doi: 10.7690/bgzdh.2020.04.014

联机模拟飞行通信关键技术

刘剑超[1]，林亚军[1]，王　伟[1]，杨　健[2]
（1. 海军航空大学教练机模拟训练中心，辽宁 葫芦岛 125001；2. 中国人民解放军 31698 部队，辽宁 锦州 121000）

摘要：为解决联机模拟飞行通信模块中的技术难点，对联机模拟飞行通信系统中的关键技术进行研究。基于 UDP 数据传输协议，通过对时间进行算法处理，采用共享内存与信号量相结合的方式，对多飞机同步算法进行对比分析，设计出最优算法，消除了网络延时和数据丢包的影响，并采用三次样条插值验证了插值算法的有效性。验证结果表明，采用该设计方法可以大幅提升联机模拟飞行时数据传输的可靠性。

关键词：算法；数据；多飞机同步；通信协议
中图分类号：TP391　　**文献标志码：**A

Communication Key Technology of Online Analog Flight

Liu Jianchao[1], Lin Yajun[1], Wang Wei[1], Yang Jian[2]
(1. *Flight Simulation Training Center of Training Aircraft*, *Navy Aviation University*, *Huludao* 125001, *China*;
2. *No*. 31698 *Unit of PLA*, *Jinzhou* 121000, *China*)

Abstract: In order to solve the technical difficulties in the online simulation flight communication module, the key technologies in the online simulation flight communication system are studied. Based on the UDP data transmission protocol, the algorithm is processed by time, and the shared memory and semaphore are combined to compare and analyze the multi-aircraft synchronization algorithm, and the optimal algorithm is designed to eliminate the influence of network delay and data loss. The effectiveness of the interpolation algorithm is verified by cubic spline interpolation. The verification results show that the design method can greatly improve the reliability of data transmission during online flight simulation.

Keywords: algorithm; data; multiple plane synchronization; communication protocol

0　引言

飞行模拟训练具有经济性、安全性等诸多优点。无论是在军事训练还是在民航飞行员的培养上，飞行模拟器都发挥着越来越重要的作用。在飞行模拟器设计过程中，为了能够更逼真地复现飞行环境及操控感觉，各生产单位在新机研发、飞行数据包、控制系统、视景系统等方面做着积极的努力[1]。笔者在综合分析国内外飞机模拟器联网模拟飞行的基础上，以满足飞行模拟器仿真任务运行的实时性要求，提高飞行模拟训练的流畅性为目的，对联机模拟飞行通信关键技术进行研究，设计了单机模拟器通信协议，提出一种最优的多飞机同步算法，并成功应用到了模拟器上。

1　通信协议

笔者采用了客户端/服务器架构，其中，服务器对应的为飞行控制系统软件中的通信模块，客户端为视景仿真系统中的通信模块，以下均简称为服务器与客户端。采用基于 UDP 的 socket 实现协议的传输，传输流程如图 1[2-4]。

1) 服务器端进程依次进行操作如下：

① 创建一个套接字；

② 绑定本地地址到创建的套接字上；

③ 从飞行仿真模块取得飞行数据，并向指定的客户端地址发送；

④ 进入休眠，等待指定时间的到来；

⑤ 如果进程退出，则进入⑥；否，则等待到达指定时间，进入③；

⑥ 服务器端关闭，注销该套接字。

2) 客户进程依次进行如下操作：

① 与服务器类似，创建一个套接字；

② 从服务器读取飞行数据；

③ 处理飞行数据，主要包括坐标系转换、单位转换等；

④ 将处理后的数据提交给视景仿真系统，继续接收下一条飞行数据至程序结束。

收稿日期：2019-12-05；**修回日期：**2020-01-19
作者简介：刘剑超(1982—)，男，河北人，硕士，工程师，从事军事仿真建模、舰载机理论研究。E-mail: liujianchao82422@163.com。

