Факультет компьютерных технологий и прикладной математики

Кафедра вычислительных технологий

02.03.02

Алгоритмы цифровой обработки мультимедиа

Лабораторная работа № 1

Тестирование вывода изображений и видео на экран, запись в файл, Формат изображения HSV, определение цвета, построение надписей и доп изображений на рисунке

Работа будет осуществлятся средства языка Python 3.10 и IDE PyCharm2022.1.2 с учебной лицензией. Для работы необходимо установить библиотеку opencv.

Задание 1. Установить библиотеку OpenCV.

Для работы необходимо включить библиотеку в проект директивой

```
import cv2
```

Далее загрузим картинку в программу, для чего воспользуемся функцией cv.imread(filename[, flags]) -> retval

Данная функция возвращает двумерную матрицу, хранящую изображение в формате, считанном из файла. Формат изображения определяется самим изображением, а не расширением файла. Полное описание функции доступно по ссылке https://docs.opencv.org/4.x/d4/da8/group__imgcodecs.html#ga288b8b3da0892bd651fce07b3bbd3a56

На этом ресурсе присутствует официальное руководство по указанной библиотеке.

В случае возникновения ошибок воспользоваться приведенным ниже описанием и рисунком 1.

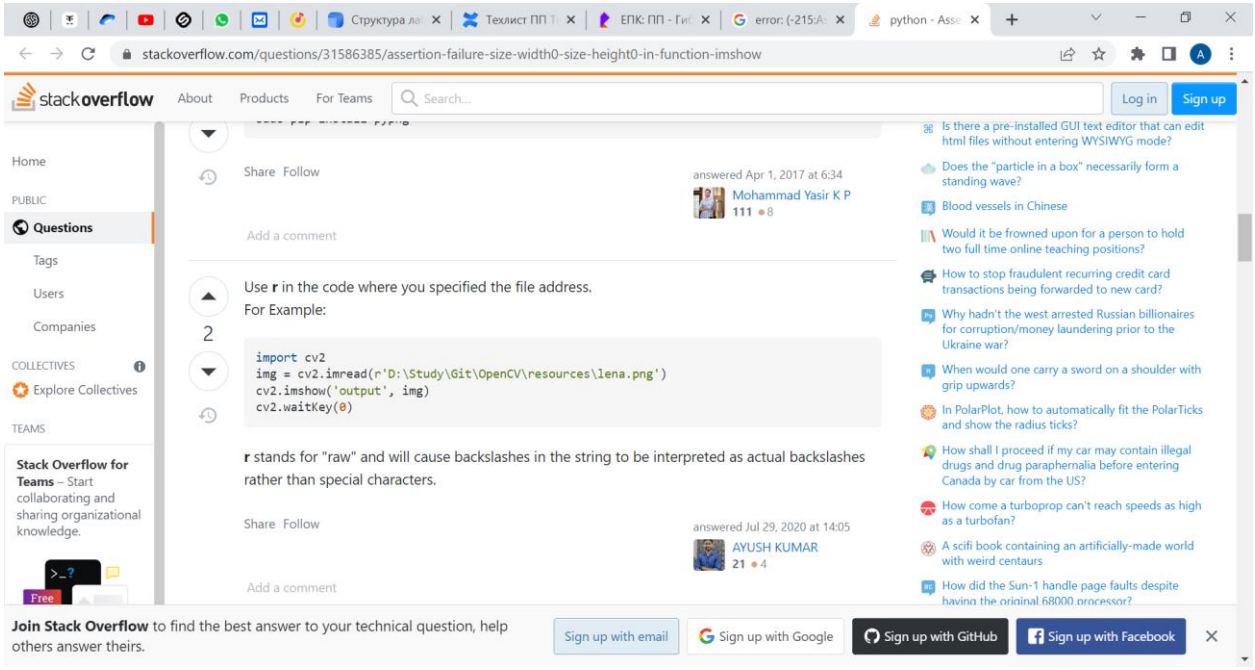



Рисунок 1 – Чтение файла

[Assertion failure : size.width>0 && size.height>0 in function imshow]

flags = **IMREAD_COLOR**

Приведенная строка означает режимы чтения файла с изображением, все возможные режимы показаны на странице 2.

