暨南大学本科实验报告专用纸

课程名称 数字图像处理 成绩评定

实验项目名称 Bmp 格式图像文件的读入与显示，24 位真彩图像转换为灰度图像，访问像素数据 指导教师 刘晓翔

实验项目编号 01 实验项目类型 综合型 实验地点 三楼机房

学生姓名 赵俊文 学号 2022104002

学院 智能科学与工程学院 系 专业 人工智能

实验时间 2024 年 9 月 2 日 上 午~ 9 月 16 日 上 午 温

（一） 实验目的

掌握 Bmp 格式图像文件的存储格式及显示方法；掌握灰度图像的调色板特点及其位图数据的存储方式。

（二） 实验内容和要求

利用 Visual C++6.0 软件开发工具编写程序，实现图像的读入与显示，执行结果应正确；实现 24 位真彩图像转换为灰度图像；对于不同颜色深度的图像，实现任意指定像素点的颜色值的读取与显示。

（三） 主要仪器设备

仪器：计算机

实验环境： Windows XP + Visual C++6.0

（四） 实验步骤怕（附代码）与调试

1. 实现图像的读入与显示

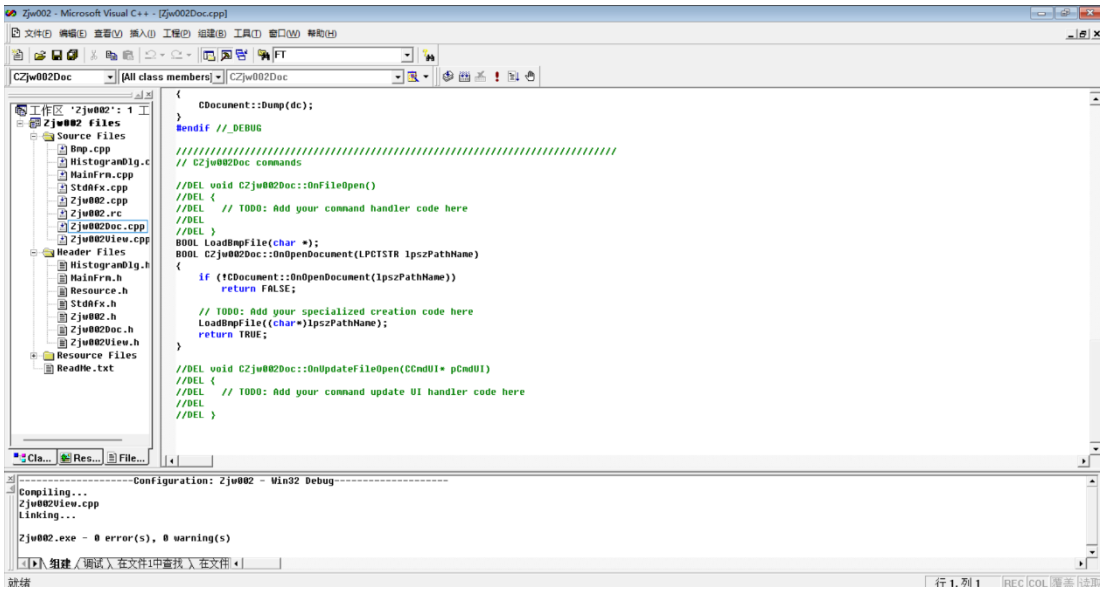
a.首先创建一个 bmp.cpp 文件，用于编写各种数字图像处理方法的代码，然后在 bmp 中编写函数 LoadBmpFile，用于实现图像的加载。LoadBmpFile 函数代码如下：t

```
BITMAPINFOHEADER bi;

BOOL LoadBmpFile(char* BmpFileName){
```

