Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №2**

**Дисциплина: Алгоритмы цифровой обработки мультимедиа**

**Тема: Реализация трекинга**

Работу выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. А. Эзри

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А. Крамаренко

**Цель работы:** реализовать трекинг красного объекта в камере, красный объект необходимо поднести к камере, система его находит и выделяет прямоугольником, далее при движении красного объекта перед камерой прямоугольник движется за ним.

**Ход работы:**

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

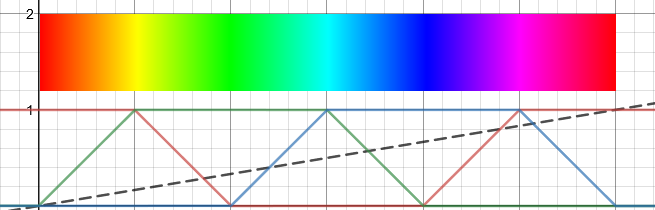
Изображение переведено в формат HSV при помощи метода cv2.cvtColor() с параметром cv2.COLOR\_BGR2HSV\_FULL. Отличие cv2.COLOR\_BGR2HSV\_FULL от cv2.COLOR\_BGR2HSV состоит в том, что если указан параметр со значением FULL, то составляющая Hue имеет 256 различных значений, иначе – 180. В процессе работы выяснилось, что увеличенный диапазон Hue оказывает положительное влияние на итоговый результат и позволяет более точно подстроить границы фильтрации цвета.

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

Функция inRange позволяет создать чёрно-белую маску, в которой белым цветом закрашены те пиксели, каждая составляющая цвета которых находится в заданных диапазонах, например, в режиме HSV можно задать диапазоны для Hue, Saturation и Value.

Для управления диапазонами при помощи ползунков используются функции createTrackbar и getTrackbarPos.

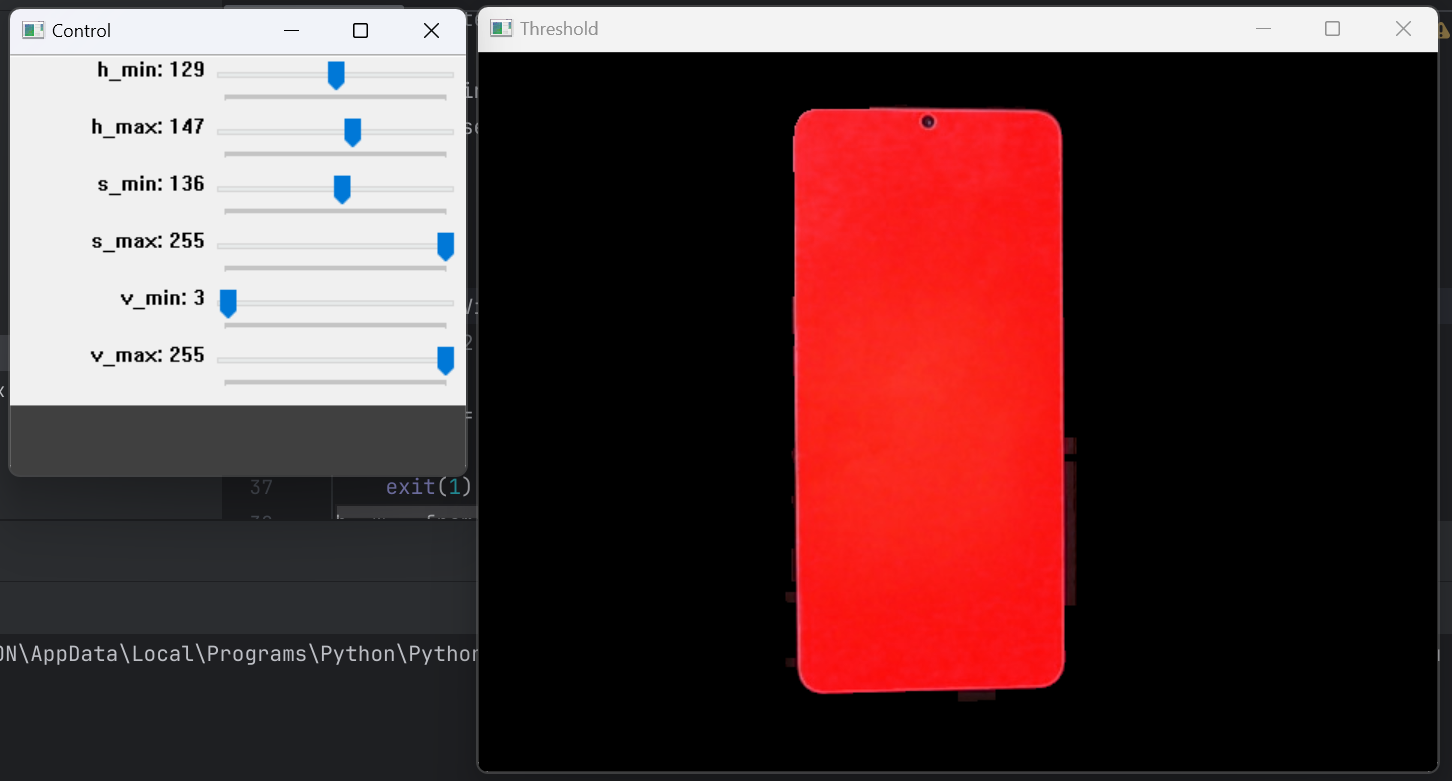
Если выстроить значения Hue в линию, то красный цвет будет находится на краях этой линии – около 0 градусов и около 359 градусов:



Решить эту проблему можно двумя способами: либо создать две маски для каждого из красных диапазонов, либо сместить красный спектр в центр, прибавляя к значению Hue 180 градусов и взятия остатка от деления на 360 (в OpenCV значения другие, для FULL это 128 и 256 соответственно).

Я реализовал вывод цвета под курсором по нажатию левой кнопки мыши в стандартный вывод, чтобы было проще подстраивать ползунки под разные условия освещения.

Кадр с только красными объектами получается путём вызова данной функции: frame\_red = cv2.bitwise\_and(frame, frame, mask=mask).



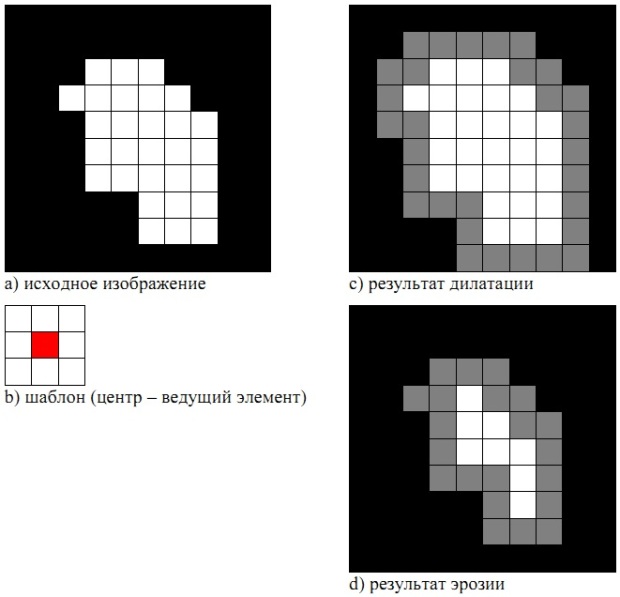
**Задание 3.** Провести морфологические преобразования (открытие и закрытие) фильтрованного изображения, вывести результаты на экран, посмотреть смысл подобного применения операций erode и dilate.

Морфологические преобразования – это операции, применяемые к изображениям с целью изменения формы и структуры объектов на изображении. Наиболее распространены такие морфологического преобразования, как это эрозия и дилатация, а также основанные на них открытие и закрытие. Эти операции часто используются в обработке изображений для удаления шума, заполнения дыр в объектах и изменения размера объектов.

Операция открытия состоит из двух шагов: сначала применяется эрозия (erode), а затем – дилатация (dilate). Сначала эрозия удаляет маленькие объекты и шум на изображении, уменьшая объекты и заполняя небольшие прорехи. Затем дилатация восстанавливает объекты близкой к их исходному размеру.

Операция закрытия также состоит из двух шагов: сначала применяется дилатация (dilate), а затем – эрозия (erode). Сначала дилатация увеличивает объекты и заполняет небольшие отверстия в объектах. Затем эрозия уменьшает объекты обратно к их исходному размеру. Закрытие полезно для закрытия небольших отверстий в объектах и объединения близко расположенных объектов.

Использование морфологических преобразований требует выбора правильных структурирующих элементов (ядро) и настройки их размера в зависимости от конкретной задачи.



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

Моменты изображения в компьютерном зрении, обработке изображений и смежных областях – определённые средневзвешенные значения интенсивности пикселей изображения (называемые моментами), или функция таких моментов.

Например, момент нулевого порядка m00 – это количество всех точек, составляющих пятно. Момент первого порядка m10 представляет собой сумму X координат точек, а m01 – сумму Y координат. Имеются также моменты m11, m20, m02, m22 и т.д.

