

**程 序 设 计 专 题**

**大 程 序 报 告**



大程名称： BMP图片编辑器

小组成员：

1. **姓名 ： 学号：电话：**
2. **姓名 ： 学号：电话：**

**指导老师： 肖少拥**

**2021~2022春夏学期 2022 年 6 月 3 日**

**报告撰写注意事项**

1. 图文并茂。文字通顺，语言流畅，无错别字。
2. 书写格式规范，排版良好，内容完整。
3. 存在拼凑、剽窃等现象一律认定为抄袭；0分
4. 蓝色文字为说明，在最后提交的终稿版本，请删除这些文字。

**目 录**

[1 大程序简介 4](#_Toc65352773)

[1.1 选题背景及意义 4](#_Toc65352774)

[1.2 目标要求 4](#_Toc65352775)

[2 需求分析 5](#_Toc65352777)

[2.1 功能需求 5](#_Toc65352779)

[2.2 数据需求 6](#_Toc65352780)

[2.3 性能需求 6](#_Toc65352781)

[3 程序开发设计 7](#_Toc65352782)

[3.1 总体架构设计 7](#_Toc65352783)

[3.2 功能模块设计 7](#_Toc65352784)

[3.3 数据结构设计 8](#_Toc65352785)

[3.4 源代码文件组织设计 9](#_Toc65352786)

[3.5 函数设计描述 12](#_Toc65352787)

[4 部署运行和使用说明 20](#_Toc65352788)

[4.1 编译安装 20](#_Toc65352789)

[4.2 运行测试 20](#_Toc65352790)

[4.3 使用操作 22](#_Toc65352791)

[5 团队合作 26](#_Toc65352792)

[5.1 任务分工 26](#_Toc65352793)

[5.2 开发计划 27](#_Toc65352794)

[5.3 编码规范 27](#_Toc65352795)

[5.4 合作总结 27](#_Toc65352796)

[5.5 收获感言 30](#_Toc65352797)

[6 参考文献资料 31](#_Toc65352798)

BMP图片编辑器设计项目

# 大程序简介

## 选题背景及意义

通过一学期对于C语言的学习，我组在深入了解libgraphics图形库与Windows API后，尝试以小组合作的方式，设计并开发一个BMP图片编辑器，实现对于BMP图片进行简易、快捷、高效的编辑以及进行一些基本的图像处理。同时，此次程序设计也极大地加深了我组对于相关知识的掌握与理解，切身经历了程序开发的整个流程，丰富了分工合作的经验。

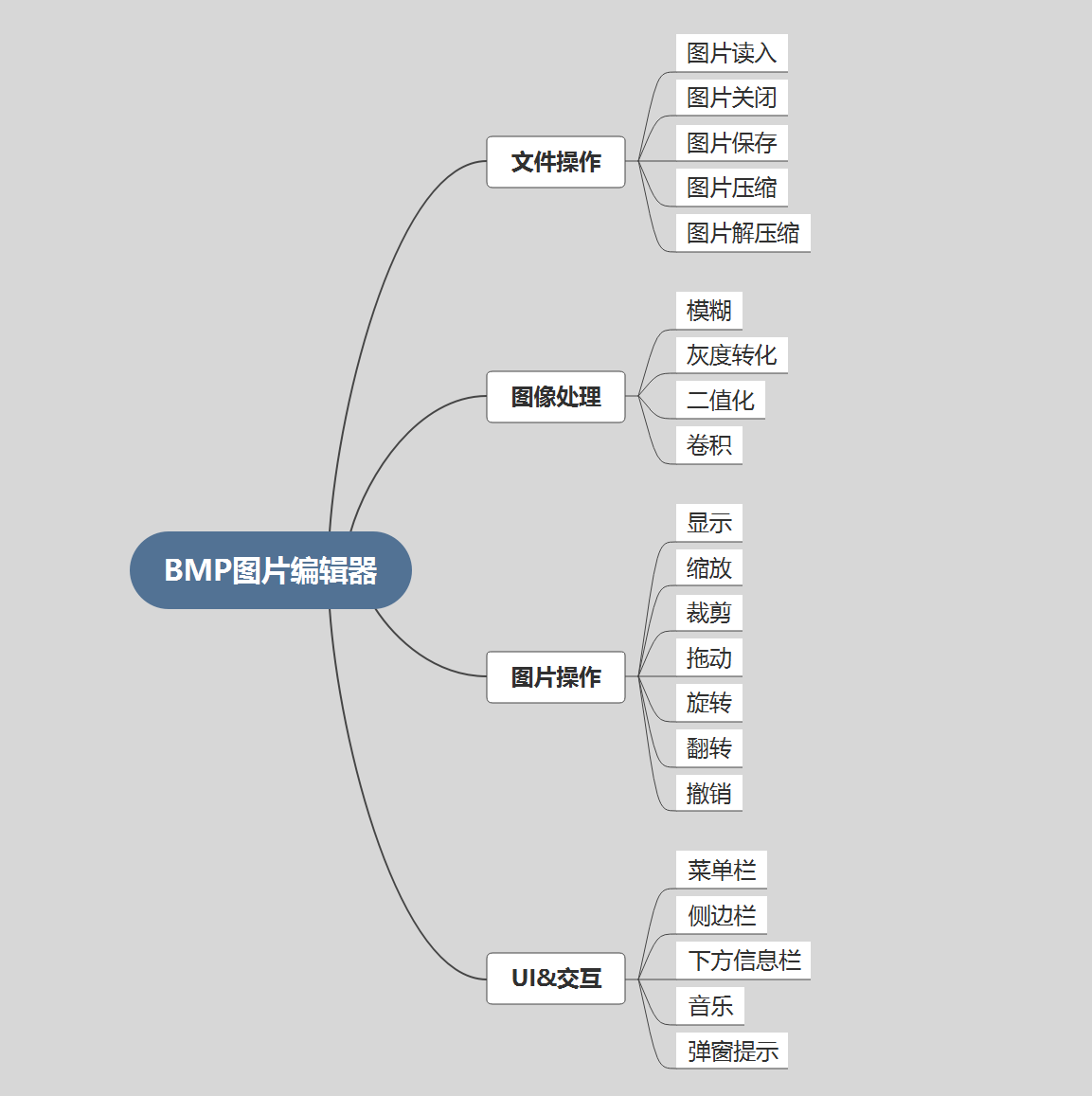
## 目标要求

本程序为基于libgraphics图形库与Windows API实现的BMP图片编辑器，主要功能有实现对于24位BMP图片的读入、保存、显示、拖动、缩放、裁剪、旋转、压缩、解压缩等功能，所有操作皆支持撤销，且对应侧边栏、菜单栏上的按钮选项，并可使用快捷键进行直接操作。图形窗口底端有状态信息栏，可实时显示图片位置、大小、操作模式等基本信息。文件的读入与保存皆支持自定义文件名，并可以弹窗的形式进行“保存成功”“文件未保存”等提醒，形成良好的反馈，优化用户的交互感与体验感。

此外，本程序还会提供一些基础的图像处理功能，能够实现对于图像的模糊处理（高斯模糊）、灰度转化、二值化、自定义卷积等操作，提供更多的操作空间，拓展了图片编辑的可能性。

# 需求分析

## 功能需求



图片读入：能够根据BMP图片的完整文件名，读取并打开图片，进入编辑窗口。

图片关闭：能够关闭正在编辑的图片，并支持打开新图片。

图片保存：能够将编辑后的图片以自定义文件名保存在当前目录下。

图片压缩：参考RLE（Run Length Encoding）游程长度压缩算法，对当前图片进行压缩。

图片解压缩：识别打开的文件格式，并自动对压缩文件进行解压缩后读入。

显示：在窗口中显示图片的效果视图。

缩放：提供对于图片进行锁定长宽比例以及不锁定长宽比例的缩放功能。

裁剪：提供对于图片进行任意裁剪的工具。

拖动；能够在窗口指定范围内任意拖动图片，改变位置。

旋转：能够对图片进行顺时针与逆时针的旋转。

翻转：能够对图片进行左右与上下的镜像翻转。

撤销：能够撤销最近一次的操作，返回图片的上一个状态。

模糊：使用高斯模糊卷积核对图片进行模糊化处理。

灰度转化：将彩色图片转化为灰度图像。

二值化：根据设定的阈值，将图片转化为黑白两色的图片。

卷积：能够自定义3\*3卷积核，对图像进行卷积处理。

菜单栏：在窗口上方添加下拉式菜单。

侧边栏：在窗口左侧添加包含快捷按键的工具栏。

下方信息栏：在窗口下方显示图片的相关信息以及操作状态。

音乐：能够在程序中打开和关闭背景音乐。

弹窗提示：在用户使用相关操作后通过弹窗的形式进行必要的提示。

## 数据需求

为实现BMP格式图片的读入与保存，程序必须能够获取BMP格式二进制文件的文件头、信息头以及图像数据。因此，本程序将引用wingdi.h中关于BITMAPFILEHEADER(文件头)以及BITMAPINFOHEADER(信息头)的结构体定义，实现对于文件头与信息头的读入与保存。同时，由于本程序所要处理的图片为24位位图，每个像素的颜色信息将分为RGB三色道分别储存在三个字节中，因此程序将通过动态申请(3\*图片长\*图片宽)个unsigned char数据类型的空间的方式来储存图片数据。

