原 BMP 转 YUV （BMP2YUV）

2013年10月29日 13:41:02　阅读数：18518

本文介绍BMP　转　YUV。其实这是以前"数据压缩"实验课上的内容，前几天有人问我相关的问题，突然发现自己有一段时间没有接触BMP也有些生疏了，因此翻出资料总结一下。

BMP文件格式解析

位图文件(Bitmap-File，BMP)格式是Windows采用的图像文件存储格式，在Windows环境下运行的所有图像处理软件都支持这种格式。BMP位图文件默认的文件扩展名是bmp或者dib。BMP文件大体上分为四个部分：

位图文件头 主要包括：

```cpp
typedef struct tagBITMAPFILEHEADER {
WORD bfType; /* 说明文件的类型 */
DWORD bfSize; /* 说明文件的大小，用字节为单位 */
WORD bfReserved1; /* 保留，设置为0 */
WORD bfReserved2; /* 保留，设置为0 */
DWORD bfOffBits; /* 说明从BITMAPFILEHEADER结构开始到实际的图像数据之间的字节偏移量 */
} BITMAPFILEHEADER;
```

位图信息头 主要包括：

```cpp
typedef struct tagBITMAPINFOHEADER {
DWORD biSize; /* 说明结构体所需字节数 */
LONG biWidth; /* 以像素为单位说明图像的宽度 */
LONG biHeight; /* 以像素为单位说明图像的高速 */
WORD biPlanes; /* 说明位面数，必须为1 */
WORD biBitCount; /* 说明位数/像素，1、2、4、8、24 */
DWORD biCompression; /* 说明图像是否压缩及压缩类型BI_RGB，BI_RLE8，BI_RLE4，BI_BITFIELDS */
DWORD biSizeImage; /* 以字节为单位说明图像大小，必须是4的整数倍*/
LONG biXPelsPerMeter; /*目标设备的水平分辨率，像素/米 */
LONG biYPelsPerMeter; /*目标设备的垂直分辨率，像素/米 */
DWORD biClrUsed; /* 说明图像实际用到的颜色数，如果为0，则颜色数为2的biBitCount次方 */
DWORD biClrImportant; /*说明对图像显示有重要影响的颜色索引的数目，如果是0，表示都重要。*/
} BITMAPINFOHEADER;
```

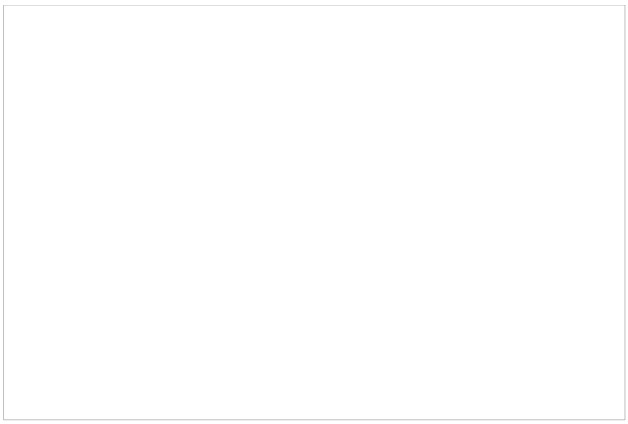
调色板 实际上是一个数组，它所包含的元素与位图所具有的颜色数相同，决定于biClrUsed和biBitCount字段。数组中每个元素的类型是一个RGBQUAD结构。真彩色无调色板部分。

```cpp
typedef struct tagRGBQUAD {
BYTE rgbBlue; /*指定蓝色分量*/
BYTE rgbGreen; /*指定绿色分量*/
BYTE rgbRed; /*指定红色分量*/
BYTE rgbReserved; /*保留，指定为0*/
} RGBQUAD;
```

