基于 DirectShow 的 PDA 视频传输过滤器①

PDA Video Transmit Filter Based on DirectShow

陈　锋　陈名松　盖晓娜　(桂林电子科技大学 信息与通信学院 广西 桂林 541004)

摘　要： 在 Windows Mobile 操作系统上利用 DirectShow 流媒体技术设计出了视频传输过滤器。该视频传输过滤器可以有效保障视频传输过程中的可靠性和实时性，为目前的 PDA、手机等移动终端上实现视频通话、视频会议等功能应用提供有力支持。文中涉及到了过滤器的开发过程、视频图像的切帧封装、接收端环形缓冲区的使用以及如何利用 JRTP 来实现底层的视频帧传输等等，解决了目前 PDA、手机等移动终端上的可视电话中的视频传输问题。

关键词： 可视电话 过滤器 切帧 环形缓冲区 JRTP

1　引言

目前，随着网络技术和人们需求的不断提高，网络可视电话逐渐兴起。近几年网络可视电话已经逐渐朝着无线方向发展，出现了利用 VOIP 技术实现的 PDA、手机可视电话。在这些可视电话业务中，其视频处理涉及到视频的采集播放、压缩、编码和传输等几个方面。在传统的视频开发中一般通过调用系统函数或硬件接口，通过这些接口设计的流媒体系统通常架构封闭、扩展性差、开发难度大时间长且不利于底层操作。在 Windows Mobile 操作系统的 PDA 上，Windows Mobile 为用户提供了 DirectShow SDK 进行流媒体开发，DirectShow 为媒体流的采集、回放以及格式之间的转换提供了强有力的支持。DirectShow 采用基于模块化的设计思想，各个模块按照功能的不同划分为过滤器，又称 Filter。过滤器具有可重复利用的优点，可提高系统开发进度。在 Windows Mobile 操作系统上，DirectShow 提供了一系列的过滤器方便开发，但并不提供所有的过滤器，开发者可以根据自己的需要设计开发过滤器。本文就是根据 PDA 上的视频传输需求，利用 DirectShow 技术设计了基于 Windows Mobile 的视频传输过滤器。

2　PDA视频传输系统

基于 DirectShow 的 PDA 视频传输系统主要由发送端的视频采集、视频编码以及视频发送和接收端的视频接收、视频解码以及视频播放组成。按照 DirectShow 的基于模块化的设计思想，PDA 视频传输系统的每个功能模块都采用 COM 组件方式(即 Filter)来实现。每个过滤器完成各自特定的功能，并按照各自处理顺序的不同在 DirectShow 中组成一个 Filter 链表，即 Graph 图。下图显示了 PDA 视频传输系统的 Graph 图。

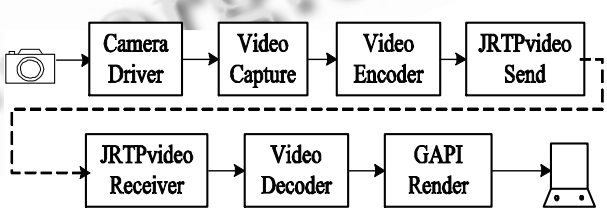

图 1　基于 DirectShow 的 PDA 视频传输系统

如图 1 中所示，在发送端 Video Capture 过滤器通过 Camera Driver 摄像头底层驱动过滤器采集到视频图像后交由 Video Encoder 进行视频图像压缩(如 H.264)，经压缩之后的图像通过 JRTPvideo Send

① 基金项目:2008 广西研究生创新项目(2008105950810M418);广西科学研究与技术开发计划(桂攻科 0630004-5N)
收稿时间:2009-01-09

完成数据发送。在接收端 JRTPvideo Receiver 过滤器从网络上接收到视频数据后交由 Video Decoder 过滤器解码后传至 GAPI Render 进行播放。可以看出，PDA 视频传输过滤器包括 JRTPvideo Send 过滤器和 JRTPvideo Receiver 过滤器，即发送和接收过滤器。PDA 视频传输过滤器设计重点包括 Filter 构建和网络传输两个部分。在网络传输中将实现对视频帧的封装、视频切帧以及切帧打包、数据发送和接收、环形缓冲区设计。

3 PDA视频传输过滤器开发

3.1 构建 Filter

Filter 实质是个 COM 组件[1]，Com 组件是一个实现了纯虚指针接口的 C++对象，在 Windows Mobile 平台上通过创建一个 Win32 智能设备项目来建立一个基于 DLL 的 COM 组件。设计中，按照 DirectShow Filter 的开发要求需要在工程项目属性中添加链接库“Strmbase.lib、strmiids.lib、ole32.lib”进行开发环境配置。

