【H.264/AVC视频编解码技术详解】 九、序列参数集 Sequence Paramater Set(SPS)解析

blog.csdn.net/shaqoneal/article/details/52771030

《H.264/AVC视频编解码技术详解》视频教程已经在“CSDN学院”上线，视频中详述了H.264的背景、标准协议和实现，并通过一个实战工程的形式对H.264的标准进行解析和实现，欢迎观看！

“纸上得来终觉浅，绝知此事要躬行”，只有自己按照标准文档以代码的形式操作一遍，才能对视频压缩编码标准的思想和方法有足够深刻的理解和体会！

链接地址：H.264/AVC视频编解码技术详解

GitHub代码地址：点击这里

在H.264标准协议中规定了多种不同的NAL Unit类型，其中类型7表示该NAL Unit内保存的数据为Sequence Paramater Set。在H.264的各种语法元素中，SPS中的信息至关重要。如果其中的数据丢失或出现错误，那么解码过程很可能会失败。SPS及后续将要讲述的图像参数集PPS在某些平台的视频处理框架（比如iOS的VideoToolBox等）还通常作为解码器实例的初始化信息使用。

1. 什么是SPS

如上文所述，SPS即Sequence Paramater Set，又称作序列参数集。SPS中保存了一组编码视频序列(Coded video sequence)的全局参数。所谓的编码视频序列即原始视频的一帧一帧的像素数据经过编码之后的结构组成的序列。而每一帧的编码后数据所依赖的参数保存于图像参数集中。

在简单的H.264编解码的Demo（例如本系列博文、课程所做的SimpleH264Analyzer工程）中，一个SPS和PPS的NAL Unit通常位于整个码流的起始位置。但在某些特殊情况下，在码流中间也可能出现这两种结构，主要原因可能为：

1. 解码器需要在码流中间开始解码；
2. 编码器在编码的过程中改变了码流的参数（如图像分辨率等）；

2. SPS的结构

在我们这个实例中，SPS NAL Unit中的二进制内容为：

```
42 00 1e e8 58 58 98 80
       1
```

为了让后续的解码过程可以使用SPS中包含的参数，必须对其中的数据进行解析。其中H.264标准协议中规定的SPS格式位于文档的7.3.2.1.1部分，如下图所示：

seq_parameter_set_data( ) {	C	Descriptor
profile_idc	0	u(8)
constraint_set0_flag	0	u(1)
constraint_set1_flag	0	u(1)
constraint_set2_flag	0	u(1)
constraint_set3_flag	0	u(1)
constraint_set4_flag	0	u(1)
constraint_set5_flag	0	u(1)
reserved_zero_2bits /* equal to 0 */	0	u(2)
level_idc	0	u(8)
seq_parameter_set_id	0	ue(v)
if(profile_idc == 100 || profile_idc == 110 || profile_idc == 122 || profile_idc == 244 || profile_idc == 44 || profile_idc == 83 || profile_idc == 86 || profile_idc == 118 || profile_idc == 128) {		
chroma_format_idc	0	ue(v)
if(chroma_format_idc == 3)		
separate_colour_plane_flag	0	u(1)
bit_depth_luma_minus8	0	ue(v)
bit_depth_chroma_minus8	0	ue(v)
qpprime_y_zero_transform_bypass_flag	0	u(1)
seq_scaling_matrix_present_flag	0	u(1)
if(seq_scaling_matrix_present_flag)		
for(i = 0; i < ((chroma_format_idc != 3) ? 8 : 12); i++) {		
seq_scaling_list_present_flag[i]	0	u(1)
if(seq_scaling_list_present_flag[i])		
if(i < 6)		
scaling_list(ScalingList4x4[i], 16, UseDefaultScalingMatrix4x4Flag[i])	0	
else		
scaling_list(ScalingList8x8[i − 6], 64, UseDefaultScalingMatrix8x8Flag[i − 6])	0	
}		
}		

log2_max_frame_num_minus4	0	ue(v)
pic_order_cnt_type	0	ue(v)
if(pic_order_cnt_type == 0)		
log2_max_pic_order_cnt_lsb_minus4	0	ue(v)
else if(pic_order_cnt_type == 1) {		
delta_pic_order_always_zero_flag	0	u(1)
offset_for_non_ref_pic	0	se(v)
offset_for_top_to_bottom_field	0	se(v)
num_ref_frames_in_pic_order_cnt_cycle	0	ue(v)
for(i = 0; i < num_ref_frames_in_pic_order_cnt_cycle; i++)		
offset_for_ref_frame[i]	0	se(v)
}		
max_num_ref_frames	0	ue(v)
gaps_in_frame_num_value_allowed_flag	0	u(1)