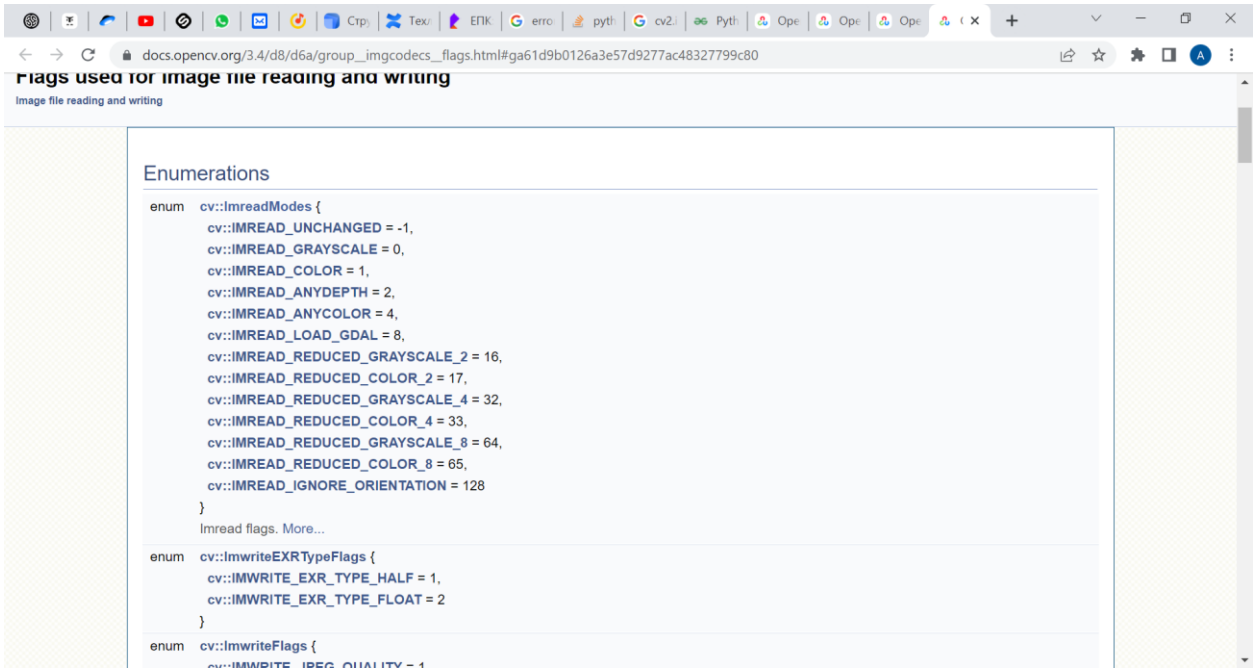



Рисунок 2 – Флаги чтения картинки

Далее рассмотрим код, который загружает картинку, отображает ее на экран и ждет, пока пользователь закроет картинку, как показано в Листинге 1.

Первая функция уже разобрана.

Вторая строка создает именованное окно с указанным именем и различными флагами. Полное описание функции доступно по ссылке и показано на рисунке 3.

https://docs.opencv.org/4.x/d7/dfc/group__highgui.html#ga5afdf8410934fd099df85c75b2e0888b

