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 №1**

**Дисциплина: Алгоритмы цифровой обработки мультимедиа**

Работу выполнила: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_К.В.Стасюк

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

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

**Цель работы:** Тестирование вывода изображений и видео на экран, запись в файл, формат изображения HSV, определение цвета, построение надписей и дополнительных изображений на рисунке.

**Ход работы:**

Задание 1. Установить библиотеку OpenCV.

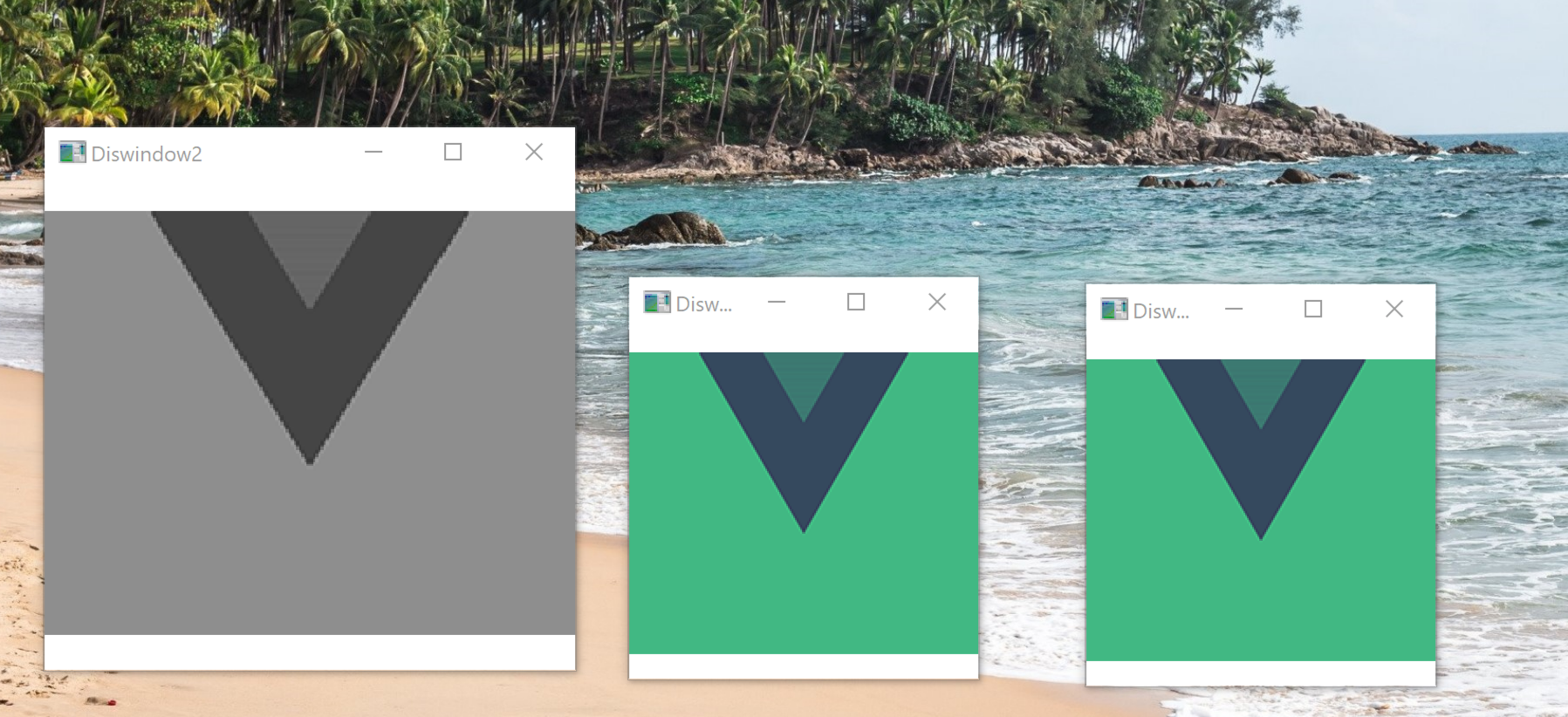
С помощью команды pip install cv2 через терминал в IDE PyCharm 2022.3 библиотека OpenCV была установлена. Для работы необходимо было включать библиотеку в проект, в частности, в каждый скрипт директивой

import cv2

Задание 2. Вывести на экран изображение. Протестировать три возможных расширения, три различных флага для создания окна и три различных флага для чтения изображения.

