**实验报告**

专业：计算机科学与技术

姓名：张三

学号：111111

日期：2022/10/09

课程名称： 图像信息处理 指导老师： 宋明黎 成绩：

实验名称： bmp文件读写及rgb和yuv色彩空间转化

**一、实验目的和要求**

实验目的：1.利用C语言实现对bmp文件格式的读取（以24色bmp为例），从文件头获取相关的各种信息。2.将图片的RGB颜色空间转换成YUV颜色空间。3.把Y作为灰度绘制原图片的灰度图。4.稍微改变Y的值，再将YUV颜色空间转换回RGB颜色空间，重新绘制一张改变了亮度的图片。

实验要求：完全用C语言实现，不能调用图形库。

**二、实验内容和原理**

*实验原理：*

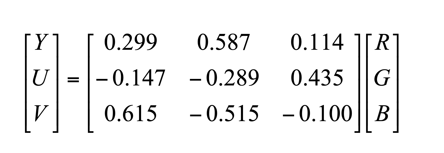
Bmp文件格式由四部分构成:BITMAPFILEHEADER,共计１４字节，带有一些关于文件的基本信息，BITMAPINFOHEADER ，共计４０字节，带有一些关于图片具体格式的信息，Palette，调色板，如果使用了调色板的话，在这里会用RGBQUAD格式存储调色板的信息。

最后就是图片的具体数据。

调色板是在使用的颜色数确定且少于时，用RGBQUAD结构来储存需要使用的颜色，比如当图里只使用了四种颜色的时候，就可以把四种颜色的RGB 值放进RGBQUAD结构中放进调色盘，并在文件头设置好对应的参数，然后每个像素就只需要２Bits来表示，００是第一个结构表示的颜色，０１是第二个结构表示的颜色，１０，是第三个结构表示的颜色，以此类推。本代码在灰度图表示中使用了调色板。

２４色RGB颜色空间用三个字节表示一个像素点，每个字节为０－２５５的无符号整数，分别表示Ｂ、Ｇ、Ｒ三种颜色的强度。

ＹＵＶ颜色空间同样是用ＹＵＶ三个分量表示一个像素点，根据不同的转换公式，ＹＵＶ的取值范围也不尽相同，本代码采用下图的转换方式（来自课堂PPT）：



可以看出Ｙ的取值范围是[0,255],Ｕ的取值范围是[-111,111],V的取值范围是[-156,156]，其中Ｖ的范围已经超过了８位，所以本代码中的转化使用float 类型来存储ＹＵＶ分量。

**三、实验步骤与分析**

//this program can only read a real color bmp file

//the test.bmp which is the source image must exist

//the program will create gray.bmp and new.bmp

//the gray.bmp will contain the gray image

//the new.bmp will contain the image whose Y is altered

#include <stdio.h>

#include <windows.h>

int main(){

    //the next three variables are for the source image

    FILE \* oldbmp;

    BITMAPFILEHEADER bf;

    BITMAPINFOHEADER bi;

    //the source image should be stored in D:

    if( !(oldbmp = fopen("D:\\test.bmp","rb"))){

        printf("Not find file in D:\\test.bmp");

        return 0;

    }

    fread(&bf,sizeof(BITMAPFILEHEADER),1,oldbmp);//read the header of bmp

    fread(&bi,sizeof(BITMAPINFOHEADER),1,oldbmp);

    //prepare to create the gray image

    BITMAPFILEHEADER graybf = bf;

    BITMAPINFOHEADER graybi = bi;

    //Because we use the Palette ,we need to change the value of Offset、SizeImage、BitCount、Size

    graybf.bfOffBits = sizeof(BITMAPFILEHEADER) + sizeof(BITMAPINFOHEADER) + 256 \* sizeof(RGBQUAD);

    graybf.bfSize = (bf.bfSize -bf.bfOffBits ) \* 8 / bi.biBitCount + graybf.bfOffBits;

    graybi.biBitCount = 8;

    graybi.biSizeImage = bi.biWidth \* bi.biHeight;

