



中华人民共和国国家标准

GB/T 25724—2017
代替 GB/T 25724—2010

公共安全视频监控数字视音频 编解码技术要求

Technical specifications for surveillance video and audio coding

2017-03-09 发布

2017-06-01 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会 发布

目 次

前言	III
引言	IV
1 范围	1
2 规范性引用文件	1
3 术语、定义和缩略语	1
3.1 术语和定义	1
3.2 缩略语	9
4 约定	10
4.1 算术运算符	10
4.2 逻辑运算符	11
4.3 关系运算符	11
4.4 位运算符	11
4.5 赋值运算符	12
4.6 数学函数	12
4.7 语法元素、变量和表	12
4.8 逻辑运算符的文字描述	13
4.9 过程	14
5 视频部分	14
5.1 编码比特流和输出数据的格式	14
5.2 语法和语义	19
5.3 解码过程	60
5.4 解析过程	114
6 音频部分	195
6.1 总体描述	195
6.2 编码器功能描述	198
6.3 解码器功能描述	244
6.4 比特分配描述	251
6.5 存储、传输接口格式	253
附录 A (规范性附录) 假设参考解码器(HRD)	259
附录 B (规范性附录) 字节流的格式	262
附录 C (规范性附录) 视频档次与级别	264
附录 D (规范性附录) 视频可用性信息(VUI)	267
附录 E (规范性附录) 补充增强信息(SEI)	270

附录 F (规范性附录) 智能分析数据描述	274
附录 G (规范性附录) 音频档次和级别	290
附录 H (规范性附录) 异常声音事件类型定义	292
附录 I (资料性附录) VAD 检测	293
附录 J (资料性附录) 噪声消除	297
参考文献	306



前　　言

本标准按照 GB/T 1.1—2009 给出的规则起草。

本标准代替 GB/T 25724—2010《安全防范监控数字视音频编解码技术要求》，与 GB/T 25724—2010 相比主要技术变化如下：

- 增加了术语(见 3.1.93~3.1.95)；
- 修改了编码单位结构(见 5.1.3,2010 年版的 5.1.3)；
- 修改了码流的语法和语义(见 5.2.3、5.2.4,2010 年版的 5.2.3、5.2.4)；
- 修改了安全参数集的语法和语义(见 5.2.3.2.5、5.2.4.4.4,2010 年版的 5.2.3.2.3、5.2.4.4.3)；
- 修改了参考图像的选择方法(见 5.3.3.4,2010 年版的 5.3.3.4)；
- 修改了帧内预测过程的内容(见 5.3.4,2010 年版的 5.3.4)；
- 修改了帧间预测过程的内容(见 5.3.5,2010 年版的 5.3.5)；
- 修改了变换量化与重建的内容(见 5.3.6,2010 年版的 5.3.6)；
- 修改了去块效应滤波过程的内容(见 5.3.7,2010 年版的 5.3.7)；
- 增加了样点自适应补偿(SAO)(见 5.3.8)；
- 增加了样点滤波补偿(见 5.3.9)；
- 修改了解析过程的内容(见 5.4,2010 年版的 5.4)；
- 修改了附录 F,删除了变长码表,增加了智能分析数据描述(见附录 F,2010 年版的附录 F)。

本标准由中华人民共和国公安部提出。

本标准由全国安全防范报警系统标准化技术委员会(SAC/TC 100)归口。

本标准起草单位：公安部第一研究所、北京中星微电子有限公司、北京中盾安全技术开发公司、中星电子股份有限公司、杭州恒生数字设备科技有限公司、公安部安全与警用电子产品质量检测中心、山西中天信科技股份有限公司、千目聚云数码科技(上海)有限公司、北京欣博电子科技有限公司、杭州海康威视数字技术股份有限公司、湖南国科微电子股份有限公司、浙江大华技术股份有限公司、苏州科达科技股份有限公司、浙江宇视科技有限公司、天津天地伟业数码科技有限公司、北京联视神盾安防技术有限公司、北京智芯原动科技有限公司、上海熙菱信息技术有限公司。

本标准主要起草人：陈朝武、邓中翰、郅晨、邱嵩、余子龙、张韵东、董骞、昝劲文、欧阳甸、卢京辉、闫雪、林冬、施巨岭、查敏中、汪人瑞、梁敏学、黄麒麟、廖双龙、周文博、马莉、夏昌盛、曾娟鹃、李伟丽、卢玉华、胡建华、王磊、孙大瑞、俞海、段争志、刘文尧、吕卓逸、姜黎、卢虹、倪昕、马伟、王秦镜、章勇、邢培银、王大治、吴参毅。

本标准所代替标准的历次版本发布情况为：



—— GB/T 25724—2010。

引　　言

在 GB/T 25724—2010《安全防范监控数字视音频编解码技术要求》(以下简称 SVAC 标准)发布之前,国内、国际没有专门针对安全防范监控应用的视音频编解码标准,所有的视音频编解码标准,都是针对广播电视和大众娱乐方面的应用,在安全防范领域直接采用具有很大的不适应性。

SVAC 标准(2010 年版)于 2010 年 12 月 23 日发布,2011 年 5 月 1 日实施。该标准是具有我国自主知识产权的、专门应用于安全防范视频监控技术领域的数字视音频编解码技术标准。该标准发布实施以后,国家标准委、公安部、工信部等部门高度重视标准的推广应用,支持成立了北京安防视音频编解码技术产业联盟(以下简称 SVAC 联盟),业内科研院所和广大企业围绕着 SVAC 产业链积极开展技术研发和产品应用。

在标准实施过程中发现,SVAC 标准在数据安全保护、提升压缩性能和编码效率、对智能化和大数据的支持等方面还有待补充和完善之处。为此,全国安全防范报警系统标准化技术委员会(代号 SAC/TC 100)组织公安部第一研究所和北京中星微电子有限公司等单位对 SVAC 标准进行了修订,使标准更具有先进性和可操作性。

近年来,视频监控系统建设应用已经从安全防范行业扩展到公共安全各行业、领域,已经成为新形势下维护国家安全和社会稳定的重要手段,在打击犯罪、治安防范、社会管理、服务民生等方面发挥着积极作用。本次修订充分考虑了公共安全视频监控联网与应用建设的需要,标准内容普遍适用于公共安全各行业、领域,因此标准名称变更为《公共安全视频监控数字视音频编解码技术要求》。

本标准主要技术特点有:

- a) 支持高精度视频数据编码,适应宽动态范围,保留更多的图像细节,满足忠实于场景的要求。
视频支持 8 比特~12 比特数据;
- b) 支持多样化的帧内及帧间预测、变换量化、二进制算术编码等技术,获得更好的图像质量和更高的编码效率;
- c) 支持感兴趣区域(ROI)变质量编码,在传输网络带宽或数据存储空间有限的情况下,优先保证 ROI 图像质量,节省非 ROI 的开销,提供更符合监控需要的高质量视频编码,提高监控系统整体性能;
- d) 支持可伸缩性视频编码(SVC),对视频数据分层次编码,满足不同传输网络带宽和数据存储环境的需求;
- e) 支持代数码书激励线性预测(ACELP)和变换音频编码(TAC)切换的双核音频编码,既保证对语音信号具有较好的编码效果,也保证环境(背景)声音的编码效果;
- f) 支持声音识别特征参数的编码,避免编码失真对语音识别和声纹识别的影响;
- g) 支持绝对时间参考信息、智能分析信息等监控专用信息。监控专用信息通过专门语法与视音频压缩编码数据一起传输和存储,规定了常用智能分析信息的携带方式,便于快速检索、分类查询、视音频同步和监控数据的综合应用;
- h) 支持数据安全保护,加强了对国密算法的支持,完善了安全参数集,增添了摘要、签名算法的标识等内容,并对密钥及数字证书相关信息的携带做了规范定义,支持视频数据加密、认证功能。

相关专利情况说明

本文件的发布机构提请注意,声明符合本文件时,可能涉及到与 5.2.3.1、5.2.3.2、5.2.4.2、5.2.4.4、5.2.4.7、6.1.2、6.1.4、6.2.6.1.3、6.2.6.1.4.10、6.5.2.2 中有关内容相关的专利的使用。

本文件的发布机构对于该专利的真实性、有效性和范围无任何立场。

该专利持有人已向本文件的发布机构保证,他愿意同任何申请人在合理且无歧视的条款和条件下,就专利授权许可进行谈判。该专利持有人的声明已在本文件的发布机构备案。相关信息可以通过以下联系方式获得:

专利持有人名称	联系地址
北京中星微电子有限公司	北京海淀区学院路 35 号世宁大厦(100191)
北京中盾安全技术开发公司	北京海淀区首体南路 1 号(100048)
中星电子股份有限公司	天津经济技术开发区第四大街 80 号天大科技园 A1 座 2 层(300457)
武汉大学	湖北武汉市武汉大学(430079)

联系人:曾娟鹃

通讯地址:北京海淀区学院路 35 号世宁大厦 16 层

邮政编码:100191

电子邮件:zengjuanjuan@vimicro.com

电话:010-68948888-8950

传真:010-68944075

联系人:李伟丽

通讯地址:北京海淀区首体南路 1 号

邮政编码:100048

电子邮件:lwl@zhongdun.com.cn

电话:010-68773553-6387

传真:010-68773553-6215

请注意除上述专利外,本文件的某些内容仍可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

公共安全视频监控数字视音频 编解码技术要求

1 范围

本标准规定了公共安全视频监控应用的数字视音频压缩编码的解码过程。

本标准适用于公共安全领域的视音频实时压缩、传输、播放和存储等业务,其他需要视音频编解码的领域也可参考采用。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件,仅注日期的版本适用于本文件。凡是不注日期的引用文件,其最新版本(包括所有的修改单)适用于本文件。

rfc 3548 The Base16, Base32, and Base64 Data Encodings



3 术语、定义和缩略语

3.1 术语和定义

下列术语和定义适用于本文件。

3.1.1

NAL 单元 NAL unit

一个语法结构,包含后续数据的类型指示和所包含的字节数,数据以 RBSP 形式出现,必要时其中还包括散布的防伪字节。

3.1.2

NAL 单元流 NAL unit stream

由 NAL 单元组成的序列。

3.1.3

保留 reserved

某些语法元素的特定取值。

注:供中国安全防范监控数字视音频编解码技术标准工作组将来使用。符合本标准的比特流不应使用这些值,但是这些值将来可能在本标准的扩展版本中用到。

3.1.4

闭环基音搜索 closed-loop pitch search

用于从加权输入信号和长时预测滤波器状态估计基音延迟。也称为自适应码书搜索。

3.1.5

比特流 bitstream

由编码视音频及其相关数据构成的比特序列。比特流既可用来表示 NAL 单元流,也可表示字节流。

3.1.6

变换系数 transform coefficient

频率域的标量,与解码过程的反变换部分中一个特定的一维或二维频率索引相关联的系数。

3.1.7

变换系数幅值 transform coefficient level

一个与特定二维频率索引相关联的整数量值,解码过程中用于计算变换系数的值。

3.1.8

编码过程 encoding process

产生符合本标准的比特流的过程,本标准对视频编码过程不做规定。

3.1.9

编码器 encoder

实现编码过程的实体,包括软件及硬件。

3.1.10

编码片 tile

一个矩形区域内部按照光栅扫描顺序排列的整数个树形编码单元。

3.1.11

编码视频序列 coded video sequence

按照解码顺序排列的 IDR 图像和紧随其后的零个或多个非 IDR 图像组成的图像序列。

3.1.12

编码图像 coded picture

一幅图像的编码表示。

注: 符合本标准的一个编码图像为一个编码帧。

3.1.13

编码图像缓存区 coded picture buffer

一个先入先出缓存区,其存储方式按解码顺序排列。

3.1.14

编码帧 coded frame

一个帧的编码表示。

3.1.15

残差 residual

样点或数据元素预测值与解码值之间的差值。

3.1.16

参考索引 reference index

参考图像的索引。

3.1.17

参考图像 reference picture

对解码顺序上后续图像的解码过程进行帧间预测的样点图像。

3.1.18

参考帧 reference frame

一个标记为参考图像的帧,用于解码过程中的帧间预测。

3.1.19

参数 parameter

序列参数集、图像参数集或安全参数集中的一语法元素。参数也用于量化参数一词中。

3.1.20

层 layer

没有分支等级关系中的一组句法结构。高层包含低层。编码层指编码图像序列层、图像层、编码片层和编码单元层。对于可伸缩性视频编码图像，不同层的图像具有不同的可伸缩性(如不同的空间分辨率)。

3.1.21

代数码书 algebraic codebook

脉冲幅度和位置组成的一个集合。通过码字索引 k 按照一定的规则得到第 k 个激励码矢量的脉冲幅度和位置。

3.1.22

档次 profile

本标准中的一个特定语法子集。

3.1.23

电导频谱对 admittance spectral pair

将逆滤波器传输函数 $A(z)$ 分解为一个偶对称和一个奇对称多项式函数，指示该函数在单位圆上的根。用于线性预测系数的变换。



3.1.24

二进制位 bin

二进制位串中的 1 比特。

3.1.25

二进制位串 bin string

一串二进制位。二进制位串为二值化的语法元素值的二进制表示。

3.1.26

二值化 binarization

语法元素所有可能值与一组二进制位串之间的唯一映射。

3.1.27

反变换 inverse transform

将变换系数矩阵转换为空域样点矩阵的过程。

3.1.28

防伪字节 emulation prevention byte

一个字节，其值等于 0x03，可能在 NAL 单元中出现。防伪字节的出现可以保证在 NAL 单元的后续字节对齐的字节流中不会含有起始码前缀。

3.1.29

非参考图像 non-reference picture

不用于对任何其他图像进行帧间编码的图像。

3.1.30

分量 component

图像的三个样点矩阵(一个亮度矩阵，两个色度矩阵)中的一个矩阵或矩阵中的单个样点。

在音频部分，也指矢量中的元素或信号中的某些频率成分。

3.1.31

感知加权滤波 perceptual weighting filter

利用共振峰处的噪声掩蔽特性，在共振峰区域分配比较大的失真，来减少峰谷主观感觉噪声的滤波。

3.1.32

功率谱 power spectrum

信号通过傅立叶变换后得到幅度谱的平方。

3.1.33

光栅扫描 raster scan

矩形二维图像到一维图像的映射过程,一维图像的第一组值来自于二维图像最上边一行的从左到右扫描,然后依次是第二行、第三行等等。对于图像每行(由上到下)都是从左到右扫描的。

3.1.34

后向预测 backward prediction

使用显示顺序上在后的解码图像中的样点对当前图像中的样点进行预测。

3.1.35

划分 partitioning

将一个集合分为子集的过程。集合中的每个元素属于且只属于某一个子集。

3.1.36

基本层图像 base layer picture

不需要参考其他图像层信息即可以解码的图像。

3.1.37

级别 level

本标准中的一个特定档次中的参数取值的限定集合。一个档次可以包含一个或多个级别。对所有档次定义了一组相同的级别,不同档次的每个级别大部分特性都是通用的。对于一个独立的实现,在一定的约束条件下,可以支持多个级别。

3.1.38

即时解码刷新(IDR)图像 instantaneous decoding refresh (IDR) picture

一幅帧内解码图像。IDR 图像解码之后,解码顺序上所有后续的编码图像都可以不用根据任何在 IDR 图像之前解码的图像来进行帧间预测解码。每个编码视频序列的第一幅图像为 IDR 图像。

3.1.39

假设参考解码器 hypothetical reference decoder

一个假设的解码器模型,规定了对于符合本标准的 NAL 单元流或字节流的可变性的约束。

3.1.40

解码过程 decoding process

读入编码的比特流后产生解码图像或者音频数据的过程。

3.1.41

解码器 decoder

实现解码过程的实体,包括软件及硬件。

3.1.42

解码顺序 decoding order

解码过程中处理语法元素的顺序。

3.1.43

解码图像 decoded picture

通过解码一幅编码图像得到的图像。符合本标准的一幅解码图像应是一个解码帧。

3.1.44

解码图像缓存区 decoded picture buffer

保存解码图像的存储空间,用于附录 A 中规定的预测参考、输出重排序或输出延时等。

3.1.45

开环基音搜索 open-loop pitch search

直接从加权输入信号中估计最优基音延迟的过程。开环基音搜索简化了基音分析，并且将闭环基音搜索限定在开环基音搜索的延迟值附近。

3.1.46

可伸缩性视频编码 scalable video coding

编码序列中的图像具有一定的可伸缩性。具有可伸缩性的图像通常包含基本层图像和增强层图像。

3.1.47

块 block

视频信号空间中的一个 $M \times N$ (M 列 N 行) 的样点矩阵，或者一个 $M \times N$ 的变换系数矩阵。

音频信号空间的一个一维矢量。

3.1.48

亮度 luma

一个样点矩阵或单个样点，用于描述信号的单色表示。亮度所用符号为 Y。

3.1.49

量化参数 quantization parameter

解码过程中对变换系数幅值进行反量化时使用的参数。

3.1.50

零输入响应 zero input response

滤波器当前输入为零时，由过去输入而产生的输出。

3.1.51

美尔 mel

一种非线性的频率刻度，根据主观音高进行划分。

3.1.52

美尔频率倒谱系数 mel-frequency cepstral coefficients

用 FFT 将时域信号转化到频域，对其对数能量谱依照 Mel 刻度分布的三角滤波器组进行卷积，对各个滤波器的输出构成的向量进行 DCT 得到的系数，即美尔频率倒谱系数。

3.1.53

内部采样频率 internal sampling frequency

音频编码器的采样频率，范围为 12 800 Hz~38 400 Hz，采用 F_s 表示。

3.1.54

逆滤波器 inverse filter

去除信号短时相关性的滤波器。

3.1.55

频率索引 frequency index

与解码过程中反变换之前的变换系数相关的一维或二维索引。

3.1.56

起始码前缀 start code prefix

字节流中唯一等于 0x000001 的 3 个字节的序列，作为每个 NAL 单元的前缀。解码器可以利用起始码前缀的位置来确定一个新的 NAL 单元的开始和前一个 NAL 单元的结束。NAL 单元中通过加入防伪字节来防止假冒的起始码前缀出现。

3.1.57

前向预测 forward prediction

使用显示顺序上在前的解码图像中的样点对当前图像中的样点进行预测。

3.1.58

色度 chroma

一个样点矩阵或单个样点,用于描述代表两个相对于基色的色差信号中的一个。色度所用符号为Cb和Cr。

3.1.59

二进制算术编码 binary arithmetic coding

一种熵编码方法,根据概率模型对二进制位进行编码,产生比特流。

3.1.60

声纹识别 voiceprint recognition

根据语音的声学特征识别该段语音所对应的说话人的过程。

3.1.61

数据比特串 string of data bits

语法元素的若干比特位的序列,出现在原始字节序列负载中原始字节序列负载截止位之前。在SODB中,最左边的比特位表示第一位即最高位,最右边的比特位表示最后一位即最低位。

3.1.62

双向预测 bidirectional prediction

使用显示顺序上在前及在后的解码图像中的样点对当前图像中的样点进行预测。

3.1.63

树形编码单元 coding tree unit

一个 $N \times N$ 的亮度样点块和相应的两个色度样点块。

3.1.64

图像 picture

源、编码或重构的图像数据的通称。符合本标准的一幅图像指一帧。

3.1.65

图像参数集 picture parameter set

一个语法结构,包含应用于一个或多个编码图像的语法元素。

3.1.66

维纳滤波器 wiener filter

根据最小均方误差准则,即滤波器的输出信号与期望信号之差的均方值最小,计算得到的最佳线性滤波器。

3.1.67

线性预测系数 LP coefficients

短时预测滤波器系数,也称为LPC系数。

3.1.68

序列参数集 sequence parameter set

一个语法结构,包含应用于一个或多个完整编码视频序列的语法元素。

3.1.69

音频超帧 audio superframe

由若干音频帧组成,目前本标准规定音频超帧中只包含一个音频帧。

3.1.70

音频子帧 audio subframe

音频帧的一部分,在 $F_s/2$ 采样频率下,由 64 个样点构成的数据块。

3.1.71

预测 prediction

使用预测值来提供当前解码的样点值或数据元素的估计。

3.1.72

预测值 predictor

以前解码的样点值或数据元素的线性组合。

3.1.73

语法结构 syntax structure

零个或多个语法元素按照规定顺序一起出现在比特流中。

3.1.74

语法元素 syntax element

比特流中表示数据的元素。

3.1.75

语音识别 speech recognition

根据语音的声学特征和语言模型,将该段语音翻译为文本的过程。

3.1.76

源 source

编码前视音频素材或者素材的某些属性。

3.1.77

原始字节序列负载 raw byte sequence payload

一个语法结构,包含整数个封装于 NAL 单元中的字节。RBSP 或者为空,或者包含具有数据比特串形式的语法元素,其后跟随 RBSP 截止位和零个或多个连续的 0 值比特。

3.1.78

原始字节序列负载(RBSP)截止位 raw byte sequence payload (RBSP) stop bit

值为 1 的一比特,出现在原始字节序列负载(RBSP)中的数据比特串之后。RBSP 中数据比特串的结束位置可以通过搜索 RBSP 中的 RBSP 截止位得到。

3.1.79

运动矢量 motion vector

二维矢量,用于帧间预测,表示匹配对象在解码图像和参考图像中的位置偏移。

3.1.80

增强层图像 enhance layer picture

需要参考其他图像层信息进行解码的图像。本标准中的一个增强层图像在解码时可以参考位于其下的图像层信息。

3.1.81

帧 frame

在视频信号空间中由一个亮度样点矩阵(Y)和两个可能存在的色度样点矩阵(Cb 和 Cr)构成。

在音频信号空间中,作为音频处理的基本数据块。在 F_s 采样频率下,512 个样本构成一帧,在 $F_s/2$ 采样频率下,256 个样本构成一帧。

3.1.82

帧间编码 inter coding

使用帧间预测对块或图像进行编码。

3.1.83

帧间预测 inter prediction

利用已解码的参考图像得到当前样点的预测值的过程。

3.1.84

帧间解码图像 inter decoded picture

使用帧内预测进行解码,或者根据先前解码的参考图像利用单前向预测、双前向预测或者双向预测进行解码的图像,对每个块进行帧间预测时最多使用两个运动矢量和参考索引。

3.1.85

帧内编码 intra coding

使用帧内预测对块或图像进行编码。

3.1.86

帧内解码图像 intra decoded picture

I 图像

只使用帧内预测解码的图像。

3.1.87

帧内预测 intra prediction

利用同一图像中已解码的样点得到当前样点的预测值的过程。

3.1.88

字节 byte

连续的 8 比特,读写时左边第一位为最高位,右边第一位为最低位。表示为比特序列时,字节的最高有效位为第一位。

3.1.89

字节对齐 byte-aligned

从比特流的第一个比特开始的 8 的倍数的位置为字节对齐的位置。比特或字节或语法元素为字节对齐的,指它出现在比特流中字节对齐的位置上。

3.1.90

字节流 byte stream

NAL 单元流的封装,包含起始码前缀和附录 B 定义的 NAL 单元。

3.1.91

自适应码书 adaptive codebook

通过长时预测滤波器状态得到的码书,由每个子帧自适应的激励矢量构成。

3.1.92

直流偏置 DC-offset

音频信号的直流分量。

3.1.93

安全前端设备 secure front end device

具备安全密码部件(如安全芯片,安全 TF 卡等)的视频监控前端设备。该设备能够利用证书存储和管理、密钥存储和管理、数据签名验签、数据加密等技术实现设备身份认证、视频签名、视频加密等功能。

3.1.94

视频加密密钥 video encryption key

安全前端设备随机产生的对称密钥,按照一定的规律变化,用于直接加密视频内容,实现视频传输的机密性保护。

3.1.95

视频密钥加密密钥 video key encryption key

安全前端设备维护的对称密钥,按照一定的规律变化,用于加密视频加密密钥,实现其传输的机密性保护。

3.2 缩略语

下列缩略语适用于本文件。

ACELP:代数码书激励线性预测(Algebraic Code Excited Linear Prediction)

ALF:样点滤波补偿(Adaptive Loopfilter)

BWE:带宽扩展(Bandwidth Extension)

CBR:恒定比特率(Constant Bit Rate)

CPB:编码图像缓存区(Coded Picture Buffer)

CRC:循环冗余校验码(Cyclic Redundancy Code)

CTU:树形编码单元(Coding Tree Unit)

DCT:离散余弦变换(Discrete Cosine Transform)

DFT:离散傅立叶变换(Discrete Fourier Transform)

DPB:解码图像缓存区(Decoded Picture Buffer)

FFT:快速傅立叶变换(Fast Fourier Transform)

FIR:有限冲击响应(Finite Impulse Response)

GOP:图像编码组(Group Of Pictures)

HRD:假设参考解码器(Hypothetical Reference Decoder)

IDCT:离散余弦逆变换(Inverse Discrete Cosine Transform)

IDFT:离散傅立叶逆变换(Inverse Discrete Fourier Transform)

IDR:即时解码刷新(Instantaneous Decoding Refresh)

IFFT:快速傅立叶逆变换(Inverse Fast Fourier Transform)

ISF:电导谱频率(Immittance Spectral Frequency)

ISP:电导谱对(Immittance Spectral Pair)

LD:低延时(Low Delay)

LP:线性预测(Linear Prediction)

LPC:线性预测编码(Linear Predictive Coding)

LSB:最低有效位(Least Significant Bit)

LTP:长时预测(Long Term Predictor)

MA:滑动平均(Moving Average)

MB:宏块(Macroblock)

MFCC:美尔频率倒谱系数(Mel-Frequency Cepstral Coefficients)

MSB:最高有效位(Most Significant Bit)

MSVQ:多级矢量量化(Multi-Stage Vector Quantization)

NAL:网络抽象层(Network Abstraction Layer)

OFB:输出反馈模式(Output Feedback)

