doi: 10.3969/j.issn.1006-1576.2012.02.023

多线程和 UDP 在等效飞行控制软件中的应用

张培辉，黄一敏
（南京航空航天大学自动化学院，南京 210016）

摘要：针对等效飞行控制软件的特点，提出一种在局域网内基于多线程技术和 UDP 协议的控制模块开发方法。利用多线程技术和 UDP 协议的多点广播通信，设计等效飞行控制软件的总体方案，详细分析多线程控制模块的创建、通信、同步等关键技术。结果表明：该方法能较好地实现等效飞行控制软件的开发，解决等效飞行控制软件的通信问题以及多任务的调度问题。

关键词：UDP；多线程；等效飞行控制软件；等效仿真环境
中图分类号：TP311.5 **文献标志码**：A

Application of Multithread and UDP Communication Protocol in Equivalent Flight Control Spftware

Zhang Peihui, Huang Yimin
(*College of Automation*, *Nanjing University of Aeronautics & Astronautics*, *Nanjing* 210016, *China*)

Abstract: Aiming at the feature of equivalent flight control, put forwards a project of equivalent flight control based on multithread technology and user datagram protocol (UDP) protocol in a LAN. Make use of multithread technology and the broadcast multipoint communication of UDP protocol, the thesis stresses on the overall design of the project, analyzes the key point in the design of multithread control module, including the multithread create, communication and synchronize and so on. The result shows that method can satisfactorily realizes the development of equivalent flight control, and solve the problem of multitask attemper and the communication for equivalent flight control.

Key words: UDP; multithread; equivalent flight control software; equivalent simulation environment

0 引言

无人机等效仿真环境是对半物理实时仿真环境的补充，主要由等效飞行控制软件和等效仿真软件组成。该环境可以在无人机飞行控制系统开发的早期阶段快速地验证飞行控制中导航和控制模块的逻辑和功能。等效仿真软件主要对仿真计算机中的无人机数学模型和传感器进行等效，它与等效飞行控制软件配合使用，两者相辅相成[1]。“等效飞行控制”是一种基于 PC 机的飞行控制软件解决方案。之所以称之为“等效”，是指其中的飞行导航和控制模块代码能够完全移植到目标机（飞行控制计算机）中运行，而无需做任何改动。

在一个复杂的等效飞行控制软件中，往往要求同时处理控制功能模块、辅助模块、显示模块、通信模块等任务，Win32 的多线程能力就适合处理这种并行性任务。Win32 中每个应用程序的执行能力就是一个进程，每个进程都有一个主线程，在主线程中可以创建多个子线程，每个子线程可以独立完成 1 个子任务，这使得一个程序可以同时完成多个任务。等效飞行控制软件可以将分解好的子任务用线程表示，如控制功能模块线程、辅助模块线程、显示模块线程、通信模块线程等。可给每个线程设定一个基于进程的优先级、操作系统根据线程的优先让 CPU 抢先执行当前最适当的线程。开发多线程以及网络通信应用程序，可利用 VC++中提供的 MFC 类库，也可利用 32 位 Windows 环境提供的 Win32 API 接口函数。因此，笔者重点介绍利用 32 位 Windows 环境提供的 Win32 API 接口函数实现多线程调用和处理、进行网络通信编程的方法，并给出多线程技术及 WIL 库函数在等效飞行控制软件中的应用实例。

1 系统设计方案

笔者采用某型无人直升机为样例，设计的等效飞行控制软件主要包括控制功能模块、显示模块、辅助模块和数据通信模块等线程，如图 1。其中控制功能模块是等效飞行控制软件的核心，是等效飞行控制软件开发的主要目的。因此笔者只对这一模块的多线程创建、线程间的通信和同步控制技术作

收稿日期：2011-09-09；**修回日期**：2011-10-10
基金项目：飞行器自主控制技术教育部工程研究中心（南京航空航天大学）资助项目
作者简介：张培辉（1979—），男，浙江人，硕士研究生，工程师，从事无人机飞行控制技术研究。

详细介绍。

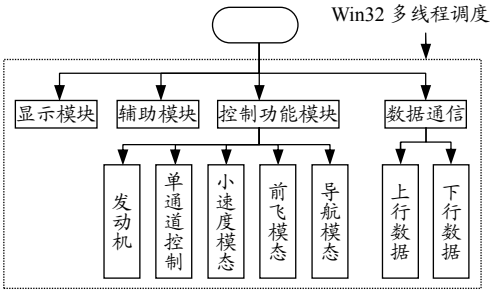

图 1 等效飞行控制软件结构

1.1 基于 UDP 的广播通信

在网络传输中使用了套接字(socket)工具，Windows Socket 有 2 种形式：流式套接字(so ckst ream)和数据报式套接字(sock dgra M)。流式套接字实际上是基于 TCP 协议实现的，提供面向连接、可靠的数据传输服务，数据无差错、无重复地发送，且按发送顺序接收，它通信可靠，对数据有校验和重发的机制，因而会有一定的传输延时；数据报式套接字是基于 UDP 协议实现的，提供无连接服务，支持双向的数据流，不能保证可靠、有序和无重复。能达到较高的通信速率，通信延时小，能够向若干个目标发送数据，接收来自若干个源的数据。工作过程如图 2。

