ISSN: 2661-3719(Print); 2661-3727(Online)

基于16色位图的反走样汉字绘制方法研究

顾卫卫
苏州长风航空电子有限公司　江苏苏州　215151

摘　要：针对单片机架构的图形产生模块中，采用单色点阵字符进行汉字显示时锯齿状明显问题，文本提出一种基于BMP图片生成带灰度等级的汉字库以提升汉字显示效果的方法。通过图片处理工具（Photoshop等）将汉字库导入到图片文件中，并保存为16色BMP图像文件，再将其转换为像素数据，使用目标平台提供的像素点颜色绘制功能实现汉字的显示且具有16级灰度等级的反走样效果。
关键词：位图；反走样；汉字；BMP

Research on drawing method of anti-aliasing Chinese characters based on 16-color bitmap

Wei Gu
Suzhou Changfeng Avionics Co. , Ltd. Suzhou 215151

Abstract: In the graphics generation module based on the single-chip microcontroller architecture, when using monochrome dot matrix characters for Chinese character display, the problem of visible aliasing is prominent. This paper proposes a method to improve the Chinese character display effect by generating a Chinese character library with grayscale levels based on BMP images. The Chinese character library is imported into the image file using image processing tools (such as Photoshop) and saved as a 16-color BMP image file. Then, it is converted into pixel data, and the drawing function of the target platform is used to display the Chinese characters with 16-level grayscale anti-aliasing effect.
Keywords: bitmap; Anti-out-of-shape; Chinese characters; BMP

随着CPU+GPU架构和APU架构的图形产生模块的广泛应用，飞机座舱显控系统所呈现的显示视觉效果得到极大地改善。相比之下，早期所采用的单片机架构图形产生模块（比较典型的ADSP系列的图形产生模块和STM32系列单片机图形产生模块）所生成的字符特别是汉字显示效果不佳，字符边缘的锯齿明显。本文研究通过BMP图像文件生成汉字库，再通过工具将其转换为像素数据，结合调色板功能，在单片机平台中实现汉字的16级灰度反走样显示，有效提升单片机架构中汉字的显示效果。

一、BMP图像文件

BMP（全称Bitmap）是Windows操作系统中的标准图像文件格式，它采用位映射存储格式，除了图像深度可选择以外，不采用其他任何压缩。BMP文件的图像深度可选1位（单色位图）、4位（16色位图）、8位（256色位图）及24位（真彩色位图）[1]。

BMP文件的格式组成包括位图头文件数据结构、位图信息数据结构、调色板及位图数据。位图信息数据结构包含BMP图像的宽、高及像素的颜色位数等信息；调色板中即是实际的RGB数值，除24位真彩色图像不需要调色板（像素数据中直接就是RGB数值），其余类型图像中像素数据引用的是调色板中颜色索引即位图数据实际就是调色板中颜色索引值。

二、汉字库设计

1.实现思路

一级汉字库共包含3755个汉字，通过汉字区位码可实现汉字的快速定位。本文将一级汉字库的所有汉字通过Photoshop导入到图像文件中（黑体16号，每个汉字占

作者简介：顾卫卫（1986-），男，江苏淮安人，大学本科，工程师，主要研究方向为飞机座舱显示与控制系统设计。

用的像素大小为22*21，即每个汉字宽度为22像素，高度为21像素），可以全部导入到一副图像或者多幅图像中，因图像数据转换工具Image2Lcd最大支持1024*768像素大小的图片，本方案选择将一级汉字库分为3幅图像存储，每幅图像大小为1024*588像素，即每行存储46个汉字、共有28行。