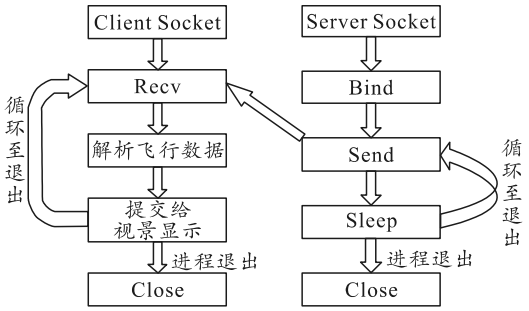

图 1 异步通信框架

系统在使用 UDP 进行数据发送时，使用固定时间间隔。为了保证视景系统显示 60 帧的效果，在不进行额外数据处理的情况下，要保证发送时间间隔小于 17 ms。为保证在发送数据的时候固定时间间隔，单纯地使用系统提供的 sleep()函数是不行的，因为仿真计算飞机飞行状态也需要时间，可能导致发送时间产生偏差[5]；因此，需要统计更新的飞行状态所用时间，用固定的发送时间间隔减去更新飞行状态所花费时间，得到实际的休眠时间。整个算法的伪代码如下：

```
While True：{
Start  = currentTime;//统计时间开始
process_();//更新飞机飞行状态函数
End = currentTime;//统计时间结束
sleepTime=targetInterval–(End-Start);// 计 算 休眠时间
if(sleepTime > 0)//如果需要休眠
 {
  highResSleep  (sleepTime);//使用高精度休眠函数休眠
 }
 }
```

通过上述算法处理，可以保证飞行控制系统软件以固定时间间隔向视景系统发送飞行数据。这样发送数据的速率均匀，但在使用过程中，视景系统出现了抖动的情况。

2 共享内存

在使用异步通信框架的时候，会出现视景抖动的问题。在实际使用的情况中，飞行控制系统软件和视景系统可以安装在同一台机器上。这种情况下，飞行控制系统软件和视景系统之间的通信就转化为 2 个进程之间的通信[6-7]。

共享内存是效率最高的进程间通信方式，为了保证访问数据的唯一性，往往要配合信号量来实现；因此，采用共享内存与信号量相结合的方式来实现飞行控制系统软件与视景系统之间的通信。整个系统的框架如图 2。

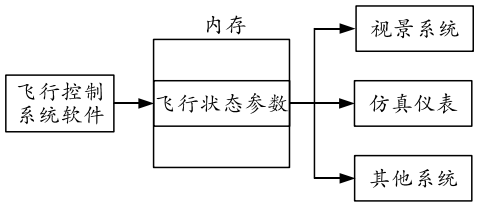

图 2 共享内存框架

通过共享内存，视景系统在需要的时候可以随时读取飞行数据，不会出现等待数据以及重复使用同一数据的情况，最大程度地减少了抖动的发生。

3 多飞机同步算法

互联网存在延时不可避免，完全的飞机状态同步情况并不存在，但是笔者可以通过隐藏延时，给用户带来几乎实时的交互体验。为更有效地实现同步算法，设置通信参数如表 1。

表 1 通信参数

参数	参数含义
timestamp	时间戳，用来同步
x/m	飞机在全局坐标系下 *x* 坐标
y/m	飞机在全局坐标系下 *y* 坐标
z/m	飞机在全局坐标系下 *z* 坐标
v_x/(m/s)	飞机在全局坐标系沿 *x* 坐标的速度
v_y/(m/s)	飞机在全局坐标系沿 *y* 坐标的速度
v_z/(m/s)	飞机在全局坐标系沿 *z* 坐标的速度
a_x/(m/s)	飞机在全局坐标系沿 *x* 坐标的加速度
a_y/(m/s)	飞机在全局坐标系沿 *y* 坐标的加速度
a_z/(m/s)	飞机在全局坐标系沿 *z* 坐标的加速度
Pitch/(°)	飞机的俯仰角
heading/(°)	飞机的偏航角
roll/(°)	飞机的滚转角
*v*_pitch/((°)/s)	飞机的俯仰角速度
*v*_heading/((°)/s)	飞机的偏航角速度
*v*_roll/((°)/s)	飞机的滚转角速度
latitude/(°)	飞机所在纬度
longitude/(°)	飞机所在经度

