

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

Институт цифрового развития
Кафедра инфокоммуникаций

«Цифровая обработка бинарных изображений»

ОТЧЕТ
по лабораторной работе №10
дисциплины
«Технологии распознавания образов»

Выполнила:

Кувшин Ирина Анатольевна
2 курс, группа ПИЖ-б-о-21-1,
010.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

(подпись)

Проверил:

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Цель работы:

Изучить основные операции геометрических преобразований изображений, такие как изменение размера, сдвиг, вращение, аффинное преобразование и т. д.

Ход выполнения:

```
File Edit View Navigate Code Refactor Run Tools Git Window Help IRT_10 - Jupyter_Lab10.ipynb
Project IRT_10 notebooks Jupyter_Lab10.ipynb README.md Jupyter_Lab10.ipynb ind.ipynb
notebooks
  checkpoints
  img
  ind.ipynb
  Jupyter_Lab10.ipynb
  gitignore
  LICENSE
  README.md
External Libraries
Scratches and Consoles

In 326 1 import cv2
      2 import numpy as np
      3 import matplotlib.pyplot as plt

In 327 1 # Функция для вывода изображения
      2 def img_print(original, res):
      3     pose = [121, 122]
      4     signature = ["Оригинал", "Измененное"]
      5     img = [original, res]
      6     i = 0
      7     while i < 2:
      8         plt.subplot(pose[i])
      9         plt.title(signature[i])
     10         plt.imshow(img[i])
     11         i += 1
     12

In 328 1 img = cv2.imread('img/cat_art.jpg', 1)
      2 img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

In 329 1 #Первый способ изменения размера задается в процентах
      2 scale_percent = 50 # процент изменения
      3 width = int(img.shape[1] * scale_percent / 100)
```

```
File Edit View Navigate Code Refactor Run Tools Git Window Help IRT_10 - Jupyter_Lab10.ipynb
Project IRT_10 notebooks Jupyter_Lab10.ipynb README.md Jupyter_Lab10.ipynb ind.ipynb
notebooks
  checkpoints
  img
  ind.ipynb
  Jupyter_Lab10.ipynb
  gitignore
  LICENSE
  README.md
External Libraries
Scratches and Consoles

In 327 1 #Первый способ изменения размера задается в процентах
      2 scale_percent = 50 # процент изменения
      3 width = int(img.shape[1] * scale_percent / 100)
      4 height = int(img.shape[0] * scale_percent / 100)
      5 dim = (width, height)
      6 resized = cv2.resize(img, dim, interpolation=
      7 cv2.INTER_AREA)
      8 print('Resized Dimensions : ', resized.shape)
      9 img_print(img, resized)

Resized Dimensions : (320, 240, 3)

0 Оригинал 0 Измененное
100 100
200 100
300 100
400 100
500 100
600 100
0 100 200 300 400 0 50 100 150 200
```

IRT_10 - Jupyter_Lab10.ipynb

Project: IRT_10

notebooks: Jupyter_Lab10.ipynb

img: img

ind.ipynb

giltignore

LICENSE

README.md

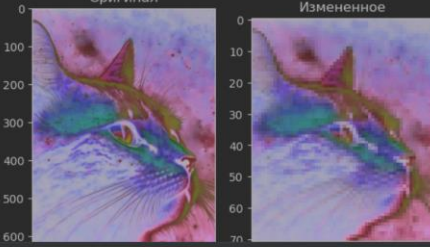
External Libraries

Scratches and Consoles

```
In 330: 1 # Второй способ изменения размера задается вручную
2 print('Original Dimensions : ', img.shape)
3 width = 58
4 height = 71
5 dim1 = (width, height)
6 # resize image
7 resized1 = cv2.resize(img, dim1, interpolation=cv2.INTER_AREA)
8 print('Resized Dimensions : ', resized1.shape)
9 img_print(img, resized1);
```

Original Dimensions : (640, 480, 3)
Resized Dimensions : (71, 58, 3)

Оригинал Измененное



IRT_10 - Jupyter_Lab10.ipynb

Project: IRT_10

notebooks: Jupyter_Lab10.ipynb

img: img

ind.ipynb

giltignore

LICENSE

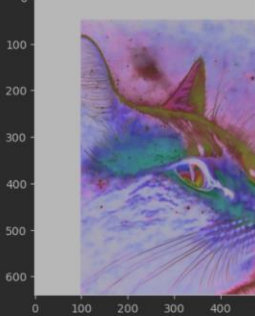
README.md

External Libraries

Scratches and Consoles

4.1.2. Сдвиг, смещение местоположения объекта

```
In 332: 1 rows, cols, colors = img.shape
2 M = np.float32([[1, 0, 100], [0, 1, 50]])
3 dst = cv2.warpAffine(img, M, (cols, rows))
4 plt.imshow(dst);
```