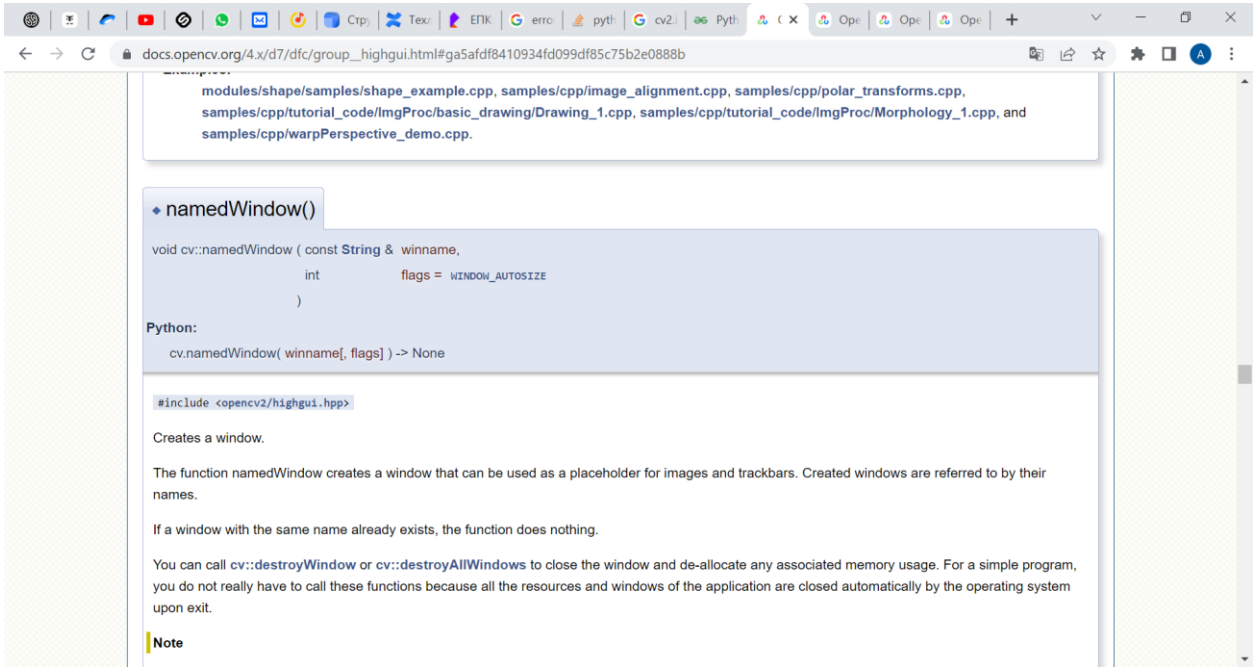



Рисунок 3 – Описание функции namedWindow()

```
img1 = cv2.imread(r'C:\Users\Arseniy.Zhuk\Documents\master-discount.png')
cv2.namedWindow('Display window', cv2.WINDOW_NORMAL)
cv2.imshow('Display window', img1)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Листинг 1 – Отображение картинки

Если окна с указанным именем не обнаружено, функция ничего не сделает, в противном случае покажет изображение. Флаги, характеризующие формат изображения доступны по ссылке и показаны на рисунке 4.

https://docs.opencv.org/4.x/d0/d90/group__highgui__window__flags.html#gabf7d2c5625bc59ac130287f925557ac3