PCM:脉冲编码调制(Pulse Code Modulation)
 RA:随机访问(Random Access)
 RBSP:原始字节序列负载(Raw Byte Sequence Payload)
 ROI:感兴趣区域(Region Of Interest)
 RPS:参考图像集(Reference Picture Set)
 SAO:样点偏移补偿(Sample Adaptive Offset)
 SEI:补充增强信息(Supplement Enhancement Information)
 SNR:信噪比(Signal Noise Ratio)
 SODB:数据比特串(String Of Data Bits)
 SVC:可伸缩性视频编码(Scalable Video Coding)
 TAC:变换域音频编码(Transform Audio Coding)
 TVC:变换域矢量编码(Transform Vector Coding)
 VAD:语音活动检测(Voice Activity Detection)
 VBR:可变比特率(Variable Bit Rate)
 VCL:视频编码层(Video Coding Layer)
 VQ:矢量量化(Vector Quantization)
 VUI:视频可用性信息(Video Usability Information)
 WPP:波前并行处理(Wavefront Parallel Processing)

4 约定

4.1 算术运算符

算术运算符定义见表 1。

表 1 算术运算符定义

编号	符号	说 明
1	+	加法运算
2	-	减法运算(二元运算符)或取反(一元前缀运算符)
3	×	乘法运算
4	\otimes	卷积运算
5	x^y	指数运算,表示 x 的 y 次幂。在不是表示指数的情况下也可表示上标
6	/	除法运算,不做截断或四舍五入
7	\div	除法运算,不做截断或四舍五入
8	$\frac{x}{y}$	除法运算,不做截断或四舍五入
9	$\sum_{i=x}^y f(i)$	自变量 i 取由 x 到 y (含 y)的所有整数值时,函数 $f(i)$ 的累加和
10	$x \% y$	模运算, x 除以 y 的余数,其中 x 与 y 都是正整数

在没有以插入括号来明确指定运算优先次序的情况下,遵守如下规则:

- 乘法和除法运算先于加法和减法运算;
- 乘法和除法运算从左到右进行;
- 加法和减法运算从左到右进行。

4.2 逻辑运算符

逻辑运算符定义见表 2。

表 2 逻辑运算符定义

编号	符号	说 明
1	&&	逻辑“与”运算
2		逻辑“或”运算
3	!	逻辑“非”运算
4	$x ? y : z$	如果 x 为真或非 0 值, 则取值为 y ; 否则取值为 z

4.3 关系运算符

关系运算符定义见表 3。



表 3 关系运算符定义

编号	符号	说 明
1	>	大于
2	\geq	大于或等于
3	<	小于
4	\leq	小于或等于
5	$=$	等于
6	\neq	不等于

4.4 位运算符

位运算符定义见表 4。

表 4 位运算符定义

编号	符号	说 明
1	&	按位“与”运算。对整数进行运算时, 以整数的二进制补码形式进行操作。如果两个二进制运算数中一个位数小于另外一个, 则较短的运算数高位加 0 补齐
2		按位“或”运算。对整数进行运算时, 以整数的二进制补码形式进行操作。如果两个二进制运算数中一个位数小于另外一个, 则较短的运算数高位加 0 补齐
3	\sim	按位“取反”运算。按位取反运算是单目运算, 用来求一个位串信息按位的反, 即为 0 的位, 结果是 1; 为 1 的位, 结果是 0
4	\wedge	按位“异或”运算。异或运算是求两个运算分量相应位值是否相异, 相异的为 1, 相同的为 0
5	$x \gg y$	将 x 以 2 的补码整数表示的形式向右移 y 位。仅当 y 取非负数时定义此运算。右移运算移入 MSB 的位应该等于移位运算前 x 的 MSB 的值
6	$x \ll y$	将 x 以 2 的补码整数表示的形式向左移 y 位。仅当 y 取非负数时定义此运算。左移运算移入 LSB 的位值为 0

4.5 赋值运算符

赋值运算定义见表 5。

表 5 赋值运算定义

编号	符号	说明
1	=	赋值运算符
2	++	递增,例如 $x++$ 相当于 $x = x + 1$;当用于数组下标时,在自加运算前先求变量值
3	--	递减,例如 $x--$ 相当于 $x = x - 1$;当用于数组下标时,在自减运算前先求变量值
4	+ =	自加指定值,例如 $x += 3$ 相当于 $x = x + 3$, $x += (-3)$ 相当于 $x = x + (-3)$
5	- =	自减指定值,例如 $x -= 3$ 相当于 $x = x - 3$, $x -= (-3)$ 相当于 $x = x - (-3)$

4.6 数学函数

数学函数计算公式如下:

$$\text{Abs}(x) = \begin{cases} x, & x \geq 0 \\ -x, & x < 0 \end{cases}$$

Ceil(x) 取不小于 x 的最小整数

Clip1Y(x) = Clip3(0, (1 << BitDepthY) - 1, x)

Clip1C(x) = Clip3(0, (1 << BitDepthC) - 1, x)

$$\text{Clip3}(x, y, z) = \begin{cases} x, & z < x \\ y, & z > y \\ z, & \text{其他} \end{cases}$$

cos(x) 表示 x 的余弦函数

$\text{C}_M^N = \frac{M!}{N! (M-N)!}$ 表示从 M 个数中取出 N 个数的组合数

exp(x) 表示 e 的 x 次幂

Floor(x) 取不大于 x 的最大整数

ln(x) 取以 e 为底的 x 的对数

log₁₀(x) 取以 10 为底的 x 的对数

Median(x, y, z) = $x + y + z - \text{Min}(x, \text{Min}(y, z)) - \text{Max}(x, \text{Max}(y, z))$

$$\text{Min}(x, y) = \begin{cases} x, & x \leq y \\ y, & x > y \end{cases}$$

$$\text{Max}(x, y) = \begin{cases} x, & x \geq y \\ y, & x < y \end{cases}$$

Round(x) = Sign(x) \times Floor(Abs(x) + 0.5)

$$\text{Sign}(x) = \begin{cases} 1, & x \geq 0 \\ -1, & x < 0 \end{cases}$$

sin(x) 表示 x 的正弦函数

4.7 语法元素、变量和表

语法元素在比特流中以粗体字出现。当表格或正文中用到某个语法元素的值时,则以常规字体出

现。每个语法元素均表示为名称(所有字母小写,以下划线连接),和一到两个代表其编码表示方式的描述符。解码过程根据语法元素以及之前已解码的语法元素的取值进行解码。

某些情况下语法表可能使用根据语法元素值导出的其他变量的值。这些变量出现在语法表或正文中,以小写和大写混合的形式命名,并且名称中不含下划线。以大写字母开头的变量是根据当前语法结构和所有相关语法结构的解码导出的。

在某些情况下,语法元素值或变量值的识记名称与其数值等同。有时,识记名称与其值无关。二者的关联在正文中做出规定。识记名称由一组或多组字母由下划线连接而成。每组字母均以大写字母开头,可包括多个大写字母。

函数用名称来描述,函数名由语法元素名称和左右圆括号中的零个或多个以逗号(若多个变量时)分隔的变量名称(用于定义)或值(用于使用)构成。

一维的阵列称为数组,二维的阵列称为矩阵。阵列可以是语法元素,也可以是变量。下标或方括号可用来表示一个阵列的索引。对于一个矩阵,第一个下标为行(垂直)索引,第二个下标为列(水平)索引。使用方括号表示时,索引的顺序则正好相反。比如,一个矩阵 S 中的水平位置 x 和垂直位置 y 上的元素可表示为 $S[x, y]$ 或 S_{yx} 。

单引号之间的一串比特值为二进制符号。例如,“10000100”表示一个第一位和倒数第三位等于 1 的 8 比特串。

十六进制符号,以前缀“0x”表示,当所表示的比特位数为 4 的整数倍时可替代二进制符号使用。例如,“0x84”表示一个第一位和倒数第三位等于 1 的 8 比特串。

不使用单引号括起来的或不带前缀“0x”的数值为十进制值。

条件语句中等于 0 的值代表假(FALSE)的情况,用其他非零值代表真(TRUE)。

4.8 逻辑运算符的文字描述

在正文中,含有逻辑运算符的下列伪码语句:

```
if(条件 0)
    语句 0
else if(条件 1)
    语句 1
.....
else /* 解释其他情况的注释 */
    语句 n
```

可描述如下:

- 如果条件 0,则语句 0
- 否则,如果条件 1,语句 1
-
- 否则(说明性文字,表示其他情况),语句 n

正文中的每个“如果……否则,如果……否则……”语句都是由“……如下……”或“……应用下列规则”引导的,后面紧跟“如果……”。最后一个“如果……否则,如果……否则……”语句的条件一般是“否则,……”。交替出现的“如果……否则,如果……否则……”语句可以通过将“……如下……”或“……应用下列规则”和最后的“否则,……”配对加以识别。

正文中,一个以下列伪码描述的逻辑运算语句:

```
if(条件 0a && 条件 0b)
    语句 0
else if(条件 1a || 条件 1b)
```

语句 1

.....

else

语句 n

可描述如下：

- a) 如果下列所有条件为真,语句 0
 - 条件 0a
 - 条件 0b
- b) 否则,如果下列任何一个条件为真,语句 1
 - 条件 1a
 - 条件 1b
- c)
- d) 否则,语句 n

正文中,一个以下列伪码描述的逻辑运算语句:

if(条件 0)

 语句 0

if(条件 1)

 语句 1



可描述如下：

- 当条件 0 时,语句 0
- 当条件 1 时,语句 1

4.9 过程

过程用于描述语法元素的解码。所有属于当前语法结构的语法元素和大写的变量,以及相关的语法结构,在过程的规范和调用中都是可用的。过程的规范中可能还含有明确指定为输入的小写的变量。每个规范均明确地规定了输出。输出可以是大写的变量,也可以是小写的变量。

在过程的规范中,一个特定宏块可用一个值与其宏块索引相等的变量名指代。

5 视频部分

5.1 编码比特流和输出数据的格式

5.1.1 比特流格式

本条规定 NAL 单元流和字节流之间的关系,二者均称为比特流。

NAL 单元流格式由一系列称为 NAL 单元的语法结构组成,按照解码顺序排序。NAL 单元流中 NAL 单元的解码顺序和内容是受约束的。

字节流可以用 NAL 单元流构造,通过将 NAL 单元按照解码顺序排列,并且为每个 NAL 单元添加一个起始码前缀和若干零值字节形成一个字节流。NAL 单元流格式可以通过在字节流中搜索唯一的起始码前缀,从字节流格式中提取出来。除字节流格式以外,构造 NAL 单元的其他方法,本标准不做规定。字节流格式在附录 B 中规定。

5.1.2 图像格式

本条规定由比特流确定的源与已解码帧之间的关系。

比特流所表示的视频源是一系列按解码顺序排列的帧。

每个源或已解码帧都是由一个或多个视频样点阵列组成的：

——仅亮度(Y)(单色)的阵列；

——亮度和两个色度(YCbCr)的阵列；

——绿、蓝和红(GBR,也称为RGB)的阵列；

——表示其他未定义的单色或三基色样点(例如YZX,也称为XYZ)的阵列。

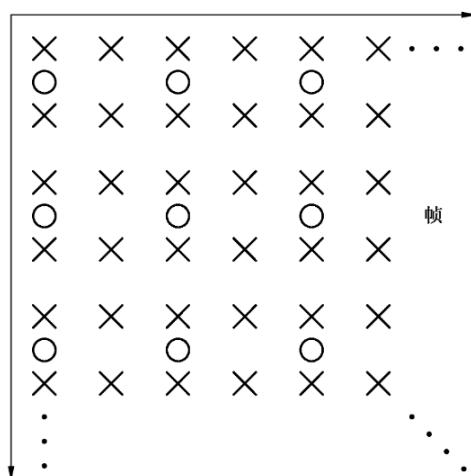
为了便于标记和命名,本标准不考虑实际使用的颜色表示方法,与这些阵列相关的变量和词语均指亮度和色度,亮度阵列用Y表示,两个色度阵列分别用Cb和Cr表示。

本标准支持的色彩格式有4:2:0和4:2:2。

在4:2:0格式下,两个色度阵列的高度和宽度均为亮度阵列的一半。在4:2:2格式下,两个色度阵列的高度等于亮度阵列的高度,宽度为亮度阵列的一半。

除非特别说明,亮度和色度(当出现时)阵列的语法顺序为:当三个分量的数据都出现时,首先是亮度阵列的数据,然后是Cb阵列数据,最后是Cr阵列数据。

视频序列中用来表示每个亮度或色度样点的比特位数至少为8,表示亮度阵列样点的比特位数和表示色度阵列样点的比特位数应相同。在4:2:0格式下,一帧中亮度和色度样点的垂直和水平相对位置如图1所示。



说明:

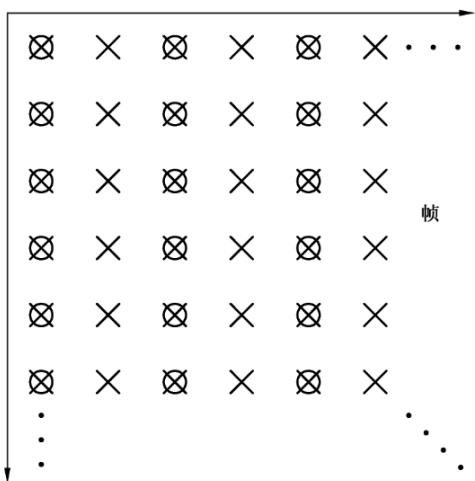
X——亮度样点的位置;

O——色度样点的位置。

图1 帧图像中4:2:0亮度和色度样点垂直和水平位置



在4:2:2格式下,色度样点和对应的亮度样点处于同一位置上,帧中的样点位置如图2所示。



说明：

×——亮度样点的位置；

○——色度样点的位置。

图 2 帧图像中 4:2:2 亮度和色度样点的垂直和水平位置

一帧图像中左上角亮度样点的位置坐标 (x, y) 为 $(0, 0)$, 样点每右移一列, x 的取值增加 1, 样点每下移一行, y 的取值增加 1。

5.1.3 图像的空间分割

5.1.3.1 编码片的划分

本条规定一幅图像如何分割为编码片(Tile)和树形编码单元(CTU)。tile_enable 等于 0 时,整帧图像只有一个编码片;tile_enable 等于 1 时,图像有可能被划分为多个编码片。编码片由一系列的树形编码单元组成。树形编码单元为编码的基本单元,每个树形编码单元包含一个亮度阵列及两个色度阵列。图像左上角的 CTU 的索引等于 0,CTU 在图像中的索引按照光栅扫描顺序递增。

将一幅图像从水平和垂直方向上分割为若干个矩形区域，每个矩形区域称为一个 Tile。每个 Tile 包含若干个 CTU，其可以并行独立编解码。

图像宽度大于等于 8 个最大树形编码单元尺寸时,可以在水平方向上划分为多个 Tile 列。图像宽度超过 64 个最大树形编码单元尺寸时,必须划分为多个 Tile 列。每个 Tile 在水平方向至少包含 4 个 CTU,最多包括 64 个 CTU。图像在垂直方向上可以划分为多个 Tile 行,Tile 行的个数可以等于 1,2 或者 4。

图 3 为图像的 Tile 划分示意图, 图中图像被划分为 2 个 Tile 行和 4 个 Tile 列, 每个 Tile 中包含 6×3 个树形编码单元。每个 Tile 的宽度和高度按照 Tile 的列数和行数决定。图像的宽度和高度分别按照 Tile 列数和行数均分, 得到 Tile 的宽度和高度。当无法整除时, 最右一列和最下一行的 Tile 与其他 Tile 宽度和高度不同。

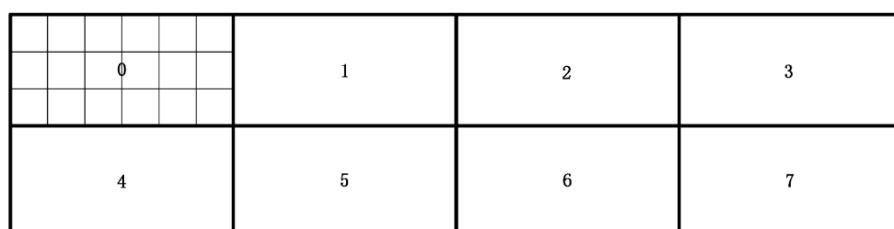


图 3 编码片划分与顺序示意图

在码流中,图像中的所有 Tile 应按照光栅扫描顺序进行传输,每个 Tile 中的 CTU 也应按照在 Tile 中的光栅扫描顺序进行传输。

5.1.3.2 树形编码单元(CTU)、预测单元(PU)、变换单元(TU)

本条规定树形编码单元如何进一步划分为预测单元、变换单元。树形编码单元之间不应重叠,树形编码单元左上角的样点不应超出图像边界,树形编码单元右下角的样点可超出图像边界。

当序列参数集中的 extended_sb_size_flag 的取值等于 1 时,树形编码单元 $2N \times 2N$ 的尺寸为 128×128 ,当 extended_sb_size_flag 的取值等于 0 时,树形编码单元的尺寸为 64×64 。树形编码单元可划分为一个或多个预测单元,划分方式由编码树决定。每级的 $2N \times 2N$ 尺寸的预测单元,可进一步划分为 $2N \times 2N$ 、 $N \times N$ 、 $2N \times N$ 、 $N \times 2N$ 等四种模式,如果该级划分为 $N \times N$,则进入下一级选择进一步的划分方式。对于 64×64 的树形编码单元,其编码树如图 4 所示,预测单元共有 13 种尺寸: 64×64 、 32×64 、 64×32 、 32×32 、 16×32 、 32×16 、 16×16 、 8×16 、 16×8 、 8×8 、 4×8 、 8×4 、 4×4 。对于 128×128 的树形编码单元,其预测单元共有 16 种尺寸: 128×128 、 64×128 、 128×64 、 64×64 、 32×64 、 64×32 、 32×32 、 16×32 、 32×16 、 16×16 、 8×16 、 16×8 、 8×8 、 4×8 、 8×4 、 4×4 。

编码树的扫描顺序如图 5 所示,矩形里的数字表示该块在编码时的处理顺序。

预测单元是帧内预测与帧间预测的基本单元。预测单元可进一步划分为一个或多个变换单元。变换单元是进行变换/量化的基本单元。变换单元共有 4 种尺寸: 32×32 、 16×16 、 8×8 、 4×4 。tx_mode 不等于 TX_MODE_SELECT 时,变换单元的尺寸为 tx_mode 允许的适合预测单元的最大尺寸;tx_mode 等于 TX_MODE_SELECT 时,变换单元的尺寸由 tx_size 确定。

如果是帧间预测,预测块尺寸与预测单元相同。当预测单元小于等于 64×64 时,变换单元在预测单元内依据从左到右,从上到下的光栅扫描顺序扫描;当预测单元大于 64×64 时,先将预测单元以 64×64 为单位按光栅扫描进一步划分,然后在每一个 64×64 的划分内部,变换单元按光栅扫描顺序依次扫描。

如果是帧内预测,预测单元可以进一步拆分为多个相同尺寸的预测块,每一个预测块对应一个变换单元。当预测单元小于等于 64×64 时,预测块及其对应的变换单元依据从左到右,从上到下的光栅扫描顺序;当预测单元大于 64×64 时,先将预测单元以 64×64 为单位按光栅扫描进一步划分,然后在每一个 64×64 的划分内部,预测块及其对应的变换单元按光栅扫描顺序依次扫描。

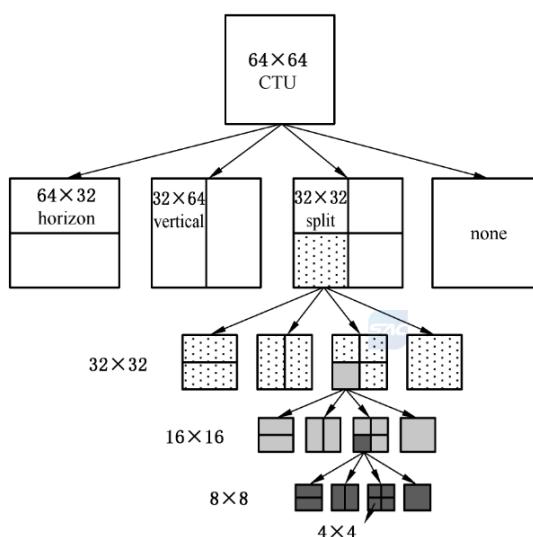


图 4 编码树划分方式

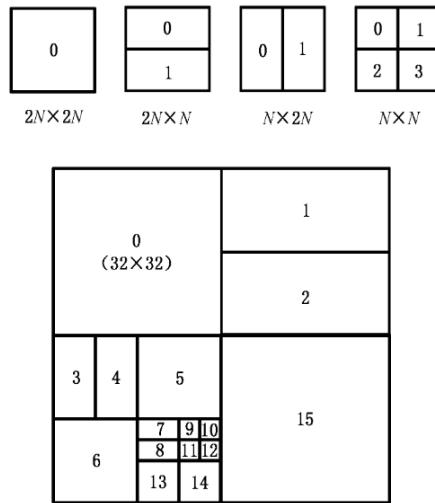


图 5 编码树的扫描顺序

5.1.3.3 相邻块可用性的推导过程

如图 6 所示,当前预测块大小为 $M \times N$,其参考样点按区域可分为 5 部分:左下(A)、左侧(B)、左上(C)、上方(D)和右上(E),一共 $2 \times (M + N) + 1$ 个点。设当前块左上角样点在图像中的坐标是 (x_0, y_0) ,右下角样点在图像中的坐标是 (x_1, y_1) ,块 X(X 为 A、B、C、D 或 E)为表 6 中列出的样点所属的块。表 6 中坐标均为样点在帧图像中的位置坐标。

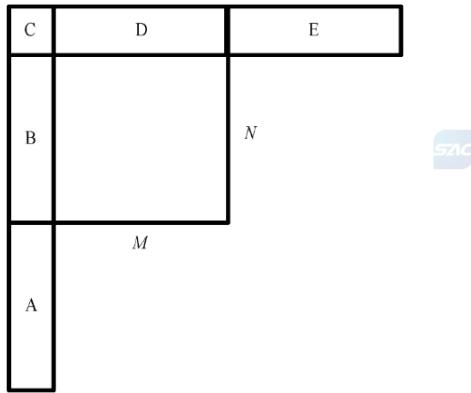


图 6 当前块与相邻块的空间位置关系

表 6 块 A、B、C、D 和 E 的位置

块 A 右上角样点坐标	块 B 右上角样点坐标	块 C 右下角样点坐标	块 D 左下角样点坐标	块 E 左下角样点坐标
$(x_0 - 1, y_1 + 1)$	$(x_0 - 1, y_0)$	$(x_0 - 1, y_0 - 1)$	$(x_0, y_0 - 1)$	$(x_1 + 1, y_0 - 1)$

如果一相邻块 X(X 为 A、B、C、D 或 E)在图像内并且该块应与当前块属于同一 Tile, 则该相邻块标记为存在; 否则标记为不存在。

如果一相邻块标记为不存在或者尚未解码, 则该块标记为不可用; 否则标记为可用。如果某样点所在的块标记为不存在或者该样点尚未解码, 则该样点标记为不可用; 否则标记为可用。

5.1.3.4 感兴趣区域(ROI)的划分

一个图像中可划分出若干感兴趣区域(ROI), ROI 的最小单位为 8×8 , 并且同一预测单元内的样点应属于同一 ROI 区域。如果图像中存在 ROI 划分, 对应图像参数集中的参数 segmentation_enable 的取值等于 1, 如果图像中不存在 ROI 划分, 对应图像参数集中的参数 segmentation_enable 的取值应等于 0。一个图像中的 ROI 区域可分为 8 个不同等级, 该等级由样点所在块的 segment_id 指示。图像中不存在 ROI 划分时, 所有样点所在块的 segment_id 应等于 0。

5.2 语法和语义

5.2.1 以表格形式描述语法的方法

语法表格规定了所有允许的比特流的超集。附加的语法限定可能在其他条中直接或间接规定。

注: 实际的解码器宜有识别比特流入口点的方法, 并且可以分辨和处理不一致的比特流。分辨和处理错误以及类似情形的方法不在本标准中描述。

表 7 给出了描述语法的伪代码例子。规定了当 syntax_element 出现时, 从比特流中解析语法元素, 并将指针移向比特流中下一个语法元素位置上的过程。

表 7 伪代码例程表

伪代码描述语言	描述符
/* 语句可以是一个关联某一语法类别的语法元素和描述符, 或者用于说明语法元素的存在、类型和数值的表达式, 下面给出两个例子。 */	
syntax_element	ue(v)
条件语句	
/* 花括号括起来的语句组是复合语句, 在功能上视作单个语句。 */	
{	
语句	
语句	
...	
}	
/* “while”语句测试条件是否为 TRUE, 如果为 TRUE, 则重复执行循环体, 直到条件不为 TRUE。 */	
while(条件)	
语句	

表 7 (续)

伪代码描述语言	描述符
/* “do … while”语句先执行循环体一次,然后测试条件是否为 TRUE,如果为 TRUE,则重复执行循环体,直到条件不为 TRUE。 */	
do	
语句	
while(条件)	
/* “if … else”语句首先测试条件,如果为 TRUE,则执行主要语句,否则执行另选语句。如果另选语句不需要执行,结构的“else”部分和相关的另选语句可忽略。 */	
if(条件)	
主要语句	
else	
另选语句	
/* “for”语句首先执行最初语句,然后测试条件,如果条件为 TRUE,则重复执行主要语句和随后语句直到条件不为 TRUE。 */	
for(最初语句;条件;随后语句)	
主要语句	

5.2.2 语法函数和描述符的规范

5.2.2.1 语法函数的规范

以下函数用于语法描述。这些函数假定解码器中存在一个比特流指针,这个指针指向比特流中解码过程要读取的下一比特的位置。具体要求如下:

byte_aligned()的规定:

- 如果比特流的当前位置是在字节的边界,即比特流中的下一比特是字节的第一个比特,那么 byte_aligned()的返回值为 TRUE;
- 否则,byte_aligned()的返回值为 FALSE。

get_left_ae_bits()的规定:

- 熵解码器中的计数器 count 加上 8 然后对 8 求模,如果等于 0,则通过固定概率 128 继续解析出 16 比特;
- 如果不等于 0,则通过固定概率 128 继续解析出求模后的值加 8 比特。

more_data_in_byte_stream(),在附录 B 规定的字节流 NAL 单元语法结构中使用,规定:

- 如果字节流中后续还有更多数据,more_data_in_byte_stream()的返回值为 TRUE;
- 否则,more_data_in_byte_stream()的返回值为 FALSE。

more_rbsp_data()的规定:

- 如果在 rbsp_trailing_bits()之前的 RBSP 中有更多数据,more_rbsp_data()的返回值为 TRUE;
- 否则,more_rbsp_data()的返回值为 FALSE。

判断 RBSP 中是否有更多数据的方法不在本标准中规定。

`next_bits(n)`提供比特流中接下来的 n 个比特,不改变比特流指针。该函数使比特流中的下 n 个比特可见。当用在附录 B 规定的字节流中时,如果剩余的字节流已不足 n 个比特,`next_bits(n)`返回值为 0。

`read_bits(n)`从比特流中读取下面的 n 个比特,并且将比特流指针向前移动 n 个比特。当 n 等于 0 时,

`read_bits(n)`的返回值为 0 并且不移动比特流指针。

5.2.2.2 描述符的规范

下述描述符规定了每个语法元素的解析过程:

- ae(v):二进制算术编码语法元素。该描述符的解析过程在 5.4.2 中规定;
- b(8):任意形式的 8 比特字节。该描述符的解析过程通过函数 `read_bits(8)`的返回值来规定;
- f(n):n 位比特串(由左至右),左位在先,该描述符的解析过程通过函数 `read_bits(n)`的返回值来规定;
- i(n):n 位有符号整数。在语法表中,如果 n 是‘v’,其比特数由其他语法元素值确定。解析过程由函数 `read_bits(n)`的返回值规定,该返回值用最高有效位在前的 2 的补码表示;
- se(v):有符号整数指数哥伦布码编码的语法元素,左位在先。解析过程在 5.4.3 中定义;
- u(n):n 位无符号整数。在语法表中,如果 n 是‘v’,其比特数由其他语法元素值确定。解析过程由函数 `read_bits(n)`的返回值规定,该返回值用最高有效位在前的二进制表示;
- ue(v):无符号整数指数哥伦布码编码的语法元素,左位在先。解析过程在 5.4.3 中定义。

5.2.3 以表格形式表示的语法

5.2.3.1 NAL 单元语法

NAL 单元语法见表 8。

表 8 NAL 单元语法表

描述符
nal_unit(NumBytesInNALunit) {
forbidden_zero_bit
nal_ref_idc
nal_unit_type
encryption_idc
authentication_idc
NumBytesInPayload = 0
for(i=1; i<NumBytesInNALunit; i++) {
if(i+2 < NumBytesInNALunit && next_bits(24) == 0x000003) {
payload_byte [NumBytesInPayload++]
payload_byte [NumBytesInPayload++]
i += 2
emulation_prevention_three_byte /* 应等于 0x03 */

表 8 (续)

nal_unit(NumBytesInNALunit) {	描述符
}	
else	
payload_byte [NumBytesInPayload++]	b(8)
}	
}	

5.2.3.2 RBSP 语法

5.2.3.2.1 序列参数集 RBSP 语法

序列参数集 RBSP 语法见表 9。

表 9 序列参数集 RBSP 语法表

seq_parameter_set_rbsp() {	描述符
profile_id	u(8)
level_id	u(8)
ldp_mode_flag	u(1)
frame_width_minus_1	u(16)
frame_height_minus_1	u(16)
chroma_format_idc	u(2)
bit_depth	u(2)
refs_per_frame	u(3)
frame_rate	u(3)
extended_sb_size_flag	u(1)
tile_enable	u(1)
wpp_enable	u(1)
sao_enable	u(1)
alf_enable	u(1)
roi_flag	u(1)
temporal_svc_flag	
if(temporal_svc_flag)	
layer_num_minus_1	u(2)
spatial_svc_flag	u(2)
if(spatial_svc_flag) {	
svc_ratio	u(3)
svc_mode	u(1)

表 9 (续)

seq_parameter_set_rbsp() {	描述符
}	
if (frame_rate >= 4){	
vui_parameters_present_flag	u(1)
if(vui_parameters_present_flag)	
vui_parameters()	
}	
rbsp_trailing_bits()	
}	

5.2.3.2.2 图像参数集 RBSP 语法

图像参数集 RBSP 语法见表 10。

表 10 图像参数集 RBSP 语法表

pic_parameter_set_rbsp() {	描述符
frame_num	u(8)
if(temporal_svc_flag)	
layer_id	u(3)
if(spatial_svc_flag){	
if(roi_flag & svc_mode)	
svc_roi_flag	u(1)
if(svc_roi_flag == 1){	
svc_top_left	u(16)
svc_bottom_right	u(16)
}	
}	
frame_type	u(1)
if(! roi_flag)	
ctu_dqp_enable	u(1)
if(ctu_dqp_enable){	
min_dqp_partition_size	u(3)
}	
if(frame_type != 0){	
refresh_frame_flags	u(5)
update_rps_flag	u(1)

表 10 (续)

pic_parameter_set_rbsp() {	描述符
rps_idx	u(6)
if (update_rps_flag){	
for(i=0;i< refs_per_frame;i++){	
if (i == 2)	
opt_minus_flag	u(1)
delta_poc[i]	u(6)
}	
}	
refresh_pictures_num	u(3)
for(i=0;i< refresh_pictures_num;i++)	
delta_poc[i]	u(6)
allow_high_precision_mv	u(1)
interp_filter_switchable	u(1)
if(! interp_filter_switchable)	
interp_filter	u(3)
}	
filter_level	u(6)
sharpness_level	u(3)
If_delta_enable	u(1)
if(lf_delta_enable){	
If_delta_update	u(1)
if(lf_delta_update){	
for(i=0;i<6;i++){	
If_ref_delta_enable[i]	u(1)
if(lf_ref_delta_enable[i]){	
If_ref_deltas[i]	u(6)
If_ref_deltas_sign[i]	u(1)
}	
}	
for(i=0;i<2;i++){	
If_mode_delta_enable[i]	u(1)
if(lf_mode_delta_enable[i]){	
If_mode_deltas[i]	u(6)
If_mode_deltas_sign[i]	u(1)

表 10 (续)

pic_parameter_set_rbsp() {	描述符
}	
}	
}	
}	
if(sao_enable)	
for (compIdx=0; compIdx<3; compIdx++)	
picture_sao_enable[compIdx]	u(1)
if(alf_enable)	
read_alf()	
base_qindex	u(8)
y_dc_delta_q_update_flag	u(1)
if(y_dc_delta_q_update_flag){	
y_dc_delta_q	u(4)
y_dc_delta_q_sign	u(1)
}	
uv_dc_delta_q_update_flag	u(1)
if(uv_dc_delta_q_update_flag){	
uv_dc_delta_q	u(4)
uv_dc_delta_q_sign	u(1)
}	
uv_ac_delta_q_update_flag	u(1)
if(uv_ac_delta_q_update_flag){	
uv_ac_delta_q	u(4)
uv_ac_delta_q_sign	u(1)
}	
if(roi_flag)	
segmentation_enable	u(1)
if(segmentation_enable){	
segmentation_update_map	u(1)
if(segmentation_update_map){	
for(i=0;i<7;i++){	
seg_tree_flag[i]	u(1)
if(seg_tree_flag)	
seg_tree_probs[i]	u(8)

表 10 (续)

pic_parameter_set_rbsp() {	描述符
}	
seg_temporal_update	u(1)
if(seg_temporal_update){	
for (i=0;i<3;i++){	
seg_pred_flag[i]	u(1)
if(seg_pred_flag)	
seg_pred_probs[i]	u(8)
}	
}	
}	
seg_update_data	u(1)
if(seg_update_data){	
seg_abs_delta	u(1)
for(i=0;i<8;i++){	
for(j=0;j<4;j++){	
feature_enable[i][j]	70 u(1)
if(feature_enable){	
seg_feature_data[i][j]	u(v)
if(is_segfeature_signed[j])	
seg_feature_data_sign[i][j]	u(1)
}	
}	
}	
}	
if(tile_enable){	
tile_cols_log2 = minLog2TileCols	
while(tile_cols_log2 < maxLog2TileCols){	
increment_tile_cols_log2	u(1)
if(increment_tile_cols_log2 == 1)	
tile_cols_log2++	
else	
break;	
}	

表 10 (续)

pic_parameter_set_rbsp() {	描述符
tile_rows_log2	u(1)
if(tile_rows_log2 == 1){	
tile_rows_delta	u(1)
tile_rows_log2 += tile_rows_delta	
}	
}	
while(byte_aligned() == FALSE){	
reserved_bit	u(1)
}	
tx_mode	ae(v)
if(tx_mode == ALLOW_TX_32×32)	
tx_mode_delta	ae(v)
if(tx_mode == TX_MODE_SELECT){	
for(i=0;i<2;i++)	
diff_update_prob_8×8	ae(v)
for(i=0;i<2;i++)	
for(j=0;j<2;j++)	
diff_update_prob_16×16	ae(v)
for(i=0;i<2;i++)	
for(j=0;j<3;j++)	
diff_update_prob_32×32	ae(v)
}	
for(t=TX_4×4;t<= MAX_TX_SIZE;t++) {	
coef_update_prob_flag	ae(v)
if(coef_update_prob_flag){	
for(i=0;i<2;i++)	
for(j=0;j<2;j++)	
for(k=0;k<6;k++)	
for(l=0;l<(k == 0 ? 3 : 6);l++)	
for(m=0;m<3;m++)	
diff_update_prob_coef	ae(v)
}	
}	
for(k = 0; k<3; k++)	

表 10 (续)

pic_parameter_set_rbsp() {	描述符
diff_update_prob_skip	ae(v)
if(alf_enable == 1){	
for(compIdx=0;compIdx<3;compIdx++)	
if(picture_alf_enable [compIdx])	
for (k = 0; k<4; k++)	
SAC diff_update_prob_alf	ae(v)
}	
if (frame_type == 1){	
for(i=0;i<7;+ i)	
diff_update_prob_newmv	ae(v)
for(i=0;i<2;+ i)	
diff_update_prob_zeromv	ae(v)
for(i=0;i<8;+ i)	
diff_update_prob_refmv	ae(v)
if(interp_filter_switchable)	
for(j=0;j<5;+ j)	
for(i=0;i<3;+ i)	
diff_switchable_interp_probs	ae(v)
for(i=0;i<4;i++)	
diff_intrainter_update_prob	ae(v)
not_single_ref	ae(v)
if(not_single_ref)	
not_compound_ref	ae(v)
if(frame_reference_mode==REFERENCE_MODE_SELECT)	
for(i=0;i<5;+ i)	
diff_inter_update_prob	ae(v)
if(frame_reference_mode! = COMPOUND_REFERENCE){	
for(i=0;i<5;+ i){	
for(j=0;j<4;+ j)	
diff_single_ref_update_prob	ae(v)
}	
}	
if(frame_reference_mode! = SINGLE_REFERENCE){	
for(i=0;i<5;+ i)	

表 10 (续)

pic_parameter_set_rbsp() {	描述符
for(j=0;j<3;j++)	
diff_comp_ref_update_prob	ae(v)
}	
for(j=0;j<20;j++)	
for(i=0;i<3;i++)	
diff_partition_update_prob	ae(v)
for(j=0;j<2;j++)	
for(i=0;i<7;i++)	
diff_delta_q_update_prob	ae(v)
read_mv_prob()	
}	
get_left_ae_bits()	
rbsp_trailing_bits()	
}	SAC

read_mv_prob()语法定义见表 11。

表 11 read_mv_prob 语法表

read_mv_prob()	描述符
for(j=0;j<3;j++)	
mv_joint_probs[j]	ae(v)
for(i=0;i<2;i++) {	
mv_sign_prob[i]	ae(v)
for(j=0;j<10;j++)	
mv_class_probs[i][j]	ae(v)
mv_class0_bit_prob[i]	ae(v)
for(j=0;j<10;j++)	
mv_bits_prob[i][j]	ae(v)
}	
for(i=0;i<2;i++) {	
for(j=0;j<2;j++)	
for(k=0;k<2;k++)	

表 11 (续)

read_mv_prob()	描述符
mv_class0_fr_probs[i][j][k]	ae(v)
for(k=0;k<2;k++)	
mv_fr_probs[i][k]	ae(v)
}	
if(allow_high_precision_mv){	
for(i=0;i<2;i++){	
mv_class0_hp_prob[i]	ae(v)
mv_hp_prob[i]	ae(v)
}	
}	
}	

read_alf()语法定义见表 12。

表 12 read_alf 语法表

read_alf()	描述符
if(alf_enable == 1){	
for(compIdx=0;compIdx<3;compIdx++)	
picture_alf_enable[compIdx]	u(1)
if(picture_alf_enable[0]==1 picture_alf_enable[1]==1 picture_alf_enable[2]==1)	
alf_parameter_set()	
}	
}	

alf_parameter_set()语法定义见表 13。

表 13 alf_parameter_set 语法表

alf_parameter_set()	描述符
if(picture_alf_enable[0]==1){	
alf_filter_num_minus1	u(4)
for(i=0;i<alf_filter_num_minus1+1;i++){	
if(i>0 && alf_filter_num_minus1 != 15)	
alf_region_distance[i]	u(4)
for(j=0;j<10;j++)	
alf_coeff_luma[i][j]	se(v)

表 13 (续)

alf_parameter_set() {	描述符
}	
}	
if(picture_alf_enable[1]==1)	
for(j=0;j<10;j++)	
alf_coeff_chroma[0][j]	se(v)
if(picture_alf_enable[2]==1)	
for(j=0;j<10;j++)	
alf_coeff_chroma[1][j]	se(v)
}	

5.2.3.2.3 安全参数集 RBSP 语法

安全参数集 RBSP 语法见表 14。

表 14 安全参数集 RBSP 语法表

sec_parameter_set_rbsp() {	描述符
encryption_flag	u(1)
authentication_flag	u(1)
if(encryption_flag) {	
encryption_type	u(4)
vek_flag	u(1)
iv_flag	u(1)
if(vek_flag) {	
vek_encryption_type	u(4)
evek_length_minus1	u(8)
evek	f(n)
vkek_version_length_minus1	u(8)
vkek_version	f(n)
}	
if(iv_flag) {	
iv_length_minus1	u(8)
iv	f(n)
}	
}	
if(authentication_flag) {	

表 14 (续)

sec_parameter_set_rbsp() {	描述符
hash_type	u(2)
hash_discard_p_pictures	u(1)
signature_type	u(2)
successive_hash_pictures_minus1	u(8)
camera_idc	f(152)
}	
if(vek_flag authentication_flag)	
camera_id	f(160)
rbsp_trailing_bits()	
}	

SAC

5.2.3.2.4 补充增强信息 RBSP 语法

补充增强信息 RBSP 语法见表 15。

表 15 补充增强信息 RBSP 语法表

sei_rbsp() {	描述符
do	
sei_message()	
while(more_rbsp_data())	
rbsp_trailing_bits()	
}	

补充增强信息消息语法见表 16。

表 16 补充增强信息消息语法表

sei_message() {	描述符
PayloadType = 0	
while(next_bits(8) == 0xFF) {	
ff_byte /* 应等于 0xFF */	f(8)
PayloadType += 255	
}	
last_payload_type_byte	u(8)
PayloadType += last_payload_type_byte	
PayloadSize = 0	

表 16 (续)

sei_message() {	描述符
while(next_bits(8) == 0xFF) {	
ff_byte /* 应等于 0xFF */ /	f(8)
PayloadSize += 255	
}	
last_payload_size_byte	u(8)
PayloadSize += last_payload_size_byte	
sei_payload(PayloadType, PayloadSize)	
}	

5.2.3.2.5 编码片 RBSP 语法

编码片 RBSP 语法见表 17。

表 17 编码片 RBSP 语法表

tile_data_rbsp() {	描述符
if(tile_enable)	
tile_idx	ae(v)
for(n=0;n<ctu_num_in_tile;n++) {	
if(wpp_enable && (0 == n % SbCols))	
substream_len	u(32)
if(sao_enable && (picture_sao_enable[0] picture_sao_enable[1] picture_sao_enable[2])) {	
if(MergeUpAvail MergeLeftAvail)	
sao_merge_flag	ae(v)
if(sao_merge_flag) {	
if(MergeUpAvail && MergeLeftAvail)	
sao_merge_type	ae(v)
else {	
for (compIdx = 0; compIdx < 3; compIdx++) {	
if(picture_sao_enable[compIdx]) {	
sao_mode[compIdx]	ae(v)
if(sao_mode[compIdx] != 0) {	
sao_type[compIdx]	ae(v)
if(sao_type[compIdx] == 0) {	
sao_start_band[compIdx]	ae(v)

表 17 (续)

tile_data_rbsp() {	描述符
for(j=0;j<4;j++) {	
sao_offset_abs[compIdx][j];	ae(v)
if(sao_offset_abs[compIdx][j] != 0)	
sao_offset_sign[compIdx][j]	ae(v)
}	
}	
else {	
sao_edge_type[compIdx]	ae(v)
sao_edge_offset[compIdx][0]	ae(v)
sao_edge_offset[compIdx][1]	ae(v)
sao_edge_offset[compIdx][2]	ae(v)
sao_edge_offset[compIdx][3]	ae(v)
}	
}	
}	
}	
}	
}	
if (alf_enable)	
for (compIdx = 0; compIdx < 3; compIdx++)	
alf_ctu_enable[compIdx]	ae(v)
coding_tree_unit(n)	
}	
get_left_ae_bits()	
rbsp_trailing_bits()	
}	

5.2.3.2.6 认证数据 RBSP 语法

认证数据 RBSP 语法见表 18。

表 18 认证数据 RBSP 语法表

authentication_data_rbsp()	描述符
frame_num	u(8)
if(spatial_svc_flag)	
spatial_el_flag	u(8)
authentication_data_length_minus1	u(8)
for(i=0;i< authentication_data_length_minus1+1;i++)	
authentication_data[i]	u(8)
rbsp_trailing_bits()	
}	

5.2.3.2.7 流结尾 RBSP 语法

流结尾 RBSP 语法见表 19。

表 19 流结尾 RBSP 语法表

end_of_stream_rbsp() {	描述符
}	

5.2.3.2.8 RBSP 尾比特语法

RBSP 尾比特语法见表 20。

表 20 RBSP 尾比特语法表

rbsp_trailing_bits() {	描述符
rbsp_stop_one_bit /* 应等于 1 */	f(1)
while(! byte_aligned())	
rbsp_alignment_zero_bit /* 应等于 0 */	f(1)
}	

5.2.3.3 CTU 语法

CTU 语法见表 21。

表 21 CTU 语法表

coding_tree_unit() {	描述符
if(hasCols hasRows)	
partition	ae(v)

表 21 (续)

描述符
coding_tree_unit() {
if(ctu_dqp_enable &&.bsize>= min_dqp_partition_size)
iscoded = 0
if (subsize < BLOCK_8×8)
block(0)
else{
switch(partition){
case PARTITION_NONE:
block(0)
break
case PARTITION_HORZ:
block(0)
block(1)
break
case PARTITION_VERT:
block(0)
block(1)
break
case PARTITION_SPLIT:
coding_tree_unit(0)
coding_tree_unit(1)
coding_tree_unit(2)
coding_tree_unit(3)
break
}
}
}

5.2.3.4 块语法

块语法见表 22。

表 22 块语法

描述符
block(j) {
if(frame_type == 0){
if (segmentation_enable &&.segmentation_update_map)

表 22 (续)

描述符
block(j) {
segment_id
if(segmentation_enable && FeatureEnabled[segment_id][SEG_LVL_SKIP])
skip_flag = 1
else
skip_flag
if(tx_mode == TX_MODE_SELECT && bsize >= BLOCK_8×8)
tx_size
switch (bsize) {
case BLOCK_4×4:{
for(i=0;i<4;i++)
read_block_intra_luma_mode(i)
break
}
case BLOCK_4×8:{
read_block_intra_luma_mode(0)
read_block_intra_luma_mode(1)
break
}
case BLOCK_8×4:{
read_block_intra_luma_mode(0)
read_block_intra_luma_mode(1)
break
}
default:
read_block_intra_luma_mode(0)
}
read_block_intra_chroma_mode()
}
else{
if (segmentation_enable) {
if (segmentation_update_map) {
if (segmentation_temporal_update){
seg_id_predicted
if (! seg_id_predicted)

表 22 (续)

block(j) {	描述符
segment_id	ae(v)
}	
else	
segment_id	ae(v)
}	
}	
if(segmentation_enable && FeatureEnabled[segment_id][SEG_LVL_SKIP])	
skip_flag = 1	
else	
skip_flag	ae(v)
if(! segmentation_enable ! FeatureEnabled[segment_id][SEG_LVL_REF_FRAME])	
inter_block	ae(v)
if((! skip_flag ! inter_block) && tx_mode == TX_MODE_SELECT && MiSize >= BLOCK_8×8)	
tx_size	ae(v)
if(inter_block){	
if(! ref_segfeature_active){	
if(frame_reference_mode == REFERENCE_MODE_SELECT)	
block_reference_mode	ae(v)
if(ref_per_frame>1)	
ref_frame	ae(v)
}	
if(! skip_segfeature_active){	
if (bsize >= BLOCK_8×8){	
mv_mode	ae(v)
if (mv_mode == NEWMV){	
for (ref = 0; ref < 1 + is_compound; ++ref){	
mv_joint	ae(v)
if(mv_joint == MV_JOINT_HZVNZ mv_joint == MV_JOINT_HNZVNZ){	
mvd_sign_0	ae(v)
mvd_value_0	ae(v)
}	
if (mv_joint == MV_JOINT_HNZVZ mv_joint == MV_JOINT_HNZNZ){	
mvd_sign_1	ae(v)

表 22 (续)

描述符
block(j) {
mvd_value_1
}
}
}
}
if (bsize<BLOCK_8×8){
for(i=0;i<2;i++){
for(j=0;j<2;j++){
mv_mode
if (mv_mode == NEWMV){
for (ref = 0; ref < 1 + is_compound; ++ref){
mv_joint
if(mv_joint == MV_JOINT_HZVNZ mv_joint == MV_JOINT_HNZVNZ){
mvd_sign_0
mvd_value_0
}
if (mv_joint == MV_JOINT_HNZVZ mv_joint == MV_JOINT_HNZVNZ){
mvd_sign_1
mvd_value_1
}
}
}
}
if(interp_filter_switchable)
interp_filter_mode
}
else{// Intra block
switch (bsize) {
case BLOCK_4×4:{
for(i=0;i<4;i++)
read_block_intra_luma_mode (i)

表 22 (续)

block(j) {	描述符
break	
}	
case BLOCK_4×8:{	
read_block_intra_luma_mode (0)	
read_block_intra_luma_mode (1)	
break	
}	
case BLOCK_8×4:{	
read_block_intra_luma_mode (0)	
read_block_intra_luma_mode (1)	
break	
}	
default:	
read_block_intra_luma_mode ()	
}	
read_block_intra_chroma_mode ()	
}	
}	
if (ctu_dqp_enable && ! iscoded) {	
if(! skip_flag){	
dqp_abs	ae(v)
if(dqp_abs)	
dqp_sign	ae(v)
iscoded=1	
}	
}	
for (plane=0;plane<3;++plane){	
if (! skip_flag){	
i=0	
while (i < max_eob) {	
coeff_value[i]	ae(v)
if(coeff_value[i]==EOB)	
break	
else{	

表 22 (续)

block(j) {	描述符
while(coeff_value[i] == 0){	
i++	
coeff_value[i]	ae(v)
}	
coeff_sign[i]	ae(v)
i++	
}	
}	
}	
}	
}	
}	

read_block_intra_luma_mode()语法定义见表 23。

表 23 read_block_intra_luma_mode 语法定义

read_block_intra_luma_mode(){	
prev_intra_luma_pred_flag	ae(v)
if(prev_intra_luma_pred_flag){	
mpm_idx0	ae(v)
if(mpm_idx0)	
mpm_idx1	ae(v)
}	
else	
rem_pred_intra_mode	ae(v)
}	

read_block_intra_chroma_mode()语法定义见表 24。

表 24 read_block_intra_chroma_mode 语法定义

read_block_intra_chroma_mode(){	
uv_fllow_y_flag	ae(v)
if(uv_fllow_y_flag)	
chroma_intra_mode	ae(v)
}	

5.2.3.5 监控扩展数据单元语法

5.2.3.5.1 监控扩展数据单元语法通则

监控扩展数据单元语法见表 25。

表 25 监控扩展数据单元语法表

surveillance_extension_rbsp() {	描述符
while(next_bits(8) != 0x80) {	
if(next_bits(8) == 0x04)	
time_extension()	
else if(next_bits(8) == 0x10)	
gis_extension()	
else if(next_bits(8) == 0x11)	
analysis_extension()	
else if(next_bits(8) == 0x12)	
osd_extension()	
else	
reserved_extension()	
}	
surveillance_extension_stop_byte	f(8)
}	

5.2.3.5.2 绝对时间信息扩展语法

绝对时间信息扩展语法见表 26。

表 26 绝对时间信息扩展语法表

time_extension() {	描述符
extension_id	u(8)
extension_length	u(8)
hour_bits	u(5)
minute_bits	u(6)
second_bits	u(6)
second_fraction_bits	u(14)
ref_date_flag	u(1)
if(ref_date_flag) {	
year_minus2000_bits	u(7)
month_bits	u(4)
day_bits	u(5)
}	
}	

5.2.3.5.3 智能分析信息扩展语法

智能分析信息扩展语法见表 27。

表 27 智能分析信息扩展语法表

analysis_extension() {	描述符
extension_id	u(8)
extension_length	u(16)
camera_id	f(160)
analysis_num	u(6)
for(i=0;i<analysis_num;i++) {	
analysis_id[i]	u(8)
description_type[i]	u(2)
data_length[i]	u(16)
for(j=0;j<data_length;j++)	
analysis_data[i][j]	u(8)
}	
while(byte_align() == FALSE)	
reserved_bit	u(1)
}	