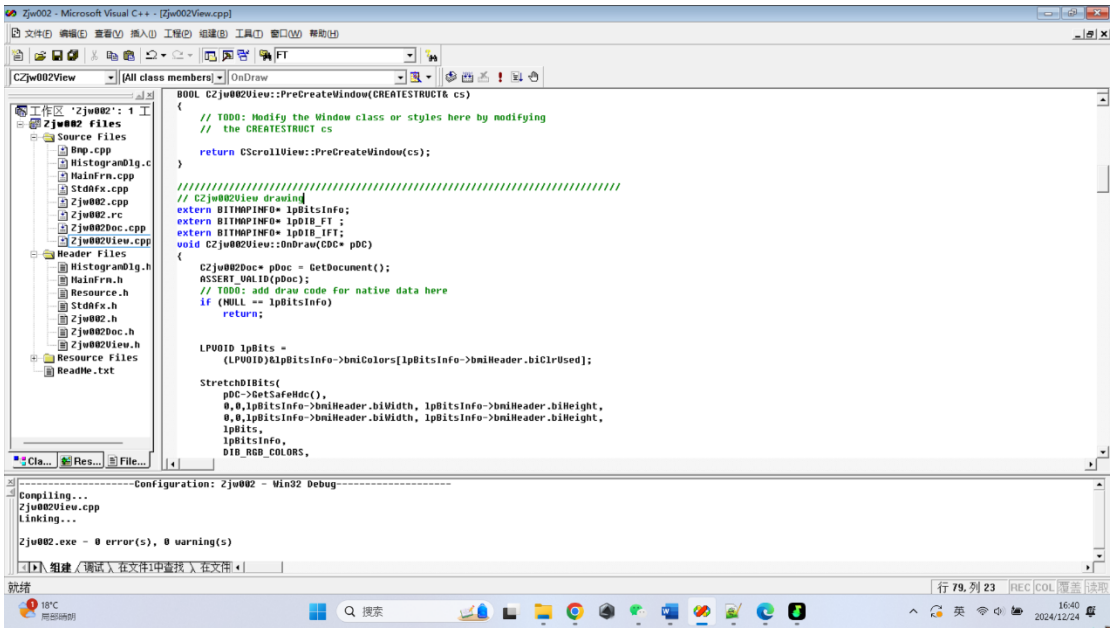
```
FILE* fp;
if (NULL==(fp=fopen(BmpFileName,"rb")))
    return FALSE;

BITMAPFILEHEADER bf;
BITMAPINFOHEADER bi;

fread(&bf,14,1,fp);
fread(&bi,40,1,fp);

DWORD NumColors;
if (bi.biClrUsed!=0)
    NumColors = bi.biClrUsed;
else
{
    switch(bi.biBitCount)
    {
    case 1:
        NumColors = 2;
        break;
    case 4:
        NumColors = 16;
        break;
    case 8:
        NumColors = 256;
        break;
    case 24:
        NumColors = 0;
        break;
    }
```

```
    }

    DWORD PalSize = NumColors * 4;
    DWORD ImgSize = (bi.biWidth * bi.biBitCount + 31) / 32 * 4 *
bi.biHeight;
    DWORD Size = 40 + PalSize + ImgSize;

    if (NULL == (lpBitsInfo = (BITMAPINFO*)malloc(Size)))
        return FALSE;

    fseek(fp,14,SEEK_SET);
    fread((char*)lpBitsInfo,Size,1,fp);

    lpBitsInfo->bmiHeader.biClrUsed = NumColors;

    return TRUE;
    }
```

