МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

Федеральное государственное автономное

образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

«Основы цифровой обработки изображений в OpenCv»

Отчет по лабораторной работе № 7

по дисциплине «Технологии распознавания образов»

Выполнил студент группы ПИЖ-б-о-21-1

Халимендик Я. Д. « » 2023г.

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

Ход работы:

Задание 1.1. Считать файл полноцветного изображения cat.jpg, создать для него матрицу изображения, затем вывести сначала полутоновое, затем цветное изображение на экран. Перед выполнением задания получить согласно номеру в списке группы свой файл с изображением.

Решение:

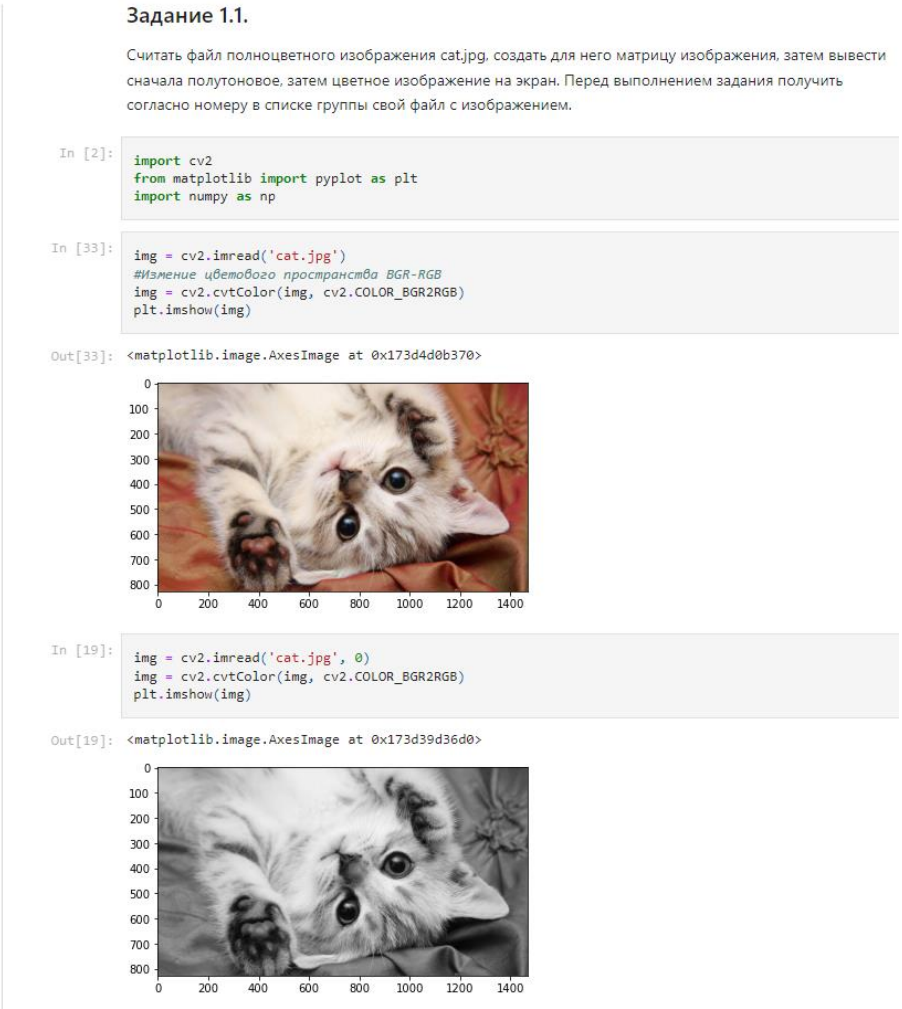


Рисунок 1 – Решение задачи

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.