为便于像素数据处理，图像文件的背景色选择白色，汉字颜色选择黑色。

图像文件制作完成后，将其保存为16色位图（这样可使用16级灰度实现反走样效果，方便对应到16级灰度调色板中）。然后使用Image2Lcd工具将16色位图转换为像素数据，选择扫描模式为水平扫描（从左至右，从上到下，第一个颜色数据为图像文件左上角第一像素点的颜色值），生成符合标准C语言规范的无符号整型数组数据，见图1。

```
const unsigned char gImage_chs1[301056] = { /* 0X10,0X04,0X04,0X00,0X02,0X4C, */
0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,
0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X04,0X20,0X00,0X00,0X0A,0X60,0X00,
0X00,0X00,0X00,0X00,0X04,0X30,0X00,0X00,0X09,0X70,0X00,0X00,0X00,0X00,0X00,0X00,
0X00,0X00,0X04,0X72,0X00,0X47,0X20,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X0A,0X80,
0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X4B,0X60,0X00,0X00,0X00,0X00,0X00,0X00,
0X39,0X40,0X00,0X00,0X00,0X4F,0X70,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0XAD,
0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X04,0XF7,0X00,0X00,0X4F,0X40,0X00,0X00,0X00,
0X00,0X40,0X00,0X00,0X00,0X00,0X00,0X03,0X85,0X00,0X00,0X00,0X00,0X02,0X74,0X00,
0X00,0X67,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,
0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X04,0X76,0X00,0X00,0X00,0X00,0X00,0X00,
0X06,0X40,0X00,0X07,0X72,0X00,0X00,0X00,0X65,0X00,0X64,0X00,0X00,0X3C,0X40,0X00,
0X00,0X00,0X00,0X00,0X88,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,
0X00,0X6E,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X82,0X00,0X00,0X08,0X70,0X00,
0X00,0X00,0X00,0X00,0X07,0X60,0X00,0X00,0X00,0X00,0XC7,0X00,0X00,0X00,0X00,0X00,
0X00,0X00,0X00,0X05,0XC0,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X06,0XB3,0X00,
0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X5B,0X00,0X00,0X00,0X00,0X00,
0X00,0X00,0X00,0X5D,0X20,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,
0X75,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,0X00,
```

图1　图像数据

图3中的像素数据，数值为0表示像素点的颜色与背景色一样，使用打点功能绘制像素点时可跳过不用绘制。

2.数据结构设计

根据汉字库制作的方式及图像数据的生成特点，设计字符配置数据结构，便于后续用此方法制作其他字体字号的数据文件，详细数据结构定义如下：

```
typedef struct
{
INT32 startPixel;          //字符在图片中起始像素
UINT32 symWidth;      //每个字符宽度
UINT32 symHeight;     //每个字符高度
UINT32 symCountPerRow;  //图片中每行包含的字符数量
UINT32 pixelsPerRow;   //图片中每行包含的像素
UINT32 bitsPerPixel;   //每个像素占用的数据位
}FONT_CFG;
```

bitsPerPixel决定了每个像素占用的数据位，比如本方案中使用16色位图，即每个像素使用4位数据表示，这样每个字节就包含2个像素点的颜色数据，各占用4位，高4位存储左边像素点数据，低4位存储右边像素点数据。

三、实现及验证

1.软件实现

一级汉字库中的所有汉字，可根据区位码快速进行定位[2]，字库中第一个汉字“啊”的区位码为（94*(0xB0 – 16) + (0xA1 – 1)），需要绘制的汉字，假设其区码为qCode、位码为wCode，则该汉字的偏移为：

（94*(qCode– 16) + (wCode– 1)）–（94*(0xB0 – 16) + (0xA1 – 1)）

计算出待绘制的汉字偏移后，即可使用该偏移定位汉字在图像中行数，然后在图像数据中进行像素点数据读取，根据上述汉字库设计原理，可知每个16号黑体汉字占用22*21像素大小（每行22个像素，高度21像素）。主要计算公式如下：