5.2.3.5.4 OSD 信息扩展语法

OSD 信息扩展语法见表 28。

表 28 OSD 信息扩展语法表

osd_extension() {	描述符
extension_id	u(8)
extension_length	u(8)
sub_type	u(8)
code_type	u(8)
align_type	u(8)
char_size	u(8)
char_type	u(8)
top_low8	u(8)
top_high8	u(8)
left_low8	u(8)
left_high8	u(8)

表 28 (续)

osd_extension() {	描述符
len	u(8)
res	u(24)
osd_data	u(n)
}	

5.2.3.5.5 地理信息扩展语法

地理信息扩展语法见表 29。

表 29 地理信息扩展语法表

gis_extension() {	描述符
extension_id	u(8)
extension_length	u(8)
longitude_type	u(1)
longitude_degree	u(8)
longitude_fraction_bits	u(20)
latitude_type	u(1)
latitude_degree	u(8)
latitude_fraction_bits	u(20)
height	i(15)
speed	u(8)
yaw_degree	u(9)
reserved	u(6)
}	

5.2.4 语义

5.2.4.1 概述

本条规定了与语法结构和语义结构中的语义元素相关的语义。当一个语义元素的语义用一个或一组表格表示时,表格中未指定的任何值都不应出现在比特流中,除非在本标准中另外规定。

5.2.4.2 NAL 单元语义

NumBytesInNALUnit 指示 NAL 单元的长度,单位为字节。一个 NAL 单元由单元头部分和单元负载部分组成,其中单元负载部分包含一个 RBSP 语法结构及可能存在的认证数据负载,同时还可能包含一些 emulation_prevention_three_byte。在 NAL 单元解码时需要用到 NumBytesInNALUnit。为了能够推导出 NumBytesInNALUnit,需要对 NAL 单元的边界进行划分。附录 B 规定了一种用于字节流格式的划分方法。其他划分方法可能会在本标准之外给出。

forbidden_zero_bit 指示视频流支持的 SVAC 标准的版本。forbidden_zero_bit 应等于 1。

forbidden_zero_bit 等于 0 表示该视频流支持 GB/T 25724—2010 标准。

nal_ref_idc 不等于 0 时,表示 NAL 单元的内容包含一个序列参数集,或一个图像参数集,或一个安全参数集,或一个参考图像的编码片。当一个编码图像的一个编码片 NAL 单元的 nal_ref_idc 等于 0 时,该编码图像的所有编码片 NAL 单元的 nal_ref_idc 都应等于 0。

nal_unit_type 指示 NAL 单元中的 RBSP 数据结构的类型,见表 30。VCL NAL 单元是指那些 nal_unit_type 值等于 1,2,3 或者 4 的 NAL 单元。所有其他的 NAL 单元都称为非 VCL NAL 单元。

注 1: VCL 的规定是为了有效的表示视频数据的内容。NAL 的规定则是为了数据的格式化,并提供头信息以便于存储或在多种通信信道上传输。每个 NAL 单元都包含整数字节。NAL 单元规定了一种既适用于面向分组系统又适用于比特流系统的通用格式。

在不影响 nal_unit_type 不等于 5 的 NAL 单元的解码过程和不影响本标准一致性的前提下,nal_unit_type 等于 5 的 NAL 单元可以被解码器丢弃。

注 2: 本标准不规定 nal_unit_type 为保留值 NAL 单元的解码过程。解码器可以忽略(从比特流中去除并丢弃)所有 nal_unit_type 为保留值的 NAL 单元的内容。

当一个编码片 NAL 单元的 nal_unit_type 值等于 2 时,编码同一图像的其他所有编码片 NAL 单元的 nal_unit_type 值都应相同,与之对应的 SVC 增强层编码图像的所有编码片 NAL 单元的 nal_unit_type 值都应等于 4。这样的图像称作 IDR 图像。

NAL 单元类型见表 30。

表 30 NAL 单元类型表

nal_unit_type	NAL 单元中 RBSP 语法结构的内容
0	保留
1	非 IDR 图像的编码片 tile_data_rbsp()
2	IDR 图像的编码片 tile_data_rbsp()
3	非 IDR 图像的 SVC 增强层编码片 tile_data_rbsp()
4	IDR 图像的 SVC 增强层编码片 tile_data_rbsp()
5	监控扩展数据单元 surveillance_extension_rbsp()
6	补充增强信息 sei_rbsp()
7	序列参数集 seq_parameter_set_rbsp()
8	图像参数集 pic_parameter_set_rbsp()
9	安全参数集 sec_parameter_set_rbsp()

表 30 (续)

nal_unit_type	NAL 单元中 RBSP 语法结构的内容
10	认证数据 authentication_data_rbsp()
11	流结尾 end_of_stream_rbsp()
12	保留
13	本标准第 6 章使用
14	保留
15	SVC 增强层图像参数集 pic_parameter_set_rbsp()

encryption_idc 指示 NAL 单元是否加密。encryption_idc 等于 0 表示该 NAL 单元中的 RBSP 没有加密, encryption_idc 等于 1 表示该 NAL 单元中的 RBSP 以安全参数集中指定的加密方法加密, 且 RBSP 的最后一个字节不加密。

authentication_idc 指示 NAL 单元是否认证。authentication_idc 等于 0 表示该 NAL 单元未经认证, authentication_idc 等于 1 表示该 NAL 单元以安全参数集中指定的认证方法认证。

当 authentication_idc 等于 1 时, 编码比特流中必须携带绝对时间扩展信息, 用于标识认证时间。

payload_byte[i] 为一个 NAL 单元负载的第 i 个字节, 等于 rbsp_byte[j]。一个 NAL 单元负载定义为一个字节的有序序列, 包括一个 RBSP(如果 encryption_idc 等于 1, 则为 RBSP 加密后生成的字节序列)。

rbsp_byte[j] 为一个 RBSP 的第 j 个字节。如果 encryption_idc 等于 1, rbsp_byte[j] 为 RBSP 加密后的字节序列的第 j 个字节, RBSP 需要经过解密过程得到, 解密过程不在本标准中规定。

一个 RBSP 定义为一个字节的有序序列, 包含一个 SODB, 如下所述:

- a) 如果 SODB 为空(长度为 0 比特), RBSP 也为空;
- b) 否则 RBSP 包括如下 SODB:

——RBSP 的第一个字节包括(最高位在前)8 比特的 SODB; RBSP 的下一个字节应包括接下来的 8 比特的 SODB, 依此类推, 直到剩下的 SODB 少于 8 比特;

——rbsp_trailing_bits() 用于 SODB 之后: 最后 RBSP 字节中前面的(从最高位算起)比特包括 SODB 的剩下的比特(如果有的话); 下一比特为单个 rbsp_stop_one_bit, 其值为 1, 并且当 rbsp_stop_one_bit 不是一个字节对齐的字节的最后一个比特时, 后面应存在一个或多个 rbsp_alignment_zero_bit 以形成一个字节对齐。

具有这些 RBSP 属性的语法结构在语法表中用“_rbsp”后缀表示。这些结构在 NAL 单元中作为 rbsp_byte[j] 数据字节的内容携带。NAL 单元到 RBSP 语法结构的关联见表 30。

注 3: 当 RBSP 的边界已知时, 解码器能够通过将 RBSP 字节连接成比特串并丢弃最后(最右边的)一个等于 1 的比特 rbsp_stop_one_bit 以及随后的任何等于 0 的比特, 从 RBSP 中解析出 SODB。解码过程所必须的数据包含在 RBSP 的 SODB 部分。

注 4: 监控扩展数据单元也满足 RBSP 的语法结构, 其中的 surveillance_extension_stop_byte 相当于 rbsp_trailing_bits()。

emulation_prevention_three_byte 等于 0x03。当一个 emulation_prevention_three_byte 出现在 NAL 单元中时, 应被解码过程丢弃。NAL 单元的最后一个字节不能等于 0x00。在 NAL 单元中, 下面

的三字节序列不应在任何字节对齐的位置出现：

```
0x000000
0x000001
0x000002
```

在一个 NAL 单元中,除了下列序列,任何以 0x000003 开头的四字节的序列都不能出现在任何字节对齐的位置：

```
0x00000300
0x00000301
0x00000302
0x00000303
```

注 5: 当 nal_unit_type 等于 0 时,在设计编码器时要避免上面列出的三字节和四字节形式出现在 NAL 单元语法结构的开头,以免 emulation_prevention_three_byte 语法元素成为 NAL 单元的第三字节。

5.2.4.3 NAL 单元的封装及约束

5.2.4.3.1 将 RBSP 封装为 NAL 单元

使用 emulation_prevention_three_byte 将一个 RBSP 封装到一个 NAL 单元中的目的：

——允许 NAL 单元中出现任意的 SODB,但要防止在 NAL 单元中出现伪起始码；

——通过在 RBSP 的结尾查找其比特 rbsp_stop_one_bit,以便能够识别 NAL 单元中 SODB 的结尾。

编码器通过下列步骤能够从一个 RBSP 中产生一个 NAL 单元：

从 RBSP 中查找字节对齐的下面二进制比特位串‘00000000 00000000 000000xx’,其中 xx 代表任意 2 比特位串‘00’‘01’‘10’或‘11’,并且在其中插入一个等于 0x03 的字节,形成‘00000000 00000000 00000011 000000xx’,得到的字节序列加上包含标识 RBSP 数据结构类型的 NAL 单元头部分即形成了整个 NAL 单元。

该过程允许任何 SODB 在一个 NAL 单元中出现,同时可以确保：

——该 NAL 单元中没有字节对齐的伪起始码；

——无论是否字节对齐,在 NAL 单元中没有 8 个值为 0 的比特后跟随起始码的序列。

5.2.4.3.2 NAL 单元的顺序及其与编码图像和视频序列的关系

5.2.4.3.2.1 序列、图像参数集 RBSP 的顺序及生效

一个序列参数集 RBSP 包括的参数可以被一个或多个图像或者包含缓存周期 SEI 消息的 SEI NAL 单元使用。每个序列参数集 RBSP 在被解码器收到的同时生效,并且会导致先前有效的序列参数集 RBSP(如果有的话)失效。至多一个序列参数集 RBSP 在解码过程中的指定时刻为有效的。

当一个序列参数集 RBSP 被一个包含缓存周期 SEI 消息的 SEI NAL 单元使用时,该 SEI NAL 单元应位于序列参数集 RBSP 之后。

图像参数集 RBSP 包括的参数可以被一个编码图像的编码片 NAL 单元使用。每个图像参数集 RBSP 在被解码器收到的同时生效,并且会导致先前有效的图像参数集 RBSP(如果有的话)失效。对 SVC 的某一层图像而言,至多一个图像参数集 RBSP 在解码过程中的指定时刻为有效的。

对序列参数集和图像参数集中的语法元素值与其他语法元素之间的关系所作出的规定,仅针对有效序列参数集和有效图像参数集。

在解码过程中,有效图像参数集和有效序列参数集的参数值应保持有效。

5.2.4.3.2.2 安全参数集 RBSP 的生效

安全参数集 RBSP 包括一些参数,这些参数可以被一个或多个其他类型的 NAL 单元使用。在解码过程的开始,每个安全参数集 RBSP 在被解码器收到的同时生效,并且会导致先前有效的安全参数集 RBSP(如果有的话)失效。当序列参数集的 ldp_mode_flag 等于 1 时,安全参数集应只出现在 IDR 图像前。至多一个安全参数集 RBSP 在解码过程中的指定时刻为有效的。

注:在某些应用中,安全参数集也可以通过其他的可靠机制传送到解码器端。

5.2.4.3.2.3 VCL NAL 单元的顺序及其与编码图像的关系

每个 VCL NAL 单元都是一个编码图像的一部分。

一个编码图像中的 VCL NAL 单元的顺序规定如下:

- 一个图像的编码片顺序应按编码片的第一个 CTU 索引递增的顺序;
- 一个图像的基本层图像参数集应位于一个编码图像的第一个 VCL NAL 单元之前;
- 一个图像的增强层图像参数集与增强层 VCL NAL 单元应在对应的基本层图像的 VCL NAL 单元之后;
- 具有 nal_unit_type 等于 6 的 NAL 单元应位于编码图像的第一个 VCL NAL 单元之前;
- 具有 nal_unit_type 等于 0,12,13 及 14 的 NAL 单元不能位于编码图像的第一个 VCL NAL 单元之前。

nal_unit_type 等于 7,8 及 9 的 NAL 单元可作为编码图像的分界。

5.2.4.4 原始字节序列负载及 RBSP 尾比特语义

5.2.4.4.1 序列参数集 RBSP 语义

序列参数集 RBSP 语义如下:

profile_id 标识比特流支持的档次,取值定义见附录 C。

level_id 标识比特流支持的级别,取值定义见附录 C。

ldp_mode_flag 等于 1 表示比特流为低延时模式,编码图像的编码顺序与显示顺序一致,帧间预测只使用前向预测;等于 0 表示比特流为 RA 模式,编码图像的编码顺序与显示顺序可能不一致,帧间预测可能使用前向预测和双向预测。

frame_width_minus_1 加 1 等于图像宽度的样点数。

frame_height_minus_1 加 1 等于图像高度的样点数。

图像尺寸参数计算如下:

FrameWidth=frame_width_minus_1+1

FrameHeight=frame_height_minus_1+1

MiCols=(FrameWidth+7) >> 3

MiRows=(FrameHeight+7) >> 3

SbCols=(MiCols+(1<<(3+extended_sb_size_flag))-1) >>(3+extended_sb_size_flag)

SbRows=(MiRows+(1<<(3+extended_sb_size_flag))-1) >>(3+extended_sb_size_flag)

当 spatial_svc_flag 等于 1 时,FrameWidth 和 FrameHeight 应为增强层图像宽度和高度的样点数。

chroma_format_idc 等于 1 表示编码图像的色度采样格式为 4:2:0;等于 2 表示编码图像的色度采样格式为 4:2:2。

bit_depth 表示图像样点的比特精度。bit_depth 等于 0 表示图像样点比特精度 BitDepth 为 8;bit_depth 等于 1 表示图像样点比特精度 BitDepth 为 10;bit_depth 等于 2 表示图像样点比特精度 BitDepth

为 12,其余取值保留。

refs_per_frame 表示当前帧参考帧个数,refs_per_frame 的值应为 1~5。

frame_rate 为 3 比特无符号数,用于标识编码图像的帧率,取值见表 31。

表 31 frame_rate 取值对应的帧率

frame_rate	帧率 fps
0	25
1	30
2	50
3	60
4	由 VUI 参数决定
5~7	保留

extended_sb_size_flag 等于 1 表示树形编码单元尺寸为 128×128 ;等于 0 表示树形编码单元尺寸为 64×64 。

tile_enable 等于 0 表示图像不会分割为多个编码片进行解码;等于 1 表示图像有可能会分割为多个编码片进行解码。

wpp_enable 等于 1 表示码流可支持 WPP 解码模式;等于 0 表示码流不支持 WPP 解码模式。当 tile_enable 等于 1 时,wpp_enable 应等于 0。

sao_enable 等于 0 表示不开启样点偏移补偿;等于 1 表示开启样点偏移补偿。

alf_enable 等于 0 表示不开启样点滤波补偿;等于 1 表示开启样点滤波补偿。

roi_flag 等于 1 表示编码视频序列中的图像允许进行分段编码;等于 0 表示编码视频序列中的图像不进行分段编码。

temporal_svc_flag 等于 0 表示不支持时域可分级编码;等于 1 表示支持时域可分级编码。

layer_num_minus_1 在 temporal_svc_flag 等于 1 时有效。**layer_num_minus_1+1** 表示时域分级编码的增强层的层级数目,layer_num_minus_1 值应为 0~3。

spatial_svc_flag 等于 1 表示支持空域可分级编码,编码视频序列中的图像可包含 nal_unit_type 等于 3 和 4 的 NAL 单元;等于 0 表示不支持空域可分级编码,编码视频序列中的图像不包含 nal_unit_type 等于 3 和 4 的 NAL 单元。

svc_ratio 为 3 比特无符号整数,在 spatial_svc_flag 等于 1 时有效,用于指示增强层与基本层的宽度和高度比,如表 32 所示。

表 32 svc_ratio 取值和增强层与基本层的比例对应关系

svc_ratio	增强层与基本层的比例
0	4 : 3
1	2 : 1
2	4 : 1
3	6 : 1
4	8 : 1
5~7	保留

svc_mode 等于 0 表示 SVC 空域增强层不使用层间预测进行编码;等于 1 表示 SVC 空域增强层使用层间预测进行编码。

vui_parameters_present_flag 等于 1 表示后续码流中存在 vui_parameters()语法结构,见附录 D;等于 0 表示后续码流中不存在 vui_parameters()语法结构。

5.2.4.4.2 图像参数集 RBSP 语义

图像参数集 RBSP 语义描述如下:

frame_num 表示当前图像的图像顺序号的低 8 位。

layer_id 表示当前帧所属时域 SVC 的层级号,取值范围应为 0~4。高层级帧可参考低层级帧,低层级帧不能参考高层级帧。基本层的帧的 layer_id 应为 0。如果码流中不存在 layer_id,默认其取值等于 0。

svc_roi_flag 等于 1 表示支持增强层使用 ROI 编码;等于 0 表示不支持增强层使用 ROI 编码。如果码流中不存在 svc_roi_flag,默认其取值等于 0。当 svc_mode 等于 0 时,svc_roi_flag 应等于 0。

svc_top_left 在 svc_roi_flag 等于 1 时有效,表示增强层 ROI 区域的左上角的坐标索引,以 8×8 块为单位。

svc_bottom_right 在 svc_roi_flag 等于 1 时有效,表示增强层 ROI 区域的右下角的坐标索引,以 8×8 块为单位。

frame_type 等于 0 表示当前解码帧为帧内预测帧;等于 1 表示当前解码帧为帧间预测帧。

ctu_dqp_enable 等于 0 表示 CTU 单元内部不能调整量化参数;等于 1 表示 CTU 单元内部允许调整量化参数。如果 ctu_dqp_enable 在码流中不存在,默认其值等于 0。

min_dqp_partition_size 表示允许调整量化参数的预测单元的最小尺寸,见表 33。

表 33 允许调整量化参数的预测单元的最小尺寸与 min_dqp_partition_size 的对应关系

min_dqp_partition_size	extended_sb_size_flag=0	extended_sb_size_flag=1
0	BLOCK_64×64	BLOCK_128×128
1	BLOCK_32×32	BLOCK_64×32
2	BLOCK_16×16	BLOCK_32×16
3	BLOCK_8×8	BLOCK_16×8
4	N/A	BLOCK_8×8

refresh_frame_flags 表示参考帧缓冲区更新标识,refresh_frame_flags 的值应为 0~31。

update_rps_flag 表示是否更新 RPS 队列。update_rps_flag 等于 0 表示当前帧的 RPS 在 RPS 列表中,直接从 RPS 列表获取;等于 1 表示当前帧的 RPS 不在 RPS 列表中,需要从码流中读取。

rps_idx 表示 RPS 的索引,当 update_rps_flag 等于 1 时表示从码流中读取的 RPS 存储在 RPS 队列中的位置,当 update_rps_flag 等于 0 时表示当前帧的 RPS 在 RPS 列表中的索引。rps_idx 的值应为 0~63。

opt_minus_flag 标识参考帧 OPTIONAL_REF 所对应的 delta_poc[2]的符号。opt_minus_flag 等于 1 表示 delta_poc[2]为负;等于 0 表示 delta_poc[2]的符号为正。当 ldp_mode_flag 等于 1 时,opt_minus_flag 应等于 0。

delta_poc[i] 表示对应参考帧与当前帧的图像顺序号的差值,delta_poc[i]的值应为 0~63。

refresh_pictures_num 表示需要刷新的参考帧数目。

allow_high_precision_mv 等于 0 表示亮度运动补偿为四分之一样点精度;等于 1 表示亮度运动补

偿为八分之一样点精度。

interp_filter_switchale 等于 0 表示整帧图像使用固定的运动补偿插值滤波器;等于 1 表示在 CTU 内部确定使用的运动补偿插值滤波器。

interp_filter 表示使用的插值滤波器索引。

filter_level 表示环路滤波级别。

sharpness_level 表示环路滤波平滑度级别。

lf_delta_enable 等于 0 表示关闭环路滤波参数更新;等于 1 表示开启环路滤波参数更新。

lf_delta_update 表示环路滤波参数差值更新。

lf_ref_delta_enable[i] 表示参考帧相关环路滤波参数差值更新使能,等于 0 表示关闭参考帧相关环路滤波参数差值更新,等于 1 表示开启参考帧相关环路滤波参数差值更新。

lf_ref_deltas[i] 表示参考帧相关环路滤波参数差值。

lf_ref_deltas_sign[i] 表示参考帧相关环路滤波参数差值符号。

$$\text{ref_deltas}[i] = \text{lf_ref_deltas}[i] \times \text{lf_ref_deltas_sign}[i]$$

lf_mode_delta_enable[i] 表示模式相关环路滤波参数差值更新使能,等于 0 表示关闭模式相关环路滤波参数差值更新,等于 1 表示开启模式相关环路滤波参数差值更新。

lf_mode_deltas[i] 表示模式相关环路滤波参数差值。

lf_mode_deltas_sign[i] 表示模式相关环路滤波参数差值符号。

$$\text{mode_deltas}[i] = \text{lf_mode_deltas}[i] \times \text{lf_mode_deltas_sign}[i]$$

picture_sao_enable[i] 表示当前图像的亮度与色度分量是否开启样点自适应补偿。**picture_sao_enable[i]** 等于 0 表示不开启,等于 1 表示开启。其中 i 等于 0 表示亮度分量;i 等于 1 或 2 表示色度分量。

picture_alf_enable[i] 图像样点滤波补偿允许标志。表示当前图像的亮度与色度分量是否开启样点滤波补偿。**picture_alf_enable[i]** 等于 0 则表示当前图像的第 i 个分量不应使用样点滤波补偿;等于 1 则表示当前图像的第 i 个分量使用样点滤波补偿。其中 i 等于 0 表示亮度分量;i 等于 1 或 2 表示色度分量。

alf_filter_num_minus1 的值加 1 表示当前图像亮度分量样点滤波补偿滤波器的个数。

alf_filter_num_minus1 的值应为 0~15。

alf_region_distance[i] 表示亮度分量第 i 个样点滤波补偿滤波区域基本单元起始标号与第 i-1 个样点滤波补偿滤波区域基本单元起始标号间的差值。**alf_region_distance[i]** 的取值范围应为 1~15。

如果比特流中不存在 **alf_region_distance[i]**,当 i 等于 0 时,则 **alf_region_distance[i]** 的值为 0,当 i 不等于 0 且 **alf_filter_num_minus1** 的值为 15 时,则 **alf_region_distance[i]** 的值为 1。比特流应满足 **alf_region_distance[i]**(i=0~**alf_filter_num_minus1**)之和应小于等于 15。

alf_coeff_luma[i][j] 表示亮度分量第 i 个样点滤波补偿滤波器的第 j 个系数。从比特流中解码得到的 **alf_coeff_luma[i][j]**(j=0~8)的取值范围应为 -64~63,**alf_coeff_luma[i][9]** 的取值范围应为 -1 088~1 071。

alf_coeff_chroma[0][j] 表示 Cb 分量第 j 个样点滤波补偿滤波器的系数。**alf_coeff_chroma[1][j]** 表示 Cr 分量第 j 个样点滤波补偿滤波器的系数。**AlfCoeffChroma[0][j]** 的值等于 **alf_coeff_chroma[0][j]** 的值,**AlfCoeffChroma[1][j]** 的值等于 **alf_coeff_chroma[1][j]** 的值。**AlfCoeffChroma[i][j]**(i=0~1,j=0~7)的取值范围应为 -64~63,**AlfCoeffChroma[i][9]**(i=0~1)的取值范围应为 -1 088~1 071。

base_qindex 表示量化参数索引。**base_qindex** 的值应为 1~255。

y_dc_delta_q_update_flag 表示亮度直流系数量化参数差值更新标志。**y_dc_delta_q_update_flag** 等于 0 表示关闭亮度直流系数量化参数差值更新;等于 1 表示开启亮度直流系数量化参数差值更新。

y_dc_delta_q 表示亮度直流系数量化参数差值。

y_dc_delta_q_sign 表示亮度直流系数量化参数差值符号。

uv_dc_delta_q_update_flag 表示色度直流系数量化参数差值更新标志。uv_dc_delta_q_update_flag 等于 0 表示关闭色度直流系数量化参数差值更新；等于 1 表示开启色度直流系数量化参数差值更新。

uv_dc_delta_q 表示色度直流系数量化参数差值。

uv_dc_delta_q_sign 表示色度直流系数量化参数差值符号。

uv_ac_delta_q_update_flag 表示色度交流系数量化参数差值更新标志。uv_ac_delta_q_update_flag 等于 0 表示关闭色度交流系数量化参数差值更新；等于 1 表示开启色度交流系数量化参数差值更新。

uv_ac_delta_q 表示色度交流系数量化参数差值。

uv_ac_delta_q_sign 表示色度交流系数量化参数差值符号。

segmentation_enable 表示图像分段使能。segmentation_enable 等于 0 表示关闭图像分段；等于 1 表示开启图像分段。如果码流中不存在 segmentation_enable，默认其值等于 0。

segmentation_update_map 表示更新图像分段图。

seg_tree_flag[i] 等于 1 表示图像分段参数概率模型将更新；等于 0 表示图像分段参数将不更新。

seg_tree_probs[i] 表示图像分段参数概率模型更新后的值。如果 seg_tree_probs[i] 在码流中不存在，默认其值等于 255。

seg_temporal_update 表示时域图像分段更新。

seg_pred_flag[i] 等于 1 表示图像分段预测概率模型将更新；等于 0 表示图像分段预测概率模型将不更新。

seg_pred_probs[i] 表示图像分段预测概率模型更新后的值。如果 seg_pred_probs[i] 在码流中不存在，默认其值等于 255。

seg_update_data 表示图像分段更新数据。

seg_abs_delta 表示图像分段数据差值。

feature_enable[i][j] 表示图像分段特征使能。

seg_feature_data[i][j] 表示图像分段特征数据。

seg_feature_data_sign[i][j] 表示图像分段特征数据符号。如果 seg_feature_data_sign 在码流中不存在，默认其值等于 0。