Filter 作为一个基于 COM 的组件，必须实现一些入口函数，完成 Filter 的 DllRegisterServer(注册)、DllUnregisterServer(反注册)等功能函数。通过.def 模块定义文件来导出入口函数，并将该.def 文件加入到创建的工程中。在 Windows Mobile 平台上可以通过 Regsvrce.exe 来实现注册、反注册功能。

Filter 的 COM 特性实现都是从基类中派生出来的，在实现 Filter 的 COM 接口时需要选择合适的基类进行派生[2]。发送 Filter 作为一个推模式的 Filter，负责将上一级 Filter 传入的数据发送到网络上，设计中可直接从 CbaseFilter 基类进行派生。而接收 Filter 作为源 Filter 位于 Graph 图中的第一级，可以选择 Csource 基类进行派生。COM 组件实现的重点是其端口(Pin)特性，对于发送 Filter 位于 Graph 图中最后一级，只有一个输入端口(InputPin)，而接收 Filter 位于 Graph 图的第一级，只有一个输出端口(OutputPin)。Filter 完成的流媒体数据处理如数据的发送接收等都是在这些 pin 端口中实现的。DirectShow 同样为实现 pin 端口提供了基类进行派生，在建立 Filter 选择派生基类的同时，也就随之决定了 Pin 端口的派生基类。于是发送 Filter 的 InputPin 端口选择 CbaseInputPin 基类进行派生；接收 Filter 的 OutputPin 端口选择从 CsourceStream 基类进行派生。在端口的实现上，按照 DirectShow 关于 Filter 连接规范，InputPin 或 OutputPin 都需要向外提供媒体格式查询功能，以便进行媒体协商。即实现发送 Filter 的 CbaseInputPin 和接收 Filter 的 Csource-Stream 的 GetMediaType、CheckMediaType、SetMediaType 方法声明。

Filter 之间根据协商之后的媒体流格式进行传递数据，数据流是从 Filter 的输出 Pin 流向下游 Filter 的输入 Pin。在发送端，位于发送 Filter 上一级 Filter 输出 Pin 调用发送 Filter 输入 Pin 上的 IMemInput-Pin::Receive 方法传递视频数据至发送 Filter，于是在发送 Filter 的 Receive 方法中可以获取到流媒体数据并进行传输。

```
CVideoInputPin::Receive(IMediaSample*pSampl
e)
    {
      /* 调用基类实现的方法，pSample 指向接收到视频缓冲区*/
      HRESULT hr=CBaseInputPin:: Receive (pSa- mple);
      BYTE *pSourceBuffer;
      long lSourceSize;
      /*获得视频帧数据*/
      pSample->GetPointer(&pSourceBuffer);
      /*获得视频帧长度*/
      lSourceSize=pSample->GetActualData
Length();
      /*此处可以完成数据的发送*/
    }
```

在接收 Filter 中，作为源 Filter 从网上接收到视频数据并传递至下一级 Filter 如解码 Filter。基类的 CSourceStream::FillBuffer 方法提供将这些视频数据流填充到 Filter 的内部缓冲区中，再由其内部基类实现向下游传递。

3.2 网络传输

在上一小节分析中可以发现在发送 Filter 的 Receive 方法中可以提取到视频流数据发送到网络上；而在接收 Filter 中的 FillBuffer 方法提供了将网络上接收到的视频流填充到 Filter 接口。因此可以通过在 Receive 方法添加发送代码以及在 FillBuffer 方法添加接收代码来实现视频的网络传输。为了达到良好的网络传输效果，并缩短加快开发周期，在视频的网络传输中决定采用 JRTP 实现视频流的传输。JRTP 传输性能好独立性强开发文档丰富，是目前较常用的 RTP 协议栈。并且利用 RTCP 协议对实时数据进行监视和控制，可以提供良好的传送机制。下图显示了视频流媒体传输过滤器中的网络传输过程：

