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 ПРАКТИЧЕСКОЙ РАБОТЫ

Высокоуровневая и низкоуровневая работа с периферийными устройствами

студентки 2 курса, группы 21207

Черновской Яны Тихоновны

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:
А.Ю. Власенко

Новосибирск 2022

СОДЕРЖАНИЕ

ЦЕЛЬ

1. *Ознакомиться с программированием периферийных устройств на примере ввода данных с Web-камеры с использованием библиотеки OpenCV.*
2. *Ознакомиться с началами низкоуровневого программирования периферийных устройств на примере получения информации о доступных USB-устройствах с помощью библиотеки libusb*

ЗАДАНИЕ

1. Реализовать программу№1 с использованием OpenCV, которая получает поток видеоданных с камеры и выводит его на экран.

2. Выполнить произвольное преобразование изображения (кроме указанных в computerlab5.pdf сглаживания и установки значений цветовых каналов в константу).

3. Измерить количество кадров, обрабатываемое программой в секунду.

Оценить долю времени, затрачиваемого процессором на обработку (ввод, преобразование, показ) видеоданных, получаемых с камеры.

4. Реализовать программу №2, получающую список всех подключенных к машине USB устройств с использованием libusb. Для каждого найденного устройства напечатать его класс, идентификатор производителя, идентификатор изделия и серийный номер.

5. Требования к содержанию отчета:

- «чистое» неоткорректированное изображение, полученное с камеры;

- это же изображение в преобразованном виде;

- полный код программы №1, выполняющей преобразование изображения;

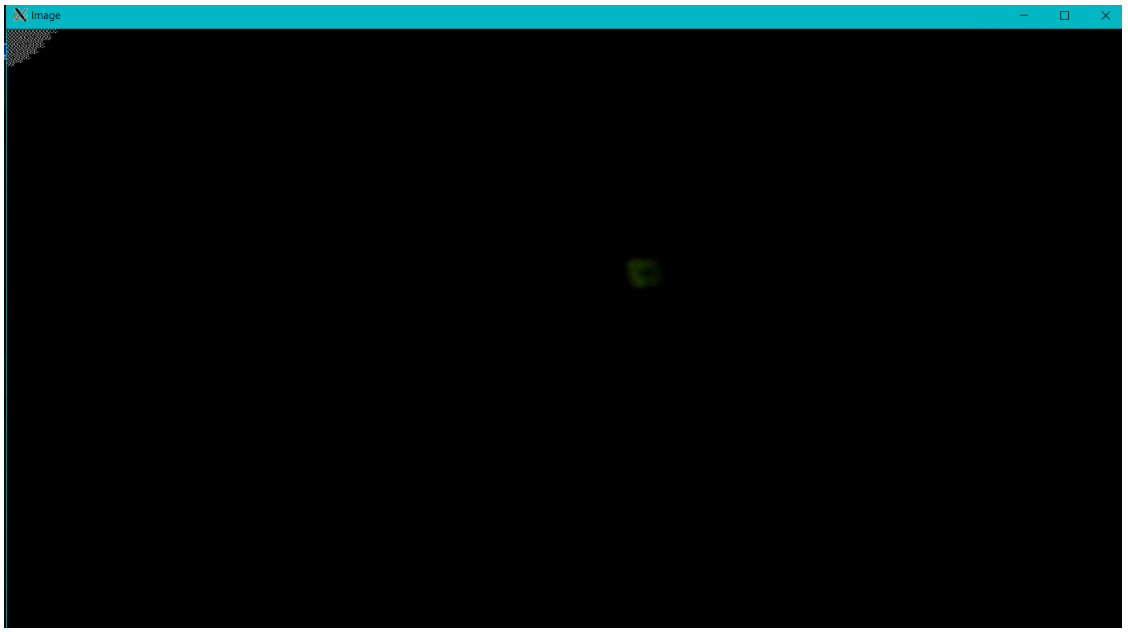
- оценку скорости обработки видео (кадров в секунду) и долю времени, затрачиваемого процессором на ввод, обработку и показ видеоданных;

6. Полный код программы №2, выводящей информацию по USB-устройствам.