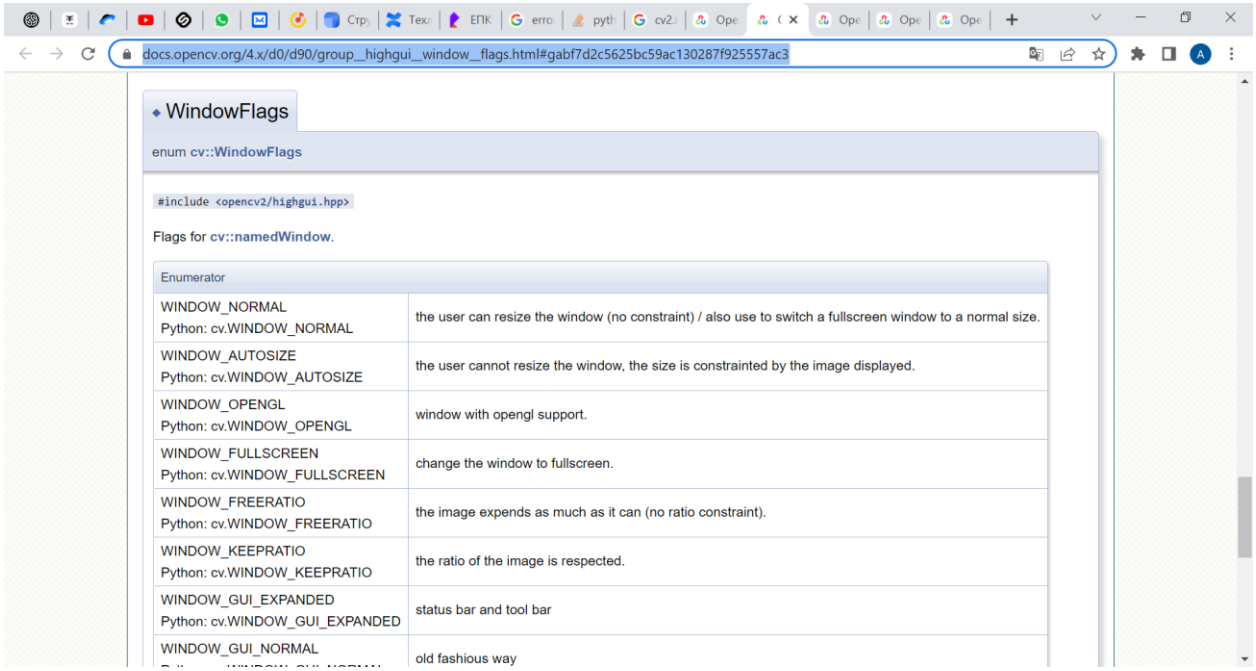



Рисунок 4 – Флаги создания окна

Следующей командой отображаем окно с введенным именем на экран. Описание функции доступно по ссылке и показано на рисунке 5.

https://docs.opencv.org/4.x/d7/dfc/group__highgui.html#ga453d42fe4cb60e5723281a89973ee563

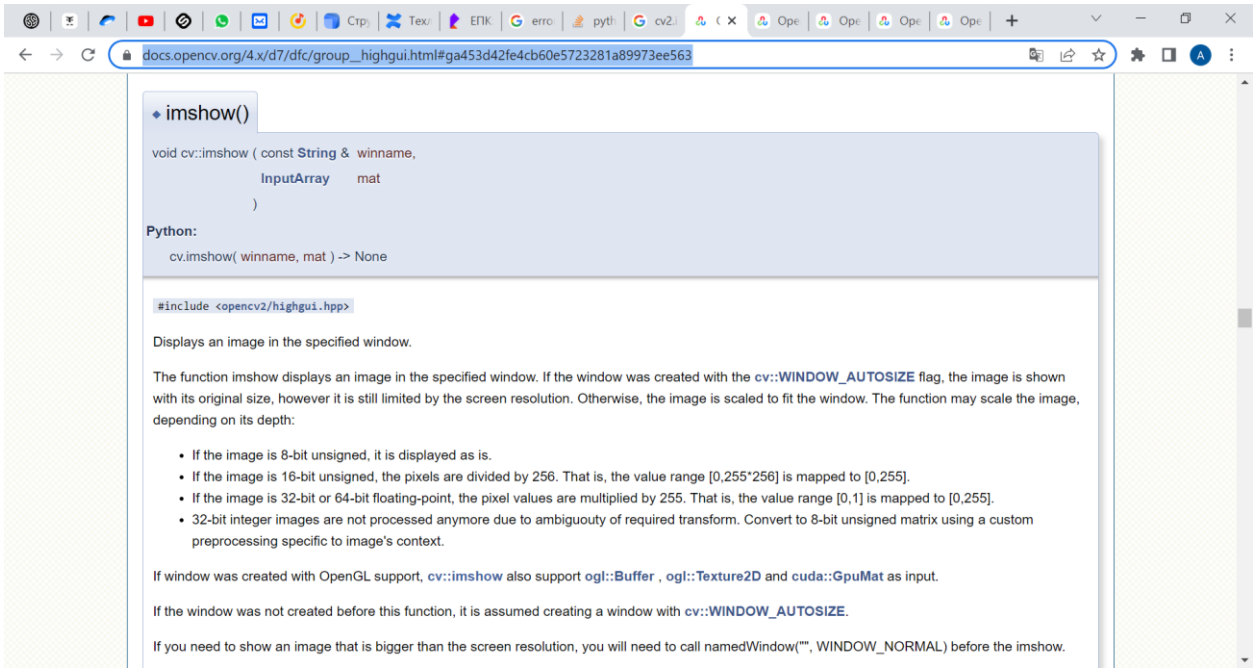



Рисунок 5 – Описание функции imshow()

Следующие две команды позволяют ожидать нажатия кнопки и закрыть окно, ниже на рисунках 6 и 7 приведено полное описание данных функций в официальной документации.

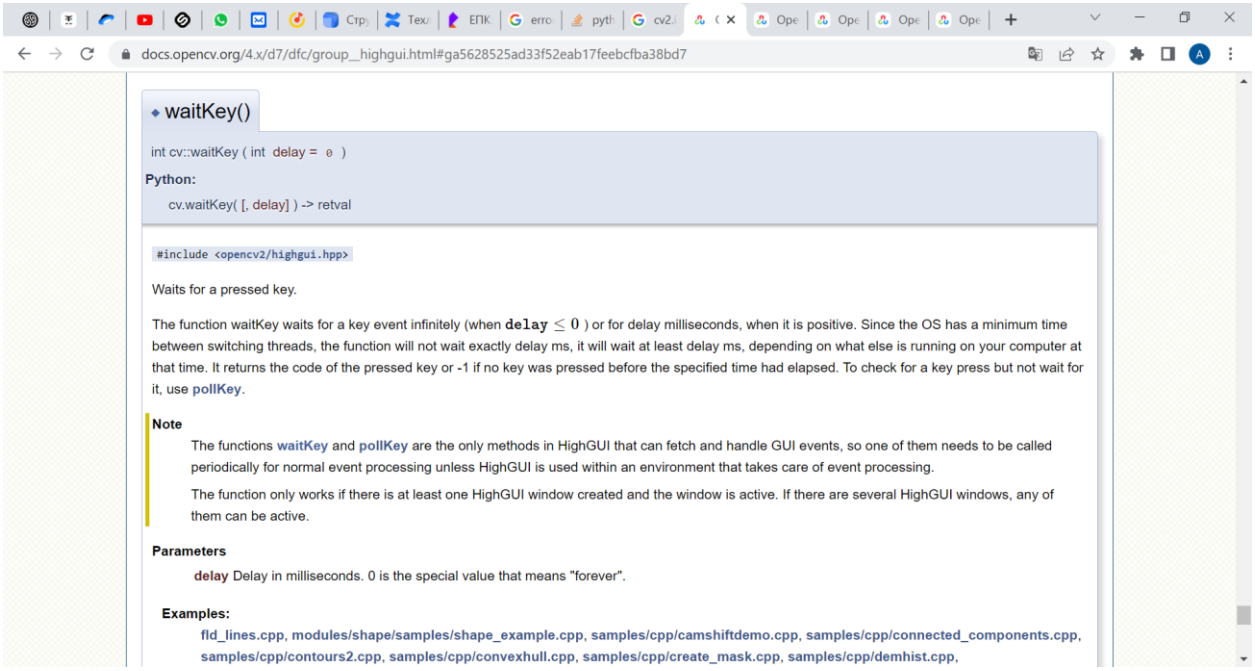



