Windows32 驱动软件 输入设置 三维空间坐标 (23)

计算机科学1999Vol.26№.11

61-63

3D 空间坐标输入装置驱动软件的研究与实现

Research and Realization of 3D Space Coordinate Input Device Driver

刘金刚 梁化有 张 倩 TP316 TP334

(首都师范大学计算机应用联合实验室 北京100037)

(国家智能计算机研究开发中心 北京100080)

Abstract In order to make three-dimension coordinate input device widely applied by the more people and technicians able to develop applications for the device under the base of the driver, this paper publishes some key technique about the device driver. The primary function of the driver is to keep the communication between the device and computer, data acquisition from the serial port of computer, data processing and transformation etc. At the same time the device driver provides an interface with operating system. The driver software is developed in Visual C++5.0 above the Windows95.

Keywords Space coordinate, Input device, Device driver

一、程序总体设计

作为硬件的驱动程序首先要保证实时性，只有满足实时性的要求才能使有用数据不丢失，而且数据的有效性也有保证。从程序总体结构的设计上利用了Win32的新特性：使用进程和线程的非抢占多任务，即允许多进程与多线程同时执行实现多任务。一个程序可以有多个进程，一个进程又可以有多个线程，可以利用不同的进程去完成不同的任务。而且象进程一样，一个线程通过分配一个优先权级别来决定获得时间片的大小，这样可以定义用哪个线程处理实时任务，用哪个处理非实时任务。

由于Win32的这些新特性为我们驱动程序的开发提供了便利，3D空间坐标输入装置的程序由主线程和一个辅助线程构成，在辅助线程中完成对串口的数据采集、处理以及和操作系统的接口，由于该线程的优先级比较高，保证了对串口数据采集的实时性，一旦有串口数据来线程立即响应，进行采集和处理。在主线程中主要完成对参数的设定，并进行消息的循环。该程序采用对Win32线程和线程同步对象进行了封装的MFC类，简化了程序的设计。

在多线程程序中存在线程调度的问题，即每个线程都不知道进程中其他进程，除非明确地使它们互相可见。为了不同时刻访问这些资源，任何共享进程资源的线程必须通过进程间通信的方式调度，不正确地调度线程会引起进程的死锁。为了防止这种情况的发生，访问共享资源必须同步线程的执行，这里主要是通过同步对象CSingleLock对象和CEvent对象来实现主从线程的同步。

关于对串口的操作主要利用操作系统提供的对串口进行操作的Win32 API函数，完成对串口设备的创建、设置、初始化以及进行读写访问等。CreateFile函数打开串口输入设备，返回设备句柄，用于对该设备的访问，ReadFile函数完成数据从串口设备读入数据缓冲区，它是按文件指针所指的位置开始从文件中读取数据的。

程序组成结构图如图1所示。

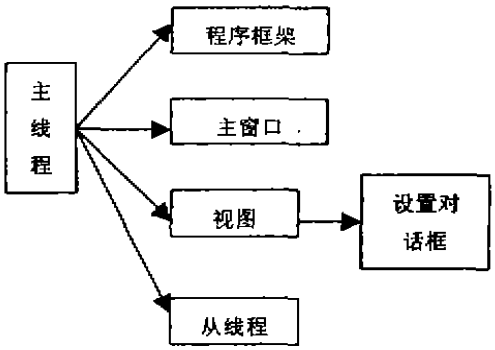

图1

二、串口数据的采集和整理

下位机采集的数据通过串口送给微机，利用哪个

串口进行数据采集可以通过软件设定，在驱动软件设置对话框中，有串口选择项，COM1/2根据输入装置实际连接的串口进行设置。

对串口的采集是通过辅助线程来完成的，该线程始终守着串口，下位机有数据发送到串口，软件立即作出反应，读取数据并进行处理，保证缓冲区中不出现数据堆积，不丢失有用数据，而且具有极高的响应速度。

驱动程序从串口采集的数据按时间的先后放在一个256字节大小的缓冲区里，然后驱动程序再对缓冲区中的数据进行处理。按通信协议的规定，每组数据由6帧组成，第六帧都是零用于分割两组数据，每帧十位，首位是起始位，末位为结束位，因此一帧数据的有效位是一个字节，这样一组数据就可以用六个字符变量来保存，处理比较方便。只需利用位运算按着协议的规定就可以得到一个空间点的原始坐标。