b.在文件类中调用上述函数

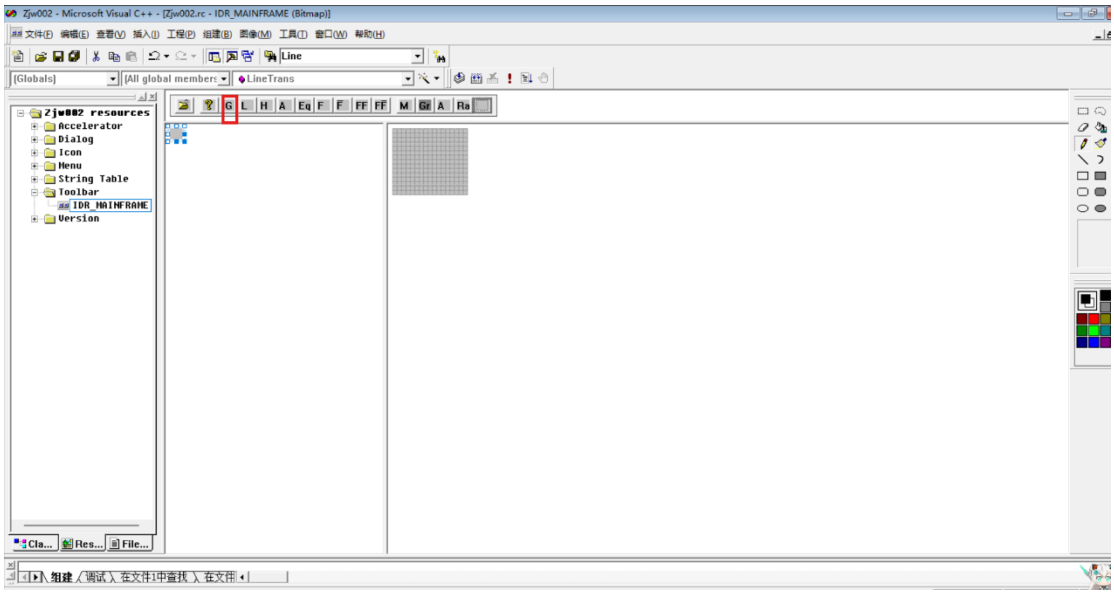


c.在视类中的 onDraw 函数中编写代码，使其可以成功显示图像



2. 实现 24 位真彩图像转换为灰度图像

a.在工具栏中添加按钮

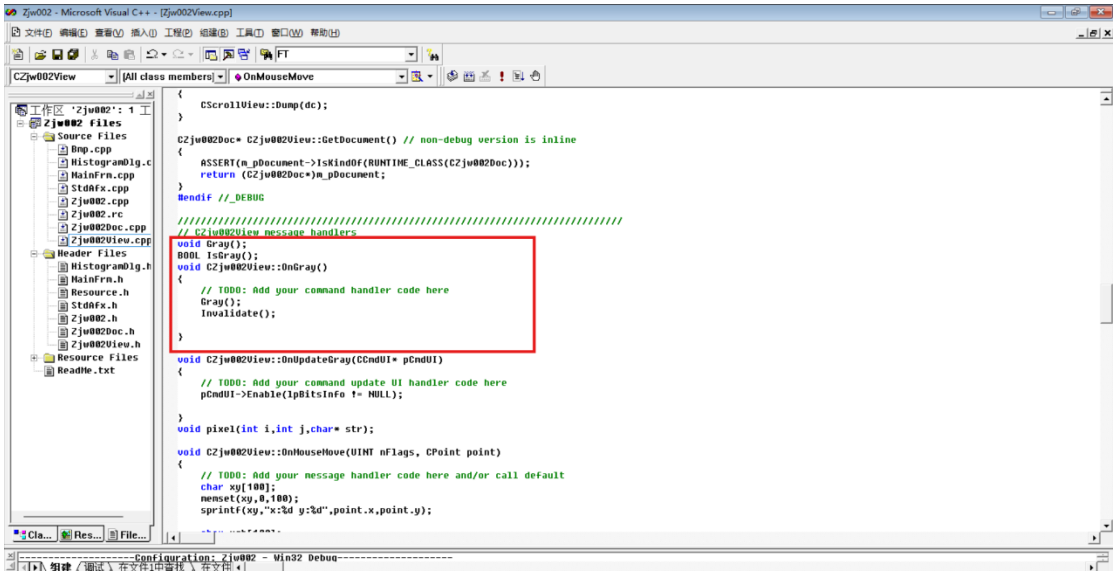


b.在 bmp 文件中编写灰度函数，在视图类调用之。函数代码如下

```
void Gray(){
    int w = lpBitsInfo->bmiHeader.biWidth;
    int h = lpBitsInfo->bmiHeader.biHeight;
    int LineBytes = (w * lpBitsInfo->bmiHeader.biBitCount + 31)/32 *
4;
```

```
    BYTE* lpBits =
(BYTE*)&lpBitsInfo->bmiColors[lpBitsInfo->bmiHeader. biClrUsed];

    int LineBytes_gray =(w *8+31)/32 *4;
    BITMAPINFO* lpBitsInfo_gray=(BITMAPINFO*) malloc(40 + 1024 +
LineBytes_gray * h);
    memcpy(lpBitsInfo_gray,lpBitsInfo, 40);

    lpBitsInfo_gray->bmiHeader.biBitCount = 8;
    lpBitsInfo_gray->bmiHeader.biClrUsed = 256;

    int i,j;
    for(i=0;i<256;i++)
    {
        lpBitsInfo_gray->bmiColors[i].rgbRed =i;
        lpBitsInfo_gray->bmiColors[i].rgbGreen=i;
        lpBitsInfo_gray->bmiColors[i].rgbBlue=i;
        lpBitsInfo_gray->bmiColors[i].rgbReserved=0;
    }

    BYTE* lpBits_gray=(BYTE*) &lpBitsInfo_gray->bmiColors[256];
    BYTE *R ,*G,*B,avg,*pixel;
    for(i=0;i<h;i++){
        for(j=0;j<w;j++){
            B=lpBits+LineBytes*(h-i-1)+j*3;
            G=B+1;
            R=G+1;
            avg=(*R+*G+*B)/3;
            pixel=lpBits_gray+LineBytes_gray*(h-i-1)+j;
```

```
            *pixel=avg;
        }
    }

    free(lpBitsInfo);
    lpBitsInfo=lpBitsInfo_gray;
    }
```