Рисунок 6 – Описание функции waitkey()

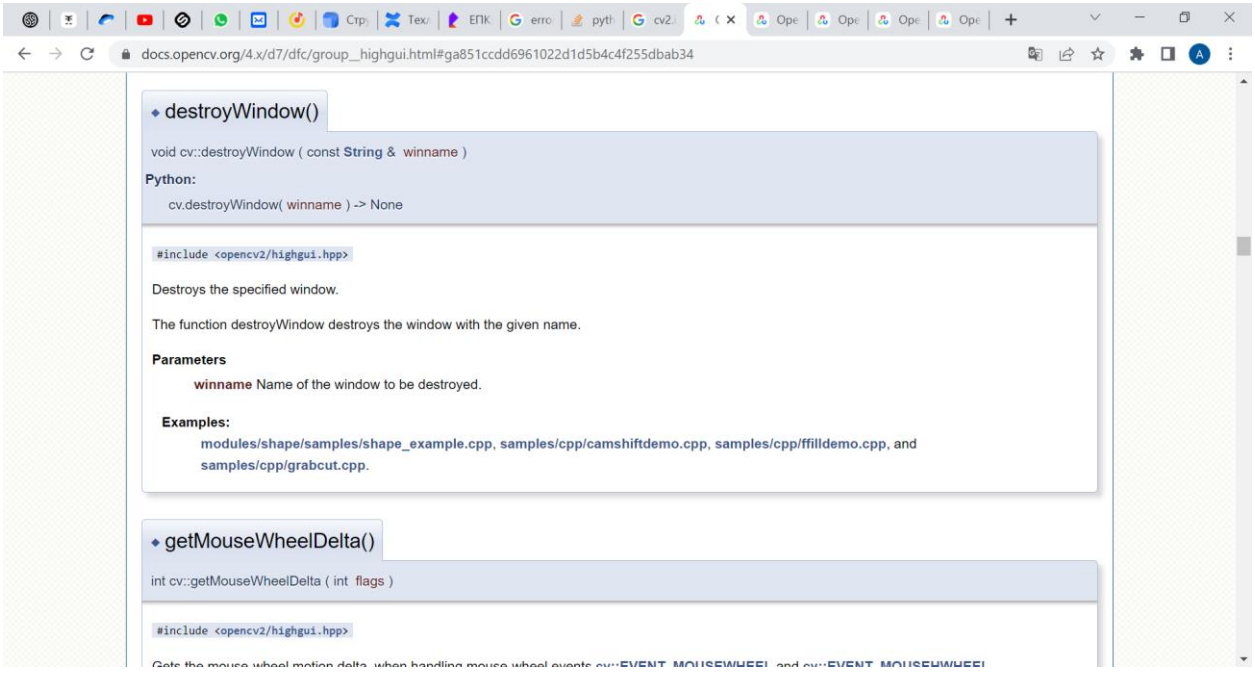



Рисунок 7 – Описание функции destroyWindow()

На основании разбора приведенной информации возможно выполнение следующего задания.

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

Далее рассмотрим возможность работы с видео потоком. За работу с видео потоком отвечает класс VideoCapture, полное описание класса доступно по ссылке

https://docs.opencv.org/4.x/d8/dfe/classcv_1_1VideoCapture.html#a8c6d8c2d37505b5ca61ffd4bb54e9a7c

Создадим экземпляр класса с помощью конструктора, как показано в листинге 2

