SINGULARIS LAB

Введение в компьютерное зрение. Практика

Катаев Александр
Ведущий инженер-программист, к.т.н.
Алексеев Алексей
Инженер-программист
Singularis Lab, Ltd.

Необходимые импорты

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

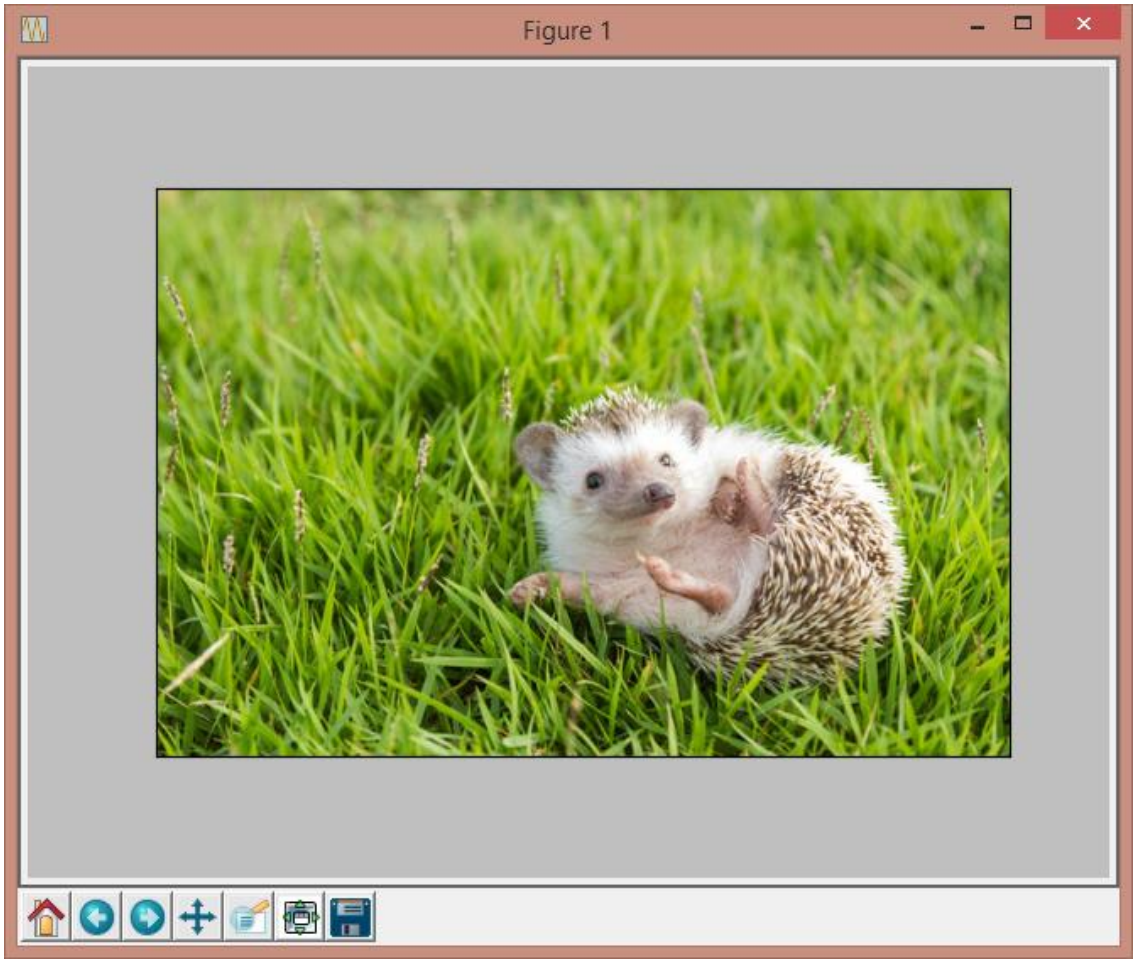
Открытие изображения

```
img = cv2.imread(image_name) # чтение файла
print("image size is: h:{0} w:{1}
c:{2}".format(*img.shape)) # img.shape содержит
ширину, высоту и количество каналов изображения

# matplotlib
plt.imshow(img[:, :, ::-1]) # добавить
изображение
plt.xticks([]), plt.yticks([]) # убрать значения
на осях
plt.show() # вывести на экран
# OpenCV
cv2.imshow('image', img) # вывести на экран
cv2.waitKey() # ожидать нажатия клавиши
```

