DoubleLi
qq: 517712484 wx: ldbgliet

2021年10月

日	一	二	三	四	五	六
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

公告

昵称: DoubleLi
园龄: 11年9个月
粉丝: 2080
关注: 29
+加关注

搜索

找找看
谷歌搜索

常用链接
我的随笔
我的评论
我的参与
最新评论
我的标签

随笔分类 (5164)
android(2)
ASP.NET(30)
ASP.NET MVC(11)
Boost(118)
c#(10)
C++/C(778)
c++11(15)
cmake/autotool(66)
com/ATL/Activex(75)
Css(16)
CxImage(12)
darwin stream server(3)
DataBase(32)
DirectX(16)
Extjs(13)
更多

随笔档案 (3864)
2021年10月(33)
2021年9月(4)
2021年8月(10)
2021年7月(43)
2021年6月(1)
2021年5月(29)
2021年4月(15)
2021年3月(13)
2021年2月(96)
2021年1月(47)
2020年12月(2)
2020年11月(27)
2020年10月(44)
2020年9月(14)
2020年8月(4)
更多

文章分类 (2)
SilverLight(1)
sql server(1)

FFMPEG内存操作（二）从内存中读取数及数据格式的转换

相关博客列表：

FFMPEG内存操作（一） avio_reading.c 回调读取数据到内存解析

FFMPEG内存操作（二）从内存中读取数及数据格式的转换

FFmpeg内存操作（三）内存转码器

在雷神的《最简单的基于FFmpeg的内存读写例子（内存播放器）》中，它是设计回调函数从输入文件中读取数据。与FFMPEG 官方给出的avio_reading.c不同的是，雷神给的例子是当需要数据的时候，回调函数才去从输入文件读取数据，而avio_reading.c 则是直接全部数据读取到内存中待后面处理。

在我的这个实例中，我是将读取到的输入文件解码成YUV420P数据格式。同时可以通过设置不同的数据格式和尺寸实现输出图像的拉伸缩小或是数据格式的装换。

```
/*=============================================================================
#     FileName: ffmpeg_mem_read.c
#         Desc: an example of ffmpeg read from memory
#       Author: licaibiao
#   LastChange: 2017-03-21
=============================================================================*/
#include <stdio.h>

#define __STDC_CONSTANT_MACROS
#include "avcodec.h"
#include "avformat.h"
#include "swscale.h"

//#define QUARTER_SHOW

charchar *input_name = "cuc60anniversary_start.ts";

FILEFILE *fp_open = NULL;

int read_buffer(voidvoid *opaque, uint8_t *buf, int buf_size){
    if(!feof(fp_open)){
        int true_size=fread(buf,1,buf_size,fp_open);
        return true_size;
    }else{
        return -1;
    }
}

int main(int argc, char* argv[])
{
    AVFormatContext *pFormatCtx;
    AVCodecContext  *pCodecCtx;
    AVCodec         *pCodec;
    AVIOContext     *avio;
    AVFrame         *pFrame;
    AVFrame         *pFrameYUV;
    AVPacket        *packet;

    struct SwsContext *img_convert_ctx;

    int             videoindex;
    int             i;
    int             ret;
    int             got_picture;
    int             y_size;
    unsigned char   *aviobuffer;
    FILE            *fp_yuv;

