

课程设计任务书

学生姓名：_____ 专业班级：_____

指导教师：_____ 工作单位：_____

题 目：

初始条件：

要求完成的主要任务：（包括课程设计工作量及其技术要求，以及说明书撰写等具体要求）

时间安排：

指导教师签名：_____ 年 月 日

系主任（或责任教师）签名：_____ 年 月 日

目录

摘要.....2

第一章 引言.....3

 1.1基础知识.....3

 1.2本数字图像处理界面处理内容介绍.....4

 1.2.1启动界面.....4

 1.2.2主界面.....4

第二章 系统分析.....4

 2.1设计要求.....4

 2.2系统处理流程分析.....5

第三章 系统设计.....5

 3.1系统的设计语言.....5

 3.2系统总体结构设计.....5

 3.2.1系统总体结构图.....5

 3.2.2系统模块设计.....6

第四章 系统详细设计.....7

 4.1基类的设计.....7

 4.2各模块的详细设计.....8

 4.2.1欢迎界面设计.....8

 4.2.2主界面设计.....9

第五章 结论与心得体会.....19

第六章 参考文献.....20

第七章 关键代码.....21

基于 VC++平台简单数字图像处理界面设计

摘要:该数字图像处理界面是基于 Microsoft Visual C++ 6.0 MFC 多文档编程的图像处理界面, 实现.bmp 格式图像的浏览与编辑, 导入和导出, 图像特效显示,边缘检测,旋转和翻转图片, 图片增强优化, 以及打印输出图片。是一款功能较简洁的数字图像管理界面,实现对.bmp 格式图像的操作。该界面包含两部分:启动界面和主界面。

BMP 文件是 Windows 操作系统所推荐和支持的图像文件格式,是一种将内存或显示器的图像数据不经过压缩而直接按位存盘的文件格式,一般来说 BMP 文件分成 4 个部分:位图头文件,位图信息头,颜色表,位图数据.同时 BMP 文件文件可以分为 DIB 格式和 DDB 格式,其中 DIB 格式是与设备无关的 BMP 文件,通用性很强.Windows 支持一些重要的 DIB 访问函数,但都没有封装到 MFC 中,这里我们封装了一个 DIB 类,其包含 DIB 文图处理所需要的基本成员变量和成员函数,进而以该类为基类派生其他类,实现程序的维护和移植,从而利用面向对象的方式实现图像的可视化编程。

关键词: BMP 文件 图像处理 DIB 位图 界面

Abstract:The digital image processing interface is based on Microsoft Visual C++ 6.0 MFC multi-document image processing programming interface, to achieve. Bmp format for image browsing and editing, import and export, image effects showed that edge detection, rotate and flip images, pictures enhance the optimization, as well as print output picture. Is a function of the digital image than the simple management interface, to achieve. Bmp format images of the operation. The interface contains two parts: interface and the main interface to start.

BMP file is a Windows operating system and support the recommended image file format, is a memory or display the compressed image data is not directly based on digital archiving of file formats, BMP files are generally divided into four parts: the first bitmap documents, BITMAPINFOHEADER, color table, the bitmap data. At the same time, BMP file formats and can be divided into DIB format DDB, DIB format which has nothing to do with the equipment of the BMP files, a strong GM. Windows in support of the DIB to visit a number of important function, but are not packaged in the MFC, where we have a package type DIB, the DIB includes Ventura needed to deal with members of the basic variables and member functions, and such a base class for other types of derivatives, the realization of the procedure maintenance and transplantation, and thus the use of object-oriented way of programming the visual image.

Key words: BMP file Image processing DIB bitmap Interface

1 引言

数字图像处理（Digital Image Processing）又称为计算机图像处理，它是指将图像信号转换成数字信号并利用计算机对其进行处理的过程。数字图像处理最早出现于 20 世纪 50 年代，当时的电子计算机已经发展到一定水平，人们开始利用计算机来处理图形和图像信息。随着计算机的发展,数字图像处理技术也不断的进步,实现了对图像图像增强、复原、编码、压缩等,在空间和军事高科技中扮演越来越重要的角色。

随着数字化与多媒体时代的来临，数字图像处理已经成为必备的基础知识。近几十年来由于计算机技术的蓬勃发展，图像处理技术也得到了空前的发展和应用。目前，图像处理技术已经广泛应用于工业、军事、医学、交通、农业、天气预报、银行、超市、重要部门的监控报警系统、可视电话、网络传输等等领域，成为各个学科学习和研究对象。随着图像处理技术的广泛应用，学习和掌握这门科学显得格外重要，图像处理已经成为信息技术相关领域的核心课程。

本课题主要实现图像的一些简单特效显示，图像的旋转和镜像、图像的增强处理,图像灰度变换,图像边缘检测等。

1.1 基础知识

组成数字图像的基本单位是像素(Pixel),也就是说数字图像是像素的集合。显示器在显示图像时采用电子枪从左到右扫描一行,为每个像素着色,然后从上倒下扫描整个屏幕的方式,由于人眼的视觉停留,可以显示出完整的画面.为了防止闪烁,每秒电子枪要重复扫描几十次的扫描过程,人眼感受不到因屏幕刷新也产生的闪烁,这种显示器称为位映像设备。所谓位映像,是指一个二维的像素矩阵,矩阵的元素值就是像素的灰度值,而位图就是采用位映像方法显示和存储的图像。

彩色图像和灰度图像:灰度图像是数字图像最基本形式,可以从黑白照片数字化得到或则通过对彩色图像去色处理得到,灰度图像只表示图像的亮度信息而没有颜色信息,灰度图像的每个像素点上只包含一个量化的灰度级(0-255来表示该点的亮度水平,计算机一般用 1 个字节(8 个二进制位)来存储。彩色图像的数据不仅包含亮度信息,还包含颜色信息,图像中每一个像素由 RGB(红绿蓝)三个分量组成,计算机一般用 3 个字节(24 位二进制)来存储像素的颜色信息。