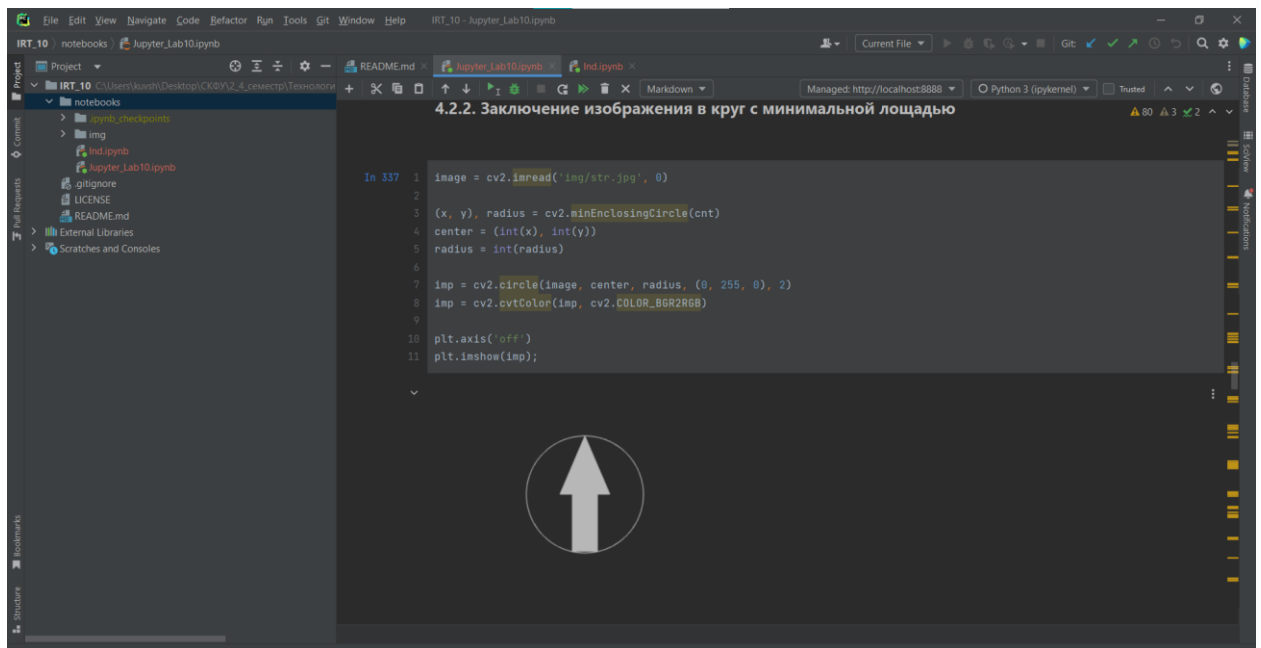


Рисунок 10.1– Код программы

Индивидуальное задание

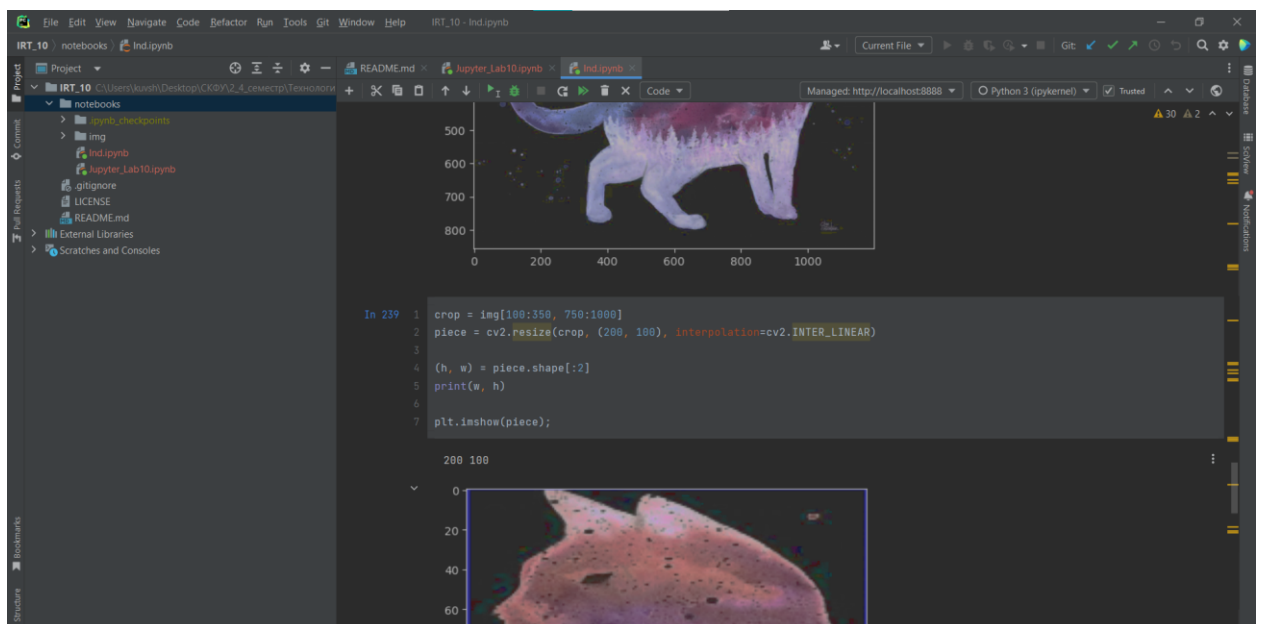


Рисунок 10.2– Код программы

Контрольные вопросы:

1. С помощью какой функции можно совершить изменение размера изображения?

`cv.resize` (`img`, `dim`, `interpolation=...`) Первый аргумент – матрица изображения, второй `dim` либо `width`, `height` – размер изображения, третий – метод интерполяции

2. Какие существуют способы изменения размера?

– Размер нового изображения указывается в процентах (например: 50%):
`scale_percent = 50`.

– Размер изображения задается вручную: `width=58, height=71`.

– Размер изображения задается с помощью коэффициента масштабирования.

3. Перечислите основные методы интерполяции.

`cv.INTER_AREA` – для сжатия, `cv.INTER_CUBIC` и `cv.INTER_LINEAR` – для масштабирования. По умолчанию используется метод интерполяции `cv.INTER_LINEAR`.

4. С помощью какой функции можно осуществить сдвиг изображения?

`cv2.warpAffine(src, M, dsize[, dst[, flags[, borderMode[, borderValue]]]])` `src` - изображение. Матрица `M` - преобразования. `dsize` - размер выходного изображения. `flags`-комбинация методов интерполяции (тип `int`!) `borderMode` - режим пикселей границы (тип `int`!) `borderValue` - (выделение) Значение заполнения границы; по умолчанию это 0.

5. С помощью какой функции можно осуществить вращение изображения?

`cv2.getRotationMatrix2D(center, angle, scale)` `center`: Центр вращения
`angle(θ)`: угол поворота. `scale`: коэффициент масштабирования

6. Что происходит при аффинной трансформации изображения?

При аффинном преобразовании все параллельные линии исходного изображения остаются параллельными и в выходном изображении.

7. Какие функции позволяют выполнить охват объекта?