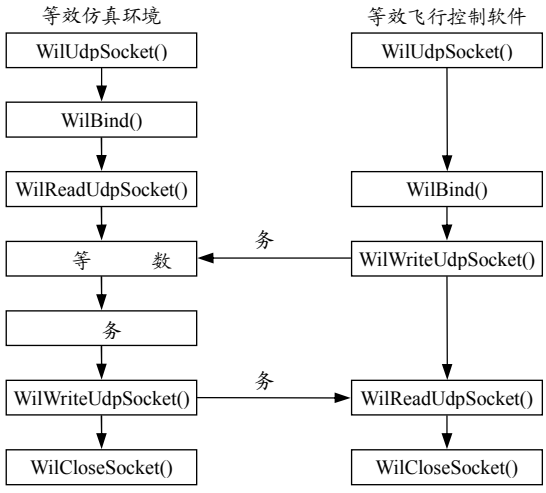

图 2 UDP 通信协议模型

客户机与服务器首先通过 WilUdpSocket()函数创建 1 个套接字，然后通过 WilBind()函数与本地端口绑定，接着通过函数 WilReadUdpSocket()与 WilWriteUdpSocket()进行数据通信。其中客户端 WilBind()函数可选，若包含 WilBind()函数客户端与服务器无明显差别。

考虑到实验室环境下，主要进行一些数据的传输，不涉及文件操作，并且数据量比较小(通常 1 帧数据只有 36 byte，远远小于 UDP 数据报的数据区最大长度——1 472 byte)，因此发生丢包的可能性比较小。因此系统采用了基于 UDP 的数据报式套接字，实现双向、不可靠、低延时的数据传输，满足实验要求。

笔者采用 WIL(winsocket interface library)库函数构建数据报式套接字。因为 WIL 库函数它对标准伯克利函数进行了封装，简化了 Winsock 编程并提供对 DNS，Finger，SMTP，POP3，FTP，NNTP，HTTP 等支持。WIL 库对 Winsock 简化主要体现在以下几方面[2]：

1) 能够根据需要将主机字节顺序自动转换为网络字节顺序(反之亦然)。

2) 能够自动处理和生成 Winsock 数据结构。

3) 根据需要提供一个直接封装接口。

4) 提供功能更多强大多功能的 Winsock 函数。

因此，WIL 库函数在 Windows XP 平台上通信实现简单，实时性高，便于等效仿真环境的通信。

等效飞行控制软件采用广播通信工作方式，实现多播工作方式的多点传送。如图 3，根据等效飞行控制软件的上/下行数据特点，在 UDP 数据报前加标志头，将局域网内计算机分为若干个多播组，当然作为等效飞仿真环境软件，始终只有一台计算机在局域网内接收。当发送消息时，向整个局域网广播，每台计算机都会收到这条消息，根据标志头来判断是否应该接收此消息。从而只有在同一个多播组内的计算机才接收此消息，给用户感觉就像将整个局域网分成了很多个多播组，只有多播组内的计算机才能相互通信。

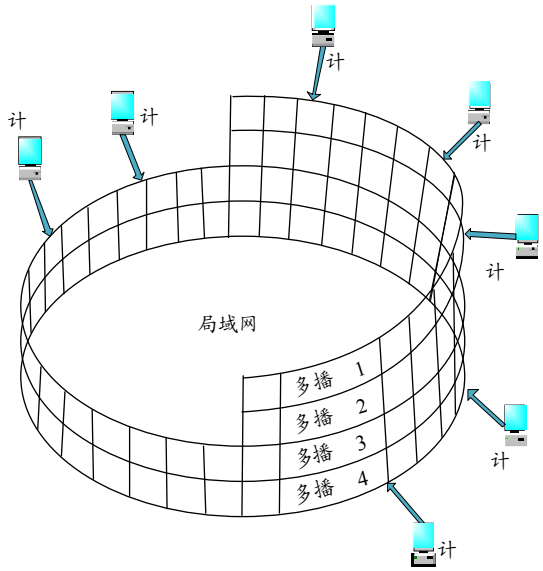

图 3 多播组网模式

实际结果表明：等效飞行控制软件与等效仿真控制台之间通信良好，数据接收、发送和解码都正确，得到了较为满意的结果。

1.2　多线程技术

通过使用多线程的同步执行技术来实现系统的多任务操作。线程被分配成一小段时间片(大约 20 ms)，当一个线程的时间片用完，它转入睡眠状态，等待下一轮有效时间片到达。线程在转入睡眠之前会保存当前上下文环境，以便重新被唤醒来恢复原先的状态。每个在进程中运行的线程都不知道进程中其他线程，除非明确使它们互相可见。任何共享进程资源的线程必须通过通信的方式调度，不正确的调度会引起进程死锁。因此，访问共享资源时必须同步线程执行。

