原 最简单的基于FFMPEG的音频编码器（PCM编码为AAC）

2014年05月11日 00:14:28　阅读数：67385

本文介绍一个最简单的基于FFMPEG的音频编码器。该编码器实现了PCM音频采样数据编码为AAC的压缩编码数据。编码器代码十分简单，但是每一行代码都很重要。通过看本编码器的源代码，可以了解FFMPEG音频编码的流程。

本程序使用最新版的类库（编译时间为2014.5.6），开发平台为VC2010。所有的配置都已经做好，只需要运行就可以了。

流程（2014.9.29更新）

下面附一张使用FFmpeg编码音频的流程图。使用该流程，不仅可以编码AAC的音频，而且可以编码MP3，MP2等等各种FFmpeg支持的音频。图中蓝色背景的函数是实际输出数据的函数。浅绿色的函数是音频编码的函数。

简单介绍一下流程中各个函数的意义：

av_register_all()：注册FFmpeg所有编解码器。

avformat_alloc_output_context2()：初始化输出码流的AVFormatContext。

avio_open()：打开输出文件。

av_new_stream()：创建输出码流的AVStream。

avcodec_find_encoder()：查找编码器。

avcodec_open2()：打开编码器。

avformat_write_header()：写文件头（对于某些没有文件头的封装格式，不需要此函数。比如说MPEG2TS）。

avcodec_encode_audio2()：编码音频。即将AVFrame（存储PCM采样数据）编码为AVPacket（存储AAC，MP3等格式的码流数据）。

av_write_frame()：将编码后的视频码流写入文件。

av_write_trailer()：写文件尾（对于某些没有文件头的封装格式，不需要此函数。比如说MPEG2TS）。

代码

```cpp
/**
 *最简单的基于FFmpeg的音频编码器
 *Simplest FFmpeg Audio Encoder
 *
 *雷霄骅 Lei Xiaohua
 *leixiaohua1020@126.com
 *中国传媒大学/数字电视技术
 *Communication University of China / Digital TV Technology
 *http://blog.csdn.net/leixiaohua1020
 *
 *本程序实现了音频PCM采样数据编码为压缩码流（MP3，WMA，AAC等）。
 *是最简单的FFmpeg音频编码方面的教程。
 *通过学习本例子可以了解FFmpeg的编码流程。
 *This software encode PCM data to AAC bitstream.
 *It's the simplest audio encoding software based on FFmpeg.
 *Suitable for beginner of FFmpeg
 */

#include <stdio.h>

#define __STDC_CONSTANT_MACROS

#ifdef _WIN32
//Windows
extern "C"
{
#include "libavcodec/avcodec.h"
#include "libavformat/avformat.h"
};
#else
//Linux...
#ifdef __cplusplus
extern "C"
{
#endif
```

```
#include <libavcodec/avcodec.h>
#include <libavformat/avformat.h>
#ifdef __cplusplus
};
#endif
#endif


int flush_encoder(AVFormatContext *fmt_ctx,unsigned int stream_index){
    int ret;
    int got_frame;
    AVPacket enc_pkt;
    if (!(fmt_ctx->streams[stream_index]->codec->codec->capabilities &
        CODEC_CAP_DELAY))
        return 0;
    while (1) {
        enc_pkt.data = NULL;
        enc_pkt.size = 0;
        av_init_packet(&enc_pkt);
        ret = avcodec_encode_audio2 (fmt_ctx->streams[stream_index]->codec, &enc_pkt,
            NULL, &got_frame);
        av_frame_free(NULL);
        if (ret < 0)
            break;
        if (!got_frame){
            ret=0;
            break;
        }
        printf("Flush Encoder: Succeed to encode 1 frame!\tsize:%5d\n",enc_pkt.size);
        /* mux encoded frame */
        ret = av_write_frame(fmt_ctx, &enc_pkt);
        if (ret < 0)
            break;
    }
    return ret;
}

int main(int argc, char* argv[])
{
    AVFormatContext* pFormatCtx;
    AVOutputFormat* fmt;
    AVStream* audio_st;
    AVCodecContext* pCodecCtx;
    AVCodec* pCodec;

    uint8_t* frame_buf;
    AVFrame* pFrame;
    AVPacket pkt;

    int got_frame=0;
    int ret=0;
    int size=0;

    FILE *in_file=NULL;                         //Raw PCM data
    int framenum=1000;                          //Audio frame number
    const char* out_file = "tdjm.aac";          //Output URL
    int i;

    in_file= fopen("tdjm.pcm", "rb");

    av_register_all();

    //Method 1.
    pFormatCtx = avformat_alloc_context();
    fmt = av_guess_format(NULL, out_file, NULL);
    pFormatCtx->oformat = fmt;


    //Method 2.
    //avformat_alloc_output_context2(&pFormatCtx, NULL, NULL, out_file);
    //fmt = pFormatCtx->oformat;

    //Open output URL
    if (avio_open(&pFormatCtx->pb,out_file, AVIO_FLAG_READ_WRITE) < 0){
        printf("Failed to open output file!\n");
        return -1;
    }

    audio_st = avformat_new_stream(pFormatCtx, 0);
    if (audio_st==NULL){
        return -1;
    }
    pCodecCtx = audio_st->codec;
    pCodecCtx->codec_id = fmt->audio_codec;
    pCodecCtx->codec_type = AVMEDIA_TYPE_AUDIO;
    pCodecCtx->sample_fmt = AV_SAMPLE_FMT_S16;
    pCodecCtx->sample_rate= 44100;
    pCodecCtx->channel_layout=AV_CH_LAYOUT_STEREO;
    pCodecCtx->channels = av_get_channel_layout_nb_channels(pCodecCtx->channel_layout);
    pCodecCtx->bit_rate = 64000;

