**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САУ**

отчет

**по лабораторной работе №5**

**по дисциплине «Техническое зрение»**

Тема: Границы и контуры.

|  |  |  |
| --- | --- | --- |
| Студент гр. 6492 |  | Михайлов Н.К. |
| Преподаватель |  | Моклева К.А. |

Санкт-Петербург

2020

**Цель работы:** изучить способы выделения границ на изображении, поиск контуров на границах и получения информации об объектах на основе контуров.

**Задания:**

1. Исследуйте все известные вам способы поиска границ на изображении. Для этого выберите несколько изображений, содержащих как четко отделимые от фона границы, так и нечеткие границы, почти сливающиеся с фоном. При применении методов аргументируйте выбор значений, передаваемых в качестве параметров методов.
2. Исследуйте работу функции findContours() на двух типах бинарных изображений:

1) бинарные изображения, полученные с помощью функции threshold()

2) бинарные изображения границ, полученные детектором границ Кенни

Как отличается количество контуров? Почему?

Возьмите изображение окружности с толщиной линии в несколько пикселей (вы можете самостоятельно нарисовать его, например, в paint). Вычислите контуры на этом изображении. Найдите один контур, который описывает окружность с внешней стороны линии, и один контур, который описывает окружность с внутренней стороны линии. Для них вычислите длину, площадь. Почему значения отличаются таким образом? Для каждого контура вычислите ограничивающий прямоугольник и ограничивающую окружность. Сравните значения площадей ограничивающих фигур с площадями контуров. Прокомментируйте результат.

**Ход работы.**

**import** cv2

img = cv2.imread('C:/Users/po/Desktop/img\_2.jpg', cv2.IMREAD\_REDUCED\_COLOR\_2)

sobel = cv2.Sobel(img, cv2.CV\_8U, 1, 1, ksize = 3)

sobel\_x = cv2.Sobel(img, cv2.CV\_8U, 1, 0, ksize = 3)

sobel\_y = cv2.Sobel(img, cv2.CV\_8U, 0, 1, ksize = 3)

laplacian = cv2.Laplacian(img, cv2.CV\_8U, ksize = 3)

blur = cv2.GaussianBlur(img, (5, 5), 0)

canny = cv2.Canny(blur, 0, 150, apertureSize = 3)

cv2.imshow('BLUR', blur)

cv2.imshow('SOBEL', sobel)

cv2.imshow('SOBEL X', sobel\_x)

cv2.imshow('SOBEL Y', sobel\_y)

cv2.imshow('LAPLACIAN', laplacian)

cv2.imshow('CANNY', canny)

cv2.waitKey(0)



Рисунок 1. Исходное изображение.