Открытие изображения

```
img = cv2.imread(image_name) # чтение файла
print("image size is: h:{0} w:{1}
c:{2}".format(*img.shape)) # img.shape содержит
ширину, высоту и количество каналов изображения

# matplotlib
plt.imshow(img[:, :, ::-1]) # добавить
изображение
plt.xticks([]), plt.yticks([]) # убрать значения
на осях
plt.show() # вывести на экран
# OpenCV
cv2.imshow('image', img) # вывести на экран
cv2.waitKey() # ожидать нажатия клавиши
```

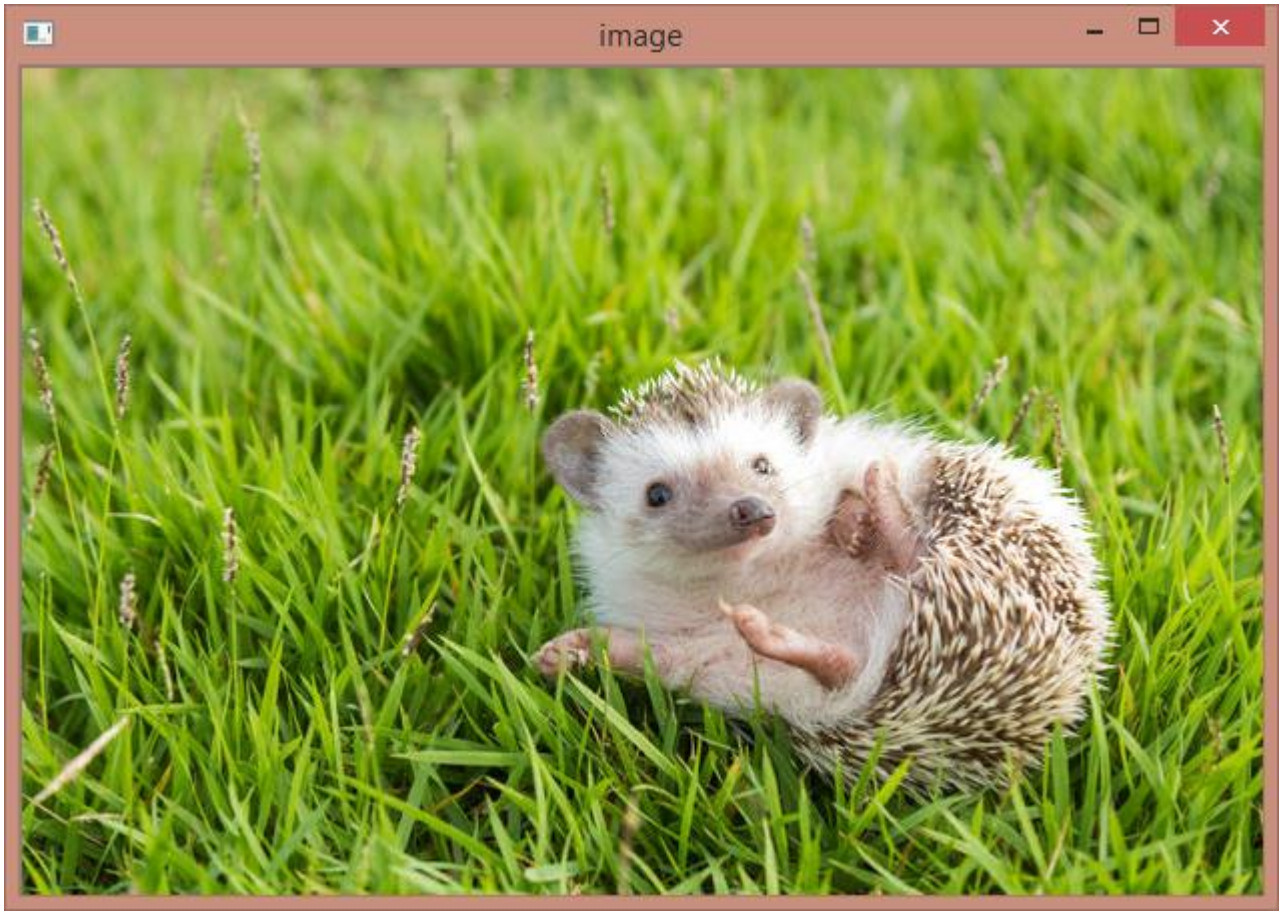
Открытие изображения



Открытие изображения

```
img = cv2.imread(image_name) # чтение файла
print("image size is: h:{0} w:{1}
c:{2}".format(*img.shape)) # img.shape содержит
ширину, высоту и количество каналов изображения

# matplotlib
plt.imshow(img[:, :, ::-1]) # добавить
изображение
plt.xticks([]), plt.yticks([]) # убрать значения
на осях
plt.show() # вывести на экран
# OpenCV
cv2.imshow('image', img) # вывести на экран
cv2.waitKey() # ожидать нажатия клавиши
```

