# UCDOS矢量字库结构分析

马艳华* 王前虹

**摘要** 本文详细分析了UCDOS矢量字库的结构及其矢量数据的组织方法，为直接使用UCDOS矢量汉字提供了依据。

**关键词** 矢量汉字 UCDOS 矢量字库

# UCDOS Vector Font Library Structure Analysis

Ma Yanhua Wang Qianhong

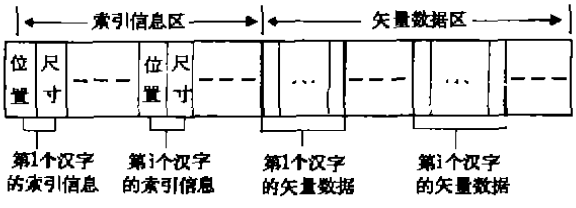
**Abstract** This paper discusses in detail the structure of UCDOS vector font library and the composing method of vector data and provids the basis for utilizing UCDOS vectorized chinese font.

**Keywords** Vectorized chinese font UCDOS Vector font library

在许多实际应用中，需要将汉字放大显示或输出，常用的一种方法就是从字库中取字模数据，然后通过特定的算法将汉字放大。目前的计算机系统中，主要使用点阵字库和矢量字库。因为与点阵汉字相比，矢量汉字具有放大失真小、美观漂亮、放缩旋转容易等特点，所以人们多希望使用矢量汉字。但要正确的输出汉字，必需清楚的了解矢量字库的结构。本文详细分析了目前常用汉字平台UCDOS矢量字库的组织结构，为在实际应用中读取汉字的矢量数据、使用矢量汉字提供了依据。

## 1 矢量字库结构

UCDOS矢量字库由索引信息区和矢量数据区两部分组成，如下图所示。




### 1.1 索引信息

根据汉字的内码可计算得到其索引信息在矢量字库中的位置，计算公式如下：

索引位置=((区码-16)*94+位码-1)*6

索引信息指明了某字的矢量数据在矢量字库中的位置和尺寸(以字节为单位)，其中位置是相对于字库开始处以字节为单位的位移量。

### 1.2 矢量数据

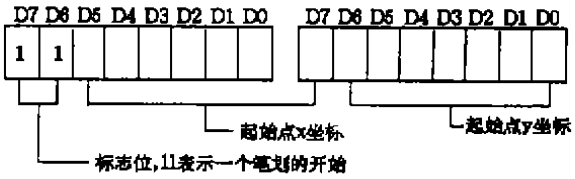
矢量数据区中的数据有绘制某汉字的所有矢量信息，这些信息又以笔划为一个数据组，每组占有若干个字节。这些字节中的首字节指明了该笔的起点，后续一个或多个字节指明了相对前一点的增量及增量的方向，亦即移动笔的距离以及笔的移动方向。

## 2 矢量数据的组织方法

矢量数据由X坐标的增量dx、y坐标的增量dy以及增量的方向组成。其中，增量方向通常用一个二进制位表示，且规定0表示正向，1表示反向。因为笔划的不同，其dx、dy的值也不同，因此占用字节数及数据位数均不一样。在UCDOS矢量字库中，为了节省空间没有对不同情况下的dx、dy规定统一的字节数，而是将矢量数据首字节的最高两位设定为标志位，标志位的四种状态表示四种不同情况，每种情况占用二个或若干个字节不等。具体情况分析如下：