3. 对于不同颜色深度的图像，实现任意指定像素点的颜色值的读取与显示

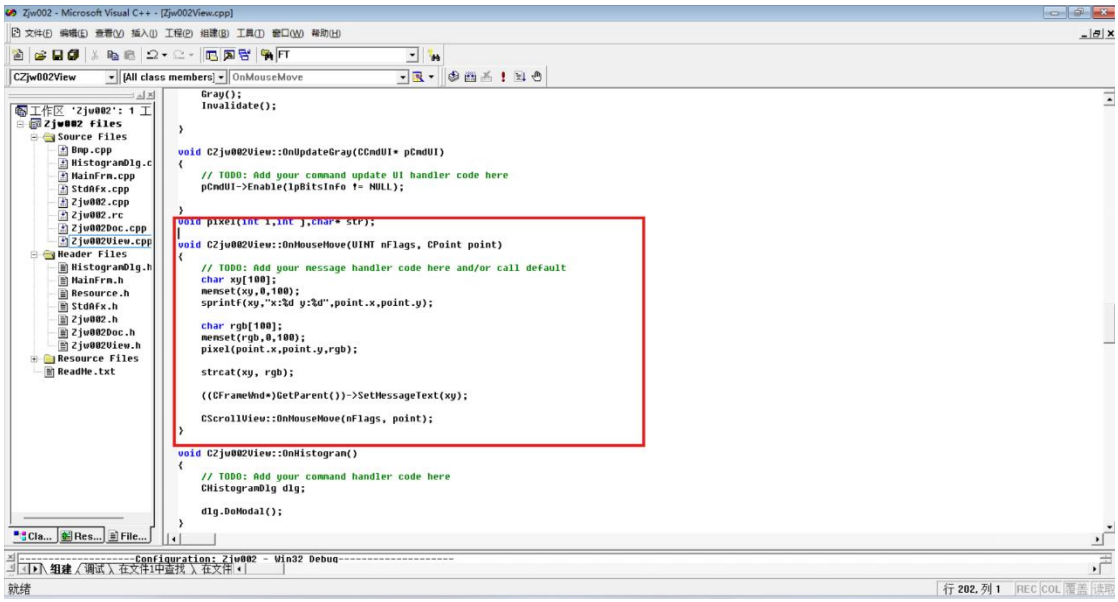
a.在 bmp 中编写函数计算每个像素点的 RGB 或灰度值并输出，然后在视图类调用。

函数代码：

```
void pixel(int i,int j,char* str)
{
    if(NULL == lpBitsInfo)
        return;

    int w = lpBitsInfo->bmiHeader.biWidth;
    int h = lpBitsInfo->bmiHeader.biHeight;
    int LineBytes = (w * lpBitsInfo->bmiHeader.biBitCount + 31)/32 *
```

