Assignment 5 – Color Blindness Simulator

Mignot Mesele

Purpose

The purpose of the program is to take images and change their coloring to simulate deuteranopia. This gives non-color-blind users an experience to see colors, the way people with deuteranopia see them. The program reads and writes the image files to a buffer. This is then used to change the rgb values, using either the 575-nm equations or 475-nm equations. The entire program consists of three c files io.c, bmp.c, and colorb.c. Colorb.c is where the main implementation of the program occurs, using functions developed in io.c and bmp.c. It also handles command options that the user inputs to then relay the appropriate outcome. Io.c enables the program to open, read, and write files. It also creates buffers for reading and writing, which is why io.c has functions that free memory allocated and closes the files. Bmp.c uses the functions in io.c to read the data in a given file, change the rgb values, and write it to the output file. The program also uses a shell script to run every command possible with the given bmp files.

How to Use the Program

To use the program you'd simply type in the command argument, ./colorb along with a command option and certain parameters. This would specify whether to read a file, write a file, or print the help message. Options:

- -i: reads from input file. Requires a filename as an argument.
- -o: writes to output file. Requires a filename as an argument.
- -h: displays help message

You need to use the Makefile, which enables you to run the commands, make filename and make clean. The make command converts the c files into executable files to be able to use colorb.c and iotest.c. The make clean command removes all files created by make, clearing any issues that the object and executable files could cause.

Program Design

Data Structures

- arrays-This is used as the buffer to store data.
- heap-This is being used to access data in other files.
- Buffer-The buffer structure is created to store the file descriptor, the offset, which can be either the next valid byte, for reading a file, and the next empty location, when writing to a file, the number of bytes remaining in the buffer, and the buffer itself.
- color-This structure stores the rgb 8 bit values.
- bmp-This structure stores the height, width, palette, and a double pointer to a 2d array.

Algorithms

```
deuteranopia simulation algorithm
    for i from o to MAX_COLORS
        declare rgb values from palette and temp rgb
        declare sqle and selq
        sqle < selq
            use 575-nm equation to temp rgb
        selq > sqle
            use 475-nm equation to temp rgb
        reassign rgb with temp rgb
```

Function Descriptions

- Buffer *read_open(const char *filename)-This function takes a dereferenced pointer to a filename, opens the files, stores it as a file descriptor,and creates the read buffer, returning the buffer.
- void read_close(Buffer **pbuf)-This function takes a double pointer to a buffer structure, for reading, and free the contents, then the buffer itself.
- Buffer *write_open(const char *filename)-This function takes a filename, uses the creat system call, stores it as a file descriptor,and creates the write buffer, returning the buffer.
- void write_close(Buffer **pbuf)-This function takes a double pointer to a buffer struct, for write, uses the write system call to write any bytes in the buffer to the file from the file descriptor, and free the contents, then the buffer itself.
- bool read_uint8(Buffer *buf, uint8_t *x)-This function takes a buffer structure and a return value, grabs the first byte of the buffer, stores it to the return value, updates both the offset and number of bytes remaining, and returns true if the buffer isn't empty, otherwise false.
- bool read_uint16(Buffer *buf, uint16_t *x)-This function takes a buffer struct and a return value, calls read_uint8 twice to read 2 bytes, and returns false if both calls returns false. Else, it copies the second byte into a uint16_t variable, shifts the new variable to the left by 8 bits, uses binary or on the first byte into the uint16_t variable, updating the return value and returning true.
- bool read_uint32(Buffer *buf, uint32_t *x)-This function takes a buffer struct and a return value, calls read_uint16 twice to read 4 bytes, and returns false if both returns false. Else, it copies the second uint16_t variable into a uint32_t variable, shifts the new variable to the left by 16 bits, uses binary or on the first unt16_t variable into the uint32_t variable, updating the return value and returning true.
- void write_uint8(Buffer *buf, uint8_t x)-This function takes a buffer struct and a uint8_t value, stores the value into the next free space in the buffer, update the offset, and returns true if the buffer isn't full, otherwise false.
- void write_uint16(Buffer *buf, uint16_t x)-This function takes a buffer struct and a uint16_t value, calls write_uint8 twice to write 2 bytes. First with the uint16_t value, then the uint16_t value logical shifted to the right by 8. No output.
- void write_uint32(Buffer *buf, uint32_t x)-This function takes a buffer struct and a uint32_t value, calls write_uint16 twice to write 4 bytes. First with the uint32_t value, then the uint132_t value logical shifted to the right by 16. No output.
- void bmp_write(const BMP *bmp, Buffer *buf)-This function takes a bmp structure and a buffer structure, serializing a bmp image and creates a new bmp file. returns nothing.
- BMP *bmp_create(Buffer *buf)-This function takes a buffer structure to access the read and write functions, creates a bmp structure, reads a bmp file into the bmp struct, and returns a pointer to the bmp struct.