Открытие изображения



Работа с камерой

```
# выбрать камеру
cap = cv2.VideoCapture(0)
while True:
    # прочитать текущий кадр
    _, frame = cap.read()
    # вывод на экран изображения
    cv2.imshow("frame", frame)
    # ждать 30 милисекунд
    pressed_key = cv2.waitKey(30)
    # прервать, если нажата ESC (код 27)
    if pressed_key == 27:
        break
```

Работа с камерой

```
# выбрать камеру
cap = cv2.VideoCapture(0)
while True:
    # прочитать текущий кадр
    _, frame = cap.read()
    # вывод на экран изображения
    cv2.imshow("frame", frame)
    # ждать 30 милисекунд
    pressed_key = cv2.waitKey(30)
    # прервать, если нажата ESC (код 27)
    if pressed_key == 27:
        break
```

Trackbar для управления параметрами

```
# пустое изображение, размер 400x400 и 1 канал, каждый пиксел занимает 8 бит [0..255]
img = np.zeros((400, 400, 1), np.uint8)
# создаем окно с именем Image
cv2.namedWindow('Image')
# создаем трэкбар, имя трэкбара, имя окна, мин макс значения, функция обработчик
cv2.createTrackbar('Value', 'Image', 0, 255, on_change)
while True:
    # получаем значение трэкбара
    val = cv2.getTrackbarPos('Value', 'Image')
    # выставляем все писклелы изображения в значение с трэкбара
    img[:] = val
    cv2.imshow('Image', img)
    cv2.waitKey(10)
```

Trackbar для управления параметрами

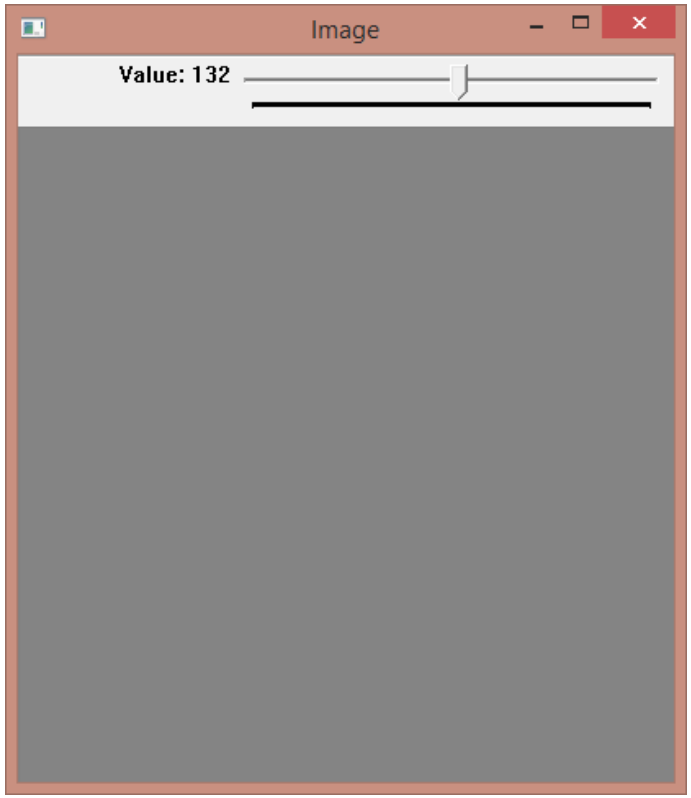
```
# пустое изображение, размер 400x400 и 1 канал, каждый пиксел занимает 8 бит [0..255]
img = np.zeros((400, 400, 1), np.uint8)
# создаем окно с именем Image
cv2.namedWindow('Image')
# создаем трэкбар, имя трэкбара, имя окна, мин макс значения, функция обработчик
cv2.createTrackbar('Value', 'Image', 0, 255, on_change)
while True:
    # получаем значение трэкбара
    val = cv2.getTrackbarPos('Value', 'Image')
    # выставляем все писклелы изображения в значение с трэкбара
    img[:] = val
    cv2.imshow('Image', img)
    cv2.waitKey(10)
```