Решение:

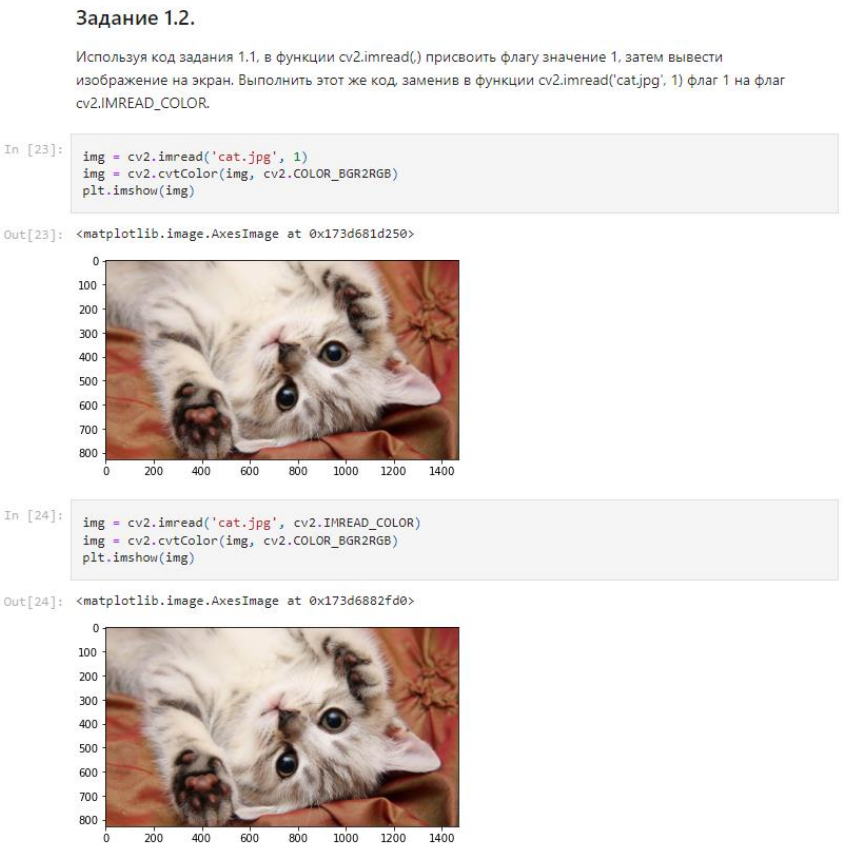


Рисунок 2 – Решение задачи

Задание 1.3. Сформировать матрицу изображения, записать ее в файл с расширением png. Изображение, записанное в этом файле, вывести на экран.

Решение:

Рисунок 3 – Решение задачи

Задание 1.4. Сформировать матрицу, у которой выше диагонали единицы, а ниже – нули, записать ее в файл, затем считать файл и вывести на экран.

Решение:

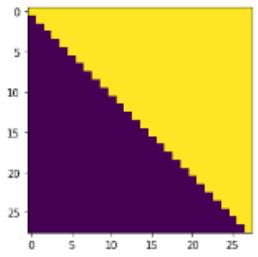
Задание 1.4.

Сформировать матрицу, у которой выше диагонали единицы, а ниже – нули, записать ее в файл, затем считать файл и вывести на экран. Строим массив 28*28:

In [3]:
```
# Строим массив 28x28, заполненный нулями
n = 28
a = np.ones([28, 28])

# Заполняем диагональ массива
for i in range(n):
    a[i][i] = 1

# Заполняем массив выше диагонали
for i in range(n):
    for j in range(0, i):
        a[i][j] = 0


cv2.imwrite('zd4.png', a) # Записываем изображение в файл
img = cv2.imread('zd4.png', 0) # Считываем изображение с файла

print(img)
plt.imshow(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]]
```

Out[3]: <matplotlib.image.AxesImage at 0x29fc2d52b20>



Задание 1.5.

Рисунок 4 – Решение задачи

Задание 1.5. Вывести свойства матрицы изображения на экран.

Решение:

Задание 1.5.

