关于 Windows 下图形绘制及输出方法的研究

何 泉 (合肥工业大学 98 号信箱 230009)

摘要:本文针对在 Windows 环境下图形绘制及图形输出的问题,以 Visual C++ 和 Delphi 编程环境为例,探讨了图形绘制和输出的基本方法。

关键词:图形绘制 图形输出

1.引言

在进行科学研究和编程过程中,经常会遇到需要输出图形的情况,如在进行频谱分析时,需将频谱分析的结果显示在计算机屏幕上或者打印出来以便进行分析研究,此时固然可以利用一些专业的图形软件如 AutoCAD,3DS 等或者某些软件自带的作图功能(如 MATLAB 等)进行图形输出,但在需要大量作图的情况下,例如在进行频谱分析的仿真过程中,需要输出各种信号、波形的谱分析结果,此时用上述软件就会觉得过于繁琐,另外有时使用这些专门的绘图软件会受到许多制约,难以实现我们所需要的特定结果。象我们在做合肥新客站 10 千伏配电所微机监控系统的项目时需要输出故障波形,按要求总共在一页内需输出两个周波共 100 个点,但我们编程所用的 Delphi 3 提供的 ActiveX 控件 VtChart 在屏幕上最多只能同时显示 24 个点,远远不能满足需要。因此,在许多时候我们必须在程序中自己实现绘制和打印功能,本文基于上述目的,以常用的编程工具 Visual C++ 和 Delphi 为例,阐述了关于图形输出的基本原理和方法。

2.图形绘制

在 Visual C++ 中,关于图形绘制的基本函数被封装在类 CDC 中,而在 Delphi 中是在类 TCanvas 中,二者十分类似,甚至大部分属性和方法都是一致的。这与 Windows 下统一的界面风格、API 函数和 Windows 图形对象(GDI 对象)有关。Windows 提供了两种影响 CDC 绘图工作方式的工具:笔(Pen)和刷子(Brush),通过对二者的调用可以定义当前的屏幕背景和画线的颜色、粗细和方式。当前影响画线的方式包括直线、曲线(如用 LineTo 或 Arc 函数绘制的)以及封闭图形周围的边框(如矩形和椭圆)。封闭图形包括两个分开的元素:边框和内部。当前刷影响封闭图形内部的绘制方式。其他图形对象有字体、位图、区域、轨迹和调色板,其中字体和位图我们也经常用到。

绘图开始时,首先要得到需绘制输出的数据,我们可以直接将数据传递给绘图函数,也可以将数据存放在文件或数据库中然后在程序中进行读取。进行绘图时首先要定义当前的设备情景对象(设备描述表),如在 Visual C++ 中,我们通过 CDC 类的派生类 CClientDC 利用 this 指针生成当前的设备情景对象,然后调用函数 SelectStockObject 获得库存的笔和刷子,也可根据需要生成定制的笔和刷子,如下面一段例子所示:

```
CClientDC MyClientDC(this);
CPen MyPen, * PtrCurPen; CBrush MyBrush, * PtrCurBrush;
MyPen.CreatePen(PS-SOLID,3,RGB(0,0,255));
MyBrush.CreateSolidBrush(RGB(225,0,0));
PtrCurPen=MyClientDC.SelectObject(&MyPen);
PtrCurPen=MyClientDC.SelectObject(&MyBrush);
```

以上程序生成了一支兰色,宽度为三个象素的实心笔和红色的实心刷子,而在 Delphi 中,我们可以通过以下程序实现相同的功能:

```
Myform.Canvas.Pen.Color:=clBlue;
Myform.Canvas.Pen.Style:=psSolid;
Myform.Canvas.Pen.Width:=3;
Myform.Canvas.Brush.Color:=clRed;
Myform.Canvas.Brush.Style:=bsSolid;
```

注意在 VC 和 Delphi 中,缺省的笔均为黑色、实心,宽度为一个象素,缺省的刷子均为白色的实心刷子。

在选定了笔和刷子之后,我们就可以通过调用两个最常用的函数 MoveTo 和 LineTo 来进行绘图了。笔者建议在绘图时最好将数据放在数组中以便运用循环语句进行绘制。如有必要还可利用其他的绘图函数绘制圆,椭圆,矩形以及不规则曲线等。具体函数请查阅有关手册。在需要输出文字时应调用函数 TextOut,但在 Visual