Trackbar для управления параметрами

```
# пустое изображение, размер 400x400 и 1 канал, каждый пиксел занимает 8 бит [0..255]
img = np.zeros((400, 400, 1), np.uint8)
# создаем окно с именем Image
cv2.namedWindow('Image')
# создаем трэкбар, имя трэкбара, имя окна, мин макс значения, функция обработчик
cv2.createTrackbar('Value', 'Image', 0, 255, on_change)
while True:
    # получаем значение трэкбара
    val = cv2.getTrackbarPos('Value', 'Image')
    # выставляем все писклелы изображения в значение с трэкбара
    img[:] = val
    cv2.imshow('Image', img)
    cv2.waitKey(10)
```

Trackbar для управления параметрами

```
# обработчик изменения положения трэкбара
def on_change(x):
    pass
```



Обработка мыши

```
drawing = False
curr_x, curr_y = 0, 0
mouse_image = np.zeros((400, 400, 1), np.uint8)
def mouse_callback(e, x, y, flags, param):
    global curr_x, curr_y, drawing, mouse_image
    if e == cv2.EVENT_LBUTTONDOWN:
        drawing = True
        curr_x, curr_y = x, y
    elif e == cv2.EVENT_MOUSEMOVE:
        if drawing:
            mouse_image = np.zeros((400, 400, 1), np.uint8)
            cv2.rectangle(mouse_image, (curr_x, curr_y), (x,
y), 255, 1)
    elif e == cv2.EVENT_LBUTTONUP:
        drawing = False
```

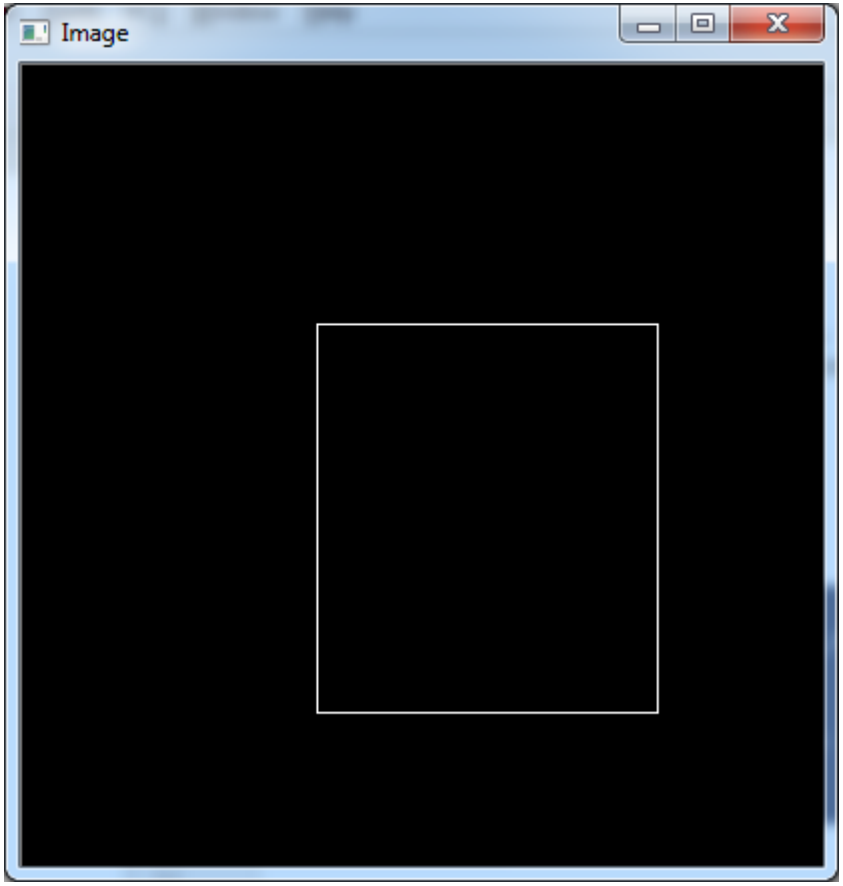
Обработка мыши

```
drawing = False
curr_x, curr_y = 0, 0
mouse_image = np.zeros((400, 400, 1), np.uint8)
def mouse_callback(e, x, y, flags, param):
    global curr_x, curr_y, drawing, mouse_image
    if e == cv2.EVENT_LBUTTONDOWN:
        drawing = True
        curr_x, curr_y = x, y
    elif e == cv2.EVENT_MOUSEMOVE:
        if drawing:
            mouse_image = np.zeros((400, 400, 1), np.uint8)
            cv2.rectangle(mouse_image, (curr_x, curr_y), (x,
y), 255, 1)
    elif e == cv2.EVENT_LBUTTONUP:
        drawing = False
```