如何保证从数据缓冲区中取出的60位数据是同一组数据呢？这就需要利用每组数据中最后一组都是零，这组数据比较容易找到，它之后就是另一组数据的开始，通过这种方法保证了数据帧不会发生混乱。

三、数据的变换和平滑处理

1. 坐标的计算

单片机发送的数据经过上述的整理，得到的是超声波发射传感器到三个接收传感器的空间距离。如果想得到空间点的坐标还必须进行变换。变换是利用三点空间定位的原理：即在一空间坐标系中，如果已知任意空间点到三个坐标确定的点的距离，就可以换算出该点的坐标。

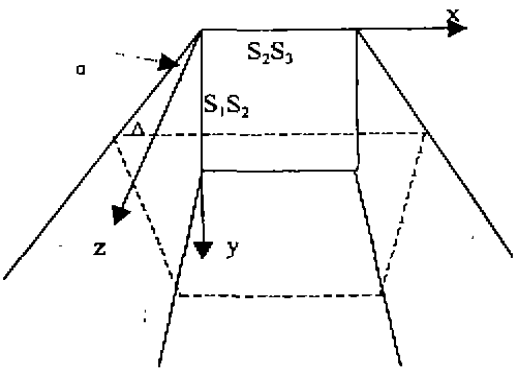

图2

如图2所示的空间坐标系，其中A，B，C是三个接收传感器所在的位置，O是发射传感器所在的位置，其坐标在不断地变化。在坐标系O(x，y，z)中A，B，C的坐标是已知的，用S_1S_2，S_2S_3表示A，B和B，C两点之间的距离，可通过实测得到。该输入装置为了适应不同大小的显示器，传感器之间的距离可调，因此S_1S_2，S_2S_3是变化的。

在该坐标系下，根据OA，OB，OC的距离和A，B，C三点的坐标就可以推算出O点的坐标(x，y，z)，具体公式如式(1)，将由下位机发送的三个距离值代入式(1)就可以算出当前O点的坐标值，实现三维坐标信息的输入。有了三维坐标信息就可以实现空间定位，进行各种三维操作。

$$\begin{cases} x=\dfrac{OB^2-OC^2+S_2S_3^2}{2S_2S_3} \\ y=\dfrac{OB^2-OA^2-S_1S_2^2}{2S_1S_2} \\ z=\sqrt{OB^2-x^2-y^2} \end{cases} \quad (1)$$

2. 平滑处理

由于外界的干扰等不稳定因素的影响，经变换得到的三维坐标稳定性不够理想，光标存在抖动现象。为了能够更加精确地定位，在该软件中对下位机发来的原始数据(即空间距离值)，进行了平滑处理，然后再将其变换为空间坐标。具体的方法是，建一个队列(FIFO)记录最新的n组距离值，在每次循环中，将这n组数据平均作为当前的空间距离，再用它来计算该时刻的空间坐标。

从实际的处理结果看，当n在10左右时，平滑处理的效果比较好，基本消除了光标的抖动，使操作更加灵活，方便，如果n过小起不到平滑的效果，过大则产生明显的滞后现象。

3. 扇面处理

为了增大发射装置移动的范围，对数据进行了进一步的处理，使发射器的有效范围呈扇形，即离显示器越远(z坐标越大)发射器移动的有效范围越大，如图3所示。

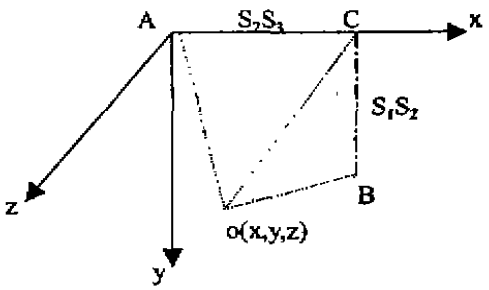

图3

变换公式为：

$\Delta=z\times\tan(\pi\cdot a/180)$

$x=x+\Delta$、$y=y+\Delta$

$x=65535\cdot x/(S_2S_1-2\cdot\Delta)$

$y=65535\cdot y/(S_1S_2+2\cdot\Delta)$

经过以上的变换，使发射器移动的有效区随着 z 坐标的增大扩展，形成扇形的区域，但都映射到0～65535这一固定的区间，即光标能够达到屏幕的任意位置。在未进行该变换之前，在任何 z 坐标固定的平面上，发射器移动的范围是不变的，都为 $S_1S_2\times S_2S_1$ 的平面。扩展之后更加灵活方便，而且扩展角 α 可根据需要进行调整，以适应于不同的应用需要。

