

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

**Кафедра инфокоммуникаций**

**Отчет по лабораторной работе № 3.7**

**«Основы цифровой обработки изображений в OpenCV. Краткие  
теоретические сведения»**

**по дисциплине «Основы программной инженерии»**

Выполнил студент группы

ПИЖ-б-о-21-1

Зиберов Александр

« » апреля 2023 г.

Подпись студента \_\_\_\_\_

Работа защищена

« » \_\_\_\_\_ 20\_\_ г.

Проверил Воронкин Р.А. \_\_\_\_\_  
(подпись)

Ставрополь, 2023

## Цель работы:

Изучение типов изображений, способов их формирования. Изучение основных функций OpenCV, применяемых для цифровой обработки изображений.

## Выполнение работы:

Проработать примеры лабораторной работы в отдельном ноутбуке.

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

def imshow(image, conversion=cv2.COLOR_BGR2RGB):
    image = cv2.cvtColor(image, conversion)
    # Show the image
    plt.imshow(image)
    # remove the axis / ticks for a clean looking image
    plt.xticks([])
    plt.yticks([])
    plt.show()
```

### Задание 1.1

Считать файл полноцветного изображения, создать для него матрицу изображения, затем вывести сначала полутоновое, затем цветное изображение на экран.

```
In [2]: image = cv2.imread('pictures/bliss.jpg', 1)
cv2.imshow('image', image)
imshow(image)
image = cv2.imread('pictures/bliss.jpg', cv2.IMREAD_GRAYSCALE)
# cv2.imshow('image', image)
imshow(image)
```



Рисунок 1 – Пример 1

## Задание 1.2

Используя код задания 1.1, в функции `cv2.imread()` присвоить флагу значение 1, затем вывести изображение на экран. Выполнить этот же код, заменив в функции `cv2.imread('cat.jpg', 1)` флаг 1 на флаг `cv2.IMREAD_COLOR`.

```
In [3]: img = cv2.imread('pictures/bliss.jpg', 1)
cv2.imshow('image', img)
imgshow(img)

img = cv2.imread('pictures/bliss.jpg', cv2.IMREAD_COLOR)
# cv2.imshow('image', img)
imgshow(img)
```



Рисунок 2 – Пример 2

## Задание 1.3

Сформировать матрицу изображения, записать ее в файл с расширением `png`. Изображение, записанное в этом файле, вывести на экран.

```
In [4]: img = cv2.imread('pictures/bliss.jpg', 1)
# запись изображения из матрицы в файл
cv2.imwrite('pictures/bliss_img.png', img)
img = cv2.imread('pictures/bliss_img.png')
# cv2.imshow('image', img)
imgshow(img)
```



Рисунок 3 – Пример 3

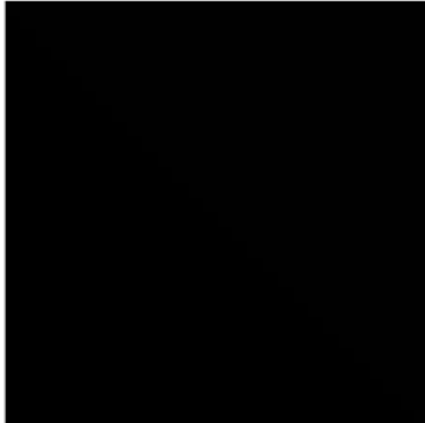
## Задание 1.4

Сформировать матрицу, у которой выше диагонали единицы, а ниже – нули, записать ее в файл, затем считать файл и вывести на экран.

```
In [5]: n = 28
a = np.ones([28,28])

for i in range(n):
    a[i][i] = 1
for i in range(n):
    for j in range(0, i):
        a[i][j] = 0

cv2.imwrite('pictures/ris.png', a)
img = cv2.imread('pictures/ris.png', 0)
imgshow(img)
# cv2.imshow('image',img)
print(img)
```



```
[[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1]]
```

Рисунок 4 – Пример 4

## Задание 1.5

Вывести свойства матрицы изображения на экран.

```
In [6]: img = cv2.imread('pictures/bliss.jpg', 0) # Считываем изображение с файла
        # cv2.imshow('image', img)
        print(type(img))
        print(img.shape) # Разрешение, каналы RGB
        print(img.size) # Общее количество пикселей
        print(img.dtype) # Тип данных изображения

<class 'numpy.ndarray'>
(1080, 1920)
2073600
uint8
```

## Задание 1.6

Определить с помощью функции `print img.shape` максимальное число пикселей по ширине и высоте изображения. Выбрать координаты так, чтобы они не выходили за пределы размеров изображения. Задать координату по горизонтали равной сумме номера по списку группы плюс 70, по вертикали равной сумме номера по списку группы плюс 50.

```
In [7]: img = cv2.imread('pictures/bliss.jpg', 1)
        print(img.shape)
        px = img[100, 150]

(1080, 1920, 3)
```

```
In [8]: # доступ только к синему пикселю
        blue = img[100, 150, 0]
        print(blue)

238
```

```
In [9]: # Изменить значения пикселей, интенсивности B, G, R цветов
        img[100, 150] = [105, 139, 185]
        print(img[100, 150])

[105 139 185]
```

```
In [10]: # доступ к красному пикселю, 2 - флаг цвета
        img.item(100, 150, 2)
        img.itemset((100, 150, 2), 100)
```

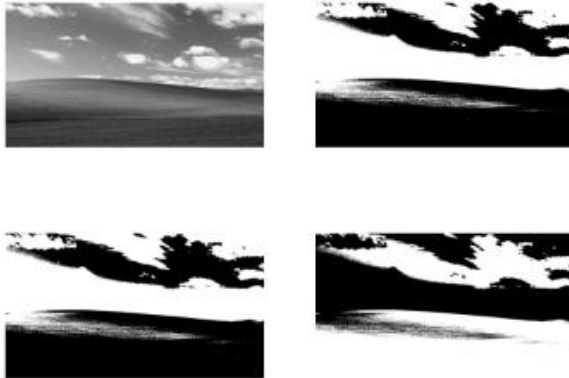
Рисунок 5 – Пример 5 и 6

## Задание 1.7

Считать файл полноцветного изображения, создать для него два места в окне в ширину и два места в высоту. Преобразовать матрицу цветного изображения в полутоновое, из него, используя функцию `cv2.threshold`, получить бинарное монохромное изображение. Из бинарного монохромного изображения получить его негатив.

