|  |  |
| --- | --- |
| 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 № 5

По курсу: «Техническое зрение»

Тема: Границы и контуры.

|  |  |  |
| --- | --- | --- |
| Студент гр. 6491 |  | Цуканов А.А. |
| Преподаватель |  | Моклева К. А. |

Санкт-Петербург

2020 г.

Цель работы: изучить способы выделения границ на изображении, поиск

контуров на границах и получения информации об объектах на основе контуров.

Задание 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.

Задание 2.1.

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?

Задание 2.2.

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.

Дополнительное задание №1

Для выполнения этого задания вам понадобится файл 5\_1.png. Для этого изображения все контуры на нечетном уровне вложенности нарисуйте красным цветом, все контуры на четном – синим цветом. На рисунке ниже приведен принцип нумерации уровней вложенности контуров.

Дополнительное задание №2

Для выполнения этого задания вам понадобится файл 5\_2.png. Для этого изображения нарисуйте все треугольники красным цветом, квадраты – синим, круги – зеленым. Используйте информацию о контурах и дополнительные операции над контурами.

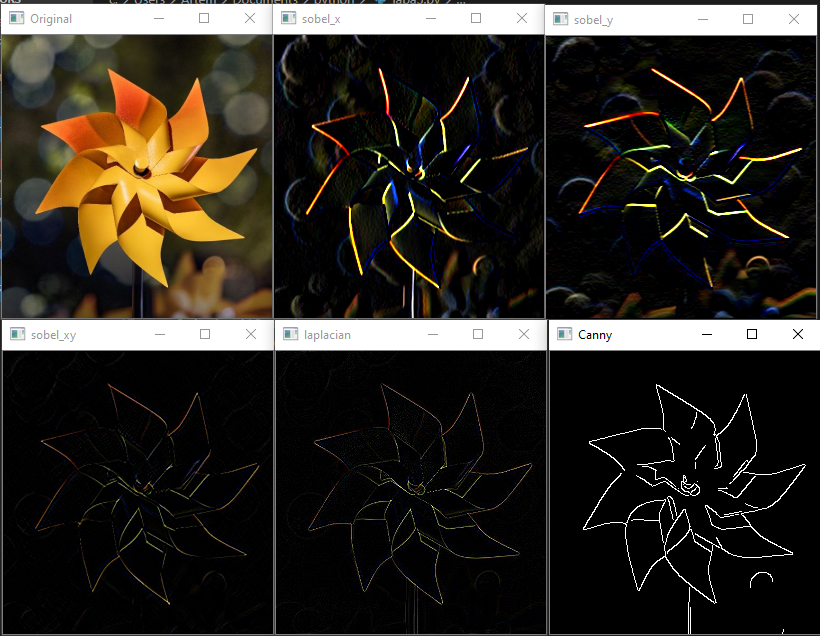
Задание №1

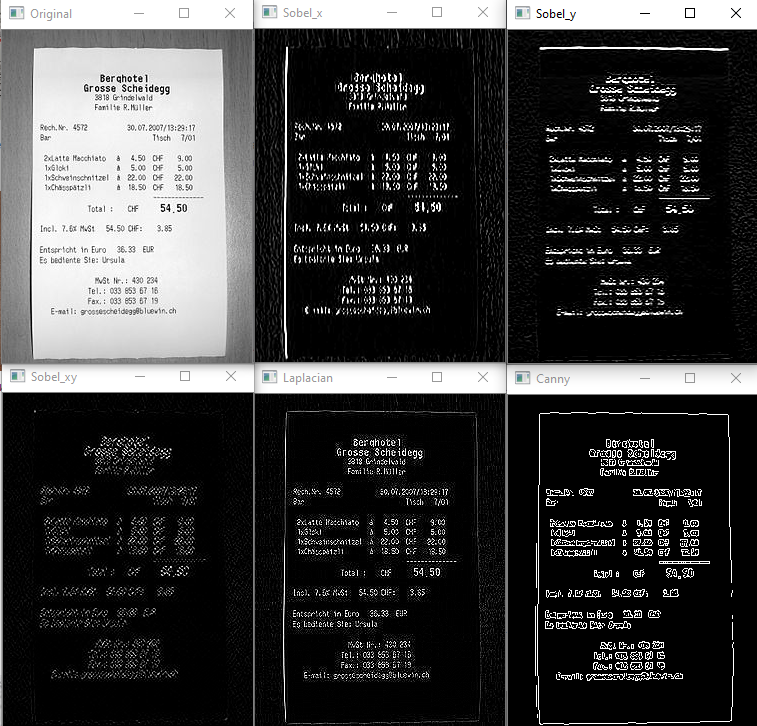
**Код программы:**

1. **from** cv2 **import** cv2
2. img = cv2.imread('321.jpg', cv2.IMREAD\_GRAYSCALE)
3. sobel\_x = cv2.Sobel(src = img, ddepth = cv2.CV\_8U, dx = 1, dy = 0, ksize = 3)
4. sobel\_y = cv2.Sobel(img, cv2.CV\_8U, 0, 1)
5. sobel\_xy = cv2.Sobel(img, cv2.CV\_8U, 1, 1)
6. laplacian = cv2.Laplacian(img,cv2.CV\_8U, cv2.BORDER\_DEFAULT)
7. blur = cv2.GaussianBlur(img,(1,1), cv2.BORDER\_DEFAULT)
8. canny = cv2.Canny(image = blur, threshold1 = 200, threshold2 = 225, apertureSize = 3, L2gradient = True)
9. cv2.imshow('Original', img)
10. cv2.imshow('Sobel\_x', sobel\_x)
11. cv2.imshow('Sobel\_y', sobel\_y)
12. cv2.imshow('Sobel\_xy', sobel\_xy)
13. cv2.imshow('Laplacian', laplacian)
14. cv2.imshow('Canny', canny)
15. cv2.waitKey(0)