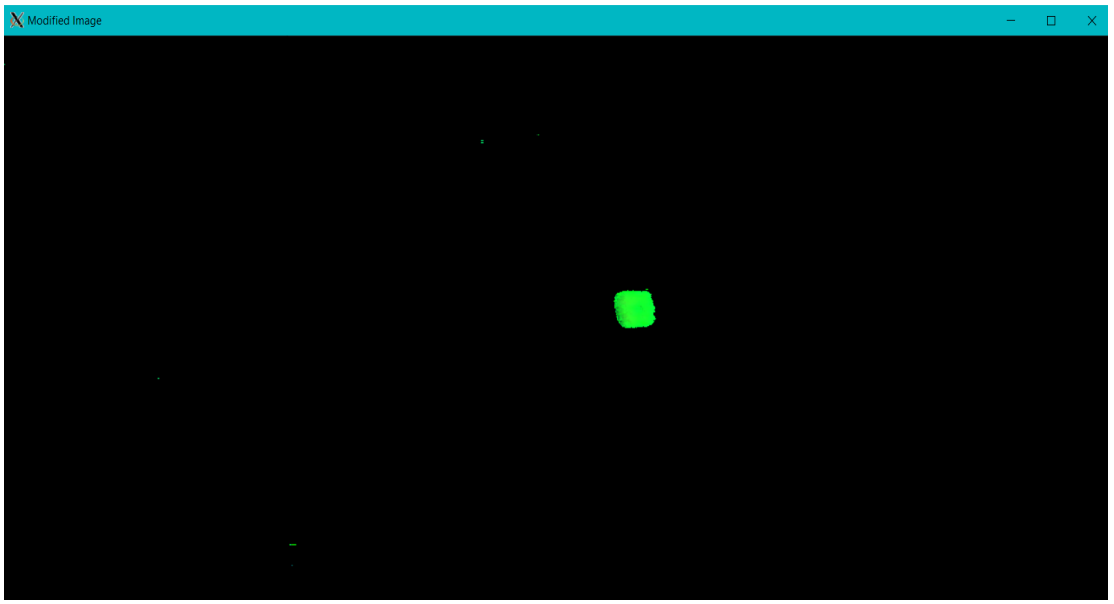
7. Описание обнаруженных USB-устройств

Описание высокоуровневой работы с периферийными устройствами

1. Для работы с OpenCV был выбран второй вариант: работа с USB - камерой, подключенной к серверу, был скачан X-сервер для Windows.
2. Для получения “чистого” не откорректированного изображения использовался файл DisplayImage из папки opencv в домашней директории пользователя evmpu.



3. Далее было выполнено преобразование изображения и замерено время выполнения программы.



На данном примере сложно понять, как именно работает функция. Пример результата обработки изображения для освещённой фотографии:


```
Frame rate: 11.6227
Input time: 2.42998%
Processing time: 18.2823%
Output time: 24.5081%
```

Описание низкоуровневой работы с периферийными устройствами

Далее была написана программа для получения информации о USB устройствах.

```
evmpu@comrade:~/21207/Chernovskaya$ g++ lab3.cpp -lusb-1.0
evmpu@comrade:~/21207/Chernovskaya$ ./a.out
найдено устройств: 12
class: ef vendor: 046d product: 0825 serial number: 95410D90
class: 09 vendor: 1d6b product: 0002 serial number: 0000:00:1d.7
class: 09 vendor: 1d6b product: 0001 serial number: 0000:00:1d.2
class: 00 vendor: 1c4f product: 0026 null
class: 00 vendor: 0458 product: 003a null
class: 09 vendor: 1d6b product: 0001 serial number: 0000:00:1d.1
class: 09 vendor: 1d6b product: 0001 serial number: 0000:00:1d.0
class: 00 vendor: 0bda product: 0181 serial number: 20060413092100000
class: 09 vendor: 1d6b product: 0002 serial number: 0000:00:1a.7
class: 09 vendor: 1d6b product: 0001 serial number: 0000:00:1a.2
class: 09 vendor: 1d6b product: 0001 serial number: 0000:00:1a.1
class: 09 vendor: 1d6b product: 0001 serial number: 0000:00:1a.0
```