从理论上讲，通常采用的航位推测算法和影子跟随算法都属于预测算法，是在预测的基础上进行修正，保证飞机运行轨迹的平滑性。与它们不同的是，插值同步算法不是通过预测的方式，而是利用已有的数据进行插值来显示飞机的位置。

3.1 拉格朗日插值算法

拉格朗日插值，对于某个多项式函数，已知给定的取值点，如

$$(x_0,y_0),\cdots,(x_k,y_k)。\tag{1}$$

其中：x_j 对应着自变量的位置；而 y_j 对应着函数在

这个位置的取值。

假设任意 2 个 x_j 都互不相同，那么应用拉格朗日插值公式所得到的拉格朗日插值多项式：

$$L(x)=\sum_{j=0}^{k} y_j l_j(x)。\tag{2}$$

其中，每个 $l_j(x)$为拉格朗日基本多项式(或称插值基函数)，其表达式为

$$l_j(x)=\prod_{i=0,i\neq j}^{k}\frac{x-x_i}{x_j-x_i}。\tag{3}$$

拉格朗日基本多项式 $l_j(x)$的特点是在 x_j 上取值为 1，在其他的点 $x_i(i\neq j)$上取值为 0。

由于拉格朗日插值当插值点增加或减少一个时，所对应的基本多项式就需要全部重新计算，于是整个公式都会变化，非常复杂，而飞机飞行过程中，插值点的个数会频繁变化；因此，没有采用拉格朗日插值算法[8]。

3.2　线性插值算法

线性插值算法是最基本和最简单的算法，指使用连接 2 个已知量的直线来确定在这 2 个已知量之间的一个未知量的值的方法，如

$$\text{newPos}=(1-t)\text{start}+t\bullet\text{end}。\tag{4}$$

其中：newPos 为所求结果，即 t 时刻所在位置；start 和 end 为物体在 t_0 和 t_1 时刻所在的位置。

3.3　三次样条插值函数

三次样条插值，函数 $S(x)\in[a,b]$，且在每个小区间$[x_j+x_{j+1}]$上是三次多项式，其中 $a=x_0<x_1<\cdots<x_n=b$ 是给定节点，则称 $S(x)$是节点 $x_0,x_1,\cdots,x_n$ 上的三次样条函数。

若在节点 x_j 上给定函数值

$$y_j=f(x_j),(j=0,1,\cdots,n)。\tag{5}$$

并成立

$$S(x_j)=y_j,(j=0,1,\cdots,n)，\tag{6}$$

则称 $S(x)$为三次样条插值函数。

3.4　算法分析

在 3.1 中已经排除拉格朗日插值算法，笔者仅对线性插值算法和三次样条函数进行对比分析，设计出最优算法。

笔者使用的局部坐标系定义如下：以飞机起点位置为原点，正东方向为 Y 轴正方向，正北方向为 X 轴正方向，Z 轴垂直于 XY 平面，方向向上。假设模拟客户端 C1 的飞行原始数据如图 3。