本系统按控制模块、显示模块、发送、接收、数字飞机模型等功能划分，每个部分对实时性的要求都很高，而且是相互联系的。其中控制模块的多线程创建，比较复杂，主要创建的任务函数如表 1 所示。等效飞行控制软件的实现在很大程度上取决于这几个模块里线程之间的协调，也就是线程之间的同步问题。如图 4，具体实现步骤如下：

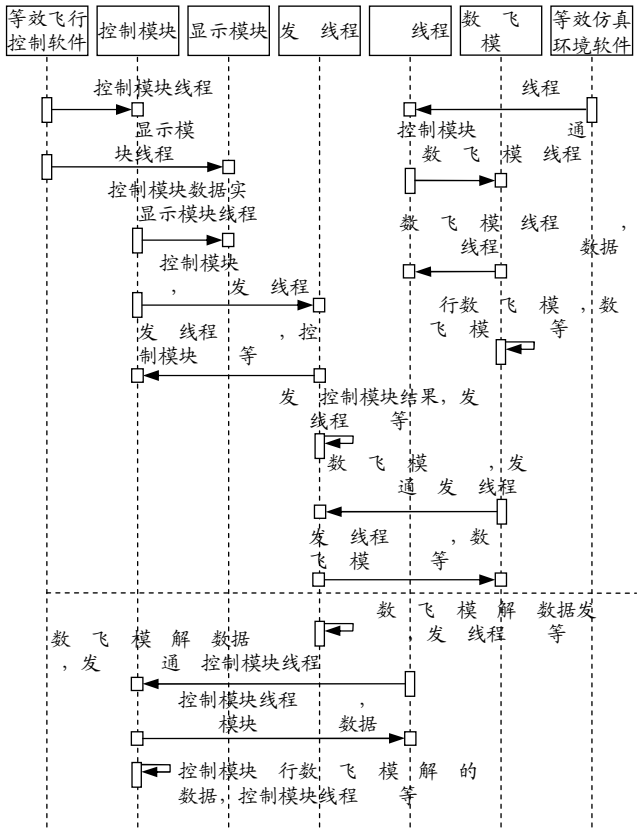

图 4　多线程等效飞行控制软件时序图

1) 控制功能模块的多线程创建

控制功能模块是整个等效飞行控制软件的核心，它的软件架构与机载部分完全相同。控制功能模块由控制律和导航 2 大部分组成，其中控制律主要有控制律的解算和模态任务，导航有起飞、着陆、自主导航、人工导航、在线导航等任务。

用户可通过调用 RTKCreateTask()函数创建线程，向该函数提供线程函数的起始地址和传给线程函数的参数来创建线程。该函数的线程定义格式为：

```
RTKCreateTask(
T, //线程函数的起始地址
P,//传递给线程函数的参数
S,//线程的堆栈的大小(byte)
N//线程的安全属性
)
```

表 1　控制模块的任务函数

数	功能
LAW_Task(void)	控制　解　任务[25 Hz]
task_100ms (void)	控任务
Console_Task(void *pdata)	控制　显示任务
LONG_TakeoffPre(void)	建　飞　任务
LONG_Takeoff(void)	建　模
LONG_WayLand(void)	模
LONG_CmdClimb(void)	模
LONG_CmdLevel(void)	飞模
LONG_CmdDive(void)	模
LONG_WayClimb(void)	模
LONG_WayLevel(void)	飞模
LONG_WayDive(void)	模
LONG_WayPass (void)	通　模
LATE_TurnLeft(void)	模
LATE_Straight(void)	飞模
LATE_TurnRight(void)	模
LATE_TrackPsi(void)	模
LATE_TrackWay(void)	模
GUID_Way(void)	线飞行
GUID_Enter(void)	点
GUID_Hover(void)	飞行
GUID_Home(void)	飞行
ONE_Enter(void)	在线点
ONE_Way(void)	在线　线飞行
TAXI_RespCmdTaxi(void)	行　任务
TAXI_RespCmdBrake(void)	制　任务
TAXI_RespCmdTakeoff(void)	飞　任务
TAXI_RespNavLand(void)	地　任务

