МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе № 3.13

«Нахождение и обработка контуров»

по дисциплине «Технологии распознавания образов»

Выполнил студент группы
ПИЖ-б-о-21-1
Зиберов Александр
« » мая 2023 г.
Подпись студента_____
Работа защищена
« »_____20__г.
Проверил Воронкин Р.А. _____
(подпись)

Ставрополь, 2023

Цель работы:

Обнаружение и выделение контуров на изображении, анализ контуров. Изучение функций cv2.findContours(), cv2.drawContours().

Выполнение работы:

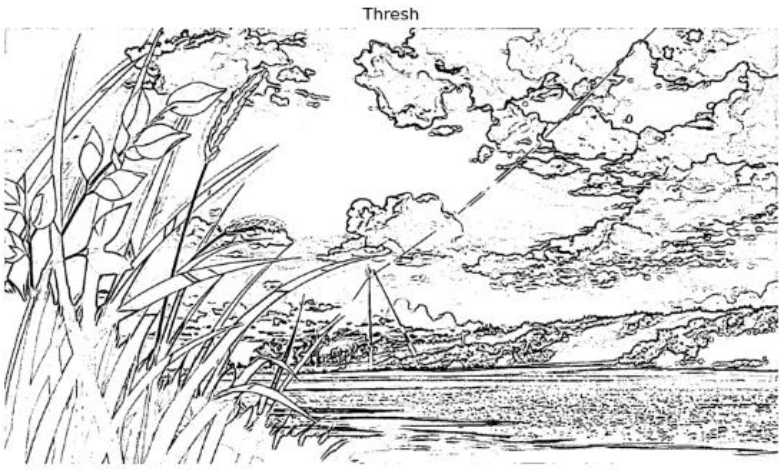
Проработать примеры лабораторной работы в отдельном ноутбуке.

Задание 7.1.

С помощью функции cv2.findContours найти все контуры изображения.

In [2]:
```
img = cv2.imread('pictures/img.jpg', 0)
img = cv2.medianBlur(img, 5)

thresh = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY,11,2)
plt.figure(figsize=(10,10))
plt.imshow(thresh,cmap = 'gray'),plt.title("Thresh")
plt.axis('off');
```



In [3]:
```
contours, hierarchy = cv2.findContours(
    thresh.copy(),
    cv2.RETR_LIST,
    cv2.CHAIN_APPROX_SIMPLE
)

for cnt in contours:
    cv2.drawContours(img, cnt, -1, (0, 255, 0), 3)

plt.figure(figsize=(10,10))
plt.imshow(img,cmap = 'gray')
plt.title("Contours")
plt.axis('off');
```

Рисунок 1 – Пример 1

Задание 7.2.

Протестировать функцию поиска контура cv2.findContours с аргументом cv2.CHAIN_APPROX_SIMPLE, который экономит память.

In [4]:
```
img = cv2.imread('pictures/nah.jpg', 0)
image = cv2.medianBlur(img, 5)

thresh = cv2.adaptiveThreshold(img, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY_INV, 3,10)

contours, hierarchy = cv2.findContours(thresh.copy(), cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
cv2.drawContours(image, contours, -1, (255,255,255),1);

plt.figure(figsize=(10,10))
plt.subplot(121),plt.imshow(thresh,cmap = 'gray'),plt.title('Thresh')
plt.axis('off')
plt.subplot(122),plt.imshow(image,cmap = 'gray'),plt.title('Contours')
plt.axis('off')
plt.show();
```

