|  |  |
| --- | --- |
| тов_знак_прав | МИНОБРНАУКИ РОССИИ  федеральное государственное автономное образовательное учреждение  высшего образования  «Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)»  **(СПбГЭТУ «ЛЭТИ»)** |

Кафедра САУ

ОТЧЕТ

по лабораторно-практической работе № 6

По курсу: «Техническое зрение»

Тема: Преобразование Хафа.

|  |  |  |
| --- | --- | --- |
| Студенты гр. 6491 |  | Зверев Г. Ю.  Михайленко Д. М. |
| Преподаватель |  | Моклева К. А. |

Санкт-Петербург

2020 г.

Цель работы: изучить принцип применения преобразования Хафа для поиска прямых и окружностей.

Задание 1.

Для выполнения этого задания нужно использовать файл

6\_1.png. На этом изображении выделите цветом самую большую

окружность и самый длинный отрезок.

Задание 2.

Для выполнения этого задания нужно использовать файл 6\_2.png. Исправьте это изображение так, чтобы линии таблицы исчезли, а числа остались. Для решения этой задачи воспользуйтесь преобразованием Хафа для поиска прямых.

Задание 1

*Код программы:*

from cv2 import cv2

import numpy as np

from math import sqrt

def draw\_line\_P(img, x1, y1, x2, y2, color = (0, 0, 255), thickness = 2, lineType=cv2.LINE\_AA):

cv2.line(img, (x1, y1), (x2, y2), color, thickness, lineType)

def draw\_circle(img, center, radius, color=(255, 0, ), thickness = 2, lineType = cv2.LINE\_AA):

cv2.circle(img, center, radius, color, thickness, lineType)

# Получим изображение

img = cv2.imread('6\_1.png')

# Переменные для вывода результата в разных окнах cv2.imshow()

lines\_img = np.copy(img)

longestLine\_img = np.copy(img)

circles\_img = np.copy(img)

biggestCircle\_img = np.copy(img)

# Получим полутоновое изображние

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

invers\_img\_gray = cv2.bitwise\_not(img\_gray)

# При таких значениях аргументов функции на изображении выделяются только линии

lines = cv2.HoughLinesP(invers\_img\_gray, rho = 1, theta = np.pi/720, threshold = 255, maxLineGap = 15)

print("Lines: ", lines)

maxLength = 0

x\_1, y\_1, x\_2, y\_2 = 0, 0, 0, 0

for line in lines:

x1, y1, x2, y2 = line[0]

print("x1 = {x1}, y1 = {y1}, x2 = {x2}, y2 = {y2}".format(x1 = x1, x2 = x2, y1 = y1, y2 = y2))

currentLength = sqrt((x2 - x1)\*\*2 + (y2 - y1)\*\*2)

print('CurrentLenght: ', currentLength, '\n')

if currentLength >= maxLength:

maxLength = currentLength

x\_1, y\_1, x\_2, y\_2 = x1, y1, x2, y2

draw\_line\_P(lines\_img, x1, y1, x2, y2)

draw\_line\_P(longestLine\_img, x\_1, y\_1, x\_2, y\_2)

edges = cv2.Canny(img\_gray, 75, 255)

circles = cv2.HoughCircles(edges, cv2.HOUGH\_GRADIENT, dp = 1, minDist = 280, param1 = 255, param2 = 97, minRadius = 0, maxRadius = 0)

print("Circles: ", circles)

# Координаты и радиус для самой большой окружности

x\_max, y\_max, r\_max = 0, 0, 0

# Выделим все найденные окружности

for circle in circles[0]:

x0, y0, r = circle

print(x0,y0,r)

if r > r\_max:

x\_max, y\_max, r\_max = x0, y0, r

draw\_circle(circles\_img, (x0, y0), r)

# Выделим самую большую окружность

draw\_circle(biggestCircle\_img, (x\_max, y\_max), r\_max)

cv2.imshow('All lines', lines\_img)

cv2.imshow('Longest line', longestLine\_img)

