再谈网游同步技术：实时动作游戏同步方式和传输协议选择 - GameRes游资网

本帖最后由 小篱 于 2015-12-22 16:08 编辑

GameRes游资网授权发布 文 / 韦易笑

实时动作游戏在近年来得到迅猛的发展。而游戏同步问题，成为大家继续解决的核心问题之一。早在 2004年，国内游戏开发还处于慢节奏 RPG满天飞的情况下，我就开始实时动作游戏研究。分别在 2005-2006期间写了一系列相关文章，被好多网站转载：

帧间同步模式：《帧锁定同步算法》（2007）：http://www.skywind.me/blog/archives/131

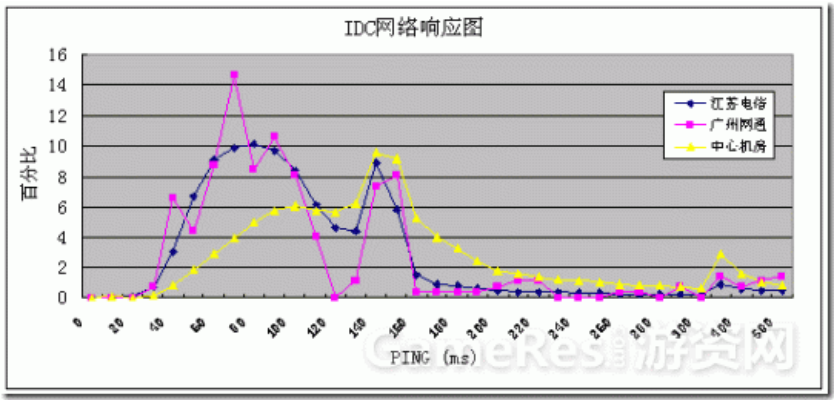
玩法规避模式：《网络游戏同步法则》（2005）：http://www.skywind.me/blog/archives/112

预测插值模式：《影子跟随算法》（2007）：http://www.skywind.me/blog/archives/1145

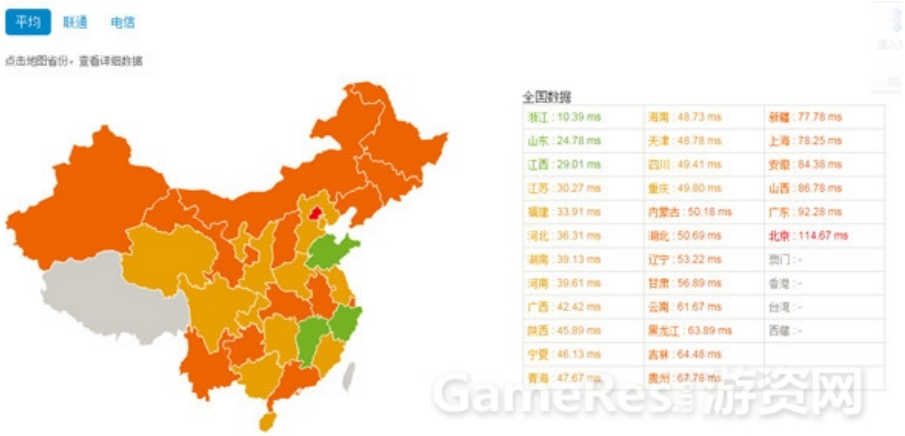
如今十年过去，网上越来越多的人开始讨论游戏同步技术了，然而很多文章往往只针对某种特定的游戏情况，而观点又经常以偏概全。很多人并没有真正开发过实时动作游戏，更别说了解同步技术的前世今生了。转载别人的观点并加上自己理解的人很多，实际动过手的人很少。避免给更多人造成无谓的误导，我今天基于先前的实践和对欧美动作游戏，战网游戏，主机游戏（PSN，XBox Live等）网络技术的了解，来对这个问题做一个简单总结：

网速的变化

开发快速动作游戏，首先要对公网的网络质量数据有详细的了解。这里所说到的网速，是指 RTT，数据往返一周的毫秒时间，而非每秒传送多少 KB/s。我写这篇文章是基于我 2005-2006年开发的东西来说的，当时国内公网质量比国外差很多：