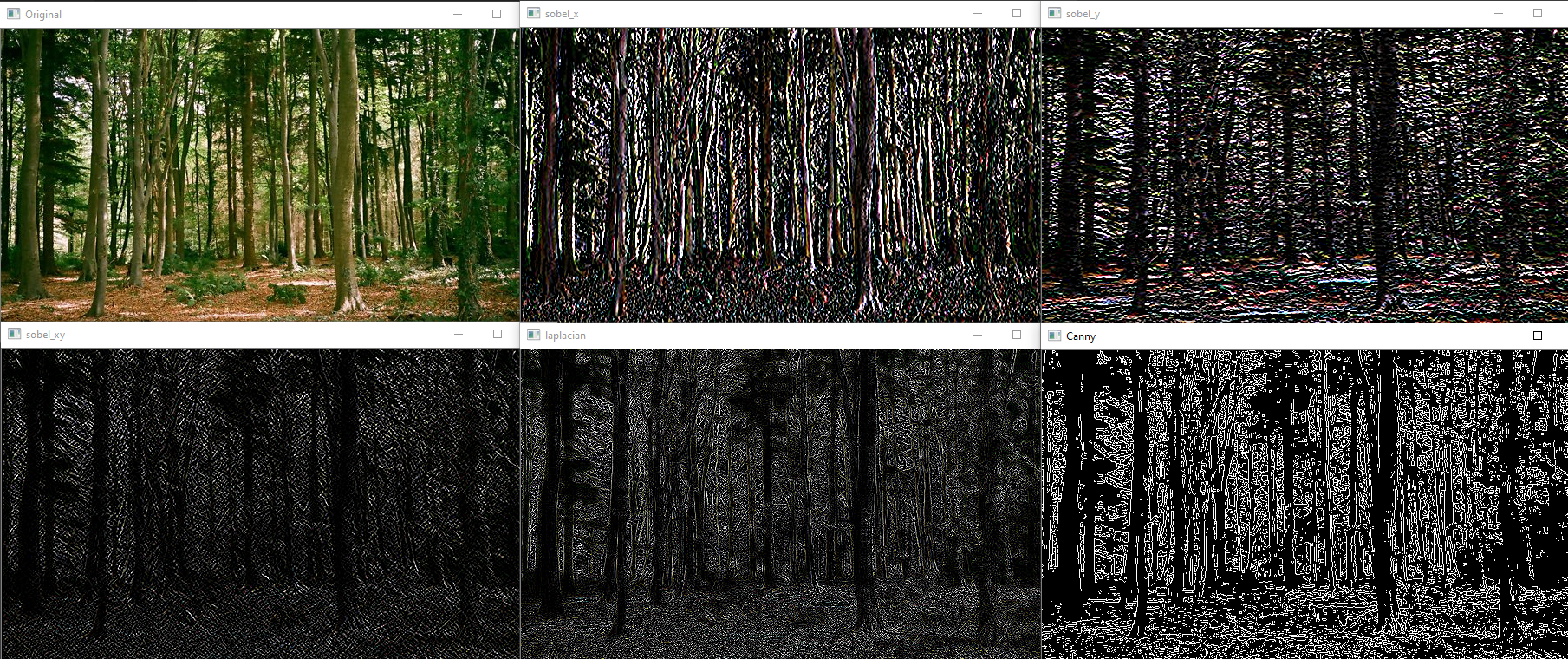
**Результат:**