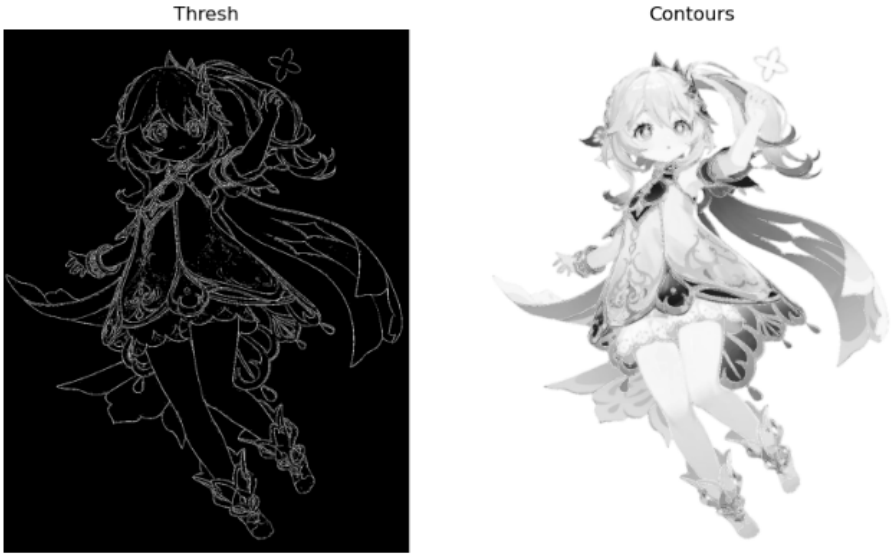


Рисунок 2 – Пример 2

Задание 7.3

Выделить границу методом Канни.

In [5]:
```
img = cv2.imread('pictures/img.jpg', 0)
img = cv2.resize(img, (900, 600))
edges = cv2.Canny(img,700,100,apertureSize = 3)

plt.figure(figsize=(20,20))
plt.subplot(121),plt.imshow(img,cmap = 'gray'),plt.title('Original')
plt.axis('off')
plt.subplot(122),plt.imshow(edges,cmap = 'gray'),plt.title('Cannys edges')
plt.axis('off')
plt.show()
```

Рисунок 3 – Пример 3

Индивидуальное задание

Индивидуальное задание

1. Найти и отрисовать контуры фигур на изображении
2. Определить, какие фигуры изображены.

In [1]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt

def imgshow(image, conversion=cv2.COLOR_BGR2RGB):
    image = cv2.cvtColor(image, conversion)
    plt.imshow(image)
    plt.axis("off")
    plt.xticks([])
    plt.yticks([])
    plt.show()

img_path = 'pictures/shapes.jpeg'
```

Задание 1.

Отобразим контуры методом Канни при помощи функции cv2.Canny(), предварительно переведя изображение в градации серого.

In [2]:
```
img = cv2.imread(img_path)
img_orig = img.copy()
imgshow(img_orig)

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
edges = cv2.Canny(gray, 50, 150, apertureSize=3)

contours, hierarchy = cv2.findContours(edges, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

len_cont = len(contours)
print("Количество контуров: ", len_cont)

imgshow(edges)
```

```
Количество контуров:  14
```

Рисунок 13 – Индивидуальное задание (1)

Задание 2.

Для того, чтобы определить, какие фигуры изображены, воспользуемся функцией cv2.approxPolyDP(). При помощи нее найдем приблизительные точки контура, которые и будем использовать в качестве определения вершин фигур.

Аргументы функции cv2.approxPolyDP():
cnt: контур, представленный в виде numpy массива точек
epsilon: точность аппроксимации. Она определяет максимально допустимое расстояние между исходным контуром и его аппроксимацией. Чем меньше значение epsilon, тем более точной будет аппроксимация.
closed: логическое значение, которое указывает, следует ли замкнуть полученный контур.

In [3]:
```
for cnt in contours:
    # Количество вершин
    approx = cv2.approxPolyDP(cnt, 0.01 * cv2.arcLength(cnt, True), True)
    if len(approx) == 3:
        cv2.drawContours(img, [approx], 0, (0, 255, 0), 2)
        cv2.putText(img, "Треугольник", (76, 207), cv2.FONT_HERSHEY_COMPLEX, 1, 0, 1)
    if len(approx) == 4:
        cv2.drawContours(img, [approx], 0, (255, 0, 0), 2)
        cv2.putText(img, "Прямоугольник", (1070, 300), cv2.FONT_HERSHEY_COMPLEX, 1, 0, 1)
    if len(approx) == 7:
        cv2.drawContours(img, [approx], 0, (0, 0, 100), 2)
        cv2.putText(img, "Семиугольник", (580, 214), cv2.FONT_HERSHEY_COMPLEX, 1, 0, 1)