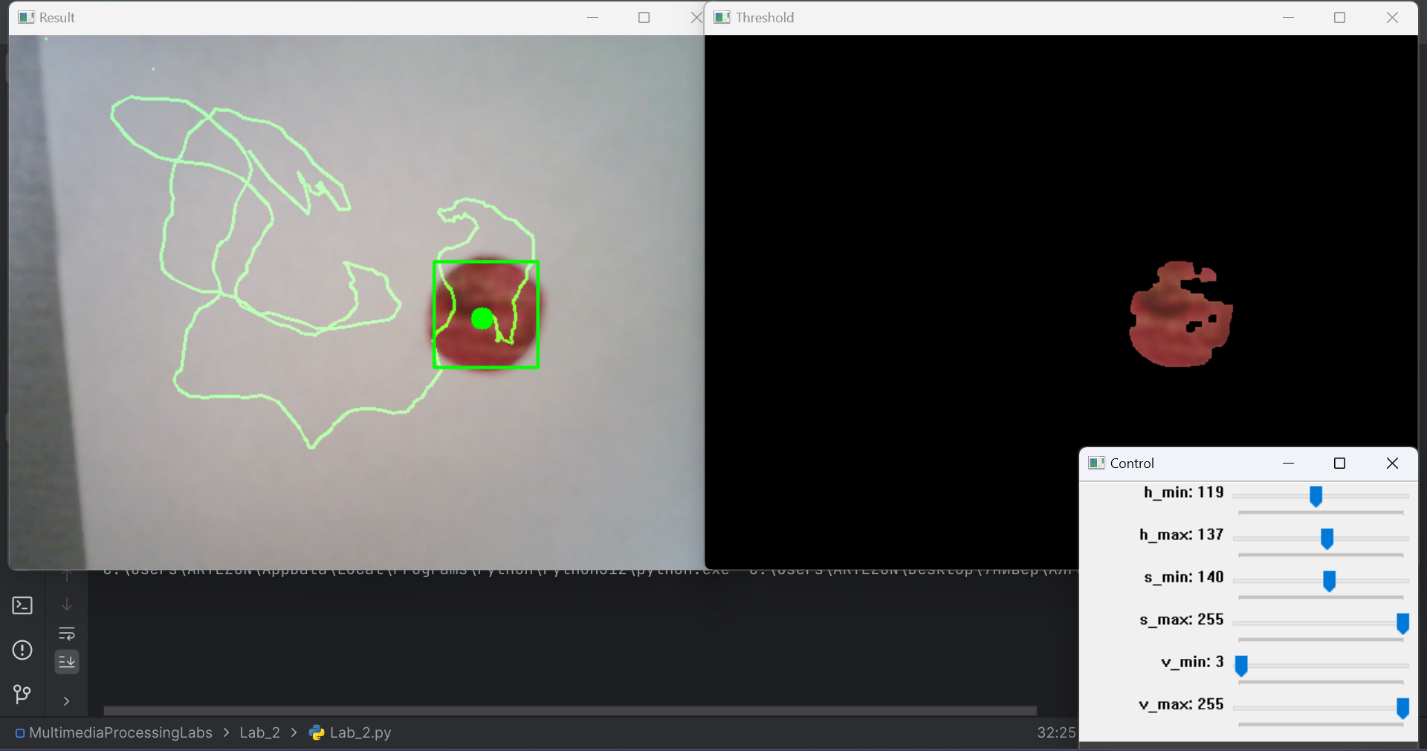
Чтобы получить координаты X и Y искомого пятна, нам следует поделить полученные моменты m10 и m01 на нулевой момент m00. Таким образом мы найдем средние координаты X и Y всех точек, а это и есть центр пятна.

moments = cv2.moments(mask, True)  
dM01 = moments['m01']  
dM10 = moments['m10']  
dArea = moments['m00']

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

Считаем, что красный объект в кадре, если площадь больше 100, это сделано для отсеивания шума. Прямоугольник получаем методом cv2.boundingRect(mask) и рисуем его при помощи cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2).

Также я сделал трекинг объекта и рисование следа за ним:



**Листинг программы**

Файл Lab\_2.py:

import numpy as np  
import cv2  
  
  
def get\_color\_on\_mouse\_click(event, cursorX, cursorY, flags, param):  
 if event == cv2.EVENT\_LBUTTONDOWN:  
 h = frame\_hsv[cursorY, cursorX, 0]  
 s = frame\_hsv[cursorY, cursorX, 1]  
 v = frame\_hsv[cursorY, cursorX, 2]  
 print(f'H = {h}, S = {s}, V = {v}')  
  
