原 最简单的基于FFmpeg的AVfilter的例子-纯净版

2016年02月01日 13:55:04　阅读数：53101

===

最简单的基于FFmpeg的AVfilter例子系列文章：

最简单的基于FFmpeg的AVfilter例子（水印叠加）

最简单的基于FFmpeg的AVfilter的例子-纯净版

===

有关FFmpeg的avfilter已经写过一个水印叠加的例子《 最简单的基于FFmpeg的AVfilter例子（水印叠加） 》，本文作为补充再记录一个纯净版的avfilter的例子。此前libavfilter一直是结合着libavcodec等类库的接口函数使用的，因此我一直以为libavfilter库与libavcodec等类库是高度耦合的（也就是如果想使用libavfilter的视音频特效功能的话必须使用libavcodec等类库的函数）。这两天空闲的时候研究了一下libavfilter的代码后发现实际情况不是这样的：libavfilter可以独立于libavcodec等类库的接口函数作为一个“纯粹”的视音频特效类库进行使用。本文记录的“纯净版”的avfilter的例子即实现了一个纯粹的视频特效添加的功能。该例子输入为一个YUV文件，输出也是一个YUV文件，通过avfilter的功能可以处理该YUV文件实现去色调、模糊、水平翻转、裁剪、加方框、叠加文字等功能。

流程图

该程序的流程图如下所示。AVFilter的初始化比较复杂，而使用起来比较简单。初始化的时候需要调用avfilter_register_all()到avfilter_graph_config()一系列函数。而使用的时候只有两个函数：av_buffersrc_add_frame()用于向FilterGraph中加入一个AVFrame，而av_buffersink_get_frame()用于从FilterGraph中取出一个AVFrame。

流程中的关键函数如下所示：

avfilter_register_all()：注册所有AVFilter。
avfilter_graph_alloc()：为FilterGraph分配内存。
avfilter_graph_create_filter()：创建并向FilterGraph中添加一个Filter。
avfilter_graph_parse_ptr()：将一串通过字符串描述的Graph添加到FilterGraph中。
avfilter_graph_config()：检查FilterGraph的配置。
av_buffersrc_add_frame()：向FilterGraph中加入一个AVFrame。
av_buffersink_get_frame()：从FilterGraph中取出一个AVFrame。

代码

[cpp]

```cpp
/**
 * 最简单的基于FFmpeg的AVFilter例子 - 纯净版
 * Simplest FFmpeg AVfilter Example - Pure
 *
 * 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
 *
 * 本程序使用FFmpeg的AVfilter实现了YUV像素数据的滤镜处理功能。
 * 可以给YUV数据添加各种特效功能。
 * 是最简单的FFmpeg的AVFilter方面的教程。
 * 适合FFmpeg的初学者。
 *
 * This software uses FFmpeg's AVFilter to process YUV raw data.
 * It can add many excellent effect to YUV data.
 * It's the simplest example based on FFmpeg's AVFilter.
 * Suitable for beginner of FFmpeg
 *
 */
#include <stdio.h>

#define __STDC_CONSTANT_MACROS

#ifdef _WIN32
```

```
#ifdef _WIN32
#define snprintf _snprintf
//Windows
extern "C"
{
#include "libavfilter/avfiltergraph.h"
#include "libavfilter/buffersink.h"
#include "libavfilter/buffersrc.h"
#include "libavutil/avutil.h"
#include "libavutil/imgutils.h"
};
#else
//Linux...
#ifdef __cplusplus
extern "C"
{
#endif
#include <libavfilter/avfiltergraph.h>
#include <libavfilter/buffersink.h>
#include <libavfilter/buffersrc.h>
#include <libavutil/avutil.h>
#include <libavutil/imgutils.h>
#ifdef __cplusplus
};
#endif
#endif




int main(int argc, char* argv[])
{
    int ret;
    AVFrame *frame_in;
    AVFrame *frame_out;
    unsigned char *frame_buffer_in;
    unsigned char *frame_buffer_out;

    AVFilterContext *buffersink_ctx;
    AVFilterContext *buffersrc_ctx;
    AVFilterGraph *filter_graph;
    static int video_stream_index = -1;

    //Input YUV
    FILE *fp_in=fopen("sintel_480x272_yuv420p.yuv","rb+");
    if(fp_in==NULL){
        printf("Error open input file.\n");
        return -1;
    }
    int in_width=480;
    int in_height=272;

    //Output YUV
    FILE *fp_out=fopen("output.yuv","wb+");
    if(fp_out==NULL){
        printf("Error open output file.\n");
        return -1;
    }

    //const char *filter_descr = "lutyuv='u=128:v=128'";
    const char *filter_descr = "boxblur";
    //const char *filter_descr = "hflip";
    //const char *filter_descr = "hue='h=60:s=-3'";
    //const char *filter_descr = "crop=2/3*in_w:2/3*in_h";
    //const char *filter_descr = "drawbox=x=100:y=100:w=100:h=100:color=pink@0.5";
    //const char *filter_descr = "drawtext=fontfile=arial.ttf:fontcolor=green:fontsize=30:text='Lei Xiaohua'";

    avfilter_register_all();

    char args[512];
    AVFilter *buffersrc  = avfilter_get_by_name("buffer");
    AVFilter *buffersink = avfilter_get_by_name("ffbuffersink");
    AVFilterInOut *outputs = avfilter_inout_alloc();
    AVFilterInOut *inputs  = avfilter_inout_alloc();
    enum PixelFormat pix_fmts[] = { AV_PIX_FMT_YUV420P, PIX_FMT_NONE };
    AVBufferSinkParams *buffersink_params;

    filter_graph = avfilter_graph_alloc();

    /* buffer video source: the decoded frames from the decoder will be inserted here. */
    snprintf(args, sizeof(args),
        "video_size=%dx%d:pix_fmt=%d:time_base=%d/%d:pixel_aspect=%d/%d",
        in_width,in_height,AV_PIX_FMT_YUV420P,
        1, 25,1,1);

    ret = avfilter_graph_create_filter(&buffersrc_ctx, buffersrc, "in",
        args, NULL, filter_graph);
    if (ret < 0) {
        printf("Cannot create buffer source\n");
        return ret;
    }