```

```
49.
50.     //fp_open = fopen(argv[1],"rb+");
51.     fp_open = fopen(input_name, "rb+");
52.     fp_yuv  = fopen("output.yuv", "wb+");
53.
54.     av_register_all();
55.     pFormatCtx = avformat_alloc_context();
56.
57.     aviobuffer=(unsigned charchar *)av_malloc(32768);
58.     avio = avio_alloc_context(aviobuffer, 32768,0,NULL,read_buffer,NULL,NULL);
59.
60.     /* Open an input stream and read the header. */
61.     pFormatCtx->pb = avio;
62.     if(avformat_open_input(&pFormatCtx,NULL,NULL,NULL)!=0){
63.         printf("Couldn't open input stream.\n");
64.         return -1;
65.     }
66.
67.     /* Read packets of a media file to get stream information. */
68.     if(avformat_find_stream_info(pFormatCtx,NULL)<0){
69.         printf("Couldn't find stream information.\n");
70.         return -1;
71.     }
72.
73.     videoindex = -1;
74.     for(i=0; i<pFormatCtx->nb_streams; i++) {
75.         if(pFormatCtx->streams[i]->codec->codec_type==AVMEDIA_TYPE_VIDEO){
76.             videoindex=i;
77.             break;
78.         }
79.     }
80.
81.     if(videoindex==-1){
82.         printf("Didn't find a video stream.\n");
83.         return -1;
84.     }
85.
86.     av_dump_format(pFormatCtx, 0, input_name, 0);
87.
88.     /* Find a registered decoder with a matching codec ID */
89.     pCodecCtx = pFormatCtx->streams[videoindex]->codec;
90.     pCodec    = avcodec_find_decoder(pCodecCtx->codec_id);
91.     if(pCodec==NULL){
92.         printf("Codec not found.\n");
93.         return -1;
94.     }
95.
96.     /* Initialize the AVCodecContext to use the given AVCodec */
97.     if(avcodec_open2(pCodecCtx, pCodec,NULL)<0){
98.         printf("Could not open codec.\n");
99.         return -1;
100.    }
101.
102.    pFrame    = av_frame_alloc();
103.    pFrameYUV = av_frame_alloc();
104.    uint8_t *out_buffer=(uint8_t *)av_malloc(avpicture_get_size(AV_PIX_FMT_YUV420P, pCodecCtx->width, pCodecCtx->height));
105.    avpicture_fill((AVPicture *)pFrameYUV, out_buffer, AV_PIX_FMT_YUV420P, pCodecCtx->width, pCodecCtx->height);
106.
107.    /*  Allocate and return an SwsContext. */
108.    /* srcW:源图像的宽
109.     * srcH:源图像的高
110.     * srcFormat:源图像的像素格式
111.     * dstW:目标图像的宽
112.     * dstH:目标图像的高
113.     * dstFormat:目标图像的像素格式
114.     * flags:设定图像拉伸使用的算法
115.     */
116. #ifndef QUARTER_SHOW
117.     img_convert_ctx = sws_getContext(pCodecCtx->width, pCodecCtx->height, pCodecCtx->pix_fmt, pCodecCtx->width, pCodecCtx->height, AV_PIX_FMT_YUV420P, SWS_BICUBIC, NULL, NULL, NULL);
```

5. Re:如何使用UDP进行跨网段广播
主机A：192.168.3.100 子网掩码 255.255.0.0（手动临时修改） 主机B：192.168.120.100 子网掩码 255.255.255.0 主机A 广播 192.168.255...

--zzhilling

```
#else
    img_convert_ctx = sws_getContext(pCodecCtx->width, pCodecCtx->height, pCodecCtx->pix_fmt, pCodecCtx->width/2, pCodecCtx->height/2, AV_PIX_FMT_YUV420P, SWS_BICUBIC, NULL, NULL, NULL);
    printf("out frame  width = %d, height = %d \n", pCodecCtx->width/2,pCodecCtx->height/2);
#endif
    packet = (AVPacket *)av_malloc(sizeof(AVPacket));
    while(av_read_frame(pFormatCtx, packet) >= 0){
        if(packet->stream_index == videoindex){
            ret = avcodec_decode_video2(pCodecCtx, pFrame, &got_picture, packet);
            if(ret < 0){
                printf("Decode Error.\n");
                return -1;
            }
            if(got_picture){
                /* Scale the image slice in srcSlice and put the resulting scaled slice in the image in dst. 图像处理函数 */
                /* c             the scaling context previously created with  sws_getContext()
                 * srcSlice      the array containing the pointers to the planes of the source slice
                 * srcStride     the array containing the strides for each plane of
                 * srcSliceY     the position in the source image of the slice to process, that is the number (counted starting from
                                 zero) in the image of the first row of the slice 输入图像数据的第多少列开始逐行扫描, 通常设为0
                 * srcSliceH     the height of the source slice, that is the number of rows in the slice 为需要扫描多少行, 通常为输入图像数据的高度
                 * dst           the array containing the pointers to the planes of the destination image
                 * dstStride     the array containing the strides for each plane of the destination image
                 */
                sws_scale(img_convert_ctx, (const uint8_t* const*)pFrame->data, pFrame->linesize, 0, pCodecCtx->height, pFrameYUV->data, pFrameYUV->linesize);

#ifndef QUARTER_SHOW
               // printf("pFrameYUV->height   = %d\n ",pFrameYUV->height);
               // printf("pFrameYUV->width    = %d\n ", pFrameYUV->width);
               // printf("pFrameYUV->pkt_size = %d\n ",pFrameYUV->pkt_size);

                y_size = pCodecCtx->width*pCodecCtx->height;
                fwrite(pFrameYUV->data[0],1,y_size,fp_yuv);    //Y
                fwrite(pFrameYUV->data[1],1,y_size/4,fp_yuv);  //U
                fwrite(pFrameYUV->data[2],1,y_size/4,fp_yuv);  //V
#else
                for(i=0; i<pCodecCtx->height/2; i++){
                    fwrite(pFrameYUV->data[0]+pCodecCtx->width * i ,1,pCodecCtx->width/2,fp_yuv);
                }
                for(i=0; i<pCodecCtx->height/2; i = i + 2){
                    fwrite(pFrameYUV->data[1]+pCodecCtx->width * i/4 ,1,pCodecCtx->width/4,fp_yuv);
                }

                for(i=0; i<pCodecCtx->height/2; i = i + 2 ){
                    fwrite(pFrameYUV->data[2]+pCodecCtx->width * i/4 ,1,pCodecCtx->width/4,fp_yuv);
                }
#endif

            }
        }
        av_free_packet(packet);
    }
    sws_freeContext(img_convert_ctx);

    fclose(fp_yuv);
    fclose(fp_open);
    av_free(out_buffer);
    av_free(pFrameYUV);
    avcodec_close(pCodecCtx);
    avformat_close_input(&pFormatCtx);

    return 0;
}
```