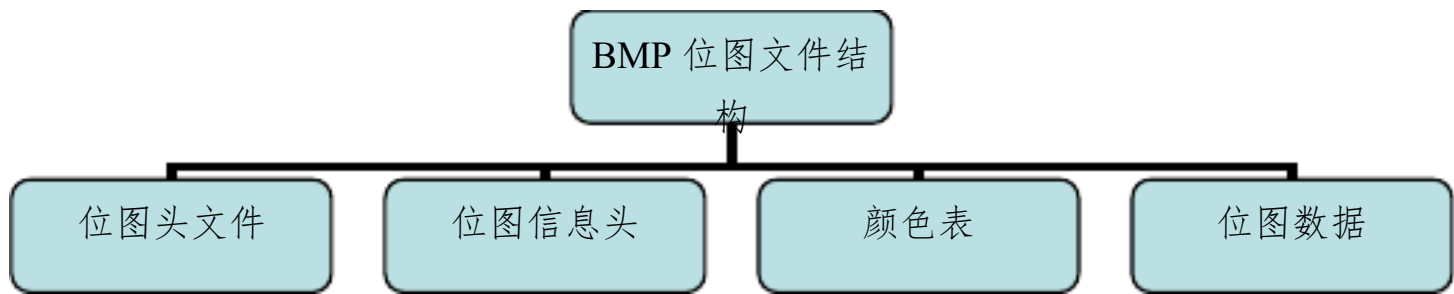


图 1.1 BMP 图像文件结构

如图 1.1 所示 BMP 文件由文件头、位图信息头、颜色信息和图形数据四部分组成。在编程时它们以结构体类型在计算机中存储。

- 1、位图文件头：主要包含文件类型(必须是 0x424D，即字符串“BM”)，文件大小以及文件头到实际的位图数据的偏移字节数；
- 2、位图信息头：包含图象的尺寸信息、每个像素所占位数(bit)、图像是否压缩、图像所用的颜色数等信息，颜色信息包含图像所用到的颜色表，显示图像时需用到这个颜色表来生成调色板；
- 3、调色板(Palette：调色板实际上是一个数组存有 RGB 信息的表；
- 4、位图数据：对于用到调色板的位图，图像数据就是该象素颜在调色板中的索引值。对于 24 位真彩色图，图像数据就是实际的 R、G、B 三个分量的值。

1.2本数字图像处理界面处理内容介绍

1.2.1启动界面

为了追求该图像处理的界面的美观性,在设计主界面前,先设计一个欢迎界面,即以一个经过 photoshop 处理过的图像作为对话框,在运行主界面程序前,先显示该对话框。

1.2.2主界面

该数字图像处理界面是基于 VC++6.0MFC 多文档编程的界面程序，实现数字图像的简单查看与编辑操作。主界面主要内容有：

- 图像文件浏览:实现对.bmp 格式图像文件的浏览。
 - 图像特效显示:主要有自上而下扫描显示,从左至右的滑动显示和渐进显示。
 - 视图-文档的设置:实现了执行操作后新旧图像在不同视图中显示设置。
 - 图像的灰度变换:包括图像的二值化,彩色图像转换灰度图像,灰度图像转换 8 位图像。
 - 图像的镜像与旋转变换:包括水平镜像,垂直镜像,顺时针旋转 90 度,逆时针旋转 90 度,图像旋转 180 度。
 - 图像的增强处理:主要是图像锐化处理。
 - 图像的边缘检测和形态学处理:包括罗伯茨算子边缘检测,图像轮廓提取
- 该界面主要实现了.bmp 格式图像的浏览、编辑、打印、特效设置、视图设置、变换等几大功能。

2 系统需求分析

需求分析简单地说就是分析用户的需求。需求分析是设计数据库的起点，需求分析的结果是否准确地反映了用户的实际要求,将直接影响到后面各个阶段的设计，并影响到设计结果是否合理和实用。

2.1设计要求

根据设计内容,设计的主要要求有:(1设计欢迎界面并在程序中实现欢迎界面先于主界面显示;(2实现主界面的功能设计:图像文件浏览(.bmp 格式图像的导入),图像特效显示,视图-文档的设置,图像的灰度变换,图像的镜像与旋转变换,图像的增强处理,图像的边缘检测和形态学处理。

2.2系统处理流程分析

通过对系统的组织结构和功能进行分析,画出业务流程图,它非常易于阅读和理解,可以帮助我们了解该业务的具体处理过程，发现和处理系统的调查工作中的错误和疏漏，修改和删除原系统的不合理部分。