plt.figure(figsize=(15,15))
plt.subplot(121),plt.imshow(img_orig,cmap = 'gray'),plt.title('Оригинал')
plt.axis('off')
plt.subplot(122),plt.imshow(img,cmap = 'gray'),plt.title('Обнаруженные фигуры')
plt.axis('off')
plt.show()
```

Рисунок 14 – Индивидуальное задание (2)

Вывод: В результате выполнения работы были изучены методы нахождения и обработки контуров.

1. Какие параметры выводит функция cv2.findContours()?

Функция cv2.findContours() выводит массив изображения, массив контуров и иерархию контуров.

2. Каковы параметры функции cv2.drawContours()?

Первым аргументом функции cv2.drawContours() является исходное изображение, вторым аргументом являются контуры, третьим аргументом является индекс контуров.

3. Каким образом можно нарисовать все контуры изображения?

Чтобы нарисовать все контуры изображения, нужно использовать функцию img = cv2.drawContours(img, contours,-1, (0,255,0),3).

4. Как изменить параметры метода аппроксимации контура в функции cv2.findContours()?

Для изменения параметров метода аппроксимации контура в функции cv2.findContours() нужно указать третий аргумент, который определяет метод аппроксимации. Доступны следующие методы

cv2.CHAIN_APPROX_NONE,

cv2.CHAIN_APPROX_SIMPLE,

cv2.CHAIN_APPROX_TC89_L1,

cv2.CHAIN_APPROX_TC89_KCOS.

5. Почему хранение всех координат пикселей контура не рационально?

Хранение всех координат пикселей контура занимает много памяти, что неэффективно.

6. Каким образом можно сохранить только начальную и конечную координаты отрезка, если контур включает в себя прямые отрезки?

Для сохранения только начальной и конечной координаты отрезка в контуре можно использовать метод аппроксимации cv2.CHAIN_APPROX_SIMPLE в функции cv2.findContours()

7. Что означает аргумент cv2.CHAIN_APPROX_SIMPLE в функции cv2.findContours()?

Аргумент cv2.CHAIN_APPROX_SIMPLE в функции cv2.findContours() позволяет удалять все лишние точки в контуре и сжимать его, что экономит память.

8. Каким образом работает метод Канни?

Метод Канни обнаруживает границы связанной области изображения, выполняя поиск локальных максимумов градиента f(x, y). Градиент вычисляется после применения фильтра Гаусса.

9. Какие аргументы принимает функция Канни cv2.Canny()?

Функция Канни cv2.Canny(img, T_lower, T_upper,apertureSize = 3) содержит в скобках следующие аргументы: img – входное изображение, T_lower: нижний порог, T_upper: верхний порог, apertureSize – размер апертуры оператора Собеля.

10. Каковы этапы алгоритма Канни?

Предварительное сглаживание по Гауссу для уменьшения шума, вычисление амплитуды и направления градиента изображения, обычно выбирается один из четырех возможных углов: 0º, 45°, 90° и 135°, не максимальное подавление: исключаются некраевые пиксели, это позволяет утончать края, использование порогового гистерезиса, для которого необходимы два порога: высокий и низкий порог. Если амплитуда градиента

исследуемых пикселей больше, чем значения высокого порога, то эти пиксели резервируются как краевые пиксели. Если амплитуда положения пикселя меньше нижнего порога, то пиксель исключается. Если амплитуда интенсивности пикселя находится между двумя порогами, пиксель сохраняется только тогда, когда он подключен к пикселю выше верхнего порога.