即时通讯——详解音视频同步技术

摘要：针对网络传输中由于延迟、抖动、网络传输条件变化等因素引起的音视频不同步的问题，设计并实现了一种适应不同网络条件的音视频同步方案。利用音视频编码技术AMR-WB和H.264具有在复杂网络环境中速率可选择的特性，结合RTP时间戳和RTCP反馈检测QOS，通过控制音视频编码方式，实现了动态网络环境下的音视频同步方案。重点介绍了可靠网络环境和动态网络环境下同步算法的设计过程，并通过实际测试验证了此方案的可行性。结果表明，此方案能够保证不同网络环境中的音视频同步。

引言

音视频媒体间同步是多媒体系统服务质量（QoS）研究中的一项重要内容。在网络上传输多媒体数据时，由于终端对数据的处理方式，以及网络中的延时、抖动，会引起音视频流的不同步。传统的解决方案往往存在实时性差，时间开销大，且无法适应动态网络环境等缺陷，针对此问题，本文在分析媒体间同步性定义、影响因素等的基础上，提出了一种基于循环缓冲队列和RTCP反馈控制的同步解决方案。

1 媒体间同步性定义

同步是多媒体通信的主要特征，也是其重要研究内容之一，同步与否直接影响多媒体通信的质量。媒体间同步即是要保持音频流和视频流之间的时间关系[1]。为了描述同步，实现相关的控制机制，定义了相应的服务质量参数（QoS）。针对音视频，采用时间差即偏差来表示。结果表明，如果偏差限制在一定的范围内，认为媒体是同步的。当偏移在-90ms（音频滞后于视频）到+20ms（音频超前视频）之间时，人感觉不到试听质量的变化，这个区域可以认为是同步区域；当偏移在-185到+90之外时，音频和视频会出现严重的不同步现象，此区域认为是不同步区域。本设计认为偏移在-120ms到+40ms之间音视频同步。

1.1 音视频同步的影响因素

在网络环境下，多媒体信息在传输过程中受到各种因素的影响，会导致其在接收端不能正确播放，即音视频不同步。引起音视频不同步的原因主要有两种：一种是终端处理数据引起的，发送端在数据的采集、编码、打包等模块和接收端在处理解包、解压、回放等模块时，由于音频和视频的数据量以及编码算法不同而引起的时间差。并且发送端没有统一的同步时钟；另一种是网络传输延时，网络传输是受到网络的实时传输带宽、传输距离和网络节点的处理速度等因素的影响，在网络阻塞时，媒体信息不能保证以连续的“流”数据方式传输，特别是不能保证数据量大的视频信息的连续传输，从而引起媒体流内和流间的失步[2-3]。

2 音视频同步系统设计

在音视频同步系统中，发送端在发送音视频流时，要给**各帧数据打上相对时间戳**，并且音频流和视频流，一个作为主流，另一个作为从流。主流连续播放，从流的播放由主流的播放状态决定，从而实现同步。考虑到人对声音更为敏感，在本设计中选择音频流作为主流，视频流作为从流。发送端通过AMR-WB和H.264编码模块对DirectShow采集到的音视频数据进行编码，经过同步处理，最后利用RTP/RTCP等协议实现媒体流的传输和控制。接收端接收到RTP传过来的音视频数据包后，对数据进行解码，然后同步处理，最后通过DirectShow播放音视频。

3 音视频同步方案设计

考虑到传统的同步方案只是在接收端通过**RTP时间戳**实现同步，即将具有相同时间戳的音视频数据同时表现出来，这种方案由于没有从有效控制和适应不同网络环境的角度去实现，并且读写时间戳的开销太大，需要全网同步时钟等缺陷，因此不适应于音视频媒体间同步[4]。针对此问题，这里提出一种结合发送端，利用RTP/RTCP以及可控音视频编码技术，适用于不同网络条件的同步方案。主要表现在以下两方面：1、发送端数据的采集、编码即发送控制；2、利用RTCP的反馈指标，通过可控速率的音视频编码算法动态适应不同的网络环境。