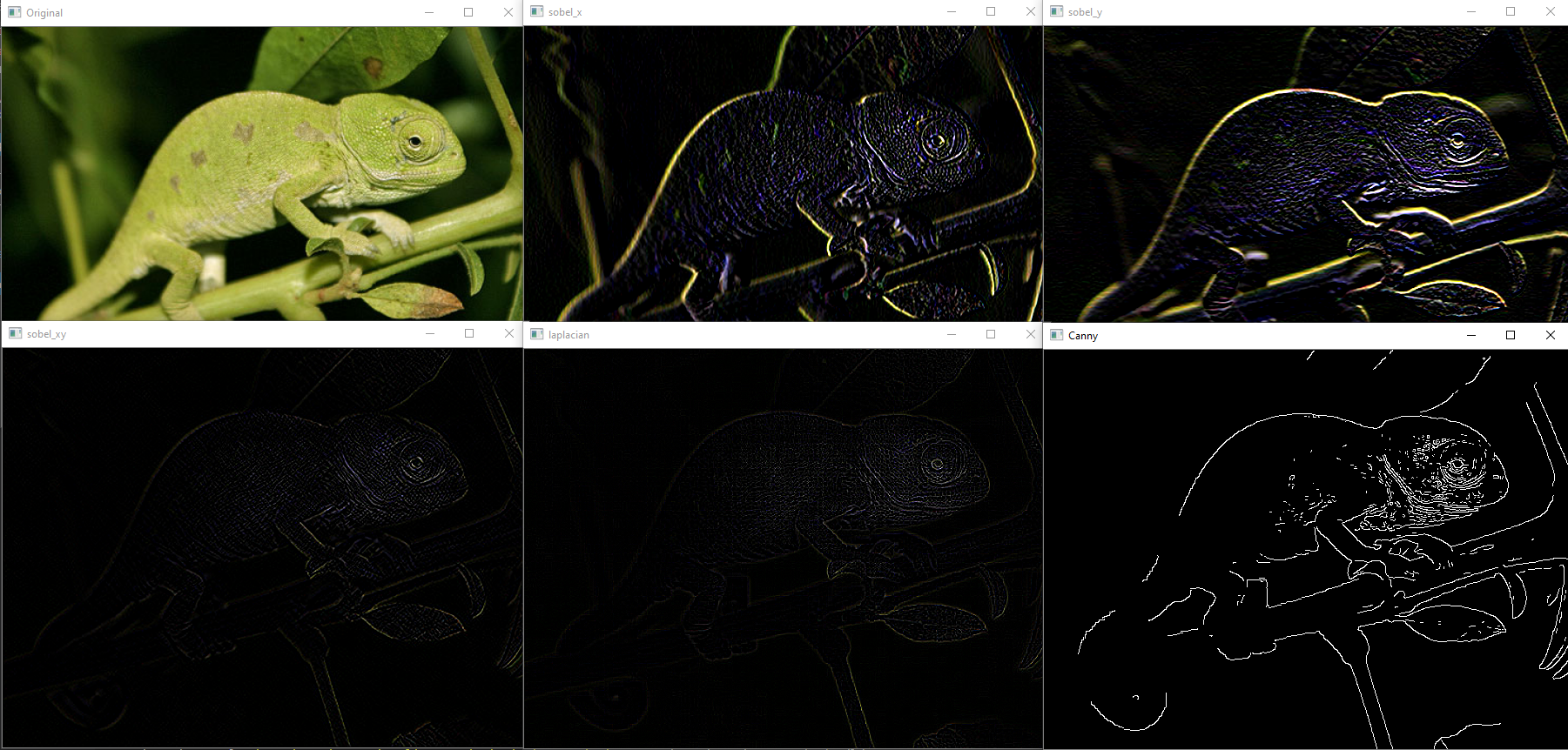
Изображения с четко отделимыми от фона границами:





Изображения с нечеткими границами, почти сливающимися с фоном:





Задание №2.1

**Код программы:**

1. **import** cv2
2. **from** math **import** pi
3. img = cv2.imread('223.png')
4. img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)
5. cv2.imshow('Orig', img)
6. ret, thresh = cv2.threshold(img\_gray, 150, 255, cv2.THRESH\_BINARY)
7. contours1, hierarchy = cv2.findContours(thresh, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_NONE)
8. blur = cv2.GaussianBlur(img\_gray,(1,1), cv2.BORDER\_DEFAULT)
9. canny = cv2.Canny(image = blur, threshold1 = 200, threshold2 = 225, apertureSize = 3, L2gradient = True)
10. contours2, hierarchy = cv2.findContours(canny, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_NONE)
11. **print**('Количество контуров функции threshold:' + str(len(contours1)) )
12. **print**('Количество контуров функции Canny():' + str(len(contours2)) + '**\n**')
13. cv2.drawContours(img, contours1[1], -1, (0, 0, 255), 2)
14. cv2.drawContours(img, contours1[2], -1, (0, 0, 255), 2)
15. '''cv2.imshow('Image gray', img\_gray)
16. cv2.imshow('Threshold', thresh)
17. cv2.imshow('Canny', canny)'''
18. **print**('Площадь внешнего контура:' + str(cv2.contourArea(contours1[1])) )
19. **print**('Площадь внутренего контура:' + str(cv2.contourArea(contours1[2])) )
20. **print**('Длина внешнего контура:' + str(cv2.arcLength(contours1[1], True)) )
21. **print**('Длина внутренего контура:' + str(cv2.arcLength(contours1[2], True)) + '**\n**')
22. x, y, w, h = cv2.boundingRect(contours1[1])
23. x1, y1, w1, h1 = cv2.boundingRect(contours1[2])
24. img=cv2.rectangle(img, (x, y+h), (x+w, y), (0, 255, 0), 2)
25. img=cv2.rectangle(img, (x1, y1+h1), (x1+w1, y1), (0, 255, 0), 2)
26. **print**('Площадь внешнего ограничивающего прямоугольника: '+ str(w\*h))
27. **print**('Площадь внутренего ограничивающего прямоугольника: '+ str(w1\*h1))
29. (x,y),radius = cv2.minEnclosingCircle(contours1[1])
30. center = (int(x),int(y))
31. **print**('Площадь внешней ограничивающей окружности: ' + str(pi\*(radius\*\*2)) )
32. radius = int(radius)
33. img=cv2.circle(img,center,radius,(222,0,0),3)
34. (x,y),radius = cv2.minEnclosingCircle(contours1[2])
35. center = (int(x),int(y))
36. **print**('Площадь внутреней ограничивающей окружности: ' + str(pi\*(radius\*\*2)))
37. radius = int(radius)
38. img=cv2.circle(img,center,radius,(222,0,0),2)
39. cv2.imshow('Orig2', img)
40. cv2.waitKey(-1)