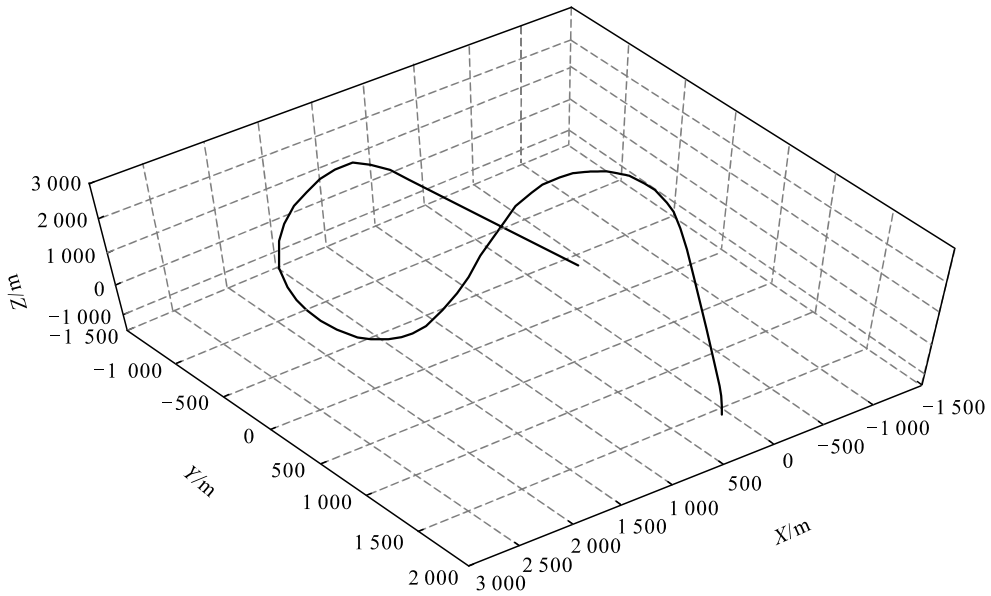

图 3　模拟客户端 C1 飞行原始数据

模拟服务器端按照 30 帧/s 的速度发送数据，客户端 C2 所接收到的部分位置数据如图 4。

如表 2，笔者分别使用线性插值和三次样条插值对接收到的数据进行插值，并使用插值后得到的数据与原始数据中该时间所对应的位置作对比。

由表可见：针对本系统，三次样条插值无论是最大误差、最小误差，还是平均误差，均优于线性插值，且线性插值也满足精度要求；因此，在对轨迹精度要求较高的情况下，可以使用三次样条插值，其他情况可以使用线性插值。

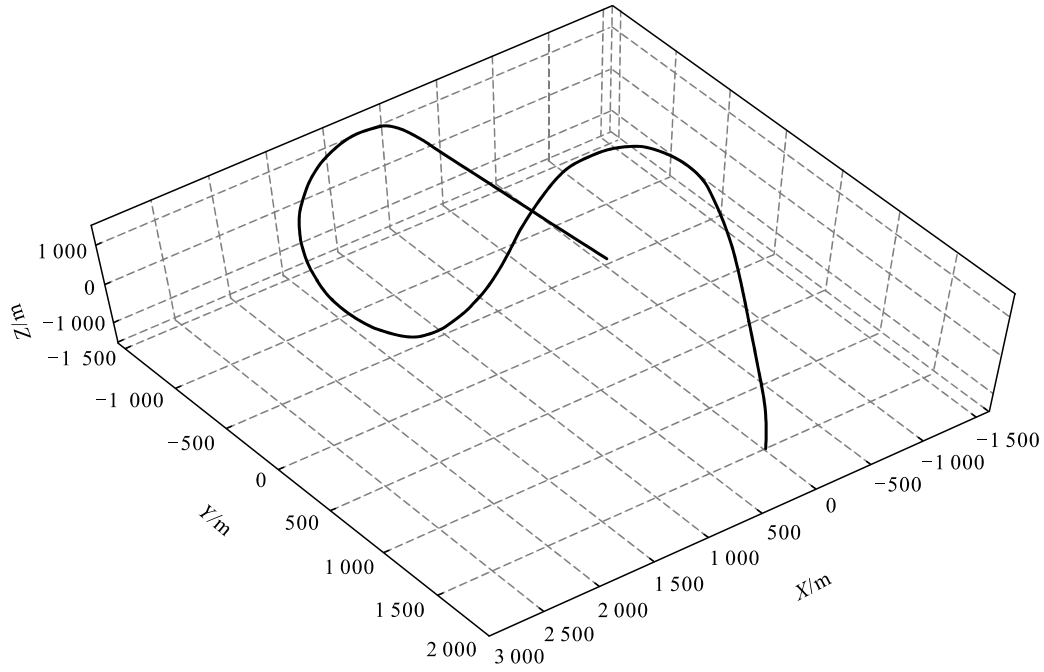

图 4 客户端 C2 接收到的位置数据

表 2 线性插值与三次样条插值对比

方法名称	误差最大值	误差最小值	平均误差
线性插值	1.387 787 427 59	$8.068\ 236\ 270\ 43\times10^{-9}$	0.069 002 358 348 40
三次样条插值	1.383 566 367 55	$7.180\ 311\ 233\ 28\times10^{-9}$	0.003 871 887 338 33