3.1 RTP时间戳同步

在网络畅通时，网络传输时延基本恒定，抖动很小，发送端和接收端的音视频帧间间隔基本保持一致，媒体数据基本没有丢失。由于音视频的RTP之间无直接关联的控制，所以不能通过关联控制同步。此时主要利用**RTP包头的时间戳**来解决。

在发送端，同一媒体内的时间戳控制：针对音频的不同采样速率和视频的不同帧率来动态的控制时间戳的递增速率；不同媒体间的同步控制：同一时间采集到的数据打上同样的时间戳，并在同一线程里交替发送音视频数据包。

在接收端，当音视频数据到达时，先对两种数据帧进行解码，在将其解码数据存入各自的动态循环缓冲区中。因**为音频和视频的每个数据帧解码时间不能准确得到，为了准确地实现音视频同步回放，采取先解码再同步处理的方法。在网络畅通时，可以把两种数据的解码时间差作为抖动延时的一部分来处理**。但是，在网络环境不好时，不采用这种方法处理。

（1）接收端对音频帧的处理如下：

SHAPE

* MERGEFORMAT

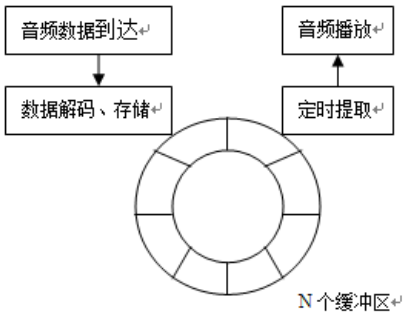
音频数据到达

数据解码、存储

音频播放

N个缓冲区

定时提取



图一 接收音频帧示意图

如图一所示，为了消除抖动，接收端采用基于**循环缓存区的方法**保证音频的连续性。这种方法有两个优点：一是可以根据RTP数据的接收情况动态的建立缓存空间，二是可以保证缓存中有足够的音频数据用于播放。接收端接收到音频帧时，首先对其解码，并存入动态的循环缓冲区中，循环缓存块节点数的门限值为N，该值比预计最长抖动时间要大。开始启动播放音频前，首先把缓冲区充满，然后**定时提取音频帧播放，并记录当前播放的时间戳**。

（2）接收端对视频帧的处理如下：

SHAPE

* MERGEFORMAT

视频数据到达

解码存储

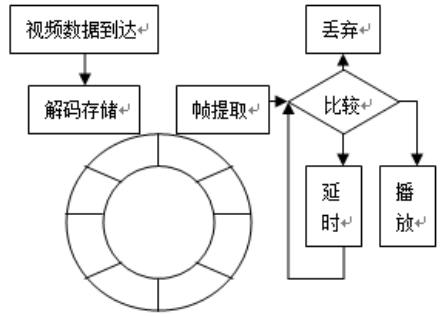
帧提取

比较

播放

丢弃

延时



图二 接收视频帧示意图

如图二所示，视频帧到达时，接收端对其解码后，将解码数据存入**循环缓冲区**。为了避免高速视频画面出现的块效应，本系统**采用事件驱动的方式来播放视频流**。当缓冲区接收到一个视频数据包时，把**该帧的时间戳TVIDEO与当前待播放的音频数据的时间戳TAUDIO**进行比较。本设计中规定音视频帧不同步的容忍度为TMAX=120ms。因此对一帧视频数据的处理结果分为以下三种：若TAUDIO-TMAX<TVIDEO<TAUDIO+TMAX,就播放该视频帧。

