用 Visual Basic 实现测控软件中的实时曲线和历史曲线

刘定晟　杨　俊　蒋迪清　华南理工大学工控系 (510640)

Abstract

How to draw the real time curve and history curve in measuring and control software with VB is presented in this article. The main methods and key points to draw the curve under different system requirements are summarized here, which is instructive to develop measuring and control software with VB.

Keywords: visual basic, measuring and control software, real time curve, history curve

摘　要

本文介绍了如何用 VB 实现测控软件中的实时曲线和历史曲线，总结了在不同的系统要求下该曲线的设计思路和实现要点，对于用 VB 开发测控软件有一定的指导意义。

关键词：Visual Basic，测控软件，实时曲线，历史曲线

1　用 Visual Basic 开发测控系统软件简介

传统的测控软件开发工具多为 DOS 或 Windows 下的 C 语言开发，对软件开发人员的要求较高，从而造成系统开发周期长，可维护性差，而且 DOS 下的应用程序不具有标准的 Windows 图形用户界面。近几年来用 VB 开发测控软件已日益流行，它既可以使用 DLL 来实现 I/O 端口的输入输出功能或提高浮点运算能力，也可通过 API 函数或 Mscomm 控件实现串口通信，还能够充分发挥 VB 数据库功能强大以及生成用户界面快等优点。本文将详细介绍用 VB 开发测控软件的一个重要环节；实时曲线和历史曲线的实现。

2　实时曲线

实时曲线反映的是现场数据的实时性，以监测该点在现场工况变化的情况下的控制稳定性，因此在实现时需显示曲线的动态变化。通常当前点在曲线的最右端显示，随着时间的推进整个曲线动态地向左移动。

实现曲线的动态平移要涉及曲线消隐和重绘。曲线消隐的实现有几种方式：用 CLS 方法清屏；用 XOR 方式，即将画线对象的 DrawMode 属性设为 XOR Pen，而后在原位置处重画该曲线；用背景色重绘曲线，此法适用于单一背景色；Bitblt 方式，API 函数 BitBlt 可将一副位图从一个设备场景 (Device Context) 复制到另一个，通过此函数为设备场景赋以初始位图亦可实现曲线消隐。曲线重绘通常通过 BitBlt 函数实现，它以块复制的形式快速地传递位图而不必重复绘制数据点。巧妙地使用该函数既可以很好地实现曲线的平滑移动，又可以减少资源的消耗。

根据系统的不同要求，实时曲线的平滑移动一般分为以下三种情形。

1)无背景图，曲线平滑移动

此时曲线平滑移动是在一个无背景图的图片框中进行。如图 1 所示，当曲线尚未绘至图片框的右边界时，曲线不用平移，到达右边界后将曲线左移一个单位的数据点水平间距，再用背景色重绘最右端一个单位间距的线段以消隐该旧线段，然后用最新的数据绘制最右端线段。这种方法每次只重画最右一段曲线，效率高，占用资源少。

该子程序在 Timer 控件的事件中循环执行，源代码如下：

```
Private Sub Timer1_Timer ()
'下文中的同名变量若未说明则含义相同。
'num: 曲线一屏欲显示的总占数，公共变量　N: 当前点的编号
'cx: 当前点的横坐标，cy: 当前点的纵坐标，1x: L 上
```

一个点的横坐标 1y: 上一个点的纵坐标

'11y: 倒数第二点的横坐标　11x: 倒数第二点的纵坐标

'nw: 曲线上相邻两点的 X 轴间距 = 图片框宽度 /总点数,公共变量,为减少累计误差,类型为单精度

```
Static N As Integer, 1x As Single, 1y As Single, 11y As Single, 11x As Single
Dim cx As Single, cy As Single
Randomize: cy = 0.1 * h + Rnd * 0.8 * h ' 纵坐标值为随机产生,不是实际采样值,此处仅用做说明
If N <= num Then ' 曲线尚未绘至右边界,用 line 方法在目标图片框上直接绘制,不用平移
    cx = N * nw
    If N > 0 Then Pic1.Line (1x, 1y) - (cx, cy), vbBlack
    If N = num - 1 Then 11x = cx: 11y = cy
    N = N + 1
Else ' 曲线开始平移,原理如图 1 所示
```