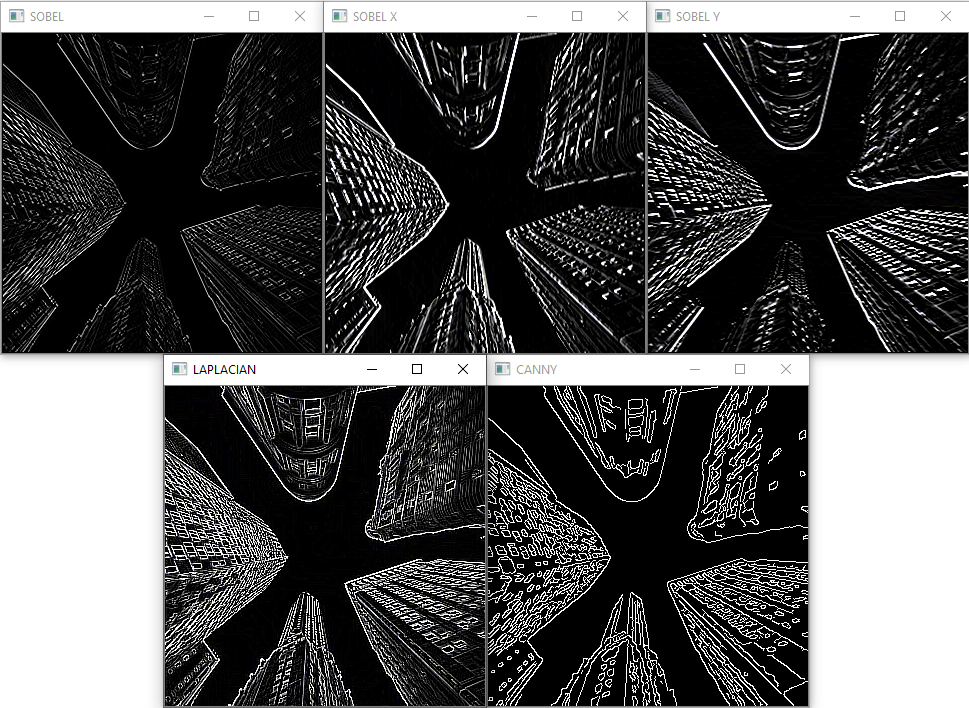


Рисунок 2. Найденные на изображении границы.



Рисунок 3. Исходное изображение.

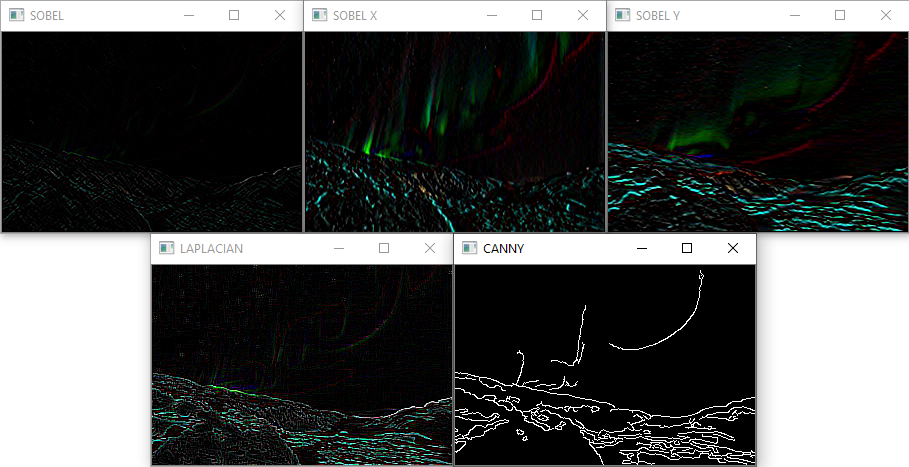


Рисунок 4. Найденные на изображении границы.

**import** cv2

color = cv2.imread('C:/Users/po/Desktop/img\_1.jpg', cv2.IMREAD\_REDUCED\_COLOR\_2)

gs = cv2.imread('C:/Users/po/Desktop/img\_1.jpg', cv2.IMREAD\_REDUCED\_GRAYSCALE\_2)

bin\_thresh, thresh = cv2.threshold(gs, 50, 150, cv2.THRESH\_BINARY)

contours1, hierarchy1 = cv2.findContours(thresh, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

**print**('Number of contours (threshold): ' + str(len(contours1)))

blur = cv2.GaussianBlur(color, (5, 5), 0)

canny = cv2.Canny(blur, 50, 150, apertureSize = 3)

contours2, hierarchy2 = cv2.findContours(canny, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

**print**('Number of contours (canny): ' + str(len(contours2)))

cv2.drawContours(color, contours2, -1, (0, 0, 255), 1)

cv2.imshow('THRESHOLD', thresh)

cv2.imshow('CANNY', color)

cv2.waitKey(0)

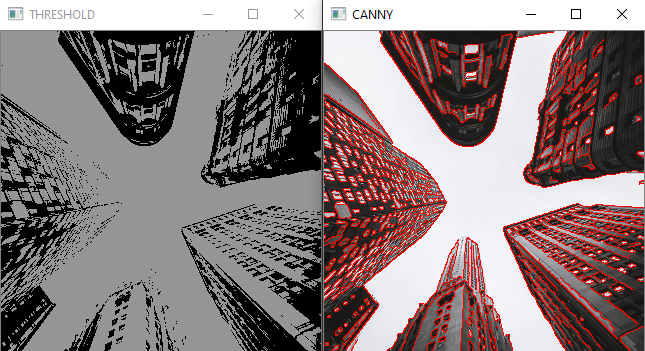


Рисунок 5. Найденные на изображении контуры.