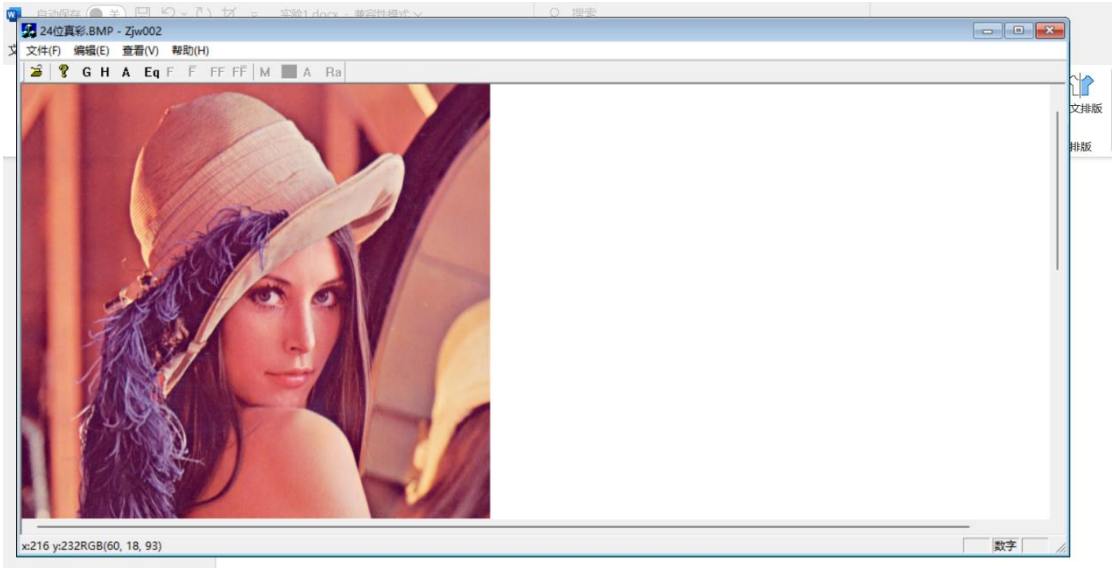
```
4;
    BYTE* lpBits =
(BYTE*)&lpBitsInfo->bmiColors[lpBitsInfo->bmiHeader.biClrUsed];

    if (i >= h || j >=w)
        return;

    BYTE* pixel, bv;
    int r,g,b;
    int colorIdx = 0;

    switch(lpBitsInfo->bmiHeader.biBitCount)
    {
    case 8:

        pixel = lpBits + LineBytes * (h - 1 - i) + j;
        if (IsGray())
           sprintf(str,"灰度:%d", *pixel);
        else
        {

            r = lpBitsInfo->bmiColors[*pixel].rgbRed;
            g = lpBitsInfo->bmiColors[*pixel].rgbGreen;
            b = lpBitsInfo->bmiColors[*pixel].rgbBlue;
            sprintf(str,"RGB(%d, %d, %d)", r,g,b);
        }
        break;
    case 24:
        pixel = lpBits + LineBytes * (h - 1 - i) + j * 3;
        r = pixel[0];
```

```
            g = pixel[1];
            b = pixel[2];
            sprintf(str,"RGB(%d, %d, %d)", r,g,b);
            break;
        case 4:
            bv = *(lpBits + LineBytes * (h - 1 - i) + j / 2);
            colorIdx = (j % 2 == 0) ? (bv >> 4) : (bv & 0x0f);
            r = lpBitsInfo->bmiColors[colorIdx].rgbRed;
            g = lpBitsInfo->bmiColors[colorIdx].rgbGreen;
            b = lpBitsInfo->bmiColors[colorIdx].rgbBlue;
            sprintf(str,"RGB(%d, %d, %d)", r,g,b);
            break;
        case 1:
            bv = *(lpBits + LineBytes * (h - 1 - i) + j/8) & (1 << (7 -
j % 8));
            if(0 == bv)
                strcpy(str,"背景点");
            else
                strcpy(str,"前景点");
            break;
        }

}
```

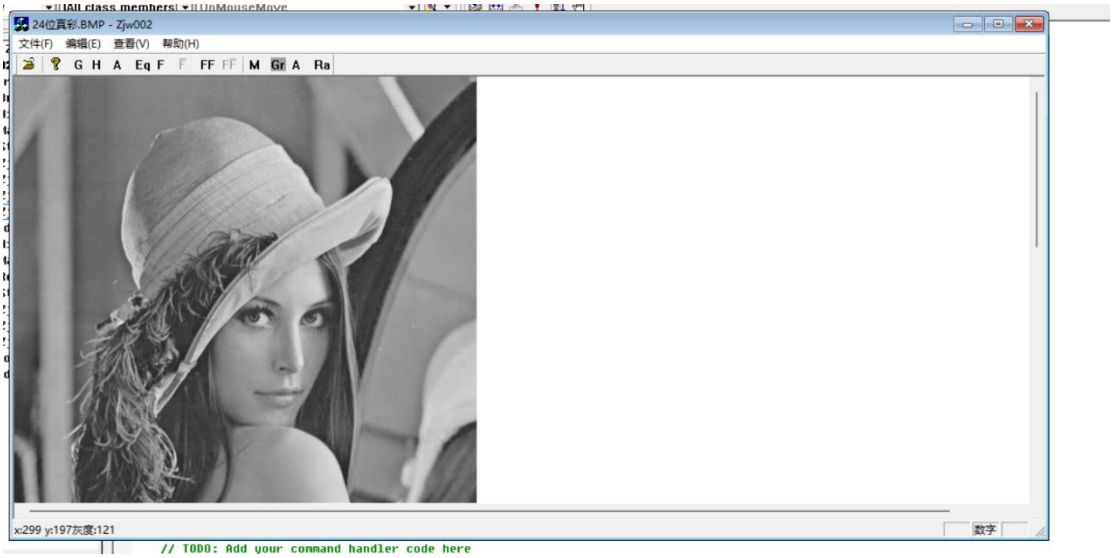


（五） 实验结果与分析

1. 图像的读入与显示，同时显示像素点的 RGB 值



2. 实现 24 位真彩图像转换为灰度图像,同时显示像素点的灰度值



3. 实验分析

通过本次实验,我初步了解了数字图像处理的概念,掌握了加载和显示图像,计算和显示图像灰度值（前景点和背景点）或者 RGB 值以及将彩色图像灰度化的方法.