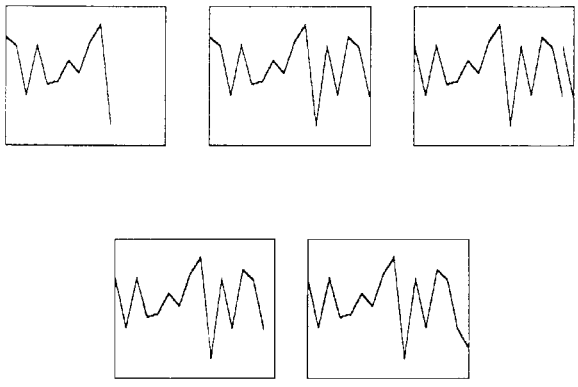

图 1　无背景图,曲线平滑移动的实现原理

```
    BitBlt Pic1.hDC, 0, 0, w - nw, h, Pic1.hDC, nw, 0, SRCCOPY ' 曲线左移一个单位
    Me.Pic1.Line - (1x, 11y), Pic1.BackColor ' 消隐旧线段
    Me.Pic1.PSet (1x, 11y), Pic1.BackColor ' 消隐该交点
    Me.Pic1.Line (w - nw, 1y) - (w, cy), vbBlack ' 绘制新线段
    11x = w - nw: 11y = 1y
End If
1x = cx: 1y = cy
End Sub
```

2)有背景图,背景与曲线一起平滑移动

这种情形在测控软件中十分常见。它的实现相对复杂,因为背景图往往不是单一颜色,所以目标图片框 (Pic2)中的最右段曲线的消隐不能使用用背景色重绘的方法, 而是通过一个辅助图片框 (Pic22)实现。Pic22 的高度与 Pic2 相同,宽度是其两倍,背景图是两幅 Pic2 图片的水平并列放置。当 Pic2 内的曲线尚未绘至右边界,用 line 方法直接在其上绘制,不用平移。

当 Pic2 中的曲线绘至右边界时,将该曲线 bitblt 至 Pic22 中并且从 Pic22 的中部继续绘制曲线。

其后, 从 Pic22 的相应位置将当前曲线 biblt 至 Pic2, 从而保证目标图片框 Pic2 中曲线及其背景的平滑移动。

当 Pic22 中的曲线绘至右边界后,把 Pic22 清空并将 Pic2 中的曲线 bitblt 至 Pic22 中,然后从 Pic22 的中部继续绘制曲线。

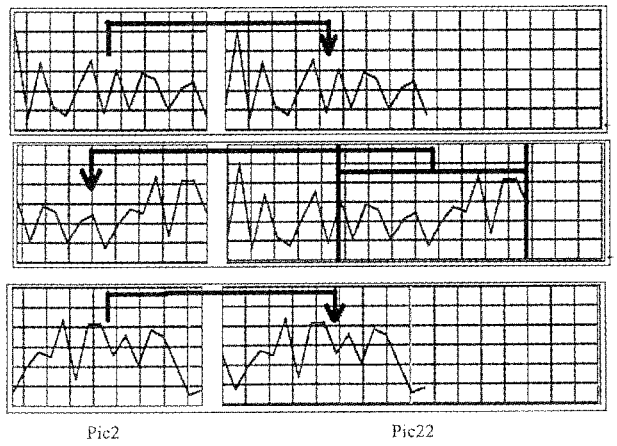

图 2　有背景图,背景与曲线一起平滑移动的实现原理

实现图 2 中所示部分的代码如下:

```
'cuCopyX: PIC22 中向 Pic2 Bitblt 曲线的起始位置的当前值,数据类型为单精度,
以下代码为当 Pic2 内的曲线绘至右边界后的情形,因篇幅限制略去变量声明
If N = num + 1 Then '此时 Pic2 或 Pic22 中的曲线恰好绘至右边界
    Pic22.Cls          '清空 Pic22
    '将 Pic2 中曲线 bitblt 至 Pic22 中
    d1& = BitBlt (Pic22.hDC, 0, 0, Pic2.Width, Pic22.Height, Pic2.hDC, 0, 0, SRCCOPY)
    cuCopyX = nw       ' 重置 cuCopyX 为初值, 即一个单位的数据点水平间距
    1x = num * nw
```

```
End If
Pic22.Line (lx, ly)-(cx, cy)
dl& = BitBlt (Pic2.hDC, 0, 0, Pic2.Width, Pic2.Height, Pic22.hDC, cuCopyX, 0, SRCCOPY)
cuCopyX = cuCopyX + nw
If N = 2 * num Then N = num
N = N + 1
```

3)有背景图,曲线平滑移动,背景图不动

要实现这种效果需要将背景图与曲线本身分开。如图 3 所示,先在一个无背景的辅助图片框(Pic33)中绘制曲线,实现方法同情形 1),然后在带背景图的目标图片框(Pic3)中用 Cls 清空上一次的曲线,最后将 Pic33 中的曲线'透明'地与 Pic3 背景图叠加后在 Pic3 中得到当前的带背景图的曲线。

