doi: 10.7690/bgzdh.2021.11.006

飞行仿真系统虚拟仪表关键技术

董　斐，刘剑超，林亚军，吕　游
（海军航空大学教练机模拟飞行训练中心，辽宁 葫芦岛 125000）

摘要：为满足模拟飞行训练的要求，构建真实性强且支持多平台运行的虚拟仪表，对虚拟仪表系统的关键技术进行研讨。基于 Visual Studio 2003 与 GL Studio 平台，在现有传统仪表的基础上，分别对虚拟仪表的按钮控件和计时器系统进行设计，通过设立缓存对象使主控送数与人工送数功能同时运行，给出参数处理逻辑算法，根据需求设置系统的修改功能，并通过构建数字仪表进行算法验证。经验证数字仪表运行状况良好，上述功能基本得到实现，运行结果证明了上述关键技术的有效性。

关键词：虚拟仪表；GL Studio；按钮控件；人机交互
中图分类号：TJ85　　**文献标志码：**A

Key Technology of Virtual Instrument in Flight Simulation System

Dong Fei, Liu Jianchao, Lin Yajun, LYU You
(*Training Airplane Simulator Training Center of Naval Aviation University*, *Huludao* 125000, *China*)

Abstract: In order to meet the needs of simulated flight training and build virtual instrument with strong authenticity and support for multi-platform operation, the key technologies of virtual instrument system are discussed. Based on Visual Studio 2003 and GL Studio platform, on the basis of existing traditional instrument, and parameters of the button control system part, provide some addition logic algorithm for implementing the following functions. By setting the cache object makes the data transmission of master and manual operation and can run at the same time, improve the instrument human-computer interaction ability. A timer system suitable for virtual instrument is designed. On the basis of the timer, a button control response system is set up to recognize the button behavior freely and improve the flexibility of the using of buttons. Finally, a digital instrument is constructed based on the algorithm operates well, verified the algorithm and the above functions are basically realized, and the above key technologies are proved to be effective by operation results.

Keywords: virtual instrument; GL Studio; button control; human-computer interaction

0　引言

虚拟仪表系统是模拟飞行系统中的重要组成部分，在飞行员等特殊驾驶员的模拟训练领域内有着重要意义。虚拟仪表系统主要任务为接收模拟飞行主控系统（以下简称“主控”）数据[1]，通过模拟器机载显示器将飞行参数等实时数据实时显示[2]。随着飞机相关仪表的交互功能愈加强大，仪表界面不再仅作为显示屏幕使用。虚拟仪表对相关按钮控件功能的要求更高，并且要求虚拟仪表系统具备一定的数据处理能力[3]，以便于飞行员可实时对数据进行修正；因此，一套较为完整的虚拟仪表系统中，按钮控件需要实现以下功能：

1) 对主控系统送来的相关参数进行修正，以达到参数调 0 或使参数按照飞行员要求进行输出的效果。修正过程中数据显示功能不能受到影响，不论从主控还是按钮控件得到的数据都要实时显示。修正后，相关参数需根据修正结果重新计算。

2) 根据训练要求，仪表界面需要一套独立的计时系统。该系统的计时方式不能通过调取 Windows 的系统时间来实现，且为保证仪表运行顺畅，时间系统不能通过延时函数进行计时。该套计时系统通过按钮控件实现激活、暂停、清零的功能。

3) 可修改表盘的基本设置，如亮度改变、表盘切换、表的精度与量程修改等。修改过程中不能影响仪表的实时显示功能[3]。

4) 按钮控件需要识别按下、抬起、长按、短按 4 个动作，按钮控件的响应事件需避免使用 Windows 鼠标事件。

笔者将基于 GL Studio 软件对该虚拟仪表系统进行开发，在传统仪表的基础上实现上述功能。

1　按钮设计

笔者采用按钮信息采集的核心函数为 ObjectEventIs（ev,“StateChange”）[4]函数。该函数控件自带，在按钮按下抬起时都可以进行响应。在调

收稿日期：2021-07-20；**修回日期：**2021-08-28
作者简介：董　斐（1993—），男，山东人，硕士，从事军用仿真技术、虚拟仪表研究。E-mail: dfgyt@163.com。