① D7=1 D6=1

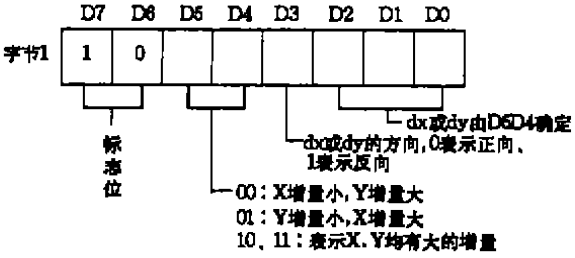
* 山东建材学院 250022 济南



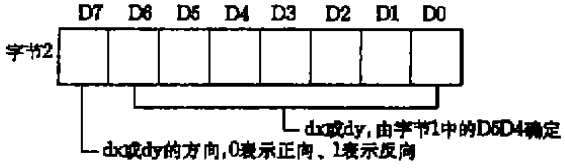

如果首字节最高两位均为1,则表示一个笔划的开始,该字节后六位与下一个字节的最高位表示起始点的X坐标值,下一个字节的剩余七位表示起始点的Y坐标值。

② D7=1 D6=0

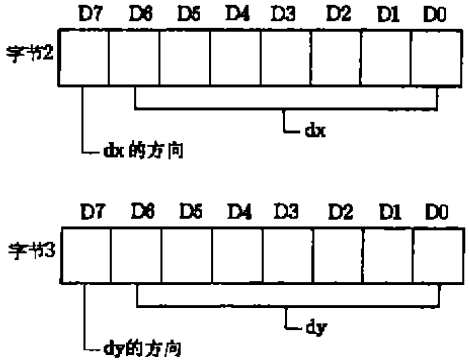
首字节最高位为1,次高位为0,此时根据D5、D4的取值确定X、Y的增量。




当D5D4等于00或01时,X、Y的增量值必定一个较大一个较小。在这两种情况下,字节1的后三位为小增量的增量值,D3为增量方向,而下一字节的后七位表示大增量的增量值,最高位(D7)为相应增量方向。

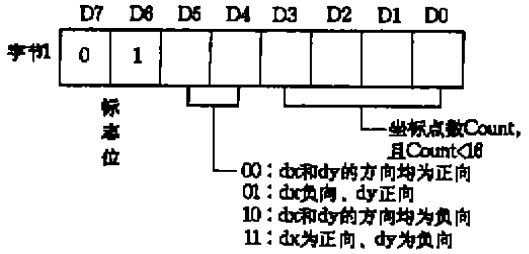



当D5D4等于10或11时,X、Y的增量值都较大。故在这两种情况下,紧接下二个字节为X、Y的增量值和增量方向。

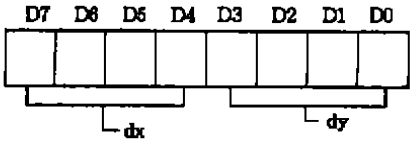



③ D7=0 D6=1

首字节最高位为0,次高位为1,表示该笔具有连续若干增量,且增量值较大。此时该字节的D5、D4表示增量的方向,当前字节的后四位表示增量方向上的坐标点数,我们记为Count。

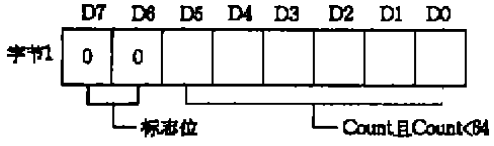



随后的Count个字节为增量值,其中高四位为dx、低四位为dy

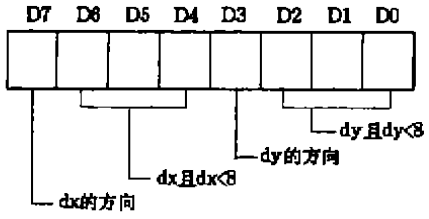



④ D7=0 D6=0

首字节最高两位均为0,则后六位为矢量点的个数,我们也记为Count。此时表示该笔有连续若干个增量,且相对于前一种情况(D7=0 D6=1),其每个增量值较小,但Count的值较大。




随后的Count个字节表示增量值及增量的方向,其中高四位为X坐标的增量及增量的方向,低四位为Y坐标的增量及增量方向。




## 3 汉字的显示、放缩与旋转

### 3.1 基本数据结构及汉字的显示

根据上述分析可知，当汉字矢量化以后，其每一笔划视情况可分为一个或若干个矢量段，每一段的长度用一个相对于段始点的增量及增量的方向表示，为了方便、有效地进行汉字的放缩、旋转及其它操作，我们定义了两个数组VectArray和PointArray，前者用于保存带方向的增量值，后者用于保存经过计算后的矢量段的起、止点坐标。两个数组C语言形式的定义如下：

```
typedef struct
    {
        char isfirst;    //若为1，则表示笔划的开始，否则表示
                           笔划的矢量段。
        int dx,dy;    //带方向的增量值，若isfirst=1则表示一
                        笔划的始点坐标(x,y)。
    } VECTOR;
VECTOR VectArray(SIZE);
typedef struct
    {
        char start;          //笔划开始标志
        unsigned int x,y;  //矢量段起止坐标
    }  POINT;
POINT PointArray(SIZE);
```