上图为 2005-2006年国内的网络环境，某三个省级 IDC的情况采样。当时公网 RTT平均值基本在100ms，120ms左右徘徊。所以我文中引用了很多 100ms。这个情况在2009 年以后已经好了很多（60ms的rtt）。到了2012年以后，公网平均 RTT已经降低到平均 40ms-50ms，省内平均10ms以内了：



上图为 2015年某省级 IDC的全国延迟情况，如若全国多布点以及区别电信联通的话，平均延迟能控制在20ms以内，延迟基本接近国外水平（当然带宽还差很多），比我当年文章中提到的网络情况好了不少。

帧间同步法

关于帧间同步的“帧锁定算法”系列的方法有很多类似实现（包括后面提到的帧间无等待改进，包括 LockStep等），但是他们的核心都是一个：保证所有客户端每帧的输入都一样
。这样的方式被格斗游戏，RTS和足球（FIFA类）、篮球（NBA）等体育和动作游戏大量使用，比如我们熟悉的各大战网平台游戏（Xbox Live等），还有很多基于模拟器的街机对战平台。以及不少大型多人横版动作游戏。以开发便利，同步逻辑直观而受到大家欢迎。

帧锁定算法多用在 C/S模型中（或者一人做主多人做从的P2P里），它和 LockStep（多用于P2P）共同存在的问题就是 “网速慢的玩家会卡到网速快的玩家”，老式游戏经常一个角色断网，所有人就在那里等待。为此出现了帧锁定的改良版本 “乐观帧锁定”（具体描述见帧锁定文章的下半部分）经过了不少游戏的实践检验。先前还有几款上线的横版格斗页游（如熟知的街机三国）用 Flash 的 TCP without NODELAY 来每秒20个关键帧的模式（特意找该游戏开发者确认了一下）跑该算法（由于近两年国内网速提高，Flash的 Tcp without NODELAY也能做很多事情了），效果还不错。

具体实施时用不着按照文所述每一个步奏都相同，可以有很多变通。比如不一定是有变化的时候才通知服务端，有线上某横版格斗页游就是也可以每秒 20次向服务端直接发送数据（flash时钟不准需要自己独立计时），服务端再每秒 40次更新回所有客户端，看具体情况而定。

也有使用 UDP的端游，客户端每秒钟上传50次键盘信息到服务端，丢了就丢了，后面持续发送过来的键盘数据会覆盖前面的数据，所以丢了没关系，更快捷。当然，UDP也不是必须的，近两年网速提高很快，省内都能做到10ms的 RTT 了，跨省也就 50ms的rtt，不少页游上用该方法上裸的 TCP 照样跑的很顺畅。

而近两年国外动作游戏领域也涌现出其他一些新的改良方法，比如 Time Warp，以客户端先行+逻辑不一致时回滚的方式，带来了更好的同步效果，俗称时间回退法。不果国内暂时没看到有游戏这么尝试，更多的是国外近两年的双人动作游戏比较多，要求游戏每帧状态都可以保存，逻辑上开发会复杂一些。国内大部分是超过两人出去副本的，在3-4人出去 PK的情况下，引入状态回退，会让整个效果大打折扣。不过2人的效果确实有所改进，有兴趣的同学可以搜索 Time Warp相关的论文。

2009年，云游戏（游戏远程渲染）技术得到广泛应用，客户端上传操作，服务端远程渲染，并以低延迟视频编码流的方式传回给客户

端，用的就是这样类似的技术。客户端不需要高额的硬件，也不存在盗版问题，其中 Gaikai和 OnLive两家公司做的比较好。

2012年，Sony推出 Playstation Now技术，可以在 PSV和 PS3/PS4上玩云游戏，玩家不需要购买游戏就可以免费体验一定时间。使得 PSV/PS3等低端硬件也可以流畅的跑 PS4游戏。

但是目前国外网络环境下跑的还比较流畅，国内的网络环境要低延迟传送 HD画质的视频流还比较困难，视频都是比较费带宽的。但是帧锁定等保证每帧输入一致的算法，在当今的网络质量下传递一下玩家操作，还是没有任何问题的。