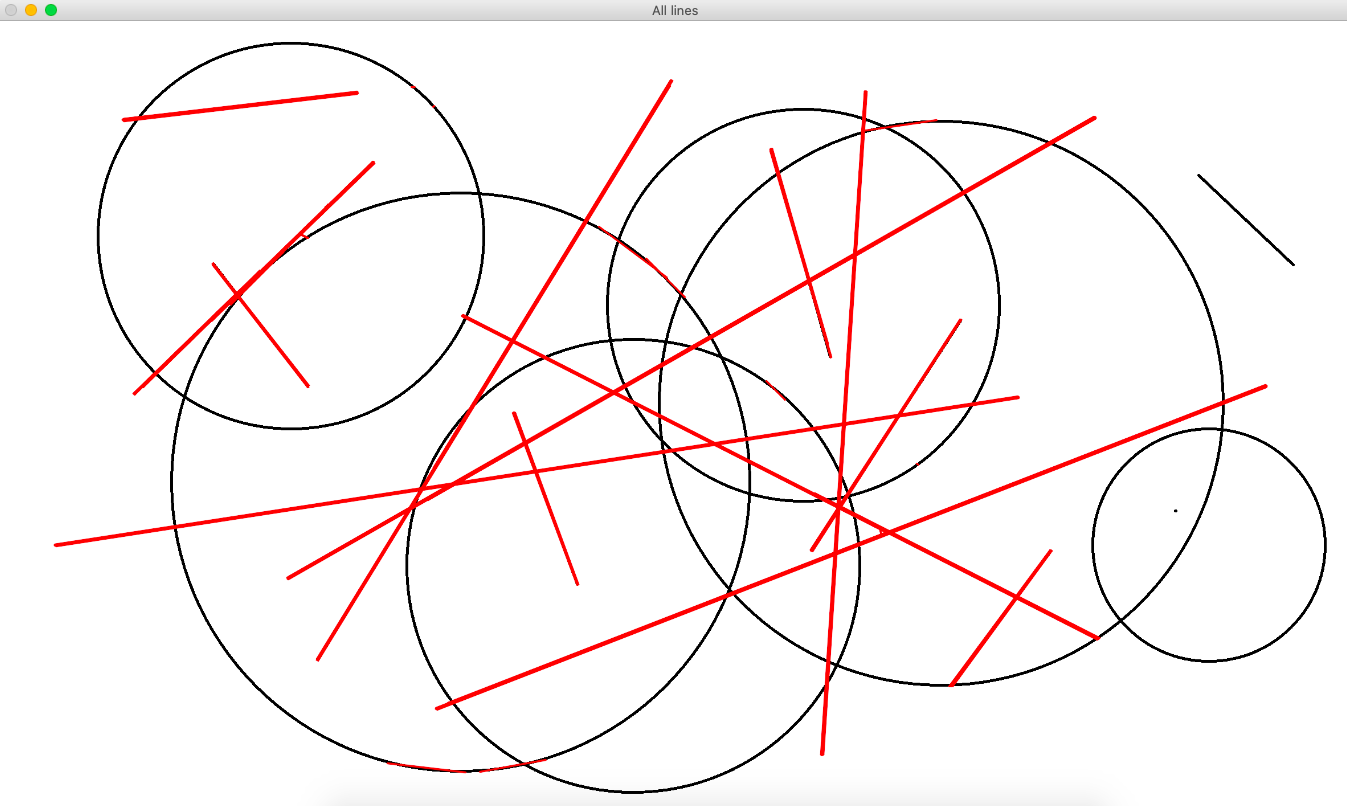
cv2.imshow('Circles', circles\_img)