Вывести свойства матрицы изображения на экран.

```
img = cv2.imread('cat.jpg', 0)
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow (img)

print(type(img))#Класс: <class 'numpy.ndarray'>
print (img.shape)# Кортеж числа строк и столбцов (разрешение), и каналов RGB: (427, 500, 3)
print (img.size)# Общее количество пикселей : 640500
print (img.dtype)# Тип данных изображения : uint8
```

```
<class 'numpy.ndarray'>
(827, 1470, 3)
3647070
uint8
```

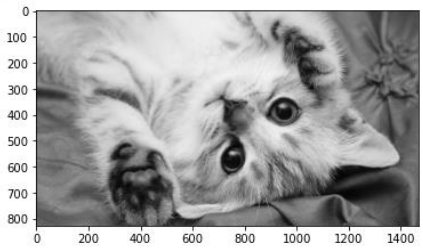


Рисунок 5 – Решение задачи

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 номе- 15 ра по списку группы плюс 70, по вертикали равной сумме номера по списку группы плюс 50.

Решение:

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.
Изменить значения пикселей, интенсивности B, G, R цветов, взяв интенсивности первого упражнения и прибавив к ним номер по списку группы.

```
img = cv2.imread('cat.jpg')
print("Максимальное число пикселей:")
print(img.shape)#определение размера матрицы

# номер по списку 29 =>
#[70; 50] + 29 = [99; 79]
px = img[99, 79] # доступ к пикселю цветного изображения с координатами 100, 150
print("\nЗначение пикселя с координатами 99, 79 (B, G, R):")
print(px)

blue = img[99, 79, 0] # доступ только к синему пикселю с координатами (99, 79)
print("\nСиний пиксель:")
print(blue)

# номер по списку 29
#[199, 205, 212] + 29 = [228, 234, 241]
img[99, 79] = [228, 234, 241]
print("\nИзмененные значения: ")
print(img[99, 79])

# Доступ с помощью функций array.item() и array.itemset()
print("\nКрасный пиксель:")
print(img.item(99, 79, 2))
# изменение значения красного пикселя
img.itemset((99, 79, 2),212)
print("Измененное значение пикселя:")
print(img.item(99, 79, 2))
```

```
Максимальное число пикселей:
(827, 1470, 3)

Значение пикселя с координатами 99, 79 (B, G, R):
[199 205 212]

Синий пиксель:
199

Измененные значения:
[228 234 241]

Красный пиксель:
241
Измененное значение пикселя:
212
```

Рисунок 6 – Решение задачи

Задание 1.7. Считать файл полноцветного изображения cat.jpg, 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.

Решение:

Задание 1.7.

Считать файл полноцветного изображения cat.jpg, 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 [17]:

```
img = cv2.imread("cat.jpg")
imag = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

# Оригинальное изображение выводится в первое окно:
plt.subplot(221)
plt.imshow(imag)
plt.axis('off')

# полутоновое ч/б изображение
gray_img = cv2.imread("cat.jpg", 0)
im_bw = cv2.threshold(gray_img, 128, 255, cv2.THRESH_BINARY)[1]

# выведем бинарное монохромное изображение во второе окно
plt.subplot(222)
plt.imshow(im_bw,'gray')
plt.axis("off")

# полутоновое цветное изображение
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")

# инвертирует монохромное изображение
im_bwb = cv2.threshold(gray_img, 128, 255, cv2.THRESH_BINARY_INV)[1]
plt.subplot(224)
plt.imshow(im_bwb, 'gray')
plt.axis("off")
plt.show()
```

Рисунок 7 – Решение задачи

Задание 1.8. На заданном изображении выделить его характерный участок.

Решение:

Задание 1.8.

На заданном изображении выделить его характерный участок.