2) 控制功能模块的线程间通信

在控制功能模块运行中，任务与任务之间，任务与中断服务程序之间需要传递数据为对方所用，就必然伴随着数据的通信。如表 1 中任务函数，即

线程。线程之间的通信有全局变量方式、消息传递方式、参数传递方式和线程同步方式等 4 种方式。根据飞行控制软件本身的特点，这里采用全局变量的方式。每个物理量都对应一个全局变量，所有任务间的数据通信都是通过对该变量操作进行的。如传感器任务变量由传感器模块进行“写”操作，而控制解算模块、自主导航模块和遥控遥测模块则通过“读”操作使该变量为自己所用，如图 5 所示。

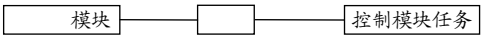

图 5 数据通信示意图

这种任务间通信的方式，其最大特点是考虑了飞行控制软件的特点，变量仅有一个“写”源而可以存在多个“读”源，能最大程度简化数据通信的复杂性，使控制软件可以把尽可能多的时间用于等效飞行控制软件的实时解算上。

3) 控制模块线程间的同步控制

在抢先式多任务系统中，系统通过上下文切换来强迫线程控制权的转移，也因此 2 个线程之间的执行次序变得不可预期。这种不可预期性便造成了所谓的条件竞争(RaceConditions)。其中的一种典型问题即是死锁问题(DeadLock)。避免条件竞争的几种同步方法为：临界段、互斥量、事件和信号量。

笔者选择事件触发任务，触发事件在飞行管理任务进行令牌赋值，从而实现对无人机的横向模态和纵向模态控制。由于控制模态任务机构复杂，判断条件较多，如果采用常用的 if 语句设计模态任务，则内部嵌套 if 语句层次就越多，直接导致模态任务结构庞大，不易设计。C 语言提供了专门用于多分支机构的 switch 语句的条件选择结构，使用 switch 结构清晰，便于设计控制模态任务，模态任务无需多重 if 嵌套，避免了模态控制任务因 if 嵌套而可能带来死循环。因此，笔者在各个模态控制任务都采用 switch 语句结构设计。控制模态任务的通用结构如下：

```
While(1){
    if(token=触发令牌){
      switch(step){
            case 0:控制执行语句 1；step++;
                   break;
            case1: 控制执行语句 2；step++;
                   break;
                   … …
            casen: 控制执行语句 2；step++;
                   break;
            default: 控制执行语句 n+1；
                   break;
            }
        }
    else {step=0;OSTimeDly();}
    }
```

2 系统软件设计实现

2.1 数据报套接字设计实现

1) 发送端

在发送端需设定发送地址，而后通过 WilWriteUdpSocket()函数将发送缓冲区中的数据发送指定目的地址。主要实现如下：

```
/*设置发送端口*/
#define send_PORT 7777
/*设置接收端口*/
#define rece_PORT 8888
void UDPInit()
{
/*初始化 WIL 库*/
 wilAttach();
/*  得到主机 IP 地址*/
 RemoteIP = wilGetHostAddr(AddrPtr,0);
/*  创建 UDP 套接字  */
 Socket = wilUdpSocket();
/*  绑定接收端口和本机地址到套接字  */
wilBind(Socket,0,rece_PORT);
}
/*发送函数*/
WilWriteUdpSocket(Socket,Buf,sizeof(Buf),RemoteIp,send_PORT);
```

2) 接收端

通过 WilReadUdpSocket()函数对发送端发送的数据进行接收。接收端定义两个地址：一个本机地址用来绑定套接字，一个地址用来从网络上的发送地址接收消息。接收端的具体函数实现为：

```
#define send_PORT 8888     /*设置发送端口*/
#define rece_PORT 7777    /*设置接收端口*/
void UDPInit()
{
#define MAX_BUF 1024
static char Temp[MAX_BUF+8];
static char AddrPtr[MAX_BUF+8];
/*初始化 WIL 库*/
wilAttach();
/*  获取主机上 IP 地址  */
```

```
    wilGetMyHostDotted(0,(LPSTR)AddrPtr,
MAX_BUF);
    /* 创建 UDP 套接字 */
    Socket = wilUdpSocket();
    /* 绑定接收端口和本机地址到套接字 */
    wilBind(Socket,0,rece_PORT);
    _beginthread(DlinkRx,4096,NULL);
    }
    /*接收函数*/
WilReadUdpSocket(Socket,(LPSTR)LINK,sizeof(LIN
K),RemoteIp,NULL);
```

对于多线程客户应用程序可设计 2 条线程：主线程和辅助线程。主线程用于处理主窗口的消息映射，辅助线程用于数据传输。在程序适当的位置，用 RTKCreateTask()来创建线程。该函数的线程定义格式为：

```
RTKCreateTask(
T, //线程函数的起始地址
P,//传递给线程函数的参数
S,//线程的堆栈的大小(字节)
N//线程的安全属性
)
```