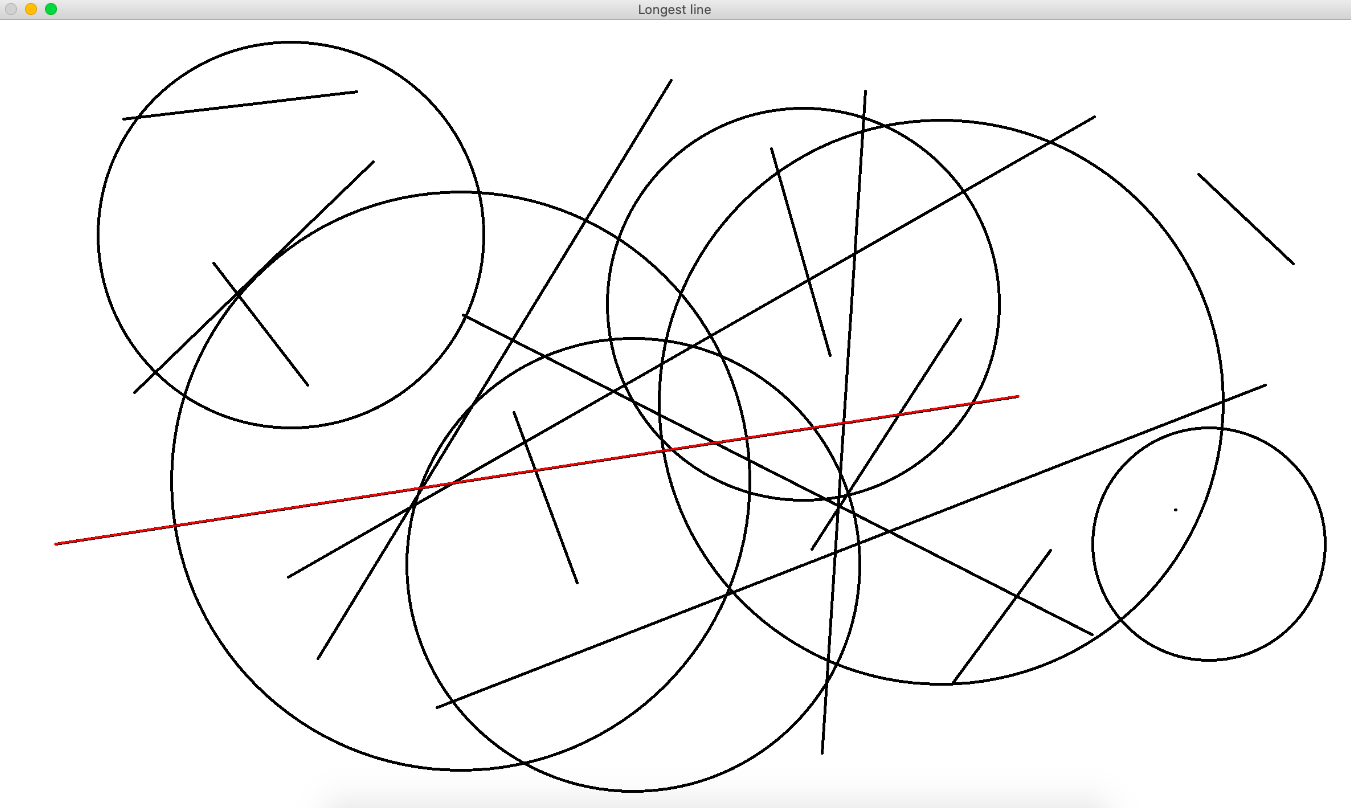
cv2.imshow('Biggest circle', biggestCircle\_img)