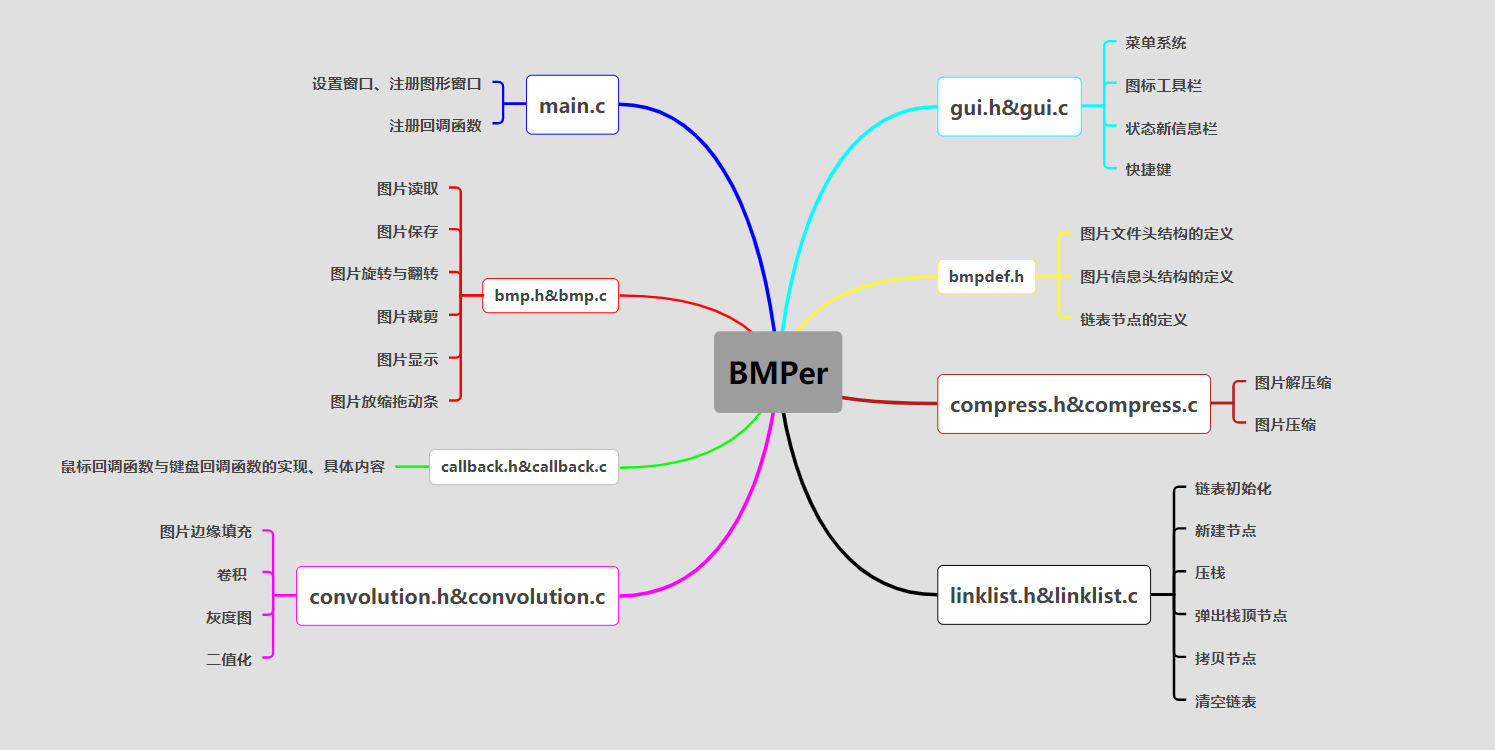
此外，由于程序需要支持图片拖动以及撤销操作的功能，储存每一次操作后的图像数据以及起始位置等信息是必要的，并且为了防止对图片多次操作后(如先缩小后放大)出现失真的问题，原图数据也必须作为基准保存。因此，本程序采用了两次栈的数据结构，分别用于保存图片操作后的结果（操作栈）以及图片基准改变后的结果(基准栈，在裁剪、二值化等操作后改变)，两个栈同步更新，每个节点将储存完整的图像数据以及起始位点、长度、宽度等信息。当用户进行操作后，将操作结果构造为节点逐一压入栈中；当使用撤销功能后，则将栈顶的节点弹出。

## 性能需求

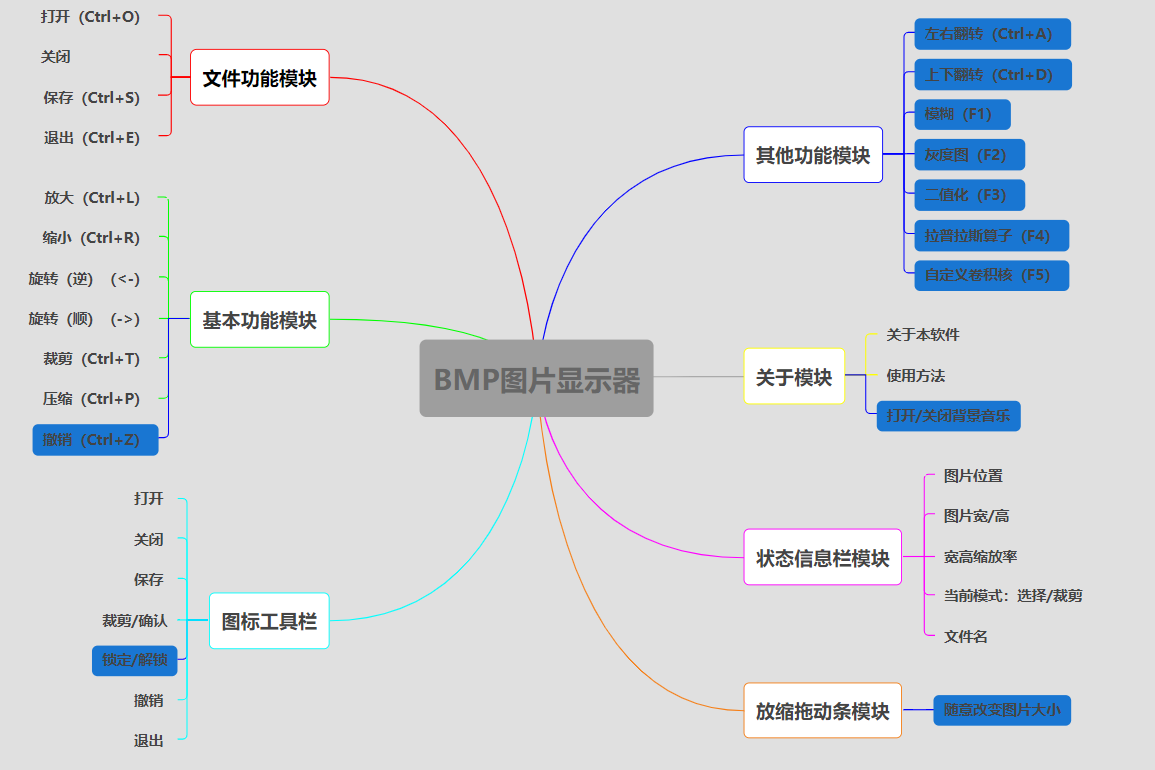
为实现对于图片进行编辑以及实时显示操作效果的功能，本程序采用了SetPixel函数，将位图数据通过逐一打点的形式显示在窗口中。由于图片操作的运算量较大，很难做到对于连续性操作进行实时以及瞬间的反馈，极大影响了用户的操作体验。因此，为优化窗口刷新频率不足的问题，本程序在进行一些连续性的操作(如拖动图片的位置、缩放图片的长宽)时，会以黑边方框的形式提示图片操作的效果预览，降低程序负荷，优化交互与反馈的效果。

# 程序开发设计

## 总体架构设计



## 功能模块设计



（标蓝色的为较有特色的功能）

## 数据结构设计

1. 整型变量

key：键盘按键回调；

button：鼠标按键回调；

event：鼠标、键盘等消息事件；

dx1,dx2,dy1,dy2：裁剪模式下四条边所要裁去的像素值；

1. 双精度浮点型变量

Xratio，Yratio：图片长、宽的缩放比例；

1. 布尔型变量

OPEN：记录是否已有文件打开；

SAVE：文件是否已经保存；

IS\_TAILOR：是否在裁剪模式；

IS\_PLAY：音乐是否打开；

IS\_LOCK：是否已经锁定图片长宽比例；

1. 字符串数组

FileName[50]：缓存读入文件名或保存文件名；

name[50]：保存当前打开文件的文件名(不含后缀名)；

1. 其他数据类型变量

FILE \*fp：打开文件的文件指针；

BMP\_node \*top：储存操作过程的栈顶指针；

BMP\_node \*base：储存图片基准数据的栈顶指针；

HWND window：窗口句柄;