2.2　控制模块设计实现

2.2.1　控制律模块

根据控制律模块功能将控制律模块划分为各个模态任务和控制解算任务。模态各个任务给出各个姿态的给定量，控制律解算任务获取各个舵面的值，并通过 DA 信号控制舵机。文中的发动机控制是开环控制，各个模态任务直接控制发动机油门。

1) 模态任务

笔者以控制模态的自主爬升任务为例，说明模态任务软件具体实现过程。自主爬升任务首先获得自主爬升令牌，进入自主爬升任务；无人机断开高度控制，发动机油门处于大车状态，同时给出舵机的俯仰角指令，无人机进入爬升状态；当接近指定爬升高度时，发动机的油门置巡航状态，同时给平飞俯仰角指令，无人机进入爬升指定高度，接入高度控制，清除自主爬升令牌，结束自主爬升任务，让无人机纵向保持在此高度飞行，具体流程如图 6。

2) 控制律解算任务

控制律模块是无人机飞行控制系统的核心部分。根据不同的飞行阶段确定不同的控制策略，控制律设计分为纵向控制律设计和横侧向控制律。其任务是获取传感器融合之后的数据，经过各个舵面的控制律解算，升降舵进行纵向控制，副翼舵和方向舵进行横侧向控制，最后输出舵控信息，该过程在无人机飞行过程中以 25 Hz 的频率持续刷新，保证无人机的实时控制[3-4]。该任务的流程图如图 7。

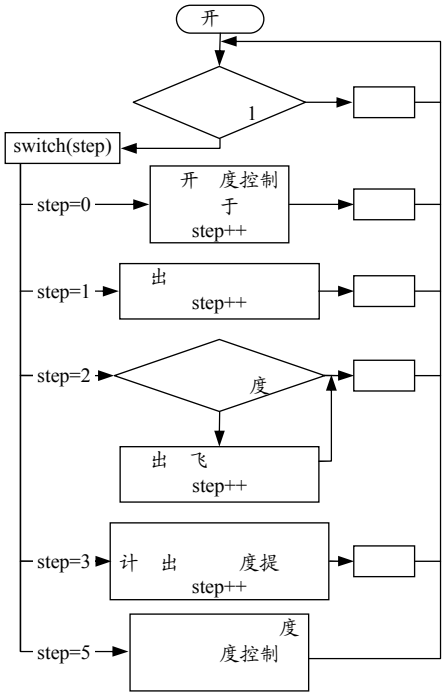

图 6　自主爬升软件流程图

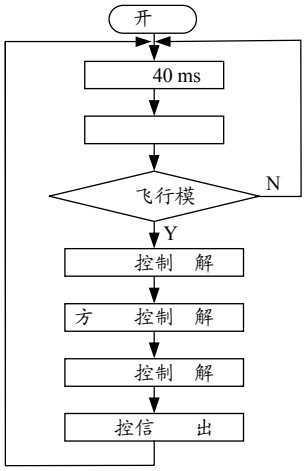

图 7　控制律解算任务流程图

2.2.2　导航模块

笔者以某型号无人机自主起飞着陆为例，其导航方式有自主导航、人工导航和在线导航 3 种方式。主要任务有起飞任务、着陆任务、人工导航、自主导航和在线导航，下面将对各个任务进行详细设计。

1) 起飞任务

起飞任务分为地面三轮滑跑、地面两轮滑跑和离地爬升 3 个阶段[5]。地面三轮滑跑段发动机处于最大推力状态，主轮刹车系统和方向舵进行纠偏控

制，当无人机滑行速度达到抬前轮速度时，进入地面两轮滑跑阶段，给出爬升姿态角，产生抬头力矩，无人机自动离地进入离地爬升段；离地爬升段并未进入自主导航模式，保持定航向飞行；当飞行高度达到一定高度时，进入自主导航飞行阶段，无人机起飞任务结束[2]。起飞任务逻辑流程如图 8。