Функция `cv2.drawContours()` возвращает структуру `box`, которая содержит 37 следующие аргументы: верхний левый угол (x, y), ширину, высоту, угол поворота. Чтобы нарисовать прямоугольник, нужны 4 угла прямоугольника, которые задаются функцией `cv2.boxPoints()`. Окружность с минимальной площадью, охватывающей объект, можно нарисовать с помощью функции `cv2.minEnclosingCircle()`. Используя функцию `cv2.ellipse()`, можно вписать изображение в эллипс с минимальной площадью.

8. Опишите процесс создания выпуклой оболочки вокруг контура

Чтобы нарисовать выпуклую оболочку вокруг контура некоторого изображения, выделяем все его крайние точки и соединяем их ломанной прямой линией. Ни одна точка изображения не должна выходить за пределы выпуклой оболочки. Импортируем цветное изображение и трансформируем его в полутоновое изображение. Функция `Canny` выделяет контуры, а с помощью функции `cv2.findContours()` создаем иерархию контуров. Выделяем только внешние контуры изображения. Затем, используя цикл `for`, проходим по каждому из контуров изображения. С помощью переменной `hull` создаем выпуклую оболочку сначала для первого контура, затем для каждого другого контура. В результате получим контур, охватывающий изображение.

9. Какая функция позволяет аппроксимировать контур?

Функция `cv2.approxPolyDP(cnt, epsilon, True)`. Первый аргумент `cnt = contours[i]` – массив с координатами пикселей контура, аргумент `epsilon` задается в процентах, с уменьшением `epsilon` максимальное расстояние между ломаной прямой, аппроксимирующей контур, и самим контуром также уменьшается. Значение этого аргумента вычисляется функцией `epsilon = 0.1 * cv2.arcLength(cnt, True)`.

10. Как осуществить выделение на изображении интересующей области, создание для нее отдельного изображения

Выделим на изображении интересующую нас область, заключив ее в прямоугольную рамку с помощью функции рисования `cv2.rectangle`. Фрагмент изображения, заключенный в рамке, выведем на экран. Используя функцию `.shape`, получим размер изображения и изменим его с помощью функции `cv2.resize`. Функция `cv2.getRotationMatrix2D` предназначена для поворота изображения, а функция `cv2.warpAffine` – для аффинного преобразования.