МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет»

Кафедра инфокоммуникаций

Отчёт по практическому занятию №13

«Нахождение и обработка контуров»

по дисциплине «Теории распознавания образов»

Выполнил студент группы ПИЖ-б-о-21-1
Образцова М.Д. « »_____20__г.
Подпись студента _____
Работа защищена « »_____20__г.
Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2023

Цель: обнаружение и выделение контуров на изображении, анализ контуров. Изучение функций cv2.findContours(),cv2.drawContours ()

Выполнение работы

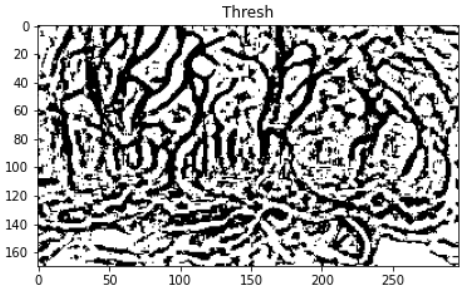
примеры

Задание 7.1. С помощью функции cv2.findContours найти все контуры изображения.

In [35]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
```

In [11]:
```
img = cv2.imread('forest.jpg', 0)
img = cv2.medianBlur(img, 5)
```

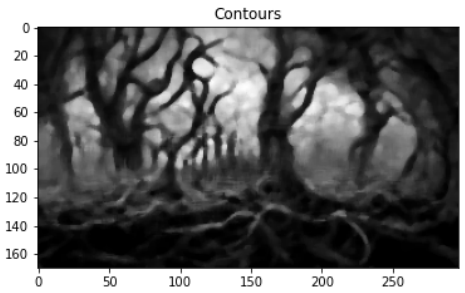
In [14]:
```
thresh = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY,11,2)
plt.imshow(thresh,cmap = 'gray'),plt.title("Thresh");
```



In [15]:
```
contours, hierarchy = cv2.findContours(thresh.copy(),
cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
cnt = contours[4]
img = cv2.drawContours(img, [cnt], 0, (0,255,0), 3)
```

In [17]:
```
plt.imshow(img,cmap = 'gray'),plt.title("Contours")
```

Out[17]:
```
(<matplotlib.image.AxesImage at 0x1fd811dba30>, Text(0.5, 1.0, 'Contours'))
```



Задание 7.2. Протестировать функцию поиска контура cv2.findContours с аргументом cv2.CHAIN_APPROX_SIMPLE, который экономит память.

In [29]:
```
img = cv2.imread('kot.jpg', 0)
img = cv2.resize(img, (900, 600))
image = cv2.medianBlur(img, 5)
```

индивидуальное задание

Дано изображение.

- Задача - применить алгоритмы обнаружения контуров с помощью методов Собеля и Робертса для выделения и обводки контуров различных объектов на изображении.

[21]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
```

Загрузка изображения

[59]:
```
image = cv2.imread('kis.jpg')
```

с помощью метода Собеля

Преобразование в оттенки серого

[60]:
```
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

применение фильтра Гаусса для сглаживания изображения и оператора Собеля для обнаружения границ

[61]:
```
# Применение фильтра Гаусса для сглаживания изображения
blurred = cv2.GaussianBlur(gray, (5, 5), 0)
```

[62]:
```
# Применение оператора Собеля для обнаружения границ
gradient_x = cv2.Sobel(blurred, cv2.CV_64F, 1, 0, ksize=3)
gradient_y = cv2.Sobel(blurred, cv2.CV_64F, 0, 1, ksize=3)
gradient = cv2.add(np.absolute(gradient_x), np.absolute(gradient_y))
gradient = gradient.astype(np.uint8)
```

применение пороговое значение, чтобы получить бинарное изображение с контурами

[63]:
```
# Применение пороговой фильтрации для получения бинарного изображения с контурами
_, binary = cv2.threshold(gradient, 50, 255, cv2.THRESH_BINARY)
```

[64]:
```
# Нахождение контуров методом cv2.findContours()
contours, _ = cv2.findContours(binary, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
```

[65]:
```
# Создание копии изображения для отображения контуров
contour_image = image.copy()

# Отрисовка контуров на изображении
cv2.drawContours(contour_image, contours, -1, (0, 255, 0), 2)
```

[65]:
```
array([[[158, 123,  80],
        [  0, 255,   0],
        [  0, 255,   0],
        ...,
        [116,  72,   0],
        [109,  70,   1],
```