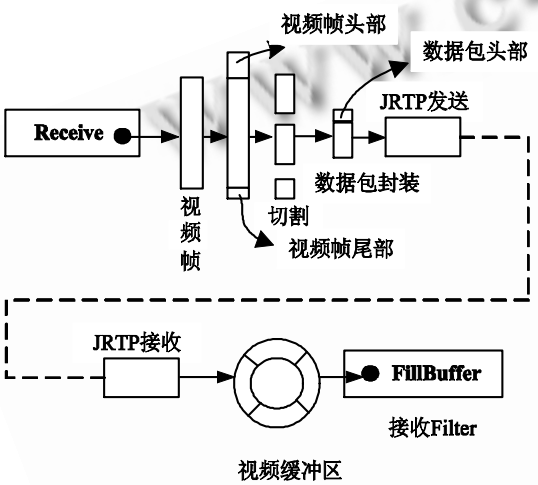

图 2　视频传输示意图

如图 2 所示在视频的传输过程中，从发送 Filter 的 Receive 方法中取出视频帧(video sample)后添加视频帧头部和尾部对视频帧进行封装，用以在接收缓冲区中对视频帧的完整性判断。经封装后的视频帧，重新根据 JRTP 的发送能力进行分段切割，经切割后的数据加上数据包头封装后利用 JRTP 进行发送。在接收端，JRTP 接收到视频数据包后去除数据包头后利用环形视频缓冲区进行保存，保存数据满帧后在接收 Filter 的 FillBuffer 函数中将完整的一帧视频填充到 Filter 的内部缓冲区中。

3.2.1 视频帧封装

从发送 Filter 的 Receive 方法中取出的视频帧需重新进行封装，封装的目的是便于在接收端对切割后的视频帧进行重新重组成帧。经封装后的视频帧报文格式如图 3 所示。

按照设计，视频帧头部起始和帧尾部分别为占 4 字节的 0xAAFE 和 4 字节的 0xFFAA。视频帧头部起始和尾部作为视频帧开始和结束的标志。2 个字节的数据包长度是封装后的整个视频帧的长度，因此其视频数据流的实际长度是该数据包长度减去 16 字节的帧头帧尾和。数据包长度之后紧接的两个字节中一个字节代表视频编码格式(如 0x 10 代表 H264 编码标准)，另一字节则代表视频的采集帧率(如 0x14 代表 30 帧率)。设计中，考虑以后音视频同步应用，留有了 4 个字节的时间戳标识该帧的编码时间。

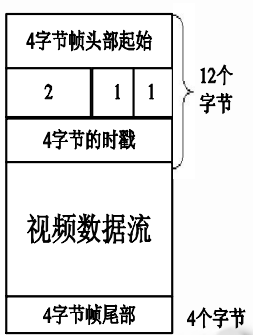

图 3　视频帧封装格式

3.2.2 视频帧切片和数据包封装

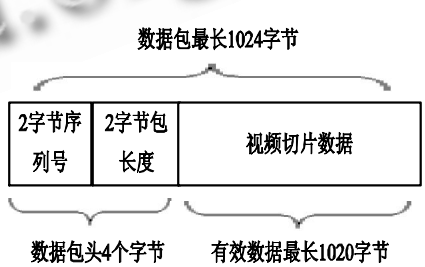

图 4　视频切片数据包封装格式

封装好的视频帧根据 JRTP 报文的发送能力进行切片，切片后的数据再次进行封包，封包格式如图 4 所示。按照在 JRTP 初始化程序中设置，JRTP 最大发送能力为 1024 个字节数据，因此如图中所示一次可以发送的最大数据报文只有 1024 个字节，其中包括

有效数据最长1020字节和定长4个字节的数据包头，数据包头含 2 个字节包长度和 2 个字节的切片报文序列号。序列号在数据包每发送一次加 1，达到最大值后重新从 0 开始计数。

3.2.3 数据包发送、接收

切片封装好的视频数据包采用 JRTP 进行发送。JRTP 开发库需要移植到 Windows Mobile 5.0 平台上。JRTP 开发库同样提供了在该平台上的编译工程，在该平台上经编译后可以得到 jrtpce.lib 和 jthreadce.lib 两个文件。其中 jrtpce.lib 是实现数据发送接收的主文件，而 jthreadce.lib 文件是多线程控制文件。在工程中添加包含这两个库后，才可实现对 JRTP 初始化、发送和接收等 API 函数接口的调用。在使用 JRTP 开发库进行发送和接收之前，当然同样需要对发送和接收参数进行初始化。初始化将完成对发送接收 IP 地址、端口、发送报文大小等参数的设置[3]。RTP 发送部分可以通过 RTPsession 的 Create 方法来建立发送渠道，并通过 SendPacket 方法实现发送任务。

RTP 接收，通过继承 OnPollThreadStep()函数来完成。当 RTP 会话建立并且增加了目标地址后，从目标地址接收到数据则重载该函数，在这个函数里面装载数据完成视频数据的接收。在 RTP 数据接收部分，一定要选择合适的重载函数来完成，避免发生内存泄露的危险。对于 windows mobile 系统终端发生内存泄露很容易造成程序和系统的崩溃。