Обработка мыши

```
cv2.namedWindow('Image')
cv2.setMouseCallback('Image', mouse_callback)
while True:

    cv2.imshow('Image', mouse_image)
    pressed_key = cv2.waitKey(30)
    if pressed_key == 27:
        break
```

Обработка мыши



Работа с каналами

```
plt.imshow(img[:, :, ::-1])
...
img = np.zeros((400, 400, 1), np.uint8)
...
channels_RGB = ['B', 'G', 'R']
img = cv2.imread(image_name)
for index in range(3):
    plt.subplot(3, 3, index + 1),
plt.imshow(img[:, :, index], 'gray')
    plt.title(channels_RGB[index])
    plt.xticks([]), plt.yticks([])
plt.show()
```

Работа с каналами

```
plt.imshow(img[:, :, ::-1])
...
img = np.zeros((400, 400, 1), np.uint8)
...
channels_RGB = ['B', 'G', 'R']
img = cv2.imread(image_name)
for index in range(3):
    plt.subplot(3, 3, index + 1),
plt.imshow(img[:, :, index], 'gray')
    plt.title(channels_RGB[index])
    plt.xticks([]), plt.yticks([])
plt.show()
```

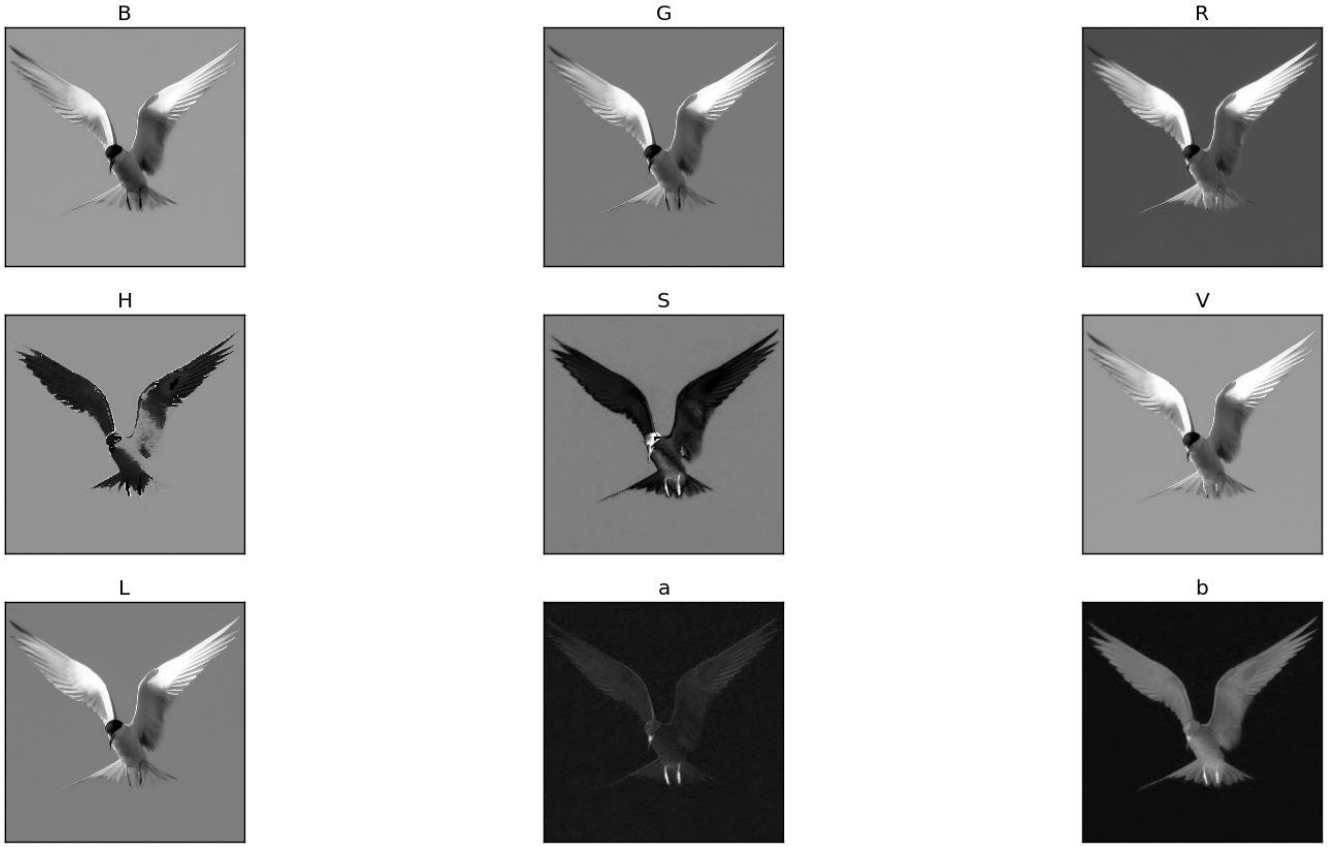
Работа с каналами

```
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
for index in range(3):
    plt.subplot(3, 3, 3 + index + 1), plt.imshow(hsv[:,
:, index], 'gray')
    plt.title(channels_HSV[index])
    plt.xticks([]), plt.yticks([])
Lab = cv2.cvtColor(img, cv2.COLOR_BGR2Lab)
for index in range(3):
    plt.subplot(3, 3, 6 + index + 1), plt.imshow(Lab[:,
:, index], 'gray')
    plt.title(channels_Lab[index])
    plt.xticks([]), plt.yticks([])
plt.show()
```