```
cap =
cv2.VideoCapture(r'C:\Users\Arseniy.Zhuk\Documents\CFS_price_copy_validation.
mp4', cv2.CAP_ANY)
```

Листинг 2 – Создание

Далее создадим цикл для отображения видео, как показано в листинге 3. В первой команде прочитаем кадр из видео потока. Описание функции cap.read() доступно по ссылке https://docs.opencv.org/4.x/d8/dfe/classcv_1_1VideoCapture.html#a473055e77dd7faa4d26d686226b292c1

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.

```
ret, frame = cap.read()
if not(ret):
    break
cv2.imshow('frame', frame)
if cv2.waitKey(1) & 0xFF == 27:
    break
```

Листинг 3 – Отображение видео

Далее ждем 1 милисекунду и меняем кадр. Если нажата клавиша Escape(код 27), отображение завершено.

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

Далее рассмотрим следующую задачу, а именно чтение видеопотока с IP камеры и запись видеопотока в файл. Решение этой задачи приведено в листинге 4. В первой строке создание экземпляра видеопотока. Далее читаем кадр и определяем размер кадра. Далее создаем видео поток на запись, класс VideoWriter, его описание доступно по ссылке https://docs.opencv.org/4.x/dd/d9e/classcv_1_1VideoWriter.html#afec93f94dc6c0b3e28f4dd153bc5a7f0

Далее происходит запись изображения в выходной файл.

