基于DirectShow组件技术的视音频存储系统的实现*
Implementation of Video and Audio Save System Based on DirectShow

胡凡良，宋 玲，李陶深
Hu Fanliang，Song Ling，Li Taoshen
（广西大学计算机与电子信息学院，广西南宁 530004）
（Coll. of Comp. & Elec. Info.，Guangxi Univ.，Nanning，Guangxi，530004，China）

摘要：采用Windows平台下的DirectShow组件技术，以及MPEG-4和DSP Group TrueSpeech(TM)视音频压缩算法，在Visual C++6.0开发环境下实现从摄像头捕捉视频，从麦克风捕捉音频，并将视音频数据压缩后合并输出到AVI文件。数据压缩比达到100：1，节省了存储空间，实现了视频的预览。

关键词：视音频存储 DirectShow MPEG-4 DSP Group TrueSpeech
中图法分类号：TP37

Abstract：Based on Directshow，a system is developed using MPEG-4 and DSP group truespeech (TM) encoding algorithms in visual C++6.0. The system implements video and audio real-time capture and preview and saves them as an AVI file. For its high compression ration(nearly 100：1)，it reduces memory space greatly.

Key words：video and audio save，DirectShow，MPEG-4，DSP Group TrueSpeech

随着计算机技术、通信技术和多媒体技术的飞速发展，多媒体应用已经深入到生活的各个方面，而如何获得数字视音频数据并进行压缩是这类应用的一个重要部分。在Windows平台上，实现视频捕捉的传统方法是采用VFW(Video For Windows)API函数[1]，但是这种方法不能提供流数据，无法满足Internet上的一些应用。Microsoft的DirectShow技术正是为了适应以上需要而设计的一种多媒体开发工具，其编程规范，软件可重用性强。MPEG-4是一种先进的超低码率运动图像和语言的压缩标准[2]，其视频数据的压缩率与其他算法相比占有明显优势。DSP Group TrueSpeech是一种高质量、低码率的语音压缩编码器，其输入的数字信号为8kHZ采样的16bits线性PCM码，压缩后的码率为8kbps[2]。本文采用Windows平台下的DirectShow组件技术以及MPEG-4和DSP Group TrueSpeech(TM)视、音频压缩算法，在Visual C++6.0开发环境下，实现从摄像头捕捉视频、从麦克风捕捉音频，将视音频数据压缩后合并输出到AVI文件，同时实现了视频的预览，节省存储空间。

2004-05-17收稿。
*广西科学基金[桂科基0342011]资助项目。

1 DirectShow体系结构[3]

DirectShow是Microsoft公司提供的一套在Windows平台上进行多媒体流处理的开发包，与DirectX开发包一起发布。它使用一种称为过滤器图(Filter Graph)的模型来管理整个数据流的处理过程；参与数据处理的各个功能模块称为过滤器(Filter)；各个过滤器在过滤器图中按一定的顺序连接在一起协同工作。

1.1 过滤器

DirectShow基本的构建模块是过滤器组件。DirectShow将多媒体数据的处理分为若干步骤，每一步由1个过滤器来完成。按照功能来分，过滤器大致分为：源过滤器、转换过滤器和渲染过滤器。源过滤器主要负责获取数据，数据源可以是文件、因特网、计算机里的采集卡或者数字摄像机等；转换过滤器主要负责数据的格式转换，例如数据流分离/合成、解码/编码等；渲染过滤器主要负责数据的最终去向——将数据送给显卡、声卡进行多媒体的演示，或者输出到文件。

1.2 Pin

Pin是2个过滤器相连的接口，它是从IPin这个COM对象派生来的。每个Pin都是过滤器的私

有对象，过滤器可以动态地创建 Pin、销毁 Pin、控制 Pin 的生存时间，等等。Pin 分为 2 类：输入 Pin 和输出 Pin。2 个相连的 Pin 必须是不同种类的，即输入 Pin 只能连接输出 Pin。数据在相连的 Pin 中流动，从上级过滤器到下级过滤器。2 个过滤器相连时，有一个协商过程，他们必须统一数据流的类型、缓存大小和数据传送机制等。如果协商不统一，2 个过滤器就无法连接。