用之前需要设立 2 个全局变量 btn_state 与 state_time，分别用于记录按钮按下的状态与时间，为设立不同的按钮响应事件打下基础。另需一个全局变量 copy_time 用于传递时间参数。由于 calculate 函数中的 time 参数非对象或全局变量，因此，调用时可能出现错误，建议另行定义全局变量，在 calculate 函数中予以赋值，以便接下来的使用。按钮响应事件的代码如下：

```
If (btn_state==true)
{
    btn_state==false;
    if(copy_time-state_time>=1.5)
    {
        //按钮长按响应事件
    }
    if(copy_time-state_time<1.5)
    {
        //按钮短按响应事件
    }
}
If (btn_state==false)
{
    btn_state=true;
    state_time=copy_time;
    //按钮按下响应事件
}
```

2 计时器设计

在实现计时器之前需明确整个虚拟仪表系统的运行流程。整个运行流程如图 1 所示。

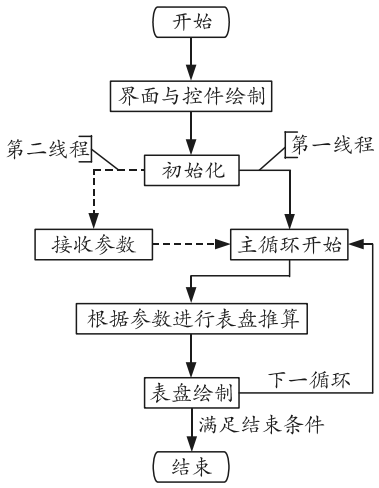

图 1 计时器运行流程

软件完成初始化后，双线程同时开始运行，同时仪表自带的计时系统也开始运行。该计时系统的精度为微秒，双线程分别以该计时系统为依据，按照自己的计时节点进行参数采样与仪表构建[5]。通过该计时系统完成所需的计时器构建可行。笔者构建一个基本计时器的思路：以文本框形式分别输出时钟、分钟与秒钟；通过一个按钮对计时器进行操控，初始状态下分钟与秒钟数字为白色，按钮短按一下，数字变绿开始计时，再次短按数字变红停止计时，长按数字清零数字重新变为白色。

显然，为实现上述功能，计时器的核心计时功能要在主循环函数即 calculate 函数中实现，计时器的控制功能要在上一节所述的按钮响应函数中实现，二者通过计时器状态参数 time_state 与节点时间参数 time_record 进行参数传递，用于传递的参数需保证为全局变量。

计时器核心计时功能代码如下：

```
If(time_state==1)
{
    text_sec->VaString("%d",fmod(copy_time-time_record,60.0f));
    text_min->VaString("%d",(int)((copy_time-time_record)/60));
    text_hour->VaString("%d", (int) ((copy_time-time_record)/360));
}
```

在给分钟赋值时，不建议使用秒钟满 60 s 分钟加一的逻辑，主循环的循环次数过快会导致多次触发分钟加一事件，从而出现错误。时钟同样如此。分钟与时钟需通过 copy_time 直接进行计算。

按钮响应函数中的代码如下：

按钮短按的响应事件：

```
If (time_state==0)
{
    time_state=1;
    text_sec->TextColor(glsColor(0,255,0,255));
    text_min->TextColor(glsColor(0,255,0,255));
    time_record=copy_time
}
If (time_state==1)
{
    time_state=2;
    text_sec->TextColor(glsColor(255,0,0,255));
    text_min->TextColor(glsColor(255,0,0,255));
    time_record=copy_time
 }
If (time_state==2)
{
    time_state=1;
    text_sec->TextColor(glsColor(0,255,0,255));
```

```
    text_min->TextColor(glsColor(0,255,0,255));
    time_record=copy_time-time_record
}
```