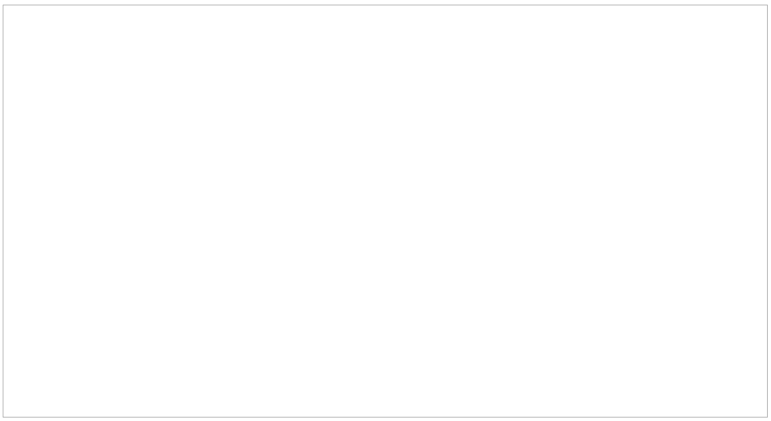
紧跟在调色板之后的是 **图像数据字节阵列** 。对于用到调色板的位图，图像数据就是该像素颜色在调色板中的索引值（逻辑色）。对于真彩色图，图像数据就是实际的R、G、B值。图像的每一扫描行由表示图像像素的连续的字节组成，每一行的字节数取决于图像的颜色数目和用像素表示的图像宽度。规定每一扫描行的字节数必需是4的整倍数，也就是DWORD对齐的。扫描行是由底向上存储的，这就是说，阵列中的第一个字节表示位图左下角的像素，而最后一个字节表示位图右上角的像素。

读取 BMP 文件，提取 RGB 数据的流程

流程如下图所示：



在这里需要注意的的是，根据BMP每像素所占的比特数不同（8，16，32 bpp），分为不同的处理方法，如下图所示。



下面看看16bpp的BMP文件操作：

R,G,B在16bit中所占的位数如下图所示

```cpp
for (Loop = 0;Loop < height * width;Loop +=2)
{
       *rgbDataOut = (Data[Loop]&0x1F)<<3;
       *(rgbDataOut + 1) = ((Data[Loop]&0xE0)>>2) +           ((Data[Loop+1]&0x03)<<6);
       *(rgbDataOut + 2) = (Data[Loop+1]&0x7C)<<1;
       rgbDataOut +=3;
}
```

1-8bpp的BMP文件操作：

```cpp
int shiftCnt = 1;
while (mask)
{
unsigned char index = mask == 0xFF ? Data[Loop] : ((Data[Loop] & mask)>>(8 - shiftCnt * info_h.biBitCount));
* rgbDataOut = pRGB[index].rgbBlue;
* (rgbDataOut+1) = pRGB[index].rgbGreen;
* (rgbDataOut+2) = pRGB[index].rgbRed;
if(info_h.biBitCount == 8)  mask = 0;
Else    mask >>= info_h.biBitCount;
rgbDataOut+=3;
shiftCnt ++;
}
```

BMP转换为YUV

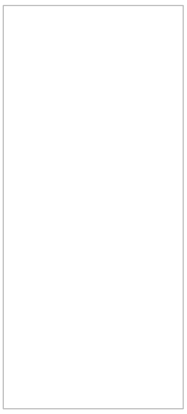
RGB到色差信号的转换如下所示：
Y＝0.2990R+0.5870G+0.1140B
R-Y＝0.7010R-0.5870G-0.1140B
B-Y＝-0.2990R-0.5870G+0.8860B

为了使色差信号的动态范围控制在0.5之间，需要进行归一化，对色差信号引入压缩系数。归一化后的色差信号为：

U＝-0.1684R-0.3316G+0.5B
V＝0.5R-0.4187G-0.0813B

YUV文件的格式

转换后的YUV数据需要存成YUV文件（在这里是YUV420P格式）。YUV文件的格式很简单，先连续存Y，然后U，然后V，如图所示。



BMP 转 YUV程序下载地址： http://download.csdn.net/detail/leixiaohua1020/6469807

文章标签： bmp yuv 转换

个人分类： 纯编程 视频编码