МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций «Бинарные изображения, основные характеристики бинарных

изображений»

Отчет по лабораторной работе № 9 по дисциплине «Технологии распознавания образов»

Выполнил студент групп	іы ПИЖ-б-о-2	1-1
Халимендик Я. Д. « »	2023г.	
Подпись студента	<u>-</u>	
Работа защищена « »	20_	_г.
Проверил Воронкин Р.А.		
	(подпись)	

Цель работы: изучение методов цифровой обработки бинарных изображений, геометрических характеристик этих изображений, способов получения дополнительных параметров бинарных изображений. Изучение основных функций OpenCv, применяемых для цифровой обработки бинарных изображений.

Ход работы:

Задание 3.1. Вычислить площадь s, периметр p, ширину w, высоту h, отношение ширины к высоте w/h, отношение площади изображения к площади описывающего прямоугольника s/(wh), эквивалентный диаметр, центр масс, моменты бинарного изображения.

Лабораторная работа №9

Бинарные изображения, основные характеристики бинарных изображений

Задание 3.1.

Вычислить площадь s, периметр p, ширину w, высоту h, отношение ширины к высоте w/h, отношение площади изображения к площади описывающего прямоугольника s/(wh), эквивалентный диаметр, центр масс, моменты бинарного изображения.

```
In [2]: import cv2
import numpy as np
import matplotlib.pyplot as plt

In [27]: img = cv2.imread('cat.jpg',0)
    imag = cv2.imread('cat.jpg',0)
    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.axis("off")

Out[27]: (-0.5, 459.5, 319.5, -0.5)
```



```
In [30]: ret, thresh = cv2.threshold(img, 0, 255, 0)
contours, hierarchy = cv2.findContours(thresh, 5, 5)
plt.imshow(cv2.cvtColor(thresh, cv2.COLOR_BGR2RGB))
plt.axis("off")
```

Out[30]: (-0.5, 459.5, 319.5, -0.5)



Рисунок 1 – Результат работы

```
In [42]: # Создание контура
                   cnt = contours[0]
                  # Вычисление площади
                   s = cv2.contourArea(cnt)
                  # Вычисление перимет
                  p = cv2.arcLength(cnt, True)
                   M = cv2.moments(cnt)
                  x, y, w, h = cv2.boundingRect(cnt)
In [36]: imag = cv2.rectangle(imag, (x, y), (x+w, y+h), (0, 255, 0), 2) #Рамка
                   plt.imshow(cv2.cvtColor(imag, cv2.COLOR_BGR2RGB))
plt.axis("off")
                   plt.show()
In [43]: asprat_ratio = float(w) / h # соотношение сторон
                   rectar = w * h
s_ratio = float(ar) / rectar
                   eqdiam = np.sqrt(4*ar / np.pi)
In [45]: print("Площадь s: ", s) print("Периметр p: ", p) print("Моменты M: ", M) print("X, y, w, h: ", x, y, w, h) print(f"Ширина w: {w}, Высота h: {h}") print(f"Отношение ширины к высоте w/h: {asprat_ratio}") print("Отношение s/(wh): ", s_ratio) print("Эквивалентный диаметр: ", eqdiam)
                  Площадь s: 146421.0
Периметр p: 1556.0
Моменты M: { "m00": 146421.0, 'm10": 33603619.5, 'm01": 23354149.5, 'm20": 10282707567.0, 'm11": 5359777310.25, 'm02": 49666491
27.0, 'm30": 3539822079939.75, 'm21": 1640091856936.5, 'm12": 1139845974646.5, 'm03": 1188270803634.75, 'mu20": 2570676891.75, 'mu11": 0.0, 'mu02": 1241662281.75, 'mu30": 0.0, 'mu21": 0.0, 'mu12": 0.0, 'mu03": 0.0, 'nu20": 0.11990595611285268, 'nu11": 0.0, 'mu02": 0.05791575889615106, 'nu30": 0.0, 'nu21": 0.0, 'nu12": 0.0, 'nu03": 0.0}
                   х, у, w, h: 0 0 460 320
Ширина w: 460, Высота h: 320
                  Отношение ширины к высоте w/h: 1.4375
Отношение s/(wh): 0.9947078804347826
Эквивалентный диаметр: 431.7742551144837
```

Рисунок 2 – Результат работы

Задание 3.2. Используя изображение маски определить крайние точки, минимальное и максимальное значения и их координаты для бинарного изображения. Найти среднюю интенсивность изображения в градациях серого, ориентацию бинарного изображения с выделенной осью.

Задание 3.2.