In [5]:
```
# выделим лапку кота
img = cv2.imread("cat.jpg", 1)
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
image = cv2.rectangle(img,(280,500),(560,790),(0,0,255), 2)
plt.axis("off")
plt.imshow(image)
```

Out[5]: <matplotlib.image.AxesImage at 0x29fe2e62b20>

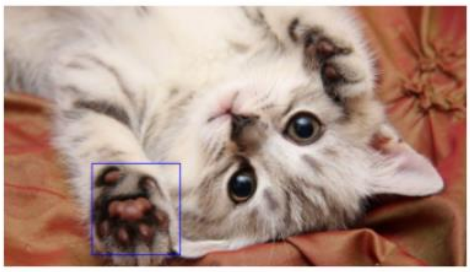


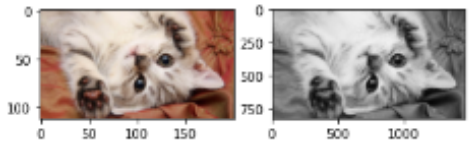
Рисунок 8 – Решение задачи

Задание 1.9. Уменьшить заданное изображение и вывести на печать матрицу уменьшенного изображения. Нам надо сохранить соотношение сторон, чтобы изображение не исказилось при уменьшении. Для этого необходимо вычислить коэффициент уменьшения стороны.

Решение:

```
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
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)

plt.subplot(221)
plt.imshow(resized)
print(resized.shape)
print(resized)

img = cv2.imread('cat.jpg', 0)
plt.subplot(222)
plt.imshow(cv2.cvtColor(img,cv2.COLOR_BGR2RGB))
print(img)
plt.show()
```

```
(112, 200, 3)
[[[192 174 166]
  [191 173 165]
  [190 171 159]
  ...
  [143  49  43]
  [143  49  47]
  [145  50  48]]

 [[195 180 171]
  [194 179 171]
  [198 179 166]
  ...
  [143  49  47]
  [142  51  48]
  [142  51  48]]

 [[195 181 172]
  [195 180 171]
  [199 182 169]
  ...
  [143  50  46]
  [145  49  46]
  [147  51  47]]

 ...

 [[159  79  52]
  [159  74  49]
  [162  78  52]
  ...
  [165 111  85]
  [138  95  69]
  [105  69  38]]

 [[164  85  58]
  [158  80  52]
  [160  74  48]
  ...
  [162 111  83]
  [130  91  66]
  [112  72  41]]

 [[165  87  60]
  [167  89  62]
  [164  81  55]
  ...
  [164 112  85]
  [119  83  58]
  [110  66  37]]]
[[176 176 176 ...  78  78  78]
 [176 176 176 ...  79  79  79]
 [177 177 177 ...  80  80  80]
 ...
 [ 97 101  72 ...  80  79  85]
 [ 77  89 161 ...  82 101  76]
 [135 116 126 ...  85 111 102]]
```



Вывод о размерах можем сделать, судя по размерной сетке

Рисунок 9 – Решение задачи

Задание 1.10. Считать цветное изображение, конвертировать его в полутоновое, затем получить негатив полутонового изображения.

Решение:

Задание 1.10.

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

```
img = cv2.imread('cat.jpg', 1)
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

img = cv2.bitwise_not(img)
plt.axis("off")
plt.imshow(img)
```

```
<matplotlib.image.AxesImage at 0x173dae50d60>
```

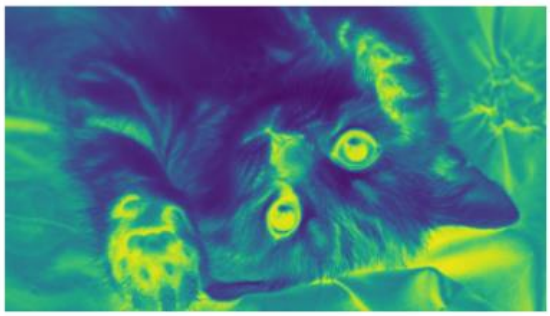


Рисунок 10 – Решение задачи

Индивидуальная задача:

Считать цветное изображение и выполнить вывод:

1. Изображение в формате RGB и BGR
2. Инверсия изображений из пункта 1
3. Монохромное цветное изображение
4. Размытое изображение
5. Выделить произвольный элемент

In [42]:

```
import cv2
from matplotlib import pyplot as plt
```

In [48]:

```
img_bgr = cv2.imread('image.jpg')

# Измение цветового пространства BGR-RGB
img_rgb = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2RGB)
# Инверсия изображения
img_bgr_inv = cv2.bitwise_not(img_bgr)
img_rgb_inv = cv2.bitwise_not(img_rgb)
# Монохромное цветное изображение
img_bgr_bwa = cv2.threshold(img_bgr, 128, 255, cv2.THRESH_BINARY)[1]
img_rgb_bwa = cv2.threshold(img_rgb, 128, 255, cv2.THRESH_BINARY)[1]
# Размытие
blur_bgr = cv2.GaussianBlur(img_bgr, (51, 51), 0)
blur_rgb = cv2.GaussianBlur(img_rgb, (51, 51), 0)
```

In [49]:

```
fig, axs = plt.subplots(4, 2)

fig.set_figheight(15)
fig.set_figwidth(15)

axs[0, 0].imshow(img_bgr)
axs[0, 0].set_title('BGR')
axs[0, 1].imshow(img_rgb)
axs[0, 1].set_title('RGB')
axs[1, 0].imshow(img_bgr_inv)
axs[1, 0].set_title('Inverted BGR')
axs[1, 1].imshow(img_rgb_inv)
axs[1, 1].set_title('Inverted RGB')
axs[2, 0].imshow(img_bgr_bwa)
axs[2, 0].set_title('Monochrome BGR')
axs[2, 1].imshow(img_rgb_bwa)
axs[2, 1].set_title('Monochrome RGB')
axs[3, 0].imshow(blur_bgr)
axs[3, 0].set_title('Blur BGR')
axs[3, 1].imshow(blur_rgb)
axs[3, 1].set_title('Blur RGB')


plt.show()
```

