МИНОБРНАУКИ РОССИИ
Санкт-Петербургский государственный
электротехнический университет
«ЛЭТИ» им. В.И. Ульянова (Ленина)
Кафедра МО ЭВМ

Курсовая РАБОТА
по дисциплине «Программировани»
Тема: «Обработка изображений»

Студент гр. 0304 _____ Мохаммед А.В.

Преподаватель _____ Чайка К.В.

Санкт-Петербург
2021

ЗАДАНИЕ

на курсовую работу

Студент Мохаммед А.В.

Группа 0304

Тема работы: «Обработка изображений»

Исходные данные:
Программа должна читать и записывать изображения PNG. Изображение должно быть обработано с использованием указанных функций. Программа должна иметь CLI. В программе должна быть реализована обработка ошибок.

Содержание пояснительной записки:
«Содержание», «Введение», «Считывание и запись изображения», «Обработка изображения»,«Заключение», «Список использованных источников»

Предполагаемый объем пояснительной записки:
Не менее 12 страниц.

Дата выдачи задания: 24.05.2021

Дата сдачи реферата:

Дата защиты реферата:

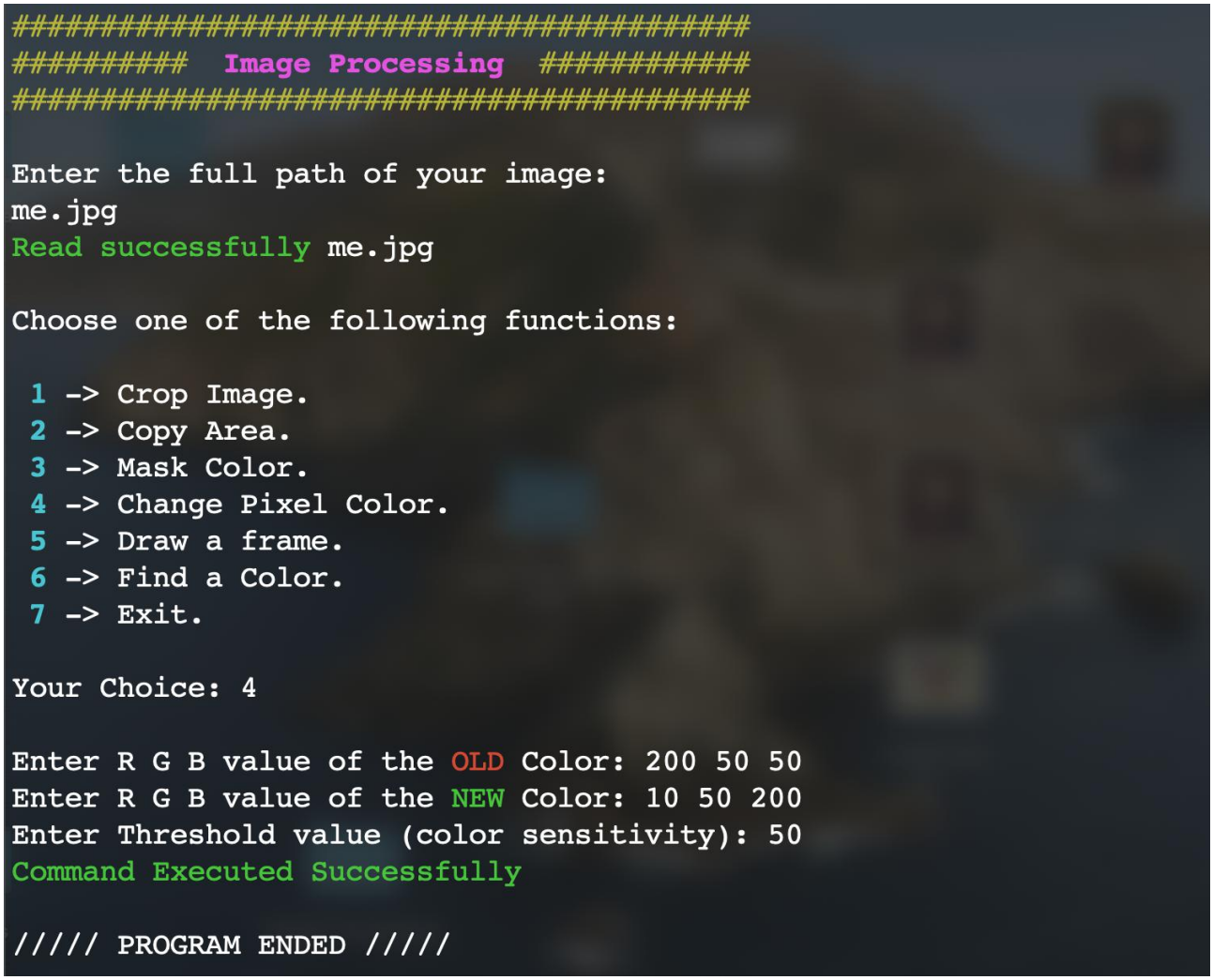
Студент _____ Мохаммед А.В.