In [47]: img = cv2.imread('cat.jpg', 0)

Используя изображение маски определить крайние точки, минимальное и максимальное значения и их координаты для бинарного изображения Найти среднюю интенсивность изображения в градациях серого, ориентацию бинарного изображения с выделенной осью.

```
ret, thresh = cv2.threshold(img, 0, 255, 0)
              contours, hierarchy = cv2.findContours(thresh, 5, 5)
cnt = contours[0]
In [48]: plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.axis("off")
Out[48]: (-0.5, 459.5, 319.5, -0.5)
In [50]: mask = np.zeros(img.shape, np.uint8)
              cv2.drawContours(mask, [cnt], 0, 255, -1)
pixpoin = np.transpose(np.nonzero(mask))
              minv, maxv, minl, maxl = cv2.minMaxLoc(img, mask=mask)
              rightmost = tuple(cnt[cnt[:, :, 0].argmin()][0])
rightmost = tuple(cnt[cnt[:, :, 0].argmax()][0])
topmost = tuple(cnt[cnt[:, :, 1].argmin()][0])
bottommost = tuple(cnt[cnt[:, :, 1].argmax()][0])
In [52]: (x,y),(MA,ma),ang=cv2.fitEllipse(cnt)
              meanv = cv2.mean(img,mask = mask)
In [55]: print(f"Пиксельные точки:\n {pixpoin}")
              print(f"Максимальное и минимальное значения и их координаты: {minv}, {max print(f"Крайние точки: {leftmost}, {rightmost}, {topmost}, {bottommost}") print(f"Средняя интенсивность: {meanv}")
                                                                           .
чения и их координаты: {minv}, {maxv}, {minl}, {maxl}")
               print(f"Ориентация: {ang}")
               Пиксельные точки:
                [[ 0 0]
[ 0 1]
[ 0 2]
                [319 457]
                [319 458]
                [319 459]]
              Максимальное и минимальное значения и их координаты: 0.0, 255.0, (115, 7), (0, 0)
Крайние точки: (0, 0), (459, 0), (0, 0), (459, 319)
Средняя интенсивность: (234.04796195652176, 0.0, 0.0, 0.0)
               Ориентация: 180.0
```

Рисунок 3 – Результат работы

Индивидуальное задание.

Считать цветное изображение, найти его контуры, вывести:

- 1. Все контуры
- 2. Половину контуров
- 3. Четверть контуров
- 4. Самый длинный контур

Задание

Считать цветное изображение, найти его контуры, вывести:

- 1. Все контуры
- 2. Половину контуров
- 3. Четверть контуров
- 4. Самый длинный контур

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

img = cv2.imread('cat.jpg', 0)
ret,thresh = cv2.threshold(img,128,255, cv2.THRESH_BINARY)
contours, hierarchy = cv2.findContours(thresh, 5, 5)

plt.axis('off')
plt.title("Оригинал")
plt.imshow(img, cmap='gray');
```





```
]: # Количества контуров
len_cont = len(contours)
print("Количество контуров: ", len_cont)
```

Количество контуров: 461

Найдем все контуры

```
imask = np.zeros(img.shape,np.uint8)
all_cont = cv2.drawContours(mask,contours,-1,255,2)
```

Найдем половину контуров

```
i = 0
cnt = []
while len(cnt) < len_cont/2:
    cnt.append(contours[i])
    i += 1

i mask = np.zeros(img.shape,np.uint8)
half_count = cv2.drawContours(mask,cnt,-1,255,2)
#plt.imshow(mask, cmap='gray');</pre>
```

Рисунок 4 – Результат работы

Найдем четверть контуров

```
i = 0
cnt = []
while len(cnt) < len_cont/4:
    cnt.append(contours[i])
    i += 1</pre>
```

Найдем самый длинный контур

```
mask = np.zeros(img.shape,np.uint8)
four_count = cv2.drawContours(mask,cnt,-1,255,2)

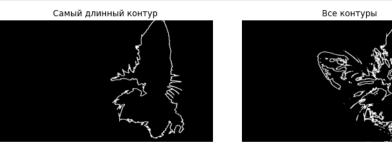
mask = np.zeros(img.shape,np.uint8)
max=0
sel_countour=None
for countour in contours:
    if countour.shape[0]>max:
        sel_countour=countour
        max=countour.shape[0]

max_kont = cv2.drawContours(mask, [sel_countour], -1, (255,255,255), 2)
```

Рисунок 5 – Результат работы

Организуем вывод изображений

```
fig, ax = plt.subplots(2, 2, figsize=(10,7))
fig.tight_layout()
signature = ["Самый длинный контур", "Все контуры", "Половина контуров", "Четверть контуров"]
images = [max_kont, all_cont, half_count, four_count]
pose = 221
i = 0
while i < 4:
    plt.subplot(pose)
    plt.axis('off')
    plt.title(signature[i])
    plt.imshow(images[i], cmap='gray')
    pose += 1
    i += 1
```



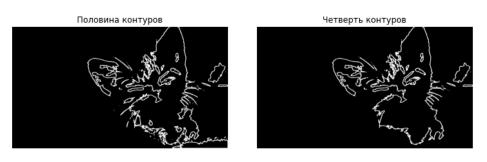


Рисунок 6 – Результат работы