Для того чтобы вывести изображение на экран необходимо использовать метод cv.imread(), отвечающий за загрузку картинки на экран. Параметры (флаги) этой функции позволяют изменить расширение изображения, способы чтения и т. д. На рисунке представлены 3 изображения. Первое изображение-

картинка в формате png с использованием флага cv2.WINDOW\_NORMAL для размера окна и cv2.IMREAD\_GRAYSCALE для серого цвета изображения. Второе изображение - картинка в формате jpg с использованием флага cv2.WINDOW\_AUTOSIZE для автоматического размера экрана и cv2.IMREAD\_COLOR для цветного изображения. Третье изображение - картинка в формате bpm с использованием флага cv2.WINDOW\_FULLSCREEN для размера окна равному размеру изображения и cv2.IMREAD\_UNCHANGED для представления изображения в начальном виде. Можно заметить, что 2 и 3 изображения представлены одинаково. Именованное окно создано с помощью команды сv2.namedWindow() .

Рисунок 1 – Изображения для 1 задания

Задание 3. Отобразить видео в окне. Рассмотреть методы класса VideoCapture и попробовать отображать видео в разных форматах, в частности размеры и цветовая гамма.

Отобразить заранее записанное видео или видеопоток с IP камеры можно с помощью методов класса VideoCapture. Создаётся экземпляр класса с помощью конструктора cv2.VideoCapture(), в качестве параметров передается название видео с расширением и флаг размерности окна. Далее используется цикл while с командой cv2.isOpened(), т. е. пока видео открыто, цикл выполняется. Кадр из видеопотока читается с помощью функции cap.read(). Эта функция возвращает два значения, первое значение ret – булевское значение, обозначающее, удалось ли выполнить чтение кадра. Сам кадр называем фреймом и сохраняем в формат картинки (двумерная матрица). Если изображение закончилось, ret вернет false и отображение завершится. Далее отобразим полученный фрейм в окне (с помощью команды cv2.imshow() и проверим условие на закрытие окна. Если нажата клавиша Escape (код 27), отображение будет завершено.

Отобразить видео в разных размерах можно с помощью команды cv2.resizeWindow(), а в разных цветах -cv2.cvtColor() . Флаги первой команды позволяют изменять размер окна, а второй – цветовую гамму. На рисунках 2 и 4 представлены результаты работы программы с разными размерами и флагами.

Изображение выглядит как пейзаж, пляж, снимок экрана, Красочность

Автоматически созданное описаниеРисунок 2 – Отображение видео в окне размером 800х600 в красном цвете.

Изображение выглядит как черно-белый, на открытом воздухе, природа, облако

Автоматически созданное описание

Рисунок 3 – Отображение видео в окне размером 1024х1000 в сером цвете.

Задание 4. Записать видео из файла в другой файл.

Для выполнения данного задания код, используемый в предыдущем задании, был изменён. В частности, была добавлена команда для получения размеров кадра (cv2.CAP\_PROP\_FRAME\_WIDTH, cv2.CAP\_PROP\_FRAME\_HEIGHT) и был создан объект класса VideoWriter для записи видео в файл с помощью команды cv2.VideoWriter('output\_1.mov', fourcc, 30.0, (width, height), в аргументах которой указано название файла, в который записано видео, код кодека, используемый для сжатия кадров, частота кадров создаваемого видеопотока и размер видеокадров.

Изображение выглядит как природа, на открытом воздухе, небо, берег

Автоматически созданное описание

Рисунок 4 – Отображение видео в исходном размере и цвете.

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 5 – Видео записано в файл.

Задание 5. Прочитать изображение, перевести его в формат HSV. Вывести на экран два окна, в одном изображение в формате HSV, в другом – исходное изображение.

Перевести изображение в формат HSV с легкостью можно с помощью ранее используемой команды cv2.cvtColor(), указав в аргументах флаг HSV формата (cv2.COLOR\_BGR2HSV).

