Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

«Бинарные изображения, основные характеристики бинарных изображений»

ОТЧЕТ
по лабораторной работе №9
дисциплины
«Основы распознавания образов»

Выполнил:
Луценко Дмитрий Андреевич
2 курс, группа ПИЖ-б-о-21-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

_____
(подпись)

Проверил:
_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Бинарные изображения, основные характеристики бинарных изображений

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.

```
img = cv2.imread('jaguar.jpg', 0)
imag = cv2.imread('jaguar.jpg', 0)
ret, thresh = cv2.threshold(img, 0, 255, 0)
contours, hierarchy = cv2.findContours(thresh, 5, 5)
cnt = contours[0]
```

У функции findContours два возвращаемых значения: первое – контур, а второе – топологическая структура (иерархия). Контур (первое возвращаемое значение) – это список, в котором хранятся все контуры изображения. Каждый контур пред- ставляет собой массив numpy, содержащий координаты точек границы объекта (x, y).

```
ar = cv2.contourArea(cnt)
print(f"Площадь: {ar}")

prm = cv2.arcLength(cnt,True)
print(f"Периметр: {prm}")

M = cv2.moments(cnt)
print(f"Моменты: {M}")

x,y,w,h = cv2.boundingRect(cnt)
print(f"{x}, {y}, Ширина: {h}, высота: {w}")

imag = cv2.rectangle(imag,(x,y),(x+w,y+h),(0,255,0),2)
cv2.imshow('Rectan', imag)
asprat = float(w) / h
rectar = w * h
extent = float(ar)/rectar
eqdiam = np.sqrt(4 * ar / np.pi)
print(f"Отношение ширины к высоте: {asprat}, отношение s/(wh): {extent}")
print(f"Эквивалентный диаметр: {eqdiam}")
cv2.waitKey(0)
```

```
Площадь: 2344473.0
Периметр: 9215.81656563282
Моменты: {'m00': 2344473.0, 'm10': 1859361941.6666665, 'm01': 1881122050.8333333, 'm20': 1960771212768.8333, 'm11': 14798078029
80.9165, 'm02': 1955727235207.0, 'm30': 2313898459600535.5, 'm21': 1548919845203393.2, 'm12': 1540345345381533.2, 'm03': 228382
4240888970.5, 'mu20': 486142658668.0027, 'mu11': -12078326204.38379, 'mu02': 446381565569.7698, 'mu30': -12259139499061.5, 'mu2
1': -5175256276443.4375, 'mu12': 8673546258907.25, 'mu03': -1703483132929.5, 'nu20': 0.088444994777566, 'nu11': -0.002197436244
322689, 'nu02': 0.08121117234145443, 'nu30': -0.0014566221108439407, 'nu21': -0.0006149202170452906, 'nu12': 0.0010305845088979
734, 'nu03': -0.00020240663686588902}
0, 0, Ширина: 1564, высота: 1564
Отношение ширины к высоте: 1.0, отношение s/(wh): 0.9584550238420733
Эквивалентный диаметр: 1727.7371718996733
```

Рисунок 1 – Задания

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.

```
img = cv2.imread('jaguar.jpg', 0)
imag = cv2.imread('jaguar.jpg', 0)
ret, thresh = cv2.threshold(img, 0, 255, 0)
contours, hierarchy = cv2.findContours(thresh, 5, 5)
cnt = contours[0]
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)
leftmost = 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])
(x,y),(MA,ma),ang=cv2.fitEllipse(cnt)
meanv = cv2.mean(img,mask = mask)
print(f"Пиксельные точки:\n {pixpoin}")
print("Максимальное и минимальное значения и их координаты:"
      f"{minv}, {maxv}, {minl}, {maxl}")
print(f"Крайние точки:{leftmost}, {rightmost}, {topmost}, {bottommost}")
print(f"Средняя интенсивность: {meanv}")
print(f"Ориентация: {ang}")
cv2.waitKey(0)
```

```
Пиксельные точки:
 [[   0    0]
 [   0    1]
 [   0    2]
 ...
 [1563 1561]
 [1563 1562]
 [1563 1563]]
Максимальное и минимальное значения и их координаты:0.0, 255.0, (984, 3), (518, 940)
Крайние точки:(0, 0), (1563, 0), (0, 0), (1563, 1563)
Средняя интенсивность: (94.45942665696055, 0.0, 0.0, 0.0)
Ориентация: 98.41520690917969
```

Рисунок 2 – Задания

Оператор Собеля

Оператор Собеля — это дискретный дифференциальный оператор, вычисляющий приближение градиента яркости изображения. Оператор вычисляет градиент яркости изображения в каждой точке. Так находится направление наибольшего увеличения яркости и величина её изменения в этом направлении. Результат показывает, насколько «резко» или «плавно» меняется яркость изображения в каждой точке, а значит, вероятность нахождения точки на грани, а также ориентацию границы.

$$F_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$F_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

```
img = cv2.imread('jaguar.jpg', cv2.IMREAD_GRAYSCALE)

def sobel(img):
    a = 0
    b = 1
    c = 2
    f_x = np.array([[-b, -c, -b],[a, a, a],[b, c, b]])
    f_y = np.array([[-b, a, b],[-c, a, c],[-b, a, b]])
    res = cv2.filter2D(img,-1,f_x) + cv2.filter2D(img,-1,f_y)
    return res


res = sobel(img);
plt.imshow(res)
plt.axis('off')
```

```
(-0.5, 1563.5, 1563.5, -0.5)
```

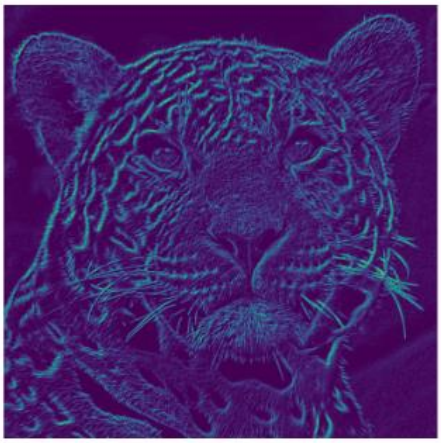

Рисунок 3 – Индивидуальное задание

Вывод: изучены методы цифровой обработки бинарных изображений, геометрических характеристик этих изображений, способов получения дополнительных параметров бинарных изображений. Изучены основные функций OpenCv, применяемые для цифровой обработки бинарных изображений.