

原 ffmpeg 从内存中读取数据（或将数据输出到内存）

2013年10月24日 00:03:25 阅读数：36291

更新记录（2014.7.24）：

- 1.为了使本文更通俗易懂，更新了部分内容，将例子改为从内存中打开。
- 2.增加了将数据输出到内存的方法。

从内存中读取数据

ffmpeg一般情况下支持打开一个本地文件，例如"C:\test.avi"

或者是一个流媒体协议的URL，例如"rtmp://222.31.64.208/vod/test.flv"

其打开文件的函数是avformat_open_input()，直接将文件路径或者流媒体URL的字符串传递给该函数就可以了。

但其是否支持从内存中读取数据呢？这个问题困扰了我很长时间。当时在做项目的时候，通过Winpcap抓取网络上的RTP包，打算直接送给ffmpeg进行解码。一直没能找到合适的方法。因为抓取的数据包是存在内存中的，所以无法传递给avformat_open_input()函数其路径（根本没有路径= =）。当然也可以将抓取的数据报存成文件，然后用ffmpeg打开这个文件，但是这样的话，程序的就太难控制了。

后来经过分析ffmpeg的源代码，发现其竟然是可以从内存中读取数据的，代码很简单，如下所示：

```
[cpp]
1. AVFormatContext *ic = NULL;
2. ic = avformat_alloc_context();

[cpp]
1. unsigned char * iobuffer=(unsigned char *)av_malloc(32768);
2. AVIOContext *avio =avio_alloc_context(iobuffer, 32768,0,NULL,fill_iobuffer,NULL,NULL);
3. ic->pb=avio;
4. err = avformat_open_input(&ic, "nothing", NULL, NULL);
```

关键要在avformat_open_input()之前初始化一个AVIOContext，而且将原本的AVFormatContext的指针pb（AVIOContext类型）指向这个自行初始化AVIOContext。当自行指定了AVIOContext之后，avformat_open_input()里面的URL参数就不起作用了。示例代码开辟了一块空间iobuffer作为AVIOContext的缓存。

fill_iobuffer则是将数据读取至iobuffer的回调函数。fill_iobuffer()形式（参数，返回值）是固定的，是一个回调函数，如下所示（只是个例子，具体怎么读取数据可以自行设计）。示例中回调函数将文件中的内容通过fread()读入内存。

```
[cpp]
1. //读取数据的回调函数-----
2. //AVIOContext使用的回调函数！
3. //注意：返回值是读取的字节数
4. //手动初始化AVIOContext只需要两个东西：内容来源的buffer，和读取这个Buffer到Ffmpeg中的函数
5. //回调函数，功能就是：把buf_size字节数据送入buf即可
6. //第一个参数(void *opaque)一般情况下可以不用
7. int fill_iobuffer(void * opaque,uint8_t *buf, int bufsize){
8.     if(!feof(fp_open)){
9.         int true_size=fread(buf,1,buf_size,fp_open);
10.        return true_size;
11.    }else{
12.        return -1;
13.    }
14. }
```

整体结构大致如下：

```
[cpp]
1. FILE *fp_open;
2.
3. int fill_iobuffer(void *opaque, uint8_t *buf, int buf_size){
4.     ...
5. }
6.
7. int main(){
8.     ...
9.     fp_open=fopen("test.h264","rb+");
10.    AVFormatContext *ic = NULL;
11.    ic = avformat_alloc_context();
12.    unsigned char * iobuffer=(unsigned char *)av_malloc(32768);
13.    AVIOContext *avio =avio_alloc_context(iobuffer, 32768,0,NULL,fill_iobuffer,NULL,NULL);
14.    ic->pb=avio;
15.    err = avformat_open_input(&ic, "nothing", NULL, NULL);
16.    ...//解码
17. }
```

将数据输出到内存

和从内存中读取数据类似，ffmpeg也可以将处理后的数据输出到内存。

回调函数如下示例，可以将输出到内存的数据写入到文件中。

```
[cpp]
1. //写文件的回调函数
2. int write_buffer(void *opaque, uint8_t *buf, int buf_size){
3.     if(!feof(fp_write)){
4.         int true_size=fwrite(buf,1,buf_size,fp_write);
5.         return true_size;
6.     }else{
7.         return -1;
8.     }
9. }
```

主函数如下所示。

```
[cpp]
1. FILE *fp_write;
2.
3. int write_buffer(void *opaque, uint8_t *buf, int buf_size){
4.     ...
5. }
6.
7. main(){
8.     ...
9.     fp_write=fopen("src01.h264","wb+"); //输出文件
10.    ...
11.    AVFormatContext* ofmt_ctx=NULL;
12.    avformat_alloc_output_context2(&ofmt_ctx, NULL, "h264", NULL);
13.    unsigned char* outbuffer=(unsigned char*)av_malloc(32768);
14.
15.    AVIOContext *avio_out =avio_alloc_context(outbuffer, 32768,0,NULL,NULL,write_buffer,NULL);
16.
17.    ofmt_ctx->pb=avio_out;
18.    ofmt_ctx->flags=AVFMT_FLAG_CUSTOM_IO;
19.    ...
20. }
```

版权声明：本文为博主原创文章，未经博主允许不得转载。 <https://blog.csdn.net/leixiaohua1020/article/details/12980423>

文章标签： [ffmpeg](#) [内存](#) [数据](#) [avformat_open_input](#) [回调](#)

个人分类： [FFMPEG](#)

所属专栏： [FFmpeg](#)

此PDF由spygg生成,请尊重原作者版权!!!

我的邮箱:liushidc@163.com