD_GRAYSCALE)
img_c = cv2.imread(pathImg, flags=cv2.IMREAD_COLOR)
height, weight = img.shape[0:2]
img = cv2.resize(img, (int(height*0.5), int(weight*0.5)))
img_c = cv2.resize(img_c, (int(height*0.5), int(weight*0.5)))


grad_x = cv2.Sobel(img,  ddepth=0, dx=1, dy=0, ksize=3, scale=1, delta=0)
grad_y = cv2.Sobel(img,  ddepth=0, dx=0, dy=1, ksize=3, scale=1, delta=0)
new_img = grad_x+grad_y


new_img = cv2.morphologyEx(new_img, cv2.MORPH_CLOSE, kernel=numpy.ones((1, 1), dtype=int),
                           anchor=(-1, -1),
                           iterations=1,
                           borderType=cv2.BORDER_CONSTANT, borderValue=(
                               255, 255, 255),
                           )

cv2.createTrackbar("rho_res", winName, 7, 20, nothing)
cv2.createTrackbar("theta_res", winName, 0, 358, nothing)
cv2.createTrackbar("threshold", winName, 6, 20, nothing)
while (1):
    rho_res = 0.1+cv2.getTrackbarPos("rho_res", winName)/10
    theta_res = math.pi/(2+cv2.getTrackbarPos("theta_res", winName))
    threshold = (1+cv2.getTrackbarPos("threshold", winName))*50

    lines = cv2.HoughLines(
        new_img,  # входное изображение
        rho_res,  # разрешение по расстоянию
        theta_res,  # разрешение по углу
        threshold,  # значение аккумулятора
        min_theta=None,  # ограничение угла
        max_theta=None  # ограничение угла
    )
    img_c_n = img_c.copy()
    for line in lines:
        rho = line[0][0]
        theta = line[0][1]
        cos_t = math.cos(theta)
        sin_t = math.sin(theta)
        x0 = cos_t * rho
        y0 = sin_t * rho
        pt1 = int(x0 - 1000*sin_t), int(y0 - 1000*cos_t)
        pt2 = int(x0 + 1000*sin_t), int(y0 + 1000*cos_t)
        cv2.line(img_c_n, pt1, pt2, (255,255,255), 10, cv2.LINE_AA)


    cv2.imshow(winName, img_c_n)
    key = cv2.waitKey(100)
    if key == 27:break
```

Задание 2

```
import cv2
import numpy
import math

def nothing(x):
    pass

pathImg1 = "./lab4/4-1.jpg"
winName = "test_window"

cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

img = cv2.imread(pathImg1, flags=cv2.IMREAD_GRAYSCALE)

cv2.createTrackbar("size", winName, 1, 10, nothing)
cv2.createTrackbar("iterations", winName, 1, 10, nothing)
cv2.createTrackbar("type", winName, 1, 4, nothing)

morph_type = [cv2.MORPH_OPEN, cv2.MORPH_CLOSE,
              cv2.MORPH_GRADIENT, cv2.MORPH_TOPHAT, cv2.MORPH_BLACKHAT]
while 1:
    # ker = numpy.full((7,7), 1)
    size = cv2.getTrackbarPos("size", winName)+1
    iter = cv2.getTrackbarPos("iterations", winName)+1
    t = cv2.getTrackbarPos("type", winName)

    img_new = cv2.morphologyEx(img, morph_type[t], kernel=numpy.ones((size, size), dtype=int),
                               anchor=(-1, -1),
                               iterations=iter,
                               borderType=cv2.BORDER_CONSTANT, borderValue=(
                                   255, 255, 255),
                               )
    break
    if key == 27:
        break

cv2.destroyWindow(winName)
cv2.namedWindow(winName)
img = img_new.copy()
_, new_img = cv2.threshold(img, 10, 255, cv2.THRESH_BINARY_INV)

lines = cv2.HoughLinesP(new_img, rho=1, theta=math.pi /180,
                            threshold=50, minLineLength=50, maxLineGap=3)
img2=cv2.cvtColor(new_img, cv2.COLOR_GRAY2BGR)
# for line in lines:
#     cv2.line(img2, (line[0][0],line[0][1]), (line[0][2],line[0][3]), (255,5,2), 2, cv2.LINE_AA)
# cv2.imshow(winName, img2)
# key = cv2.waitKey(10000)

circles = cv2.HoughCircles(new_img, method=cv2.HOUGH_GRADIENT, dp=2,
                            minDist=20, param1=80, param2=100, minRadius=0, maxRadius=0)
max_R = circles[0][0][2]
max_circle = circles[0][0]
for c in circles[0]:
    if c[2]>max_R:
      max_R = c[2]
      max_circle = c
x,y,R = int(max_circle[0]),int(max_circle[1]),int(max_circle[2])
cv2.circle(img2, center = (x, y), radius = R, color = (255, 0, 0), thickness = 3, lineType = cv2.LINE_8)