通过对系统的需求分析,系统的基本功能已经确定。整个系统各个模块的业务流，如图 2.1所示：

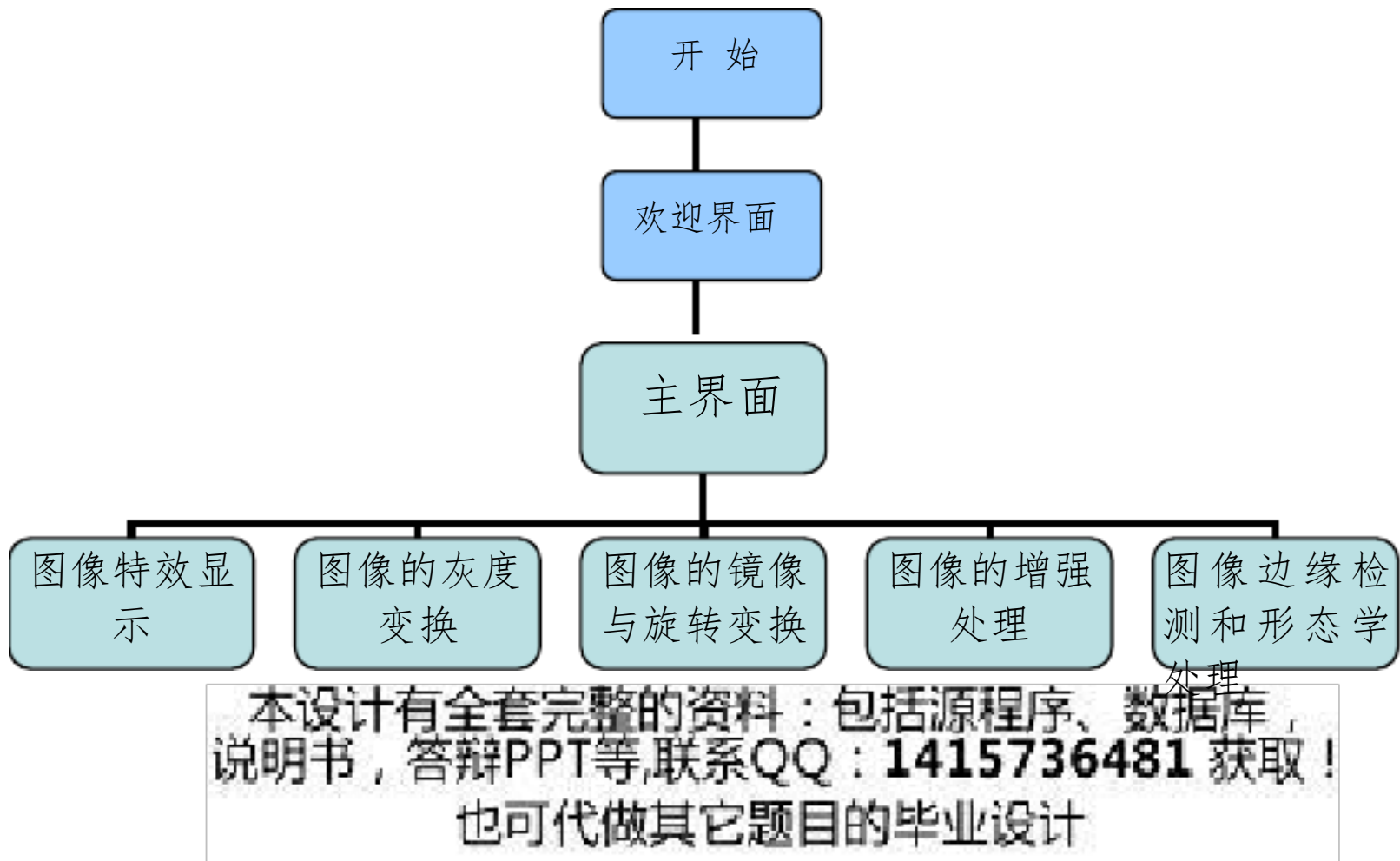


图 2.1业务流程图

3 系统设计

本阶段的主要任务是设计程序的体系结构，确定程序组成模块，以及各模块之间的关系。

3.1 系统的设计语言

数字图像处理系统采用 Microsoft Visual C++ 6.0 MFC 作为开发平台。

3.2 系统总体结构设计

3.2.1 系统总体结构图

数字图像处理界面的总体结构设计如图 3.1所示：

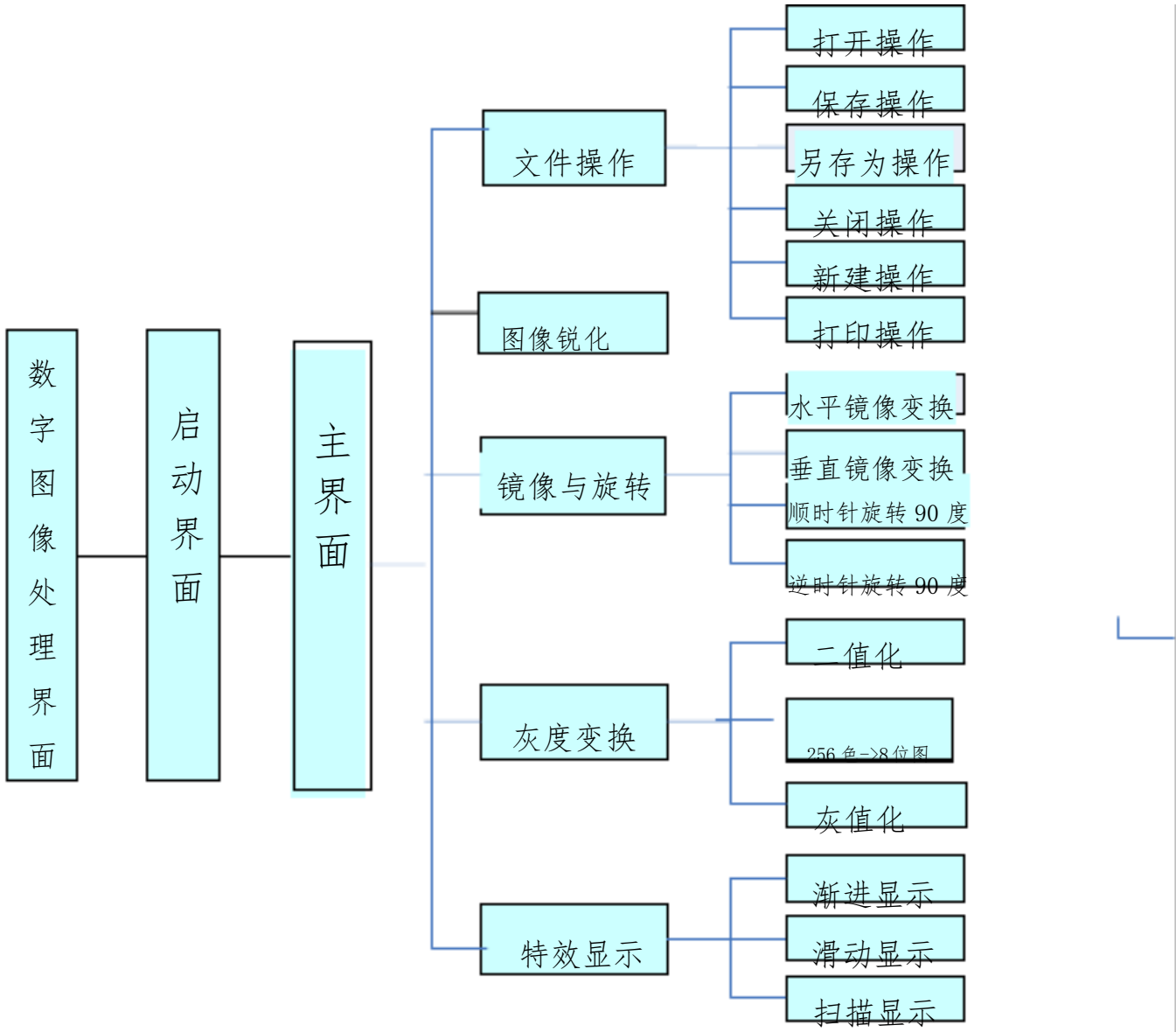


图 3.1 数字图像处理界面总体结构设计

3.2.2 系统模块设计

模块设计是对处理流程图的注释性书面文件，以帮助程序设计人员进一步了解模块的功能和设计要求数字图像处理系统主要分为两大模块，即欢迎界面和主界面,其中主界面又分为文件操作模块,特效显示模块,图像镜像与旋转模块,图像增强处理模块,图像边缘检测和形态学处理模块。

具体的模块设计如下:

1.欢迎界面:该模块实现以一个经过photoshop 处理过的图像作为对话框,在运行主界面程序前,先显示该对话框。

2.主界面:

(1) 文件操作模块: 该模块主要实现图像的文件操作,包括新建操作,打开操作.保存操作,另存为操作,关闭操作,打印操作等。

(2) 特效显示模块:该模块实现了图像在视图中的显示方式,包括自上而下扫描显示,从左至右的滑动显示和渐进显示。

(3) 图像镜像与旋转模块:该模块实现了图像的镜像变换和图像旋转显示,包括水平镜像,垂直镜像,顺时针旋转 90 度,逆时针旋转 90 度,图像旋转 180 度。

(4) 图像增强处理模块:该模块主要实现了图像锐化效果。

(5) 图像边缘检测模块:该模块实现了图像边缘轮廓检测,包括罗伯茨算子边缘检测,图像轮廓提取。

(6) 图像灰度变换模块:该模块实现了彩色图像变灰度格式,灰度图像变彩色格式和二值化操作。

4 系统详细设计

4.1 基类的设计

BMP 文件是 Windows 操作系统所推荐和支持的图像文件格式,是一种将内存或显示器的图像数据不经过压缩而直接按位存盘的文件格式,同时 BMP 文件文件可以分为 DIB 格式和 DDB 格式,其中 DIB 格式是与设备无关的 BMP 文件,通用性很强。Windows 支持一些重要的 DIB 访问函数,但都没有封装到 MFC 中,这里我们封装了一个 DIB 类,其包含 DIB 位图处理所需要的基本成员变量和成员函数,进而以该类为基类派生其他类,实现程序的维护和移植。在这里我们命名这个基类为 `ImgCenterDib`,各种算法的实现都是以 `ImgCenterDib` 类为基类派生出来。根据前面预备知识和程序编写的需要,这里定义 `ImgCenterDib` 类代码如下:

```
class ImgCenterDib
{
public:
    unsigned char * m_pImgData;    //图像数据指针
    LRGBQUAD m_lpColorTable; //图像颜色表指针
    int m_nBitCount; //每像素占的位数
private:
    LPBYTE m_lpDib; //指向 DIB 的指针
    HPALETTE m_hPalette; //逻辑调色板句柄
    int m_nColorTableLength; //颜色表长度 (多少个表项)
protected:
    int m_imgWidth; //图像的宽, 像素为单位
    int m_imgHeight; //图像的高, 像素为单位
    LPBITMAPINFOHEADER m_lpBmpInfoHead; //图像信息头指针
public:
    ImgCenterDib(); //不带参数的构造函数
    ImgCenterDib(CSize size, int nBitCount, LRGBQUAD lpColorTable,
        unsigned char *pImgData); //带参数的构造函数
    ~ImgCenterDib(); //析构函数
    CSize GetDimensions(); //获取 DIB 的尺寸 (宽高)
    BOOL Read(LPCTSTR lpszPathName); //DIB 读函数
    BOOL Write(LPCTSTR lpszPathName); //DIB 写函数
    BOOL Draw(CDC* pDC, CPoint origin, CSize size); //显示 DIB
    void ReplaceDib(CSize size, int nBitCount, LRGBQUAD lpColorTable,
```

```
    unsigned char *pImgData); //用新的数据替换 DIB
    int ComputeColorTabalLength(int nBitCount); //计算颜色表的长度
private:
    void MakePalette();//创建逻辑调色板
    void Empty();    /清理空间    };
```

同时将各个模块函数封装起来成为一个类,这些类都以 **ImgCenterDib** 类共有继承,在主程序中调用,减少代码的重复性,提高利用率。派生关系如图 4. 1:

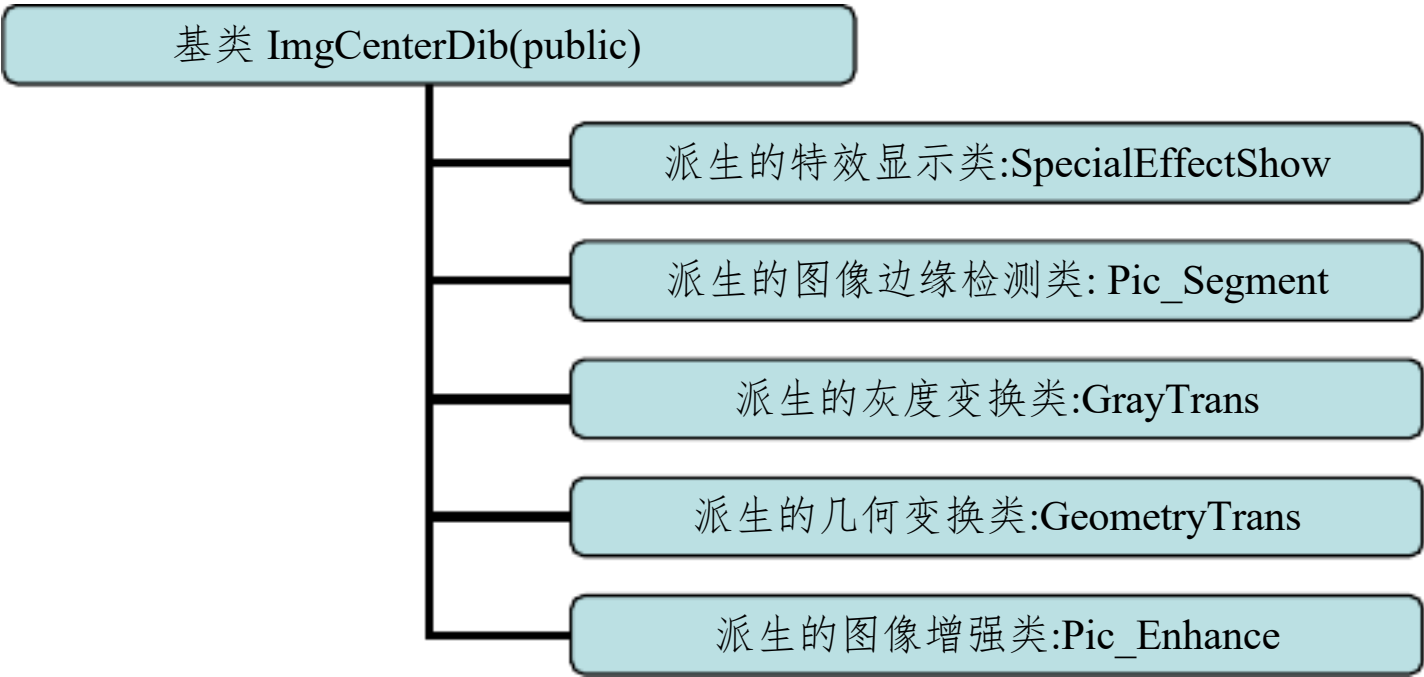


图 4. 1 基类与派生类结构关系图

4. 2 各模块的详细设计

4. 2. 1欢迎界面设计

用 **photoshop** 软件根据内容的需要处理一张图片,并保存为**.bmp** 格式,然后在 **Microsoft Visual C++ 6.0** 中载入该图片。

设计思想:欢迎界面在程序运行时是以图形式显现的,这里要封装一个位图的类 **Cbtmap**(派生于位图类 **CBitmap**),获取图像的宽,高,创建逻辑调色板等等;之后创建一个启动窗口类, 名为 **Csplash**(派生于窗口基类 **CWnd**) 这两个类均封装了一些函数,它们之间通过参数传递和函数调用,在主程序代码中声明 **Splash** 类的对象,调用 **Csplash** 类中函数从而实现设计目标。