Работа с каналами

```
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
for index in range(3):
    plt.subplot(3, 3, 3 + index + 1), plt.imshow(hsv[:,
:, index], 'gray')
    plt.title(channels_HSV[index])
    plt.xticks([]), plt.yticks([])
Lab = cv2.cvtColor(img, cv2.COLOR_BGR2Lab)
for index in range(3):
    plt.subplot(3, 3, 6 + index + 1), plt.imshow(Lab[:,
:, index], 'gray')
    plt.title(channels_Lab[index])
    plt.xticks([]), plt.yticks([])
plt.show()
```

Работа с каналами



Цветовая сегментация

```
img = cv2.imread(image_name)
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
lower_blue = np.array([100, 80, 80])  # 120 - blue
upper_blue = np.array([130, 255, 255])

mask = cv2.inRange(hsv, lower_blue, upper_blue)

result_img = cv2.bitwise_and(img, img, mask=(255 - mask))
plt.subplot(1, 3, 1), plt.imshow(mask, 'gray')
plt.title('mask')
plt.xticks([]), plt.yticks([])
plt.subplot(1, 3, 2), plt.imshow(img[:, :, ::-1])
plt.title('source')
plt.xticks([]), plt.yticks([])
plt.subplot(1, 3, 3), plt.imshow(result_img[:, :, ::-1])
plt.title('result')
plt.xticks([]), plt.yticks([])
plt.show()
```

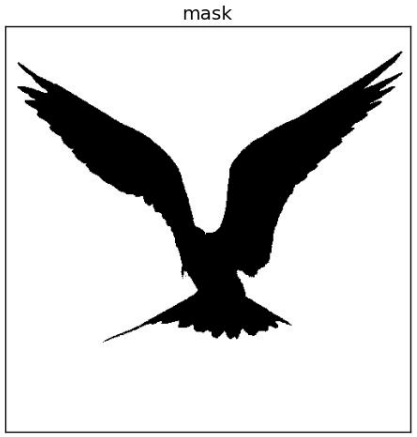
Цветовая сегментация

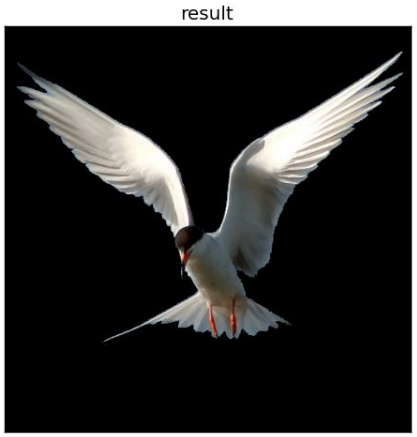
```
img = cv2.imread(image_name)
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
lower_blue = np.array([100, 80, 80])  # 120 - blue
upper_blue = np.array([130, 255, 255])

mask = cv2.inRange(hsv, lower_blue, upper_blue)

result_img = cv2.bitwise_and(img, img, mask=(255 - mask))
plt.subplot(1, 3, 1), plt.imshow(mask, 'gray')
plt.title('mask')
plt.xticks([]), plt.yticks([])
plt.subplot(1, 3, 2), plt.imshow(img[:, :, ::-1])
plt.title('source')
plt.xticks([]), plt.yticks([])
plt.subplot(1, 3, 3), plt.imshow(result_img[:, :, ::-1])
plt.title('result')
plt.xticks([]), plt.yticks([])
plt.show()
```

