###### МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

###### ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

###### НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

###### Факультет информационных технологий

**Кафедра параллельных вычислений**

**ОТЧЕТ**

**О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

«ВЫСОКОУРОВНЕВАЯ РАБОТА С ПЕРИФЕРИЙНЫМИ УСТРОЙСТВАМИ»

студента 2 курса, группы 23201

Смирнова Гордея Андреевича

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

Матвеев Алексей Сергеевич

Новосибирск 2024

**Цель**

1. Ознакомиться с программированием периферийных устройств на примере ввода данных с Web-камеры с использованием библиотеки OpenCV;

2. Применить полученные знания в реализации программы вывода в окно искаженного изображения с камеры.

**Задание**

Изучить принципы работы OpenCV, ознакомиться с работой с растровыми изображениями и их форматами, работой с видеоданными и их форматами, интерфейсом для работы с камерами, реализацией упрощенного оконного интерфейса, операций над векторами и матрицами. Реализовать с помощью полученных знаний об этой библиотеке программу, которая будет принимать видеосигнал с камеры, распознавать лица на видеопотоке и менять их местами. Замерить время, затрачиваемое на получение, обработку и вывод кадров на окно программы. Провести анализ полученных данных.

**Листинг программы**

#include <opencv2/opencv.hpp>

#include <iostream>

#include <vector>

int swapFaces(const cv::Mat& img) {

cv::CascadeClassifier face\_cascade;

if (!face\_cascade.load("/usr/share/opencv4/haarcascades/haarcascade\_frontalface\_default.xml")) {

return -1;

}

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

face\_cascade.detectMultiScale(img, faces, 1.2, 5, 0, cv::Size(30, 30));

if (faces.size() >= 2) {

cv::Mat face1 = img(faces[0]);

cv::Mat face2 = img(faces[1]);

cv::Mat resized\_face1, resized\_face2;

resize(face1, resized\_face1, faces[1].size());

resize(face2, resized\_face2, faces[0].size());

resized\_face1.copyTo(img(faces[1]));

resized\_face2.copyTo(img(faces[0]));

}

return static\_cast<int>(faces.size());

}

int main() {

cv::VideoCapture cap(0);

if (!cap.isOpened()) {

std::cerr << "Error opening video stream" << std::endl;

return -1;

}

cv::Mat frame;

int frameCount = 0;

auto start = std::chrono::system\_clock::now();

std::string fpsText, currentTime;

while (true) {

cap >> frame;

if (frame.empty()) break;

std::string faceCountText = "Face count: " + std::to\_string(swapFaces(frame));

frameCount++;

if (frameCount % 25 == 0) {

auto current = std::chrono::high\_resolution\_clock::now();

std::chrono::duration<double> elapsed = current - start;

start = current;

double fps = frameCount / elapsed.count();

frameCount = 0;

fpsText = "FPS: " + std::to\_string(fps);

}

cv::putText(frame, fpsText, cv::Point(10, 30), cv::FONT\_HERSHEY\_SIMPLEX, 1.0, cv::Scalar(95, 230, 32), 3);

cv::putText(frame, faceCountText, cv::Point(10, 60), cv::FONT\_HERSHEY\_SIMPLEX, 1.0, cv::Scalar(95, 230, 32), 3);

imshow("FaceSwap", frame);

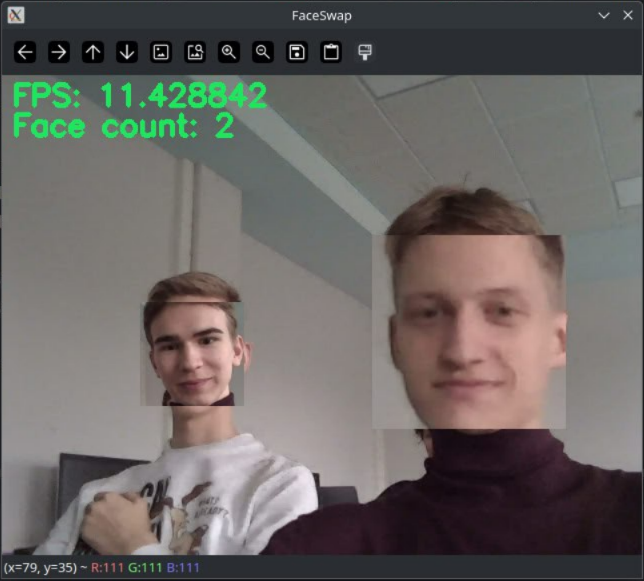
if (cv::waitKey(30) >= 0) break;

}

return 0;

}

**Результат работы программы:**

****

Программа обнаруживает лица на изображении и меняет местами (совершая подгонку соответствующих размеров) первые два обнаруженные лица.

**Выводы**

В ходе работы с OpenCV были освоены:

работа с камерой;

работа с изображениями, полученными с камеры;

работа с функциями, работающими с предобученными моделями распознавания лиц;

вывод обработанного изображения на экран и работа с окнами в OpenCV.