Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования «Кубанский государственный университет»

Кафедра вычислительных технологий

**ОТЧЕТ**

о выполнении лабораторной работы №2

по дисциплине «Алгоритмы цифровой обработки мультимедиа»

Выполнил: ст. гр. ФИ 41/2

Миков Н.С.

Проверил: ст. преподаватель

Крамаренко А.А.

Краснодар

2025

**Задание**

Задание 1 Прочитать изображение с камеры и перевести его в формат HSV.

Задание 2 Применить фильтрацию изображения с помощью команды inRange и оставить только красную часть, вывести получившееся изображение на экран(treshold), выбрать красный объект и потестировать параметры фильтрации, подобрав их нужного уровня.

Задание 3 Провести морфологические преобразования (открытие и закрытие) фильтрованного изображения, вывести результаты на экран, посмотреть смысл подобного применения операций erode и dilate.

Задание 4 Найти моменты на полученном изображении 1 первого порядка, найти площадь объекта.

Задание 5 На основе анализа площади объекта найти его центр и построить черный прямоугольник вокруг объекта. Сделать так, чтобы на видео выводился полученный черный прямоугольник, причем на новом кадре новый.

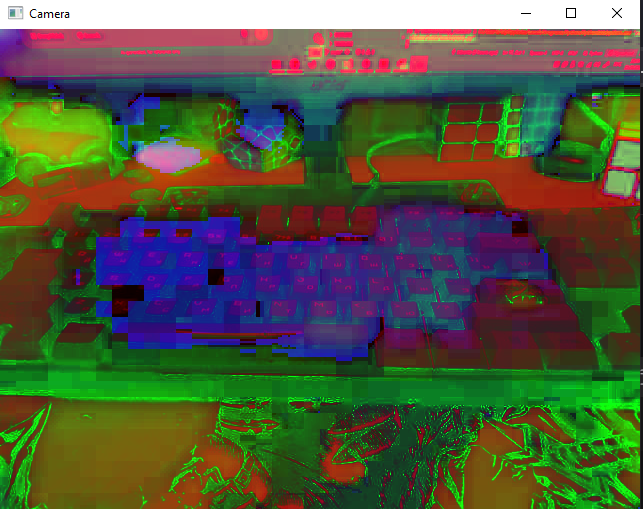
**Ход работы**

**Задание 1** Прочитать изображение с камеры и перевести его в формат HSV.

Было сделано ранее, нужно использовать метод библиотеки cv2, чтобы перевести кадр из BGR в HSV и отобразить его.

hsv = cv2.cvtColor(img, cv2.COLOR\_BGR2HSV) #COLOR\_BGR2RGB, COLOR\_BGR2RGB

cv2.imshow('Camera', hsv)



**Задание 2** Применить фильтрацию изображения с помощью команды inRange и оставить только красную часть, вывести получившееся изображение на экран(treshold), выбрать красный объект и потестировать параметры фильтрации, подобрав их нужного уровня.

Для выполнения этого задания нужно определить верхние и нижние границы нужно цвета, после чего можно будет сделать маску.

low\_redor = np.array([0, 133, 133])

up\_redor = np.array([5, 255, 255])

low\_redpur = np.array([175, 133, 133])

up\_redpur = np.array([180, 255, 255])

В cv2 значения по S и V представлены не как проценты и не отрезок 0-1, а диапазон от 0 до 255. Первый параметр – Hue отвечает за цветовой тон, с помощью него можно выбрать два отрезка с верхними и нижними пределами, где можно найти красный цвет (на палитре идеально видно, всегда самые левые и правые концы). S – насыщенность, чем ближе значение к 0, тем цвет более серый, поэтому для нижнего порога поставим что-то среднее в диапазоне 0-255, а для верхнего возьмем максимально красный. И V – значение или же яркость цвета, работает так же, как и насыщенность. Далее с помощью методов inRange() и bitwise() сделаем нужную нам маску.

redor\_maska = cv2.inRange(hsv, low\_redor, up\_redor)

redpur\_maska = cv2.inRange(hsv, low\_redpur, up\_redpur)

red\_maska = cv2.bitwise\_or(redor\_maska, redpur\_maska)

hsv\_red\_result = cv2.bitwise\_and(img, img, mask=red\_maska)

inRange() создает нам саму маску из наших переданных массивов нужного цвета и входного изображения. Bitwise – это битовые операции, которые выполняют And или Or, с помощью or мы создаем маску, где будут включены два диапазона по краям, а с помощью and ищем нужные цвета на картинке.



**Задание 3** Провести морфологические преобразования (открытие и закрытие) фильтрованного изображения, вывести результаты на экран, посмотреть смысл подобного применения операций erode и dilate.

Операция открытия(размыкания) – это применение дилатации к эрозии исходного изображения. Закрытия(замыкания) – обратная открытии.

Дилатация – морфологическое расширение. Заполняет маленькие дыры и объединяет близкие объекты, общий размер объекта увечилиться.