**Результат:**

Количество контуров функции threshold:3

Количество контуров функции Canny():4

Площадь внешнего контура:142390.0

Площадь внутренего контура:130732.0

Длина внешнего контура:1409.449914932251

Длина внутренего контура:1351.5087900161743

Площадь внешнего ограничивающего прямоугольника: 181476

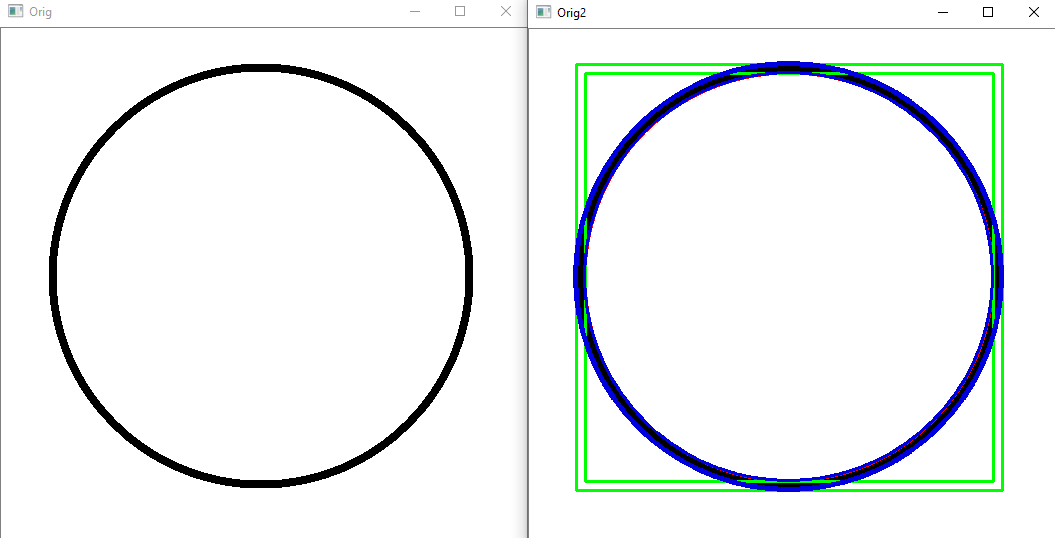
Площадь внутренего ограничивающего прямоугольника: 166464

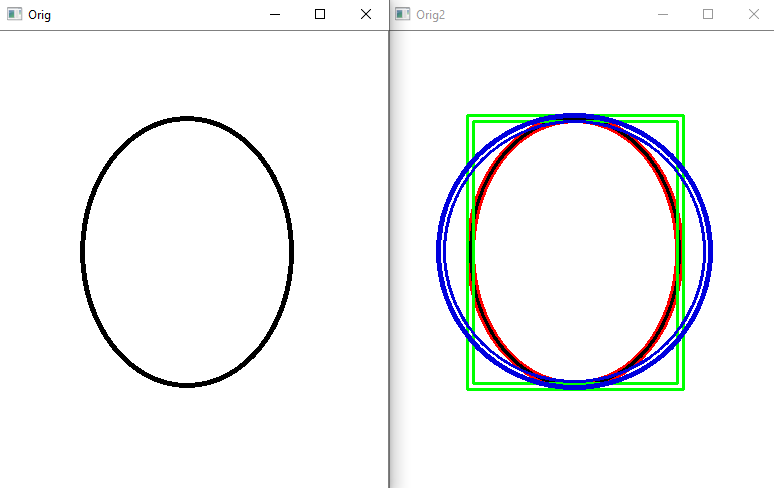
Площадь внешней ограничивающей окружности: 143140.4229897925

Площадь внутреней ограничивающей окружности: 131413.2874316227

Причина, по которой мы получаем разное количество контуров, заключается в том, что мы применяем детектор края Кенни к нашему изображению. Это дает результат «белых линий» - краев на изображении. Получаются "отдельные" линии, которые также являются контурами.

Значения площадей и длин получены в пикселях. Площади фигур, которые ограничивают контуры, больше, потому что они больше пикселей занимают на изображении.