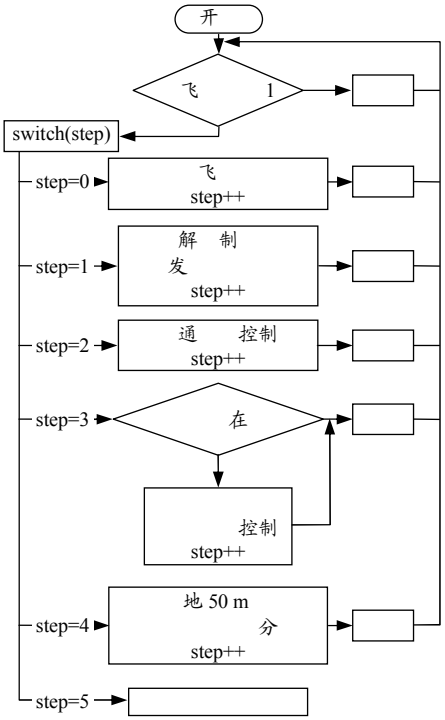

图 8 起飞任务逻辑图

2) 自主导航

无人机设定航线是由多个航段组成，而航段是由 2 个航点构成，航点信息有点号、经纬度、高度以及任务特征字，航点信息在初始化时从飞行控制计算机的内存 Flash 中读取。无人机在完成起飞任务后，进入自主导航，需要根据无人机当前位置和航路信息计算侧偏距、待飞距、航向角和转弯距离。文中的待飞距是指无人机当前位置到下一航线的垂线距离，侧偏距是指当前位置到当前航线的垂直距离，侧偏在航线的右侧为正左侧为负。而航线信息是球面坐标的经纬度，这样就需要将纬度转化成为平面直角坐标系下的坐标，进一步计算出导航信息需要的待飞距和侧偏距。首先可采用高斯公式转化坐标：

$$X = A_x \cos\overline{\varphi} - B_x \cos 3\overline{\varphi}(\lambda - \lambda_0) \tag{1}$$

$$Y = A_y - B_y \cos 2\overline{\varphi}(\varphi - \varphi_0) \tag{2}$$

其中：$\overline{\varphi} = (\varphi + \varphi_0)/2$，$A_x$=6 383 487.606，$B_x$=5 357.31，$A_y$=6 367 449.134，$B_y$=320 770。$(\lambda_0, \varphi_0)$ 为坐标原点处的经纬度；(λ, φ) 为飞机当前位置的经纬度。经高斯公式解算得到的坐标(X,Y)，(X,Y)是东北坐标系下坐标值。

航点 1 和航点 2 组成的航线方程可表示为：$AX + BY + C = 0$，航点 1 的坐标为(X_1, Y_1)，航点 2 坐标为(X_2, Y_2)，如图 9。根据点到直线间的距离公式，可计算得到待飞距和侧偏距，公式如下：

$$\mathrm{d}X = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \tag{3}$$

$$\mathrm{d}Y = \frac{X_1 Y_2 - X_2 Y}{\sqrt{X_2^2 + Y_2^2}} \tag{4}$$

而飞机的转弯时需要转弯提前量 D，转弯提前量半径 D 依据两段航路之间的夹角、最大滚转角及当前的飞行速度而计算获得，可由式 (5) 计算得出转弯半径。

$$D = \frac{v^2 \times \tan(\frac{\varphi}{2})}{g \times \tan\gamma} \tag{5}$$

其中：v 为无人机当前飞行的地度；g 为重力加速度；$\Delta\varphi$ 为当前航段与下一航段的之间夹角，范围在 ±180°之间。

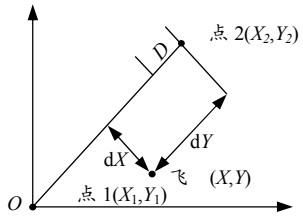

图 9 东北坐标系图

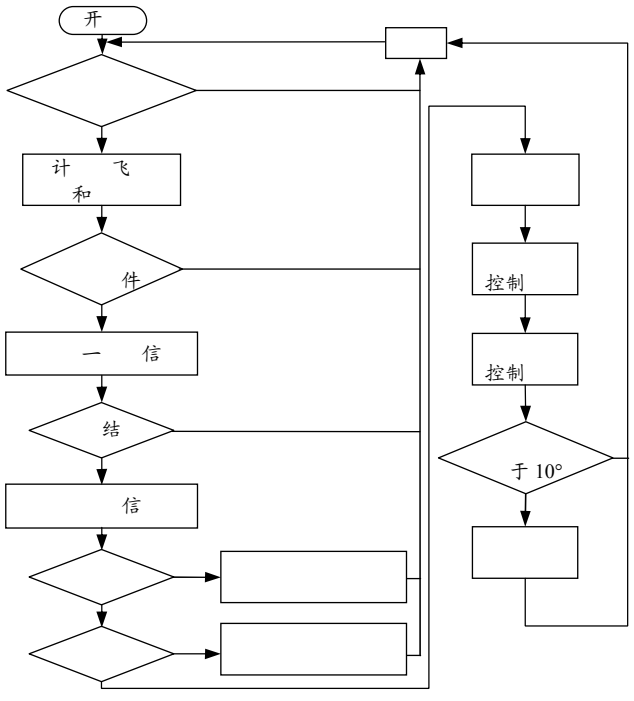

图 10 自主飞行管理任务流程图

无人机沿航路飞行时控制律模块进行纵向和横侧向控制，当发生航段切换时，根据转弯半径判断是否需要进行航段切换，当待飞距小于转弯半径时，无人机完成从第 n−1 个航点和第 n 个航点确定的航段，自动切换到第 n 个航点和第 n+1 个航点所确定的航段，从而使无人机按照规划航线自主飞行。自主飞行管理任务具体软件流程逻辑如图 10。