  
def callback(value):  
 pass  
  
  
cv2.namedWindow('Control')  
cv2.createTrackbar('h\_min', 'Control', 128 - 9, 255, callback)  
cv2.createTrackbar('h\_max', 'Control', 128 + 9, 255, callback)  
cv2.createTrackbar('s\_min', 'Control', 140, 255, callback) # 90 - 110 - 150  
cv2.createTrackbar('s\_max', 'Control', 255, 255, callback)  
cv2.createTrackbar('v\_min', 'Control', 3, 255, callback)  
cv2.createTrackbar('v\_max', 'Control', 255, 255, callback)  
# cv2.createTrackbar('blur', 'Control', 10, 20, callback)  
  
cv2.namedWindow('Result')  
cv2.setMouseCallback('Result', get\_color\_on\_mouse\_click)  
  
lastX = -1  
lastY = -1  
  
cap = cv2.VideoCapture(1)  
# cap = cv2.VideoCapture('http://192.168.14.104:8080/video')  
  
ok, frame = cap.read()  
if not ok:  
 exit(1)  
h, w = frame.shape[:2]  
path = np.zeros((h, w, 3), np.uint8)  
  
while True:  
 ok, frame = cap.read()  
 key = cv2.waitKey(1) & 0xFF  
 if not ok or key == 27:  
 break  
 elif key == 32: # пробел  
 path.fill(0) # очистить след  
  
 frame\_hsv = cv2.cvtColor(frame, cv2.COLOR\_BGR2HSV\_FULL)  
 frame\_hsv[:, :, 0] = (frame\_hsv[:, :, 0] + 128) % 0xFF # смещение красного цвета в центр  
  
 h\_min = cv2.getTrackbarPos('h\_min', 'Control')  
 h\_max = cv2.getTrackbarPos('h\_max', 'Control')  
 s\_min = cv2.getTrackbarPos('s\_min', 'Control')  
 s\_max = cv2.getTrackbarPos('s\_max', 'Control')  
 v\_min = cv2.getTrackbarPos('v\_min', 'Control')  
 v\_max = cv2.getTrackbarPos('v\_max', 'Control')  
 # blur\_strength = cv2.getTrackbarPos('blur', 'Control')  
 hsv\_min = np.array((h\_min, s\_min, v\_min))  
 hsv\_max = np.array((h\_max, s\_max, v\_max))  
 # blur = cv2.medianBlur(frame\_hsv, blur\_strength \* 2 + 1) # сглаживание изображения  
 mask = cv2.inRange(frame\_hsv, hsv\_min, hsv\_max)  
  
 # mask = cv2.erode(mask, np.ones((5, 5)))  
 # mask = cv2.dilate(mask, np.ones((15, 15)))  
 # mask = cv2.erode(mask, np.ones((5, 5)))  
 mask = cv2.morphologyEx(mask, cv2.MORPH\_OPEN, np.ones((5, 5))) # erosion + dilation (remove small objects)  
 mask = cv2.morphologyEx(mask, cv2.MORPH\_CLOSE, np.ones((5, 5))) # dilation + erosion (remove small holes)  
  
 frame\_red = cv2.bitwise\_and(frame, frame, mask=mask)  
  
 moments = cv2.moments(mask, True)  
 dM01 = moments['m01'] # Y  
 dM10 = moments['m10'] # X  
 dArea = moments['m00']  
 if dArea > 100:  
 posX = int(dM10 / dArea)  
 posY = int(dM01 / dArea)  
 cv2.circle(frame, (posX, posY), 10, (0, 255, 0), -1)  
  
 x, y, w, h = cv2.boundingRect(mask)  
 cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)  
  
 if lastX >= 0 and lastY >= 0:  
 cv2.line(path, (lastX, lastY), (posX, posY), (0, 255, 0), 2)  
 lastX = posX  
 lastY = posY  
 else:  
 lastX = -1  
 lastY = -1  
 frame = cv2.add(frame, path)  
  
 cv2.imshow('Result', frame)  
 cv2.imshow('Threshold', frame\_red)  
  
cv2.destroyAllWindows()

**Ответы на вопросы**

1. Опишите принципы фильтрации командой inRange

2. Опишите, что делают команды erode и dilate, опишите принципы математических преобразований изображения для этих команд.

3. Опишите, для чего применяем морфологическое открытие и закрытие, чем обусловлена такая необходимость?

Открытие – это операция, которая включает в себя сначала вызов оператора эрозии, а после вызывается оператор дилатации над полученным изображение после оператора эрозии.

Закрытие – это операция, которая включает в себя сначала вызов операции дилатации, а после вызывается операция эрозии над полученным изображением после операции дилатации.

4. Опишите, что такое моменты изображения?

Момент нулевого порядка m00 – это количество всех точек, составляющих пятно. Момент первого порядка m10 представляет собой сумму X координат точек, а m01 – сумму Y координат. Имеются также моменты m11, m20, m02, m22 и т.д.

5. Опишите, что такое центроид объекта изображения, как его находим и для чего?