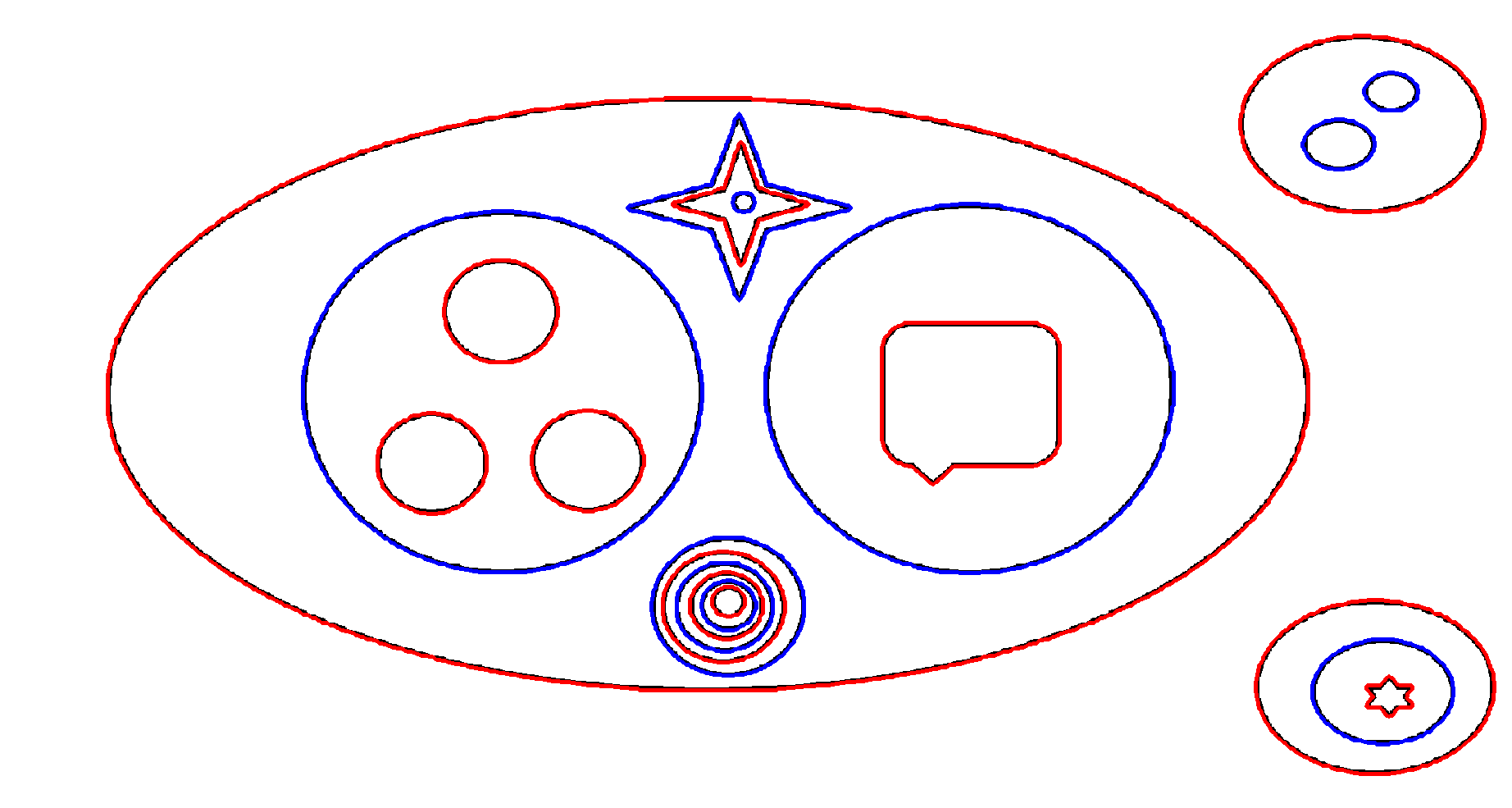


Дополнительное задание №1

**Код программы:**

1. **import** cv2
2. **from** math **import** pi
3. **import** numpy **as** np
4. img = cv2.imread('12345.png')
5. cv2.imshow('Orig', img)
6. circle\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)
7. ret, thresh = cv2.threshold(circle\_gray, 150, 255, cv2.THRESH\_BINARY)
8. contours1, hierarchy = cv2.findContours(thresh, cv2.RETR\_TREE , cv2.CHAIN\_APPROX\_NONE)
9. k=0
10. ii=0
11. **for** i **in** range (1,len(contours1),2):
12. k=hierarchy[0][i][3]
13. **while** k!=0:
14. ii+=1
15. k=hierarchy[0][hierarchy[0][k][3]][3]
16. **if** ii%2==0:
17. cv2.drawContours(img, contours1[i], -1, (0, 0, 255), 3)
18. **if** ii%2==1:
19. cv2.drawContours(img, contours1[i], -1, (255, 0, 0), 3)
20. ii=0
21. cv2.imshow('Orig', img)
22. cv2.waitKey(-1)