Расшифровка полученных значений:

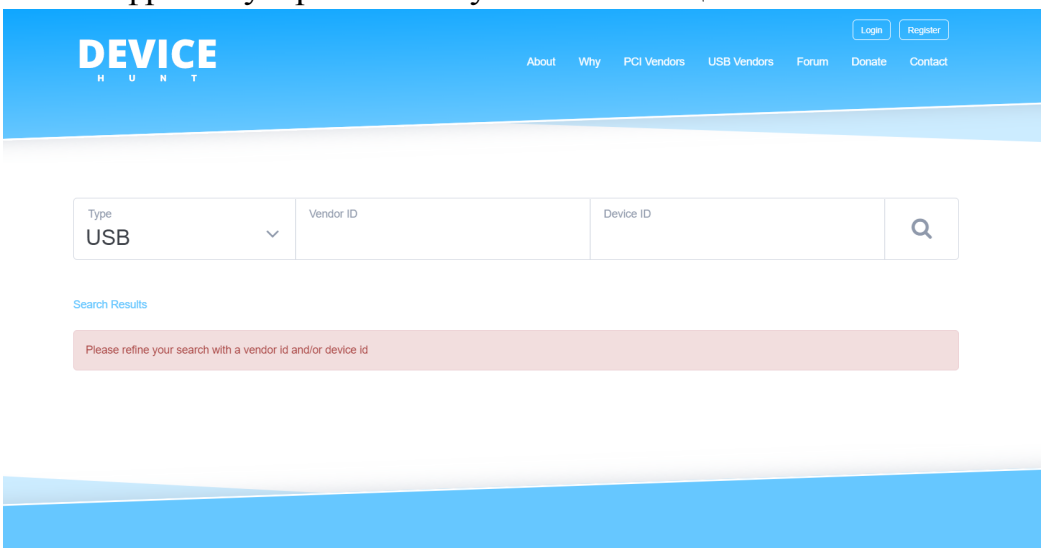
Расшифровка кодов классов:

00 – Unclassified device
09 – Input device controller
ef - “устройство класса различные устройства”

Расшифровка самих устройств:

046d 0825 - Webcam C270
1d6b 0002 - 2.0 root hub
1d6b 0001 - 1.1 root hub
1c4f 0026 - Keyboard
0458 003a - NetScroll+ Mini Traveler / Genius NetScroll 120 (мышка)
1d6b 0001 - 1.1 root hub
0bda 0181 - Realtek Semiconductor Corp.
1d6b 0002 - 2.0 root hub

*Расшифровка устройств получена с помощью сайта Device Hunt



5. Чтобы проверить корректность выводимых данных используем команду lsusb:

```
evmpu@comrade:~$ lsusb
Bus 002 Device 004: ID 046d:0825 Logitech, Inc. Webcam C270
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 008 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 007 Device 003: ID 1c4f:0026 SiGma Micro Keyboard
Bus 007 Device 002: ID 0458:003a KYE Systems Corp. (Mouse Systems) NetScroll+ Mini Traveler / Genius NetScroll 120
Bus 007 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 006 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 002: ID 0bda:0181 Realtek Semiconductor Corp. USB2.0-CRW
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

ЗАКЛЮЧЕНИЕ

В данной работе мы познакомились с программированием периферийных устройств на примере ввода данных с Web-камеры с использованием библиотеки OpenCV, а также с началами низкоуровневого программирования периферийных устройств на примере получения информации о доступных USB-устройствах с помощью библиотеки libusb.

Приложение 1 Полный листинг программы 1