Рисунок 11 – Решение задачи

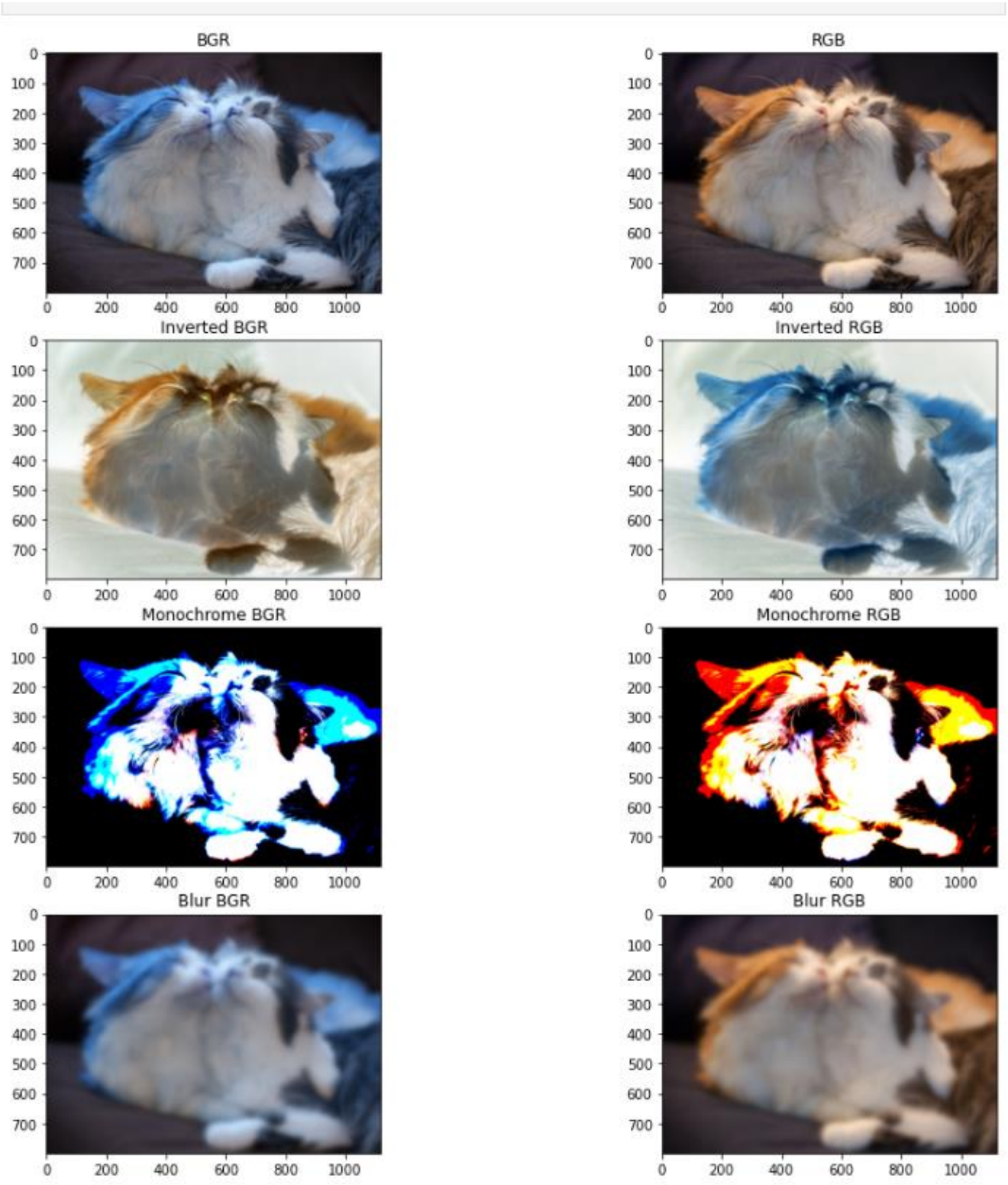
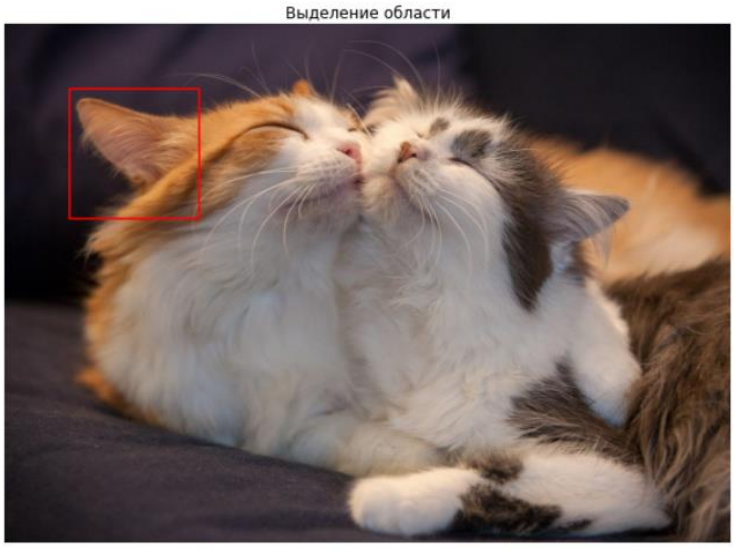


Рисунок 12 – Решение задачи

In [55]:

```
image = cv2.rectangle(img_rgb,(100,100),(300,300),(255,0,0), 2)
plt.figure(figsize=(10, 10))
plt.axis("off")
plt.title("Выделение области")
plt.imshow(image)
```

Out[55]: <matplotlib.image.AxesImage at 0x29d1a6b1610>



e feedback

Рисунок 13 – Решение задачи