# img2=cv2.cvtColor(new_img, cv2.COLOR_GRAY2BGR)
max_line = lines[0]
line_len = 0
for line in lines:
    l = ((line[0][0]-line[0][2])**2+(line[0][1]-line[0][3])**2)**0.5
    if line_len <= l:
        max_line = line
        line_len = l
cv2.line(img2, (max_line[0][0],max_line[0][1]), (max_line[0][2],max_line[0][3]), (0,0,255), 5,
cv2.LINE_AA)

cv2.imshow(winName, img2)
key = cv2.waitKey(10000)
```

Задание 3

```
import cv2
import numpy
import math

def nothing(x):
    pass

morph_type = [cv2.MORPH_OPEN, cv2.MORPH_CLOSE,
              cv2.MORPH_GRADIENT, cv2.MORPH_TOPHAT, cv2.MORPH_BLACKHAT]

pathVideo1 = "./lab5/stop_line_1.mp4"
pathVideo2 = "./lab5/stop_line_2.mp4"

cap = cv2.VideoCapture(pathVideo1)

winName = "test_window"
cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)

p = 1
while 1:
    if p == 1:
        ret, frame = cap.read()
    else:
        lines = cv2.HoughLinesP(img_new, rho=1, theta=math.pi /360,
                                threshold=50, minLineLength=50, maxLineGap=6)
        img2=cv2.cvtColor(img_new, cv2.COLOR_GRAY2BGR)

        for line in lines:
            cv2.line(img2, (line[0][0],line[0][1]), (line[0][2],line[0][3]), (255,5,2), 2, cv2.LINE_AA)
        cv2.imshow(winName, img2)
        cv2.waitKey()
        p=1-p
        ret, frame = cap.read()

    threshold1 = 120 #cv2.getTrackbarPos("threshold1", winName)
    threshold2 = 250 #cv2.getTrackbarPos("threshold2", winName)
    size = 4 #cv2.getTrackbarPos("size", winName)
    # if frame is read correctly ret is True
    size_1 = 2 #cv2.getTrackbarPos("size", winName)+1
    iter_1 = 7 #cv2.getTrackbarPos("iterations", winName)+1

    size = 3 #cv2.getTrackbarPos("size", winName)+1
    iter = 0 #cv2.getTrackbarPos("iterations", winName)+1
    t = 0 #cv2.getTrackbarPos("type", winName)

    img_new = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    img_new = cv2.medianBlur(img_new, size*2+1)
    threshold_new, img_new = cv2.threshold(img_new, 120, 255, cv2.THRESH_BINARY)

    img_new = cv2.dilate(
                    img_new,
                    kernel=numpy.ones((size_1, size_1), dtype=int),
                    # kernel = ker,
                    anchor=(-1, -1),
                    iterations=iter_1,
                    borderType=cv2.BORDER_CONSTANT, borderValue=(255, 255, 255))

    img_new = cv2.morphologyEx(img_new, morph_type[t], kernel=numpy.ones((size, size), dtype=int),
                               anchor=(-1, -1),
                               iterations=iter,
                               borderType=cv2.BORDER_CONSTANT, borderValue=(255, 255, 255),
                               )
    img_new = cv2.Canny(img_new, threshold1, threshold2)

    cv2.imshow(winName, img_new)
    key = cv2.waitKey(100)
    if key == 32: p=1-p
    if key == 27:
        break
    if not ret:
        print("Can't receive frame (stream end?). Exiting ...")
        break

cap.release()
cv2.destroyAllWindows()
```

Задание 4

```
import cv2
import numpy
import math

def nothing(x):
    pass

pathImg1 = "./lab7/7_5_1.jpg"
pathImg2 = "./lab7/7_5_2.jpg"
pathImg3 = "./lab7/7_5_3.jpg"
pathImg4 = "./lab7/7_5_4.jpg"
pathImg5 = "./lab7/7_5_5.jpg"
pathImg6 = "./lab7/7_5_6.jpg"
winName = "Test Window"

path = [pathImg1,pathImg2,pathImg3,pathImg4,pathImg5,pathImg6]
images = list()
images_color = list()
for p in path:
    images_color.append(cv2.imread(p, flags=cv2.IMREAD_COLOR))
    images.append(cv2.cvtColor(images_color[-1], cv2.COLOR_BGR2GRAY))
img_filtr = list()
for img in images:
    img_filtr.append(cv2.medianBlur(img, 25))
circleInImg = list()
new_img = list()
for img in img_filtr:
    circleInImg.append(cv2.HoughCircles(img, method = cv2.HOUGH_GRADIENT, dp = 2, minDist = 100, param1 =
275, param2 = 400, minRadius = 0, maxRadius = 0))

for i in range(len(images)):
    new_img.append(cv2.cvtColor(images[i], cv2.COLOR_GRAY2RGB))
    if not (circleInImg[i] is None):
        for j in range(0,circleInImg[i].shape[1]):
            x = int(circleInImg[i][0][j][0])
            y = int(circleInImg[i][0][j][1])
            R = int(circleInImg[i][0][j][2])


            cv2.circle(new_img[i], center = (x, y), radius = R, color = (0, 0, 255), thickness = 5,
lineType = cv2.LINE_8)



k = 0

cv2.namedWindow(winName, cv2.WINDOW_GUI_NORMAL)
while (1):
    cv2.imshow(winName, new_img[k])
    key = cv2.waitKey(100)
    if key == 27:break
    if key == 32: k = (k+1)%6
```