其中的每一个语法元素及其含义如下：

(1). profile_idc：

标识当前H.264码流的profile。我们知道，H.264中定义了三种常用的档次profile：

- 基准档次：baseline profile;
- 主要档次：main profile;
- 扩展档次：extended profile;

在H.264的SPS中，第一个字节表示profile_idc，根据profile_idc的值可以确定码流符合哪一种档次。判断规律为：

- profile_idc = 66 → baseline profile;
- profile_idc = 77 → main profile;
- profile_idc = 88 → extended profile;

在新版的标准中，还包括了High、High 10、High 4:2:2、High 4:4:4、High 10 Intra、High 4:2:2 Intra、High 4:4:4 Intra、CAVLC 4:4:4 Intra等，每一种都由不同的profile_idc表示。

另外，constraint_set0_flag ~ constraint_set5_flag是在编码的档次方面对码流增加的其他一些额外限制性条件。

在我们实验码流中，profile_idc = 0x42 = 66，因此码流的档次为baseline profile。

(2). level_idc

标识当前码流的Level。编码的Level定义了某种条件下的最大视频分辨率、最大视频帧率等参数，码流所遵从的level由level_idc指定。

当前码流中，level_idc = 0x1e = 30，因此码流的级别为3。

(3). seq_parameter_set_id

表示当前的序列参数集的id。通过该id值，图像参数集pps可以引用其代表的sps中的参数。

(4). log2_max_frame_num_minus4

用于计算MaxFrameNum的值。计算公式为MaxFrameNum = 2^(log2_max_frame_num_minus4 + 4)。MaxFrameNum是frame_num的上限值，frame_num是图像序号的一种表示方法，在帧间编码中常用作一种参考帧标记的手段。

(5). pic_order_cnt_type

表示解码picture order count(POC)的方法。POC是另一种计量图像序号的方式，与frame_num有着不同的计算方法。该语法元素的取值为0、1或2。

(6). log2_max_pic_order_cnt_lsb_minus4

用于计算MaxPicOrderCntLsb的值，该值表示POC的上限。计算方法为MaxPicOrderCntLsb = 2^(log2_max_pic_order_cnt_lsb_minus4 + 4)。

(7). max_num_ref_frames

用于表示参考帧的最大数目。

(8). gaps_in_frame_num_value_allowed_flag

标识位，说明frame_num中是否允许不连续的值。

(9). pic_width_in_mbs_minus1

用于计算图像的宽度。单位为宏块个数，因此图像的实际宽度为:

frame_width = 16 × (pic_width_in_mbs_minus1 + 1);
1

(10). pic_height_in_map_units_minus1

使用PicHeightInMapUnits来度量视频中一帧图像的高度。PicHeightInMapUnits并非图像明确的以像素或宏块为单位的高度，而需要考虑该宏块是帧编码或场编码。
PicHeightInMapUnits的计算方式为：

PicHeightInMapUnits = pic_height_in_map_units_minus1 + 1;
1

(11). frame_mbs_only_flag

标识位，说明宏块的编码方式。当该标识位为0时，宏块可能为帧编码或场编码；该标识位为1时，所有宏块都采用帧编码。根据该标识位取值不同，PicHeightInMapUnits的含义也不同，为0时表示一场数据按宏块计算的高度，为1时表示一帧数据按宏块计算的高度。

按照宏块计算的图像实际高度FrameHeightInMbs的计算方法为：

$$FrameHeightInMbs = (2 - frame\_mbs\_only\_flag) * PicHeightInMapUnits$$

1

(12). mb_adaptive_frame_field_flag

标识位，说明是否采用了宏块级的帧场自适应编码。当该标识位为0时，不存在帧编码和场编码之间的切换；当标识位为1时，宏块可能在帧编码和场编码模式之间进行选择。

(13). direct_8x8_inference_flag

标识位，用于B_Skip、B_Direct模式运动矢量的推导计算。

(14). frame_cropping_flag

标识位，说明是否需要对输出的图像帧进行裁剪。

(15). vui_parameters_present_flag

标识位，说明SPS中是否存在VUI信息。