图像分段特征数据 features_data[8][4] 的计算方法如下：

```
is_segfeature_signed[4]={ 1, 1, 0, 0 };
for(i=0;i<8;i++){
    for(j=0;j<4;j++){
        if(feature_enable[i][j]){
            sign=1;
            if(is_segfeature_signed(j) && seg_feature_data_sign[i][j])
                sign=-1
        }
        features_data[i][j]=sign×seg_feature_data[i][j]
    }
}
```

increment_tile_cols_log2 表示是否增加 Tile 的列数，用于计算 tile_cols_log2。

tile_rows_log2 为 Tile 的行数标识，用于计算 TileRows。

log2_tile_rows_flag 表示 Tile 行标志。

tile_rows_delta 用于计算 tile_rows_log2。

图像划分的 Tile 的相关参数计算如下：

```

minLog2TileCols=0
while ((SbCols >> minLog2TileCols) >64)
    minLog2TileCols++
maxLog2TileCols=0
while ((SbCols >> maxLog2TileCols) >=8)
    maxLog2TileCols++
TileCols=1 << tile_cols_log2
TileRows=1 << tile_rows_log2

```

对第 c 个 Tile 列($c=0..$ TileCols -1)，其水平方向起始样点位置计算如下：

`tile_col_start[c]=((c×SbCols) >> tile_cols_log2)<<(6+extended_sb_size_flag)`

对第 r 个 Tile 行($r=0..$ TileRows -1)，其垂直方向起始样点位置计算如下：

`tile_row_start[r]=((r×SbRows) >> tile_rows_log2)<<(6+extended_sb_size_flag)`

当 `tile_enable` 等于 0 时，`TileCols` 和 `TileRows` 应等于 1。

`reserved_bit` 保留位，其值应等于 0。

tx_mode 表示变换模式。`tx_mode` 等于 0 表示变换模式为 TX_4×4，只支持 4×4 块变换，最大变换尺寸 MAX_TX_SIZE 为 TX_4×4；`tx_mode` 等于 1 表示变换模式为 ALLOW_TX_8×8，最大变换尺寸 MAX_TX_SIZE 为 TX_8×8；`tx_mode` 等于 2 表示变换模式为 ALLOW_TX_16×16，最大变换尺寸 MAX_TX_SIZE 为 TX_16×16；`tx_mode` 等于 3 表示变换模式为 ALLOW_TX_32×32，最大变换尺寸 MAX_TX_SIZE 为 TX_32×32。

tx_mode_delta 表示变换模式差值。如果 `tx_mode_delta` 在码流中不存在，默认其值等于 0。如果 `tx_mode_delta` 等于 1，则 `tx_mode` 等于 4，变换模式为 TX_MODE_SELECT。

`diff_update_prob_8×8` 表示 8×8 变换概率模型更新差值。

`diff_update_prob_16×16` 表示 16×16 变换概率模型更新差值。

`diff_update_prob_32×32` 表示 32×32 变换概率模型更新差值。

`coef_update_prob_flag` 表示残差系数概率模型更新标志。

`diff_update_prob_coef` 表示残差系数概率模型更新差值。

`diff_update_prob_skip` 表示跳过模式概率模型更新差值。

`diff_update_prob_alf` 表示自适应环路滤波使能概率模型更新差值。

`diff_update_prob_newmv` 表示 NEWMV 概率模型更新差值。

`diff_update_prob_zeromv` 表示 ZEROMV 概率模型更新差值。

`diff_update_prob_refmv` 表示 REFMV 概率模型更新标志差值。

`diff_switchable_interp_probs` 表示帧间运动补偿插值滤波器概率模型差值。

`diff_intrainter_update_prob` 表示帧内帧间概率模型差值。

`not_single_ref` 等于 0 表示本帧使用前向单参考帧，`frame_reference_mode` 取值为 SINGLE_REFERENCE；`not_single_ref` 等于 1 表示本帧可能使用双向参考模式。当 `refs_per_frame` 小于 3 时，`not_single_ref` 应等于 0，`frame_reference_mode` 取值应为 SINGLE_REFERENCE。

`not_compound_ref` 在 `not_single_ref` 等于 1 时有效。`not_compound_ref` 等于 0 表示本帧使用双向参考模式，`frame_reference_mode` 取值为 COMPOUND_REFERENCE，`not_compound_ref` 等于 1 表示本帧的参考帧模式由块级的 `block_reference_mode` 决定，`frame_reference_mode` 取值为 REFERENCE_MODE_SELECT。

`diff_inter_update_prob` 表示帧间编码概率模型更新差值。

`diff_single_ref_update_prob` 表示单向参考帧概率模型更新差值。

diff_comp_ref_update_prob 表示双向参考帧概率模型更新差值。
diff_mode_update_prob 表示模式概率模型更新差值。
diff_partition_update_prob 表示块划分概率模型更新差值。
diff_delta_q_update_prob 表示量化差值概率模型更新差值。
mv_joint_probs 表示运动矢量分量编码类型的概率模型。
mv_sign_prob 表示运动矢量符号的概率模型。
mv_class_probs 表示 mv_class 的概率模型。
mv_class0_bit_prob 表示 mv_class0_bit 的概率模型。
mv_bits_prob 表示 mv_bits 的概率模型。
mv_class0_fr_probs 表示 mv_class0_fr 的概率模型。
mv_fr_probs 表示 mv_fr 的概率模型。
mv_class0_hp_prob 表示 mv_class0_hp 的概率模型。
mv_hp_prob 表示 mv_hp 的概率模型。
上述概率模型的更新过程见 5.4.2.2.2。

5.2.4.4.3 安全参数集 RBSP 语义

encryption_flag 等于 1 表示支持对图像编码片,或序列参数集,或图像参数集,或扩展数据单元进行加密,即 NAL 单元中的 RBSP 可能经过加密。**encryption_flag** 等于 0 表示不支持对 NAL 单元中的 RBSP 进行加密。

authentication_flag 等于 1 表示支持对整帧图像数据内容进行认证,进行认证的 NAL 单元包括编码片,以及在该帧传输的序列参数集、图像参数集、安全参数集以及扩展数据单元。当支持对上述数据内容进行认证时,编码比特流中必须携带绝对时间扩展信息,且携带在编码比特流中的认证数据应经过 Base64 编码。认证数据通过 nal_unit_type 等于 10 的 NAL 单元传输。如果图像中存在 authentication_idc 等于 1 的编码片、序列参数集、图像参数集、安全参数集、扩展数据单元等,对一个图像中 authentication_idc 为 1 的 NAL 单元按解码顺序排列后进行认证产生该图像的摘要数据。**authentication_flag** 等于 0 表示不支持对图像进行认证,编码比特流中不应包含 nal_unit_type 等于 10 的 NAL 单元。

如果 spatial_svc_flag 等于 1,对同一帧图像的基本层图像和增强层图像分别进行认证,增强层的图像参数集与编码片按增强层进行认证,序列参数集、安全参数集以及扩展数据单元与基本层的图像参数集及编码片一起按基本层进行认证。

encryption_type 指示加密所采用的算法,具体对应关系见表 34。

表 34 **encryption_type** 与具体加密算法的对应关系

encryption_type	加密算法
0	SM1
1	SM4
2~15	保留

vek_flag 等于 1 表示携带 vek;等于 0 表示不携带 vek。

iv_flag 等于 1 表示携带 iv;等于 0 表示不携带 iv。

camera_id 为 20 个字节的字符串,表示图像来源的摄像机 ID。

vek_encryption_type 指示密钥加密采用的算法,具体对应关系同 **encryption_type**。

ekek_length_minus1 为加密后的密钥长度减 1,以字节为单位。

ekek 为加密后的密钥,用于加密计算,长度为 **ekek_length_minus1** 加 1 字节。

vkek_version_length_minus1 为加密密钥版本号长度减 1,以字节为单位。

vkek_version 为加密密钥版本号,长度为 **vkek_version_length_minus1** 加 1 字节。

iv_length_minus1 为初始向量长度减 1,以字节为单位。

iv 为初始向量,用于分组加密,长度为 **iv_length_minus1** 加 1 字节。

hash_type 表示进行认证所采用的算法,见表 35。

表 35 **hash_type** 与具体算法的对应关系

hash_type	认证算法	摘要数据长度 byte
0	SM3	32
1~3	保留	保留

hash_discard_p_pictures 等于 1 表示对非 IDR 图像不进行认证;等于 0 表示对非 IDR 图像进行认证。如果 **hash_discard_p_pictures** 不在码流中,默认其值等于 1。不进行认证的图像中每个 NAL 单元的 authentication_idc 均应等于 0。

successive_hash_pictures_minus1 加 1 表示按解码顺序进行数字签名的连续图像个数,且这些连续图像仅限在一个 IDR 图像间隔中。**successive_hash_pictures_minus1** 的取值应为 0~255。

如果 **successive_hash_pictures_minus1** 大于 0,首先对按解码顺序连续的 SuccessiveHashPictures 个图像的摘要数据产生树型摘要数据,再对树顶摘要数据进行数字签名。如图 7 所示,n 个图像的树顶摘要数据是对前 n-1 个图像的树顶摘要数据和第 n 个图像的摘要数据排列后,按 **hash_type** 所示的方法产生的摘要数据。

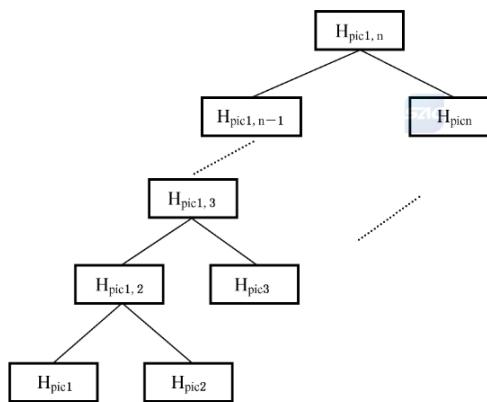


图 7 树型形摘要示意图

$$\text{SuccessiveHashPictures} = \text{successive_hash_pictures_minus1} + 1$$

注: 安全参数集激活后的第一个进行认证的图像应为连续 SuccessiveHashPictures 个图像的第一个。一个 IDR 图像应为连续 SuccessiveHashPictures 个图像的第一个。如果一个 IDR 图像间隔中进行认证的图像不足 SuccessiveHashPictures, 签名数据对应的摘要数据为前一个 IDR 图像间隔包含的所有图像摘要数据。

如果 **successive_hash_pictures_minus1** 等于 0,对每一个进行认证的图像的摘要数据进行数字签名。

signature_type 表示对图像的摘要数据进行数字签名的算法,见表 36。

表 36 signature_type 与具体签名算法的对应关系

signature_type	签名算法
0	SM2
1~3	保留

camera_idc 为 19 个字节的字符串, 用于表示图像来源的摄像机的证书标识。

5.2.4.4.4 补充增强信息 RBSP 语义

补充增强信息(SEI)消息与图像的输出及显示有关, 在 VCL NAL 单元的解码过程中不是必要的。

一个 SEI NAL 单元可以包括多个 SEI 消息。每个 SEI 消息中规定了 SEI 负载类型 PayloadType 和 SEI 负载长度 PayloadSize。

ff_byte 为一个等于 0xFF 的字节。

last_payload_type_byte 表示负载类型的最后一个字节。

last_payload_size_byte 表示负载长度的最后一个字节。

sei_payload 的规定见附录 E。

5.2.4.4.5 编码片 RBSP 语义

tile_idx 表示当前 Tile 在图像中的索引。

substream_len 表示当前 CTU 行的编码码流长度, 以字节为单位。

sao_merge_flag 表示当前 CTU 的 SAO 参数融合标志。sao_merge_flag 等于 0 表示参数不融合; 等于 1 表示参数融合, 此时其 SAO 参数与其左侧相邻或者上方相邻的 CTU 的 SAO 参数相同。

sao_merge_type 等于 1 表示当前 CTU 的 SAO 参数使用左侧相邻的 CTU 的 SAO 参数; 等于 0 表示当前 CTU 的 SAO 参数使用上方相邻的 CTU 的 SAO 参数。

sao_mode[compIdx] 等于 0 表示当前 CTU 中第 compIdx 个分量的 SAO 模式为 SAO_OFF; 等于 1 表示当前 CTU 中第 compIdx 个分量的 SAO 模式由 sao_type[compIdx] 确定。

sao_type[compIdx] 等于 0 表示当前 CTU 中第 compIdx 个分量的 SAO 模式为 SAO_BO; 等于 1 表示当前 CTU 中第 compIdx 个分量的 SAO 模式为 SAO_EO。

sao_start_band[compIdx] 表示当前 CTU 中第 compIdx 个分量在 SAO_BO 模式下的起始补偿区间, 取值应为 0~31。

sao_offset_sign[compIdx][j] 表示 SAO_BO 模式下 sao_offset[compIdx][j] 的符号。sao_offset_sign[compIdx][j] 等于 0 表示对应的 sao_offset[compIdx][j] 的取值为正, 等于 1 表示对应的 sao_offset[compIdx][j] 的取值为负。

sao_offset_abs[compIdx][j] 表示 SAO_BO 模式下的补偿值 sao_offset[compIdx][j] 的绝对值, 取值应为 0~(1<(Min(bit_depth, 10)-5))-1。

sao_edge_type[compIdx] 表示当前 CTU 中第 compIdx 个分量在 SAO_EO 模式下的角度方向。sao_edge_type[compIdx] 等于 0 表示 EO_0°; 等于 1 表示 EO_90°; 等于 2 表示 EO_135°; 等于 3 表示 EO_45°。

sao_edge_offset[compIdx][j] 表示 SAO_EO 模式下的对应的补偿值。

alf_ctu_enable[compIdx] 等于 0 表示当前 CTU 的第 compIdx 个分量不进行样点滤波补偿, alf_ctu_enable[compIdx] 等于 1 表示当前 CTU 的第 compIdx 个分量进行样点滤波补偿。

5.2.4.4.6 认证数据 RBSP 语义

frame_num 表示应包含认证数据的图像,该图像为在认证数据 NAL 单元之前最临近的 frame_num 与认证数据的 frame_num 相同的图像。successive_hash_pictures_minus1 等于 0 时,frame_num 指示认证数据对应的图像;大于 0 时,frame_num 指示连续 SuccessiveHashPictures 个图像的最后一个。

spatial_el_flag 等于 1 表示该认证数据为增强层签名数据,spacial_el_flag 等于 0 表示该认证数据为基本层签名数据。如果 spacial_el_flag 在码流中不存在,默认其值等于 0。

authentication_data_length_minus1 加 1 表示签名数据的长度,以字节为单位,取值应为 0~255。

authentication_data[i] 为一个签名数据的第 i 个字节。

签名数据应经过 Base64 编码,Base64 编码方法见 rfc3548。

5.2.4.4.7 流结尾 RBSP 语义

流结尾 RBSP 表示按解码顺序,在流结尾 RBSP 之后没有任何其他的 NAL 单元。流结尾 RBSP 的内容为空。

5.2.4.4.8 RBSP 尾比特语义

rbsp_stop_one_bit 应等于 1。

rbsp_alignment_zero_bit 应等于 0。

5.2.4.5 CTU 语义

partition 表示当前块划分类型,取值可能为 PARTITION_NONE, PARTITION_HORZ, PARTITION_VERT 和 PARTITION_SPLIT 中的一个。当前块在水平方向上属于图像内的部分小于等于块宽度的一半时,hasCols 等于 0,否则 hasCols 等于 1;当前块在垂直方向上属于图像内的部分小于等于块高度的一半时,hasRows 等于 0,否则 hasRows 等于 1。当 hasCols 等于 1 或 hasRows 等于 1 时,partition 通过码流解析获得;当 hasCols 和 hasRows 均等于 0 时,默认 partition 取值等于 PARTITION_SPLIT。

5.2.4.6 块语义

segment_id 表示当前段的编码,取值范围为 0~7。

seg_id_predicted 表示段的编码是否采用预测编码。

inter_block 表示当前块是否是帧间编码块。

skip_flag 表示当前块是否是跳过编码。

coeff_value 表示块系数的取值。

coeff_sign 表示块系数的符号。

tx_size 表示当前块采用的变换矩阵尺寸。tx_size 等于 0 表示变换矩阵为 TX_4×4;等于 1 表示变换矩阵为 TX_8×8;等于 2 表示变换矩阵为 TX_16×16;等于 3 表示变换矩阵为 TX_32×32。

prev_intra_luma_pred_flag 表示亮度帧内预测模式是否位于帧内预测模式预测列表中,预测列表含 5 个最有可能的预测模式。

mpm_idx0 等于 0 为表示当前亮度预测模式为预测列表中的第一个模式;等于 1 表示当前亮度预测模式不是预测列表中的第一个模式。

mpm_idx1 当 mpm_idx0 为 1 时,mpm_idx1+1 表示当前预测模式位于预测列表中的位置。mpm_idx1 取值应为 0~3。

rem_pred_intra_mode 表示当前亮度预测模式在除预测列表中的 5 个预测模式外, 剩余 32 个预测模式中的索引。rem_pred_intra_mode 取值应为 0~31。

uv_fflow_y_flag 等于 1 表示色度帧内预测模式与其对应位置的亮度帧内预测模式一致, uv_fflow_y_flag 等于 0 表示色度帧内预测模式与其对应位置的亮度帧内预测模式不一致。

chroma_intra_mode 表示色度帧内预测模式索引。

block_reference_mode 表示当前块的参考帧模式, 取值为 SINGLE_REFERENCE 或者 COMPOUND_REFERENCE。如果 block_reference_mode 在码流中不存在, 则 block_reference_mode 取值等于 frame_reference_mode。如果 block_reference_mode 等于 COMPOUND_REFERENCE, 则 is_compound 等于 1, 否则 is_compound 等于 0。

ref_frame 表示当前预测块参考帧索引。当 block_reference_mode 等于 SINGLE_REFERENCE 时, ref_frame 有 5 种可能的取值, 分别为 DYNAMIC_REF, STATIC_REF, OPTIONAL_REF, DYNAMIC_REF_1 和 DYNAMIC_REF_2。当 block_reference_mode 等于 COMPOUND_REFERENCE 时, ref_frame 有 4 种可能的取值, 分别为 DYNAMIC_REF, STATIC_REF, DYNAMIC_REF_1 和 DYNAMIC_REF_2。

mv_mode 表示运动矢量模式。mv_mode 有 4 种可能的取值, 分别为 NEARESTMV, NEARMV, ZEROMV, NEWMV。

mv_joint 表示运动矢量分量编码类型。

mvd_sign_0, mvd_sign_1 表示运动矢量与预测运动矢量差值的符号。

mvd_value_0, mvd_value_1 表示运动矢量与预测运动矢量的差值。当 allow_high_precision_mv 等于 1 且对应的 PMV 的水平分量与垂直分量均小于 8 时, uschp 等于 1, 此时 mvd_value0 和 mvd_value1 为 1/8 样点精度, 否则 mvd_value0 和 mvd_value1 为 1/4 样点精度。

interp_filter_mode 表示帧间运动补偿滤波器模式。interp_filter_mode 取值为 0~3, 分别对应滤波器模式 Regular、Smooth-1、Sharp 和 Smooth-2。

dqp_abs 表示量化系数差值的绝对值。如果 dqp_abs 在码流中不存在, 默认取值为 0。

dqp_sign 表示量化系数差值的符号。0 代表负, 1 代表正。如果 dqp_sign 在码流中不存在, 默认取值为 0。

5.2.4.7 监控扩展数据单元语义

5.2.4.7.1 监控扩展数据单元语义通则

监控扩展数据单元用于传递监控相关信息, 由扩展单元标识(extension_id)区分。有效的 extension_id 不应等于 0x80。

注: extension_id 等于 0x05 的监控扩展数据单元用于第 6 章。

监控扩展数据单元中的信息在图像解码过程中不是必要的。

surveillance_extension_stop_byte 应等于 0x80。

5.2.4.7.2 绝对时间信息扩展语义

extension_id 为 8 位无符号整数。绝对时间信息扩展的标号 extension_id 应等于 4。

extension_length 为 8 位无符号整数, 表示 extension_length 之后的本扩展语法元素长度, 以字节为单位。

hour_bits 表示小时信息。hour_bits 取值应为 0~23。

minute_bits 表示分钟信息。minute_bits 取值应为 0~59。

second_bits 表示秒信息。second_bits 取值应为 0~59。

second_fraction_bits 表示秒的分数信息,以 1/16 384 秒为单位。**second_fraction_bits** 取值应为 0~16 383。

ref_date_flag 等于 1 表示绝对时间信息扩展中包含绝对日期参考信息,**ref_date_flag** 等于 0 表示绝对时间信息扩展中不包含绝对日期参考信息。

year_minus2000_bits 取值应为 0~127,加 2000 表示年份信息 Year。计算如下:

$$\text{Year} = \text{year_minus2000_bits} + 2000$$

month_bits 表示月份信息。**month_bits** 取值应为 1~12。

day_bits 表示日期信息。**date_bits** 取值应为 1~31。

5.2.4.7.3 智能分析信息扩展语义

extension_id 为 8 位无符号整数,智能分析信息扩展的标号 **extension_id** 应等于 0x11。

extension_length 为 16 位无符号整数,表示 **extension_length** 之后的本扩展语法元素长度,以字节为单位。

camera_id 为摄像机标识。

analysis_num 为图像分析结果的数量。

analysis_id[i] 为第 i 项图像分析结果的分析功能标识,其定义见表 37。

表 37 **analysis_id** 的取值定义

analysis_id	定义
0x01	图像分析规则
0x02	运动目标检测
0x03	人员属性分析
0x04	机动车特征分析
0x05	人脸比对
0x06	车牌识别
0x07	绊线检测
0x08	入侵检测
0x09	逆行检测
0x0A	徘徊检测
0x0B	遗留物检测
0x0C	目标移除检测
0x0D	目标数量统计
0xE~0xFF	保留

description_type[i] 为第 i 项图像分析结果的描述形式,其取值定义见表 38。

表 38 **description_type** 的取值定义

description_type	定义
0x00	保留
0x01	根据附录 F 表示
0x02	自定义表示
0x03	保留

data_length[i] 为第 i 项图像分析结果描述内容的字节长度。

analysis_data[i][j] 为第 i 项图像分析结果描述内容的第 j 个字节。

5.2.4.7.4 OSD 信息扩展语义

extension_id 为 8 位无符号整数, OSD 信息扩展的标号 extension_id 应等于 0x12。

extension_length 为 8 位无符号整数, 表示 extension_length 之后的本扩展语法元素长度, 以字节为单位。

sub_type 为 8 位无符号整数, 表示 OSD 扩展信息子类型。可为不同的 OSD 信息分配不同的子类型, sub_type 等于 32 时, 表示时间 OSD 信息; sub_type 等于 33 时, 表示摄像机名称 OSD 信息; sub_type 等于 34 时, 表示地点标注 OSD 信息。

一个 NAL 单元中不应出现两个及以上的 sub_type 取值相同的 OSD 扩展信息。

注: 当多个 sub_type 取值相同的 OSD 扩展信息出现在同一个 NAL 单元时, 最后一个扩展信息有效。

code_type 为 8 位无符号整数, 表示 OSD 字符的编码格式。code_type 的值为 0 时, 表示使用 UTF-8 编码。

align_type 为 8 位无符号整数, 表示 OSD 字符的对齐格式。align_type 的值等于 0 为左对齐; 等于 1 为右对齐。

char_size 为 8 位无符号整数, 表示 OSD 字符字体大小, 以样点为单位表示。

char_type 为 8 位无符号整数, 表示 OSD 字符字符格式。char_type 等于 0 为白底黑边; 等于 1 为黑底白边; 等于 2 为白色; 等于 3 为黑色; 等于 4 为自动反色。

top_low8 和 **top_high8** 组成一个 16 位无符号整数 top, 表示 OSD 字符信息上边界在图像画面中的位置, 以样点为单位表示。top 的取值计算如下:

$$\text{top} = (\text{top_high8} \ll 8) + \text{top_low8}$$

left_low8 和 **top_high8** 组成一个 16 位无符号整数 left, 表示 OSD 字符信息左边界在图像画面中的位置, 以样点为单位表示。left 的取值计算如下:

$$\text{left} = (\text{left_high8} \ll 8) + \text{left_low8}$$

len 为 8 位无符号整数, 表示 osd_data 占用的字节长度, 取值应为 0~243。

res 为 8 位无符号整数, 取值应为 0~255 之间。

osd_data OSD 字符数据。其中, 定义 '\n' 为换行符, '\0' 为结束符。osd_data 的长度为 len 字节。

5.2.4.7.5 地理信息扩展语义

extension_id 为 8 位无符号整数, 地理信息扩展的标号 extension_id 应等于 0x10。

longitude_type 等于 0 表示东经; 等于 1 表示西经。

longitude_degree 为 8 位无符号整数, 表示经度的度数。

longitude_fraction_bits 表示度的分数信息, 以 1/1 048 576 度为单位, 取值应为 0~1 048 575。

latitude_type 等于 0 表示北纬; 等于 1 表示南纬。

latitude_degree 为 8 位无符号整数, 表示纬度的度数。

latitude_fraction_bits 表示度的分数信息, 以 1/1 048 576 度为单位, 取值应为 0~1 048 575。

height 表示高度, 单位为米。

speed 表示速度, 单位为米/秒。

yaw_degree 表示方向角度数, 0 表示方向为正北, 角度数沿顺时针方向递增。

reserved 为保留比特位。

5.3 解码过程

5.3.1 视频解码器

视频解码流程示例见图 8。视频解码器接收编码比特流, 对图像中的树形编码单元, 经熵解码、逆

扫描、反量化及反变换产生一组残差数据 D' , 并根据码流中的信息通过帧内预测或帧间预测得到预测数据 PRED, 预测数据与残差数据通过计算生成重建图像 F' 。重建图像经去块效应滤波, 样点偏移补偿(SAO)、样点滤波补偿(ALF)后产生最终的解码图像。