1. 结构与链表节点

/\*BMP文件头结构体\*/

typedef struct tagBITMAPFILEHEADER

{

unsigned short bfType; //保存图片类型。 'BM'

unsigned int bfSize; //位图文件的大小（3-6字节，低位在前）

unsigned short bfReserved1;//位图文件保留字，必须为0(7-8字节）

unsigned short bfReserved2;//位图文件保留字，必须为0(9-10字节）

unsigned int bfOffBits; //RGB数据偏移地址（11-14字节，低位在前）

}BITMAPFILEHEADER;

/\*BMP信息头结构体\*/

typedef struct tagBITMAPINFOHEADER

{

unsigned int biSize; //本结构所占用字节数（15-18字节）

unsigned int biWidth; //位图的宽度，以像素为单位（19-22字节）

unsigned int biHeight; //位图的高度，以像素为单位（23-26字节）

unsigned short biPlanes; //目标设备的级别，必须为1(27-28字节）

unsigned short biBitCount; //每个像素所需的位数

unsigned int biCompression;//位图压缩类型（31-34字节）

unsigned int biSizeImage; //位图的大小(其中包含了为了补齐行数是4的倍数而添加的空字节)（35-38字节）

unsigned int biXPelsPerMeter;//位图水平分辨率，每米像素数（39-42字节）

unsigned int biYPelsPerMeter;//位图垂直分辨率，每米像素数（43-46字节)

unsigned int biClrUsed; //位图使用的颜色表中的颜色数（47-50字节）

unsigned int biClrImportant; //位图显示过程中重要的颜色数（51-54字节）

}BITMAPINFOHEADER;

/\*图片信息结构体，链表节点\*/

typedef struct node

{

BITMAPFILEHEADER head;//文件头

BITMAPINFOHEADER info;//信息头

unsigned char \*bmp\_data; //图片数据

int InitX; //图片起始X坐标

int InitY; //图片起始Y坐标(左下角)

struct node \*next; //链表指针

bool changebase; //记录此次操作是否改变了图像基准栈

}BMP\_node;

1. 宏定义

#define UI\_X 2.8 //缩放条UI的起始X坐标

#define UI\_Y 2.0 //缩放条UI的起始Y坐标

#define UILenth 2 //缩放条的长度

#define UIInterval 0.5 //长、宽缩放条UI的间距

#define UI\_Button\_X 0.1 //拖动按钮的宽度

#define UI\_Button\_Y 0.4 //拖动按钮的高度

#define ScreenY 10 //窗口的高度

#define WindowH 10 //窗口的高度

#define WindowW 16 //窗口的宽度

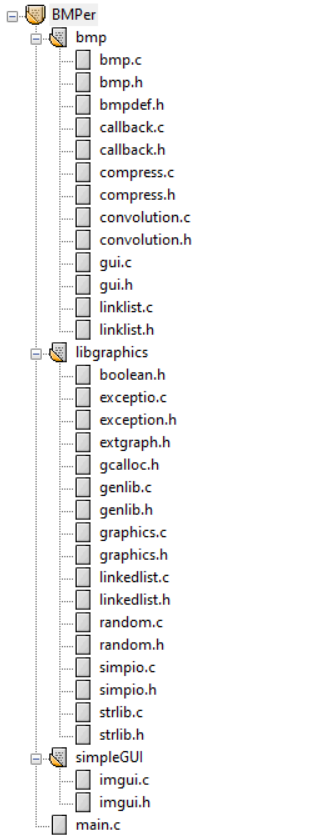
#define INTERVAL 1 //图标工具栏每个按键的间隔

#define INIT\_X\_POS 6 //图片的初始化X坐标

#define INIT\_Y\_POS 2 //图片的初始化Y坐标

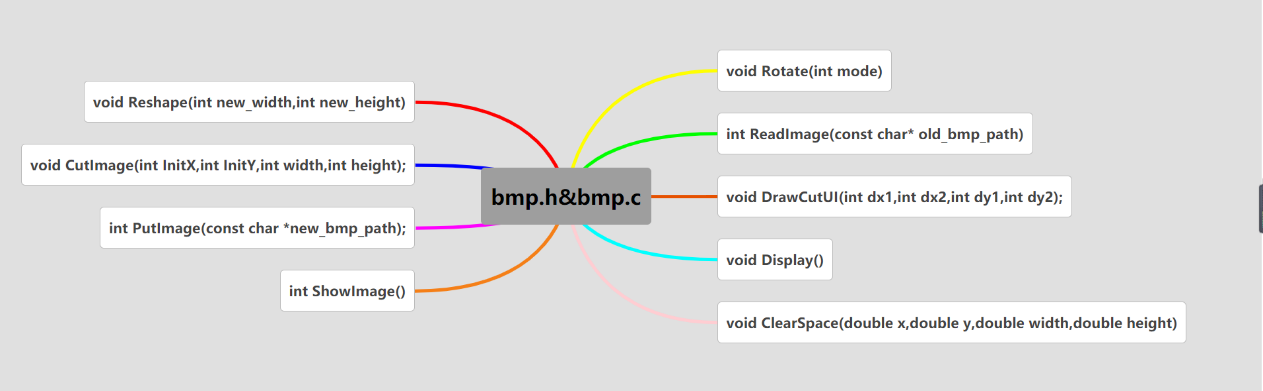
## 源代码文件组织设计

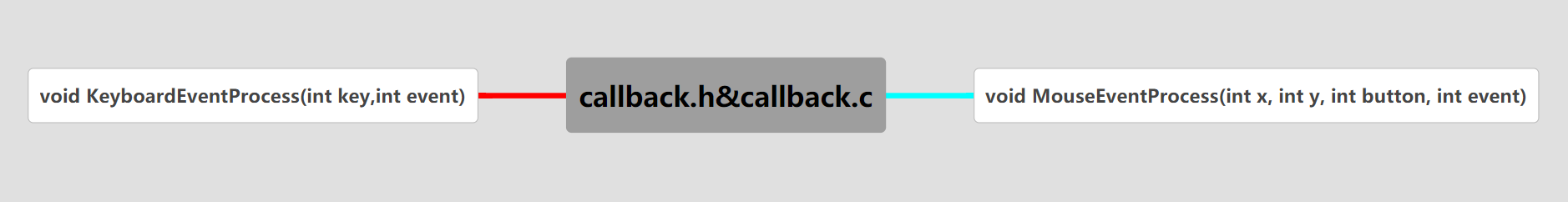
<文件目录结构>

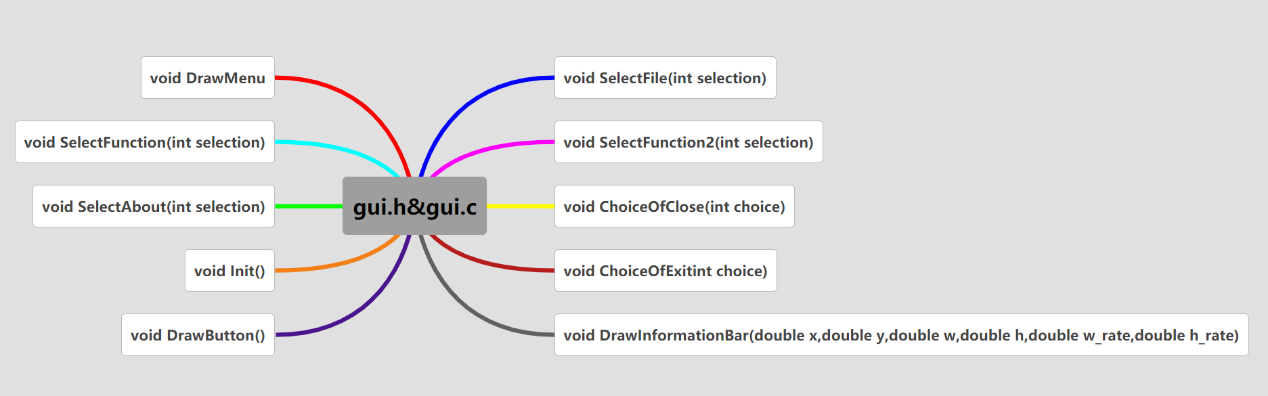


1. 文件函数结构

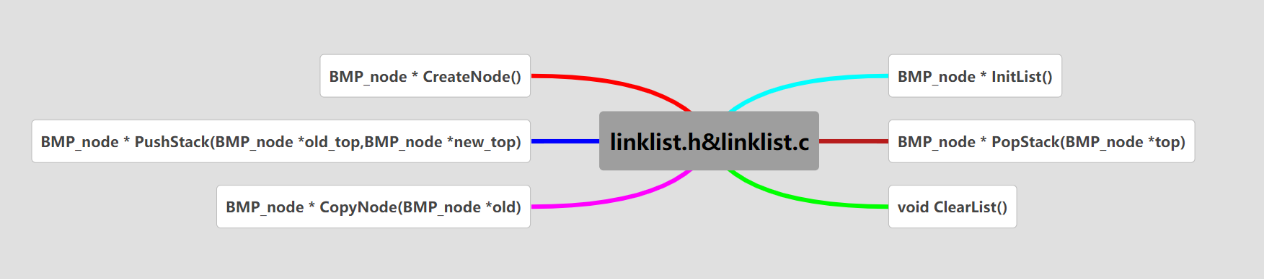
main.c——void Main()













