УДК 681.3

ОБРАБОТКА РАСТРОВЫХ ИЗОБРАЖЕНИЙ, ПРЕДСТАВЛЕННЫХ В ФОРМАТЕ DIB

А.В. Харченко
(представил д.т.н., проф. Г.П. Кулемин)

Предлагается методика преобразования структурированного файла изображения, представленного в аппаратно - независимом растровом формате DIB, в двумерный массив, позволяющий получить доступ к битовому образу непосредственно из программ, написанных на языках высокого уровня.

В процессе ультразвукового и радиолокационного зондирования объектов, а также сканирования изображений, получают графические изображения, которые сохраняются в памяти ПЭВМ в виде файлов на диске [1, 2]. Наиболее распространенным форматом представления и хранения графической информации является аппаратно - независимый растровый формат DIB (Device Independent Bitmap). Однако при работе с данным форматом пользователь не имеет прямого доступа к битовому образу.

В статье предлагается методика преобразования структурированного *.bmp файла в двумерный массив, позволяющий получить оперативный доступ к битовому образу непосредственно из программ, написанных на языках высокого уровня.

Подобные преобразования могут быть выполнены при помощи библиотечных подпрограмм в системе Windows. Например, в библиотеке языка Delphi содержится объект **Timage**, обеспечивающий вывод на экран изображения форматов BMP, WMF и ICO. Свойство **Pixels** позволяет получить доступ к каждому пикселю изображения на экране как к двумерному массиву. Однако такой способ работы с графикой может эффективно использоваться только на стадиях моделирования и проверки методик обработки изображений. Он требует достаточно больших ресурсов компьютера и плохо приспособлен для решения задач реального времени. При нехватке ресурсов существенно падает быстродействие выполнения программ.

Особенностью файлов формата DIB является то, что в них используется кодировка цветов с одной битовой плоскостью. Файл данного формата имеет четко определенную структуру. Он состоит из двух заголовков и битового образа. Битовый образ представляет собой массив, определяющий соответствие между цветами палитры и пикселями изображения.

Для преобразования *.bmp файла в массив данных, удобных для последующей обработки графической информации, необходимо:

1) идентифицировать формат файла и верифицировать его объем;

2) получить данные о смещении битового образа;

3) определить основные атрибуты изображения: размеры по горизонтали и по вертикали, количество цветов, разрешение;

4) установить цвета используемой палитры (за исключением полноцветных изображений);

5) считать битовый образ и разместить его в видеопамяти.

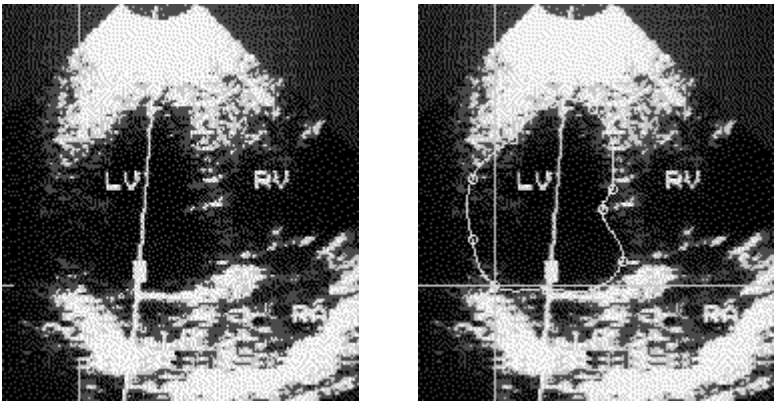
Для извлечения информации из заголовка следует точно указать его структуру, т.е. последовательность и размер всех его полей. Первый заголовок файла имеет структуру **BITMAPFILEHEADER**, представленную в табл. 1.

Таблица 1

Структура **BITMAPFILEHEADER**

№ п/п	Название поля	Размер, (бит)	Описание поля
1	bfType	16	тип файла (имеет значение = 424Dh)
2	bfSize	32	размер файла в байтах
3	bfReserved1	16	не используется
4	bfReserved2	16	не используется
5	bfOffbits	32	смещение данных bitmap от заголовка в байтах

В качестве примера рассмотрим заголовок файла, описывающего шестнадцатицветный рисунок (рис. 1а) размером 640 x 480 пикселей. Он имеет вид: 42 4D - 76 58 02 00 - 00 00 - 00 00 - 76 00 00 00.