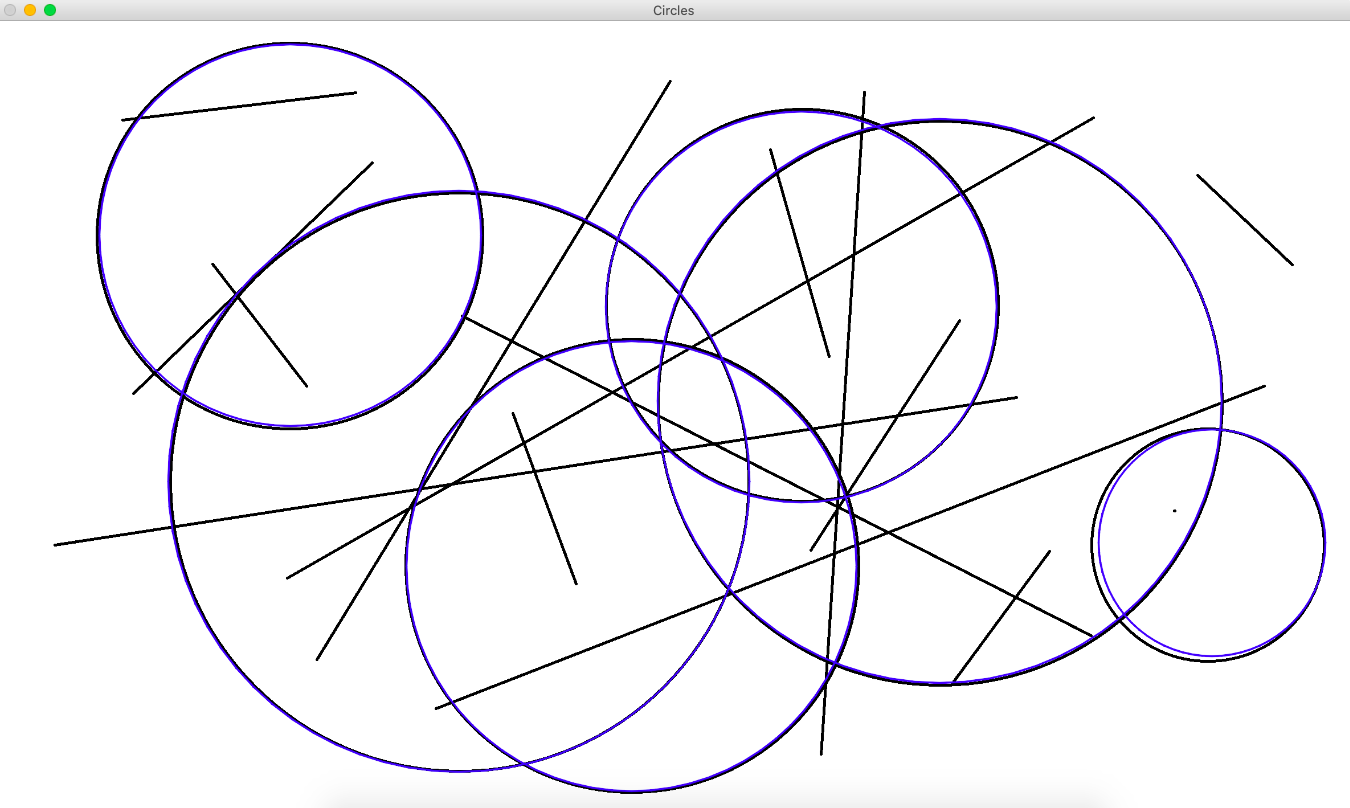
cv2.waitKey(0)