2）多文件构成机制

main.c为主文件，包含所有其他源文件以及libgraphics库，C语言标准库。

bmpdef.h中定义了24位BMP图片的文件头、信息头，也定义了储存图片数据信息的结构体，使用头文件保护以保证只被编译一次；

bmp.h中包含有bmp.c中定义的函数的声明、宏定义和全局变量声明，使用头文件保护以保证只被编译一次；

linklist.h中包含了linklist.c中定义的创建节点、初始化链表等函数声明和宏定义，使用头文件保护以保证只被编译一次；

gui.h中包含了gui.c中定义的函数的声明、宏定义和全局变量声明，使用头文件保护以保证只被编译一次；

convolution.h中包含了convolution.c中定义的函数的声明，使用头文件保护以保证只被编译一次；

compress.h中包含了compress.c中定义的压缩和解压缩函数的声明，使用头文件保护以保证只被编译一次；

callback.h中包含了callback.c中定义的函数的声明，使用头文件保护以保证只被编译一次；

## 函数设计描述

**3.5.1 gui.c**

1. 函数原型：void DrawMenu()

功能描述：在图形界面画出菜单

参数描述：无

返回值描述：无

重要局部变量用途描述：

char \*menuListFile[] 菜单中的文件一栏

char \*menuListFunction[] 菜单中基本功能一栏

char \*menuListFunction2[] 菜单中其他功能一栏

char \*menuListAbout[] 菜单中关于一栏

函数算法描述： 画出菜单，并通过调用其他函数实现菜单功能

1. 函数原型：void SelectFile(int selection)

功能描述：用户对菜单中文件一栏进行操作时做出处理

参数描述：

selection 用户的选择

返回值描述：无

重要局部变量用途描述：无

函数算法描述： 主要通过if语句（也可以用switch语句），对用户的选择进行判断，并进行相关的操作

1. 函数原型：void ChoiceOfClose(int choice)

功能描述：若用户在未保存并且点击关闭时会弹出一个弹窗进行询问是否保存，此函数用来处理用户对弹窗操作后的结果

参数描述：

choice 用户的选择

返回值描述：无

重要局部变量用途描述：无

函数算法描述： 主要通过if语句，对用户的选择进行判断，并进行相关的操作

1. 函数原型：void ChoiceOfExit(int choice)

功能描述：若用户在未保存并且点击退出时会弹出一个弹窗进行询问是否保存，此函数用来处理用户对弹窗操作后的结果

参数描述：

choice 用户的选择

返回值描述：无

重要局部变量用途描述：无

函数算法描述： 主要通过if语句，对用户的选择进行判断，并进行相关的操作

1. 函数原型：void Init()

功能描述：在打开文件时初始化一些变量，防止因为文件的多次打开关闭出现判断错误的情况

参数描述：无

返回值描述：无

重要局部变量用途描述：无

函数算法描述：使SAVE=TRUE,OPEN=FALSE

1. 函数原型：void SelectFunction(int selection);

功能描述：用户对菜单中基本功能一栏进行操作时做出处理

参数描述：

selection 用户的选择

返回值描述：无

重要局部变量用途描述：无

函数算法描述： 主要通过switch语句，对用户的选择进行判断，并通过其他函数进行相关的操作

1. 函数原型：void SelectFunction2(int selection);

功能描述：用户对菜单中其他功能一栏进行操作时做出处理

参数描述：

selection 用户的选择

返回值描述：无

重要局部变量用途描述：无

函数算法描述： 主要通过switch语句，对用户的选择进行判断，并通过调用函数进行相关的操作

1. 函数原型：void SelectAbout(int selection);

功能描述：用户对菜单中基本功能一栏进行操作时做出处理

参数描述：

selection 用户的选择

返回值描述：无

重要局部变量用途描述：无

函数算法描述： 主要通过if语句，对用户的选择进行判断，并通过调用函数进行相关的操作

1. 函数原型：DrawInformationBar(double x,double y,double w,double h,double w\_rate,double h\_rate);

功能描述：画出图形用户界面底部的状态信息栏

参数描述：

x 起始位置横坐标

y 起始位置纵坐标

w 宽度

h 高度

w\_rate 图片宽度缩放率

h\_rate 图片高度缩放率

返回值描述：无

重要局部变量用途描述：

char location[20] 记录图片位置

char size[25] 记录图片的宽和高

char hw\_rate[30] 记录图片的缩放率

函数算法描述： 获取各种状态信息，并显示

1. 函数原型：void DrawButton()

功能描述：用户对菜单中基本功能一栏进行操作时做出处理

参数描述：

selection 用户的选择

返回值描述：无

重要局部变量用途描述：无

函数算法描述：无

* + 1. **compress.c**

1. 函数原型：int Compress(unsigned char \*source,unsigned int n)

功能描述：对传入的图像数据数组进行压缩，并返回压缩后的数组的长度

参数描述：

source 传入的单通道图像数据数组

n 传入数组的长度

返回值描述：压缩后数组的长度

重要局部变量用途描述：

unsigned char copy 复制一份传入的数组

函数算法描述：对连续的相同的数据进行压缩，并用0做压缩后的标记（如5 5 5 5 5 5压缩后为0 6 5）,而压缩后不连续相同的0表示为00（如0压缩后为0 0，而0 0 0 0 0压缩后为0 5 0 ），其他的不连续相同的数据不变

1. 函数原型：void SaveCompress(const unsigned char \*source,const char \*bmp\_path)

功能描述：将传入的图像数据分成r,g,b三个数组，并调用压缩函数进行压缩，最后将压缩结果保存在.a文件中

参数描述：

source 传入的图像数据数组

bmp\_path保存为压缩文件的名字

返回值描述：无

重要局部变量用途描述：

unsigned char \*r 将图像数据中的R通道数组组成一个数组

unsigned char \*g 将图像数据中的G通道数组组成一个数组

unsigned char \*b 将图像数据中的B通道数组组成一个数组

函数算法描述：无

1. 函数原型：void Unzip(const unsigned char\*compress,int n,unsigned char \*unzip)

功能描述：对图像信息进行解压缩

参数描述：

compress 压缩后的单通道图像数据

n 传入的图像数据数组的长度

unzip 储存解压后的图像数据

返回值描述：无

重要局部变量用途描述：无

函数算法描述：按照图像的压缩原理进行解压，读取到0的时候进行判断，如果下一位是0，则解压为0，否则则解压为n位相同数据…

1. 函数原型：void ReadCompressImage(const char\* path)

功能描述：调用解压缩函数进行解压，将解压后的r、g、b图像数据合成为一个数组，最终将图像输出在界面中。

参数描述：

bmp\_path需要解压的压缩文件的名字

返回值描述：无

重要局部变量用途描述：

unsigned char \*r 储存R通道解压后的数组

unsigned char \*g 储存G通道解压后的数组

unsigned char \*b 储存B通道解压后的数组

函数算法描述：读取压缩文件，调用压缩函数进行解压，再将图片输出在图形界面中

* + 1. **bmp.c**

1. 函数原型：void Reshape(int new\_width,int new\_height)

功能描述：对图片进行缩放操作，并将操作结果保存在操作栈中

参数描述：

new\_width 缩放后图片的宽度

new\_height 缩放后图片的高度

返回值描述：无

重要局部变量用途描述：

int srcX 缩放后图片像素的X坐标所对应的原图像素X坐标

int srcY 缩放后图片像素的Y坐标所对应的原图像素Y坐标

函数算法描述：采用最邻近插值法，根据新图片与基准图片的缩放比例，构建一个映射关系，使得新图片中的像素分别对应到一个基准图片中的像素，以最近的像素来填补空缺或表示特征。映射规则以X坐标举例，新图片横坐标为X的像素对应原图中横坐标为(X/新图片宽度\*基准图片宽度)的像素。

1. 函数原型：void Rotate(int mode)

功能描述：对图片进行旋转或翻转操作，并将操作结果保存在操作栈中

参数描述：

mode 旋转模式，值为1时顺时针旋转，2时逆时针旋转，3时左右翻转，4时上下翻转

返回值描述：无

重要局部变量用途描述：

无

函数算法描述：将图片看作一个二阶矩阵进行操作，对矩阵进行顺时针、逆时针以及镜像翻转变换，从而实现图片的旋转与镜像翻转操作。

1. 函数原型：void CutImage(int InitX,int InitY,int width,int height)