- void bmp_free(BMP **bmp)-This function takes a double pointer to a bmp structure, rounds the width, stored in bmp, of a bmp file,free all memory in bmp by index, and free both the 2d array and bmp itself.
- void bmp_reduce_palette(BMP *bmp)-This function takes a bmp structure, calculates new rgb, values using either the 575-nm or 475-nm equations, and updates the rgb values stored in bmp with the new ones. Returns nothing.

Results

My colorb ran as expected. It converted the images' rgb to simulate deuteranopia. I even tried putting in my own pictures and it did exactly that. In the Numeric results section have pictures of the color chooser bmp file and the deuteranopia version. From creating this program, I was fascinated by how there's an algorithm that takes any pixel and converts it into a pixel viewed by deuteranopia. I also gained an understanding of how to use unix system calls to open and read the data, in a given file, to write to another file with the help of buffers.

Error Handling

A huge error that I encountered was when my output images had a vertical line, that was of a different color, which I fixed by finding one of my for loops that I forgot to change it's bool operator from less than to less than or equal to. In terms of error handling, my io.c had conditions that checked if it couldn't read, open, or write data and would return null if the function had a return type. In bmp.c, I had to verify each read function call's output if it was false, which meant that it didn't read or it reached the end of the file before getting all necessary data. If so, then it would return NULL and in the case of specific variables, I had to check if they didn't equal the right values to return NULL as well. In colorb.c, I first checked if the h command option was the only argument and print the help message, then I also checked if they didn't put any arguments for -i or -o and return the help message. I also made Null checks for each of the functions that would return NULL and would free any memory allocated.

Figure 1: Screenshot of the color chooser colorb bmp that had the vertical line magnified

Numeric results

Figure 2: Screenshot of the original color chooser bmp file

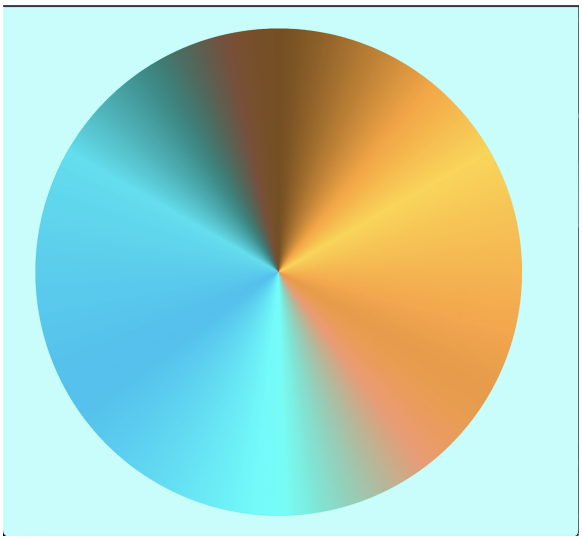

Figure 3: Screenshot of the deuteranopia version of color chooser bmp file. the black outline is from another window.