Цветовая сегментация

```
img = cv2.imread(image_name)
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
lower_blue = np.array([100, 80, 80])  # 120 - blue
upper_blue = np.array([130, 255, 255])

mask = cv2.inRange(hsv, lower_blue, upper_blue)

result_img = cv2.bitwise_and(img, img, mask=(255 - mask))
plt.subplot(1, 3, 1), plt.imshow(mask, 'gray')
plt.title('mask')
plt.xticks([]), plt.yticks([])
plt.subplot(1, 3, 2), plt.imshow(img[:, :, ::-1])
plt.title('source')
plt.xticks([]), plt.yticks([])
plt.subplot(1, 3, 3), plt.imshow(result_img[:, :, ::-1])
plt.title('result')
plt.xticks([]), plt.yticks([])
plt.show()
```

Цветовая сегментация



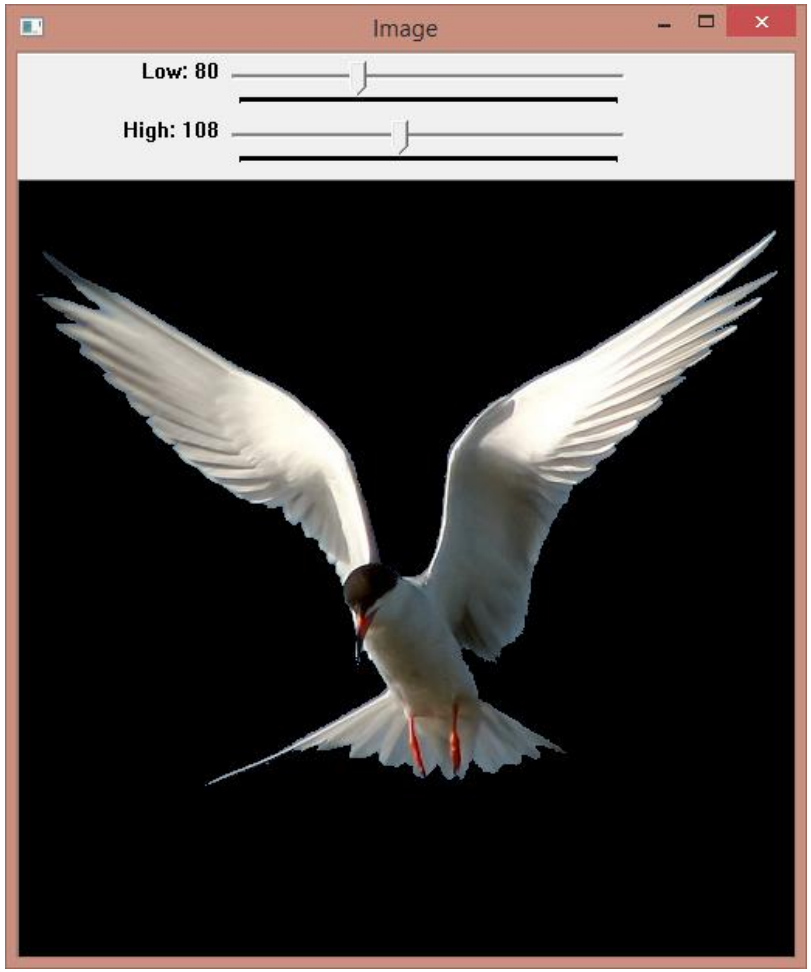


Цветовая сегментация. Задание

Задавать пороговые значения цвета с помощью трэкбаров и брать изображение с веб-камеры

```
cv2.createTrackbar('Value',
'Image', 0, 255, on_change)
```

Цветовая сегментация. Задание



Цветовая сегментация. Задание

```
cap = cv2.VideoCapture(0)
cv2.namedWindow('Image')
cv2.createTrackbar('Low', 'Image', 0, 255, on_change)
cv2.createTrackbar('High', 'Image', 0, 255, on_change)
while True:
    low = cv2.getTrackbarPos('Low', 'Image')
    high = cv2.getTrackbarPos('High', 'Image')
    _, frame = cap.read()
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    lower_blue = np.array([low, 80, 80])
    upper_blue = np.array([high, 255, 255])

    mask = cv2.inRange(hsv, lower_blue, upper_blue)
    result_img = cv2.bitwise_and(frame, frame, mask=(255 -
mask))
    cv2.imshow('Image', result_img)
    cv2.waitKey(10)
```