功能描述：对图片进行裁剪操作

参数描述：

InitX 裁剪后图片的起始位点所在原图中的X坐标

InitX 裁剪后图片的起始位点所在原图中的Y坐标

width 裁剪后图片的宽度

height 裁剪后图片的高度

返回值描述：无

重要局部变量用途描述：

int base\_width 依据当前图片宽度相对于基准的比例，逆向计算在基准中所要裁剪的宽度

int base\_height 依据当前图片高度相对于基准的比例，逆向计算在基准中所要裁剪的高度

函数算法描述：先按比例逆向计算出基准中所要裁剪的长度与宽度，随后创建一个新的图片基准数据节点，逐行将像素信息从原基准图片拷贝到新的基准中，从而实现裁剪的效果。对基准操作完成后，再将裁剪后的图片按比例恢复到当前大小，保存在操作栈中，防止图像出现失真。

1. 函数原型：int ReadImage(const char\* old\_bmp\_path)

功能描述：从二进制文件中读入BMP图片数据

参数描述：

old\_bmp\_path 所要读入文件的完整文件名

返回值描述：无意义

重要局部变量用途描述：

int bias 计算源图宽度相对于四字节的偏差值

函数算法描述：新建基准节点，依次将源图的文件头与信息头读入结构体中，从中获取图片的长宽，随后将图像数据读入动态申请的内存中，再裁剪掉其中相对于四字节偏差值的占位字节(bmp格式图片在行大小不为四字节的倍数时，会以值为0的字节进行补齐)，保存在节点当中，调用图片显示函数进行显示。

1. 函数原型：int PutImage(const char \*new\_bmp\_path)

功能描述：将图片编辑的结果保存在自定义文件名的文件中

参数描述：

new\_bmp\_path 所要保存文件的完整文件名

返回值描述：无意义

重要局部变量用途描述：无

函数算法描述：用wb+的方式打开新的二进制文件，依次将构造完成的文件头、信息头和图像信息写入文件中。

1. 函数原型：int ShowImage()

功能描述：将读入的图片显示在窗口中

参数描述：无

返回值描述：无意义

重要局部变量用途描述：无

函数算法描述：用wb+的方式打开新的二进制文件，依次将构造完成的文件头、信息头和图像信息写入文件中。

1. 函数原型：void Display()

功能描述：显示窗口的所有UI

参数描述：无

返回值描述：无

重要局部变量用途描述：无

函数算法描述：依次绘制窗口的菜单栏、图表工具栏、下方信息栏等UI。

1. 函数原型：void ClearSpace(double x,double y,double width,double height)

功能描述：将指定区域清空

参数描述：

x 所要清空区域的起始X坐标

y 所要清空区域的起始Y坐标

width 所要清空区域的宽度

height 所要清空区域的高度

返回值描述：无

重要局部变量用途描述：无

函数算法描述：用DrawLine()函数画出长方形封闭区域，使用StartFilledRegion()与EndFilledRegion()函数将区域填充为密度0(全白)。

* + 1. **callback.c**

1. 函数原型：void KeyboardEventProcess(int key,int event)

功能描述：键盘回调函数，在按下按键时发生回调

参数描述：

key 所按下按键的虚拟码值

event 对按键操作的事件

返回值描述：无

重要局部变量用途描述：无

函数算法描述：当回调此函数时，判断所操作的按键虚拟码值以及按键状态，分别进入不同的分支，调用不同的函数实现操作效果。

1. 函数原型：void MouseEventProcess(int x, int y, int button, int event)

功能描述：鼠标回调函数，在鼠标移动或按下、抬起时发生回调

参数描述：

x 函数回调时鼠标的X坐标

y 函数回调时鼠标的Y坐标

button 所操作的按键

event 对按键操作的事件 （按下、抬起）

返回值描述：无

重要局部变量用途描述：

static int originX，originY 保存上一次回调时鼠标的位置

static int InitX,InitY 记录起始点的移动

函数算法描述：当回调此函数时，判断所操作的按键以及按键状态。当按下鼠标左键时，判断鼠标的坐标，根据选中区域的不同，分别实现拖动图片、拖动宽度、拖动高度等效果，在抬起鼠标时，调用清屏函数，再重画UI以及图片，实现画面的更新。

* + 1. **convolution.c**

1. 函数原型：unsigned char\* fill()

功能描述：对图片实现边缘填充

参数描述：无

返回值描述：边缘填充后的图像数据

重要局部变量用途描述：

int con 对图片实现常数边缘填充的像素值

函数算法描述：创建一长度、宽度分别较原图多2个单位像素的图像数据数组，将边缘的一圈像素设定为定值con，为后续的卷积操作实现预处理。

1. 函数原型：void convolution(double core[3][3])

功能描述：对图片实现卷积操作

参数描述：

core[3][3] (3\*3)大小的卷积核

返回值描述：无

重要局部变量用途描述：无

函数算法描述：先对图像进行边缘填充，获得长宽比原图多2个像素的数据，随后根据传入的卷积核，依次对图片的像素计算卷积，处理计算结果中大于255以及小于0的像素值，获得长宽与原图相同的图像数据，将操作的结果保存在基准栈中，并更新在操作栈中。

1. 函数原型：void Gray()

功能描述：将图片转化为灰度图

参数描述：无

返回值描述：无

重要局部变量用途描述：

int gray 依据三色道的数据所计算出的灰度值

函数算法描述：根据灰度转换算法，逐个获取像素的三色道信息并计算得出灰度值，构造新图片的图像数据，保存在基准栈中，并更新在操作栈中。

1. 函数原型：void Binarization(int n)

功能描述：将图片二值化

参数描述：

n 图片二值化的阈值

返回值描述：无

重要局部变量用途描述：

int gray 依据三色道的数据所计算出的灰度值

函数算法描述：先将图片转换为灰度图，随后将灰度值大于阈值的像素设置为255，灰度值小于阈值的像素设置为0，构造新图片的图像数据，保存在基准栈中，并更新在操作栈中。

* + 1. **linklist.c**

1. 函数原型：BMP\_node \* CreateNode()

功能描述：新建一个图像数据节点

参数描述：无

返回值描述：新建节点的地址

重要局部变量用途描述：

无

函数算法描述：用malloc()函数动态申请内存，创建一个链表节点。

1. 函数原型：BMP\_node \* InitList()

功能描述：初始化链表

参数描述：无

返回值描述：初始化链表的首地址

重要局部变量用途描述：

无

函数算法描述：用CreateNode()函数创建节点，并初始化第一个节点的成员变量。

1. 函数原型：BMP\_node \* PushStack(BMP\_node \*old\_top,BMP\_node \*new\_top)

功能描述：将新节点压入栈中

参数描述：

old\_top 旧的栈顶指针

new\_top 新压入节点的地址

返回值描述：新的栈顶指针

重要局部变量用途描述：

无

函数算法描述：使新节点的next指针指向旧的栈顶，使其成为新的栈顶。

1. 函数原型：BMP\_node \* PopStack(BMP\_node \*top)

功能描述：将栈顶的节点弹出

参数描述：

top 栈顶指针

返回值描述：新的栈顶指针

重要局部变量用途描述：

无

函数算法描述：若栈不为空栈则执行。暂存栈顶next指针的值，先释放节点成员变量的内存，再释放节点的内存，将next指针的值作为返回值传回，成为新的栈顶指针。

1. 函数原型：void ClearList()

功能描述：清空栈

参数描述：无

返回值描述：无

重要局部变量用途描述：

无

函数算法描述：依次将栈顶的节点弹出，直至栈被清空，将top指针以及base指针的值都初始化为NULL。

1. 函数原型：BMP\_node \* CopyNode(BMP\_node \* old)

功能描述：深拷贝一个节点

参数描述：

old 所要拷贝的节点地址

返回值描述：拷贝获得的节点地址

重要局部变量用途描述：

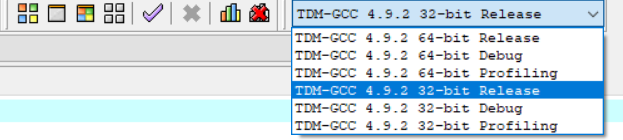
无

函数算法描述：先新建一个节点，将所有非指针成员变量皆拷贝给它，再为指针动态申请内存，对指针所指向的数据也进行拷贝，执行深拷贝操作，防止节点之间操作发生冲突。

# 部署运行和使用说明

## 编译安装

安装Dev-C++，将压缩包解压，打开解压后的文件夹，再打开devC文件夹，再打开BMPer.dev文件，并将TMD-GCC调整为32-bit Release,编译后不会有任何error或warning提示（若TMD-GCC未调整为32-bit Release则可能会出现warning提示，但也不影响程序运行），并生成BMPer.exe文件，再运行程序即可。



## 运行测试

**案例一：对特殊分辨率图片的读入**

在实现基本的BMP图片读入、显示等功能后，发现对于某些特殊的图片，程序无法进行正常的显示，而是会呈现为倾斜错位的状态，如下图所示。



由于程序的其他功能依旧运行正常，初步将问题定位在读入二进制文件的ReadImage()函数中。通过大量测试不同的图片，提取出无法正常显示的图片进行对比分析，发现这些图片之中存在共性，那就是图片宽度的像素值不为四的倍数。