Рисунок 6. Количество контуров

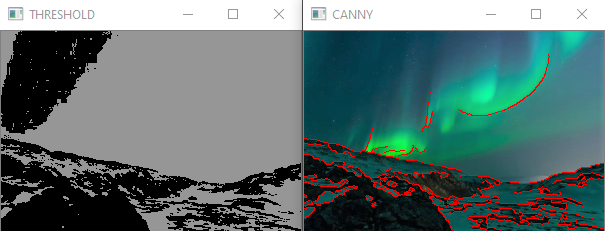


Рисунок 7. Найденные на изображении контуры.



Рисунок 8. Количество контуров

**import** cv2

**from** math **import** pi

img = cv2.imread('C:/Users/po/Desktop/circle.jpg', cv2.IMREAD\_REDUCED\_COLOR\_2)

blur = cv2.GaussianBlur(img, (5, 5), 0)

canny = cv2.Canny(blur, 0, 150, apertureSize = 3)

contours, hierarchy = cv2.findContours(canny, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

**print**('Number of contours: ' + str(len(contours)))

cv2.drawContours(img, contours, 1, (0, 0, 255), 2)

cv2.drawContours(img, contours, 2, (0, 255, 0), 2)

outside = contours[1]

inside = contours[2]

**print**('Outside circle length: ' + str(cv2.arcLength(outside, True)))

**print**('Inside circle length: ' + str(cv2.arcLength(inside, True)))

**print**('Outside circle area: ' + str(cv2.contourArea(outside)))

**print**('Inside circle area: ' + str(cv2.contourArea(inside)))

x, y, w, h = cv2.boundingRect(outside)

cv2.rectangle(img, (x,y), (x + w, y + h), (255, 0, 0), 2)

**print**('Outside rectangle area: ' + str(w \* h))

(x,y), radius = cv2.minEnclosingCircle(outside)

center = (int(x), int(y))

radius = int(radius)

cv2.circle(img, center, radius, (128, 128, 0), 2)

**print**('Outside enclosing circle area: ' + str(pi \* (radius \*\* 2)))

x, y, w, h = cv2.boundingRect(inside)

cv2.rectangle(img, (x,y), (x + w, y + h), (0, 128, 128), 2 cv2.LINE\_4)

**print**('Inside rectangle area: ' + str(w \* h))

(x,y), radius = cv2.minEnclosingCircle(inside)

center = (int(x), int(y))

radius = int(radius)

cv2.circle(img, center, radius, (128, 128, 128), 2)

**print**('Inside enclosing circle area: ' + str(pi \* (radius \*\* 2)))

cv2.imshow('CIRCLE', img)

cv2.waitKey(0)

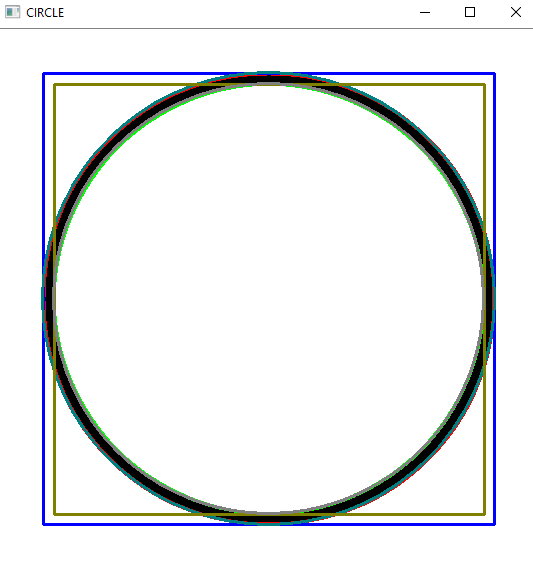


Рисунок 10. Окружность с контурами и ограничивающими фигурами

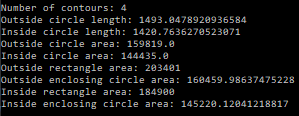


Рисунок 9. Информация об окружности и ограничивающих ее фигурах.

**Вывод:** в ходе выполнения работы были изучены методы поиска контуров и границ на изображениях. Также алгоритм поиска контуров был применен для получения информации о представленном на изображении объекте.