```

```
    //Show some information
    av_dump_format(pFormatCtx, 0, out_file, 1);

    pCodec = avcodec_find_encoder(pCodecCtx->codec_id);
    if (!pCodec){
        printf("Can not find encoder!\n");
        return -1;
    }
    if (avcodec_open2(pCodecCtx, pCodec,NULL) < 0){
        printf("Failed to open encoder!\n");
        return -1;
    }
    pFrame = av_frame_alloc();
    pFrame->nb_samples= pCodecCtx->frame_size;
    pFrame->format= pCodecCtx->sample_fmt;

    size = av_samples_get_buffer_size(NULL, pCodecCtx->channels,pCodecCtx->frame_size,pCodecCtx->sample_fmt, 1);
    frame_buf = (uint8_t *)av_malloc(size);
    avcodec_fill_audio_frame(pFrame, pCodecCtx->channels, pCodecCtx->sample_fmt,(const uint8_t*)frame_buf, size, 1);

    //Write Header
    avformat_write_header(pFormatCtx,NULL);

    av_new_packet(&pkt,size);

    for (i=0; i<framenum; i++){
        //Read PCM
        if (fread(frame_buf, 1, size, in_file) <= 0){
            printf("Failed to read raw data! \n");
            return -1;
        }else if(feof(in_file)){
            break;
        }
        pFrame->data[0] = frame_buf;  //PCM Data

        pFrame->pts=i*100;
        got_frame=0;
        //Encode
        ret = avcodec_encode_audio2(pCodecCtx, &pkt,pFrame, &got_frame);
        if(ret < 0){
            printf("Failed to encode!\n");
            return -1;
        }
        if (got_frame==1){
            printf("Succeed to encode 1 frame! \tsize:%5d\n",pkt.size);
            pkt.stream_index = audio_st->index;
            ret = av_write_frame(pFormatCtx, &pkt);
            av_free_packet(&pkt);
        }
    }

    //Flush Encoder
    ret = flush_encoder(pFormatCtx,0);
    if (ret < 0) {
        printf("Flushing encoder failed\n");
        return -1;
    }

    //Write Trailer
    av_write_trailer(pFormatCtx);

    //Clean
    if (audio_st){
        avcodec_close(audio_st->codec);
        av_free(pFrame);
        av_free(frame_buf);
    }
    avio_close(pFormatCtx->pb);
    avformat_free_context(pFormatCtx);

    fclose(in_file);

    return 0;
}
```

结果

程序运行完成后，会将一个PCM采样数据文件（*.pcm）编码为AAC码流文件（*.aac）。

下载

simplest ffmpeg audio encoder

项目主页

SourceForge： https://sourceforge.net/projects/simplestffmpegaudioencoder/

Github： https://github.com/leixiaohua1020/simplest_ffmpeg_audio_encoder

开源中国： http://git.oschina.net/leixiaohua1020/simplest_ffmpeg_audio_encoder

CSDN工程下载地址：

http://download.csdn.net/detail/leixiaohua1020/7324091

PUDN工程下载地址：

http://www.pudn.com/downloads644/sourcecode/multimedia/detail2605236.html

更新-1.1 (2015.2.13)==========

这次考虑到了跨平台的要求，调整了源代码。经过这次调整之后，源代码可以在以下平台编译通过：

VC++：打开sln文件即可编译，无需配置。

cl.exe：打开compile_cl.bat即可命令行下使用cl.exe进行编译，注意可能需要按照VC的安装路径调整脚本里面的参数。编译命令如下。

```plain
::VS2010 Environment
call "D:\Program Files\Microsoft Visual Studio 10.0\VC\vcvarsall.bat"
::include
@set INCLUDE=include;%INCLUDE%
::lib
@set LIB=lib;%LIB%
::compile and link
cl simplest_ffmpeg_audio_encoder.cpp /link avcodec.lib avformat.lib avutil.lib ^
avdevice.lib avfilter.lib postproc.lib swresample.lib swscale.lib /OPT:NOREF
```

MinGW：MinGW命令行下运行compile_mingw.sh即可使用MinGW的g++进行编译。编译命令如下。

```cpp
g++ simplest_ffmpeg_audio_encoder.cpp -g -o simplest_ffmpeg_audio_encoder.exe \
-I /usr/local/include -L /usr/local/lib -lavformat -lavcodec -lavutil
```

GCC：Linux或者MacOS命令行下运行compile_gcc.sh即可使用GCC进行编译。编译命令如下。

```cpp
gcc simplest_ffmpeg_audio_encoder.cpp -g -o simplest_ffmpeg_audio_encoder.out \
-I /usr/local/include -L /usr/local/lib -lavformat -lavcodec -lavutil
```

PS：相关的编译命令已经保存到了工程文件夹中

CSDN下载地址： http://download.csdn.net/detail/leixiaohua1020/8445209

SourceForge上已经更新。

文章标签： ffmpeg pcm aac 编码 音频

个人分类： FFMPEG 我的开源项目

所属专栏： FFmpeg