查阅相关资料，得知Windows规定一个扫描行所占的字节数必须是4的倍数，24位的BMP图片一个像素占三字节，因此当一行的像素数不为四的倍数时，需要对其进行4字节对齐，不足的字节会以0来填充，这样的BMP格式图片才是合法的。

了解这一点后，我们在ReadImage ()函数中增加了bias变量，用于计算读入图片每行的填充0的字节数，并将其从图像数据中删去，最终实现对于任意分辨率的24位BMP图片的正常读入与显示。

**案例二：撤销功能与其他功能的兼容性问题**

在添加撤销功能之后，发现在某些特定情况下使用撤销功能，会导致后续操作出现异常，产生意想不到的结果，如使用缩小功能后图片却被放大，或使用放大功能后图片却被缩小等。

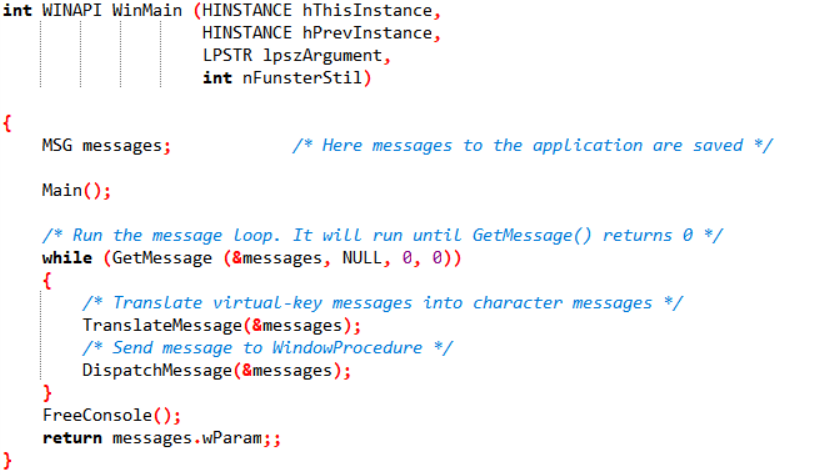
为探明问题发生的原因，我们对于程序的各项功能以及撤销进行了大量的测试与分析，任意组合操作的先后顺序，从中总结共性，得出如下结论：先对图像进行缩小操作，随后执行撤销，再进行缩小操作时，图片会一下缩小很多。

因此我们将问题定位在了撤销操作后对图片相关数据的重置与还原上。我们发现，当操作被撤销后，尽管图像数据被重置为了上一次的状态，但保存图像长宽缩放比例的全局变量Xratio与Yratio并未重置，导致其依旧保持着图片撤销前的状态，当再一次进行操作时，图片的缩放就会在此基础上发生改变，从而导致问题的爆发。

通过在撤销操作后重置Xratio与Yratio的值，此问题被成功解决。

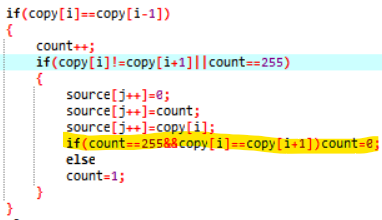
**案例三：压缩连续相同的数据超过255的图片时出错的问题**

在添加了压缩与解压缩功能之后，我们发现在解压缩图片时程序会出错。我们一度以为是解压缩算法的问题，为此我们进行了大量的调试，但每次程序都在graphics.c中的WinMain函数中的While循环的第三次出错。



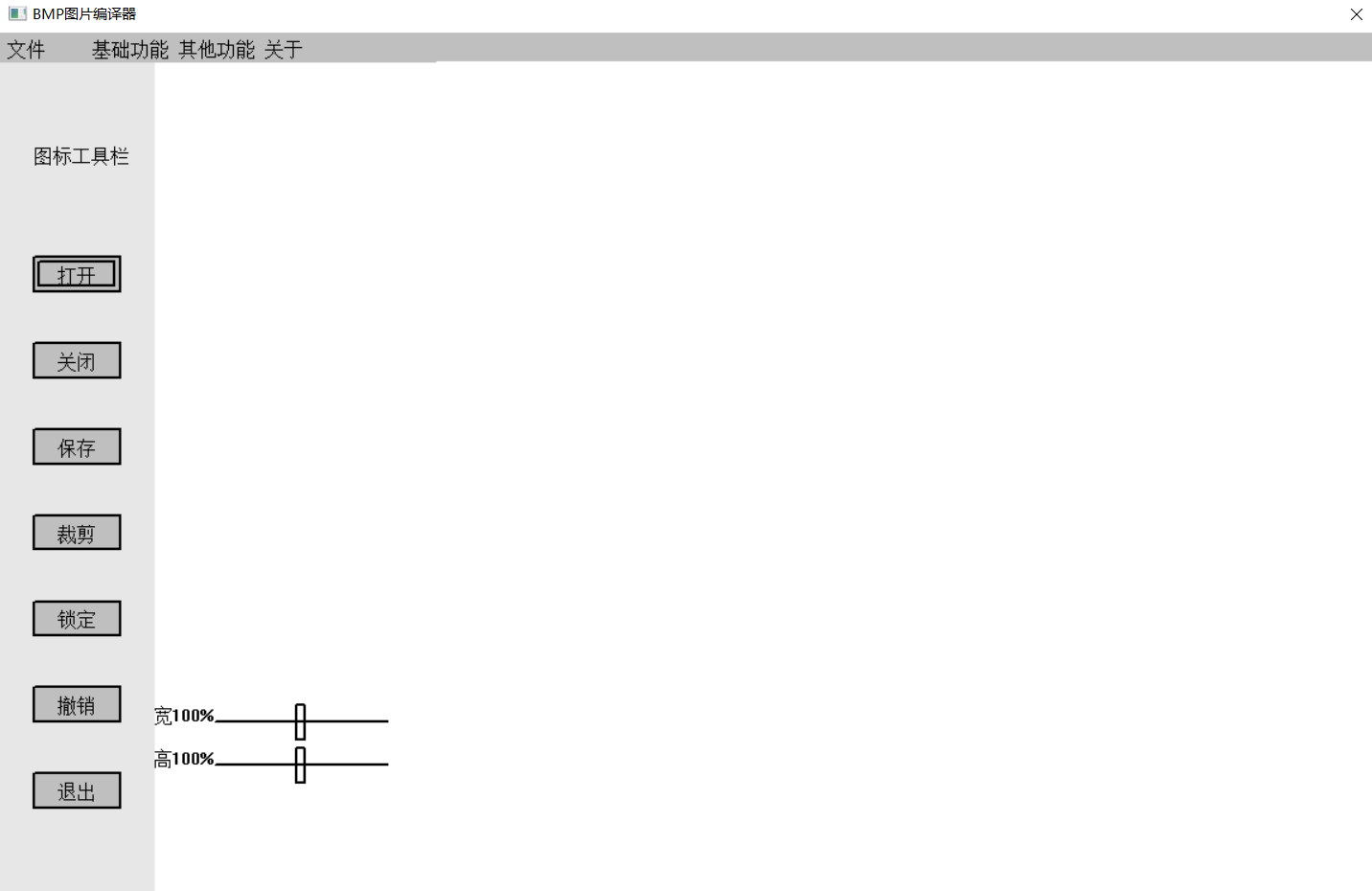
因为无法得知哪里出错，我们又尝试了解压缩其他图片，发现有的图片能够成功解压缩，而有些会使程序奔溃。为此我们也分析了图片的一些特征，最终我们认为可能是压缩算法的错误，推测可能是连续的黑色过多导致的问题。

于是我们单独将压缩算法拿出来测试，测试了500个连续的0和500个连续的1的情况，发现压缩结果为0 255 0 0 246 0和0 255 1 0 246 1（前面的0为标记符，至于为什么不能压缩为0 500 1，是因为unsigned char 范围为0-255，无法表示大于255的数），解压后会多一个数据，最终发现是连续相同的数据多于255时，应该把计数的count重置为0而不是1的问题。



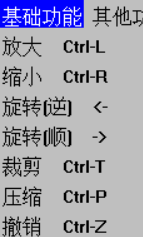
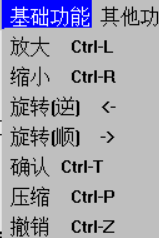
## 使用操作

运行程序后会出现如下的界面：





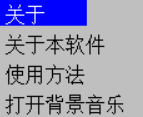
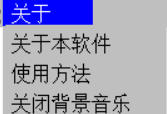
菜单栏中的文件有打开、关闭、保存、退出按键。点击打开按键后会跳出控制台提醒您输入想要打开的文件名称（包括文件后缀），然后如果打开成功文件就会显示在界面中（若失败会弹窗提醒）；关闭就是关闭打开的文件，在文件打开时点击关闭就能关闭文件，若文件被修改且未保存会弹窗提醒；在有图片打开时点击保存按键会弹出控制台，提醒您输入保存后的文件名称，然后将其保存；点击退出按键可以退出程序，若文件被修改且未保存，也会给出提醒。