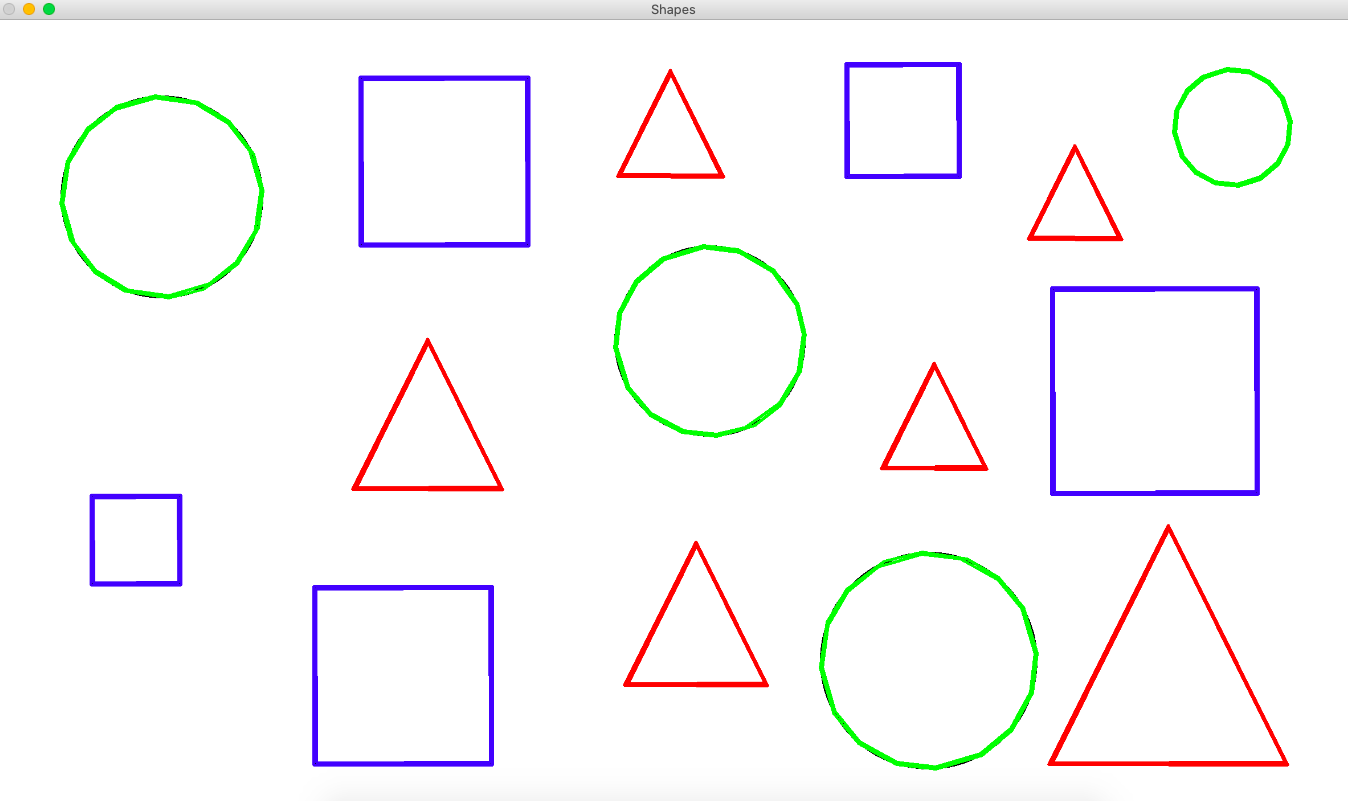
**Результат:**



Дополнительное задание №2

**Код программы:**

1. **from** cv2 **import** cv2
2. **import** numpy **as** np
3. img = cv2.imread('5\_2.png')
4. img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)
6. ret, thresh = cv2.threshold(img\_gray, 200, 255, cv2.THRESH\_BINARY\_INV)
8. contours, hierarchy = cv2.findContours(thresh, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_NONE)
10. **for** i **in** range(len(contours)):
11. approx = cv2.approxPolyDP(curve = contours[i], epsilon = 0.01 \* cv2.arcLength(contours[i], True), closed = True)
12. **print**(approx)
13. **if** len(approx) == 3:
14. cv2.drawContours(image = img, contours = [approx], contourIdx = 0, color = (0, 0, 255), thickness = 3)
15. **elif** len(approx) == 4:
16. cv2.drawContours(img, [approx], 0, (255, 0, 0 ), 3)
17. **else**:
18. cv2.drawContours(img, [approx], 0, (0, 255, ), 3)
20. cv2.imshow('Shapes', img)
21. cv2.waitKey(0)
22. cv2.destroyAllWindows

**Результат:**

Вывод: в ходе лабораторной работы были изучены способы выделения границ на изображении, поиск контуров на границах и получения информации об объектах на основе контуров.