3) 人工导航

各个指令模态任务和模态任务实现方式相同，不再赘述。以点号切入任务为例说明人工导航实现，点号切入任务是指无人机沿航线飞行时，需要从一航段切换到另外一航段飞行的过程。从遥控指令获得点号切入的令牌和需要切入的航段号；计算当前无人机位置到指定航段的待飞距和航向，将无人机纵向模态置为平飞，横侧向置为按航向飞行；判断待飞距是否满足无人机转弯条件，若满足进行航段切换，不满足条件则无人机继续飞行直到满足转弯条件进入航段的切换，具体软件流程如图 11。

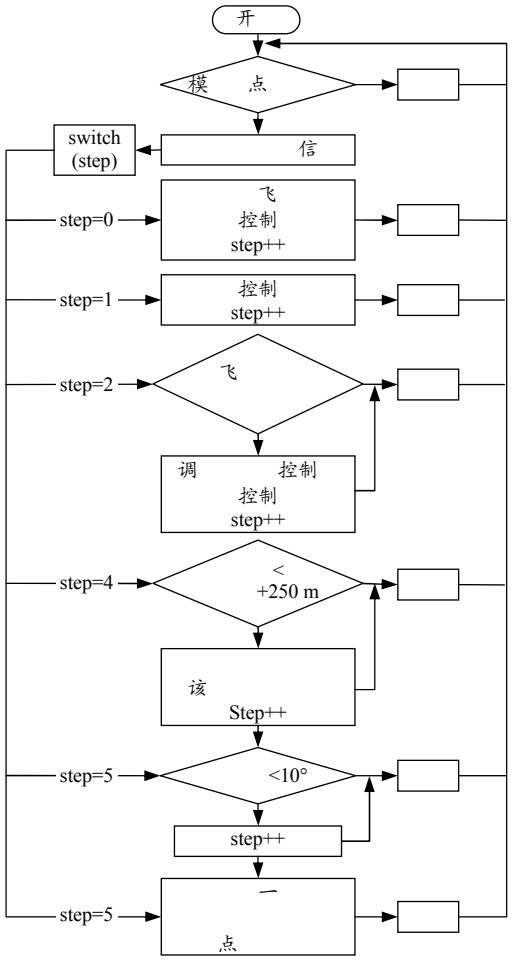

图 11　点号切入任务流程图

4) 在线导航

在线航线管理任务是在无人机正常飞行中，临时改变飞行航路，让无人机按照临时航路飞行过程。无人机完成在线航导航需要在线点号切入任务和在线航线管理飞行的任务。临时航线的生成是由地面站的遥控遥测软件中人工规划的，生成的航路信有航线航段号、航段长度、方位角、航段高度、任务特征信息，通过无线电通信上传给无人机飞行控制计算机，在线航线的流程如图 12。

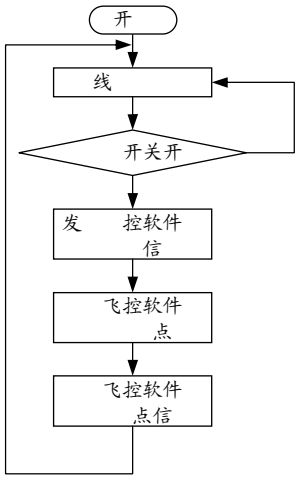

图 12　在线航线装订流程图

依据式 (3)～(4) 将以上信息转成航点信息以共享数据结构存储，无人机从当前点生成的航路中进行自主飞行，也可进行点号切入。这样就需要设计在线航路管理任务和在线点号切入，他们与自主飞行的航路管理和点号切入任务设计思路是相同的。

5) 着陆任务

无人机着陆是飞行控制系统的一个关键阶段，无人机着陆过程分为进场平飞段、下滑引导段、末端拉起段和地面滑跑减速段 4 个阶段，着陆下滑轨迹线如图 13。

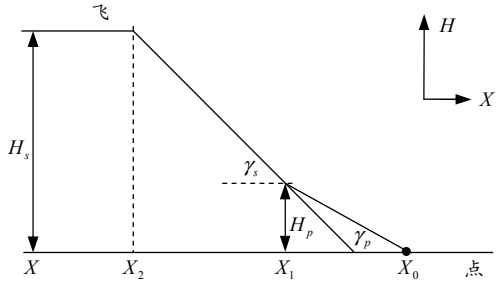

图 13　进场着陆下滑轨迹线图

其中：γ_s 为下滑引导段轨迹倾角；γ_p 为拉起轨迹倾角；H_p 为参考拉起高度；H_s 为参考平飞进场高度；X 为飞机当前位置；X_0 为着陆点位置；X_1 为末端拉起段开始位置；X_2 为下滑引导段开始位置。