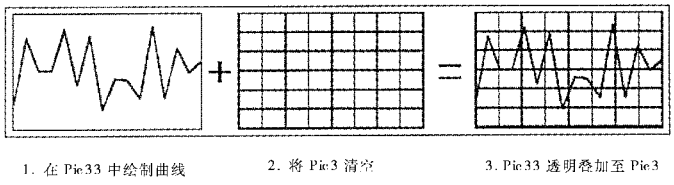

图 3 有背景图,曲线平滑移动,背景图不动的实现原理

此法的关键在于如何实现图形的透明叠加,有一个未公开的 API 函数(msimg32.dll 库中的 TransparentBlt)可以实现这个功能。但在使用中发现,多次使用该函数会导致系统的资源减少至至崩溃,而测控系统一般都需要长时间地循环采集数据以绘制实时曲线,所以不能采用该函数。我们可以自行编写一个通用的子程序 TransBlt,然后在情形 1)折源代码中加入以下语句实现 Pic33 中曲线与 Pic3 背景图的透明叠加。

```
TransBlt Pic3.hDC, 0, 0, Pic3.Width, Pic3.Height, Pic33.hDC, 0, 0, vbBlack.
```

子程序 TransBlt 的源代码如下:

```
Public Sub TransBlt (ByVal hDestDC As Long, ByVal x As Long, ByVal y As Long, ByVal n Wdth As Long, _
    ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long_
    ByVal ySrc As Long, ByVal TransColor As OLE_COLOR)
'本子程序中使用的 API 函数需另外声明且因篇幅限制,本子程序变量声明略去。
'创建设备场景
    saveDC = CreateCompatibleDC (hDestDC): maskDC = CreateCompatibleDC (hDestDC)
    invDC = CreateCompatibleDC (hDestDC): resultDC = CreateCompatibleDC (hDestDC)
'创建与设备有关的位图
    hMaskBmp = CreateBitmap (nWidth, nHeight, 1, 1, ByVal 0&):
    hInvBmp = CreateBitmap (nWidth, nHeight, 1, 1, ByVal 0&)
    hResultBmp = CreateCompatibleBitmap (hDestDC, nWidth, nHeight)
    hSaveB = CreateCompatibleBitmap (hDestDC, nWidth, nHeight)
    hSavePrevBmp = SelectObject (saveDC, hSaveBmp): hMaskPrevBmp = SelectObject (maskDC, hMaskBmp)
    hInvPrevBmp = SelectObject (nvDC, hInvBmp): hDestPrevBmp = SelectObject (esultDC, hResultBmp)
'产生 Mask 图象
    origColor = SetBkColor (hSrcDC, TransColor)
    dl& = BitBlt (maskDC, 0, 0, nWidht, nHeight, hSrcDC, xSrc, ySrc, vbSrcCopy)
    TransColor = SetBkColor (hSrcDC, OrigColor)
'invDC 的图象将与 maskDC 图象相反
    dl& = BitBlt (nvDC, 0, 0, nWidth, nHeight, maskDC, 0, 0, vbNotSrcCopy)
'resultDC 的图象将成为被写位置的图象
    dl& = BitBlt (esultDC, 0, 0, nWidth, nHeight, hDestDC, x, y, vbSrcCopy)
'resultDC 中,需要新写的位置将变为黑色
    dl& = BitBlt (esultDC, 0, 0, nWidth, nHeight, maskDC, 0, 0, VBsrcAnd)
    dl& = BitBlt (saveDC, 0, 0, nWidth, nHeight, hSrcDC, xSrc, ySrc, vbSrcCopy)
'resultDC 中不被写入的颜色成为黑色
    dl& = BitBlt (saveDC, 0, 0, nWidht, nHeight, invDC, 0, 0, vbSrcAnd)
'将两幅图合并起来
    dl& = BitBlt (esultDC, 0, 0, nWidth, nHeight, saveDC, 0, 0, vbSrcInvert)
'完工后输出
    dl& = BitBlt (hDestDC, x, y, nWidht, nHeight,
```

resultDC, 0, 0, vbSrcCopy)

SelectObject saveDC, hSavePrevBmp: SelectObject resultDC, hDestPrevBmp

SelectObject maskDC, hMaskPrevBmp: SelectObject invDC, hInvPrevBmp

'释放资源

DeleteObject hSaveBmp: DeleteObject hMaskBmp: DeleteObject hInvBmp: DeleteObject hResultBmp