正常操作是将输入文件解码成YUV数据。

当我们对sws_getContext 设置不同参数的时候，我们可以得到不同的输出数据。在这里需要注意几点：

（1）out_buffer 的大小要跟着sws_getContext 参数的设置而改变，如果out_buffer分配得比输出数据小，会出现内存溢出问题。

（2）sws_scale 函数中，我们一般设置从输入数据的第0行开始扫描,扫描的高度（也就是行数）一般是输入数据的高度。

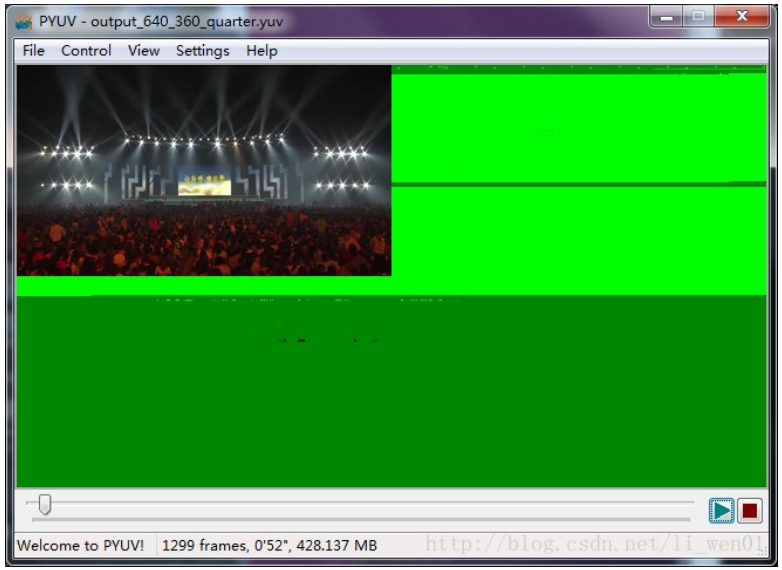
（3）得到的YUV数据是储存在pFrameYUV->data 的三个分量里，需要分开读取。

举例：

在sws_getContext 中设置输出格式的宽和高都是输入格式的一半。正常的显示应该是图像变为了原来的1/4大小。如果我们还是按原图尺寸提取输出数据：

```
 y_size = pCodecCtx->width*pCodecCtx->height;
fwrite(pFrameYUV->data[0],1,y_size,fp_yuv);    //Y
fwrite(pFrameYUV->data[1],1,y_size/4,fp_yuv);  //U
fwrite(pFrameYUV->data[2],1,y_size/4,fp_yuv);  //V
```

得到的图像将会是：

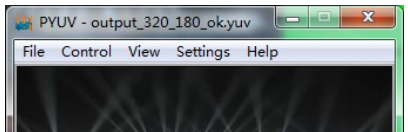


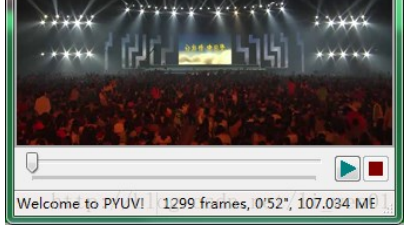
如上图，图像显示是正常的，但同时也提取了很多的无用数据，实际图像并没有缩小到原来的1/4

所以我们需要自己对输出数据再重组：

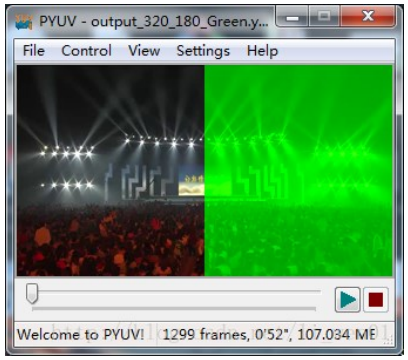
```
for(i=0; i<pCodecCtx->height/2; i++){
    fwrite(pFrameYUV->data[0]+pCodecCtx->width * i ,1,pCodecCtx->width/2,fp_yuv);
}
for(i=0; i<pCodecCtx->height/2; i = i + 2){
    fwrite(pFrameYUV->data[1]+pCodecCtx->width * i/4 ,1,pCodecCtx->width/4,fp_yuv);
}

for(i=0; i<pCodecCtx->height/2; i = i + 2 ){
    fwrite(pFrameYUV->data[2]+pCodecCtx->width * i/4 ,1,pCodecCtx->width/4,fp_yuv);
}
```

