

## 转 使用FFmpeg类库实现YUV视频序列编码为视频

2013年09月22日 01:36:13 阅读数：20415

搞视频处理的朋友肯定比较熟悉YUV视频序列，很多测试库提供的视频数据都是YUV视频序列，我们这里就选用YUV视频序列来做视频。关于YUV视频序列，我就不多讲了，可以看书学习，通常的视频序列都是YUV420格式的。

步骤也就那几步，添加视频流，打开编码器，开辟相应的内存空间，然后就可以打开YUV序列逐帧写入数据了，so easy!记得最后要做好文件的关闭和内存的释放，因为FFmpeg是c风格的（不知道新版本是否是c++风格的），这些工作都需要自己做好啊。过多的说明是没用的，直接上代码：

这里我补充一下，大多数的视频格式好像只支持YUV格式的视频帧AVFrame，我试图直接把RGB的视频序列直接编码到视频这条路好像走不通，都需要把RGB的视频帧再转成YUV视频帧才行，不知道高手有没有其他高见。

```
[cpp]
1.  #include <stdio.h>
2.  #include <string.h>
3.
4.  extern "C"
5.  {
6.  #include <libavcodec\avcodec.h>
7.  #include <libavformat\avformat.h>
8.  #include <libswscale\swscale.h>
9.  };
10.
11. void main( int argc, char ** argv)
12. {
13.     AVFormatContext* oc;
14.     AVOutputFormat* fmt;
15.     AVStream* video_st;
16.     double video_pts;
17.     uint8_t* video_outbuf;
18.     uint8_t* picture_buf;
19.     AVFrame* picture;
20.     // AVFrame* pictureRGB;
21.     int size;
22.     int ret;
23.     int video_outbuf_size;
24.
25.     FILE *fin = fopen( "akiyo_qcif.yuv" , "rb" ); //视频源文件
26.
27.     const char * filename = "test.mpg" ;
28.     // const char* filename;
29.     // filename = argv[1];
30.
31.     av_register_all();
32.
33.     // avcodec_init(); // 初始化codec库
34.     // avcodec_register_all(); // 注册编码器
35.
36.     fmt = guess_format(NULL, filename, NULL);
37.     oc = av_alloc_format_context();
38.     oc->oformat = fmt;
39.     snprintf(oc->filename, sizeof (oc->filename), "%s" , filename);
40.
41.     video_st = NULL;
42.     if (fmt->video_codec != CODEC_ID_NONE)
43.     {
44.         AVCodecContext* c;
45.         video_st = av_new_stream(oc, 0);
46.         c = video_st->codec;
47.         c->codec_id = fmt->video_codec;
48.         c->codec_type = CODEC_TYPE_VIDEO;
49.         c->bit_rate = 400000;
50.         c->width = 176;
51.         c->height = 144;
52.         c->time_base.num = 1;
53.         c->time_base.den = 25;
54.         c->gop_size = 12;
55.         c->pix_fmt = PIX_FMT_YUV420P;
56.         if (c->codec_id == CODEC_ID_MPEG2VIDEO)
57.         {
58.             c->max_b_frames = 2;
59.         }
60.         if (c->codec_id == CODEC_ID_MPEG1VIDEO)
61.         {
62.             c->mb_decision = 2;
63.         }
64.         if (!strcmp(oc->oformat->name, "mp4" ) || !strcmp(oc->oformat->name, "mov" ) || !strcmp(oc->oformat->name, "3gp" ))
65.         {
66.             c->flags |= CODEC_FLAG_GLOBAL_HEADER;
67.         }
68.     }
69.
70.     if (av_set_parameters(oc, NULL)<0)
71.     {
```

```

72.     return ;
73. }
74.
75. dump_format(oc, 0, filename, 1);
76. if (video_st)
77. {
78.     AVCodecContext* c;
79.     AVCodec* codec;
80.     c = video_st->codec;
81.     codec = avcodec_find_encoder(c->codec_id);
82.     if (!codec)
83.     {
84.         return ;
85.     }
86.     if (avcodec_open(c, codec) < 0)
87.     {
88.         return ;
89.     }
90.     if (!(oc->oformat->flags & AVFMT_RAWPICTURE))
91.     {
92.         video_outbuf_size = 200000;
93.         video_outbuf = (uint8_t*)av_malloc(video_outbuf_size);
94.     }
95.     picture = avcodec_alloc_frame();
96.     size = avpicture_get_size(c->pix_fmt, c->width, c->height);
97.     picture_buf = (uint8_t*)av_malloc(size);
98.     if (!picture_buf)
99.     {
100.         av_free(picture);
101.     }
102.     avpicture_fill((AVPicture*)picture, picture_buf, c->pix_fmt, c->width, c->height);
103. }
104.
105. if (!(fmt->flags & AVFMT_NOFILE))
106. {
107.     if (url_fopen(&oc->pb, filename, URL_WRONLY) < 0)
108.     {
109.         return ;
110.     }
111. }
112. av_write_header(oc);
113.
114. for ( int i=0; i<300; i++)
115. {
116.     if (video_st)
117.     {
118.         video_pts = ( double )(video_st->pts.val * video_st->time_base.num / video_st->time_base.den);
119.     }
120.     else
121.     {
122.         video_pts = 0.0;
123.     }
124.     if (!video_st /* || video_pts >= 5.0*/ )
125.     {
126.         break ;
127.     }
128.     AVCodecContext* c;
129.     c = video_st->codec;
130.     size = c->width * c->height;
131.
132.     if (fread(picture_buf, 1, size*3/2, fin) < 0)
133.     {
134.         break ;
135.     }
136.
137.     picture->data[0] = picture_buf; // 亮度
138.     picture->data[1] = picture_buf+ size; // 色度
139.     picture->data[2] = picture_buf+ size*5/4; // 色度
140.
141.     // 如果是rgb序列, 可能需要如下代码
142.     //     SwsContext* img_convert_ctx;
143.     //     img_convert_ctx = sws_getContext(c->width, c->height, PIX_FMT_RGB24, c->width, c->height, c-
144.     // >pix_fmt, SWS_BICUBIC, NULL, NULL, NULL);
145.     //     sws_scale(img_convert_ctx, pictureRGB->data, pictureRGB->linesize, 0, c->height, picture->data, picture->linesize);
146.
147.     if (oc->oformat->flags & AVFMT_RAWPICTURE)
148.     {
149.         AVPacket pkt;
150.         av_init_packet(&pkt);
151.         pkt.flags |= PKT_FLAG_KEY;
152.         pkt.stream_index = video_st->index;
153.         pkt.data = (uint8_t*)picture;
154.         pkt.size = sizeof (AVPicture);
155.         ret = av_write_frame(oc, &pkt);
156.     }
157.     else
158.     {
159.         int out_size = avcodec_encode_video(c, video_outbuf, video_outbuf_size, picture);
160.         if (out_size > 0)
161.         {
162.             AVPacket pkt;