其中SIZE为数组尺寸，应保证能放下一个汉字的所有矢量数据。

汉字的显示算法基于数组PointArray，其基本方法就是在各点之间画线。设PointArray中实际点数为PointNum，则汉字显示的C语言函数如下：

```
Void DispHZ(int Xposition,int Yposition)
{
    int i,x,y;
    for(i=0;i<PointNum;i++)
    {   x=Xposition+PointArray(i).x;
        y=Yposition+PointArray(i).y;
        if(PointArray(i).start)
            Moveto(x,y);
        else
            lineto(x,y);
    }
}
```

### 3.2 汉字的放缩

如果用$(x_0,y_0)$表示一笔划中某矢量段的始点，$(x_1,y_1)$表示终点，dx,dy为带有方向的增量(正值表示正向增，负值表示反向增)，则有：

$$x_1=x_0+dx$$

$$y_1=y_0+dy$$

因此，只要将$(x_0,y_0)$和$(x_1,y_1)$之间的距离dx和dy乘以一个放大(或缩小)系数，就能实现矢量汉字的放缩。如果这个系数大于1，则汉字被放大，如果小于1，则缩小。

放缩算法的基本思想是将存于VectArray中的每个矢量段的增量乘以放缩系数Factor，然后将结果转换为坐标点存入PointArray中，实现放缩算法的C语言函数如下：

```
Void VectToPoint(int Factor)
{
    int i;
    for(i=0;i<PointNum;i++)
    {
        if(VectArray(i).isFirst)
        {
            PointArray(i).start=1;
            PointArray(i).x=VectArray(i).dx;
            PointArray(i).y=VectArray(i).dy;
        }
        else
        {
            PointArray(i).start=0;
            PointArray(i).x=PointArray(i-1).x+VectArray(i).
            dx*Factor;
            PointArray(i).y=PointArray(i-1).y+VectArray(i).
            dy*Factor;
        }
}
```

如果在该函数之后调用DispHZ就将显示出放缩以后的汉字。

### 3.3 汉字的旋转

任何点(x,y)可以通过它离原点的半径r和它离x轴的夹角$\phi$表示：

$$x=r\times\cos\phi$$

$$y=r\times\sin\phi$$

若(x,y)以逆时针方向旋转一角度$\theta$，则转换后的(x',y')表示如下：

$$x'=r\times\cos(\phi+\theta)$$

$$y'=r\times\sin(\phi+\theta)$$

根据三角函数变换，可以得到

$$x'=x\times\cos\theta-y\times\sin\theta$$

$$y'=x\times\sin\theta+y\times\cos\theta$$

上式表示将点(x,y)旋转一角度$\theta$后得到的点坐标(x',y')。根据上式将汉字的所有矢量点变换一遍，就会得到旋转后的汉字。假设在此之前已将VectArray中的矢量数据转换为PointArray中的坐标点。其C语言函数如下：

```
Void RotateHZ(int  θ ) //θ —用弧度值表示的旋转角度
{
    int i,x,y;
    for(i=0;i<PointNum;i++)
    {
            x=PointArray(i).x;
            y=PointArray(i).y;
            PointArray(i).x=x*cos θ -y*sin θ ;
            PointArray(i).y=y*cos θ +x*sin θ ;
    }
}
```

如果需要按顺时针旋转，则应将旋转角度$\theta$写为$-\theta$代入转换公式中，得到下式：

(下转第25页)

是否被击穿，判定电流至少应设得比$I_2$大。所以，判定电流应根据试验产品电路的实际情况来确定，原则就是能正确地判定绝缘是否被击穿了。一般标准中规定，试验电压既可为交流电压，也可为直流电压，两者是等效的。对于试验部位含有电容的可施加直流电压，其大小为指定交流电压的峰值即指定交流电压的1.414倍。