无人机着陆任务读取自主飞行航线的航点标志位判断是否进入着陆阶段，如果着陆标志为 1，则开始进场着陆，或者人为通过着陆指令让无人机进场着陆。首先从 X_1 进入进场平飞段。此阶段无人机以巡航状态飞行进行调整姿态，为进入下滑引导段做准备，并放下起落架避免在引导段放下产生的附加作用力；到达下滑点 X_2 时，无人机开始沿指定轨迹线下滑，发动机收至慢车，将无人机引导至机场跑到上空，为末端拉起段做准备，此阶段要实现高精度的轨迹跟踪和高度控制；当高度小于 H_p 时无人机进入末端拉起段，为了准确对准跑道横侧向通过航迹控制，同时需要控制无人机的着陆速度；触地后进入地面滑跑减速段，横侧向控制侧偏距以确保沿跑道中心线滑行；纵向避免再次升空，升降舵给出一定的角度以压住机头，使刹车系统能够控制住，并将发动机停车，具体软件流程如图 14。

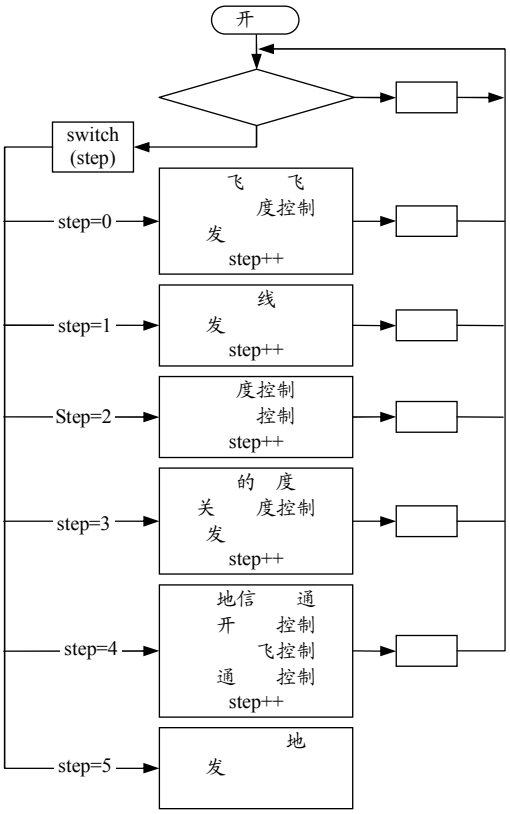

图 14　着陆任务流程图

2.2.3　辅助模块

辅助模块是 100 ms 定时任务，定时处理控制律软化、数据融合、通信链路监测以及数据记录。数据记录是实时记录等效飞行控制软件仿真时，将重要的数据保存到 Windows XP 的.Text 文件中，便于仿真后的数据分析，辅助任务流程如图 15。

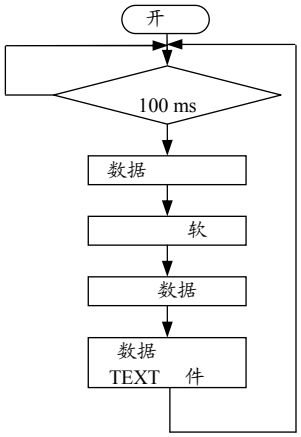

图 15　定时任务流程图

3　结束语

笔者基于 μC/OS-II，在 Windows XP 下开发了等效飞行控制软件，搭建了 μC/OS-II 实时操作系统在 Windows XP 下运行的环境，利用多线程技术与 UDP 协议解决了等效飞行控制软件的通信问题以及多任务的调度问题。下一步，将该软件完全等效到机载飞行控制软件中，以最终完成机载飞行控制软件的设计开发。

参考文献：

[1] . 基于 μCOS-II 的 飞行控制 软件设计[D]. : , 2008.
[2] Winsock Interface LibraryReference Manual[S], MarshallSoft Computing, Inc.
[3] , . 在线 技术 及 程实现[J]. , 2011, 31(3): 14.
[4] . 基于 MPC555 和 μCOS-II 的 飞行控制软件开发技术 [D]. : , 2010.
[5] 张 . / 度 的飞行控制 设计技术 [D]. : , 2009.

****

（上接第 81 页）

参考文献：

[1] . [M]. : 通 出 , 1995: 96-99.
[2] Bean Banerjee. PLL Perfomance,simulation and Design Handbook[M]. 4th Edition 2006: 20-24.
[3] Lascari, Lance. Accurate Phase Noise Prediction in PLL Synthesizers[J]. Applied Microwave and Wireless, 2000, 12(5): 20-40.
[4] National Semiconductor Date Sheet[M]. Edtion 2000.