RGB использует аддитивное цветовое пространство для отображения цветов на экране, состоящее из красного, зеленого и синего цветов. В этом случае цвет создается путем смешивания света разных цветовых компонентов, например, красного, зеленого и синего света смешиваются, чтобы получить цвет.

С другой стороны, HSV использует цветовое пространство, которое представляет цвета в виде оттенка, насыщенности и значения. Он позволяет более интуитивно описывать цвета, где оттенок указывает на цветовой тон, насыщенность определяет насыщенность цвета, а значение определяет яркость.

Изображение выглядит как снимок экрана, Красочность, Графика, дизайн

Автоматически созданное описание

Рисунок 6 – Изображение в формате RGB(слева) и в формате HSV(справа).

Задание 6. (самостоятельно) Прочитать изображение с камеры. Вывести в центре на экране Красный крест в формате, как на изображении. Указать команды, которые позволяют это сделать.

Для создания прямоугольников на изображении с определенными размерами, цветом и шириной линий сначала вычислялись координаты их углов так, чтобы они находились в центре изображения. Для этого использовались следующие формулы:

Левый верхний угол прямоугольника находился в точке (xi\_1, yi\_1), где xi\_1 = (ширина изображения / 2) - (ширина прямоугольника / 2), а yi\_1 = (высота изображения / 2) - (высота прямоугольника / 2).

Правый нижний угол прямоугольника находился в точке (xi+1\_1, yi+1\_1), где xi+1\_1 = (ширина изображения / 2) + (ширина прямоугольника / 2), а yi+1\_1 = (высота изображения / 2) + (высота прямоугольника / 2).

Затем для отрисовки прямоугольников использовалась команда cv2.rectangle(), в которую передавались изображение, координаты углов прямоугольника, его цвет и толщина линий.

Для размытия центра пересечения горизонтального и вертикального прямоугольников применялся метод GaussianBlur(). Этот метод использует фильтр Гаусса для размытия изображения. Он принимает следующие параметры:

Исходное изображение.

Размер ядра (окна), которое определяет, сколько пикселей будет участвовать в вычислении нового значения пикселя.

Стандартное отклонение по X и Y, которые влияют на степень размытия. Чем больше размер ядра, тем сильнее размытие, и чем больше стандартное отклонение, тем меньше влияние дальних пикселей на вычисление нового значения пикселя.Изображение выглядит как в помещении, стена, ваза, стол

Автоматически созданное описание

Рисунок 7 – Изображение с размытым треугольником.

Задание 7. (самостоятельно) Отобразить информацию с вебкамеры, записать видео в файл, продемонстрировать видео.

Для выполнения этого задания код, используемый в задании 3 был модифицирован в функцию readIPWriteTOFile(). В конструкторе cv2.VideoCapture(0) был указан 0, отвечающий за подключение веб-камеры, в отличие от предыдущего задания, где было указано название используемого видео.

Изображение выглядит как текст, в помещении, снимок экрана, видео

Автоматически созданное описание

Рисунок 8 – Запись с вебкамеры

Задание 8. (самостоятельно) Залить крест одним из 3 цветов – красный, зеленый, синий по следующему правилу: НА ОСНОВАНИИ ФОРМАТА RGB определить, центральный пиксель ближе к какому из цветов красный, зеленый, синий и таким цветом заполнить крест.

Сначала был получен цвет центрального пикселя изображения в формате RGB.Затем вычислялось расстояние между этим цветом и каждым из трех возможных цветов (красным, зеленым и синим) в пространстве RGB. Для этого использовалось евклидово расстояние между координатами цветов. То есть, для каждого из цветов (красного, зеленого и синего) вычислялась "удаленность" от текущего цвета центрального пикселя в RGB-пространстве. Ближайший цвет из списка возможных цветов определялся по индексу, соответствующему минимальному расстоянию. То есть, цвет, который имел наименьшее евклидово расстояние до цвета центрального пикселя, выбирался как ближайший. После определения ближайшего цвета, была использована функция cv2.rectangle() . Были нарисованы прямоугольники, но с измененными параметрами, чтобы закрасить оба прямоугольника в цвет ближайшего цвета.