Преподаватель _____ Чайка К.В.

Аннотация

Была поставлена задача создать приложение, которое будет обладать функционалом считывания изображения в формате PNG, его записью в файл, а также функциями для обработки данного изображения.

Программа состоит из заголовочных файлов библиотеки, заголовочного файла структур изображений и файлов исходного кода. Таким образом была разделена логика взаимодействия с пользователем (Интерфейс командной строки) и логика работы с изображениями (считывание, запись, обработка изображений в формате PNG).



Screenshot: Программный интерфейс

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

1.1. Цель курсовой работы и исходные условия

Целью данной работы является создание программы, соответствующей условию задания:

Условие задания:

● **Формат картинки PNG (рекомендуем использовать библиотеку libpng)**

● файл всегда соответствует формату PNG

● обратите внимание на выравнивание; мусорные данные, если их необходимо дописать в файл для выравнивания, должны быть нулями.

● все поля стандартных PNG заголовков в выходном файле должны иметь те же значения что и во входном (разумеется кроме тех, которые должны быть изменены).

Программа должна реализовывать следующий функционал по обработке PNG-файла

1. Копирование заданной области. Функционал определяется:
    - ○ Координатами левого верхнего угла области-источника
    - ○ Координатами правого нижнего угла области-источника
    - ○ Координатами левого верхнего угла области-назначения
2. Заменяет все пиксели одного заданного цвета на другой цвет. Функционал определяется:
    - ○ Цвет, который требуется заменить
    - ○ Цвет на который требуется заменить
3. Сделать рамку в виде узора. Рамка определяется:
    - ○ Узором (должно быть несколько на выбор. Красивый узор можно получить используя фракталы)
    - ○ Цветом
    - ○ Шириной
4. Поиск всех залитых прямоугольников заданного цвета. Требуется найти все прямоугольники заданного цвета и обвести их линией. Функционал определяется:
    - ○ Цветом искомых прямоугольников
    - ○ Цветом линии для обводки
    - ○ Толщиной линии для обводки

1.2 Задачи

Для достижения поставленной цели требуется решить следующие задачи:

- Создание файловой структуры программы — разбиение программы на подпрограммы для разделения логики работы с изображением и логики работы с пользователем.
- Разработка программного кода, выполняющего необходимые задачи.
- Написание Makefile для сборки программы.
- Сборка и тестирование программы.

2. СЧИТЫВАНИЕ И ЗАПИСЬ ИЗОБРАЖЕНИЯ

2.1. Структура для хранения изображения

Информация о фотографии сохраняется в структуре, содержащей разные типы данных для разной информации.

```
uint8_t* data = NULL;
size_t size = 0;
int w;
int h;
int channels;
```

Данные (информация о пикселях) хранятся в массиве типа unit8_t размером (ширина * высота * каналы). size_t переменная для хранения размера. переменные int для ширины, высоты и каналов.