```
In [11]: img = cv2.imread('pictures/bliss.jpg', 0) # считываем изображение с файла
          imag = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

          # Оригинальное изображение выводится в первое окно:
          plt.subplot(221)
          plt.imshow(imag)
          plt.axis("off")
          gray_img = cv2.imread('bliss.jpg',0) # полутоновое изображение
          im_bw = cv2.threshold(gray_img, 128, 255, cv2.THRESH_BINARY)[1]
          plt.subplot(222)
          # Выведем бинарное монохромное изображение во второе окно
          plt.imshow(im_bw, 'gray')
          plt.axis("off")
          im_bwa = cv2.threshold(imag, 128, 255,
                                cv2.THRESH_BINARY)[1]
          # Выводим монохромное цветное изображение в третье окно
          plt.subplot(223)
          plt.imshow(im_bwa)
          plt.axis("off")
          im_bwb = cv2.threshold(gray_img, 128, 255,
                                cv2.THRESH_BINARY_INV)[1]
          plt.subplot(224) # Выведем инвертированное изображение
          plt.imshow(im_bwb, 'gray')
          plt.axis("off")
          plt.show()
```



## Задание 1.8

На заданном изображении выделить его характерный участок.

```
In [12]: img = cv2.imread('pictures/murk.jpg', 1) # считываем изображение с файла
          image = cv2.rectangle(img, (423, 254), (508, 294), (0, 0, 255), 2)
          # cv2.imshow('selected', image)
          imshow(img)
```



Рисунок 6 – Пример 7 и 8

## Задание 1.9

Уменьшить заданное изображение и вывести на печать матрицу уменьшенного изображения.

```
In [13]: img = cv2.imread('pictures/bliss.jpg', 0) # считываем изображение с файла
final_wide = 200
r = float(final_wide) / img.shape[1]
# уменьшаем изображение до подготовленных размеров
dim=(final_wide, int(img.shape[0]*r))
resized=cv2.resize(img,dim,interpolation=cv2.INTER_AREA)
cv2.imshow("Resize image", resized)
print(resized.shape)
print(resized)

img = cv2.imread('bliss.jpg', 0)
# cv2.imshow('image', img)
print(img)

(112, 200)
[[124 119 126 ... 231 230 228]
 [215 188 190 ... 227 227 229]
 [240 244 244 ... 225 224 227]
 ...
 [ 55  59  58 ...  51  54  54]
 [ 54  57  55 ...  53  51  53]
 [ 50  55  52 ...  47  50  48]]
[[122 120 120 ... 229 228 228]
 [117 116 117 ... 229 228 228]
 [119 120 120 ... 229 228 228]
 ...
 [ 59  58  57 ...  48  46  44]
 [ 59  60  60 ...  47  45  43]
 [ 59  60  62 ...  47  44  42]]
```

## Задание 1.10

Считать цветное изображение, конвертировать его в полутоновое, затем получить негатив полутонового изображения.

```
In [14]: img = cv2.imread('pictures/bliss.jpg')
img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
imgshow(img)
# Функция инвертирования изображения
img = cv2.bitwise_not(img)
# cv2.imshow('img', img)
imgshow(img)
```



Рисунок 7 – Пример 9 и 10



## Индивидуальное задание.

### Задание

Выделить прямоугольником на изображении всех людей красным цветом и объекты - зеленым.

```
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)
    # Show the image
    plt.imshow(image)
    # remove the axis / ticks for a clean looking image
    plt.xticks([])
    plt.yticks([])
    plt.show()
```

```
In [2]: image = cv2.imread('pictures/people.jpg')
print(image.shape)
```

(533, 800, 3)

```
In [3]: image = cv2.rectangle(image,(406,194),(438,236),(0,0,255),3)
image = cv2.rectangle(image,(96,207),(135,243),(0,0,255),3)
image = cv2.rectangle(image,(244,181),(306,240),(0,0,255),3)
image = cv2.rectangle(image,(517,169),(555,224),(0,0,255),3)
image = cv2.rectangle(image,(153,193),(205,255),(0,0,255),3)
image = cv2.rectangle(image,(675,270),(699,313),(0,255,0),3)
image = cv2.rectangle(image,(58,222),(86,253),(0,255,0),3)
imgshow(image)
```

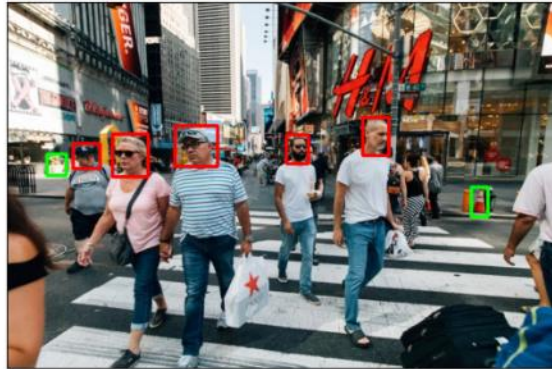


Рисунок 8 – Индивидуальное задание

**Вывод:** В результате выполнения работы были исследованы основные функции OpenCV для обработки цифровых изображений.