Изображение выглядит как снимок экрана, Красочность, линия, Графика

Автоматически созданное описание

Рисунок 9 – Закрашенный в красный цвет крест.

Задание 9. (самостоятельно). Подключите телефон, подключитесь к его камере, выведете на экран видео с камеры. Продемонстрировать процесс на ноутбуке преподавателя и своем телефоне.

Для выполнения этого задания предварительно на телефон было установлено приложение DroidCam , позволяющее подключить телефон как IP-камеру. В конструктор cv2.VideoCapture("http://192.168.0.101:4747/video") был передан URL-адрес потока видео. Также этот URL-адрес можно использовать в браузере на компьютере, чтобы увидеть видео с телефона.

Изображение выглядит как в помещении, стена, дизайн интерьера, занавеска

Автоматически созданное описаниеРисунок 10 – Запись с камеры телефона.

**Листинг программ**

Файл task1­.py

import cv2  
  
# Флаги для создания окна  
cv2.namedWindow('Diswindow1',cv2.WINDOW\_AUTOSIZE)  
cv2.namedWindow('Diswindow2',cv2.WINDOW\_NORMAL)  
cv2.namedWindow('Diswindow3',cv2.WINDOW\_FULLSCREEN)  
  
# Флаги для чтения изображения  
img1 = cv2.imread('logo2.jpg', cv2.IMREAD\_COLOR)  
img2 = cv2.imread('logo.png', cv2.IMREAD\_GRAYSCALE) #серый  
img3 = cv2.imread('logo 3.bmp', cv2.IMREAD\_UNCHANGED)  
  
cv2.imshow('Diswindow1', img1)  
cv2.imshow('Diswindow2', img2)  
cv2.imshow('Diswindow3', img3)  
cv2.waitKey(0)

Файл task3.py

import cv2  
  
# отображение видео в окне  
cap =cv2.VideoCapture(r'C:\Disk CD\обои\фото\Новая папка\весна 2019\BDRD7996.MOV', cv2.WINDOW\_NORMAL)  
cv2.namedWindow('Video', cv2.WINDOW\_NORMAL)  
  
# изменение размера окна  
# cv2.resizeWindow('Video', 800, 600)  
cv2.resizeWindow('Video', 1024, 1000)  
# cv2.resizeWindow('Video', 1800, 800)  
  
# чтение видеофайла кадр за кадром  
while cap.isOpened():  
 ret, frame = cap.read()  
 if ret:  
 # изменение цветовой гаммы кадра  
 gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)  
 vsh = cv2.cvtColor(frame, cv2.COLOR\_BGR2HSV)  
 lab = cv2.cvtColor(frame, cv2.COLOR\_BGR2LAB)  
  
 # отображение кадра в окне  
 cv2.imshow('Video',gray)  
  
 # выход при нажатии клавиши 'esc'  
 if cv2.waitKey(1) & 0xFF == 27:  
 break  
 else:  
 break  
  
cap.release()  
cv2.destroyAllWindows()

Файл task4.py

import cv2  
  
# отображение видео в окне  
cap = cv2.VideoCapture(r'C:\Disk CD\обои\фото\Новая папка\весна 2019\BDRD7996.MOV',cv2.WINDOW\_NORMAL)  
cv2.namedWindow('Video', cv2.WINDOW\_NORMAL)  
  
# получение размеров кадра  
width = int(cap.get(cv2.CAP\_PROP\_FRAME\_WIDTH))  
height = int(cap.get(cv2.CAP\_PROP\_FRAME\_HEIGHT))  
  
# создание объект VideoWriter для записи видео в файл  
fourcc = cv2.VideoWriter\_fourcc(\*'mp4v')  
out = cv2.VideoWriter('output\_1.mov', fourcc, 30.0, (width, height))  
  
while cap.isOpened():  
 ret, frame = cap.read()  
 if ret:  
 out.write(frame)  
 # отображение кадра в окне  
 cv2.imshow('Video', frame)  
  
 # выход при нажатии клавиши 'esc'  
 if cv2.waitKey(1) & 0xFF == 27:  
 break  
 else:  
 break  
  
cap.release()  
out.release()  
cv2.destroyAllWindows()