启动窗口类的主要代码如下:

```
class Csplash : public CWnd
{
public:
    Csplash();//构造函数
public:
    void Create( UINT nBitmapID );//调用 CreateEx 创建窗口
public:
    virtual ~Csplash();//析构函数
protected:
   //{{AFX_MSG(CWzdSplash)
    afx_msg void OnPaint();
   //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
private:
    Cbtmap m_bitmap;//私有成员变量
};
```

```
class Cbtmap : public CBitmap
{
public:
    DECLARE_DYNAMIC(Cbtmap)
    Cbtmap();
    void LoadBitmapEx(UINT nID, BOOL bIconBkgrd );
    CPalette *GetPalette(){return m_pPalette;};
public:
    virtual ~Cbtmap();
    int m_Width;
    int m_Height;
private:
    CPalette *m_pPalette;
};
```

在 CPicture_sysView::CPicture_sysView()函数中加入如下代码:

```
Csplash wndSplash; //声明 Splash 类的对象
wndSplash.Create(IDB_WZDSPLASH); //调用 Create()
wndSplash.UpdateWindow();
Sleep(2000); //定时 2 秒
```

程序运行截图如下: 欢迎界面)

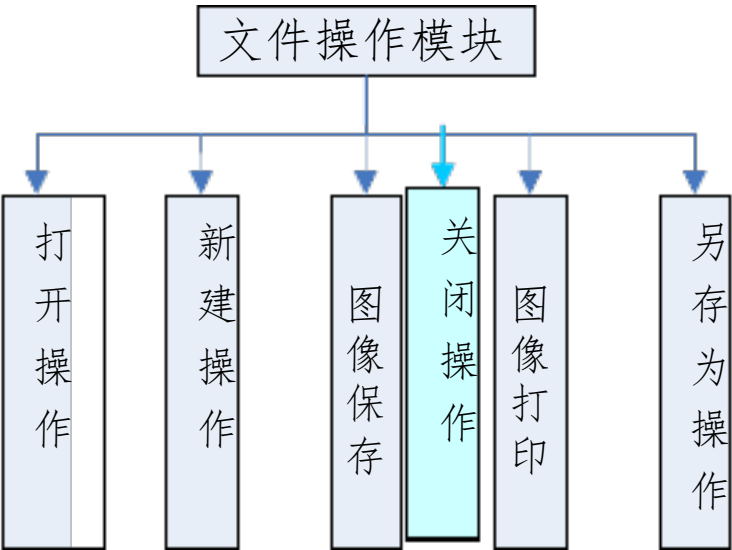


主界面设计

(1) 文件操作模块

文件操作模块是整个系统中最基本的组成成分,该模块主要实现图像的文件操作,包括新建操作,打开操作.保存操作,另存为操作,关闭操作,打印操作。

通过对用户的需求分析得出文件操作模块的功能结构图,文件操作模块的功能结构图如图 4. 2所示:



4.2 文件操作模块功能结构图

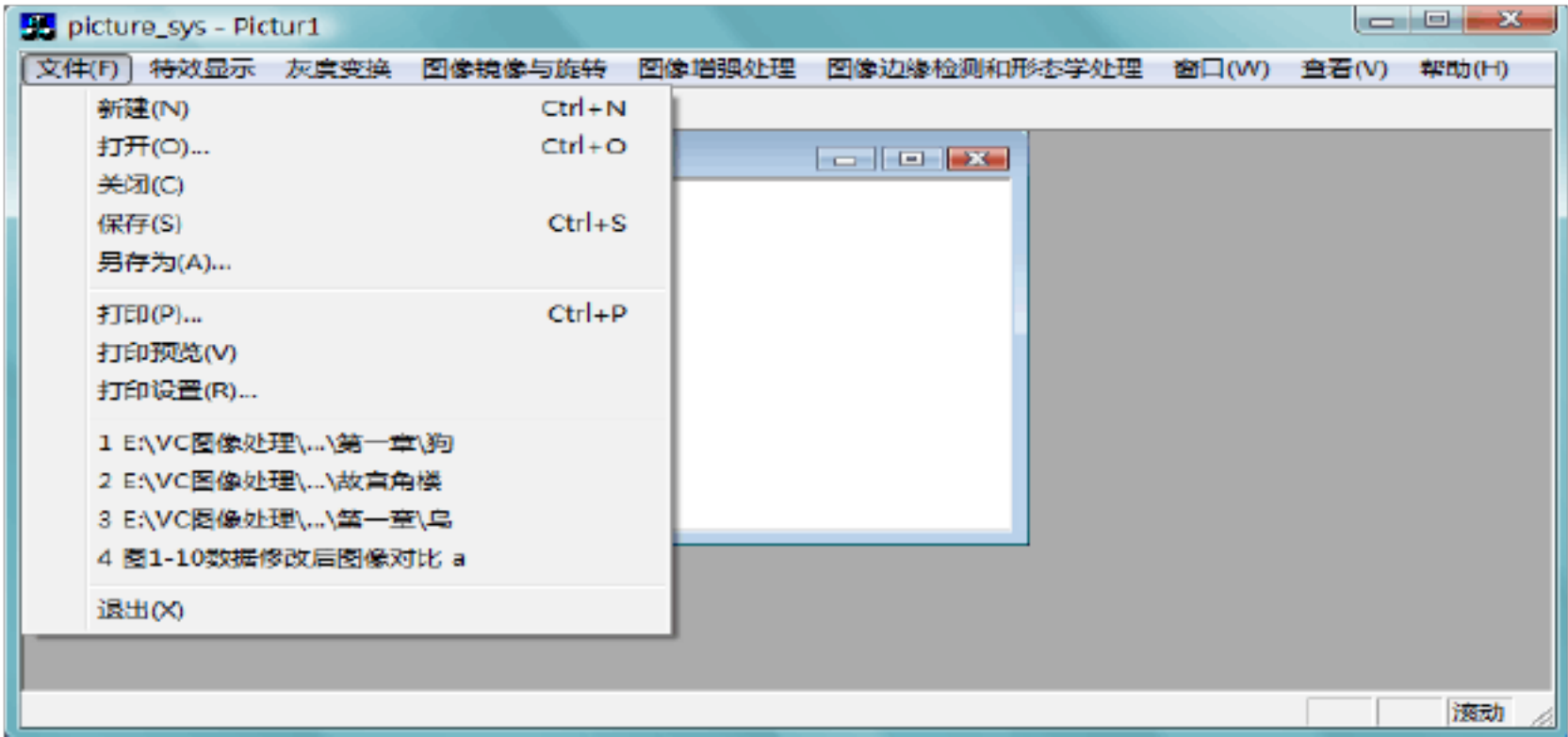
在文件操作模块,新建操作,关闭操作和图像打印操作可以直接调用系统提供的不需要重载的函数很容易就实现,其他操作都需要将文件下的.bmp 格式的图像数据读进计算机内存中生成 DIB, 存放在 m_lpDib(指向 DIB 的指针)所指向的缓冲区。