а б

Рис. 1. Исходное (а) и обработанное (б) изображения

Содержание заголовка интерпретируется следующим образом:

42 4D – идентификатор, он соответствует символам «BM» (Bit Map);
76 58 02 00 – файл имеет размер 153718 байт;
00 00 - 00 00 – два зарезервированных поля;
76 00 00 00 – от начала файла до начала битового образа 118 байт.

Атрибуты графического изображения и коды палитры содержатся во втором заголовке. Он находится непосредственно за первым и имеет структуру **BITMAPINFO,** представленную в табл. 2, которая состоит из двух частей. Первая часть **BITMAPINFOHEADER** описывает размеры, разрешение и другие атрибуты битового образа. Вторая содержит массив данных, соответствующих структуре **RGBQUAD**, определяющий цветовую палитру.

Таблица 2

Структура **BITMAPINFO**

№ п/п	Название поля	Размер, (бит)	Описание поля
BITMAPINFOHEADER			
1	biSize	32	число байт, занимаемых структурой
2	biWidth	32	ширина битового образа в пикселях
3	biHeight	32	высота битового образа в пикселях
4	biPlanes	16	число битовых плоскостей устройства
5	biBitCount	16	число битов на пиксель
6	biCompression	32	тип сжатия
7	biSizeImage	32	размер картинки в байтах
8	biXPelsPerMeter	32	горизонт. разрешение устройства пикс/м
9	biYPelsPerMeter	32	вертик. разрешение устройства, пикс/м
10	biClrUsed	32	число используемых цветов
11	biClrImportant	32	число значимых цветов
RGBQUAD			
12	rgbRed	8	интенсивность красного цвета
13	rgbGreen	8	интенсивность зеленого цвета
14	rgbBlue	8	интенсивность синего цвета
15	rgbRserved	8	не используется

Данные этого заголовка используются в качестве параметров при инициализации графического режима. Графический режим выбирается в соответствии с возможностями используемой операционной системы и зависит от количества цветов и размера изображения. Для настройки палитры используются элементы массива структур RGBQUAD, их значения заносятся в соответствующие регистры графического адаптера. Фрагмент программы настройки палитры при использовании компилятора Borland Turbo C++ представлен на рис. 2. Для вывода на экран изо-

бражения достаточно считать битовый образ и разместить его в видеопамяти. Для большинства систем это старшие адреса оперативной памяти, к которой имеется непосредственный доступ. Фрагмент программы считывания битового образа показан на рис. 3.

```
{int CRed, CGreen, CBlue;
// Объявление переменной палитры
struct palettetype pal;
// Инициализация переменной палитры
getpalette(&pal);
// Настройка цветов
for(MainColor=0; MainColor<pal.size; MainColor++)
    {CRed =  (BI.bmiColors[MainColor].  rgbRed   / 4);
    CGreen = (BI.bmiColors[MainColor]. rgbGreen / 4);
    CBlue =   (BI.bmiColors[MainColor]. rgbBlue    / 4);
    setrgbpalette(pal.colors[MainColor], CRed, CGreen, CBlue);};}
```

Рис. 2. Фрагмент программы настройки палитры

```
// Смещение указателя на начало битового образа
fseek(InStream, bfOffbits, SEEK_SET);
// Считывание битового образа в заданную область памяти
fread(&V, biSizeImage, 1, InStream)
```

Рис. 3. Фрагмент программы считывания битового образа

При последующей обработке графической информации (фильтрации, выделении контуров, сжатии и т.д.) все изменения будут отображаться на экране компьютера в реальном времени. В качестве примера на рис. 1 б представлены результаты оконтуривания изображения левого желудочка сердца, полученного в процессе ультразвукового сканирования. Оконтуривание было выполнено с использованием специальной процедуры интерполяции и изложенной выше методики преобразования, что обеспечило большую оперативность измерения параметров левого желудочка и формирования признаков кардиологического заболевания.

ЛИТЕРАТУРА

1. *Кенцл Т. Форматы файлов Internet. – СПб: Питер, 1997. – 320 с.*
2. *Бондарев В.Н., Трёстер Г., Чернега В.С. Цифровая обработка сигналов. – Севастополь: Изд-во СевГТУ, 1999. – 398 с.*

Поступила 03.01.2002

***ХАРЧЕНКО Алексей Викторович**, инженер отделения клинико-инструментальной и ультразвуковой диагностики Института общей и неотложной хирургии АМН Украины. В 2001 году окончил Национальный аэрокосмический университет «ХАИ». Область научных интересов – программно-аппаратные методы выделения признаков кардиологических заболеваний. E-mail: alxmail2000@rambler.ru .*