Файл task5.py

import cv2  
  
img = cv2.imread('logo.png')  
  
# окна для отображения изображений  
cv2.namedWindow('Original Image', cv2.WINDOW\_NORMAL)  
cv2.namedWindow('HSV Image', cv2.WINDOW\_NORMAL)  
  
# преобразование изображение в формат HSV  
hsv = cv2.cvtColor(img, cv2.COLOR\_BGR2HSV)  
  
#отображение изображений  
cv2.imshow('HSV Image', hsv)  
cv2.imshow('Original Image', img)  
  
cv2.waitKey(0)  
cv2.destroyAllWindows()

Файл task6.py

import cv2  
  
img = cv2.imread('cam\_photo.jpg')  
cv2.namedWindow('Display window', cv2.WINDOW\_NORMAL)  
  
# цвет и толщина прямоугольников и линии  
color = (0, 0, 255) # красный цвет  
thickness = 2 # толщина  
  
# размеры изображения  
height, width, \_, = img.shape  
  
# ширина и высота вертикального прямоугольника  
rect\_width\_1 = 50  
rect\_height\_1 = 400  
  
# координаты углов для вертикального прямоугольника  
x1\_1 = width // 2 - rect\_width\_1 // 2 # левый верхний угол по оси x  
y1\_1 = height // 2 - rect\_height\_1 // 2 # левый верхний угол по оси y  
x2\_1 = width // 2 + rect\_width\_1 // 2 # правый нижний угол по оси x  
y2\_1 = height // 2 + rect\_height\_1 // 2 # правый нижний угол по оси y  
  
# ширина и высота горизонтального прямоугольника  
rect\_width\_2 = 50  
rect\_height\_2 = 350  
  
# координаты углов горизонательного прямоугольника  
x1\_2 = width // 2 - rect\_height\_2 // 2 # левый верхний угол по оси x  
y1\_2 = height // 2 - rect\_width\_2 // 2 # левый верхний угол по оси y  
x2\_2 = width // 2 + rect\_height\_2 // 2 # правый нижний угол по оси x  
y2\_2 = height // 2 + rect\_width\_2 // 2 # правый нижний угол по оси y  
  
# отрисовка  
cv2.rectangle(img, (x1\_1, y1\_1), (x2\_1, y2\_1), color, thickness)  
cv2.rectangle(img, (x1\_2, y1\_2), (x2\_2, y2\_2), color, thickness)  
  
# размер ядра для размытия  
kernel\_size = (71, 11) # ширина и высота ядра в пикселях  
  
# часть изображения, соответствующая горизонтальному прямоугольнику  
img\_part = img[y1\_2:y2\_2, x1\_2:x2\_2]  
  
img\_part\_blur = cv2.GaussianBlur(img\_part, kernel\_size, 30)  
  
# замена части изображения размытой версией  
img[y1\_2:y2\_2, x1\_2:x2\_2] = img\_part\_blur  
  
cv2.imshow('Display window', img)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

Файл task7.py

import cv2  
  
# Функция для чтения видео с веб-камеры и записи его в файл  
def readIPWriteTOFile():  
 # Создание объекта для захвата видео с устройства (0 - индекс камеры по умолчанию)  
 video = cv2.VideoCapture(0)  
  
 # Получение ширины и высоты кадра  
 w = int(video.get(cv2.CAP\_PROP\_FRAME\_WIDTH))  
 h = int(video.get(cv2.CAP\_PROP\_FRAME\_HEIGHT))  
  
 # Установка кодека для записи видео  
 fourcc = cv2.VideoWriter\_fourcc(\*'XVID')  
  
 # Создание объекта для записи видео в файл "output\_3.mov" с заданными параметрами  
 video\_writer = cv2.VideoWriter("output\_3.mov", fourcc, 25, (w, h))  
  
 # Бесконечный цикл для захвата и записи видео  
 while (True):  
 # Захват кадра с веб-камеры  
 ok, img = video.read()  
  
 # Отображение кадра на экране  
 cv2.imshow('Webcam video', img)  
  
 # Запись кадра в файл  
 video\_writer.write(img)  
  
 # Выход из цикла при нажатии клавиши 'esc'  
 if cv2.waitKey(1) & 0xFF == 27:  
 break  
  
 # Освобождение ресурсов и закрытие окон  
 video.release()  
 cv2.destroyAllWindows()  
  
  
readIPWriteTOFile()

