ffplay播放器-15-16-暂停、逐帧、音量

《FFmpeg/WebRTC/RTMP音视频流媒体高级开发教程》 – Darren老师：QQ326873713
课程链接：https://ke.qq.com/course/468797?tuin=137bb271

常用的播放器操作：
•播放：程序启动即播放，处于暂停时通过p或空格键
•静音：m键
•音量+：0键
•音量–：9键
•暂停：p或空格键
•快退/快进：左箭头/右箭头
•逐帧：s键
•退出：q或Esc键
•全屏：f或者鼠标左键双击

event_loop做按键响应

播放、暂停

- **画面要停止**
  - **画面停留在最后一帧**
- **声音要停止**

    - 音频回调接口请求数据帧时直接填0
- 读取数据是否要停止?
    - 音视频包缓存队列满时进入休眠。
- 暂停->继续 ： 时钟的恢复
- 暂停：toggle_pause()

暂停/继续状态切换

函数调用关系如下：

```
main() -->
event_loop() -->
toggle_pause() -->
stream_toggle_pause()
```

stream_toggle_pause()实现状态翻转：

```
/* pause or resume the video */
static void stream_toggle_pause(VideoState *is)
{
    if (is->paused) {
        // 这里表示当前是暂停状态，将切换到继续播放状态。在继续播放之前，先将暂停期间流逝的时间加到frame_timer中
        is->frame_timer += av_gettime_relative() / 1000000.0 - is->vidclk.last_updated;
        if (is->read_pause_return != AVERROR(ENOSYS)) {
            is->vidclk.paused = 0;
        }
        set_clock(&is->vidclk, get_clock(&is->vidclk), is->vidclk.serial);
    }
    set_clock(&is->extclk, get_clock(&is->extclk), is->extclk.serial);
    is->paused = is->audclk.paused = is->vidclk.paused = is->extclk.paused = !is->paused;
}
```

暂停状态下的视频播放

在video_refresh()函数中有如下代码：

```
/* called to display each frame */
static void video_refresh(void *opaque, double *remaining_time)
{
    ......

    // 视频播放
    if (is->video_st) {
        ......
        // 暂停处理：不停播放上一帧图像
        if (is->paused)
            goto display;

        ......
    }

    ......
}
```

在暂停状态下，实际就是不停播放上一帧(最后一帧)图像。画面不更新。

暂停状态下的音频播放

sdl_audio_callback
    ->audio_decode_frame

```
static int audio_decode_frame(VideoState *is)
{
    int data_size, resampled_data_size;
    int64_t dec_channel_layout;
    av_unused double audio_clock0;
    int wanted_nb_samples;
    Frame *af;

    if (is->paused)
        return -1;      // 暂停返回-1，  但这里返回-1并不会导致程序结束。
    ....
 }
```

逐帧、调音量、静音

逐帧

- **逐帧播放的本质是，播放一帧图像，然后暂停。**

- 涉及到的函数和变量：
  - step_to_next_frame()
  - is->step = 1时单步播放一帧，然后paused

- 逐帧播放流程
  a. 按s键，如果当前处于暂停则启动播放；
  b. 播放一帧数据然后进入暂停状态

逐帧播放是用户每按一次s键，播放器播放一帧画现。
逐帧播放实现的方法是：每次按了s键，就将状态切换为播放，播放一帧画面后，将状态切换为暂停。
函数调用关系如下：

```
main() -->
event_loop() -->
step_to_next_frame() -->
stream_toggle_pause()
```

实现代码比较简单，如下：

```
static void step_to_next_frame(VideoState *is)
{
    /* if the stream is paused unpause it, then step */
    if (is->paused)
        stream_toggle_pause(is);        // 确保切换到播放状态，播放一帧画面
    is->step = 1;
}
```

```
/* called to display each frame */
static void video_refresh(void *opaque, double *remaining_time)
{
    ......

```

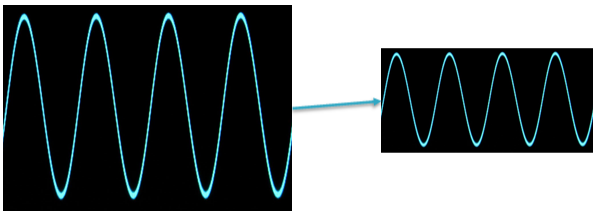
```
    // 视频播放
    if (is->video_st) {
        ......
        if (is->step && !is->paused)
            stream_toggle_pause(is);    // 逐帧播放模式下，播放一帧画面后暂停
        ......
    }

    ......
}
```

调音量

音量控制的本质：控制采样点的幅值

- 静音，将采样点数值置为0
- 音量+，提升采样点的幅值
- 音量-，降低采样点的幅值



降低音量

ffplay控制音量的方式

- 最大音量：输出解码后的原始数据
- 静音：即是输出数值为0的数据
  - toggle_mute()
- 改变音量：通过SDL_MixAudioFormat改变解码后数据的幅值
  - update_volume()
- •比如下图所示（sdl_audio_callback函数内）：

以下是ffplay的方式供参考

```
if ( !is->muted && is->audio_buf && is->audio_volume == SDL_MIX_MAXVOLUME )
{
    1. 最大音量则直接拷贝解码后的原始数据
    memcpy( stream, (uint8_t *) is->audio_buf + is->audio_buf_index, len1 );
}
else
{
     2. 先将数据清零，如果不再拷贝新数据给stream，则输出静音
    memset( stream, 0, len1 );

    3. 不为静音时则通过SDL_MixAudio来调整解码后数据的幅值以实现音量的变化
    /* 如果处于mute状态则直接使用stream填0数据，暂停时is->audio_buf = NULL */
    if ( !is->muted && is->audio_buf )                                  /* 处理音量，实际上是改变PCM数据的幅值 */
        SDL_MixAudio( stream, (uint8_t *) is->audio_buf + is->audio_buf_index, len1, is->audio_volume );
}
```

静音

```
static void toggle_mute(VideoState *is)
{
    is->muted = !is->muted;
}
```

```
memset(stream, 0, len1);
            // 3.调整音量
            /* 如果处于mute状态则直接使用stream填0数据, 暂停时is->audio_buf = NULL */
            if (!is->muted && is->audio_buf)
                SDL_MixAudioFormat(stream, (uint8_t *)is->audio_buf + is->audio_buf_index,
                                   AUDIO_S16SYS, len1, is->audio_volume);
```