```

```
162.     av_init_packet(&pkt);
163.     pkt.pts = av_rescale_q(c->coded_frame->pts, c->time_base, video_st->time_base);
164.     if (c->coded_frame->key_frame)
165.     {
166.         pkt.flags |= PKT_FLAG_KEY;
167.     }
168.     pkt.stream_index = video_st->index;
169.     pkt.data = video_outbuf;
170.     pkt.size = out_size;
171.     ret = av_write_frame(oc, &pkt);
172. }
173. }
174. }
175.
176. if (video_st)
177. {
178.     avcodec_close(video_st->codec);
179.     //     av_free(picture->data[0]);
180.     av_free(picture);
181.     av_free(video_outbuf);
182.     //     av_free(picture_buf);
183. }
184. av_write_trailer(oc);
185. for ( int i=0; i<oc->nb_streams; i++)
186. {
187.     av_freep(&oc->streams[i]->codec);
188.     av_freep(&oc->streams[i]);
189. }
190. if (!(fmt->flags & AVFMT_NOFILE))
191. {
192.     url_fclose(oc->pb);
193. }
194. av_free(oc);
195. }
```

原文地址：[http://blog.csdn.net/yang\\_xian521/article/details/7698742](http://blog.csdn.net/yang_xian521/article/details/7698742)

文章标签：

ffmpeg

编码

视频

yuv

类库

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