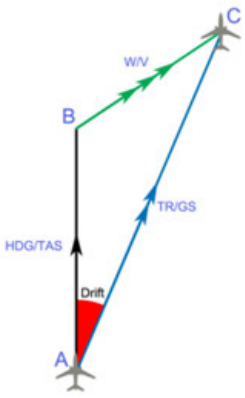
状态同步法

对于逻辑不需要精确到帧的游戏类型而言（RPG/ARPG，FPS，赛车），允许每个客户端屏幕上显示的内容不同，只要将他们统一到一个逻辑中即可，这部分见：“网络游戏同步法则”（最好给策划看看这篇，从玩法上规避）。如果是 RPG游戏，其实更多是使用障眼法从玩法和动画效果上减少“一次性的”，“决定性”的事件即可：

RPG 游戏的移动很简单，只需要“谁在哪里朝着哪里移动”，客户端再做一些简单的平滑处理即可，不需要额外的“时间”参数。比如《魔兽世界》移动时，就是差不多每秒发送一次（坐标，朝向，速度），别的客户端收到以后就会矫正一下，如果矫正错误，比如 A本来往北走突然拐弯向东，这个数据包传到B上，B屏幕上的A可能在拐弯前往北跑了更远，致使拐弯向东时被树卡住，那么B就会看到A被树卡了两秒无法移动，然后突然瞬间移动到新的坐标，继续朝着东跑。

通常 RPG攻击分为“有锁定攻击”和“无锁定攻击”，有锁定攻击意思是，我朝你发射火球，不管你怎么跑，火球都会追踪并射击到你，比如你在我面前横着跑过，我向你发射火球，可以发现火球并不是直线飞行，而是曲线追踪着你就过去了，这叫有锁定攻击。无锁定攻击一般是范围攻击，先播放个动画（比如挥刀），然后将攻击请求提交服务器，服务器结果回来时，动画刚好播放完毕，然后大家一起减血。

而 FPS和 赛车类游戏的同步性要求比 RPG高很多，每秒发包量也会多很多（10-30个），多半采用位置预测及坐标差值的“导航推测算法（DR）”，具体实现见我的：“影子跟随算法”（DR算法的一个改进实现）。



这类算法由于位置判定更为精确，所以计算量大，很多没法服务端判断，而是客户端直接判断，比如 FPS射击是否打到别人，客户端先判断，除了狙击这种一枪毙命的射击外基本都是客户端判断的。由于计算更为复杂，每秒同步发包差不多到 30个以上，这样的模式下，每局游戏的人数也不可能很多，一般16人左右。而且很多才用 P2P的方式运行，具体 FPS游戏的实现，及 DR算法的代码编写，见

“影子跟随算法
”这篇文章。

其实状态同步是一种乐观的同步方法，认为大家屏幕上的东西不同没关系，只要每次操作的结果相同即可，不需要象“帧间同步”那样保证每帧都一样，因此，对网速的要求也没有“帧间同步”系列算法那么苛刻，一般100ms-200ms都是能够接受的（DiabloIII里面300ms的延迟照样打），偶尔网络抖一下，出现1秒的延迟，也能掩盖过去。然而比起“帧间同步”，状态同步方式对玩法有不少要求，诸如“一次性”，“决定性”的事件要少很多，而且代码编写会复杂一些，不果由于能容忍更坏的网络情况，以及容纳更多同时游戏的人数，在一些玩法确定的游戏中（RPG，FPS，赛车），被广泛使用。

而状态同步又分为“DR同步”和“非DR同步”，前者针对FPS，赛车或者更激烈点的ARPG，后者针对RPG和普通ARPG。他们对网速的要求和错误的容忍度也是不一样，当然，带来的游戏即时感也是不同的。

总得来说，你希望游戏体验更爽快，即时感更强，那么你每秒发包数就越多，每局（副本）支持的人数越少；而你如果追求对网络的容忍，想降低发包数，并且增加同时游戏的人数，那么相应的就需要以降低即时感为代价，其二者不可得兼。然而聪明的策划和程序们总能想出很多好主意，利用障眼法和玩法规避，动作掩盖等方法，在相同的情况下来掩盖延迟，让玩家“看起来”更加“即时”和“爽快”，而这个方法具体该怎么做，并没有统一的做法，就得大家结合自己的游戏和玩法，发挥自己的聪明才智了。