    //Create the Palette for gray image

    RGBQUAD rgbquad[256];

    for(int i=0; i<256; i++){

        rgbquad[i].rgbBlue = i;

        rgbquad[i].rgbGreen = i;

        rgbquad[i].rgbRed = i;

        rgbquad[i].rgbReserved = 0 ;

    }

    //Create the file and write the metadata

    FILE \* grayf = fopen("D:\\gray.bmp","wb");

    fwrite(&graybf,sizeof(BITMAPFILEHEADER),1,grayf);

    fwrite(&graybi,sizeof(BITMAPINFOHEADER),1,grayf);

    fwrite(rgbquad,sizeof(rgbquad),1,grayf);

    //prepare to read the rgb from source image and calculate the YUV

    unsigned char red,green,blue,y;

    float yuv[bi.biSizeImage];

    for(int i =0; i< graybi.biSizeImage; i++){

        //remember that the color is stored in BGR

        fread(&blue,1,1,oldbmp);

        fread(&green,1,1,oldbmp);

        fread(&red,1,1,oldbmp);

        //y will store the luminance and be writen into gray image

        unsigned char y = 0.299\*red + 0.587\*green + 0.114\*blue;

        yuv[3\*i] = 0.299\*red + 0.587\*green + 0.114\*blue ;

        yuv[3\*i+1] = -0.147\*red + -0.289\*green + 0.435 \*blue;

        yuv[3\*i+2] = 0.615\*red + -0.515\*green + -0.1\*blue;

        fwrite(&y,1,1,grayf);

    }

    //up to now , we have create the gray image and the YUV data array

    //prepare to create the image with altered Y

    BITMAPFILEHEADER newbf = bf;

    BITMAPINFOHEADER newbi = bi;

    unsigned char new\_color[newbi.biSizeImage];

    for( int i=0; i<bi.biHeight\*bi.biWidth; i++){

        //add 10 to every Y if Y is bigger than 255 ,set it 255

        if((yuv[3\*i] += 25 ) > 255){

            yuv[3\*i] = 255;

        }

        //use the reverse matrix to change YUV to RGB

        new\_color[3\*i+2] = 1\*yuv[3\*i] + 0\*yuv[3\*i+1] + 1.140\*yuv[3\*i+2];

        new\_color[3\*i+1] = 1\*yuv[3\*i] + -0.395\*yuv[3\*i+1] + -0.58\*yuv[3\*i+2];

        new\_color[3\*i] = 1.002\*yuv[3\*i] + 2.036\*yuv[3\*i+1] + 0\*yuv[3\*i+2];

    }

    //create the new file and write the data into it

    FILE \* newbmp = fopen("D:\\new.bmp","wb");

    fwrite(&newbf,sizeof(BITMAPFILEHEADER),1,newbmp);

    fwrite(&newbi,sizeof(BITMAPINFOHEADER),1,newbmp);

    fwrite(&new\_color,sizeof(new\_color),1,newbmp);

}

**四、实验环境及运行方法**

源程序直接使用gcc正常编译即可，运行时需要将原bmp文件放在目录D:\test.bmp，然后运行编译产生的可执行文件，即可产生两张结果图gray.bmp 与new.bmp

**五、实验结果展示**

Test.bmp:  


Gray.bmp:



New.bmp:



YUV颜色空间的数据仅以float数据格式在过程中存储在数组内，不另作输出。

**六、心得体会**

本次实验复习了Ｃ中文件读写相关的操作。同时亲身实践了解了bmp 图片的存储格式，以及各种不同颜色空间之间的互相转换，如何利用ＲＧＢ颜色空间到ＹＵＶ颜色空间的转化来生成灰度图。

本次实验仍然有许多不足之处，比如只能对真彩色的ｂｍｐ图进行处理，再比如在修改了Ｙ后重新生成的图片中有一些难以去除的绿色点和蓝色点，希望能够通过后期的学期不断进步。