3.2.4 视频缓冲区设计

为了对接收到的视频切帧进行重组，在接收端中设计了视频缓冲区。通过不断的设计考量，最终决定采用环形缓冲区[4]设计如图 5 所示。在 Windows Mobile 这种嵌入式平台上开发应用程序首要考虑的是存储器的利用问题。环形缓冲区设计可以有效的利用存储器空间，减少存储器的浪费。在设计中，开辟了写线程和读线程，写线程在接收到一帧数据后去除包头后写入缓冲区；读取线程则不断检测缓冲区数据，在缓存数据满一帧时立刻将数据取出。由于涉及到读写两个线程对缓冲区的处理，因此还需要解决缓冲区的读写同步问题。

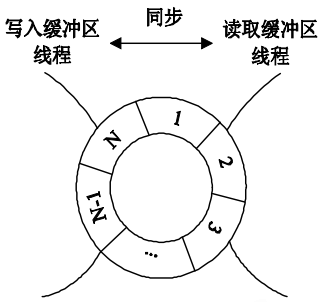

图 5　环形视频缓冲区处理

在环形缓冲区中，通过一系列的缓冲区位置指针来指导视频数据的写入和读出。在写入线程中，需要判定接收到的视频帧格式和帧切片序列号是否正确，以及视频写入的位置；读出线程不断检测视频缓冲区数据，只要满一帧则直接读出，视频满帧可以通过视频帧的帧头、帧尾以及帧长标志来进行检测。当然，读出之后需要更新缓冲区的位置指针方便数据写入。在读写线程同步方面，采用了基于用户模式的 Critical Section 临界区访问控制。在 Windows Mobile 上，对于多线程的同步处理同样具有用户模式和内核模式[5]，内核模式当然最为著名的是信号灯方法，但处理速度较慢，不适用于高速切换的缓冲区读取。速度慢也将大大影响到整个视频画面的显示效果。处于用户模式的 CriticalSection 临界区访问控制速度较快，并且操作较简易。

4 测试数据

在该 PDA 视频传输过滤器的具体测试中，我们将其应用于 PDA 的视频通话软件中如可视电话。通过 Wireshark 抓包工具可以对视频通话过程中的 RTCP 报文进行抓包分析，验证该视频传输过滤器的传输质量。在无线局域网内，三次视频通话话务中，我们抓包得到以下数据：

通过表 1 统计数据可以发现，在三个通话期间，其丢包率分别为 0.31%、0.26%、0.39%，其丢包率较低。较低的丢包率可以减少视频传输过程中的帧丢失，从表上可以发现帧丢失率基本在 1%左右，处于一个较低的水平。这样的视频传输标准基本在可接受范围之内，特别是在后期如果可以采用一些丢包恢复技

表 1　通话期间报文统计数据

	话务 1	话务 2	话务 3
发包数	321	742	1021
丢包数	1	2	4
发送帧数	112	213	356
丢失帧数	1	3	4
包往返时延	26ms	23ms	25ms
平均帧时延	37.25 ms	40.06 ms	35.84 ms

术来加以补充完善的话，该传输过滤器可以表现出较好的视频传输性能。从测试中，RTCP 报文得出的数据可以看出其包往返时延分别为 26ms、23ms、25ms，利用这些数据可计算出其平均帧传输时延分别为 37.25ms、40.06ms、35.84ms，其时延处于一个相对较低且稳定的状态。

5　结语

在 PDA 上采用 DirectShow 流媒体处理技术开发视频传输系统不仅有效节省了开发进度，并且设计的发送和接收 Filter 可以重复利用于 PDA、手机平台上的视频传输需求。凭借 DirectShow 提供的良好的处理接口、JRTP 优良的传输性能以及环形缓冲区的高效读写，PDA 上的视频传输系统表现了良好的传输性能。但仍然存在一定的缺点，具体表现在基于 DirectShow 的视频传输系统在建立传输链路的时候耗时过长，另外对于该视频缓冲区处理中未能实现视频帧乱序处理能力。相信今后随着该模块的不断完善，这些问题都可以得到解决。

参考文献

1 陆其明.DirectShow 开发指南.北京:清华大学出版社, 2003:4－7.

2 鲁漫红,孙星明,杨高波.DirectShow 图像传输过滤波的设计与实现.科学技术与工程, 2006,6(15):2281－2285.

3 王天成,尹丽萍,毛征.基于 DirectShow 的 MPEG-4 视频传输系统的研究.微计算机信息, 2007,26(11):35－37.

4 刘尉悦,郁专,武杰.基于 DirectShow 的 MPEG4 网络视频流处理系统.核电子学与探测技术, 2007,27(1):82－84.

5 傅曦,齐宇,徐骏.Windows Mobile 手机应用开发.北京:[illegible]