C++中输出数字时应注意要用 sprintf 函数对之进行格式化,否则会造成输出不符或是乱码。值得注意的是绘图属性和映象方式的选取,如在 VC 中调用函数 SetROP2(int nDrawMode)时,将 nDrawMode 设置为 R2-NOT时表示与原有颜色反相,而缺省模式 R2-COPY-PEN表示以当前笔的颜色进行绘图,映象方式则定义用于显示图形和文本的单位和坐标增长方式。但当前映象方式不影响接受设备坐标的函数。另外要注意逻辑坐标与设备坐标的区别和转换。

3.图形输出

经过图形的定位和绘制(一般情况下要获得满意的输出需要反复的进行定位和调试),我们可以在计算机屏幕上得到我们需要的输出。但大多数情况下还需要将所绘制的图形插入自己的文本中或打印出来以供研究。这时我们有两种最常用的方法:一种是通过剪贴簿将屏幕上的图形剪下来然后贴入文本编辑器如 Word 中;一种是直接在打印机上进行输出。下面分别介绍这两种方法:

第一种方法:以 Visual C++为例,这时又有两种处理方法:一种是将绘出的图形直接送到剪贴簿,另一种是仿照一般的图形处理工具自己制作菜单或工具栏以实现剪贴功能,以第一种方法为例,假设需要将屏幕上对角线上两个顶点分别为 Point1 和 Point2(Point2 在 Point1 的右下角)的矩形区域复制到剪贴簿上,程序如下:

```
CBitmap BitmapClip; CClientDC ClientDC(this); CDC MemDC;
//create an empty bitmap;
BitmapClip.CreateCompatibleBitmap(&ClientDC,
    Point2.x-Point1.x, Point2.y-Point1.y);
//create a memory DC object and select bitmap into it:
MemDC.CreateCompatibleDC(&ClientDC);
MemDC.SelectObject(&BitmapClip);
//copy contents of view window into bitmap:
MemDC.BitBlt
    (0, 0,
    Point2.x-Point1.x, Point2.y-Point1.y,
    &ClientDC,
    m-Point1.x, Point1.y,
    SRCCOPY);
//1.openClipboard:
if(! OpenClipboard())
return;
//2.remove current Clipboard content:
::EmptyClipboard();
//3.give bitmap handle to Clipboard:
:: SetClipboardData (CF-BITMAP, BitmapClip.m-hObject);
BitmapClip.Detach();
//4.close Clipboard:
::CloseClipboard();
```

第二种方法复杂一些,它类似于 Windows 下的"画图",可以将屏幕上用鼠标框取的一个矩形区域剪贴到目标文档中,方法是先在鼠标消息处理函数 OnLButtonDown, OnMouseMove, OnLButtonUp 中加入绘图语句以实现矩形的绘制,然后通过同上例相同的方法将之剪贴或是复制到剪贴簿中,应注意"剪切"和"复制"后处理的方法不同。

第二种方法是将图形直接在打印机上输出。以 Delphi 为例,在打印机上进行输出极为容易,只需利用 Printer Unit 中的 Printer.Canvas 便可象在屏幕上进行输出一样对打印机进行输出,如下例所示:

```
Printer.Canvas.TextOut('Hello, World');
```

用这种方法时要注意输出图形在打印纸上的位置和大小,在不满足需要时可通过坐标平移和加入比例因子的方法来改变输出,直到获得满意的效果。另外要注意的一点是在 Delphi 中进行图形输出时不要把绘图语句放在 FormCreate 事件中,否则将得不到输出。

4.小结

通过对在 Windows 环境下图形绘制和输出方法的研究,我们可以很方便的将数据以图形方式表述以便进行进一步的研究。笔者曾在对非整周期采样利用 Burg 算法进行频谱分析和合肥新客站 10 千伏配电所微机监控系统中采用了上述方法,得到了令人满意的效果。

参考文献

[1] Microsoft Visual C++ 2.0 for WIN32 大全。
[2] Delphi 3 从入门到精通。

(来稿时间:1999 年 1 月)