视频会议中的同步缓冲设计*)

张岩峰[1] 王翠荣[2] 赵煜辉[2] 高 远[2]

(东北大学信息科学与工程学院 沈阳 110004)[1] (东北大学秦皇岛分校 秦皇岛 066004)[2]

摘 要 接收缓冲区对于保证视频流的播放质量起着关键的作用。针对接收实时视频流这种特殊需求,本文提出了一种时间驱动的具有同步功能的缓冲区设计方案。经过实验测试,缓冲区在平滑视频呈现、解决恶劣网络环境下的丢包乱序等问题有良好的效果,已经应用到某大型网通视频直播接收系统中,可以广泛应用于任何接收实时视频流的客户端系统中。

关键词 缓冲区,实时视频流,UDP,丢包,乱序

Design of a Synchronous Buffer in Video Conference

ZHANG Yan-Feng[1] WANG Cui-Rong[2] ZHAO Yu-Hui[2] GAO Yuan[2]

(School of Information Science & Engineering, Northeastern University, Shenyang 110004)[1]

(Northeastern University at Qinhuangdao, Qinhuangdao 066004)[2]

Abstract Receive buffer plays a key role on ensuring the playback quality of video media stream. In light of this special request on receiving real time video media stream, a design of buffer based on time driver is proposed in this paper. After a test, this buffer has good results both on smoothing video show and on solving the packet drop and packet disorder problems brought from the poor network environment. This design can be widely applied in the client systems receiving real time video media stream.

Keywords Buffer, Real time video stream, UDP, Packet drop, Packet disorder

1 引言

随着国际互联网技术和多媒体技术的发展,网络视频流媒体成为近年来研究的重点。视频会议系统、网络直播互动平台、网络电视等的研究与开发引起了众多软件企业和研究单位的极大兴趣。特别是视频媒体的实时传输,以其实时性和良好的交互性成为最近研究的热点技术。为了保证传输的实时性,视频流媒体的实时传输都采用点对点尽力传送的传输层协议 UDP,而放弃了有差错检验和错误重传机制的 TCP 协议。虽然在应用层可以采用 RTP/RTCP 协议对 QoS 提供一定的保证,但在传输层上的包丢失和乱序的情况仍然无法避免,所以应用程序必须采用必要的策略对丢包和乱序数据进行处理。如果对乱序数据不加以处理而直接交给下游的解码器处理,将产生解码后图像严重失真,并且经过测试表明,过多的乱序或错误数据包传给解码器还会让解码器死掉[1]。为解决这个问题,一般的方法是在数据递交给解码器之前缓冲一定数量的数据包,即添加缓冲区。缓冲区负责对数据包重新排序,制定丢包策略,平滑播放,计算丢包率反馈给 RTP 服务器等。本文针对接收实时视频流的特殊需求,提出一种基于时间驱动、具有同步功能的视频数据缓冲区的设计方案。缓冲区除了完成排序组帧、制定丢包策略等功能外,由时间精确控制视频的解码和播放,实现视频的平滑播放。

2 同步缓冲设计

本节给出具体的同步视频缓冲区设计方案。

2.1 同步缓冲的整体结构

缓冲区采用二级缓冲策略,一级缓冲为数据暂存缓冲,二级缓冲为数据信息缓冲,且为环状结构,并启动数据接收线程和同步功能线程两个线程。首先由数据接收线程接收数据包,并保存在一级数据暂存缓冲区中,读取其帧序号及帧内包序号信息来确定此数据包在二级缓冲环中的具体位置,完成二级缓冲队列的排序。二级缓冲环以组合完整的帧为单位,而不是以数据包为单位,在出队列时检测此帧是否完整,然后把整帧数据交给解码器处理。缓冲环设计一个出队列指针,由时间线程来驱动出队列指针的移动,这样就能精确地控制音视频数据向下游解码器传递,使视频图像可以平滑地显示,有效地避免了视频延时抖动情况的发生。整体结构如图 1 所示(一级缓冲提交给二级缓冲的数据结构包括:DataLen 数据包长度、FrameSeq 帧序号、PackSeq 帧内包序号、TotalPack 帧内包总数、FrameType 帧类型、DataAdress 数据包有效载荷在一级缓冲中的地址)。