```
#include <iostream> // for standard I/O
#include <string> // for strings
#include <iomanip> // for controlling float print precision
#include <sstream> // string to number conversion
#include <opencv2/core.hpp> // Basic OpenCV structures (cv::Mat, Scalar)
#include <opencv2/imgproc.hpp> // Gaussian Blur
#include <opencv2/videoio.hpp>
#include <opencv2/highgui.hpp> // OpenCV window I/O
#include <opencv2/imgcodecs.hpp>
#include <ctime>

using namespace std;
using namespace cv;

int main(int argc, char* argv[])
{
	VideoCapture cap(0);//captRefrnc(sourceReference);//, captUndTst(sourceCompareWith);

	Mat img;
	Mat newImage;
	int frameCounter = 0;
	double totalTime = 0, inputTime = 0, procTime = 0, outputTime = 0;

	while (true)
	{
		clock_t c_start = clock();
		cap.read(img);
		inputTime += (clock() - c_start);

		c_start = clock();
		cvtColor(img, newImage, COLOR_BGR2HSV);
		//Canny(img, newImage1, 100, 200)
		procTime += (clock() - c_start);

		c_start = clock();
		imshow("Modificed Image", newImage);
		outputTime += (clock() - c_start);

		frameCounter++;

		char c = waitKey(10);
		if (c == 27) break;
	}

	totalTime = clock();
	double percent = totalTime / 100.1;

	cout << "Frame rate: " << (frameCounter / (totalTime / CLOCKS_PER_SEC)) << endl;
	cout << "Input time: " << inputTime / percent << "%" << endl;
	cout << "Processing time: " << procTime / percent << "%" << endl;
	cout << "Output time: " << outputTime / percent << "%" << endl;
```

```
return 0;}
```

Приложение 1 Полный листинг программы 2

```
#include <libusb-1.0/libusb.h>
#include <cstdio>

void printdev(libusb_device* dev);

int main(){
    libusb_device** devs; // указатель на указатель на устройство, используется для получения списка устройств
    libusb_context* ctx = NULL; // контекст сессии libusb

    int r; // для возвращаемых значений
    ssize_t cnt; // число найденных USB-устройств
    r = libusb_init(&ctx); // инициализировать библиотеку libusb, открыть сессию работы с libusb

    if (r < 0){
        fprintf(stderr, "Ошибка: инициализация не выполнена, код: %d.\n", r);
        return 1;
    }

    // получить список всех найденных USB- устройств
    cnt = libusb_get_device_list(ctx, &devs);
    if (cnt < 0){
        fprintf(stderr, "Ошибка: список USB устройств не получен. Код: %d\n", r);
        return 1;
    }

    printf("найдено устройств: %ld\n", cnt);

    for (ssize_t i = 0; i < cnt; i++) { // цикл перебора всех устройств
        printdev(devs[i]); // печать параметров устройства
    }

    libusb_free_device_list(devs, 1); // освободить память, выделенную функцией получения списка устройств
    libusb_exit(ctx); // завершить работу с библиотекой libusb, закрыть сессию работы с libusb

    return 0;
}

void printdev(libusb_device* dev){
    libusb_device_descriptor desc{}; // дескриптор устройства
    libusb_device_handle* handle = NULL;

    unsigned char str[256];

    int r = libusb_get_device_descriptor(dev, &desc); // получить дескриптор
    if (r < 0){
        fprintf(stderr, "Ошибка: дескриптор устройства не получен, код: %d.\n", r);
        return;
    }

    printf("class: %.2x vendor: %.4x product: %.4x ",
```

```
        (int)desc.bDeviceClass,
        desc.idVendor,
        desc.idProduct
    );

    libusb_open(dev, &handle);
    if (handle && desc.iSerialNumber) {
        r = libusb_get_string_descriptor_ascii(handle, desc.iSerialNumber, str, sizeof(str));
        if (r != 0) {
            printf("serial number: %s\n", str);
        }
        else{
            printf("null\n");
        }
    }
}
```