结果同步法

结果同步往往比较简单，位置即使全部错乱或者延迟很久都没有关系，因为游戏过程完全不在乎位置，只在乎最后的结果，比如《梦幻西游》这样的“回合制RPG”游戏，屏幕上的人走到哪里确实无所谓，所有操作都是要点击或者选择菜单来下命令，象这样的游戏背后其实是文字游戏，只是加了一个图形的壳。

游戏表面上看起来是动作/RTS游戏，但是没有玩家直接协作和对抗，都是单机游戏，并不需要同步什么东西，服务端只要监测下结果不离谱即可，延迟检测都没关系。基本是PVE，而且无协作。即使是PVP也就是打一下别人的离线数据，和无同步回合制游戏并无本质上的区别。

传输协议选择

老话题TCP还是UDP，答案是大部分时候，TCP打开NODELAY即可，现在网络情况好了很多，没必要引入新的复杂度。即便是“帧锁定算法”上线的多人实时格斗游戏，也有在用TCP跑着的。帧间同步如果能够做到更好的架设机房，那么延迟基本能控制在10ms以内，将游戏玩家按照区域分服务器，让他们选择更快的服务器。

即便是带DR的状态同步，很多也都是TCP的，《魔兽世界》和《暗黑破坏神3》都是基于TCP来实现的，所以我的建议是，先上TCP，把你的游戏发布出去。

当然，等到你的游戏发布出去了，开始挣钱了，你想改进你的游戏效果，特别是高峰期的卡顿比例（需要收集客户端统计），那么你可以使用UDP来改进，《街霸4》和《英雄联盟》都是使用UDP的，比如你可以使用libenet（英雄联盟用的）。不过libenet所采用的传输技术，是上世纪的标准ARQ做法了，《街霸4》所采用的传输技术远远高过libenet，如果你想采用更为现代的传输技术，赢得更低延迟的话，可以使用我的“快速传输协议-KCP”（http://www.skywind.me/blog/archives/1048
），被再若干上线项目和开源项目使用的协议，效果远远PK libenet。

在使用KCP时，你可以用在你TCP的基础上，再登陆时服务端返回UDP端口和密钥，客户端通过TCP收到以后，向服务端的UDP端口每隔一秒重复发送包含握手信息，直到服务端返回成功或者失败。服务端通过UDP传上来的密钥得知该客户端sockaddr对应的TCP连接，这样就建立TCP连接到UDP连接的映射关系。为了保持连接和NAT出口映射，客户端一般需要每60秒就发送一个UDP心跳，服务端收到后回复客户端，再在这个UDP连接的基础上增加调用KCP的逻辑，实现快速可靠传输，这样一套TCP/UDP两用的传输系统就建立了。

中国的网络情况比较特殊，会存在有些网络 UDP连接不上的情况，因此都是先连接 TCP，然后试图 UDP，UDP不通的情况下，退回TCP也能正常游戏，一旦 TCP断开，则认为 UDP也断开了。

不果归根结底，还是先上 TCP，再根据自己游戏的特点和是否出现传输问题，选择 UDP。

话题总结

根据游戏类型，选择恰当的同步方式和传输协议是最基础的问题，很多讲述网络同步的文章一般就是只会强调上述那么多种算法的其中一种方式，好像使用该方式就可以 hold住所有游戏一样的，其实并非如此。技术需要多和策划沟通，别策划一个需求下来，技术就来一句：无法实现，这样的游戏永远没有竞争力。就像国内当时都是慢节奏 RPG，偶尔有点 ARPG的时候，大家觉得《DNF》这样的游戏无法实现，于是韩国实现了，在市场上取得了先机，国内才慢慢跟进，再一看，哇塞，好多坑呢。

然后开发者开始在网上寻找各种同步算法，东一榔头西一棒子，明明是一款需要帧间同步的格斗游戏，结果却上了导航推测，最后发现问题永远解决不了，一堆 BUG这就叫误导。正因为我 2004年就开始弄同步相关的问题，期间也指导过不少游戏设计他们的同步方案，所以这次相当于将以前的观点做一个总结和点评，根据自己的游戏类型选择最适合的同步算法，为玩家提供更好的体验才是关键。

相关阅读：

手机格斗网游该如何避免延迟？

影子跟随算法：FPS游戏中游戏同步性的实现