若TVIDEO<TAUDIO-TMAX，视频帧滞后，就丢弃该帧。

若TVIDEO>TAUDIO+TMAX，视频帧超前，**等待下一次定时读取音频帧时再处理**。

接收端对视频帧进行同步处理的实现代码如下：

```
OnRTPPacket ( RTPPacket *pack,
const RTPTime &receivetime,
const RTPAddress *senderaddress )
{
size_t
buffsize=pack->GetPayloadLength();
memset(m_buf,0,MAX_PACKET_SIZE);
//接收视频流的RTP数据包 memcpy(m_buf,(void*)pack->GetPayloadData(),buffsize);
//对视频数据进行同步处理
```

```
m_psynvideo->Issynvideo(TAUDIO,m_buf);
switch(Issyn)
case
1: //播放该视频帧
m_pVideoOut->ReceiveVideo(m_buf,buffsize);
break;
case
2:
delete(m_buf);
//视频帧滞后，丢弃该帧
break;
case
3:
waite(m_buf);//视频帧超前，等待下次处理
break;
}
}
```

3.2 RTCP反馈控制

当网络环境较差，无法为系统提供RSVP时，音视频流不能按原定的传输速率传送，否则会出现数据包丢失严重的情况，这时需要采用RTCP来进行反馈控制。即利用RTCP的发送报告SR和接收报告RR包监测QOS[5]。

接收端将RR包发送给源端，该报告包含用来估算分组丢失和分组延迟抖动等必要信息。源端根据这些信息控制媒体数据的发送量，及时有效地解决同步问题。

根据评估RR包的参数，得到长时指标丢包率和短时指标间隔抖动。当丢包率和抖动达到一定值时：音频方面，当网络丢包率和抖动达到某一区域时，选择不同的AMR-WB传输速率，来降低音频传输码率，提高传输效率和系统容量，为视频传输减少了带宽负担。

视频方面，根据不同值调整视频数据的发送量，即在发送端对视频的空域和时域性能进行平衡，选择丢帧：

（1）当丢包率和抖动很高，即信道速率很低时，通过降低视频帧率，使每一帧能够具有较好的空域质量，使用户在较低的速率条件下，任然可以得到较好的图像质量。

（2）当丢包率和抖动保持在中等水平，即信道速率中速时，在保持一定的空域质量条件下，应优先考虑时域质量，增强视频的连续性。

（3）当丢包率和抖动回到较好的水平，即信道速率较高时，在空域质量达到一定程度后，继续提高空域质量，效率不会太高，反而是图像连续性的提高对视频质量的改善更明显。

4 例子：

AnyChat采用动态缓冲技术，会根据不同的网络状况实时调节缓冲区的大小，在实时性和流畅性之间保持平衡。

当网络状况较好时，AnyChat会减小缓冲区的容量，提高音视频的实时性；

当网络状况较差时，AnyChat会增大缓冲区的容量，这样会带来一些延迟的增加，但是能保障音视频的流畅性，有效消除网络抖动对音视频播放质量的影响；

根据实际网络测试，AnyChat的音视频延迟指标如下：

网络状态较好时(无丢包，网络延迟<=10ms)：<1s

网络状态一般时（无丢包，网络延迟<=50ms）：<=1s, >=0.5s

网络状态较差时（丢包率<=5%，网络延迟<=100ms）：<=1.5s

网络状态较好时（无丢包，网络延迟<10ms）：<100ms

网络状态一般时（无丢包，网络延迟<50ms）：<=100ms

网络状态较差时（丢包率<=5%，网络延迟<100ms）：<=250ms

网络状态很差时（丢包率<=20%，网络延迟<500ms）：<=1100ms

注：上述指标为发言模式下的测试值，如采用放歌模式，则内核为了保障播放的流畅性，会适当增加缓冲区大小，导致延迟增大。

AnyChat Platform Core SDK V4.6对延迟进行了优化，局域网环境下，实时高清视频（720P，25fps）通话延迟<100ms。

5 结论

本文设计实现了一种适应不同网络环境的音视频同步方案。设计中利用**RTP时间戳及循环缓冲区**在可靠网络环境下对音视频进行同步，以及在动态网络环境下，利用RTCP反馈控制来动态改变音视频编码方式的同步方案。此方案已经成功应用于作者开发的网络多媒体终端上，保持了较低的丢包率，保证了终端之间多媒体信息的传输质量。