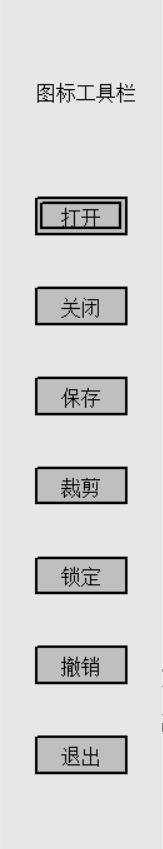
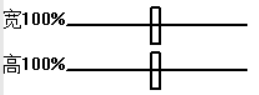
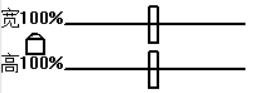
菜单栏中的基础功能有放大、缩小、旋转（逆）。旋转（顺）、裁剪\确认、压缩、撤销按键。点击放大按键可以将图片等比例放大10%，缩小按键可以将图片等比缩小10%；点击旋转（逆）能将图片逆时针旋转90度，旋转（顺）能将图片顺时针旋转90度；点击裁剪会进入裁剪模式，之后裁剪按键会变成确认按键，操作后点击确认就能完成裁剪（注意：裁剪模式下进行某些图片操作，如旋转，会直接退出裁剪模式）；点击压缩会将图像压缩并保存为同名.a文件，用打开按键打开.a后缀的文件会自动解压缩；若不小心进行了错误的图片操作，可以点击撤销按键，图片就会返回至前一步的状态。



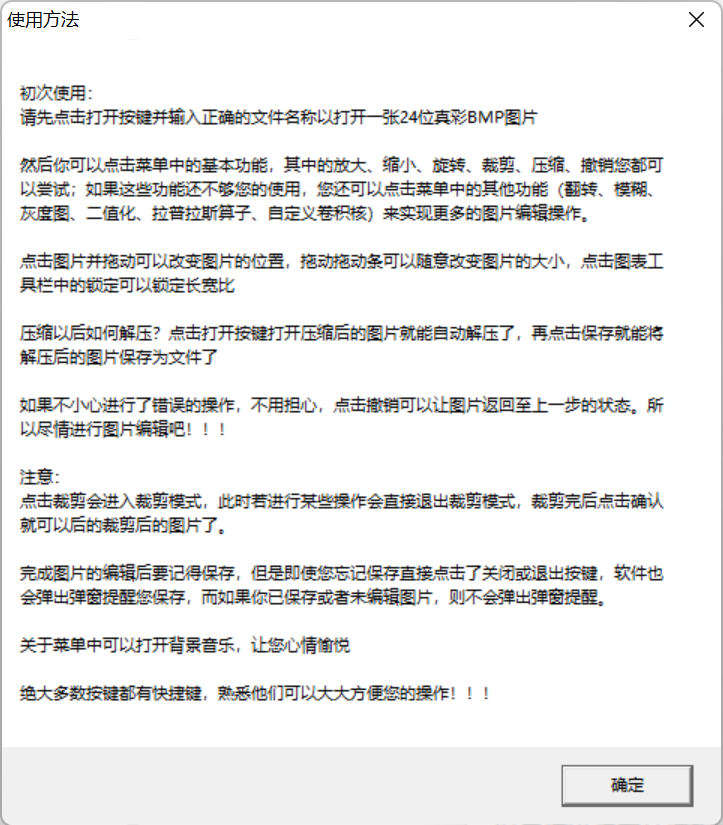
菜单栏中的其他功能有左右翻转、上下翻转、模糊。灰度图、二值化、拉普拉斯算子、自定义卷积核按键。点击左右翻转可以将图像左右翻转，上下翻转可以将图像上下翻转；模糊可以对图片进行高斯模糊处理；灰度图会将图片由彩色转化为灰度图；二值化会将图片根据阈值转化为黑白两色；拉普拉斯算子会先对图片进行二值化操作，随后使用拉普拉斯卷积核对图片进行卷积处理，计算边界；自定义卷积核可以弹出控制台，输入任意的(3\*3)卷积核对图像进行卷积处理。

菜单栏中的关于功能有关于本软件、使用方法、打开背景音乐\关闭背景音乐按键。点击关于本软件会弹出一个弹窗，显示关于本软件的一些信息；点击使用方法会弹出弹窗，显示软件的一些使用方法和技巧；点击打开背景音乐就会开始播放背景音乐，让你心情愉悦，再次点击即可关闭。

图标工具栏的按键除了锁定\解锁以外菜单栏中都有，这里只介绍锁定\解锁按键。在文件打开时，点击锁定按键可以锁定长宽比，此时放缩拖动条之间会出现一个锁的图案，此时拖动长和宽拖动条会按比例改变长宽，再次点击即可解锁。在解锁状态，长和宽可以通过拖动条任意改变。



# 团队合作

## 任务分工

：

代码量约970行，构思所有图片操作的基本逻辑，选定数据结构以及数据类型，并完成图片操作的具体实现，包括图片的读入、保存、显示、缩放、翻转、拖动、旋转等功能。

主要完成bmp.c、linklist.c、convolution.c、callback.c文件的编写。

：

代码量约900行，构思图片操作的交互逻辑，完善弹窗提示功能，编写压缩与解压缩算法，并调用simpleGUI以及Windows API完成UI的绘制，包括菜单栏、图标工具栏、下方信息栏。

主要完成gui.c、compress.c文件的编写。

## 开发计划

## 编码规范

学在浙大课件—代码规范示例—C 编码规范.pdf

## 合作总结

本程序的最终代码量大致为1870行，开发周期约为六周，共计进行了10次小组会议，讨论并实现了程序的全部功能。

**应用知识点：**

1. 模块化程序设计，具体包括预处理命令，全局变量 extern引用，static 全局变量和多文件组织等。

2. 链表数据结构的应用，包括动态内存申请，节点创建，节点插入，节点删除等。

3. 图形程序设计，具体包括第三方图形库基本图形函数，编程模型以及回

调函数等。

**开发亮点：**

1. 撤销功能的实现。本程序采用栈的数据结构来储存每一次操作的结果，实时保存用户对图片进行的操作，也使得图片的数据易于操作和管理。通过此功能，用户可以快捷地撤销操作并使图片返回到上一个状态，防止误操作导致图片数据发生不可逆的丢失。

2. 保证图片不发生失真。在原始的程序中，如果用户先对图像进行缩小操作，随后进行放大，会导致图片发生严重的失真现象。为防止这种情况的发生，本程序建立了一个基准栈，双栈并行，基准栈中保存了图片的源数据，当用户进行操作时，程序将根据全局变量所保存的长宽缩放比例，基于基准数据，实时获取到图片的操作结果。如此一来，无论对图片进行多少次缩放操作，图片都不再会发生失真现象。

3. UI界面与图片的刷新分离。在写用户界面时，屏幕的刷新是实时的，而在整合的过程中我们发现，如果把图片实时刷新会非常的卡顿，图片也会跟不上操作和鼠标的速度，因为画图函数的效率很低，因此不可能实现将图片实时刷新。最终我们决定将图片和UI分离开，经过研究，我们实现了在对图片进行操作时UI（除状态信息栏）不刷新，在非必要的时候不重复画图，解决了卡顿的问题。

4. 使用MessageBox进行必要的弹窗提示。起始我们原本想通过画图画出提示的图形，最终由于过于繁琐，且效果也不好而放弃。最终我们决定采用MessageBox函数，在网络上了解了其用法之后，我们使用它在打开、关闭、保存、退出等操作时跳出弹窗提示，并将关于本软件和使用方法也以弹窗的形式呈现出来，提升了用户体验。

**挑战点：**

1**.**BMP格式图片的读入与保存。BMP格式图片有规定格式的文件头与信息头，如何依据其格式获取到图片的大小、长、宽信息是一大难点，同时在保存图片时，如何重新构造文件头、信息头使得保存的BMP图片能够正常打开亦是一个问题。

