**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра ВТ**

**ОТЧЁТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №5**

**ПО ДИСЦИПЛИНЕ**

**«Обработка изображений и компьютерное зрение (OpenCV)»**

**ТЕМА: «БАЗОВЫЕ ФУНКЦИИ ДЛЯ РЕШЕНИЯ ЗАДАЧИ ДЕТЕКТИРОВАНИЯ ЛИЦ НА ПОТОКЕ ВИДЕОДАННЫХ»**

**Бригада №1**

| Студенты гр. 8307: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Лукьянов А.В. |
| --- | --- | --- |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Якушев Г.С. |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Крылов А. |
| Преподаватель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Костичев С.В. |
|  |  |  |

Санкт-Петербург

2023

**Задание на лабораторную работу:**

Выполнить следующие базовые операции

Загрузка видео из файла.

1. Открыть видео файл Jalinga\_360.mp4 (метод open), вывести его содержимое в окно. Для проверки открытия видеопотока используйте метод isOpened).
2. Извлечение кадров видеопотока

Дополните программу из предыдущего пункта покадровым выводом видеопотока в окно. Для этого можно использовать перегруженную операцию перенаправления потока ввода >>. Определите количество кадров в видеопотоке.

1. Сохранение кадров видеопотока в видеофайл.

Дополните программу из предыдущего пункта сохранением кадров видеопотока в новый видеофайл.

1. Детектирование лиц на изображении с использованием классификатора Хаара.
   1. Используйте классификатор Хаара для одного изображения

Загрузите и отобразите изображение lena.jpg в окне "window1"

Загрузите натренированную модель лица (классификатор Хаара haarcascade\_frontalface\_alt2.xml) с помощью метода load.

Чтобы найти лица разного масштаба на изображении, используйте метод detectMultiScale.

Отобразите результат детектирования лица в виде прямоугольника и эллипса.

* 1. Используйте классификатора Хаара для видео потока.

Для этого дополните программу из пункта 2 операциями из пункта 4.1 для каждого кадра из видеопотока.

**Программное и аппаратное окружение при выполнении работы**

Поскольку члены нашей группы пользуются различными ОС, мы не смогли сделать выбор в пользу какой-то конкретной ОС, потому мы использовали машины под управлением ОС Ubuntu и ОС Windows 11, а также среду разработки CLion для языков C/C++ с поддержкой кроссплатформенной системой сборки CMake.

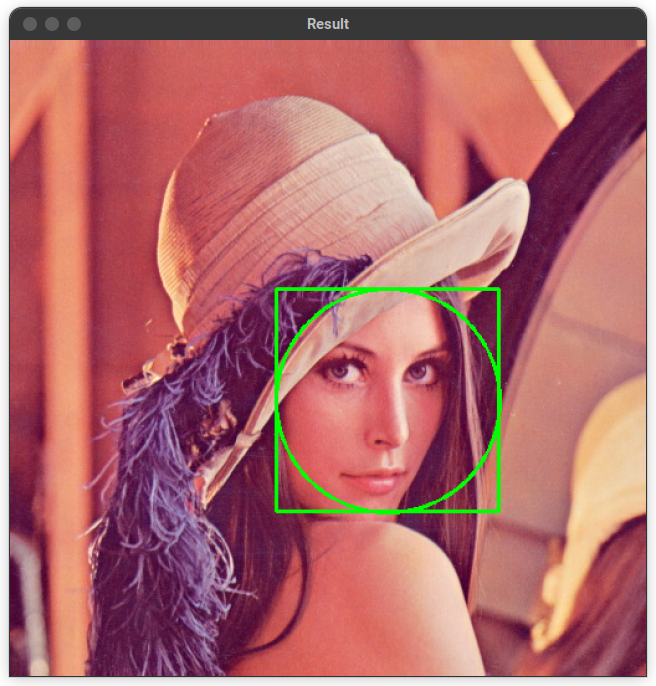
Аппаратное обеспечение варьируется довольно сильно - в нашем распоряжении оказались процессоры Intel Core i5-6600(4C|4T, 32 gb ram), i5-12600k(10c|16T, 32gb ram + 40 gb swap), а также AMD A10-9620P(4C|4T, 6gb ram + 20gb swap).