文件打开操作的伪代码描述如下所示： 文件的另存为操作伪代码如下：

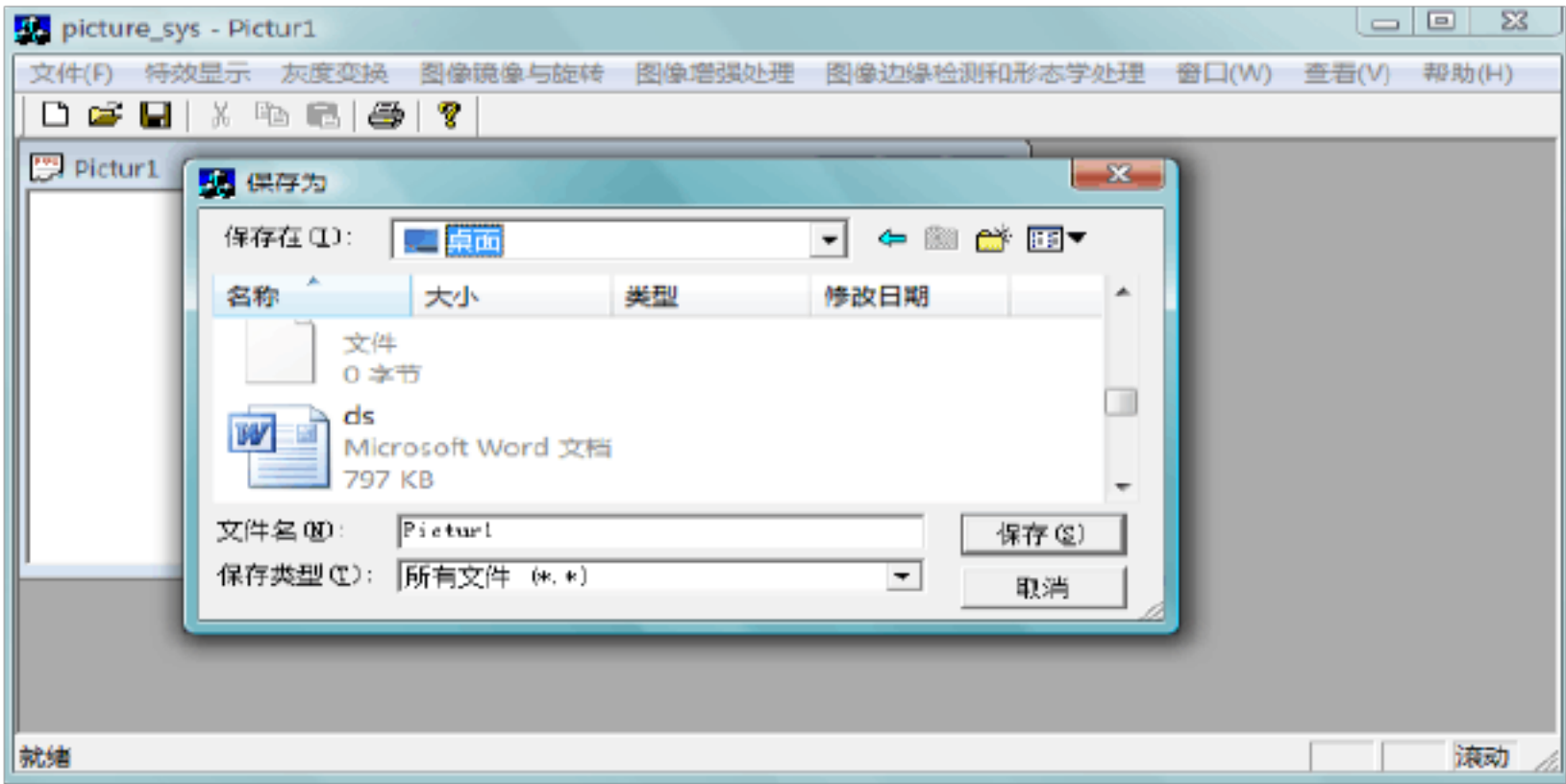
取得文件路径名称;
调用 ImgCenterDib 的 Read
函数读图形入内存中;
判断读操作是否成功,若
成功设置标记并返回 True,
否则返回 FALSE

取得文件路径名称;
调用 ImgCenterDib 的 write
函数读图形入内存中;
判断读操作是否成功,若
成功设置标记并返回 True, 否
则返回 FALSE

功能截图如下：
菜单栏：



另存为操作：



）特效显示模块

特效显示模块实现了图像在视图中的显示方式,包括自上而下扫描显示,从左至右的滑动显示和渐进显示。经过分析图像特效显示功能，得出图像处理的功能模块图如图 4. 3所示：

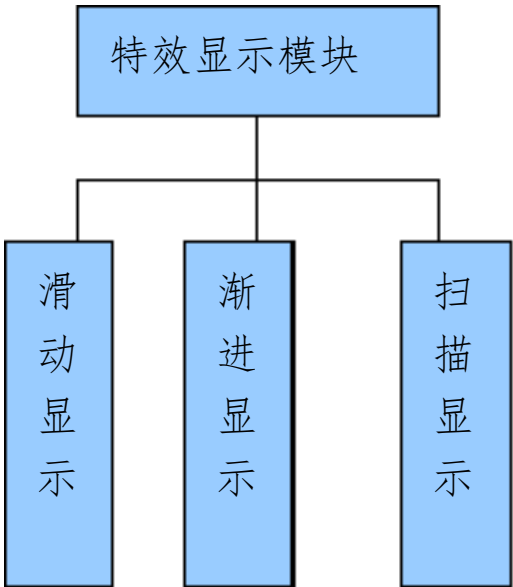


图 4. 3特效显示模块功能结构图

特效显示类 `SpecialEffectShow` 是在基类 `ImgCenterDib` 基础上派生出来的,特效显示类设计代码如下：

```
class SpecialEffectShow : public ImgCenterDib
{
public:
    SpecialEffectShow( CSize size, int nBitCount,
                      LPRGBQUAD lpColorTable, unsigned char *pImgData);
//带参数的构造函数

    void Scan(CDC*pDC); //扫描特效显示
    void Slide(CDC*pDC); //滑动特效显示
    void FadeIn(CDC * pDC); //渐进特效显示
    virtual ~SpecialEffectShow();
protected:
    SpecialEffectShow(); //防止从外部访问无参数的构造函数
```

表 1 主要的消息处理函数表

序号	ID 号及类名	消息处理函数	函数功能
1	ID_scan	Onscan	扫描显示
2	ID_slide	Onslide	滑动显示
3	ID_Fadein	OnFadein	渐进显示