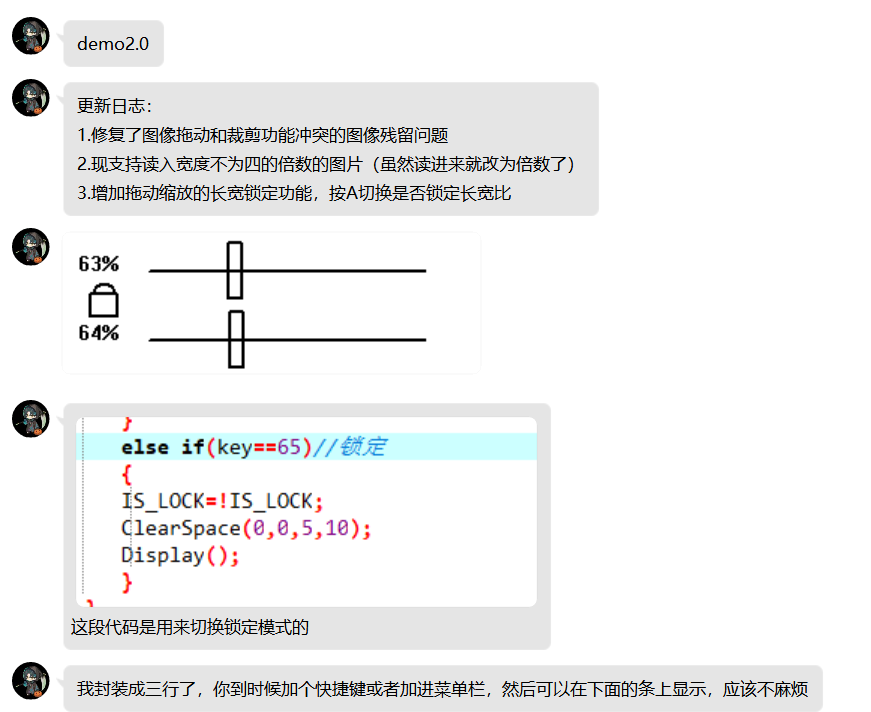
2.BMP图片数据的显示。libgraphics图形库并没有直接提供在窗口显时图片的库函数，为实现图片在指定窗口内的正常显示，我们需要更改库函数，在其中添加在窗口指定位置画点的函数，从而通过逐个打点的方式，将图像数据最终显示在窗口之中。

3.压缩算法的优化。由于BMP图片的格式和其数据量过大，较难对其进行无损压缩，又加上一些特殊的原因，我们采用了RLE算法，但这种算法在遇到非连续数据时会将图片压大，比如1 2会变成1 1 1 2，为此我们决定将0作为标志，在遇到不能压缩的地方就原封不动，以此尽可能地减少将图片压大的问题。（但最终还是可能会出现将图片压大的问题）

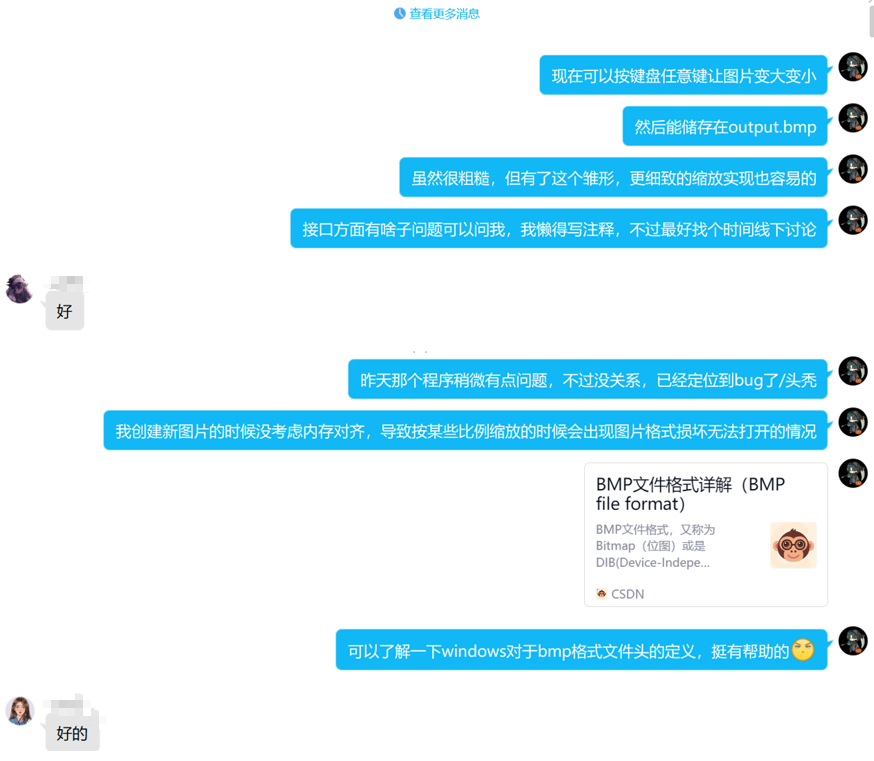
4.操作卡顿的优化。由于图片显示算法的缺陷，当对图片进行某些连续性的操作时（如拖动），短时间内高频率的清屏重画会导致窗口画面异常卡顿。为解决这个问题，我们想到用黑色边框来提示图片位置的方式来减少程序的运算量，增加其运行速率，并分离了UI与图片的刷新，使用户得到更好的交互体验。

**沟通记录：**

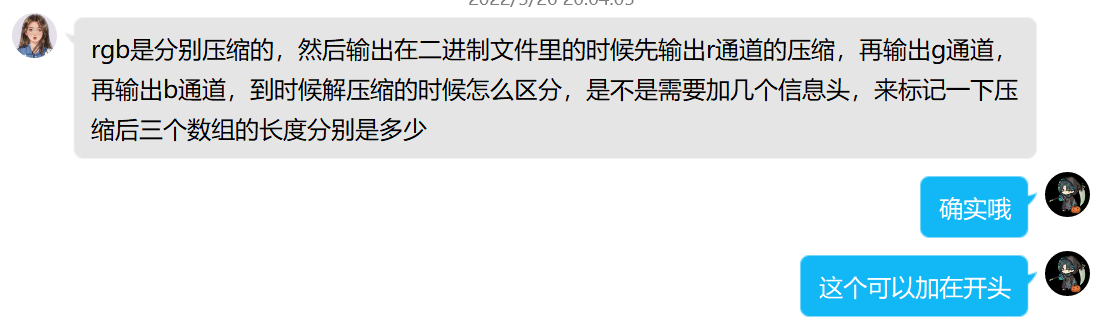
2022年5月4日：



2022年5月14日：



2022年5月20日：



## 收获感言

**团队体会与心得、经验与教训**

对于这样的团队合作项目来说，合理的任务分配以及充分的沟通的是必不可少的。我们深刻理解到了团队合作的重要性，想要实现这个程序的所有功能，仅凭一个人的力量是远远不够的。同时，我们也吸取到了不少的教训，在中期的程序整合过程中，由于事先缺乏交流，导致不同的模块结合时，产生了十分严重的矛盾与冲突，甚至险些要修改程序的底层逻辑。这让我们意识到了明确分工、函数接口、全局变量定义的重要性，还需要尽可能的降低代码之间的耦合度，使得不同功能的模块可以分别独立作用，这将大大降低调试的难度。

**：**

在本次大作业的设计以及开发过程中，我有了许多的感悟。作为本小组的组长，我需要组织小组成员的交流以及分配各自的任务，还需敲定程序模块的划分以及数据结构的选取，而这并不是一件容易的事。我意识到自己未能很好地实现职责，没有在一开始就决定好程序的架构，导致在后续功能的实现中，我们不得不对一些底层的逻辑进行大刀阔斧的修改，大大降低了效率。

但同时，我也从这次体验中收获了很多。我深入了解了BMP格式二进制文件的储存格式，学习了一些图像处理的相关操作，并钻研了库函数以及Windows API的调用，提升了自己的自学能力以及综合编程水平。更重要是，我认识到了在小组合作的程序开发过程中，交流是无比重要的一环，在缺乏沟通的情况下，各自模块的整合将会变得无比的困难，许多问题也会在其中爆发出来。

在此次程序开发中，我认为自己的表现尚可，但由于缺乏经验，依然犯了不少的错误。我会吸取这些教训，尽量保证在以后的程序开发中不出现类似的失误。同时也感谢小组成员的配合，能够认真完成自己的任务并积极参与讨论，最终解决了这项任务艰巨的作业。

**：**

本次合作开放的过程中，我收获了很多。现代的软件开发基本都是由团队合作完成的，一个人是很难独立完成一个专业的大型的软件的，因此团队合作的重要性不言而喻。在这次合作开发中，我认为前期我们的交流不太足够，主要是各自完成各自的部分。但在中后期我们加强了交流，基本做到每个星期当面交流1-2次，最终成功完成了大程开发。

其次，在这次程序开发中，我成功地把课堂中的所学和课外所学应用到实践中来，虽然这次开发中我的任务主要是GUI和压缩/解压缩，没有涉及太多对图片的操作，但是通过GUI的制作我学会了如何利用已有的不了解的库进行编程，而压缩/解压缩让我将学过的字符串处理方式应用到实践中来，总体而言，我提高了自己的自学能力和实践能力。

在这次程序开发中，我认为自己表现的还好，但说不上是优秀，因此我也应该好好反思自己的一些问题与错误，另外也要感谢我的队友，能够在遇到困难时相互鼓励，一起解决。也非常感谢这门课让我积累了一定的程序设计经验，这一定能够让我在以后的学习和工作收益。

# 参考文献资料

<https://blog.csdn.net/chunqi7983/article/details/100680012>

（BMP文件格式详解）

<https://blog.csdn.net/zhongyanghu27/article/details/8221312>

（RLE压缩算法）

<https://blog.csdn.net/qq_21237549/article/details/104264621>

（DEV C++设置背景音乐）

<https://blog.csdn.net/QQ1910084514/article/details/80754123>

（PlaySound函数用法）

<https://blog.csdn.net/qq_36333591/article/details/88858314>

（MessageBox函数用法）