Файл task8.py

import cv2  
import numpy as np  
  
  
img = cv2.imread('logo.png')  
cv2.namedWindow('Display window', cv2.WINDOW\_NORMAL)  
  
# Установка параметров для прямоугольников и линии  
color = (0, 0, 255) # Красный цвет в формате BGR (синий, зеленый, красный)  
thickness = 2 # Толщина линии  
  
# Получение размеров изображения (ширина, высота)  
height, width, \_ = img.shape  
  
# Вертикальный прямоугольник  
rect\_width\_1 = 50  
rect\_height\_1 = 400  
x1\_1 = width // 2 - rect\_width\_1 // 2  
y1\_1 = height // 2 - rect\_height\_1 // 2  
x2\_1 = width // 2 + rect\_width\_1 // 2  
y2\_1 = height // 2 + rect\_height\_1 // 2  
  
# Горизонтальный прямоугольник  
rect\_width\_2 = 50  
rect\_height\_2 = 350  
x1\_2 = width // 2 - rect\_height\_2 // 2  
y1\_2 = height // 2 - rect\_width\_2 // 2  
x2\_2 = width // 2 + rect\_height\_2 // 2  
y2\_2 = height // 2 + rect\_width\_2 // 2  
  
# Отрисовка прямоугольников на изображении  
cv2.rectangle(img, (x1\_1, y1\_1), (x2\_1, y2\_1), color, thickness)  
cv2.rectangle(img, (x1\_2, y1\_2), (x2\_2, y2\_2), color, thickness)  
  
# Размер ядра для размытия  
kernel\_size = (71, 11)  
  
# Выделение части изображения, соответствующей горизонтальному прямоугольнику  
img\_part = img[y1\_2:y2\_2, x1\_2:x2\_2]  
  
# Применение размытия к выделенной части изображения  
img\_part\_blur = cv2.GaussianBlur(img\_part, kernel\_size, 30)  
  
# Замена исходной части изображения размытой версией  
img[y1\_2:y2\_2, x1\_2:x2\_2] = img\_part\_blur  
  
# Определение цвета центрального пикселя  
cx = width // 2  
cy = height // 2  
r, g, b = img[cy][cx]  
  
# Список возможных цветов в формате RGB  
colors = [(255, 0, 0), (0, 255, 0), (0, 0, 255)]  
  
# Расчет расстояний до каждого цвета и выбор ближайшего цвета  
distances = [np.sqrt((r - color[0])\*\*2 + (g - color[1])\*\*2 + (b - color[2])\*\*2) for color in colors]  
min\_index = distances.index(min(distances))  
nearest\_color = colors[min\_index]  
  
# Закрашивание прямоугольников ближайшим цветом  
cv2.rectangle(img, (x1\_1, y1\_1), (x2\_1, y2\_1), nearest\_color, -1)  
cv2.rectangle(img, (x1\_2, y1\_2), (x2\_2, y2\_2), nearest\_color, -1)  
  
# Отображение изображения  
cv2.imshow('Display window', img)  
  
# Ожидание нажатия клавиши и закрытие окна по нажатию  
cv2.waitKey(0)  
cv2.destroyAllWindows()

Файл task9.py

import cv2  
  
# Создание объекта VideoCapture для подключения к IP-камере  
#URL-адрес потока видео с IP-камеры  
cap = cv2.VideoCapture("http://192.168.0.101:4747/video")  
  
while True:  
 # Считывание кадра с IP-камеры  
 ret, frame = cap.read()  
  
 if ret:  
 # Отображение кадра с IP-камеры на экране  
 cv2.imshow("Phone's camera", frame)  
  
 # Ожидание нажатия клавиши 'q' для выхода из цикла  
 if cv2.waitKey(1) & 0xFF == 27:  
 break  
 else:  
 # Если возникла ошибка при чтении видео, выходим из цикла  
 print("Ошибка чтения видео")  
 break  
  
# Освобождение ресурсов и закрытие окон  
cap.release()  
cv2.destroyAllWindows()