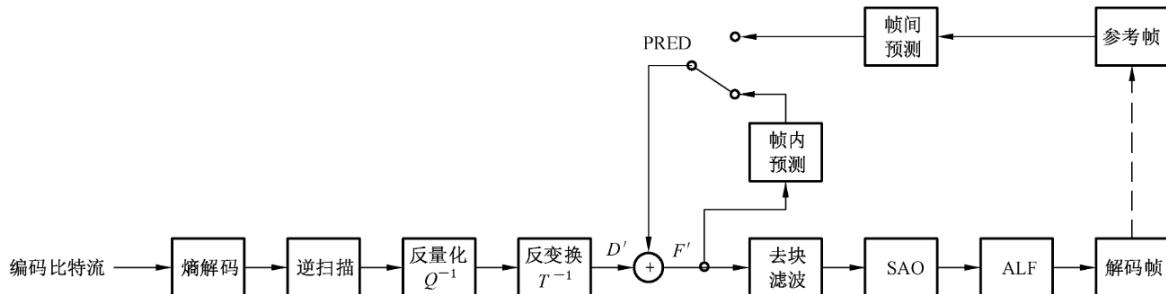


图 8 解码流程示例

5.3.2 NAL 单元解码过程

NAL 单元解码过程的输入为 NAL 单元, 输出为封装在 NAL 单元中的 RBSP 语法结构。该过程为从 NAL 单元中提取出 RBSP 语法结构。如果 encryption_idc 等于 1, 则从 NAL 单元中提取 RBSP 语法结构时需对加密的 RBSP 进行解密处理, 得到未加密的 RBSP。该解密过程不在本标准中规定。

对 NAL 单元中的 RBSP 语法结构按照如下的方式进行解码:

- nal_unit_type 的值为 1,2,3 和 4 时 NAL 单元的解码过程, 见 5.3.3;
- nal_unit_type 的值为 1 和 2 时 NAL 单元的帧内预测过程, 见 5.3.4;
- nal_unit_type 的值为 1 时 NAL 单元中的的帧间预测过程, 见 5.3.5;
- nal_unit_type 的值为 1 和 2 时 NAL 单元中的树形编码单元在去块效应滤波前变换系数的解码过程和图像重建过程, 见 5.3.6;
- nal_unit_type 的值为 1,2,3 和 4 时 NAL 单元的重建图像的去块效应滤波过程, 见 5.3.7;
- nal_unit_type 的值为 1,2,3 和 4 时 NAL 单元的重建图像的样点偏移补偿过程, 见 5.3.8;
- nal_unit_type 的值为 1,2,3 和 4 时 NAL 单元的重建图像的样点滤波补偿过程, 见 5.3.9;
- nal_unit_type 的值为 3 和 4 时 NAL 单元中的树形编码单元在去块效应滤波之前的解码过程, 见 5.3.10;
- nal_unit_type 的值为 7,8 和 9 时 NAL 单元中的 RBSP 分别为序列参数集, 图像参数集及安全参数集。有效的序列参数集、图像参数集及安全参数集用于其他 NAL 单元的解码过程;
- nal_unit_type 的值为 13 的 NAL 单元的解码过程, 见第 6 章;
- nal_unit_type 的值为 0,12,14 和 15 的 NAL 单元的解码过程不在本标准规定。

5.3.3 图像解码过程

5.3.3.1 图像的分类和对应关系

编码视频序列中可能解码出以下三种图像:

- 帧内解码图像(I 图像);
- 即时解码刷新图像(IDR 图像);
- 帧间解码图像。

所有图像为帧图像。

符合本标准的 I 图像同时应为 IDR 图像。IDR 图像解码之后, 解码顺序上所有后续的编码图像均不用根据任何在该 IDR 图像之前解码的图像来进行帧间预测解码。每个编码视频序列的第一幅图像

应为 IDR 图像。可通过 nal_unit_type 来识别一幅图像是否为 IDR 图像。

帧内解码图像和帧间解码图像均可作为参考图像。

编码图像序列以图像编码组(GOP)组成。连续编码的两帧图像的图像顺序号的差值应不大于 16。参考图像支持分层与不分层结构。当 temporal_svc_flag 等 0 时,表示当前编码图像不支持时域分层编码;当 temporal_svc_flag 等 1 时,表示当前编码支持时域分层编码,此时在码流中的 layer_id 表示当前帧的时域 SVC 层级。处于同一层级的图像帧应按照显示顺序进行编码。低层级的帧(layer_id 较小的帧)不能参考高层级的帧(layer_id 较大的帧)。

RA 模式下典型的分层结构如图 9 所示。

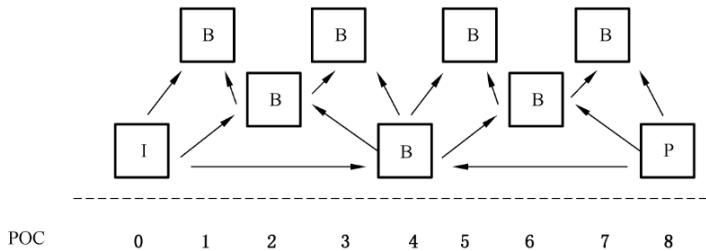


图 9 RA 模式下的时域 SVC 分层结构模型

RA 模式下的解码顺序为 $0 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 6 \rightarrow 1 \rightarrow 3 \rightarrow 5 \rightarrow 7$, 层级结构可设为第 0、8 帧为第 0 层级(layer_id 为 0), 第 4 帧为第 1 层级(layer_id 为 1), 第 2、6 帧为第 2 层级(layer_id 为 2)第 1、3、5、7 帧为第 3 层级(layer_id 为 3)。第 8 帧参考第 0 帧, 第 4 帧参考第 0 和第 8 帧, 第 2 帧参考第 0 和第 4 帧, 第 6 帧参考第 4 和第 8 帧, 第 1、3、5、7 帧属于最高层级可以参考之前已经解码的所有帧。

LD 模式下典型的分层结构如图 10 所示。

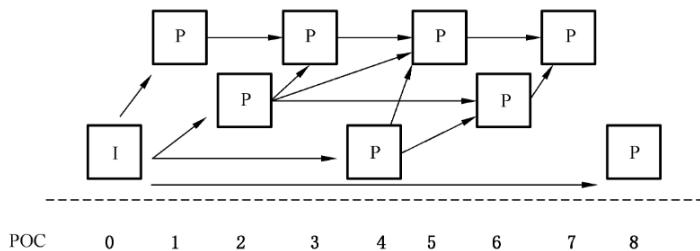


图 10 LD 模式下的时域 SVC 分层结构模型

LD 模式下解码顺序为 $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8$, 层级结构可设 0、8 帧为第 0 层级(layer_id 为 0), 第 4 帧为第 1 层级(layer_id 为 1), 第 2、6 帧为第 2 层级(layer_id 为 2), 第 1、3、5、7 为第 3 层级(layer_id 为 3)。参考帧配置可以为: 第 1 帧参考之前已经解码的所有帧, 这里参考第 0 帧; 第 2 帧参考之前已经解码的所有图像顺序号为偶数的帧, 这里参考第 0 帧; 第 3 帧参考之前已经解码的所有帧, 这里参考第 0、1、2 帧; 第 4 帧参考之前已经解码的所有图像顺序号为 4 的倍数的帧, 这里参考第 0 帧; 第 5 帧可以参考之前已经解码的所有帧, 这里参考第 2、3、4 帧; 第 6 帧参考之前已经解码的所有图像顺序号为 2 的倍数的帧, 这里参考第 0、2、4 帧; 第 7 帧参考之前已经解码的所有帧, 这里参考第 4、5、6 帧; 第 8 帧参考之前已经解码的所有图像顺序号为 8 的倍数的帧, 这里参考第 0 帧。

5.3.3.2 图像顺序号的解码过程

SVC 图像顺序号按照图像的显示顺序依次递增。当前帧的图像顺序号 cur_frame_poc 计算如下:

$$\text{cur_frame_poc} = \text{poc_msb} + \text{poc_lsb}$$

其中, poc_lsb 等于当前图像的 frame_num。poc_msb 在解码开始或者当前帧为 IDR 帧时置为 0,

否则,当前帧的 poc_msb 计算如下:

```

if((last_poc_lsb-poc_lsb)>=128)
{
    poc_msb=poc_msb+256
}
else if((poc_lsb-last_poc_lsb)>=128)
{
    poc_msb=poc_msb-256
    if(poc_msb<0)
        poc_msb=0
}
else
{
    poc_msb=poc_msb
}

```

其中 last_poc_lsb 为编码顺序上前一帧图像的 poc_lsb。

5.3.3.3 编码片的解码过程

编码片解码开始时,根据 FrameWidth、FrameHeight、TileCols、TileRows、tile_idx 可以计算出编码片起始的图像样点位置以及本编码片中包含的 CTU 个数(记作 ctu_num_in_tile)。依次解码编码片中的 CTU,解码 ctu_num_in_tile 个 CTU 后,编码片解码结束。

5.3.3.4 参考图像选择

5.3.3.4.1 参考图像集

参考图像集(RPS)用于表示当前帧对应的各参考帧图像,每个 RPS 中的参考帧数量等于 refs_per_frame。

每帧图像具有唯一的图像顺序标志 poc。当前帧所用的第 i 个参考帧对应的 poc 计算如下:

```

poc_refframe[i]=cur_frame_poc-sign_ref[i]×delta_poc[i]
sign_ref[0]=1   (i=0,1,3,4)
sign_ref[2]=Sign(0-opt_minus_flag) (i=2)

```

i 等于 0~4 分别对应参考帧 DYNAMIC_REF、STATIC_REF、OPTIONAL_REF、DYNAMIC_REF_1 和 DYNAMIC_REF_2,其中只有 OPTIONAL_REF 参考帧有可能是后向参考帧,其他都应为前向参考帧。

当 delta_poc[i](i>0) 等于 delta_poc[0] 时,表示 RPS 中的第 i 个参考帧无效,

不同图像帧可以采用相同的 RPS,因此采用队列来存储 RPS,其中队列最大存储 RPS 个数为 64 个。当 update_rps_flag 等于 1,表示当前帧的 RPS 需要更新,应从码流中读取 delta_poc 计算出 RPS,并存入 RPS 队列,参考索引值为 rps_idx。当 update_rps_flag 等于 0 表示当前帧的 RPS 在 RPS 队列,直接从 RPS 队列中获取,参考索引值为 rps_idx。

5.3.3.4.2 参考帧缓冲区的刷新

参考帧缓冲区内最多可以存在 8 个参考帧。从码流中获取 refresh_frame_flags,refresh_flag[i] 表示 refresh_frame_flags 从低位算起的第 i 个比特,refresh_flag[i] 对应 RPS 中第 i 个参考帧的刷新状

态。如果 refresh_flag [i] 等于 1，则当前帧解码完成后，对应的参考帧将从参考帧缓冲区中移除；如果 refresh_flag [i] 等于 0，则解码完当前帧后，对应的参考帧将保留在参考帧缓冲区中。解码当前帧后，根据从码流获取的 refresh_pictures_num 以及之后的 delta_poc，计算对应参考帧的图像顺序号，然后将对应参考帧从参考帧缓冲区中移除。

5.3.4 帧内预测过程

5.3.4.1 帧内预测模式

帧内预测模式共 37 种，取值为 0~36，如图 11 所示。luma_intra_mode 与 chroma_intra_mode 的取值与帧内预测模式的对应关系见表 39 和表 40。

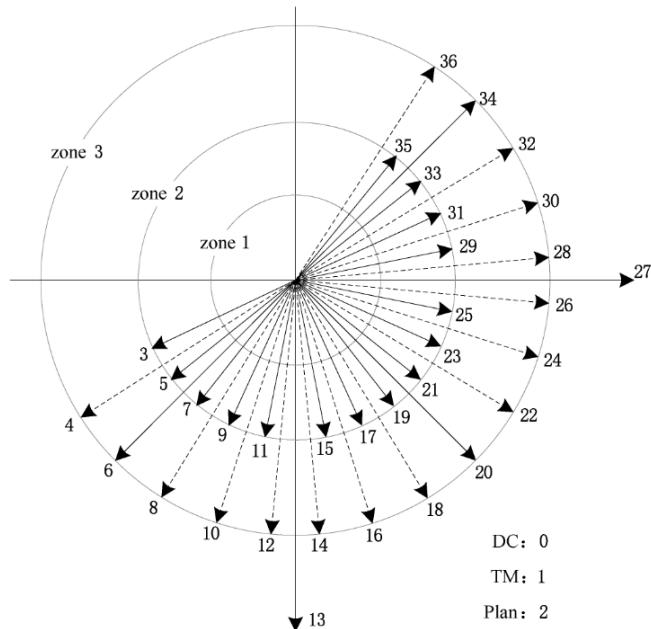


图 11 帧内预测模式

表 39 亮度块帧内预测模式

luma_intra_mode	帧内预测模式
0	DC_PRED
1	TM_PRED
2	PLAN_PRED
3~12	ANG_PRED
13	Vertical_PRED
14~26	ANG_PRED
27	Horizontal_PRED
28~36	ANG_PRED

表 40 色度块帧内预测模式

chroma_intra_mode	帧内预测模式
0	DC_PRED
1	PLAN_PRED
2	Vertical_PRED
3	Horizontal_PRED
4	DM_PRED

表 40 中, DM_PRED 表示当前色度块对应位置亮度块的预测模式, 当对应的亮度块包含多个预测块时, 为最后一个预测块的预测模式。当 DM_PRED 等于 DC_PRED、PLAN_PRED、Vertical_PRED、Horizontal_PRED 其中的一种, 则将表 40 中对应索引的预测模式用 TM_PRED 代替, 而 DM_PRED 本身不变。

5.3.4.2 帧内预测模式的确定

5.3.4.2.1 亮度块的帧内预测模式

亮度块的帧内预测模式计算如下:

- a) 根据当前块的左边块和上边块的预测模式构建预测列表 intra_candlis [i], i=0~4:
 如果左边块存在并且是帧内预测块, 则将左边块的帧内预测模式赋值给 intraPredModeLeft; 否则 intraPredModeLeft 等于 0(DC_PRED)。
 如果上边块存在并且是帧内预测块, 则将上边块的帧内预测模式赋值给 intraPredModeAbove; 否则 intraPredModeAbove 等于 0(DC_PRED)。
 令 PredMode0 等于 min(intraPredModeLeft, intraPredModeAbove), PredMode1 等于 max(intraPredModeLeft, intraPredModeAbove)。

1) 当 PredMode0 等于 PredMode1 时,

——如果 PredMode0 大于 PLAN_PRED, 预测列表的构建按如下方式进行:

```

intra_candlis[0]=PredMode0
intra_candlis[1]=DC_PRED
intra_candlis[2]=TM_PRED
intra_candlis[3]=((PredMode0+31) % 34)+2
intra_candlis[4]=((PredMode0-1) % 34)+2
并且令 mpm_contex_idx=0。

```

——否则, 按如下方式进行:

```

intra_candlis[0]=PredMode0
intra_candlis[1]=PredMode0==DC_PRED ? TM_PRED:DC_PRED
intra_candlis[2]=PredMode0> TM_PRED ? TM_PRED: PLAN_PRED
intra_candlis[3]=V_PRED
intra_candlis[4]=H_PRED
并且令 mpm_contex_idx=1。

```

2) 当 PredMode0 不等于 PredMode1 时,

——如果 PredMode0 大于 PLAN_PRED(即上边块和左边块都为角度预测时):

```
if(PredMode1-PredMode0 <4){
```

```

        intra_candlis[0]=(PredMode0+PredMode1) >> 1
        intra_candlis[1]=intra_candlis[0]-2
        intra_candlis[2]=intra_candlis[0]-1
        intra_candlis[3]=intra_candlis[0]+1
        intra_candlis[4]=PredMode1 == ANG_36 ? H_PRED : intra_candlis[0]+2
        mpm_contex_idx=2
    }
    else if(PredMode1-PredMode0<12){
        intra_candlis[0]=(PredMode0+PredMode1) >> 1
        intra_candlis[1]=PredMode0
        intra_candlis[2]=PredMode0+1
        intra_candlis[3]=PredMode1
        intra_candlis[4]=PredMode1-1
        mpm_contex_idx=3
    }
    else {
        intra_candlis[0]=PredMode0
        intra_candlis[1]=TM_PRED
        intra_candlis[2]=PredMode0+1
        intra_candlis[3]=PredMode1
        intra_candlis[4]=PredMode1-1
        mpm_contex_idx=4
    }

```

——否则,预测列表的构建按如下方式进行:

```

if(PredMode1 > ANG_3) {
    intra_candlis[0]=PredMode0
    intra_candlis[1]=PredMode0==TM_PRED? DC_PRED : TM_PRED
    intra_candlis[2]=PredMode1-1
    intra_candlis[3]=PredMode1
    intra_candlis[4]=PredMode1 == ANG_36 ? H_PRED : ((PredMode1-1) %35)+2
    mpm_contex_idx=5
}
else {
    intra_candlis[0]=DC_PRED
    intra_candlis[1]=TM_PRED
    intra_candlis[2]=PLAN_PRED
    intra_candlis[3]=PredMode1>PLAN_PRED? PredMode1 : V_PRED
    intra_candlis[4]=H_PRED
    mpm_contex_idx=6
}

```

b) 计算 luma_intra_mode 的值:

如果码流中解析的 prev_intra_luma_pred_flag 的值为 1, 则继续解析码流中的 mpm_idx0, 如果 mpm_idx0 为 0, 则 luma_intra_mode 等于 intra_candlis[0], 如果 mpm_idx0 为 1, 则继续解

析码流中的 mpm_idx1,luma_intra_mode 等于 intra_candlis[1+mpm_idx1]。

如果码流中解析的 prev_intra_luma_pred_flag 的值为 0, 则从码流中解析出 rem_pred_intra_mode, 并对其值作如下调整:

遍历 $i=0 \sim 4$, 如果 rem_pred_intra_mode 大于 intra_candlis[i], 则将 rem_pred_intra_mode 加 1, 最后令 luma_intra_mode 等于 rem_pred_intra_mode。

根据 luma_intra_mode 的值, 查表 39 得到亮度块帧内预测模式。

5.3.4.2.2 色度块的帧内预测模式

如果码流中解析的 uv_fflow_y_flag 的值为 1, 则 chroma_intra_mode 等于 4; 否则, 从码流中解析得到 chroma_intra_mode。

根据 chroma_intra_mode 的值, 查表 40 得到色度块帧内预测模式。

5.3.4.3 帧内预测时相邻块参考样点可用性判断

对于帧内预测, 预测块尺寸与变换块尺寸绑定, 因变换只有 $N \times N$ 的变换, 因而预测块尺寸也为 $N \times N$ 。相邻块可用性参见 5.1.3.3。

5.3.4.4 亮度参考样点的获取

对于 $N \times N$ 的亮度块, 当前块上边的参考样点记为 $r[i]$, 左边的参考样点记为 $c[j]$, 其中 $r[0]=c[0]$ 。

用 I 表示当前块所在图像的补偿后(也就是滤波前)样点的亮度样值矩阵。

设当前块左上角样点在图像中的坐标是 (x_0, y_0) , 其参考样点按以下规则获取:

- 初始化 $r[i], c[j]$ 为 $2^{\text{bitdepth}-1}$, $i=0 \sim 2N, j=0 \sim 2N$;
- 如果上边块可用, 则 $r[i]=I[x_0+i-1, y_0-1]$, $i=1 \sim N$, $r[i]$ 可用; 否则 $r[i]$ 不可用;
- 如果有上块可用, 则 $r[i]=I[x_0+i-1, y_0-1]$, $i=N+1 \sim 2N$, $r[i]$ 可用; 否则 $r[i]$ 等于 $r[N]$, $r[i]$ 是否可用取决于 $r[N]$ 是否可用;
- 如果左边块可用, 则 $c[j]=I[x_0-1, y_0+j-1]$, $j=1 \sim N$, $c[j]$ 可用; 否则 $c[j]$ 不可用;
- 如果左下块可用, 则 $c[j]=I[x_0-1, y_0+j-1]$, $j=N+1 \sim 2N$, $c[j]$ 可用; 否则 $c[j]$ 等于 $c[N]$, $c[j]$ 是否可用取决于 $c[N]$ 是否可用;
- 如果坐标为 (x_0-1, y_0-1) 的样点可用, 则 $r[0]=I[x_0-1, y_0-1]$, $r[0]$ 可用, 否则 $r[0]$ 不可用。

5.3.4.5 色度参考样点的获取

色度参考样点与亮度参考样点的获取方法一致, 只是亮度块变为对应的色度块。

5.3.4.6 预测样点的计算

各帧内预测模式下的亮度块和色度块的预测样点矩阵 predMatrix 导出如下:

- Horizontal_PRED
 $\text{predMatrix}[x, y]=c[y+1]$ ($x=0 \sim N-1, y=0 \sim N-1$)
- Vertical_PRED
 $\text{predMatrix}[x, y]=r[x+1]$ ($x=0 \sim N-1, y=0 \sim N-1$)
- TM_PRED
 $\text{predMatrix}[x, y]=\text{clip}(r[x+1]+c[y+1]-r[0])$ ($x=0 \sim N-1, y=0 \sim N-1$)
- DC_PRED



1) 如果 $r[i], c[j]$ ($i=1 \sim N, j=1 \sim N$) 均可用, 则

$$predMatrix[x, y] = (\sum_{i=1}^N r[i] + \sum_{j=1}^N c[j] + N)/2N \text{ 其中 } (x=0 \sim N-1, y=0 \sim N-1)$$

2) 否则, 如果 $r[i]$ 可用 (i 等于 $1 \sim N$), 则 $predMatrix[x, y] = (\sum_{i=1}^N r[i] + (N \gg 1)) \gg \log_2(N)$ 其中 $(x=0 \sim N-1, y=0 \sim N-1)$

3) 否则, 如果 $c[j]$ 可用 (j 等于 $1 \sim N$), 则 $predMatrix[x, y] = (\sum_{i=1}^N c[j] + (N \gg 1)) \gg \log_2(N)$ 其中 $(x=0 \sim N-1, y=0 \sim N-1)$

4) 否则, $predMatrix[x, y] = 2^{\text{BitDepth}-1}$ 其中 $(x=0 \sim N-1, y=0 \sim N-1)$

e) PLAN_PRED

$ib_mult[5] = \{13, 17, 5, 11, 23\}$

$ib_shift[5] = \{7, 10, 11, 15, 19\}$

$ia = (r[N] + c[N]) \ll 4$

$imh = ib_mult[\log_2(N)-2]$

$ish = ib_shift[\log_2(N)-2]$

$imv = ib_mult[\log_2(N)-2]$

$isv = ib_shift[\log_2(N)-2]$

$ih = (\sum_{i=0}^{(N \gg 1)-1} (i+1) \times (r[(N \gg 1)+1+i] - r[(N \gg 1)-1-i]))$

$iv = (\sum_{j=0}^{(N \gg 1)-1} (j+1) \times (c[(N \gg 1)+1+j] - c[(N \gg 1)-1-j]))$

$ib = ((ih \ll 5) \times imh + (1 \ll (ish-1))) \gg ish$

$ic = ((iv \ll 5) \times imv + (1 \ll (isv-1))) \gg isv$

$predMatrix[x, y] = Clip1((ia + (x - (N \gg 1) + 1) \times ib + (y - (N \gg 1) + 1) \times ic + 16)) \gg 5$

$(x=0 \sim N-1, y=0 \sim N-1)$

f) ANG_PRED

初始化 $r[-1] = c[1], r[-2] = c[2], c[-1] = r[1], c[-2] = r[2]$ 。根据 luma_intra_mode 的值查表 41 得到角度预测模式的预测方向 dx、dy 以及 xyaxis、xysign、imx、isx、imy 和 isy。

$dx = dirDx[luma_intra_mode]$

$dy = dirDy[luma_intra_mode]$

$xyaxis = dirXYflag[luma_intra_mode]$

$xysign = dirXYSign[luma_intra_mode]$

$imx = div_dxy[luma_intra_mode][0]$

$isx = div_dxy[luma_intra_mode][1]$

$imy = div_dyx[luma_intra_mode][0]$

$isy = div_dyx[luma_intra_mode][1]$

1) 如果 xysign 小于 0, 则

——如果 xyaxis 等于 0, 则有

$$offset = (((y+1) \times imx \times 32) \gg isx) - ((y+1) \times imx \gg isx)$$

$$iX = x + ((y+1) \times imx \gg isx)$$

$$iY = -1$$

——如果 xyaxis 等于 1, 则有

$$offset = (((x+1) \times imy \times 32) \gg isy) - ((x+1) \times imy \gg isy)$$

- $iX = -1$
 $iY = y + ((x+1) \times imy) \gg isy)$
- 2) 如果 xysign 大于 0, 则
 $offsetx = (((y+1) \times imx \times 32) \gg isx) - ((y+1) \times imx) \gg isx)$
 $iXx = x((y+1) \times imx) \gg isx)$
 $offsety = (((x+1) \times imy \times 32) \gg isy) - ((x+1) \times imy) \gg isy)$
 $iYy = y - ((x+1) \times imy) \gg isy)$
 ——如果 iYy 小于或等于 -1 , $offset = offsetx$, $iX = iXx$, $iY = -1$;
 ——否则, $offset = offsety$, $iX = -1$, $iY = iYy$;
- 3) 如果 iY 等于 -1 :
 ——如果 $dx \times dy$ 小于 0, 则 $iXn = iX + 1$, $iXnP2 = iX + 2$, $iXnN1 = iX - 1$;
 ——如果 $dx \times dy$ 大于 0, 则 $iXn = iX - 1$, $iXnP2 = iX - 2$, $iXnN1 = iX + 1$;
 $predMatrix[x, y] = (r[iXnN1 + 1] \times (32 - offset) + r[iX + 1] \times (64 - offset) + r[iXn + 1] \times (32 + offset) + r[iXnP2 + 1] \times (offset + 64)) \gg 7$, ($x = 0 \sim N - 1$, $y = 0 \sim N - 1$)
- 4) 如果 iX 等于 -1 :
 ——如果 $dx \times dy$ 小于 0, 则 $iYn = iY + 1$, $iYnP2 = iY + 2$, $iYnN1 = iY - 1$;
 ——如果 $dx \times dy$ 大于 0, 则 $iYn = iY - 1$, $iYnP2 = iY - 2$, $iYnN1 = iY + 1$;
 $predMatrix[x, y] = (c[iYnN1 + 1] \times (32 - offset) + c[iY + 1] \times (64 - offset) + c[iYn + 1] \times (32 + offset) + c[iYnP2 + 1] \times (offset + 64)) \gg 7$, ($x = 0 \sim N - 1$, $y = 0 \sim N - 1$)