При работе мы использовали CLion версии 2022.3.3., CMake 3.24, libopencv 4.7.0-dev, libopencv-4.0.1-vmix (пакет OpenCV с персонального репозитория НПП Видеомикс (место работы одного из студентов)).

**Пример запуска программы**

К сожалению, ни один из пакетов OpenCV, даже под OC Windows не захотел открывать приложенное видео. В случае с ОС Ubuntu данную проблему, потенциально, можно решить, перекомилировав пакет OpenCV в ручном режиме с использованием инструментов OpenCL и ffmpeg.

По этой причине часть работы оказалось выполнить невозможно. Проблема повторилась как при использовании Python, так и при использовании C++.

  
Рисунок 1 - Результат детектирования лица на Лене.

**Заключение**

Как было сказано выше, в рамках выполнения данной работы возникли сложности, решить которые не представилось возможным. Всевозможные попытки добавить в текущую ОС недостающие кодеки или перекомпилировать библиотеку не увенчались успехом. Как итог, нами было написано 2 программы. В первой программе предлагается реализация извлечения видеопотока из файла с дальнейшим сохранением в новый файл, во второй - детекция лиц на фото и видеоряде. В случае со второй программой проблем возникло чуть меньше и нам, в рамках решения задачи из пункта 4 нам всё же удалось опробовать классификатор Хаара. Также в первой программе нами был перегружен оператор вывода “<<”, который позволяет использовать стандартный поток вывода для отрисовки видеоряда.