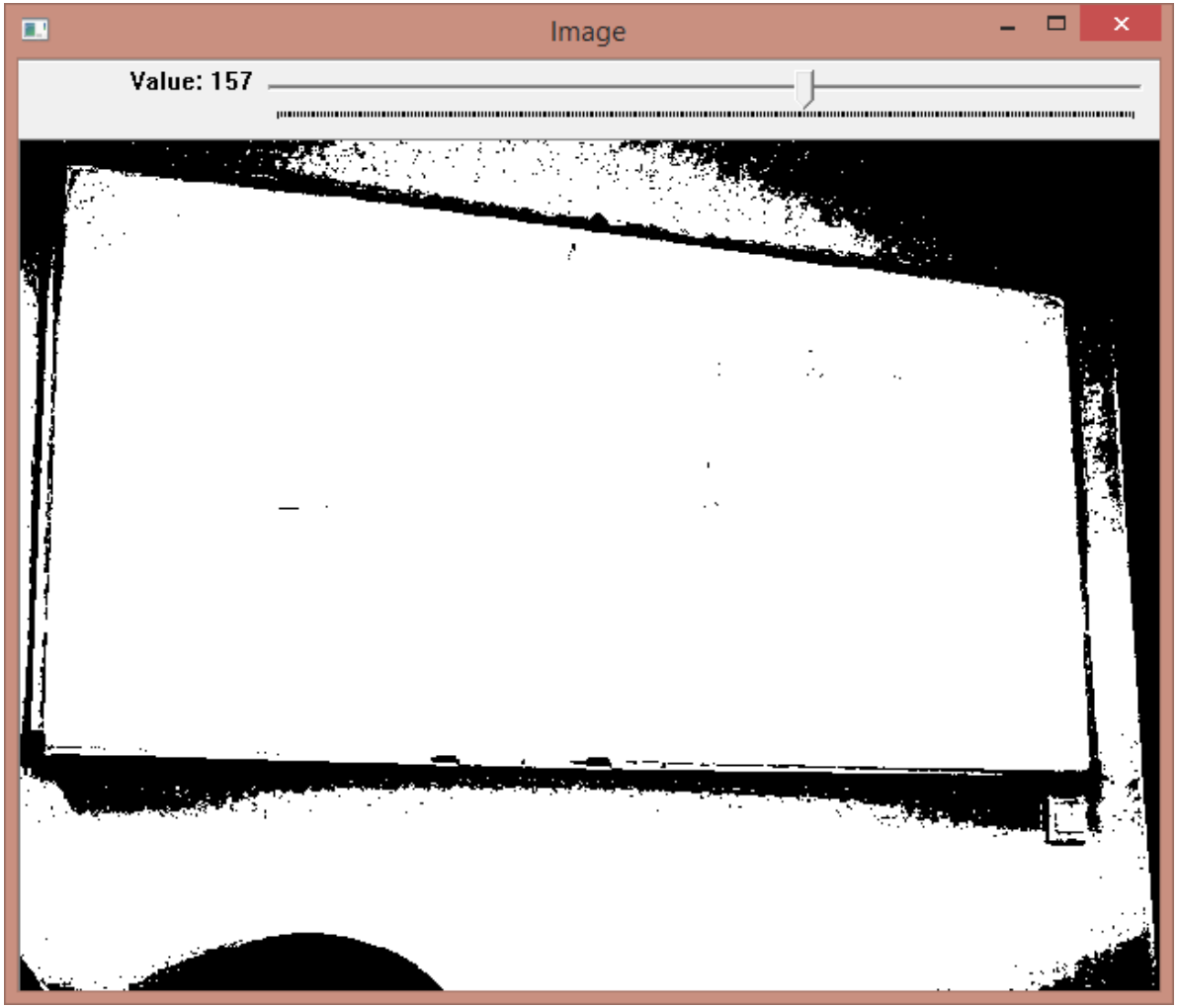
Пороговое разделение. Задание

Применить пороговое преобразование к выводу с веб-камеры, порог должен задаваться с помощью трэкбара

```
cap = cv2.VideoCapture(0)
while True:
    _, frame = cap.read()
    cv2.imshow("frame", frame)
    pressed_key = cv2.waitKey(30)
    if pressed_key == 27:
        break
```

```
ret, thresh1 = cv2.threshold(img, _value, 255, cv2.THRESH_BINARY)
```

Пороговое разделение. Задание



Спасибо за внимание

Александр Катаев

- alexander.kataev@singularis-lab.com

Алексей Алексеев

- aleksey.alekseev@singularis-lab.com

https://www.singularis-lab.com/

https://www.linkedin.com/company/singularis-lab-llc

http://habrahabr.ru/company/singularis

http://vk.com/singularis_lab