Объявление нескольких функций-конструкторов, копирование другого изображения и создание нового изображения с определенными размерами. Объявление деконструктора. Объявление вспомогательных функций и основных функций.

2.2. Считывание из файла

чтение файла изображения выполняется путем вызова (stbi_load) из файла библиотеки (stb_image.h) и передачи полного пути изображения в качестве аргумента. Библиотека будет обрабатывать чтение заголовка, знание правильного типа файла и сохранение всей необходимой информации, такой как ширина, высота и количество каналов.

2.3. Запись в файл

Программа позволяет сохранить результат в четырех различных типах файлов изображений (PNG, BMP, JPG, TGA). После указания пути для сохранения, имени файла и расширения файла программа считывает расширение, вызывая функцию (get_file_type), и выбирает правильную функцию из библиотеки для сохранения с требуемым типом файла.

2.4. Получение информации о файле

Когда вызывается функция (print_img_info), на экран выводится основная информация, такая как ширина, высота и количество каналов.

3. ОБРАБОТКА ИЗОБРАЖЕНИЙ

Все функции обработки изображений объявлены в заголовочном файле Image.h и определены в исходном файле Image.cpp.

3.1 Вспомогательные функции

Метод конструктора *Image(const char* filename)* , который принимает путь и имя файла в качестве аргумента и вызывает функцию (read()) для файла. Печатает “read successfully” и устанавливает размер (ширина * высота * каналы). Или печатает “Failed to read” в условиях сбоя.

метод конструктора *Image(int w, int h, int channels)* , который принимает в качестве аргументов ширину, высоту и количество каналов и возвращает черное изображение с этими характеристиками.

Метод конструктора *(const Image& img)* который берет ссылку на другое изображение и копирует информацию об изображении в новую ячейку памяти с передачей той же информации о ширине, высоте, каналах и размере.

Метод Разрушитель *~Image()* удаляет информацию об изображении, вызывая stbi_image_free из библиотеки STB.

Функция *read(const char* filename)*, который вызывает stbi_load из библиотеки и передает имя файла в качестве аргумента, возвращает логическое значение успешного чтения.

Функция *write(const char* filename)* вызывает функции get_file_type, чтобы определить расширение желаемого типа изображения, которое нужно сохранить.

Функция *get_file_type(const char* filename)* использует *strrchr* для поиска определенного расширения, если не нашла. она вернет расширение файла PNG.

Функция *print_img_info()* для отображения некоторой информации об изображении, такой как ширина, высота и каналы.

Функция *resize(int x, int y)*, которая принимает новую ширину и новую высоту в качестве аргументов и пытается вызвать *stbir_resize_uint8* для применения к алгоритмам изменения размера. Эта функция работает некорректно и требует некоторых изменений, т.к. в результате получается искаженное изображение. Он был написан для будущей реализации.

3.2. Цветная маска

Функция *color_mask(float r, float g, float b)*

Изменяет насыщенность цветов всего изображения.
принимает три аргумента, представляющих коэффициенты для значений Red, Green и Blue. Эти коэффициенты имеют значение от 0 до 1. Функция работает с изображениями, имеющими три или более цветовых канала. Если каналов меньше трех, функция выведет ошибку на экран.

Принцип работы функции заключается в умножении каждого значения цвета на коэффициент, введенный пользователем для каждого пикселя изображения.

3.3. Сменить цвет

Функция *change_color(int f_r, int f_g, int f_b, int to_r, int to_g, int to_b, int threshold)*

Пользователь выбирает цвет, который нужно изменить, функция будет искать все пиксели в изображении, которые имеют одинаковые значения R G B, и изменяет их на желаемый новый цвет.

С возможностью выбора порогового значения любой пиксель в пределах порогового значения будет изменен. Чтобы изменить только точно выбранный пиксель, введите 0 в качестве порога. Функция не изменит цвет изображений менее трех каналов и в этом случае выдаст ошибку.