表 41 角度预测模式对应的方向参数

luma_intra_mode	dirDx	dirDy	dirXYflag	dirXYsign	div_dxy	div_dyx
DC	—	—	—	—	—	—
TM	—	—	—	—	—	—
PLAN	—	—	—	—	—	—
3	11	-4	0	-1	{11,2}	{93,8}
4	2	-1	0	-1	{2,0}	{1,1}
5	11	-8	0	-1	{11,3}	{93,7}
6	1	-1	0	-1	{1,0}	{1,0}
7	13	-16	0	-1	{13,4}	{315,8}
8	5	-8	0	-1	{5,3}	{13,3}
9	7	-16	0	-1	{7,4}	{9,2}
10	5	-16	0	-1	{5,4}	{205,6}
11	3	-16	0	-1	{3,4}	{171,5}
12	3	-32	0	-1	{3,5}	{171,4}
Ver	—	—	—	—	—	—
14	3	32	0	1	{3,5}	{171,4}
15	3	16	0	1	{3,4}	{171,5}
16	5	16	0	1	{5,4}	{205,6}
17	7	16	0	1	{7,4}	{9,2}
18	5	8	0	1	{5,3}	{13,3}

表 41 (续)

luma_intra_mode	dirDx	dirDy	dirXYflag	dirXYsign	div_dxy	div_dyx
19	13	16	0	1	{13,4}	{315,8}
20	1	1	0	1	{1,0}	{1,0}
21	16	13	0	1	{315,8}	{13,4}
22	8	5	0	1	{13,3}	{5,3}
23	16	7	0	1	{9,2}	{7,4}
24	16	5	0	1	{205,6}	{5,4}
25	16	3	0	1	{171,5}	{3,4}
26	32	3	0	1	{171,4}	{3,5}
Hor	—	—	—	—	—	—
28	32	-3	1	-1	{171,4}	{3,5}
29	16	-3	1	-1	{171,5}	{3,4}
30	16	-5	1	-1	{205,6}	{5,4}
31	16	-7	1	-1	{9,2}	{7,4}
32	8	-5	1	-1	{13,3}	{5,3}
33	16	-13	1	-1	{315,8}	{13,4}
34	1	-1	1	-1	{1,0}	{1,0}
35	8	-11	1	-1	{93,7}	{11,3}
36	1	-2	1	-1	{1,1}	{2,0}

5.3.5 帧间预测过程

5.3.5.1 概述

本过程的输出为当前预测单元的帧间预测样点矩阵 predMatrix, 包含一个亮度矩阵 predL, 如果 chroma_format_idc 不等于 0, 还包含相应的色度样点矩阵 predCb 及 predCr, 分别对应色度分量 Cb 和 Cr。

5.3.5.2 运动矢量

5.3.5.2.1 候选亮度运动矢量集导出

候选运动矢量 PMV[0]和 PMV[1]初始设置为{0,0}。

按下述步骤进行搜索, 满足条件的运动矢量加入候选运动矢量集。其中, 第一个满足条件的 MV (mvx, mvy) 为候选运动矢量 PMV[0], 第二个满足条件且不同于 PMV[0] 的运动矢量为 PMV[1]。当 PMV[1] 得到或下述所有步骤执行完成后, 本过程结束。

第一步, 按顺序搜索候选位置(Xc(i), Yc(i))(i=0~7)的 8×8 块, 如果该位置在图像内、且该块使用的某一参考帧与当前块的参考帧相同, 则该块对应参考帧的 MV 加入候选运动矢量集。当所有候选位置均被搜完后, 进入第二步。

其中, 候选块相对于当前块(X0, Y0)的位置根据当前块的 sub_size 查 mv_ref_blocks 得到。

mv_ref_blocks[BLOCK_SIZES][MVREF_NEIGHBOURS]=

```

{
    {{-1, 0}, {0, -1}, {-1, -1}, {-2, 0}, {0, -2}, {-2, -1}, {-1, -2}, {-2, -2}},
    {{-1, 0}, {0, -1}, {-1, -1}, {-2, 0}, {0, -2}, {-2, -1}, {-1, -2}, {-2, -2}},
    {{-1, 0}, {0, -1}, {-1, -1}, {-2, 0}, {0, -2}, {-2, -1}, {-1, -2}, {-2, -2}},
    {{-1, 0}, {0, -1}, {-1, -1}, {-2, 0}, {0, -2}, {-2, -1}, {-1, -2}, {-2, -2}},
    {{-1, 0}, {0, -1}, {-1, -1}, {-2, 0}, {0, -2}, {-2, -1}, {-1, -2}, {-2, -2}},
    {{0, -1}, {-1, 0}, {1, -1}, {-1, -1}, {0, -2}, {-2, 0}, {-2, -1}, {-1, -2}},
    {{-1, 0}, {0, -1}, {-1, 1}, {-1, -1}, {-2, 0}, {0, -2}, {-1, -2}, {-2, -1}},
    {{-1, 0}, {0, -1}, {-1, 1}, {1, -1}, {-1, -1}, {-3, 0}, {0, -3}, {-3, -3}},
    {{0, -1}, {-1, 0}, {2, -1}, {-1, -1}, {-1, 1}, {0, -3}, {-3, 0}, {-3, -3}},
    {{-1, 0}, {0, -1}, {-1, 2}, {-1, -1}, {1, -1}, {-3, 0}, {0, -3}, {-3, -3}},
    {{-1, 1}, {1, -1}, {-1, 2}, {2, -1}, {-1, -1}, {-3, 0}, {0, -3}, {-3, -3}},
    {{0, -1}, {-1, 0}, {4, -1}, {-1, 2}, {-1, -1}, {0, -3}, {-3, 0}, {2, -1}},
    {{-1, 0}, {0, -1}, {-1, 4}, {2, -1}, {-1, -1}, {-3, 0}, {0, -3}, {-1, 2}},
    {{-1, 3}, {3, -1}, {-1, 4}, {4, -1}, {-1, -1}, {-1, 0}, {0, -1}, {-1, 6}}
}

```

候选块的位置计算如下：

$$(X_{c(i)}, Y_{c(i)}) = (X_0, Y_0) + \text{mv_ref_blocks}[\text{sub_size}][i] \times 8;$$

其中,如果 i 小于 2 且当前块的 sub_size 小于 3,则取候选位置块中对应位置的块的 MV 作为候选 MV。

第二步,如果解码顺序上前一帧中,与当前块相同位置的块使用的某一参考帧,与当前块的参考帧相同,则该块对应参考帧的 MV 加入候选运动矢量集。

第三步,按顺序搜索候选位置 $(X_{c(i)}, Y_{c(i)})$ ($i=0 \sim 7$) 的 8×8 块,如果该位置在图像内且该块使用的某一参考帧与当前块的参考帧不同,则该块对应参考帧的 MV 加入候选运动矢量集。如果该参考帧与当前块参考帧的方向不同,则该块的 MV 符号取反后 $(-\text{mvx}, -\text{mvy})$ 加入候选运动矢量集。当所有候选位置均被搜完后,进入第四步。其中,候选位置与第一步相同。

第四步,如果解码顺序上前一帧中,与当前块相同位置的块使用的某一参考帧,与当前块的参考帧不同,则该块对应参考帧的 MV 加入候选运动矢量集。如果该参考帧与当前块参考帧的方向不同,则该块的 MV 符号取反后 $(-\text{mvx}, -\text{mvy})$ 加入候选运动矢量集。

5.3.5.2.2 亮度运动矢量导出

如果当前块的 skip_flag 等于 1,则当前块的 MV 为 $\{0, 0\}$,对应参考帧为 DYNAMIC_REF。否则:

如果当前块的 block_reference_mode 等于 SINGLE_REFERENCE,则:

- a) 如果当前块的 mv_mode 为 ZEROMV,则当前块的 MV 为 $\{0, 0\}$;
- b) 如果当前块的 mv_mode 为 NEARESTMV,则当前块的 MV 为 $\text{PMV}[0]$;
- c) 如果当前块的 mv_mode 为 NEARMV,则当前块的 MV 为 $\text{PMV}[1]$;
- d) 如果当前块的 mv_mode 为 NEWMV,则当前块的 MV 为 $\text{MVP}[0] + \text{MVD}[0]$ 。

如果当前块的 block_reference_mode 等于 COMPOUND_REFERENCE,则当前块处于双向预测模式,帧间预测有两个参考帧,其中第一参考帧从码流中读取,第二参考帧固定为 OPTIONAL_REF。两个参考帧中依次导出两个运动矢量,分别记为 $\text{MV}[0], \text{MV}[1]$,计算如下:

- a) 如果当前块的 mv_mode 为 ZEROMV,则 $\text{MV}[0]$ 和 $\text{MV}[1]$ 均为 $\{0, 0\}$;
- b) 如果当前块的 mv_mode 为 NEARESTMV,则 $\text{MV}[0]$ 和 $\text{MV}[1]$ 均为 $\text{PMV}[0]$;
- c) 如果当前块的 mv_mode 为 NEARMV,则 $\text{MV}[0]$ 和 $\text{MV}[1]$ 均为 $\text{PMV}[1]$;
- d) 如果当前块的 mv_mode 为 NEWMV,则 $\text{MV}[0]$ 和 $\text{MV}[1]$ 分别为:

$$\begin{aligned} \text{MV}[0] &= \text{PMV}[0] + \text{MVD}[0] \\ \text{MV}[1] &= \text{PMV}[1] + \text{MVD}[1] \end{aligned}$$

5.3.5.3 参考点的导出过程

5.3.5.3.1 亮度样点插值过程

图 12 给出了参考图像亮度分量整数样点、1/2 样点、1/4 样点和 1/8 样点的位置示意图，其中用大写字母标记的为整数样点位置，用小写字母标记的为 1/2、1/4 和 1/8 样点位置。亮度分量分数样点的值由 8 抽头滤波器生成。滤波器分为四种类型，分别为 Regular, Smooth-1, Sharp, Smooth-2，其对应的抽头系数由表 42～表 45 给出。当 interp_filter_switchale 等于 0 时，亮度样点插值采用的滤波器类型为 interp_filter；当 interp_filter_switchale 等于 1 时，亮度样点插值采用的滤波器类型由 interp_filter_mode 的取值决定。

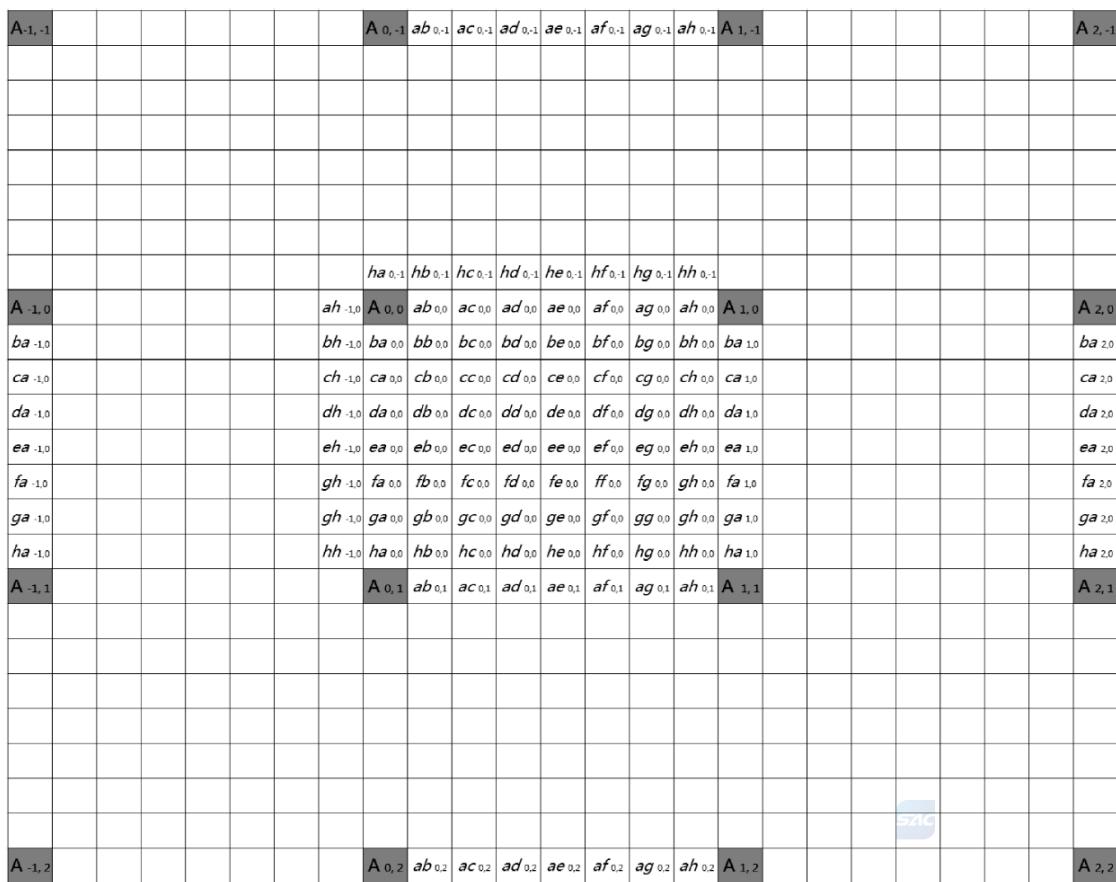


图 12 亮度分量整数样点、1/2、1/4 和 1/8 样点的位置

表 42 Regular 亮度插值滤波器抽头系数

分数样点位置	抽头系数(Regular 类型)
1/8	-1, 3, -9, 121, 19, -7, 2, 0
2/8	-1, 4, -15, 111, 38, -12, 4, -1
3/8	-1, 5, -18, 96, 59, -17, 5, -1

表 42 (续)

分数样点位置	抽头系数(Regular 类型)
4/8	-1, 6, -19, 78, 78, -19, 6, -1
5/8	-1, 5, -17, 59, 96, -18, 5, -1
6/8	-1, 4, -12, 38, 111, -15, 4, -1
7/8	0, 2, -7, 19, 121, -9, 3, -1

表 43 Smooth-1 亮度插值滤波器抽头系数

分数样点位置	抽头系数(Smooth1 类型)
1/8	-2, -1, 28, 62, 42, 1, -2, 0
2/8	-2, -2, 23, 61, 47, 4, -3, 0
3/8	-1, -3, 17, 58, 52, 8, -3, 0
4/8	-1, -3, 13, 55, 55, 13, -3, -1
5/8	0, -3, 8, 52, 58, 17, -3, -1
6/8	0, -3, 4, 47, 61, 23, -2, -2
7/8	0, -2, 1, 42, 62, 28, -1, -2

表 44 Sharp 亮度插值滤波器抽头系数

分数样点位置	抽头系数(Sharp 类型)
1/8	-2, 5, -14, 126, 18, -7, 3, -1
2/8	-4, 9, -21, 116, 38, -14, 6, -2
3/8	-4, 10, -25, 101, 60, -20, 9, -3
4/8	-4, 11, -24, 81, 81, -24, 11, -4
5/8	-3, 9, -20, 60, 101, -25, 10, -4
6/8	-2, 6, -14, 38, 116, -21, 9, -4
7/8	-1, 3, -7, 18, 126, -14, 5, -2

表 45 Smooth-2 亮度插值滤波器抽头系数

分数样点位置	抽头系数(Smooth2 类型)
1/8	-1, 7, 29, 46, 36, 12, 0, -1
2/8	-1, 5, 26, 46, 38, 14, 1, -1
3/8	-1, 4, 23, 44, 41, 17, 1, -1
4/8	-1, 2, 20, 43, 43, 20, 2, -1
5/8	-1, 1, 17, 41, 44, 23, 4, -1
6/8	-1, 1, 14, 38, 46, 26, 5, -1
7/8	-1, 0, 12, 36, 46, 29, 7, -1

运动补偿插值滤波器包含四种不同类型的系数,其插值的方法都是相同的,因此下面均参照 Regular 类型抽头系数来描述,具体计算过程如下:

亮度分量样点位置 $ab_{0,0}, ac_{0,0}, ad_{0,0}, ae_{0,0}$ 的预测值由水平方向距离插值点最近的 8 个整数值滤波得到,计算方法如下:

$$\begin{aligned} ab'_{0,0} &= -A_{-3,0} + 3 \times A_{-2,0} - 9 \times A_{-1,0} + 121 \times A_{0,0} + 19 \times A_{1,0} - 7 \times A_{2,0} + 2 \times A_{3,0} \\ ac'_{0,0} &= -A_{-3,0} + 4 \times A_{-2,0} - 15 \times A_{-1,0} + 111 \times A_{0,0} + 38 \times A_{1,0} - 12 \times A_{2,0} + 4 \times A_{3,0} - A_{4,0} \\ ad'_{0,0} &= -A_{-3,0} + 5 \times A_{-2,0} - 18 \times A_{-1,0} + 96 \times A_{0,0} + 59 \times A_{1,0} - 17 \times A_{2,0} + 5 \times A_{3,0} - A_{4,0} \\ ae'_{0,0} &= -A_{-3,0} + 6 \times A_{-2,0} - 19 \times A_{-1,0} + 78 \times A_{0,0} + 78 \times A_{1,0} - 19 \times A_{2,0} + 6 \times A_{3,0} - A_{4,0} \end{aligned}$$

$ab_{0,0}, ac_{0,0}, ad_{0,0}, ae_{0,0}$ 最终预测值计算方法如下:

$$\begin{aligned} ab_{0,0} &= \text{Clip1}((ab'_{0,0} + 64) \gg 7) \\ ac_{0,0} &= \text{Clip1}((ac'_{0,0} + 64) \gg 7) \\ ad_{0,0} &= \text{Clip1}((ad'_{0,0} + 64) \gg 7) \\ ae_{0,0} &= \text{Clip1}((ae'_{0,0} + 64) \gg 7) \end{aligned}$$

其中,样点位置 $af_{0,0}, ag_{0,0}, ah_{0,0}$ 的预测方式同与样点位置 $ab_{0,0}, ac_{0,0}, ad_{0,0}$ 的预测方法类似,均采用对应位置的插值系数计算获得。

亮度分量样点位置 $ba_{0,0}, ca_{0,0}, da_{0,0}, ea_{0,0}$ 的预测值由垂直方向距离插值点最近的 8 个整数值滤波得到,计算方法如下:

$$\begin{aligned} ba'_{0,0} &= -A_{0,-3} + 3 \times A_{0,-2} - 9 \times A_{0,-1} + 121 \times A_{0,0} + 19 \times A_{0,1} - 7 \times A_{0,2} + 2 \times A_{0,3} \\ ca'_{0,0} &= -A_{0,-3} + 4 \times A_{0,-2} - 15 \times A_{0,-1} + 111 \times A_{0,0} + 38 \times A_{0,1} - 12 \times A_{0,2} + 4 \times A_{0,3} - A_{0,4} \\ da'_{0,0} &= -A_{0,-3} + 5 \times A_{0,-2} - 18 \times A_{0,-1} + 96 \times A_{0,0} + 59 \times A_{0,1} - 17 \times A_{0,2} + 5 \times A_{0,3} - A_{0,4} \\ ea'_{0,0} &= -A_{0,-3} + 6 \times A_{0,-2} - 19 \times A_{0,-1} + 78 \times A_{0,0} + 78 \times A_{0,1} - 19 \times A_{0,2} + 6 \times A_{0,3} - A_{0,4} \end{aligned}$$

$ba_{0,0}, ca_{0,0}, da_{0,0}, ea_{0,0}$ 的最终预测值应通过下列式子得到:

$$\begin{aligned} ba_{0,0} &= \text{Clip1}((ba'_{0,0} + 64) \gg 7) \\ ca_{0,0} &= \text{Clip1}((ca'_{0,0} + 64) \gg 7) \\ da_{0,0} &= \text{Clip1}((da'_{0,0} + 64) \gg 7) \\ ea_{0,0} &= \text{Clip1}((ea'_{0,0} + 64) \gg 7) \end{aligned}$$

其中,样点位置 $fa_{0,0}, ga_{0,0}, ha_{0,0}$ 的预测方式同与样点位置 $ba_{0,0}, ca_{0,0}, da_{0,0}$ 的预测方式类似,均采用对应位置的插值系数计算获得。

其他亮度分量的分数样点,如: $bb_{0,0}, bc_{0,0}, \dots, bh_{0,0}, \dots, hb_{0,0}, hc_{0,0}, \dots, hh_{0,0}$,需要使用整行的分数样点值($ab'_{0,0}, ac'_{0,0}, ad'_{0,0}, ae'_{0,0}, af'_{0,0}, ag'_{0,0}, ah'_{0,0}$)来计算得出,计算方法如下:

$$\begin{aligned} bb'_{0,0} &= -ab'_{0,-3} + 3 \times ab'_{0,-2} - 9 \times ab'_{0,-1} + 121 \times ab'_{0,0} + 19 \times ab'_{0,1} - 7 \times ab'_{0,2} + 2 \times ab'_{0,3} \\ cc'_{0,0} &= -ac'_{0,-3} + 4 \times ac'_{0,-2} - 15 \times ac'_{0,-1} + 111 \times ac'_{0,0} + 38 \times ac'_{0,1} - 12 \times ac'_{0,2} + 4 \times ac'_{0,3} - ac'_{0,4} \\ ee'_{0,0} &= -ae'_{0,-3} + 6 \times ae'_{0,-2} - 19 \times ae'_{0,-1} + 78 \times ae'_{0,0} + 78 \times ae'_{0,1} - 19 \times ae'_{0,2} + 6 \times ae'_{0,3} - ae'_{0,4} \\ hh'_{0,0} &= 2 \times ah'_{0,-2} - 7 \times ah'_{0,-1} + 19 \times ah'_{0,0} + 121 \times ah'_{0,1} - 9 \times ah'_{0,2} + 3 \times ah'_{0,3} - ah'_{0,4} \end{aligned}$$

$bb_{0,0}, cc_{0,0}, ee_{0,0}, hh_{0,0}$ 最终预测值计算方法如下:

$$\begin{aligned} bb_{0,0} &= \text{Clip1}((bb'_{0,0} + 8192) \gg 14) \\ cc_{0,0} &= \text{Clip1}((cc'_{0,0} + 8192) \gg 14) \\ ee_{0,0} &= \text{Clip1}((ee'_{0,0} + 8192) \gg 14) \\ hh_{0,0} &= \text{Clip1}((hh'_{0,0} + 8192) \gg 14) \end{aligned}$$

同样,其余亮度样点位置的预测方式也类似,均采用对应位置的插值系数计算获得。

5.3.5.3.2 色度样点插值过程

图 13 给出了参考图像色度分量整数样点、1/2 样点、1/4 样点、1/8 和 1/16 样点的位置示意图。其

中用大写字母标记的为整数样点位置,用小写字母标记的为 $1/2$ 、 $1/4$ 、 $1/8$ 和 $1/16$ 样点位置。色度分量分数样点的值由8抽头滤波器生成。滤波器分为四种类型,分别为Regular,Smooth-1,Sharp,Smooth-2,其对应的抽头系数由表46~表49给出。色度样点插值采用的滤波器类型与对应位置的亮度样点插值滤波器相同。