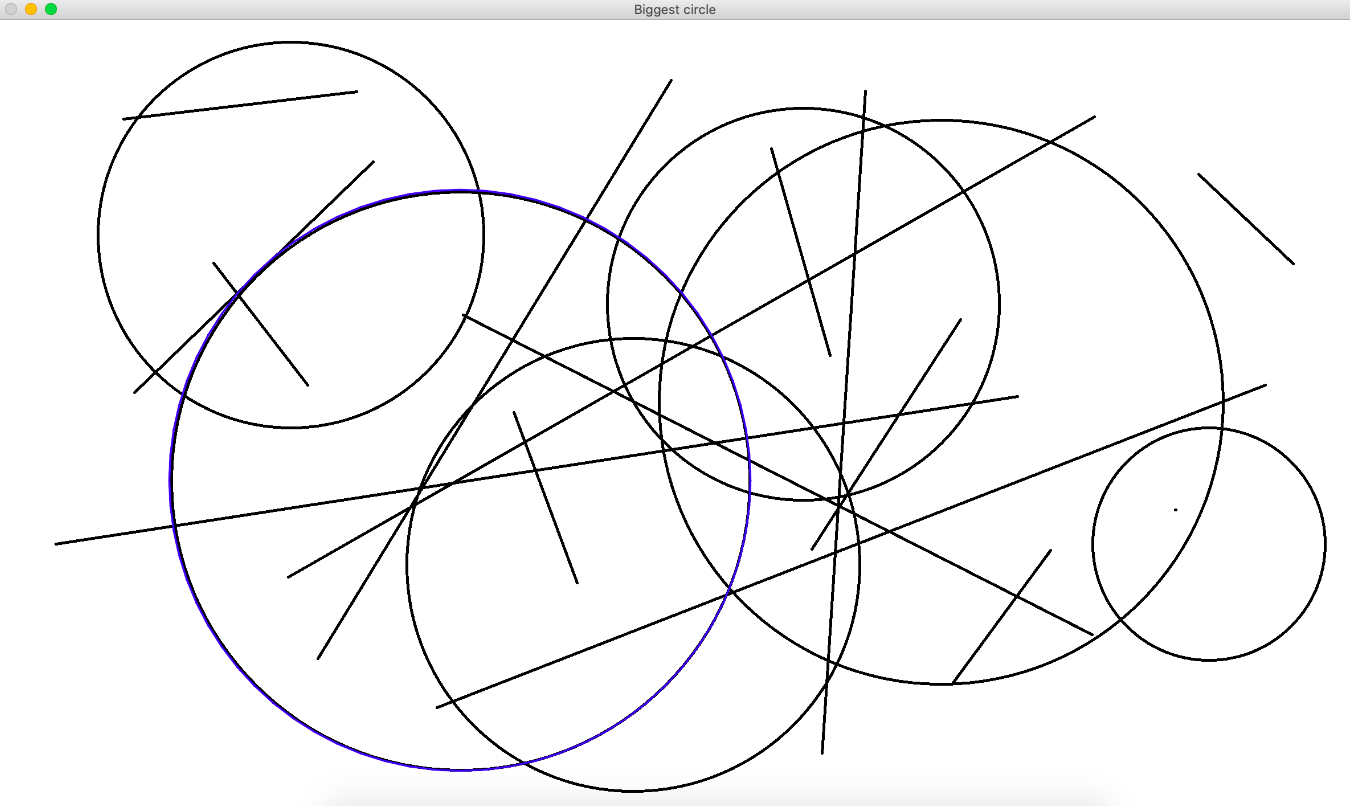
cv2.destroyAllWindows()

*Результат:*









Задание 2

Данное задание было решено 2мя способами. С помощью стандартного преобразования Хафа и прогрессивного вероятностного преобразования Хафа.

*Код программы:*

from cv2 import cv2

from numpy import pi, cos, sin, shape

def draw\_line\_P(img, x1, y1, x2, y2, color = (255, 255, 255), thickness = 2, lineType=cv2.LINE\_AA):

cv2.line(img, (x1, y1), (x2, y2), color, thickness, lineType)

# 1й Способ: HoughLines() - стандартное преобразование Хафа

img = cv2.imread('6\_2.png')

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

invers\_img\_gray = cv2.bitwise\_not(img\_gray)

edges = cv2.Canny(img\_gray, 75, 230)

lines = cv2.HoughLines(image = edges, rho = 1, theta = pi/360, threshold = 150)

for line in lines:

rho, theta = line[0]

a = cos(theta)

b = sin(theta)

print("a, b: ", a,b)

x0 = a \* rho

y0 = b \* rho

print("xo, yo: ", x0, y0)

x1 = int(x0 + 1200 \* (-b))

y1 = int(y0 + 1200 \* (a))

x2 = int(x0 - 1200 \* (-b))

y2 = int(y0 - 1200 \* (a))

cv2.line(img = img, pt1 = (x1, y1), pt2 = (x2, y2), color = (255, 255, 255), thickness = 15)

cv2.imshow('Edges', edges)

cv2.imshow('Image', img)

cv2.waitKey()

cv2.destroyAllWindows()