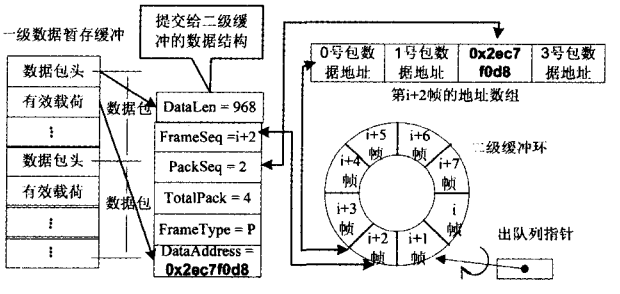

图 1 缓冲区整体结构

2.2 同步数据包头格式

媒体发送端把采集编码出来的帧数据分割成较小的固定

*)国家自然科学基金(06273078)、河北省科技厅博士基金项目(55470130-3)。

长度的数据包发送。经过测试，选择 968 字节效果比较理想。如果在媒体发送端直接以帧为单位打包发送，则有些帧的数据量较大，会产生大数据包，在 UDP 布可靠网络上很容易丢包，而且对于一些路由器不支持大数据包的传输，会过滤掉较大数据包或产生分片。在接收缓冲区组包要配合媒体发送端打入包头的信息。本文参考 RTP 数据包头结构[2]，在应用层设计了图 2 所示的数据包结构，其中包括数据包长度 4 字节、帧序号 4 字节、帧内序号 1 字节、帧包总数 1 字节、帧类型 1 字节、扩展字段 1 字节、时间戳 4 字节、有效载荷 968 字节。利用这些信息配合接收缓冲区完成乱序组包，确定丢包策略和同步功能。

<table>
<tr><td colspan="4">数据包长度</td></tr>
<tr><td colspan="4">帧序号</td></tr>
<tr><td>帧内序号</td><td>帧包总数</td><td>帧类型</td><td>扩展字段</td></tr>
<tr><td colspan="4">时间戳</td></tr>
<tr><td colspan="4">有效载荷
…</td></tr>
</table>

图 2　数据包结构

2.3　数据接收线程设计

假设缓冲区缓存 K 帧数据，通过回环时间和实时要求综合确定 K 的最优值[3]。首先为一级缓冲分配足够容纳 K 帧数据的一块内存区域，作为数据暂存。接收到的数据包依次放入一级缓冲中，当缓冲即将溢出时再从缓冲区起始内存地址开始接收。应用程序分析数据包，把包头中用来排序组帧的一系列信息和一级缓冲暂存此数据包有效载荷的内存地址提取出来交给二级缓冲。

二级缓冲环长度为 K，对应缓存的 K 个帧的信息。帧信息包括接收到的数据长度、帧序号、帧内包总数、帧类型和记录帧内各个分包在一级缓冲中内存地址的数组。根据一级缓冲提交上来帧信息的帧序号，计算此帧对应于二级缓冲环的位置，然后根据帧内包序号计算此数据包对应的地址数组的索引，并递增此帧接收到的数据长度。由于媒体数据量大，接收端会频繁接收到数据包。如果拷贝数据，必将消耗大量 CPU 资源。而通过传递内存地址而不是拷贝数据，减少了一次拷贝内存的开销，节省了可观的 CPU 资源[4]。

用一个长度为 K 的数组（Frame[0]，Frame[1]，…，Frame[K−1]）来模拟缓冲环，index 是数组索引，对应缓冲环的出队列指针，即 Frame[index]对应当前向解码器传递的帧，Frame[(index+1)%K]为下一个向解码器传递的帧。下面给出数据包排序的算法：