图像中每行占用的字节数量：

bytesPerRow = txtCfg.pixelsPerRow*txtCfg.bitsPerPixel/8；

txtCfg.pixelsPerRow – 每行图像的像素点数；

txtCfg.bitsPerPixel – 每个像素点占用的数据位数。

待绘制字符在图片中行数：

row = symOffset/txtCfg.symCountPerRow；

symOffset – 待绘制字符在汉字库中的偏移；

txtCfg.symCountPerRow – 图像中每行包含的汉字数量。

待绘制字符像素数据的在数据中的起始位置：

k = row*bytesPerRow*txtCfg.symHeight + (txtCfg.startPixel + (symOffset%txtCfg.symCountPerRow)*txtCfg.symWidth)/2；

row – 待绘制字符在图片中行数；

bytesPerRow –图像中每行占用的字节数量；

txtCfg.symHeight – 每个汉字的高度；

txtCfg.startPixel – 每行第一个汉字在图像中的起始X坐标（单位：像素）；

symOffset –待绘制字符在汉字库中的偏移；

txtCfg.symCountPerRow – 图像中每行包含的汉字数量；

txtCfg.symWidth – 每个汉字的宽度。

计算出汉字的每个像素点颜色数值后，调用目标平台提供的打点函数绘制每个像素点即可完成汉字的显示。

为提高功能的可移植性，将打点函数设计为回调方

ISSN: 2661-3719(Print); 2661-3727(Online)

式，即字符绘制该功能本身并不实现打点功能，使用者根据所用的目标平台自行设计打点函数接口，以目前我厂使用ADSP架构的显示器为例，因不同的液晶屏型号差异，其扫描方式也不同，有的从左上角开始扫描（从左至右、从上向下），有的从左下角开始扫描（从左至右、从下向上），因此回调函数的方式可极大提升模块的移植性，实现方式如下：

typedef void (*p_draw_point)(INT32, INT32, INT32);// 函数指针

打点功能函数需包含3个输入参数，分别为起始X坐标，起始Y坐标和像素点颜色索引。然后再提供设置打点函数实体的接口，详细如下：

void SetDrawPointCallback(void *drawPoint);

使用者只需根据目标平台实现相应的打点函数（必须有3个输入参数，X方向从左向右为正向，Y方向从下到上为正向），然后再调用回调函数设置接口将打点功能函数传入字符绘制模块即可。

字符串绘制功能接口定义如下：

INT32 DrawBitmapText(INT32 xStart, INT32 yStart, INT8 *txt, UINT32 color, INT32 antialias);

xStart – 起始X坐标；

yStart – 起始Y坐标；

txt – 待绘制字符串，支持中文、英文、数字、常用符号的混合输入

color – 绘制的颜色；

antialias – 反走样标记，0启用反走样效果，1关闭反走样效果。

2.效果验证

在ADSP-TS101+FPGA架构设计的硬件平台中进行效果验证，效果图见图5，其中左下角绿色“汉字反走样效果测试”，第一行是使用本方法实现的具有反走样功能的显示效果，第二行是使用本方法实现的未开启反走样功能的显示效果，最后一行为常用的基于字模工具产生的单色点阵实现的显示效果，从图中可看出，使用本方法实现的反走样汉字绘制，可明显改善字符显示效果。

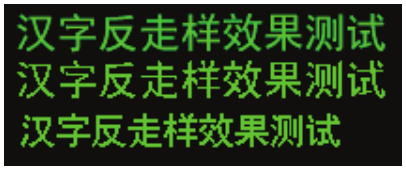

图2　显示效果

四、结束语

本文对单片机平台中反走样字符的绘制方式进行了探索研究，并着重介绍了字库的设计及绘制过程实现。经过测试验证，本文的方法的有效且可行的，可明显提升单片机平台中字符显示效果，但因为每个像素点颜色用4位数据表示，以实现16级灰度，使得整个字库相比常用的单色点阵字库（每个像素点只占1位，用0或1表示点亮或点暗）需占用更多的存储空间，理论上每个汉字的占用空间是原来的4倍，汉字库的数据量较大。而通常单片机平台的存储空间较为有限，因此在实际使用过程中，可根据具体需求对字库进行相应的裁剪。

参考文献：

[1]宋亮，陈瑜轩.浅谈图像处理与BMP图像文件格式[J].电子设计工程.1674-6236(2014)07-0188-03

[2]魏小龙，戴祥，施亿平.液晶汉字显示与汉字库实验设计[J].实验室研究与探索.1006-7167(20 03)06-0077-03