按钮长按函数：

```
Time_state=0;
text_sec->TextColor(glsColor(255,255,255,255));
text_min->TextColor(glsColor(255,255,255,255));
text_sec->VaString("%d",0);
//分钟时钟同样如此
```

3　修改功能设置

软件提供 3 个可设置的功能：界面设置、参数设置计算与亮度设置。

GL studio 具有一个较为成熟的对象编辑系统，通常的仪表数据都通过对象的形式予以编辑[6]。在定义一个新对象时，对象的数值成员变量与赋值函数已经自动定义好，直接使用即可。仪表的相关数据保存在成员变量中，由该参数控制的控件响应函数保存在赋值函数内；因此，只要该对象被赋值，则相关控件随之响应。

综上所述，修改功能设置的基本思路：为相关控件设立观察对象，对象通过识别成员变量控制控件状态；按钮响应函数通过识别按钮行为为对象赋值。

3.1　界面设置

随着相关多功能表的普及，表的界面功能愈发多样化，通过按钮控制可实现多界面切换、控件的消失与显示。功能实现的核心功能与上文类似：通过按钮控制相关参数的改变，在相关参数的赋值函数中编写界面的响应事件。

建立对象 Page，在相关按钮控件中通过 Page 赋值函数进行赋值，在 Page 赋值函数中通过修改界面的可视与否实现界面切换。如果界面和相关参数数量较少，但界面显示的逻辑较为复杂的情况下使用该方法较为有利。核心代码如下：

```
Interface->Visible(0);
Interface->Visible(1);
```

1 代表显示，0 代表隐藏，interface 代表需要隐藏或显示的相关控件。

若界面数量较多，且显示逻辑较为简单，即每次只显示一个界面。为方便管理，直接将所有界面建立成一个 Mutex 组群，通过调用组群的 Mutex State 参数进行赋值，赋值方式同样通过一个专门记录界面状态的对象进行，在按钮响应控件中对该对象通过赋值函数进行赋值。核心代码如下：

```
Display->MutexState(0);
Display->MutexState(1);
```

Display 用于代指所建立的 Mutex 组群的名字，0 和 1 分别代指不同的屏幕状态。

3.2　亮度设置

表盘亮度设置的核心思路为在需要调整亮度的表盘上另加图层，图层的绘图模式为 filled，且颜色为纯黑色，通过修改图层的透明度以达到调整表盘亮度的功能；因此，首先定义一个 Brightness 对象，建立一个 bkg_bright 图层，并在其赋值函数中控制该图层透明度。通过按钮控件调用该对象的赋值函数。按钮控件中代码如下：

```
unsigned short newBrightness = Brightness();
    newBrightness =
(newBrightness+16>255?:newBrightness+16);
    Brightness(newBrightness);//变暗代码
unsigned short newBrightness = Brightness();
    newBrightness =
(newBrightness-16＜0?:newBrightness-16);
    Brightness(newBrightness);//变亮代码
```

在 Brightness 对象赋值函数中代码如下：

```
_brightness=value;
glsColor curColor=bkg_bright->GetFillColor();
curColor.A(_brightness);
bkg_bright->SetFillColor(curColor);
```

3.3　参数设置与计算

随着多功能表的发展，仪表不再只具有显示功能，而需要对数据具有一定的保存与计算能力，以供飞行员随时对其修改，但是目前所用的模拟仪表都采用 UDP 协议，信息传输是单向的，数据修改的结果不能返回主控只能保存在仪表计算机中；因此，主控系统与按钮控件同时调用赋值函数时将会冲突，最后的数据只由一方决定而不能二者兼顾。笔者解决该问题的核心思路：为同一个表设立 2 个对象，由此获得 2 个赋值函数分别分给主控系统与按钮控件使用，这样通过二者获得的数据互不干涉；同时，2 个对象的赋值函数中分别加入控制仪表显示的函数，因此，不管哪一个发生变化都能迅速反馈到仪表上。控制仪表显示的函数须同时调用 2 个对象的成员变量，以保证 2 个对象的值都能反应在仪表上。赋值系统结构如图 2 所示。