DeleteDC saveDC: DeleteDC maskDC: DeleteDC invDC: DeleteDC resultDC

End Sub

3 历史曲线

历史曲线反映的是过去一段时间内某个或多个监测点的变化趋势,并供工艺人员分析工艺流程的稳定性和故障原因,不需要实时的平滑移动显示而且通常会进行一定的数据分析, 可以采用 Adodc 控件和 MS Chart 控件的数据绑定来实现。首先根据设定的查询条件,使用 SQL 语句通过 ADO 控件获得相应的数据记录集, 然后将它与 MS Chart 控件绑定即可实现曲线的自动绘制。如图 4 所示,MS Chart 控件功能较强,可设置多个序列 (即多个测控点的历史曲线)以及图例、坐标、脚注等属性,并可直观地得到曲线的最值、均值、方差和一维线性回归等, 所得曲线可以用 EditCopy 方法传递给剪贴板而进行打印或报表输出。具体用法参见该控件的帮助。若要进行数据的深入定性分析可以利用 ActiveX Automation 技术访问 Excel 的对象集合,将数据导出为 Excel 格式,然后用 MatLab 等工具分析。

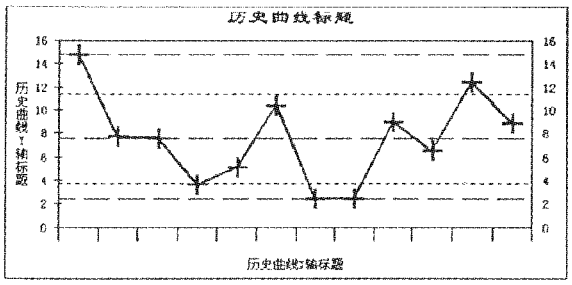

图 4 用 MS Chart 控件的数据绑定实现历史曲线

在上述图形编程中要注意图形对象 (如 Picture, Form 等)的 Autoredraw 属性的设置,该属性为 True 时可使所画区域的图象在被其它窗体覆盖撤除后能自动重画,但会严重影响画线速度, 占用系统内存资源。例如在实现有背景图的曲线平滑移动中,当目标图片框 pic2 的 Autoredraw 属性为 True 且采样周期较短时还会造成图形显示尚未完成却又进行了下一次的刷新操作, 造成运行时曲线显示为空。若想在 Autoredraw 属性设为 false 时保留自动重画功能,可在图形对象的 Paint 事件中, 编制重画代码或用 Timer 控件不断刷新显示。上述例子中各图片框的 Autoredraw 属性分别为: Pic1: True, Pic2: False, Pic22: True, Pic3: False, Pic33: True。此外在使用 API 图形函数时, 图形对象的坐标位置, 图片大小的均以象素为单位, 若图形对象的 ScaleMode 属性不为 Pixel,则要做相应的转化,如 Twip 与 Pixel 的转换关系可以利用 Screen 对象的 TwipsPerPixelX、TwipsPerPixelY 属性。

4 小结

测控系统中数据的图形显示主要涉及实时曲线和历史曲线的绘制。在 VB 中使用图形的 API 函数和 MsChart, ADO 等控件可以快速、高效地实现测控软件中的实时曲线和历史曲线。本文介绍的方法在我们开发的果树生长室温度测控系统中取得了较好的成果, 程序在 VB6.0 中文版, Windows98 环境下调试成功。

(上接第 41 页)

3) 速度高,ISP 技术实际是硬件实现模糊控制,使输入输出延迟时间降为微秒级, 这是模糊单片机等软件实现技术难以比拟的。

4) ISP 在系统编程技术意味着无需电路板返工,安装好的模糊控制器可以通过 PC 机烧写电缆升级成新的硬件组态模式,甚至可通过网络实现遥控升级。

5 结束语

关于 ISP 制作模糊控制系统是一个系统工程,本文仅给出了基本的思想和方法步骤,ISP 技术在模糊控制器的应用还有待进一步探索和完善, 如果在 ISP 芯片中包含计算模块,通过复杂的设计还可以实现更为复杂的模糊控制, 如自调整控制规则的自组织模糊控制等。ISP 技术在模糊控制中的系统级应用因具备许多优点而有广阔的应用前景。

参考文献

1. 韩启纲等, 计算机模拟控制技术与仪表装置, 中国计量出版社,1999

2. 刘笃仁等, 在系统可编程技术及其器件原理与应用, 西安电子科技大学出版社,1999

3. 孙增圻等,智能控制理论与技术,清华大学出版社,1997