(1)接收数据包到缓冲区；

(2)分析数据包头，读取数据包的帧序号，计算此数据包是否在缓冲范围内，如果在缓冲范围内，进入第(3)步；否则返回第(1)步；

(3)根据帧序号计算数据包对应的帧数组索引 i，向其对应的项 Frame[i]中写入帧属性信息；

(4)根据帧内包序号确定第(3)步 Frame[i]的地址数组索引，向其对应的项中写入数据包在一级缓冲中的内存地址；

(5)更新此帧已接收到的数据长度。返回第(1)步。

2.4　同步功能线程设计

传统缓冲区设计为 FIFO 队列，利用满则溢的原理，当接收数据包插入缓冲队列的尾部结点，又接收到其后数据包需要加入缓冲队列时，位于队列头部结点的帧数据将向下传递。这种方案的驱动力是网络接收的数据包，假设应该到的数据包由于网络原因比较集中地到达或比较分散地到达，表现到视频呈现上就是快动作或慢动作，影响视频会议效果。

选择时间作为出队列指针移动的驱动力，可以精确地驱动帧数据向下游解码器传递，提供连续流畅的视频呈现。通过线程循环调用时间阻塞函数 Wait()来实现这个功能。

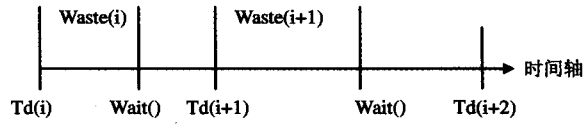

图 3　时间线程流程图

如图 3 所示，Td(i)表示传递第 i 帧的时刻，也即时间阻塞函数 Wait()被激活的时刻。根据第 i 帧和第 i+1 帧的时间戳可以计算出传递第 i+1 帧的时刻：

$$Td(i+1)=Td(i)+Frame[i+1].TimeStam-Frame[i].TimeStamp \quad (1)$$

Waste(i)表示传递第 i 帧所耗费的时间，我们可以通过记录系统时间来得出。这样，程序运行到时间阻塞函数 Wait()的时刻为 Td(i)+Waste(i)，时间阻塞函数 Wait()所等待的时间$\triangle t$ 也可以由下式计算出：

$$\triangle t=Td(i+1)-Td(i)-Waste(i) \quad (2)$$

而对于丢包等特殊情况，无法取得 Frame[i+1].TimeStam。这时，采用上一次计算出的$\triangle t$ 来代替。这种根据时间戳动态确定$\triangle t$ 的方法，会灵活地根据发送端发送帧速率的动态改变而变化，这样就通过时间精确控制了视频流的播放。

2.5　同步缓冲的丢包策略

在一个视频序列 GOP 中，假设某帧 Frame[i]不正确，这时如果对此帧解码，就会错误解码。如果 Frame[i]为参考帧，解码器在解码 Frame[i+1]的时候就会以 Frame[i−1]作为参考帧解码。显然，这种错误解码会导致图像质量下降，还会导致误码扩散，即 Frame[i+2]，Frame[i+3]……都会错误解码，严重影响视频效果。所以，在缓冲区要根据帧类型制定合理的丢包策略。

当时间事件被触发，而出队列指针指向的帧 Frame[i]尚不完整，有部分数据包还未到达，应用程序就放弃将此帧向下游解码器传递。然后判断 Frame[i]的帧类型：如果 Frame[i]是非参考帧，就简单将此帧丢弃，而不影响后续帧接收传递；如果为参考帧，则放弃向解码器传递缓冲环中后续帧{ Frame[i+1]，……，Frame[MaxIndex]，Frame[0]，……}以及后接收到的帧，直到检测到下一个关键帧为止，也即是丢失该视频序列的后续帧。一般来说，I 帧(关键帧)、P 帧(向前预测帧)作为参考帧，而 B 帧(双向预测帧)作为非参考帧。当然也有特殊情况，H. 264 标准中有时会将 B 帧也作为参考帧[5~7]。而传统的丢包策略不区分参考帧与非参考帧，这样会增加一些不必要的丢帧。下面是丢包控制算法：