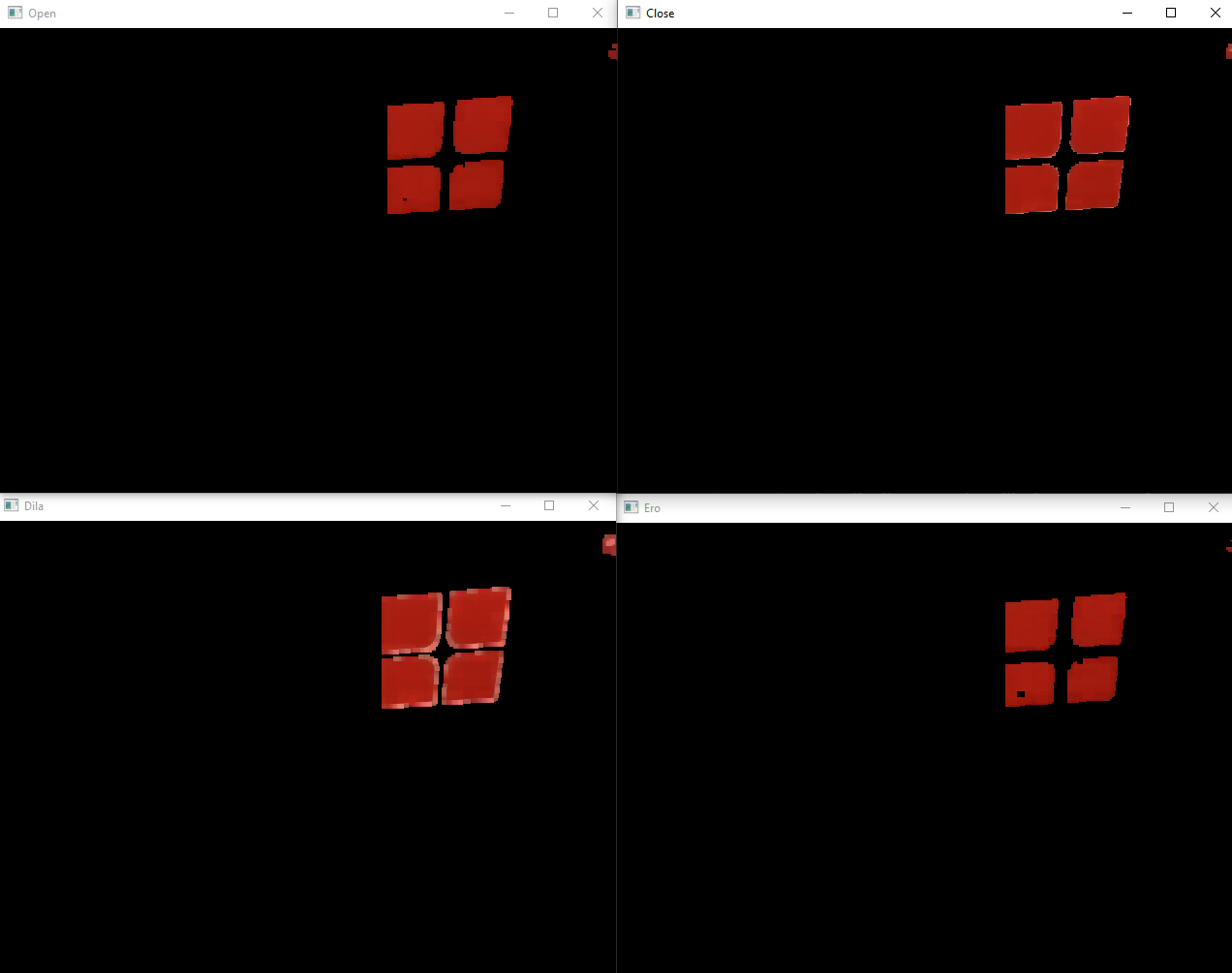
Эрозия – морфологическое уменьшение. Эта операция съедает маленькие объекты и увеличивает большие, но в итоге размер объектов все равно будет меньше, чем до эрозии.

    opening\_img = cv2.morphologyEx(hsv\_red\_result, cv2.MORPH\_OPEN, kernel)

    closing\_img = cv2.morphologyEx(hsv\_red\_result, cv2.MORPH\_CLOSE, kernel)

    dilated\_img = cv2.dilate(hsv\_red\_result, kernel)

    eroded\_img = cv2.erode(hsv\_red\_result, kernel)



**Задание 4** Найти моменты на полученном изображении 1 первого порядка, найти площадь объекта.

Моменты — это статистические характеристики формы объекта, вычисляемые на основе распределения пикселей. К моментам первого порядка относятся: сумма координат по x и по y. Площадь относится к нулевому моменту. В коде это все вычисляется с помощью встроенных функций.

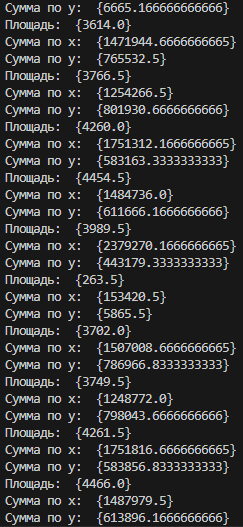
contours, \_ = cv2.findContours(red\_maska, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

square = first\_moments['m00']

summ\_x = first\_moments['m10']

summ\_y = first\_moments['m01']

Метод findContours находит контуры нужных нам объектов, первый флаг – это наша маска для красного цвета, второй флаг говорит как надо искать контуры (в данном случае ищется внешний контур, а внутренние дыры игнорируются), третий флаг говорит как надо запоминать контуры (в данном случае только угловые точки запоминаюстя).



**Задание 5** На основе анализа площади объекта найти его центр и построить черный прямоугольник вокруг объекта. Сделать так, чтобы на видео выводился полученный черный прямоугольник, причем на новом кадре новый.

Из предыдущего анализа используем моменты для нахождения центра по x и y (сумму координат делим на площадь). Сам контур можно найти с помощью метода

x, y, w, h = cv2.boundingRect(contour)

который вернет нам координаты левого верхнего угла, ширину и высоту прямоугольника. Далее в уже известную нам функцию для построения прямоугольников нужно подставить имеющиеся значения и контур вокруг объекта будет готов.