```
def readIPWriteTOFile():
    video = cv2.VideoCapture("rtsp://192.168.1.217/stream1")
    ok, img = video.read()
    w = int(video.get(cv2.CAP_PROP_FRAME_WIDTH))
    h = int(video.get(cv2.CAP_PROP_FRAME_HEIGHT))
    fourcc = cv2.VideoWriter_fourcc(*'XVID')
    video_writer = cv2.VideoWriter("output.mov", fourcc, 25, (w, h))
    while (True):
        ok, img = video.read()
        cv2.imshow('img', img)
        video_writer.write(img)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    video.release()
    cv2.destroyAllWindows()
```

Листинг 4 – Чтение видеопотока с IP камеры и запись в файл

Далее на листинге 5 и рисунке 8 приведен пример чтения информации с webкамеры и изображение.

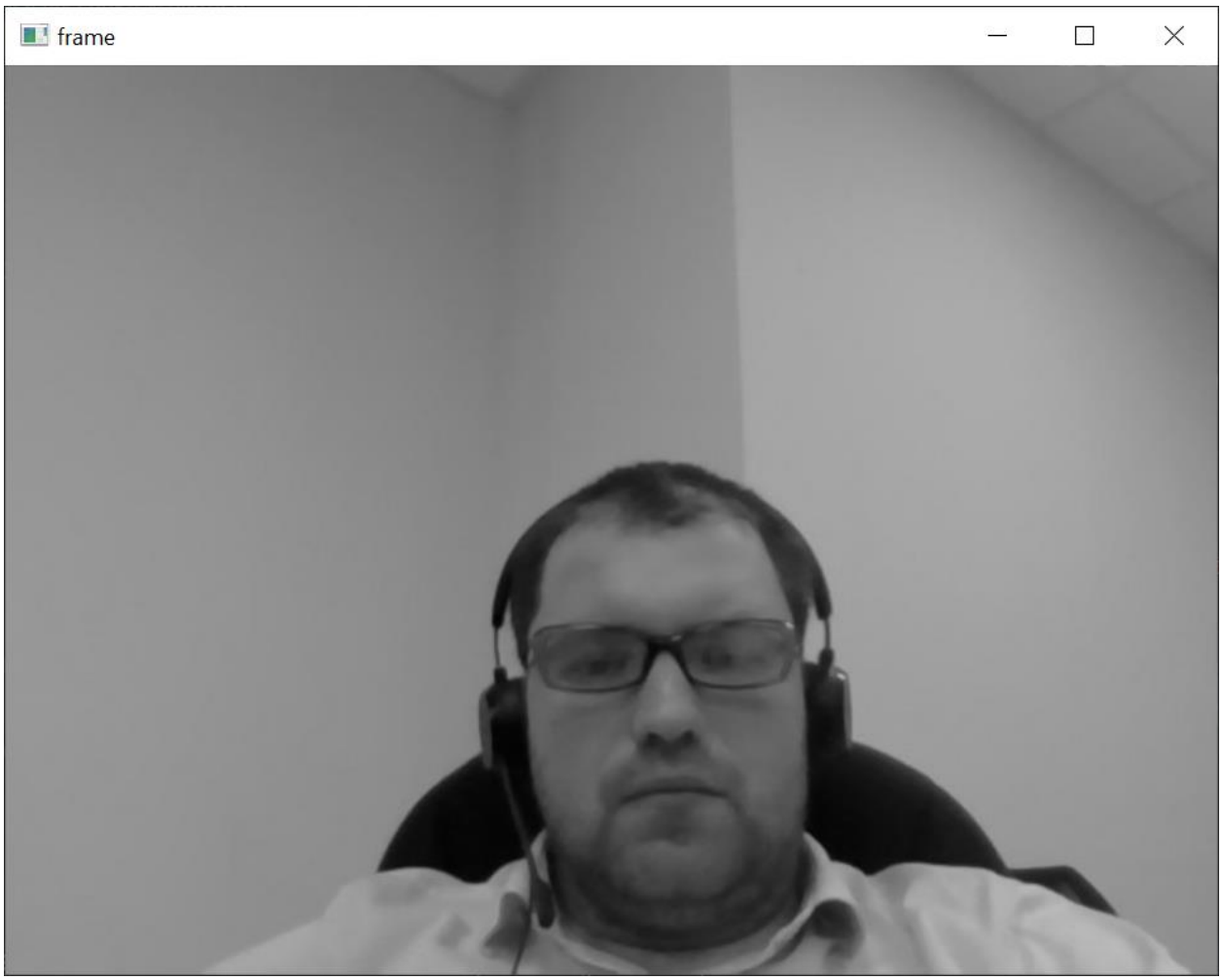



Рисунок 8 – Отображение черно-белого видео с вебкамеры

```
    def print_cam():
cap = cv2.VideoCapture(0)
cap.set(3, 640)
cap.set(4, 480)
while True:
  ret, frame = cap.read()
  # Convert to grayscale
  gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
  # Display the resulting frame
  cv2.imshow('frame', gray)
  if cv2.waitKey(1) & 0xFF == 27:
    break
# When everything done, release the capture
```