(1)等待时间驱动事件；

(2)从二级缓冲队列中读取队列头指向的帧信息；

(3)检查此帧数据。如果完整，进入第(4)步；否则进入第(6)步；

(4)检查此帧类型。如果是关键帧，设置传递标志为真；否则进入第(5)步；

(5)检查传递标志。如果为真，将此帧数据向下游解码器传递；否则进入第(7)步；

(6)检查此帧类型。如果为参考帧，设置传递标志为假；否则进入第(7)步；

(7)移动出队列指针。返回第(1)步。

3　实验方法与测试结果

采取在发送端主动丢包和主动乱序来模拟网络上的复杂情况。具体方法是以时间为种子取伪随机数 R，指定一个丢

包或乱序概率 p。当 $R\ \%\ 100 < p * 100$ 时，在概率范围内，即此时主动丢包或主动乱序。模拟丢包可以简单地放弃发送实现。对于模拟乱序情况，在发送的时候将需传输乱序的数据包缓存起来，然后将它延后该数据包原发送位置 W 的位置后进行发送（W 为乱序偏差），来模拟数据包传输乱序。

本次测试在 CPU 主频 1.70GHz、内存 256MB 环境下进行。测试时间为 1min，发送帧速率 15fps，最大关键帧间隔为 15，缓冲区缓存长度为 6 帧。测试丢包情况设置图像分辨率为 352×288，图 4 为其测试结果；测试乱序情况分别取乱序偏差值为 5、10、15，并分别测试图像分辨率为 176×144 和 352×288 两种情况，其测试结果如图 5、图 6 所示。

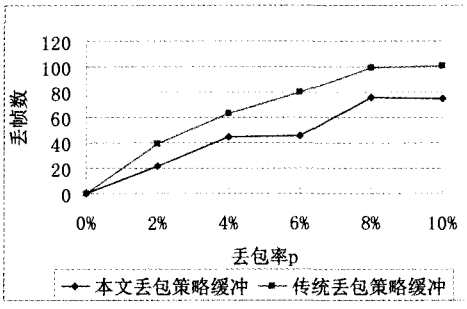

图 4　丢包测试结果

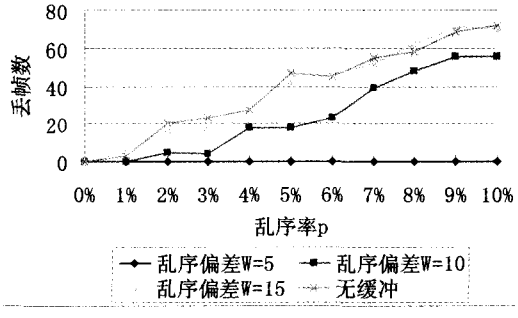

图 5　乱序测试结果（176×144）

从图 4 中可以看出，新缓冲增加了判断非参考帧策略，避免了一些不必要的丢帧，减少了一定数量的丢帧数。由图 5 可知，在乱序偏差较小的情况下，本缓冲设计有较好的性能，但是随着乱序偏差增大，性能提高不明显。这是由于在乱序偏差较大的情况下，乱序包所在帧的间隔可能会大于缓冲区缓冲长度 6 帧。这样，缓冲区将认为乱序包不在缓冲范围之内，丢弃乱序包。而在图 6 所示的较大分辨率下，由于每一帧的数据量较大，分包较多，缓冲区暂存了 6 帧，即暂存了更多的数据包，因此可以适应更大的乱序偏差。综合可知缓冲区性能与图像分辨率和缓冲区长度有关。但是整体来看仍然有不错的性能指标，并且缓冲区由时间驱动，精确控制帧数据向解码器传递，同步了发送端和接收端的视频数据，平滑了图像显示，具有良好的主观评测性能。此外，该缓冲区减少了一次拷贝，节省了相当可观的系统资源。

