# 原 H.264官方软件JM源代码简单分析-解码器ldecod

2015年11月17日 20:07:41　阅读数：28671

==========

H.264/H.265 官方源代码分析文章：

H.264官方软件JM源代码简单分析-编码器lencod

H.264官方软件JM源代码简单分析-解码器ldecod

HEVC官方软件HM源代码简单分析-编码器TAppEncoder

HEVC官方软件HM源代码简单分析-解码器TAppDecoder

==========

前一阵子看了一下H.264官方参考软件JM的源代码，在这里总结一下它的结构。JM编解码H.264的速度相对于FFmpeg来说是非常慢的，但是它的代码写得清晰易懂，更适合做学术方面的研究。JM包含了视频解码器ldecod和视频编码器lencod。本文记录视频解码器ldecod的结构。

## 函数调用关系图

JM中的H.264视频解码器ldecod的函数调用关系图如下所示。

单击查看更清晰的大图

下面解释一下图中关键标记的含义。

**函数背景色**

函数在图中以方框的形式表现出来。不同的背景色标志了该函数不同的作用：
白色背景的函数：普通内部函数。
粉红色背景函数：解析函数（Parser）。这些函数用于解析SPS、PPS等信息。
紫色背景的函数：熵解码函数（Entropy Decoding）。这些函数读取码流数据并且进行CABAC或者CAVLC熵解码。
绿色背景的函数：解码函数（Decode）。这些函数通过帧内预测、帧间预测、DCT反变换等方法解码压缩数据。
黄色背景的函数：环路滤波函数（Loop Filter）。这些函数对解码后的数据进行滤波，去除方块效应。

**箭头线**

箭头线标志了函数的调用关系：
黑色箭头线：不加区别的调用关系。
粉红色的箭头线：解析函数（Parser）之间的调用关系。
紫色箭头线：熵解码函数（Entropy Decoding）之间的调用关系。
绿色箭头线：解码函数（Decode）之间的调用关系。
黄色箭头线：环路滤波函数（Loop Filter）之间的调用关系。

**函数所在的文件**

每个函数标识了它所在的文件路径。

下文记录结构图中的几个关键部分。

## 普通内部函数

普通内部函数指的是ldecod中还没有进行分类的函数。例如：
解码器的main()函数中调用的参数配置函数Configure()、打开解码器函数OpenDecoder()、解码函数DecodeOneFrame()、输出函数WriteOneFrame()、关闭解码器函数CloseDecoder()等。
解码器主要解码函数DecodeOneFrame()逐层调用的函数decode_one_frame()、decode_slice()、decode_one_slice()等。

## 解析函数（Parser）

解析函数（Parser）用于解析H.264码流中的一些信息（例如SPS、PPS、Slice Header等）。在read_new_slice()中都调用这些解析函数完成了解析。下面举几个解析函数的例子。

read_next_nalu()：解析NALU。这个函数是后几个解析函数的前提。
ProcessPPS()：解析Slice Header。其中调用了InterpretPPS()用于解析PPS。
ProcessSPS()：解析SEI。其中调用了InterpretSPS()用于解析SPS。

其中read_next_nalu()调用了下面几个函数：

get_annex_b_NALU()：读取annexb格式的NALU。
GetRTPNALU()：读取RTP格式的NALU。
NALUtoRBSP()：将NALU处理为RBSP格式。
（PS：annexb和RTP是H.264码流的2种格式）

## 熵解码函数（Entropy Decoding）

熵解码函数（Entropy Decoding）读取码流数据并且进行CABAC或者CAVLC熵解码。熵解码工作是在Slice结构体的read_one_macroblock()函数中完成的。该函数根据宏块类型的不同，会调用不同的熵解码函数。CAVLC对应的解码函数是：

read_one_macroblock_p_slice_cavlc()：解码P宏块。
read_one_macroblock_b_slice_cavlc()：解码B宏块。
read_one_macroblock_i_slice_cavlc()：解码I宏块。

CABAC对应的解码函数是：

read_one_macroblock_p_slice_cabac()：解码P宏块。
read_one_macroblock_b_slice_cabac()：解码B宏块。
read_one_macroblock_i_slice_cabac()：解码I宏块。

## 解码函数（Decode）

解码函数（Decode）通过帧内预测、帧间预测等方法解码宏块压缩数据。解码工作都是在decode_one_macroblock()中完成的。decode_one_macroblock()调用了Slice结构体中的decode_one_component()解码一个亮度分量。decode_one_component()根据宏块所在的Slice的不同，会调用不同的处理函数：

decode_one_component_i_slice()：解码I Slice中的宏块。
decode_one_component_p_slice()：解码P Slice中的宏块。
decode_one_component_b_slice()：解码B Slice中的宏块。

I Slice中只包含Intra类型的宏块。因此decode_one_component_i_slice()调用下面函数对Intra类型宏块进行解码：

mb_pred_intra16x16()：帧内预测16x16宏块
mb_pred_intra4x4()：帧内预测4x4宏块
mb_pred_intra8x8()：帧内预测8x8宏块

P Slice中包含Intra类型和Inter类型（单向）的宏块。因此decode_one_component_p_slice()调用下面函数对Intra类型宏块进行解码：

mb_pred_intra16x16()：同上
mb_pred_intra4x4()：同上
mb_pred_intra8x8()：同上
mb_pred_p_inter16x16()：单向帧间预测16x16宏块
mb_pred_p_inter16x8()：单向帧间预测16x8宏块
mb_pred_p_inter8x16()：单向帧间预测8x16宏块
mb_pred_p_inter8x8()：单向帧间预测8x8宏块

B Slice中包含Intra类型和Inter类型（双向）的宏块。因此decode_one_component_b_slice()调用的函数和P Slice中的宏块是类似的，唯一的不同在于它增加了对B宏块的处理。

对于逐行扫描的Intra16x16宏块，具体的预测函数是intra_pred_16x16_normal()，其中根据帧内预测类型的不同，调用不同的函数进行处理：

intra16x16_vert_pred()：垂直模式
intra16x16_hor_pred()：水平模式
intra16x16_dc_pred()：DC模式
intra16x16_plane_pred()：Plane模式

对于逐行扫描的Inter16x16的P类型宏块，具体的预测函数是mb_pred_p_inter16x16()，其中会调用perform_mc()完成运动补偿。对于单向运动补偿，perform_mc()会调用perform_mc_single()；对于双向运动补偿，perform_mc()会调用perform_mc_bi()。运动补偿的过程中，会调用get_block_luma()完成四分之一像素的插值工作。

无论Inter类型宏块还是Intra类型的宏块，最后都会调用iMBtrans4x4()完成DCT反变换和残差叠加的工作。其中DCT反变换在inverse4x4()中完成，而残差叠加在sample_reconstruct()中完成。

## 环路滤波函数（Loop Filter）

环路滤波函数（Loop Filter）对解码后的数据进行滤波，去除方块效应。去块效应滤波是在DeblockPicture()中完成的。DeblockPicture()调用了DeblockMb()。而DeblockMb()中调用GetStrengthVer()、GetStrengthHor()函数获取滤波强度；调用EdgeLoopLumaVer()、EdgeLoopLumaHor()进行滤波。

**雷霄骅**
**leixiaohua1020@126.com**
**http://blog.csdn.net/leixiaohua1020**



文章标签： JM H.264 视频编码 解码器 源代码

个人分类： JM