```

```c
    /* buffer video sink: to terminate the filter chain. */
    buffersink_params = av_buffersink_params_alloc();
    buffersink_params->pixel_fmts = pix_fmts;
    ret = avfilter_graph_create_filter(&buffersink_ctx, buffersink, "out",
        NULL, buffersink_params, filter_graph);
    av_free(buffersink_params);
    if (ret < 0) {
        printf("Cannot create buffer sink\n");
        return ret;
    }

    /* Endpoints for the filter graph. */
    outputs->name       = av_strdup("in");
    outputs->filter_ctx = buffersrc_ctx;
    outputs->pad_idx    = 0;
    outputs->next       = NULL;

    inputs->name       = av_strdup("out");
    inputs->filter_ctx = buffersink_ctx;
    inputs->pad_idx    = 0;
    inputs->next       = NULL;

    if ((ret = avfilter_graph_parse_ptr(filter_graph, filter_descr,
        &inputs, &outputs, NULL)) < 0)
        return ret;

    if ((ret = avfilter_graph_config(filter_graph, NULL)) < 0)
        return ret;

    frame_in=av_frame_alloc();
    frame_buffer_in=(unsigned char *)av_malloc(av_image_get_buffer_size(AV_PIX_FMT_YUV420P, in_width,in_height,1));
    av_image_fill_arrays(frame_in->data, frame_in->linesize,frame_buffer_in,
        AV_PIX_FMT_YUV420P,in_width, in_height,1);

    frame_out=av_frame_alloc();
    frame_buffer_out=(unsigned char *)av_malloc(av_image_get_buffer_size(AV_PIX_FMT_YUV420P, in_width,in_height,1));
    av_image_fill_arrays(frame_out->data, frame_out->linesize,frame_buffer_out,
        AV_PIX_FMT_YUV420P,in_width, in_height,1);

    frame_in->width=in_width;
    frame_in->height=in_height;
    frame_in->format=AV_PIX_FMT_YUV420P;

    while (1) {

        if(fread(frame_buffer_in, 1, in_width*in_height*3/2, fp_in)!= in_width*in_height*3/2){
            break;
        }
        //input Y,U,V
        frame_in->data[0]=frame_buffer_in;
        frame_in->data[1]=frame_buffer_in+in_width*in_height;
        frame_in->data[2]=frame_buffer_in+in_width*in_height*5/4;

        if (av_buffersrc_add_frame(buffersrc_ctx, frame_in) < 0) {
            printf( "Error while add frame.\n");
            break;
        }

        /* pull filtered pictures from the filtergraph */
        ret = av_buffersink_get_frame(buffersink_ctx, frame_out);
        if (ret < 0)
            break;

        //output Y,U,V
        if(frame_out->format==AV_PIX_FMT_YUV420P){
            for(int i=0;i<frame_out->height;i++){
                fwrite(frame_out->data[0]+frame_out->linesize[0]*i,1,frame_out->width,fp_out);
            }
            for(int i=0;i<frame_out->height/2;i++){
                fwrite(frame_out->data[1]+frame_out->linesize[1]*i,1,frame_out->width/2,fp_out);
            }
            for(int i=0;i<frame_out->height/2;i++){
                fwrite(frame_out->data[2]+frame_out->linesize[2]*i,1,frame_out->width/2,fp_out);
            }
        }
        printf("Process 1 frame!\n");
        av_frame_unref(frame_out);
    }

    fclose(fp_in);
    fclose(fp_out);

    av_frame_free(&frame_in);
    av_frame_free(&frame_out);
    avfilter_graph_free(&filter_graph);

    return 0;
}
```

结果

本程序输入为一个名称为“sintel_480x272_yuv420p.yuv”的YUV420P视频数据，输出为一个名称为“output.yuv” 的YUV420P视频数据。输入的视频数据的内容如下所示。

程序中提供了几种特效：

- lutyuv='u=128:v=128'
- boxblur
- hflip
- hue='h=60:s=-3'
- crop=2/3*in_w:2/3*in_h
- drawbox=x=100:y=100:w=100:h=100:color=pink@0.5
- drawtext=fontfile=arial.ttf:fontcolor=green:fontsize=30:text='Lei Xiaohua'

可以通过修改程序中的filter_descr字符串实现上述几种特效。下面展示几种特效的效果图。

lutyuv='u=128:v=128'

boxblur

hflip

hue='h=60:s=-3'

crop=2/3*in_w:2/3*in_h

drawbox=x=100:y=100:w=100:h=100:color=pink@0.5

drawtext=fontfile=arial.ttf:fontcolor=green:fontsize=30:text='Lei Xiaohua'

下载

simplest ffmpeg video filter

项目主页
SourceForge：https://sourceforge.net/projects/simplestffmpegvideofilter/

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

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

CSDN下载地址：http://download.csdn.net/detail/leixiaohua1020/9424521

本程序使用包含下面两个项目：
simplest_ffmpeg_video_filter：可以将一张PNG图片作为水印叠加到视频上，结合使用了libavfilter，libavcodec等类库。
simplest_ffmpeg_video_filter_pure：可以给YUV像素数据加特效，只用了libavfilter库。

文章标签： ffmpeg 特效 yuv 滤镜 libavfilter
个人分类： FFMPEG 我的开源项目