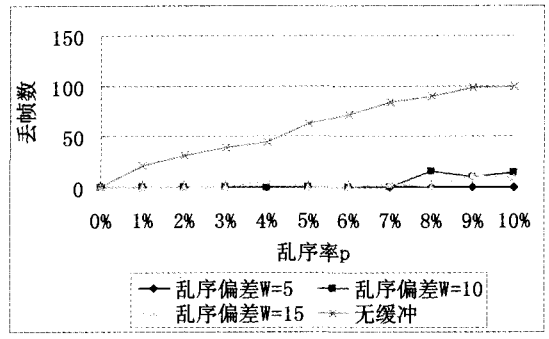

图 6　乱序测试结果（352×288）

结论　实时视频流应用已经越来越广泛，对接收端缓冲的性能要求也越来越高。本文提出了一种基于时间驱动的具有同步功能的缓冲区设计方案。仿真测试表明，该方案有良好的客观性能指标和较好的主观感觉，已应用于秦皇岛东大软件视频会议系统、北京网通网上直播系统等实时视频流接收客户端的设计中，取得较好的效果。

参 考 文 献

1　Richardson I E G. Video Codec Design[M]. Wiley, 2002. 27～46

2　Casner S, Frederick R, Jacobson V, et al. RFC3550, RTP[S]. USA.: Columbia University, 2003. 13～52

3　Liu Jing, Niu Zhisheng. An Adaptive Receiver Buffer Adjust Algorithm for Voice & Video on IP Applications[C]. In: 2005 Asia-Pacific Conference on Communications, Perth, Western Australia, October 2005. 669～673

4　Zhou Yuanyuan, Chen Zhifeng, Li Kai. Second-level Buffer Cache Management[J]. IEEE Transactions on Parallel and Distributed Systems, 2004, 15: 505～519

5　毕厚杰. 新一代视频压缩编码标准——H. 264/AVC[M]. 人民邮电出版社, 2006

6　Richardson I E G. H. 264[S]. Switzerland: ITU-T, 2005. 93～184

7　Wenger S H. 264/AVC Over IP[J]. IEEE Trans Circuits Syst. Video Technol, 2003, 13: 645～656

（上接第 53 页）

参 考 文 献

1　Kumar S, Jantsch A, Soininen J, et al. A Network on Chip Architecture and Design Methodology[C]. In: Proceedings of IEEE Computer Society Annual Symposium on VLSI, Washington DC, USA: IEEE Computer Society, 2002

2　Wang H S, Zhu X, Peh L S, et al. Orion: a power-performance simulator for interconnection networks[C]. In: Proceedings of the 35th Annual ACM/IEEE International Symposium on Microarchitecture, Washington DC, USA: IEEE Computer Society, 2002. 294～305

3　Hemani A. Jantsch A, Kumar S, et al. Network on a chip: An architecture for billion transistor era[C]. In: Proceeding of the IEEE NorChip Conference, November 2000

4　Hu J, Marculescu R. Energy-aware Mapping for Tile-based NoC Architectures Under Performance Constraints[C]. In: Proceedings of the 2003 onference on Asia South Pacific Design Automation. New York, USA: ACM Press, 2003. 233～239

5　Garey M R, Johnson D S. Computers and intractability: a guide to the theory of NP-completeness[M]. New York, USA: W H Freeman & Co, 1979

6　Hu Jingcao, Marculescu R. A survey of wormhole routing techniques in direct networks[C]. Los Alamitos, CA, USA: IEEE Computer Society Press, 1993

7　Khan G N, Gu Wei. Fault-tolerant Wormhole Routing Using a Variation of Distributed Recovery Block Approach[C]. IEEE Proceedings of Computers and Digital Techniques, 2000, 147(6): 397～402

8　Ye T T, Benini L, De Micheli G. Analysis of power consumption on switch fabrics in network routers[C]. In: Proceedings of Design Automation Conference, June 2002. 524～529

9　孙增圻，张再兴，邓志东. 智能控制理论与技术[M]. 北京：清华大学出版社，1997

10　张传顺，莫蓉，石胜友，等. 基于遗传算法的制造网格资源调度方法研究[J]. 中国机械工程，2006, 17(18)