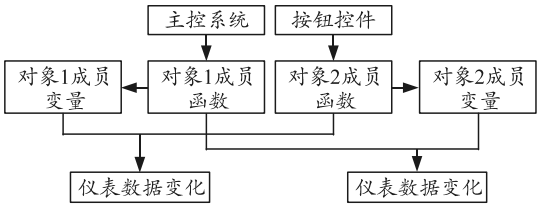

图 2　赋值系统结构

将需要通过按钮设置的对象设为 num1，相对应其赋值函数为 num1()，成员变量_num1，如上文所述，需另设一个对象 num2 供主控系统进行赋值，赋值对象与成员变量分别为 num2()与_num2，num2 对象交由主控系统进行赋值。赋值过程笔者暂用 RampFloat 函数进行替代；num1 对象交由按钮控件系统进行赋值，由 2 个对象的成员变量相加得到仪表的最终结果，该最终结果由 num_text 文本框控件进行显示。

核心代码如下：

```
num1(_num1+1);
//按钮控件中代码
num2(RampFloat(time*.07,0,100));
//Calculate 函数中代码
num_text->VaString(“%d”,_num1+_num2);
//2 个对象赋值函数中的核心代码
```

4　界面设置与功能介绍

软件界面的基本构成如图 3 所示。M 键为功能设置，短按一次，则可用加、减号按钮控制按钮获取参数；M 键再短按一次，加、减号按钮控制界面亮度。M 键长按 1.5 s 为界面切换功能。测键为时间测量功能键，用于激活、暂停、清零计时器。界面切换结果如图 4 所示，亮度变化结果如图 5 所示。

图 3　功能测试

图 4　界面切换

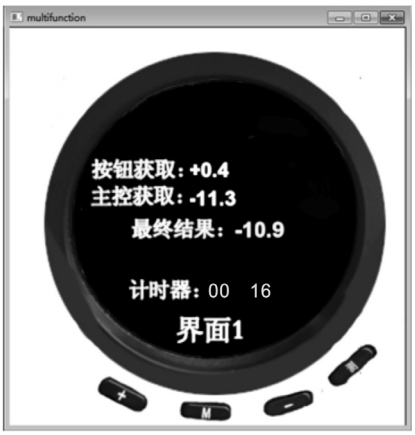

图 5　亮度调节

5　结论

通过对 GL studio 与相关算法的研究，基本实现了目标功能。按钮响应机制、仪表数据处理机制的研究对于目前研发较为先进的多功能表与光电系统具有重要意义。笔者对按钮控件系统与参数处理部分的部分逻辑算法进行了补充，如果在逻辑算法的基础上进行功能性的延伸，可较好丰富目前传统仪表的功能与种类。

参考文献：

[1] 于辉, 赵经成, 付占平, 等. GL Studio 虚拟仪表技术应用与系统开发[M]. 北京: 国防工业出版社, 2010: 5-6.

[2] 刘鲁峰. 基于 GL Studio 的虚拟仪表的仿真研究[J]. 甘肃科学学报, 2015, 27(2): 15-18.

[3] 胡旭, 张宏伟. 基于 GL Studio 虚拟仪表人机交互关键技术研究[J]. 仪表技术, 2019(9): 46-48.

[4] Distributed Simulation Technology Inc.GL studio User’s Guide[Z]. Version, 2006: 20-23.

[5] 周尧. 基于 Gl Studio 的虚拟仪表开发与关键技术研究[J]. 机械工程与自动化, 2015, 193(6): 80-82.

[6] 孙艳丽, 王玲玲, 陈佳琪. 基于 GL Studio 的虚拟仪器仪表与仿真[J]. 系统仿真技术, 2015, 11(2): 61-65.