另外，因为耐压试验有时可能会对试验部位的绝缘造成损伤，所以有的标准对进行耐压试验的次数和持续时间做出了规定。GB4793-84中规定，试验电压超过2kV时，产品只允许进行两次试验(即重复一次)，每次试验均在100%的试验电压下进行，若再进行重复试验，则应在80%的试验电压下进行。对于试验持续时间而言，一般标准规定为1min，但也有例外。GB4793-84中规定，定型试验时，在规定试验电压下持续1min，而交收试验时，在规定的试验电压下持续2s。GB4943-90中规定，对生产试验，耐压试验的持续时间可以减小到1s。这里值得注意的是：缩短持续时间并不是以提高试验电压为代价的，同样，降低试验电压时，也并没有延长持续时间。

## 2 泄漏电流试验

泄漏电流试验是测量电源输入端的每个极(相线和交流地线)与连在一起的所有可触及元件(包括接地端子)之间的电流。测试电路如图1所示。

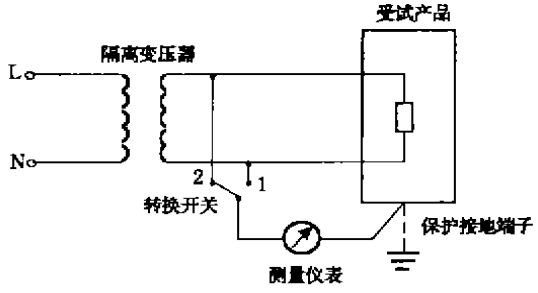



图 1 单相设备对地泄漏电流试验电路

测试时，受试产品的供电电压一般标准规定为1.1倍的额定电源电压，例如对于220V额定电压，测试时需施加242V的电压。但也有例外，GB4706.1-92规定，对于在工作温度下的泄漏电流，电动器具和组合器具在1.06倍额定电压的供电电压下测试；电热器具(组合型除外)供电电压应调整到使输入功率等于最大额定输入功率的1.15倍时测试。

一般中小功率的产品(例如消耗功率小于200W的产品)可以用专用泄漏电流测试仪器进行测量，而功率较大的产品因无专用仪器，就需要试验人员根据试验电路图，选取合适的电流表和变压器进行测量。这里需特别指出的是测试泄漏电流时，受试产品最好通过隔离变压器供电，否则受试产品必须与大地绝缘。因为若受试产品不与大地绝缘，则构成如图2所示的电路，这样不仅测不出标准所要求的泄漏电流，而且会烧坏仪表。由图2可知，若通电后测试，转换开关由1扳到2时，就对测量仪表加上了220V的电压，电流急剧上升，超过仪表的测量范围，而使之烧毁。

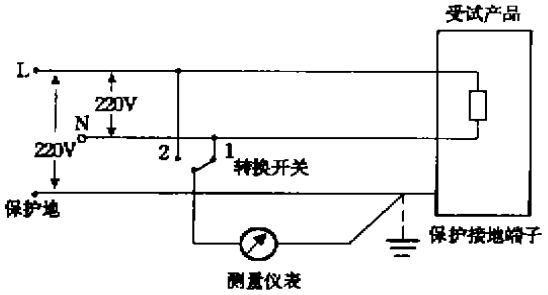



图 2 受试产品未与大地绝缘时的电原理图

防电击试验是安全试验的重要组成部分，是电子应用产品进行型式试验的必检项目，应引起产品设计、生产和检验人员的高度重视。

(收稿日期1996年5月12日)

(上接第19页)

$$x' = x \times \cos\theta + y \times \sin\theta$$
$$y' = y \times \cos\theta - x \times \sin\theta$$

据上式将函数RotateHZ()中的表达式做相应改动即可。

## 4 结束语

以上就UCDOS矢量字库的结构及矢量数据的组织方法进行了详细分析，据此我们能够容易地读取汉字的矢量数据并按要求(如放大、旋转等)显示输出。通过分析还得知UCDOS矢量汉字的基本大小为96*96，所以在进行放缩运算时要考虑到96这个因子。

## 5 参考文献

1 李卫国，《矢量汉字库的结构及汉字变换算法》，计算机世界月刊，1994.12。

2 郑春峰、麻伟，《利用UCDOS矢量字库制作立体汉字"透明贴"》，中国计算机用户，1996.2。

(收稿日期1996年5月20日)