1.3 过滤器图

任何用 DirectShow 开发的应用程序，都必须创建多个过滤器并进行恰当的连接，以便数据流可以从源过滤器经转换过滤器传送到渲染过滤器，被用户所使用。这些过滤器的集合称为过滤器图(Filter Graph)。图 1 是一个典型过滤器的连接。

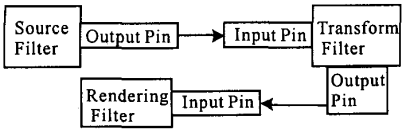

图 1 典型过滤器连接

1.4 过滤器图管理器

过滤器以及过滤器图的管理是由一个更高一级组件来完成，即过滤器图管理器，它了解所有可利用的过滤器，并能根据媒体文件的类型自动选择所需的过滤器，通过过滤器图来控制数据流。

2 基于 DirectShow 视音频系统的实现

2.1 开发环境的配置

使用 DirectShow 组件开发应用程序时，首先安装 Microsoft DirectX SDK，并且将 DX SDK 下的 Include 和 Lib 目录配置到 VC 的系统目录中，放在标准的 VC 目录之前，以确保编译器能够获得最新版本的源文件。另外，在开发基于 DirectShow 技术的应用程序时，需要把 Dshow.h 头文件包含在应用程序中。同时，把 Strmiids.lib 库文件加入到附加库。

2.2 视音频捕捉和压缩的实现

因为 DirectShow 过滤器都是以 COM 形式存在，所以用户在使用时必须初始化 COM Library，调用 CoInitialize 函数嵌入所有的动态链接库和资源。在程序结束时，调用 CoUninitialize 函数释放所有的动态链接库和资源。

笔者以摄像头捕捉视频数据、麦克风捕捉音频，数据压缩后合并存储到一个 AVI 文件，在 VC++6.0 环境下实现基于 DirectShow 的视音频捕捉存储的过程，图 2 是过滤器图的应用实例。本系统的视频、音频的编码分别采用 MPEG-4 和 DSP Group Truespeech 算法。

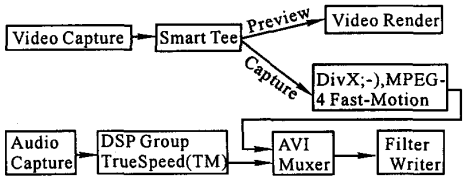

图 2 过滤器图实例

系统具体实现过程如下：

(1)创建一个基于对话框的工程 Capture。

(2)在对话框中加入一个 Static Text 控件和一个 Button 控件。

(3)为 Static Text 控件添加一个控制变量 m_VideoWin，用来实现视频的实时预览；为 Button 控件添加单击函数实现视音频的捕捉功能。

2.3 实现的部分代码

(1)在 OnInitDialog()函数中实现过滤器的创建：

```
//Create the Filter Graph Manager.
CoCreateInstance (CLSID_FilterGraph, NULL, CLSCTX_INPROC, IID_IGraphBuilder, (void **)&pGraph);
//Create the Capture Graph Builder.
CoCreateInstance(CLSID_CaptureGraphBuilder2, NULL, CLSCTX_INPROC,
IID_ICaptureGraphBuilder2, (void **)&pBuilder);
pBuilder-> SetFiltergraph (pGraph); // Associate the graph with the builder.
pGraph-> QueryInterface(IID_IMediaControl, (void **)&pControl); //获得媒体控制接口
CoCreateInstance(CLSID_SystemDeviceEnum, NULL, CLSCTX_INPROC,
IID_ICreateDevEnum, (void **)&pDevEnum);//创建设备枚举指针
```

(2)枚举视频捕捉设备并添加到过滤器图中.

```
pDevEnum-> CreateClassEnumerator(CLSID_VideoInputDeviceCategory, &pClassEnum, 0);
while (pClassEnum-> Next (1, &pMoniker, &cFetched) == S_OK)
{// Bind the first moniker to a filter object.
pMoniker-> BindToObject(0, 0, IID_IBaseFilter, (void **)&pVideoSrc);}
pGraph-> AddFilter(pVideoSrc, NULL);
```

(3)枚举视频压缩算法，直到找到 DivX;-)