# 2й способ: HoughLinesP() - прогрессивное вероятностное преобразование Хафа

img\_orig = cv2.imread('6\_2.png')

img\_gray = cv2.cvtColor(img\_orig, cv2.COLOR\_BGR2GRAY)

img\_result = img\_orig.copy()

\_, thresh = cv2.threshold(img\_gray, 125, 255 , cv2.THRESH\_BINARY\_INV)

lines = cv2.HoughLinesP( image = thresh, rho = 1, theta = pi/180, threshold = 255, maxLineGap = 15)

print(shape(lines))

for line in lines:

print(line)

x1 , y1 , x2 , y2 = line[0]

draw\_line\_P(img\_result, x1 , y1 ,x2, y2)

cv2.imshow('Original image', img\_orig)

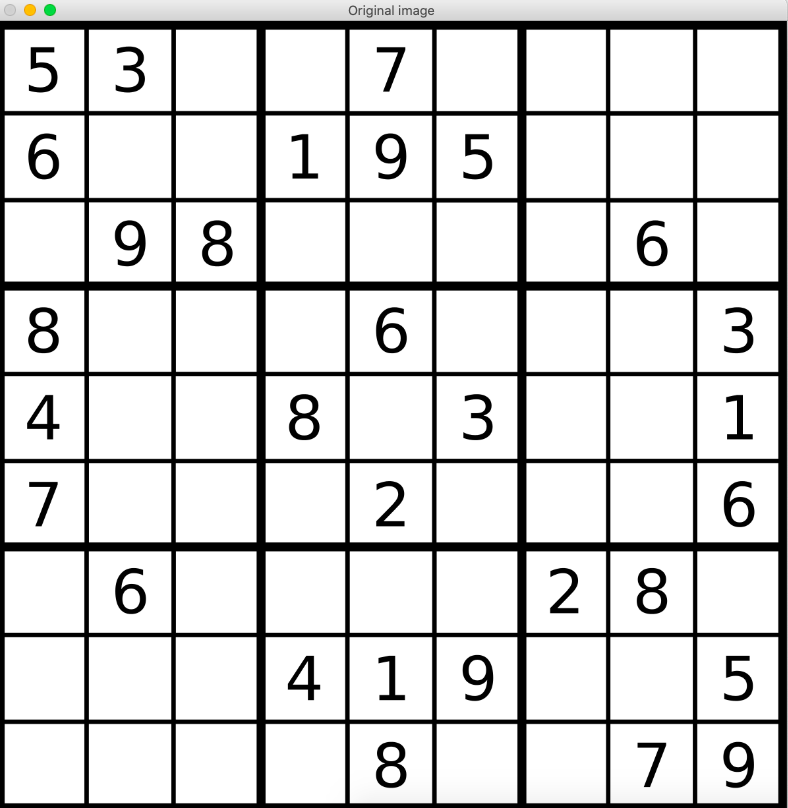
cv2.imshow('Without lines', img\_result)

cv2.imshow('thresh', thresh)