3.5 算法优化设计

有许多情况，比如客户端飞行的方向和速度是瞬间改变，航位是没法处理的，只能在随后的过程中进行修改。面对这种情况，可以使用插值算法。该算法的核心思想是，牺牲掉一帧的实时性，保证自己的状态是当前的，而看到的其他联机飞行状态是使用历史状态进行插值得到的。求当前时间看到飞机(记为 currTime)的位置转化成了求

$$\text{targetTime} = \text{currTime} - \Delta t$$

时刻的位置，其中 Δt 为估算的网络延时，考虑到可扩展性，插值处理的过程如下：

1) 将接收到的数据放入队列 list。记队列的第 i 项为 list[i]，并记下接收时间为 list[i].time。

2) 计算 targetTime 的位置时，从队尾遍历 list，找到第 1 个满足 targetTime ≥ list[i].time 的项，并记录该项序号为 j。如果为队尾(即 list 最后一项)，前往 3)，否则前往 4)。

3) 直接使用队尾项坐标，即新位置为 list[j]的位置。

4) 选择相应的插值算法求解 targetTime 时飞机的位置，如：

$$\text{newPos} = \text{interp}(\text{list}[0\cdots j+1], \text{targetTime}, \text{interpMethod}) \quad (7)$$

其中：newPos 为所求的坐标位置；list[0⋯j+1]为 list 中第 0 项到第 j+1 项；int erpMethod 为所选择的插值算法的具体实现。这样就可以根据不同的需求，使用不同的插值算法求值[9]。

3.6 算法验证

笔者采用三次样条插值来验证插值算法对于飞行轨迹同步的有效性。实验过程中有 2 个客户端 C1 及 C2，在客户端 C1 观察 C2 的位置。使用图 3 中的数据作为 C2 的原始数据，同时，服务器仍旧以 30 帧/s 的速度进行数据的收集与广播。因插值数据较为密集，图 5 中对插值数据进行了采样，并与原始数据进行对比。其中，x 表示插值数据，⋯表示原始数据，坐标为局部坐标系下的坐标[10-11]。

由图可知：从 C1 观察 C2 轨迹基本与原始数据轨迹相吻合，同时，利用 C1 插值得到的 C2 数据，计算其速度曲线如图 6。从图中可以看出：C2 的速度曲线是平滑的，因此，算法的结果可以接受，但是每一帧所看到的是 C2 过去的位置。