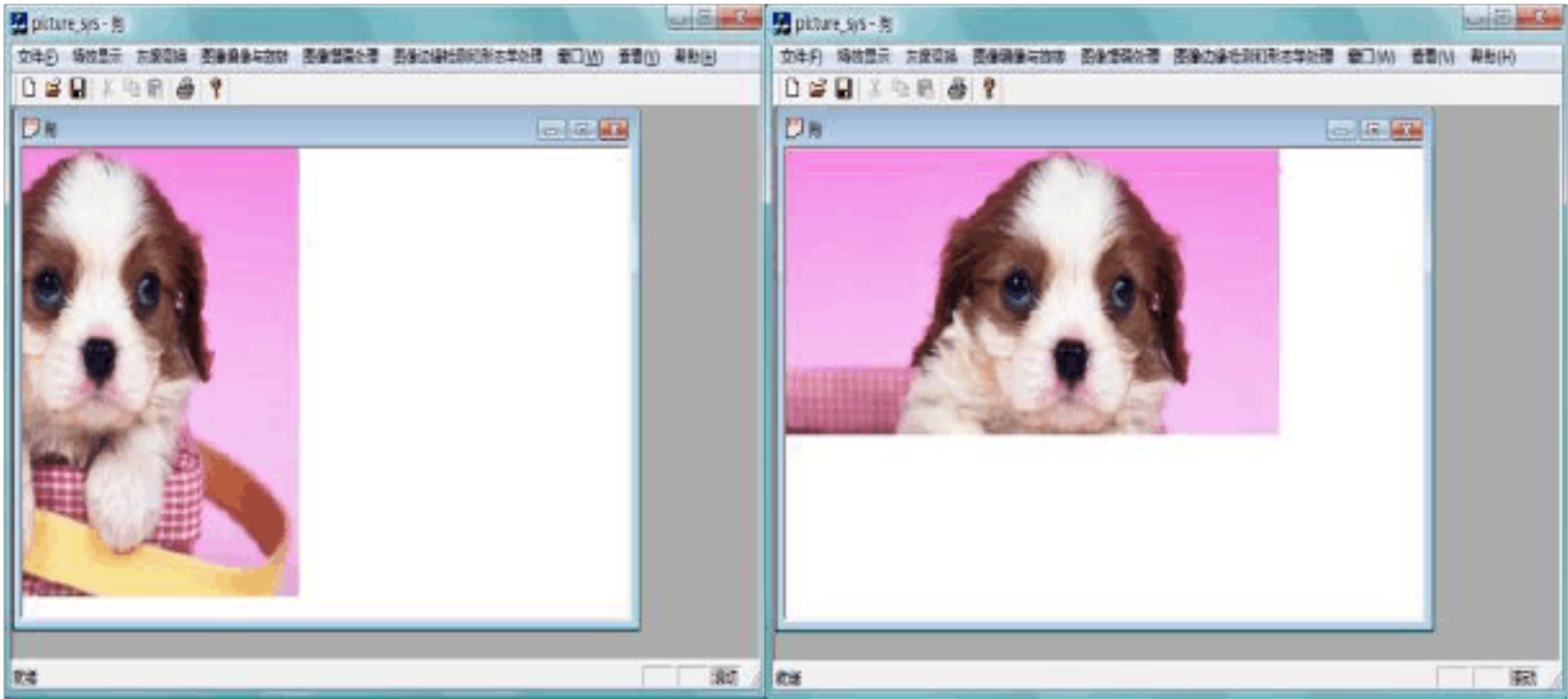
在 `picture_sysView.cpp` 重载消息处理函数,这三个消息处理函数代码基本相似,共同操作伪代码如下:

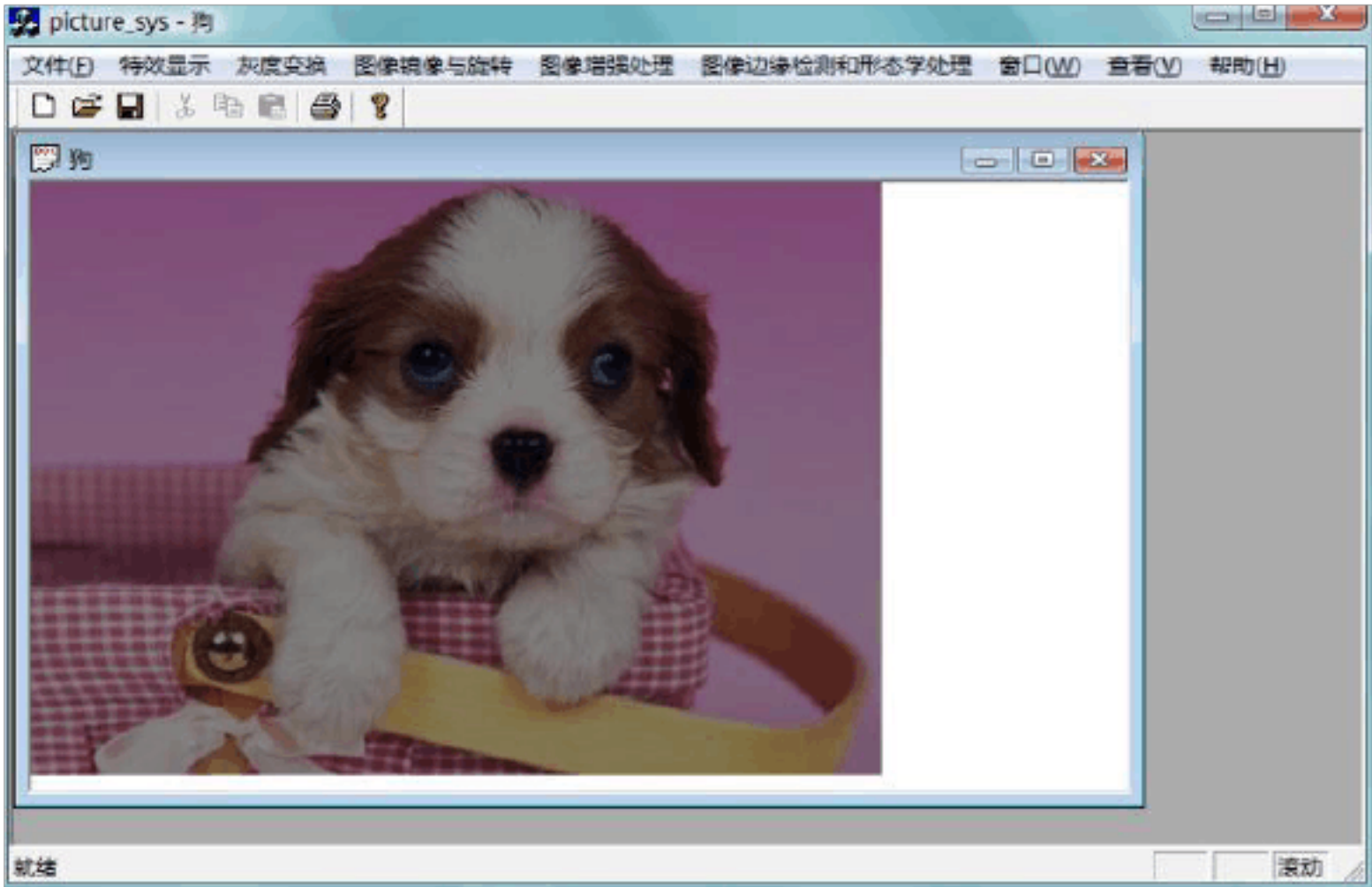
获取文档类中 `m_dib` 的指针,访问当前 DIB 数据;
定义 `SpecialEffectShow` 类的对象 `SpShow`,用当前 DIB 数据为其初始化;
获得设备环境变量;
传递设备环境指针,调用滑动/扫描/渐进显示程序;