MPEG-4 Fast-Motion 算法为止并将其添加到过滤器图中：

```
pDevEnum-> CreateClassEnumerator(CLSID_VideoCompressorCategory, &pClassEnum, 0); while (S_OK == pClassEnum-> Next(1,&pMoniker, NULL))
{
if (lstrcmp("DivX ;-) MPEG-4 Fast-Motion", friendName)==0)
// friendname 为过滤器的名字,选用 MPEG-4 视频压缩算法
pMoniker-> BindToObject(NULL,NULL,IID_IBaseFilter,void * * )&pVideoComp);}
pGraph-> AddFilter(pVideoComp,NULL);
```

(4)枚举音频捕捉设备并添加到过滤器图中：

```
pDevEnum-> CreateClassEnumerator(CLSID_AudioInputDeviceCategory,&pClassEnum,0);
while (pClassEnum-> Next(1, &pMoniker, &cFetched) == S_OK)
{ // Bind the first moniker to a filter object.
pMoniker-> BindToObject(0, 0, IID_IBaseFilter,(void * * )&pAudioSrc);}
pGraph-> AddFilter(pAudioSrc,NULL);
```

(5)枚举音频压缩算法，直到找到 DSP Group TrueSpeech(TM)算法为止并将其添加到过滤器图中：

```
pDevEnum-> CreateClassEnumerator(CLSID_AudioCompressorCategory,&pClassEnum,0);
while (S_OK == pClassEnum-> Next(1, &pMoniker, NULL))
{
if (lstrcmp("DSP Group TrueSpeech(TM)", friendName)==0)
//选用 DSP Group TrueSpeech(TM)音频压缩算法
pMoniker-> BindToObject(NULL, NULL, IID_IBaseFilter,(void * * )&pAudioComp);}
pGraph-> AddFilter(pAudioComp,NULL);
```

(6)OnCaptureButton()函数实现捕捉，过滤器通过 DirectShow 提供的智能连接来协同工作：

```
pBuilder-> SetOutputFileName
(&MEDIASUBTYPE_Avi, L"E:\Example.avi", &ppf, NULL);//输出捕捉后的文件
pBuilder-> RenderStream(&PIN_CATEGORY_CAPTURE,&MEDIATYPE_Video, pVideoSrc,pVideoComp,ppf); //视频采样压缩
pBuilder-> RenderStream(&PIN_CATEGORY_CAPTURE,&MEDIATYPE_Audio, pAudioSrc,pAudioComp,ppf); //音频采样压缩
pBuilder-> RenderStream(&PIN_CATEGORY_PREVIEW, &MEDIATYPE_Video, pVideoSrc, NULL,NULL);//视频实时预览
pControl-> Run();
```

2.4 系统运行效果

笔者采用 MPEG-4 视频压缩算法对视频数据进行压缩，压缩比达到 120∶1；利用 DSP Group TrueSpeech(TM)的音频压缩算法对音频数据进行压缩，压缩比约为 16∶1；把视频和音频数据压缩后合并输出，压缩比达到 100∶1，1 个 1 min 长度的 AVI 文件只有 1.5M，节省存储空间的同时并获得令人满意的效果。

3 结束语

本文探讨在 VC++6.0 开发环境采用 DirectShow 组件技术开发视音频存储系统的具体过程。利用 MPEG4 视频压缩算法和 DSP Group TrueSpeech(TM)音频压缩算法对采集到的视频、音频数据进行压缩，获得理想效果。实践证明，基于 DirectShow 组件技术开发多媒体应用程序是一种高效的新方法。当需要在网络上传输视音频多媒体信息时，通常采用的方法是视、音频信号占用不同通道独立传输，但这样在实现音形同步方面要做大量工作以保证传输的 QoS。本系统运行测试结果表明，运用性能优越的视音频编码器，把捕捉到的视频和音频数据压缩后合并输出，数据压缩率相当可观。基于这样的前提，我们设想把视频和音频数据压缩后使用同一通道合并传输，到接收端再进行分离，这样可简化音形同步问题，也是我们后续研究工作。

参考文献：

1 Microsoft. Microsoft MSDN Library. http://www.microsoft.com,2003-05-05.

2 谢亚光，章 琦，刘济林. 基于 Microsoft DirectShow 的多媒体应用程序开发. 计算机应用研究，2003，4：72～74.

3 陆其明. DirectShow 开发指南. 北京：清华大学出版社，2003.

（责任编辑：黎贞崇）