**ПРИЛОЖЕНИЕ А.**

**Текст программы задания 1-2-3**

**#include <iostream>**

**#include <opencv2/opencv.hpp>**

**#include <opencv2/core.hpp>**

***// Перегрузка операции << для покадрового вывода видеопотока в окно***

**std::ostream& operator<<(std::ostream& stream, const cv::Mat& frame)**

**{**

**cv::imshow("Видео", frame);**

**cv::waitKey(1); *// Задержка между кадрами***

**return stream;**

**}**

**int main()**

**{**

***// Загрузка видео из файла***

**cv::VideoCapture video("/media/artyom/Новая папка/ETU/10\_sem/OpenCV/Jalinga\_360.mp4", cv::*CAP\_FFMPEG*);**

**try**

**{**

***// Проверка открытия видеопотока***

**if (!video.isOpened())**

**{**

**std::cout << "Не удалось открыть видео файл" << std::endl;**

**}**

**}**

**catch( cv::Exception& e )**

**{**

**const char\* err\_msg = e.what();**

**std::cout << "exception caught: " << err\_msg << std::endl;**

**return 1;**

**}**

***// Извлечение кадров видеопотока и вывод в окно***

**cv::Mat frame;**

**int frameCount = 0;**

**while (video.read(frame))**

**{**

***//Показ видео***

**std:: cout << &frame;**

***// Сохранение кадров в новый видеофайл***

**cv::VideoWriter writer("output.mp4", cv::VideoWriter::fourcc('M', 'J', 'P', 'G'), 30, frame.size());**

**writer.write(frame);**

**writer.release();**

**frameCount++;**

**}**

**std::cout << "Количество кадров: " << frameCount << std::endl;**

***// Закрытие видеопотока и окна***

**video.release();**

**cv::destroyAllWindows();**

**return 0;**

**}**

**ПРИЛОЖЕНИЕ А.**

**Текст программы задания 4**

**#include <functional>**

**#include <iostream>**

**#include <string>**

**#include <utility>**

**#include <opencv2/opencv.hpp>**

***// Перегрузка операции << для покадрового вывода видеопотока в окно***

**std::ostream& operator<<(std::ostream& stream, const std::pair<std::string,cv::Mat>& frame)**

**{**

**cv::imshow(frame.first, frame.second);**

**cv::waitKey(30); *// Задержка между кадрами***

**return stream;**

**}**

**std::ostream& operator<<(std::ostream& stream, const std::pair<std::string,cv::Mat&>& frame)**

**{**

**cv::imshow(frame.first, frame.second);**

**cv::waitKey(); *// Задержка между кадрами***

**return stream;**

**}**

**int main()**

**{**

***// Загрузка и отображение изображения lena.jpg***

**cv::Mat image = cv::imread("/media/artyom/Новая папка/ETU/10\_sem/OpenCV/lena.jpg");**

***// Загрузка натренированной модели лица (классификатор Хаара)***

**cv::CascadeClassifier faceCascade;**

**if (!faceCascade.load("/usr/share/opencv4/haarcascades/haarcascade\_frontalface\_alt2.xml")) {**

**std::cout << "Не удалось загрузить классификатор Хаара" << std::endl;**

**return 1;**

**}**

***// Преобразование изображения в оттенки серого***

**cv::Mat grayImage;**

**cv::cvtColor(image, grayImage, cv::*COLOR\_BGR2GRAY*);**

**cv::equalizeHist(grayImage, grayImage);**

***// Детектирование лиц на изображении***

**std::vector<cv::Rect> faces;**

**faceCascade.detectMultiScale(grayImage, faces);**

***// Отображение результатов детектирования***

**for (const cv::Rect& face : faces) {**

***// Выделение прямоугольником***

**cv::rectangle(image, face, cv::Scalar(0, 255, 0), 2);**

***// Выделение эллипсом***

**cv::Point center(face.x + face.width / 2, face.y + face.height / 2);**

**cv::Size axes(face.width / 2, face.height / 2);**

**cv::ellipse(image, center, axes, 0, 0, 360, cv::Scalar(0, 255, 0), 2);**

**}**

**std::string res = "Resuult";**

**cv::namedWindow(res, cv::*WINDOW\_NORMAL*);**

**std::cout << std::make\_pair(res, std::ref(image));**

***// 4.2 Детектирование лиц в видеопотоке с использованием классификатора Хаара***

***// Загрузка видео из файла***

**cv::VideoCapture video("/media/artyom/Новая папка/ETU/10\_sem/OpenCV/Jalinga\_360.mp4");**

***// Проверка открытия видеопотока***

**if (!video.isOpened()) {**

**std::cout << "Не удалось открыть видео файл" << std::endl;**

**return 1;**

**}**

**cv::Mat frame;**

**try**

**{**

**while (video.read(frame))**

**{**

***// Преобразование кадра в оттенки серого***

**cv::cvtColor(frame, grayImage, cv::*COLOR\_BGR2GRAY*);**

**cv::equalizeHist(grayImage, grayImage);**

***// Детектирование лиц на кадре***

**faces.clear();**

**faceCascade.detectMultiScale(grayImage, faces);**

***// Отображение результатов детектирования***

**for (const cv::Rect& face : faces) {**

***// Выделение прямоугольником***

**cv::rectangle(frame, face, cv::Scalar(0, 255, 0), 2);**

***// Выделение эллипсом***

**cv::Point center(face.x + face.width / 2, face.y + face.height / 2);**

**cv::Size axes(face.width / 2, face.height / 2);**

**cv::ellipse(frame, center, axes, 0, 0, 360, cv::Scalar(0, 255, 0), 2);**

**}**

**std::cout << std::make\_pair(std::string("Video"), frame);**

**}**

**}**

**catch( cv::Exception& e )**

**{**

**const char\* err\_msg = e.what();**

**std::cout << "exception caught: " << err\_msg << std::endl;**

**return 1;**

**}**

**return 0;**

**}**