四、与操作系统的接口

1. 接口函数

原始数据经过以上各种变换和处理，提取出了坐标的按键信息和空间坐标，利用 mouse_event 函数，作为鼠标事件将这些信息发给操作系统。

mouse_event 函数综合了鼠标的按键状态和鼠标的移动信息，函数的原型如下：

```
VOID mouse_event(
    DWORD dwFlags,
    DWORD dx,
    DWORD dy,
    DWORD dwExtraInfo
);
```

参数说明

dwFlags：说明鼠标的各种运动形式和按键状态的一组标志位。该标志可以是各种值的合理组合。MOUSEEVENTF_ABSOLUTE 说明 dx 和 dy 两个参数包含的是绝对坐标，如果该标志没有设定则包含的是相对坐标：即坐标的变化。MOUSEEVENTF_MOVE 说明有鼠标移动事件发生。

dx：指鼠标在 x 轴方向上的绝对位置或最后一次鼠标事件发生后鼠标的位移量，具体由 MOUSEEVENTF_ABSOLUTE 的设置决定。

dy：指鼠标在 y 轴方向上的绝对位置或最后一次鼠标事件发生后鼠标的位移量，具体由 MOUSEEVENTF_ABSOLUTE 的设置决定。

dwExtraInfo：和鼠标事件有关的32位值，用于对鼠标信息的扩展。其他应用程序调用 GetMessageExtraInfo 函数来获得该扩展信息。

该输入装置是工作在绝对坐标方式下，设为 MOUSEEVENTF_ABSOLUTE，因此 dx 和 dy 是鼠标的绝对坐标，第三维坐标信息通过 dwExtraInfo 发送给操作系统。在发送坐标信息的同时，按键的变化状态也同时被发送，从而实现了该驱动程序和操作系统的接口。由于三维坐标消息和按键信息是发送给操作系统的，这样上层三维软件的开发就比较简单，只要从操作系统获得三维坐标和按键信息，利用 GetMessageExtraInfo 可以获得 z 坐标信息，作到与设备无关。

2. 按键的防抖动处理

在对样机的实际操作中，发现当发射器（带在手上）在移动或不动的情况下，效果都比较理想。当用光标对某菜单、按钮进行单击或双击操作时，由于该装置悬在空间，当拇指要做用力对按键进行按压操作，此时手必然产生相对较强的抖动，使单击和双击操作无法实现，也就无法进行有效的操作。

解决这一问题的出发点是既然要求在按键的瞬间鼠标的位置不发生变化，那么可以在检测到鼠标按键被按下的同时，强迫鼠标的位置不发生变化，即 mouse_event 函数的 dx，dy 和 dwExtraInfo 三个参数保持不变。由于按键信息和坐标是同时发送的，按键信息是必须要的，因此此处又不能停止 mouse_event 发送鼠标事件。

具体的解决办法是在每次发送鼠标事件，对鼠标的位置信息都进行备份。当检测到有鼠标键按下事件发生时，mouse_event 发送的坐标是上次备份的坐标，而不是当前采集到的实际坐标，这样使鼠标位置不发生变化，而且按键信息照常，这样保持几十毫秒双击和单击操作都可以完成了，经过这一处理之后，彻底解决了按键时坐标的抖动，而且操作时没有异常感觉。

小结 本文较详细地阐述了三维空间坐标输入装置驱动软件开发的方法及解决和处理问题的关键技术。通过该驱动软件的介绍，使人们对三维输入装置能够有更深入的了解。在本驱动程序的基础上，二次开发人员可以简单地获取该装置的三维实时坐标信息。方式如下：在主程序消息处理循环中：

```
case WM_MOUSEMOVE:
    g_MouseZ=GetMessageExtraInfo();
```

其中，g_MouseZ 为 LONG 型变量，返回值即为 Z 轴上0—65535的数值信息。通过获取的该数值，二次开发人员可根据实际应用程序进行变换处理。

参 考 文 献

1 Norton P，McGregor R. MFC 开发 Windows 95/NT4应用程序. 清华大学出版社，1998

2 Richter T. Windows 95/NT3 5高级编程技术. 清华大学出版社，1996

3 黄遵熹. 单片机原理接口与应用 西北工业大学出版社，1997