分别重新提取了Y U V 三个分量上的数据，得到图像如下：

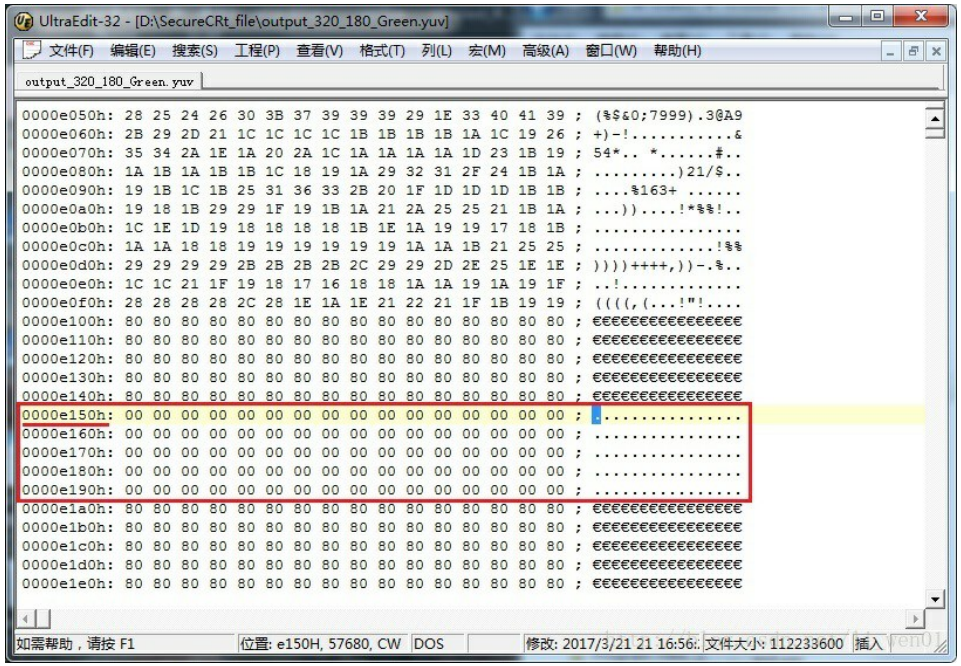




在读取输出数据的时候，如果发现读取的输出数据显示有问题，最好是使用UltraEdit 工具对某帧数据进行分析，先确认输出的一帧数据是否正常，然后才进行视频格式的转换。比如出现下面的这种情况



从图像中大概能分析出来，Y分量数据是正常的，图像右半边的UV分量可能丢失。使用UltraEdit查看原始数据，如下图。我们可以看到从地址0xe150开始有一段是全0的数据，因此我们可以到我们的程序中去检查是否UV分量读取错误。



总结：我们直接使用sws_getContext和sws_scale 可以直接对图像进行拉伸和缩放，同时可以进行数据格式的转换，只需要设置sws_getContext 就可以了，非常简单。但是需要注意，输出的数据需要我们自己重新组合，否则读取到的输出数据很有可能出错。

from:http://blog.csdn.net/li_wen01/article/details/64904586

分类: ffmpeg、ffplay

好文要顶 关注我 收藏该文

DoubleLi
关注 - 29
粉丝 - 2080
+加关注

0 推荐 0 反对

« 上一篇： FFmpeg内存操作（三）内存转码器
» 下一篇： FFMPEG内存操作（一） avio_reading.c 回调读取数据到内存解析

posted on 2017-08-11 15:41 DoubleLi 阅读(1527) 评论(0) 编辑 收藏 举报

刷新评论 刷新页面 返回顶部

编辑推荐：
· 聊聊我在微软外服的工作经历及一些个人见解
· 死磕 NIO — Reactor 模式就一定意味着高性能吗？
· 消息队列那么多，为什么建议深入了解下RabbitMQ？
· 技术管理进阶——管人还是管事？
· 以终为始：如何让你的开发符合预期

最新新闻：
· 何小鹏：争取2024年实现飞行汽车量产 价格100万以内（2021-10-24 23:35）
· 供应链危机提振美国在线二手市场 全年销售额预计超650亿美元（2021-10-24 22:00）
· CityTree：一款利用苔藓和机器学习来捕捉空气污染的设备（2021-10-24 20:53）
· 1024程序员节各家怎么过：送霸王洗发水、集体穿格子衫、盲人按摩（2021-10-24 20:00）
· 新卫星图展示泰国季风洪水所带来的巨大影响（2021-10-24 19:00）
» 更多新闻...