功能截图如下:

扫描显示:

滑动显示:





() 图像灰度变换模块

图像灰度变换模块实现了彩色图像变灰度格式,灰度图像变彩色格式和二值化操作。经过分析图像灰度变换功能,得出图像处理的功能模块图如图4. 4所示:

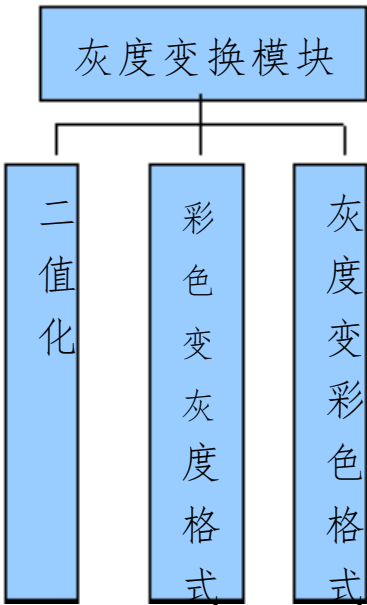


图 4. 4 灰度变换模块功能结构图

灰度变换类 `GrayTrans` 是在基类 `ImgCenterDib` 基础上派生出来的,灰度变换类设计代码如下:

```
class GrayTrans:public ImgCenterDib
{
public:
    int m_nBitCountOut; //输出图像每像素位数
    unsigned char * m_pImgDataOut; //输出图像位图数据指针
    LPRGBQUAD m_lpColorTableOut; //输出图像颜色表
private:
    int m_imgWidthOut; //图像的宽, 像素为单位
    int m_imgHeightOut; //图像的高
```

```
public:
    GrayTrans();//不带参数的构造函数
    GrayTrans(CSize size, int nBitCount, LPRGBQUAD lpColorTable,
    unsigned char *pImgData); //带参数的构造函数
    ~GrayTrans();
    CSize GetDimensions();//以像素为单位返回输出图像的宽和高
    void ColorToGray();//彩色变灰度格式
    void GrayToColor();//灰度变彩色格式
    void Binary(int threshold=128); //二值化
};
```

利用类向导，给相应的控件定义消息处理函数，如下表

表 2 主要的消息处理函数表

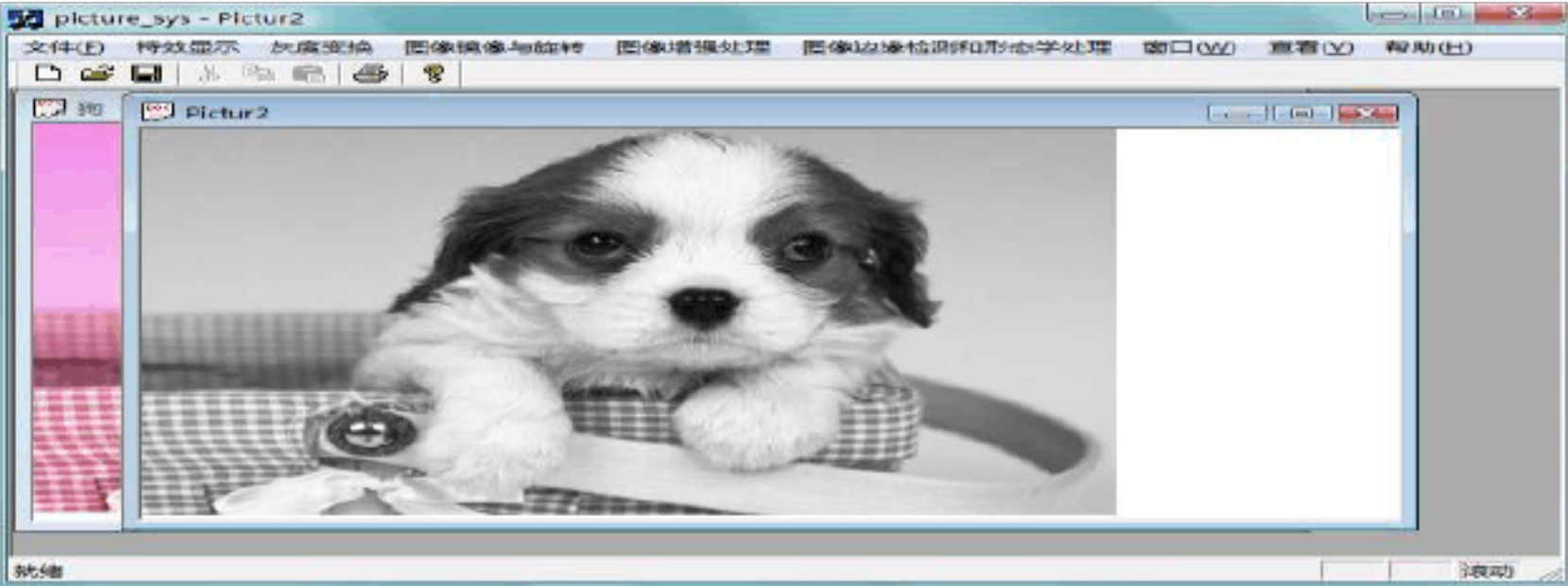
序号	ID 号及类名	消息处理函数	函数功能
1	ID_TOGRAY	OnTogray	彩色变灰度格式
2	ID_Graytocolor	OnGraytocolor	灰度变彩色格式
3	ID_BinTwo	OnBinTwo	二值化

在 picture_sysView.cpp 重载消息处理函数,这三个消息处理函数代码基本相似，共同操作伪代码如下：

获取文档类中 m_dib 的指针，访问当前 DIB 数据；
判断灰度图像还是彩色图像；
分别定义 GrayTrans 类的对象,调用相应的函数；
建立一个新视图，显示分割结果；

功能截图如下：

彩色变灰度格式：



灰度变彩色格式：

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：<https://d.book118.com/778135041044006121>