3.4. Обрезать

Функция *crop(uint16_t cx, uint16_t cy, uint16_t cw, uint16_t ch)*

Вырезать область из изображения и сохранить ее в другом изображении. если вырезанная область больше исходного изображения, новые пиксели будут заполнены черным цветом. Функция принимает начальную позицию в виде значений (x и y), затем запрашивает у пользователя ширину и высоту области.

После вычисления нового размера, выделения памяти и копирования информации о новых пикселях функция вернет ссылку на информацию о новом изображении.

3.5. Область копирования

Функция *copy_area(uint16_t cx, uint16_t cy, uint16_t cw, uint16_t ch, uint16_t to_x, uint16_t to_y)*

Для копирования определенной области изображения с определенной шириной и высотой. функция принимает координаты и размеры требуемой области путем ввода пользователем, а затем копирует пиксели в желаемое место.

Принцип работы функции прост. Вычисление размера желаемой области, а затем выделение временной памяти для сохранения информации о пикселях.
затем снова скопируйте пиксели с желаемой информацией. Функция возвращает ссылку на новое изображение.

3.6. Оверлей

Функция *overlay(const Image& source, int x, int y)*

Принцип работы функции несложен для понимания, это простое копирование информации о пикселях с одного изображения на другое изображение с учетом ширины и высоты. Но когда мы начинаем рассматривать случаи разного количества каналов между двумя изображениями, все немного усложняется.

При объединении двух изображений с разными значениями альфа-канала значения R G B и Alpha необходимо рассчитывать по определенным формулам. Функция возвращает ссылку на новое изображение.

3.7. Нарисуйте рамку

Функция *frame(int ptrn_num, int frame_size, int r, int g, int b)*

Кадры обрабатываются как изображения, используя функцию наложения, мы можем скопировать определенное изображение несколько раз, чтобы создать узор вокруг изображения.

Теперь представлены четыре формы рамок, каждый кадр имеет четыре размера (50 пикселей, 100 пикселей, 200 пикселей, 300 пикселей) с возможностью добавления бесконечного количества кадров и размеров.

Пользователь также может указать цвет рамки, введя значения R G B. Функция с построит путь для требуемого фрейма, затем прочитает этот фрейм. Скопируйте рамку в темпераментное место и примените функцию change_color.

Функция выполняет точные вычисления положения и количества копий, чтобы получить лучший и самый красивый результат.

3.8. В поисках цвета

Функция *find_color(uint8_t srch_r, uint8_t srch_g, uint8_t srch_b, uint8_t border_r, uint8_t border_g, uint8_t border_b, uint8_t thickness)*

Эта функция отвечает за поиск определенного цвета и рисование границы вокруг него с определенной толщиной и цветом, выбранным пользователем.
Функция принимает в качестве аргументов (искомые значения цвета, значения цвета границы, значение толщины границы). Функция возвращает ссылку на новое изображение.

Функция имеет несколько условных операторов, таких как (проверка, находится ли пиксель вне границ изображения), поэтому она может правильно обрабатывать большинство случаев.

ЗАКЛЮЧЕНИЕ

Была разработана программа, выполняющая поставленные задачи. Данная программа была разделена на подпрограммы, выполняющие задачи считывания, записи и обработки изображения и задачи взаимодействия с пользователем посредством интерфейса командной строки. При этом был реализован требуемый функционал: изменение определенного цвета, копирование области изображения, рисование рамки вокруг изображения и рисование границы вокруг цвета.

Функционал считывания и записи изображения был реализован при помощи библиотеки STB . Обработка изображения производилась путем изменения карты пикселей. Программа успешно компилируется и работает на платформе MacOS .

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. GitHub репозиторий библиотеки STB // URL: https://github.com/nothings/stb

2. Официальная документация по c ++ // URL: https://www.cplusplus.com/reference/