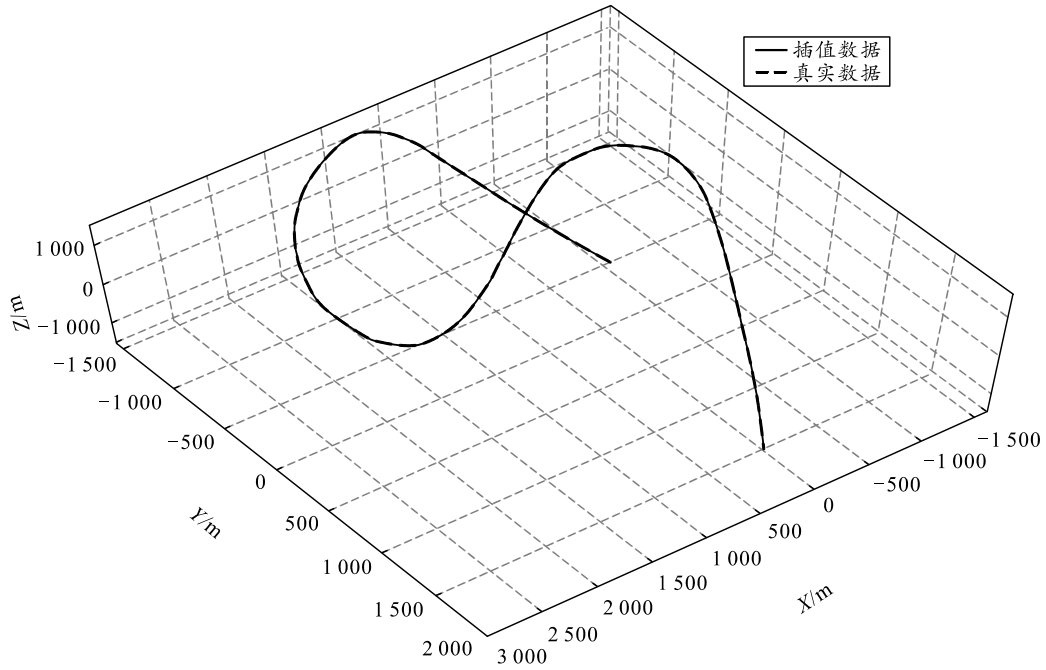

图 5　C1 插值数据与 C2 真实数据对比

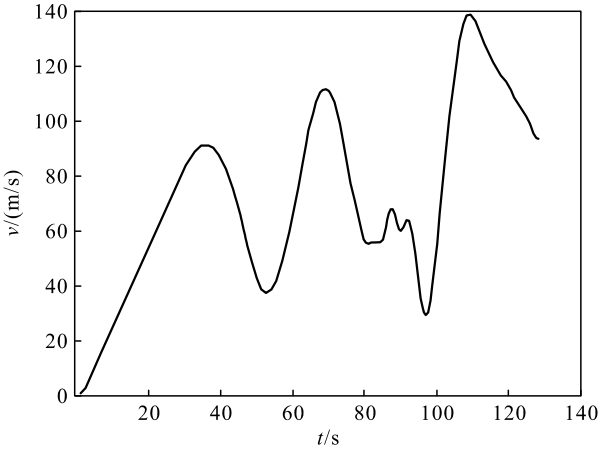

图 6　C2 速度曲线

4　结论

笔者基于 RUDP 网络协议，通过优化通信模块设计，采用线性插值算法和三条样插值算法相结合的优化算法，把飞行控制系统软件的服务端当作服务器，每一台飞行模拟器当作客户端。整个过程就是每一帧从客户端发送数据到服务器，并且每一帧服务器处理来自每一个客户端的数据，更新数据信息，然后把更新后的结果返回给客户端进行显示，成功消除了互联网下数据丢包及网络延时对于联机飞行的影响。经过验证，该方法是可行的。

参考文献：

[1] 郑书朋. 飞行模拟器的计算机系统实时调度与通信关键技术研究[D]. 哈尔滨: 哈尔滨工业大学, 2011.

[2] 邱竟峰. 模拟训练系统网络通信控制技术研究[D]. 沈阳: 沈阳理工大学, 2017.

[3] 邓晴莺, 李国罿, 王宝奇, 等. 某飞行模拟器视景系统的设计与实现[J]. 兵工自动化, 2016, 35(8): 75−79.

[4] XU B L, LIN H, GONG J H. Architecture of HLA based distributed Virtual geographic environment[J]. Geo-spatial Information Science, 2006, 9(2): 127−134.

[5] LIU J W S. 实时系统[M]. 北京: 高等教育出版社, 2002: 161−178.

[6] 李佳, 朱元, 田光宇. CAN 与 TTCAN 通信延迟时间的分析[J]. 清华大学学报, 2006, 46(2): 261−265.

[7] 刘树锋, 陈欣, 刘标. 小型化无人机实时飞行仿真系统结构设计[J]. 兵工自动化, 2014, 33(6): 27−31.

[8] 黄安祥. 现代军机飞行仿真系统的研究[D]. 北京: 北京航空航天大学, 2002.

[9] 张继夫, 邓华. 多机飞行仿真系统扩展性研究[J]. 兵工自动化, 2010, 29(8): 25−27.

[10] 王行仁, 贾荣珍, 彭晓源. 飞行实时仿真系统及技术[M]. 北京: 北京航空航天大学出版社, 2003: 79−136.

[11] 郑宗汉. 实时系统软件基础[M]. 北京: 清华大学出版社, 2003: 213−243.