```
cap.release()
cv2.destroyAllWindows()
```

Листинг 5 – Отображение черно-белого видео с вебкамеры

Далее рассмотрим формат изображения HSV. Откройте изображение и сохраните его в переменную frame. Далее переведите его в формат HSV с помощью команды ниже

hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

Отобразите фрейм hsv. Посмотрите, как система отображает этот фрейм. Разберите структуру формата.

Выполните задания 5 и 6.

Контрольные вопросы

1. Опишите формат представления изображений в библиотеке OpenCv
2. Опишите принцип отображения окон в OpenCV
3. Каким образом возможно управлять параметрами отображения окон в OpenCV
4. Каким класс отвечает за работу с видеопотоком?
5. Откуда возможно получение видеопотока?
6. Каковы общие принципы работы с изображениями в потоке? Опишите понятие frame и принцип работы метода read()
7. Что такое fourcc? Зачем применяется?
8. Опишите основные особенности класса video_writer?
9. Что же значит эта проверка «cv2.waitKey(1) & 0xFF == 27»?
10. Объяснить, зачем применяется формат HSV, рассказать значения каждого из параметров, найти и указать формулы перевода из HSV в RGB и обратно. Объяснить геометрический смысл таких преобразований.

Итоговый список заданий:

Задание 1. Установить библиотеку OpenCV.

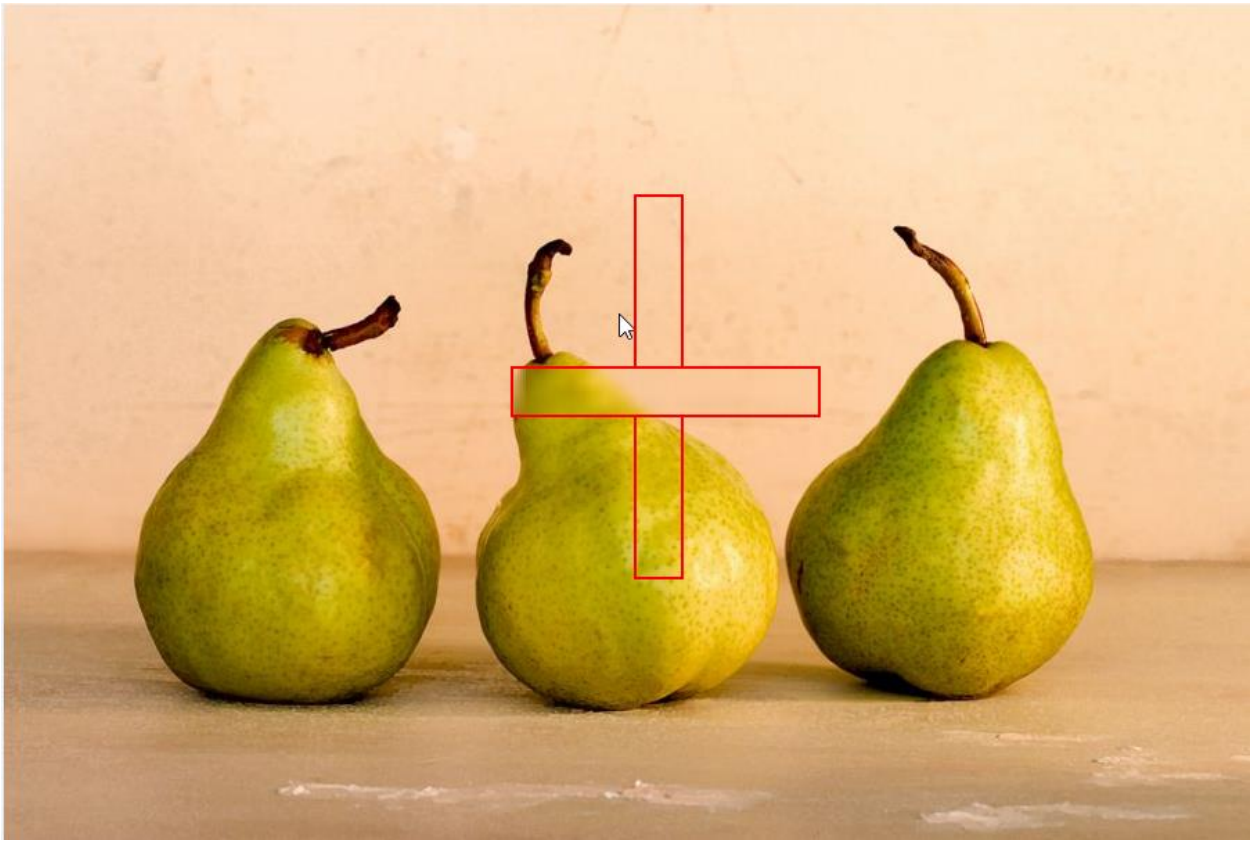
Задание 2. Вывести на экран изображение. Протестировать три возможных расширения, три различных флага для создания окна и три различных флага для чтения изображения.

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

Задание 4. Записать видео из файла в другой 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.

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.



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

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.

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

Формат оценивания выполнения заданий на лабораторной работе:

- оценка «+» ставится на лабораторной работе, если студент выполняет задания 1-5;

- оценка «удовлетворительно» ставится на лабораторной работе, если студент выполняет задания 1-7;

- оценка «хорошо» ставится на лабораторной работе, если студент выполняет все задачи;

- оценка «отлично» ставится на лабораторной работе, если студент отвечает правильно на все теоретические вопросы.

Если студент сдаёт работу позже, то применяется формат оценивания, указанный в документе «Структура лаб АЦОМ», то есть необходим отчет, гит и полноценная защита лабораторной работы.