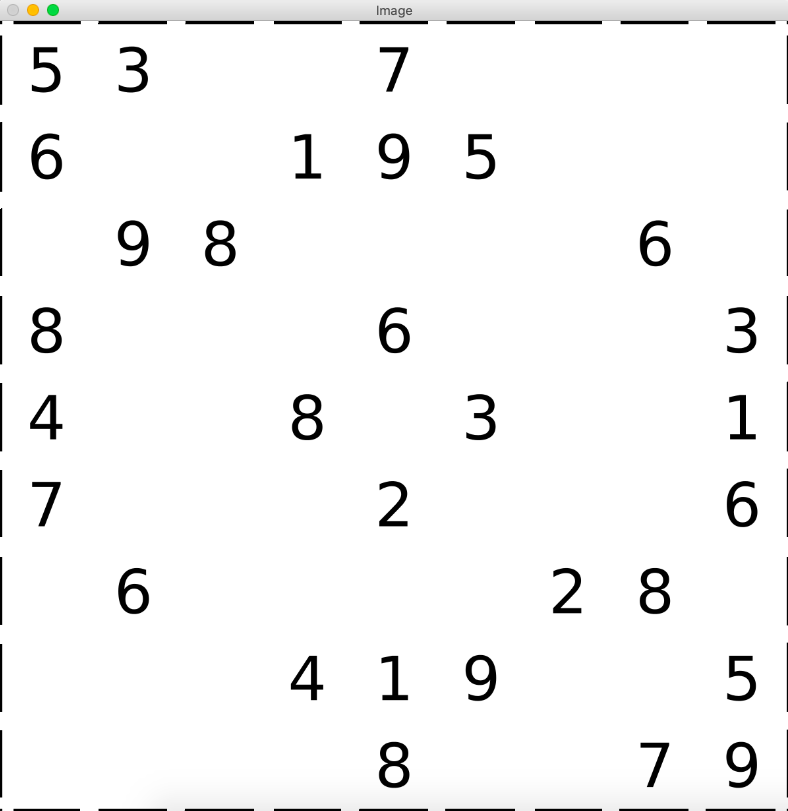
cv2.waitKey(0)

cv2.destroyAllWindows()

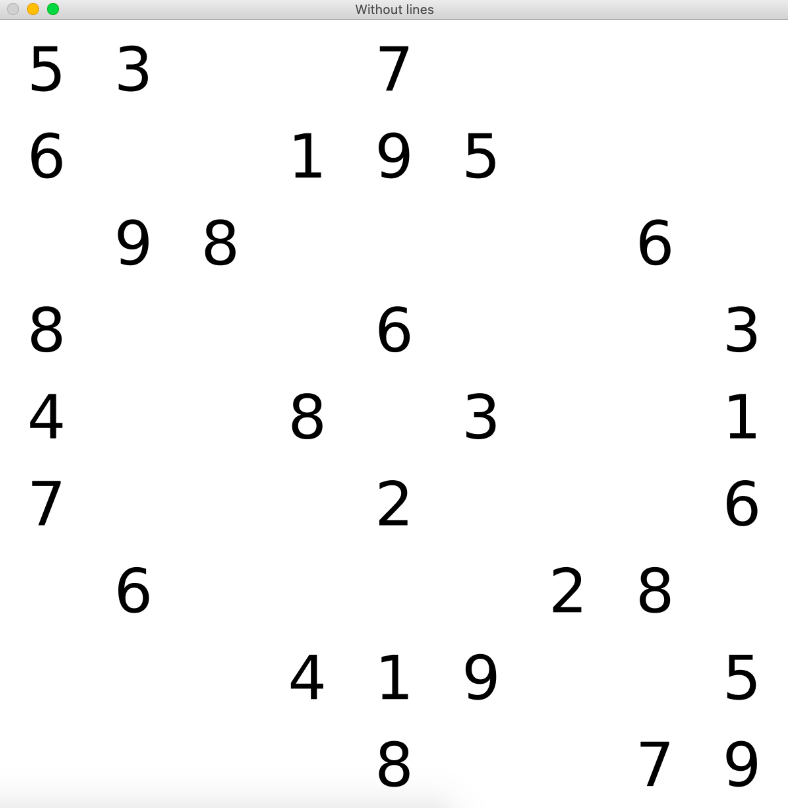
*Результат:*

**

HoughLines():



HoughLinesp():



Вывод: в ходе лабораторной работы были изучены принципы преобразования Хафа для нахождения линий и окружностей на различных изображениях.