	$pa_{0.1}$	$pb_{0.1}$	$pc_{0.1}$	$pd_{0.1}$	$pe_{0.1}$	$pf_{0.1}$	$pg_{0.1}$	$ph_{0.1}$	$pi_{0.1}$	$pj_{0.1}$	$pk_{0.1}$	$pl_{0.1}$	$pm_{0.1}$	$pn_{0.1}$	$po_{0.1}$	$pp_{0.1}$	
$ap_{-1,0}$	B $0,0$	$ab_{0,0}$	$ac_{0,0}$	$ad_{0,0}$	$ae_{0,0}$	$af_{0,0}$	$ag_{0,0}$	$ah_{0,0}$	$ai_{0,0}$	$aj_{0,0}$	$ak_{0,0}$	$al_{0,0}$	$am_{0,0}$	$an_{0,0}$	$ao_{0,0}$	$ap_{0,0}$	B $1,0$
$bp_{-1,0}$	$ba_{0,0}$	$bb_{0,0}$	$bc_{0,0}$	$bd_{0,0}$	$be_{0,0}$	$bf_{0,0}$	$bg_{0,0}$	$bh_{0,0}$	$bi_{0,0}$	$bj_{0,0}$	$bk_{0,0}$	$bl_{0,0}$	$bm_{0,0}$	$bn_{0,0}$	$bo_{0,0}$	$bp_{0,0}$	$ba_{1,0}$
$cp_{-1,0}$	$ca_{0,0}$	$cb_{0,0}$	$cc_{0,0}$	$cd_{0,0}$	$ce_{0,0}$	$cf_{0,0}$	$cg_{0,0}$	$ch_{0,0}$	$ci_{0,0}$	$cj_{0,0}$	$ck_{0,0}$	$cl_{0,0}$	$cm_{0,0}$	$cn_{0,0}$	$co_{0,0}$	$cp_{0,0}$	$ca_{1,0}$
$dp_{-1,0}$	$da_{0,0}$	$db_{0,0}$	$dc_{0,0}$	$dd_{0,0}$	$de_{0,0}$	$df_{0,0}$	$dg_{0,0}$	$dh_{0,0}$	$di_{0,0}$	$dj_{0,0}$	$dk_{0,0}$	$dl_{0,0}$	$dm_{0,0}$	$dn_{0,0}$	$do_{0,0}$	$dp_{0,0}$	$da_{1,0}$
$ep_{-1,0}$	$ea_{0,0}$	$eb_{0,0}$	$ec_{0,0}$	$ed_{0,0}$	$ee_{0,0}$	$ef_{0,0}$	$eg_{0,0}$	$eh_{0,0}$	$ei_{0,0}$	$ej_{0,0}$	$ek_{0,0}$	$el_{0,0}$	$em_{0,0}$	$en_{0,0}$	$eo_{0,0}$	$ep_{0,0}$	$ea_{1,0}$
$fp_{-1,0}$	$fa_{0,0}$	$fb_{0,0}$	$fc_{0,0}$	$fd_{0,0}$	$fe_{0,0}$	$ff_{0,0}$	$fg_{0,0}$	$fh_{0,0}$	$fi_{0,0}$	$fj_{0,0}$	$fk_{0,0}$	$fl_{0,0}$	$fm_{0,0}$	$fn_{0,0}$	$fo_{0,0}$	$fp_{0,0}$	$fa_{1,0}$
$gp_{-1,0}$	$ga_{0,0}$	$gb_{0,0}$	$gc_{0,0}$	$gd_{0,0}$	$ge_{0,0}$	$gf_{0,0}$	$gg_{0,0}$	$gh_{0,0}$	$gi_{0,0}$	$gj_{0,0}$	$gk_{0,0}$	$gl_{0,0}$	$gm_{0,0}$	$gn_{0,0}$	$go_{0,0}$	$gp_{0,0}$	$ga_{1,0}$
$hp_{-1,0}$	$ha_{0,0}$	$hb_{0,0}$	$hc_{0,0}$	$hd_{0,0}$	$he_{0,0}$	$hf_{0,0}$	$hg_{0,0}$	$hh_{0,0}$	$hi_{0,0}$	$hj_{0,0}$	$hk_{0,0}$	$hl_{0,0}$	$hm_{0,0}$	$hn_{0,0}$	$ho_{0,0}$	$hp_{0,0}$	$ha_{1,0}$
$ip_{-1,0}$	$ia_{0,0}$	$ib_{0,0}$	$ic_{0,0}$	$id_{0,0}$	$ie_{0,0}$	$if_{0,0}$	$ig_{0,0}$	$ih_{0,0}$	$ii_{0,0}$	$ij_{0,0}$	$ik_{0,0}$	$il_{0,0}$	$im_{0,0}$	$in_{0,0}$	$io_{0,0}$	$ip_{0,0}$	$ia_{1,0}$
$jp_{-1,0}$	$ja_{0,0}$	$jb_{0,0}$	$jc_{0,0}$	$jd_{0,0}$	$je_{0,0}$	$jf_{0,0}$	$dg_{0,0}$	$jh_{0,0}$	$ji_{0,0}$	$jj_{0,0}$	$jk_{0,0}$	$jl_{0,0}$	$jm_{0,0}$	$jn_{0,0}$	$jo_{0,0}$	$jp_{0,0}$	$ja_{1,0}$
$kp_{-1,0}$	$ka_{0,0}$	$kb_{0,0}$	$kc_{0,0}$	$kd_{0,0}$	$ke_{0,0}$	$kf_{0,0}$	$kg_{0,0}$	$kh_{0,0}$	$ki_{0,0}$	$kj_{0,0}$	$kk_{0,0}$	$kl_{0,0}$	$km_{0,0}$	$kn_{0,0}$	$ko_{0,0}$	$kp_{0,0}$	$ka_{1,0}$
$lp_{-1,0}$	$la_{0,0}$	$lb_{0,0}$	$lc_{0,0}$	$ld_{0,0}$	$le_{0,0}$	$lf_{0,0}$	$lg_{0,0}$	$lh_{0,0}$	$li_{0,0}$	$lj_{0,0}$	$lk_{0,0}$	$ll_{0,0}$	$lm_{0,0}$	$ln_{0,0}$	$lo_{0,0}$	$lp_{0,0}$	$la_{1,0}$
$mp_{-1,0}$	$ma_{0,0}$	$mb_{0,0}$	$mc_{0,0}$	$md_{0,0}$	$me_{0,0}$	$mf_{0,0}$	$mg_{0,0}$	$mh_{0,0}$	$mi_{0,0}$	$mj_{0,0}$	$mk_{0,0}$	$ml_{0,0}$	$mm_{0,0}$	$mn_{0,0}$	$mo_{0,0}$	$mp_{0,0}$	$ma_{1,0}$
$np_{-1,0}$	$na_{0,0}$	$nb_{0,0}$	$nc_{0,0}$	$nd_{0,0}$	$ne_{0,0}$	$nf_{0,0}$	$ng_{0,0}$	$nh_{0,0}$	$ni_{0,0}$	$nj_{0,0}$	$nk_{0,0}$	$nl_{0,0}$	$nm_{0,0}$	$nn_{0,0}$	$no_{0,0}$	$np_{0,0}$	$na_{1,0}$
$op_{-1,0}$	$oa_{0,0}$	$ob_{0,0}$	$oc_{0,0}$	$od_{0,0}$	$oe_{0,0}$	$of_{0,0}$	$og_{0,0}$	$oh_{0,0}$	$oi_{0,0}$	$oj_{0,0}$	$ok_{0,0}$	$ol_{0,0}$	$om_{0,0}$	$on_{0,0}$	$oo_{0,0}$	$op_{0,0}$	$oa_{1,0}$
$pp_{-1,0}$	$pa_{0,0}$	$pb_{0,0}$	$pc_{0,0}$	$pd_{0,0}$	$pe_{0,0}$	$pf_{0,0}$	$pg_{0,0}$	$ph_{0,0}$	$pi_{0,0}$	$pj_{0,0}$	$pk_{0,0}$	$pl_{0,0}$	$pm_{0,0}$	$pn_{0,0}$	$po_{0,0}$	$pp_{0,0}$	$pa_{1,0}$
	B $0,1$	$ab_{0,1}$	$ac_{0,1}$	$ad_{0,1}$	$ae_{0,1}$	$af_{0,1}$	$ag_{0,1}$	$ah_{0,1}$	$ai_{0,1}$	$aj_{0,1}$	$ak_{0,1}$	$al_{0,1}$	$am_{0,1}$	$an_{0,1}$	$ao_{0,1}$	$ap_{0,1}$	B $1,1$

图 13 色度分量整数样点、 $1/2$ 、 $1/4$ 、 $1/8$ 和 $1/16$ 样点的位置

表 46 色度插值滤波器抽头系数-1

分数样点位置	抽头系数(Regular 类型)
$1/16$	0, 1, -4, 125, 9, -4, 1, 0
$2/16$	-1, 3, -9, 121, 19, -7, 2, 0
$3/16$	-1, 4, -12, 117, 28, -10, 3, -1
$4/16$	-1, 4, -15, 111, 38, -12, 4, -1
$5/16$	-1, 5, -17, 104, 49, -15, 4, -1
$6/16$	-1, 5, -18, 96, 59, -17, 5, -1
$7/16$	-1, 6, -18, 87, 69, -19, 5, -1
$8/16$	-1, 6, -19, 78, 78, -19, 6, -1
$9/16$	-1, 5, -19, 69, 87, -18, 6, -1
$10/16$	-1, 5, -17, 59, 96, -18, 5, -1
$11/16$	-1, 4, -15, 49, 104, -17, 5, -1
$12/16$	-1, 4, -12, 38, 111, -15, 4, -1

表 46 (续)

分数样点位置	抽头系数(Regular 类型)
13/16	-1, 3, -10, 28, 117, -12, 4, -1
14/16	0, 2, -7, 19, 121, -9, 3, -1
15/16	0, 1, -4, 9, 125, -4, 1, 0

表 47 色度插值滤波器抽头系数-2

分数样点位置	抽头系数(Smooth1 类型)
1/16	-3, -1, 31, 63, 39, 1, -2, 0
2/16	-2, -1, 28, 62, 42, 1, -2, 0
3/16	-2, -1, 25, 62, 44, 3, -3, 0
4/16	-2, -2, 23, 61, 47, 4, -3, 0
5/16	-2, -2, 20, 59, 50, 6, -3, 0
6/16	-1, -3, 17, 58, 52, 8, -3, 0
7/16	-1, -3, 15, 56, 54, 11, -3, -1
8/16	-1, -3, 13, 55, 55, 13, -3, -1
9/16	-1, -3, 11, 54, 56, 15, -3, -1
10/16	0, -3, 8, 52, 58, 17, -3, -1
11/16	0, -3, 6, 50, 59, 20, -2, -2
12/16	0, -3, 4, 47, 61, 23, -2, -2
13/16	0, -3, 3, 44, 62, 25, -1, -2
14/16	0, -2, 1, 42, 62, 28, -1, -2
15/16	0, -2, 1, 39, 63, 31, -1, -3

表 48 色度插值滤波器抽头系数-3

分数样点位置	抽头系数(Sharp 类型)
1/16	-1, 3, -8, 127, 10, -4, 1, 0
2/16	-2, 5, -14, 126, 18, -7, 3, -1
3/16	-3, 7, -18, 122, 28, -11, 5, -2
4/16	-4, 9, -21, 116, 38, -14, 6, -2
5/16	-4, 10, -24, 109, 49, -17, 8, -3
6/16	-4, 10, -25, 101, 60, -20, 9, -3
7/16	-4, 11, -25, 91, 71, -22, 10, -4
8/16	-4, 11, -24, 81, 81, -24, 11, -4
9/16	-4, 10, -22, 71, 91, -25, 11, -4

表 48 (续)

分数样点位置	抽头系数(Sharp 类型)
10/16	-3, 9, -20, 60, 101, -25, 10, -4
11/16	-3, 8, -17, 49, 109, -24, 10, -4
12/16	-2, 6, -14, 38, 116, -21, 9, -4
13/16	-2, 5, -11, 28, 122, -18, 7, -3
14/16	-1, 3, -7, 18, 126, -14, 5, -2
15/16	0, 1, -4, 10, 127, -8, 3, -1



表 49 色度插值滤波器抽头系数-4

分数样点位置	抽头系数(Smooth2 类型)
1/16	-1, 8, 31, 47, 34, 10, 0, -1
2/16	-1, 7, 29, 46, 36, 12, 0, -1
3/16	-1, 6, 28, 46, 37, 13, 0, -1
4/16	-1, 5, 26, 46, 38, 14, 1, -1
5/16	-1, 4, 25, 45, 39, 16, 1, -1
6/16	-1, 4, 23, 44, 41, 17, 1, -1
7/16	-1, 3, 21, 44, 42, 18, 2, -1
8/16	-1, 2, 20, 43, 43, 20, 2, -1
9/16	-1, 2, 18, 42, 44, 21, 3, -1
10/16	-1, 1, 17, 41, 44, 23, 4, -1
11/16	-1, 1, 16, 39, 45, 25, 4, -1
12/16	-1, 1, 14, 38, 46, 26, 5, -1
13/16	-1, 0, 13, 37, 46, 28, 6, -1
14/16	-1, 0, 12, 36, 46, 29, 7, -1
15/16	-1, 0, 10, 34, 47, 31, 8, -1

色度分量样点运动补偿插值方法与亮度分量类似,也是先对整数样点所在的行或列进行插值,再利用其结果对其余分数样点位置进行插值。同亮度插值一样,下面均参照 Regular 类型抽头系数来描述。例如,图 13 中的 $ah_{0,0}$, $ha_{0,0}$, $hh_{0,0}$ 的计算方法如下:

$$ah'_{0,0} = -B_{-3,0} + 6 \times B_{-2,0} - 18 \times B_{-1,0} + 87 \times B_{0,0} + 69 \times B_{1,0} - 19 \times B_{2,0} + 5 \times B_{3,0} - B_{4,0}$$

$$ha'_{0,0} = -B_{0,-3} + 6 \times B_{0,-2} - 18 \times B_{0,-1} + 87 \times B_{0,0} + 69 \times A_{0,1} - 19 \times B_{0,2} + 5 \times B_{0,3} - B_{0,4}$$

$ah_{0,0}$, $ha_{0,0}$ 最终预测值计算方法如下:

$$ah_{0,0} = Clip1((ah'_{0,0} + 64) \gg 7)$$

$$ha_{0,0} = Clip1((ha'_{0,0} + 64) \gg 7)$$

其他色度分量的分数样点,如: $bb_{0,0}$, $bc_{0,0}$... $bh_{0,0}$, ... $hb_{0,0}$, $hc_{0,0}$... $hh_{0,0}$,需要使用第一步中计算的整数样点位置的行的分数样点值($ab'_{0,0}$, $ac'_{0,0}$, $ad'_{0,0}$, $ae'_{0,0}$, $af'_{0,0}$, $ag'_{0,0}$, $ah'_{0,0}$)来计算得出,计算方法如下:

$$hh'_{0,0} = -ah'_{0,-3} + 6 \times ah'_{0,-2} - 19 \times ah'_{0,-1} + 78 \times ah'_{0,0} + 78 \times ah'_{0,1} - 19 \times ah'_{0,2} + 6 \times ah'_{0,3} - ah'_{0,4}$$

$hh_{0,0}$ 的最终预测值计算方法如下：

$$hh_{0,0} = \text{Clip1}((hh'_{0,0} + 8192) \gg 14)$$

同样,其余色度样点位置的预测方式也类似,均采用对应位置的插值系数计算获得。

5.3.6 变换系数解码过程以及图像重建过程

5.3.6.1 概述

本过程在去块滤波过程之前进行,包括块系数解析,逆扫描,反量化,反变换和重建。

逆扫描的输入为由块解析生成的数组 QuantCoeffArray,输出为量化系数矩阵 QuantCoeffMatrix。

反量化的输入为量化系数矩阵 QuantCoeffMatrix,当前块的量化参数 QP,输出为反量化后的变换系数矩阵 CoeffMatrix。

反变换的输入为 4×4 或 8×8 或 16×16 或 32×32 变换系数矩阵 CoeffMatrix,输出为 4×4 或 8×8 或 16×16 或 32×32 残差样点矩阵 ResidueMatrix。

重建过程的输入为残差样点矩阵 ResidueMatrix 和预测样点矩阵 predMatrix,输出为重建样点矩阵 RecMatrix。

5.3.6.2 块系数解析

设置 max_eob 为当前块中的样点个数,块系数数组 QuantCoeffArray 初始化为全 0,块系数索引 i 等于 0。块解析按如下步骤进行:

第一步,解析 coeff_value[i],其过程见 5.4.2,如果该值为 EOB,则索引值大于等于 i 的块系数均为 0,块解析过程结束,否则进入下一步;

第二步,如果 coeff_value[i] 等于 0,则 QuantCoeffArray[i] 等于 0,块系数索引 i 增加 1,继续解析下一个 coeff_value[i],直至解析到一个 coeff_value[i] 大于 0 后进入下一步;

第三步,获得 coeff_sign[i],如果 coeff_sign[i] 等于 0,块系数 QuantCoeffArray[i] 的取值等于 coeff_value[i];如果 coeff_sign[i] 等于 1,块系数 QuantCoeffArray[i] 的取值等于 -coeff_value[i];

第四步,块系数索引 i 增加 1。如果当前块中的所有系数均已解析出,块解析过程结束,否则回到第一步。

5.3.6.3 逆扫描

设 QuantCoeffArray 数组中某元素的地址为 k,通过逆扫描找到地址为 k 的单元对应的列号 i 和行号 j,然后将 QuantCoeffArray[k] 赋给 QuantCoeffMatrix[i,j]。根据块尺寸及编码模式的不同,扫描方式定义如图 14。

当编码块采用帧间编码或编码块为色度块时,根据块尺寸,分别采用 4×4 扫描、 8×8 扫描、 16×16 扫描和 32×32 扫描。

当编码块采用帧内编码且为亮度块时:

- 1) 若 luma_intra_mode 的取值大于等于 9 小于等于 17 时,根据块尺寸分别采用 4×4 行扫描、 8×8 行扫描、 16×16 行扫描和 32×32 行扫描;
- 2) 若 luma_intra_mode 的取值大于等于 24 小于等于 31 时,根据块尺寸分别采用 4×4 列扫描、 8×8 列扫描、 16×16 列扫描和 32×32 列扫描;
- 3) 否则,根据块尺寸,分别采用 4×4 扫描、 8×8 扫描、 16×16 扫描和 32×32 扫描。

	0	1	2	3	i
0	0	4	1	5	
1	8	2	12	9	
2	3	6	13	10	
3	7	14	11	15	
					j

a) 4×4 扫描

	0	1	2	3	i
0	0	1	4	2	
1	5	3	6	8	
2	9	7	12	10	
3	13	11	14	15	
					j

b) 4×4 行扫描

	0	1	2	3	i
0	0	4	8	1	
1	12	5	9	2	
2	13	6	10	3	
3	7	14	11	15	
					j

c) 4×4 列扫描图 14 4×4 、 8×8 、 16×16 、 32×32 块的逆扫描

	0	1	2	3	4	5	6	7	i
0	0	8	1	16	9	2	17	24	
1	10	3	18	25	32	11	4	26	
2	33	19	40	12	34	27	5	41	
3	20	48	13	35	42	28	21	6	
4	49	56	36	43	29	7	14	50	
5	57	44	22	37	15	51	58	30	
6	45	23	52	59	38	31	60	53	
7	46	39	61	54	47	62	55	63	

j

d) 8×8 扫描

	0	1	2	3	4	5	6	7	i
0	0	1	2	8	9	3	16	10	
1	4	17	11	24	5	18	25	12	
2	19	26	32	6	13	20	33	27	
3	7	34	40	21	28	41	14	35	
4	48	42	29	36	49	22	43	15	
5	56	37	50	44	30	57	23	51	
6	58	45	38	52	31	59	53	46	
7	60	39	61	47	54	55	62	63	

j

e) 8×8 行扫描

图 14 (续)

	0	1	2	3	4	5	6	7	i
0	0	8	16	1	24	9	32	17	
1	2	40	25	10	33	18	48	3	
2	26	41	11	56	19	34	4	49	
3	27	42	12	35	20	57	50	28	
4	5	43	13	36	58	51	21	44	
5	6	29	59	37	14	52	22	7	
6	45	60	30	15	38	53	23	46	
7	31	61	39	54	47	62	55	63	

j

f) 8×8 列扫描

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	i
0	0	16	1	32	17	2	48	33	18	3	64	34	49	19	65	80	
1	50	4	35	66	20	81	96	51	5	36	82	97	67	112	21	52	
2	98	37	83	113	6	68	128	53	22	99	114	84	7	129	38	69	
3	100	115	144	130	85	54	23	8	145	39	70	116	101	131	160	146	
4	55	86	24	71	132	117	161	40	9	102	147	176	162	87	56	25	
5	133	118	177	148	72	103	41	163	10	192	178	88	57	134	149	119	
6	26	164	73	104	193	42	179	208	11	135	89	165	120	150	58	194	
7	180	27	74	209	105	151	136	43	90	224	166	195	181	121	210	59	
8	12	152	106	167	196	75	137	225	211	240	182	122	91	28	197	13	
9	226	168	183	153	44	212	138	107	241	60	29	123	198	184	227	169	
10	242	76	213	154	45	92	14	199	139	61	228	214	170	185	243	108	
11	77	155	30	15	200	229	124	215	244	93	46	186	171	201	109	140	
12	230	62	216	245	31	125	78	156	231	47	187	202	217	94	246	141	
13	63	232	172	110	247	157	79	218	203	126	233	188	248	95	173	142	
14	219	111	249	234	158	127	189	204	250	235	143	174	220	205	159	251	
15	190	221	175	236	237	191	206	252	222	253	207	238	223	254	239	255	

j

g) 16×16 扫描

图 14 (续)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	i
0	0	1	2	16	3	17	4	18	32	5	33	19	6	34	48	20	
1	49	7	35	21	50	64	8	36	65	22	51	37	80	9	66	52	
2	23	38	81	67	10	53	24	82	68	96	39	11	54	83	97	69	
3	25	98	84	40	112	55	12	70	99	113	85	26	41	56	114	100	
4	13	71	128	86	27	115	101	129	42	57	72	116	14	87	130	102	
5	144	73	131	117	28	58	15	88	43	145	103	132	146	118	74	160	
6	89	133	104	29	59	147	119	44	161	148	90	105	134	162	120	176	
7	75	135	149	30	60	163	177	45	121	91	106	164	178	150	192	136	
8	165	179	31	151	193	76	122	61	137	194	107	152	180	208	46	166	
9	167	195	92	181	138	209	123	153	224	196	77	168	210	182	240	108	
10	197	62	154	225	183	169	211	47	139	93	184	226	212	241	198	170	
11	124	155	199	78	213	185	109	227	200	63	228	242	140	214	171	186	
12	156	229	243	125	94	201	244	215	216	230	141	187	202	79	172	110	
13	157	245	217	231	95	246	232	126	203	247	233	173	218	142	111	158	
14	188	248	127	234	219	249	189	204	143	174	159	250	235	205	220	175	
15	190	251	221	191	206	236	207	237	252	222	253	223	238	239	254	255	
																j	

h) 16×16 行扫描

图 14 (续)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	i
0	0	16	32	48	1	64	17	80	33	96	49	2	65	112	18	81	i
1	34	128	50	97	3	66	144	19	113	35	82	160	98	51	129	4	i
2	67	176	20	114	145	83	36	99	130	52	192	5	161	68	115	21	i
3	146	84	208	177	37	131	100	53	162	224	69	6	116	193	147	85	i
4	22	240	132	38	178	101	163	54	209	117	70	7	148	194	86	179	i
5	225	23	133	39	164	8	102	210	241	55	195	118	149	71	180	24	i
6	87	226	134	165	211	40	103	56	72	150	196	242	119	9	181	227	i
7	88	166	25	135	41	104	212	57	151	197	120	73	243	182	136	167	i
8	213	89	10	228	105	152	198	26	42	121	183	244	168	58	137	229	i
9	74	214	90	153	199	184	11	106	245	27	122	230	169	43	215	59	i
10	200	138	185	246	75	12	91	154	216	231	107	28	44	201	123	170	i
11	60	247	232	76	139	13	92	217	186	248	155	108	29	124	45	202	i
12	233	171	61	14	77	140	15	249	93	30	187	156	218	46	109	125	i
13	62	172	78	203	31	141	234	94	47	188	63	157	110	250	219	79	i
14	126	204	173	142	95	189	111	235	158	220	251	127	174	143	205	236	i
15	159	190	221	252	175	206	237	191	253	222	238	207	254	223	239	255	i
																	j

SAC

i) 16 × 16 列扫描

图 14 (续)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	i
0	0	32	1	64	33	2	96	65	34	128	3	97	66	160	129	35	98	4	67	130	161	192	36	99	224	5	162	193	68	131	37	100	
1	225	194	256	163	69	132	6	226	257	288	195	101	164	38	258	7	227	289	133	320	70	196	165	290	259	228	39	321	102	352	8	197	
2	71	134	322	291	260	353	384	229	166	103	40	354	323	292	135	385	198	261	72	9	416	167	386	355	230	324	104	293	41	417	199	136	
3	262	387	448	325	356	10	73	418	231	168	449	294	388	105	419	263	42	200	357	450	137	480	74	326	232	11	389	169	295	420	106	451	
4	481	358	264	327	201	43	138	512	482	390	296	233	170	421	75	452	359	12	513	265	483	328	107	202	514	544	422	391	453	139	44	234	
5	484	297	360	171	76	515	545	266	329	454	13	423	203	108	546	485	576	298	235	140	361	330	172	547	45	455	267	577	486	77	204	362	
6	608	14	299	578	109	236	487	609	331	141	579	46	15	173	610	363	78	205	16	110	237	611	142	47	174	79	206	17	111	238	48	143	
7	80	175	112	207	49	18	239	81	113	19	50	82	114	51	83	115	640	516	392	268	144	20	672	641	548	517	424	393	300	269	176	145	
8	52	21	704	673	642	580	549	518	456	425	394	332	301	270	208	177	146	84	53	22	736	705	674	643	612	581	550	519	488	457	426	395	
9	364	333	302	271	240	209	178	147	116	85	54	23	737	706	675	613	582	551	489	458	427	365	334	303	241	210	179	117	86	55	738	707	
10	614	583	490	459	366	335	242	211	118	87	739	615	491	367	243	119	768	644	520	396	272	148	24	800	769	676	645	552	521	428	397	304	
11	273	180	149	56	25	832	801	770	708	677	646	584	553	522	460	429	398	336	305	274	212	181	150	88	57	26	864	833	802	771	740	709	
12	678	647	616	585	554	523	492	461	430	399	368	337	306	275	244	213	182	151	120	89	58	27	865	834	803	741	710	679	617	586	555	493	
13	462	431	369	338	307	245	214	183	121	90	59	866	835	742	711	618	587	494	463	370	339	246	215	122	91	867	743	619	495	371	247	123	
14	896	772	648	524	400	276	152	28	928	897	804	773	680	649	556	525	432	401	308	277	184	153	60	29	960	929	898	836	805	774	712	681	
15	650	588	557	526	464	433	402	340	309	278	216	185	154	92	61	30	992	961	930	899	868	837	806	775	744	713	682	651	620	589	558	527	
16	496	465	434	403	372	341	310	279	248	217	186	155	124	93	62	31	993	962	931	869	838	807	745	714	683	621	590	559	497	466	435	373	
17	342	311	249	218	187	125	94	63	994	963	870	839	746	715	622	591	498	467	374	343	250	219	126	95	995	871	747	623	499	375	251	127	
18	900	776	652	528	404	280	156	932	901	808	777	684	653	560	529	436	405	312	281	188	157	964	933	902	840	809	778	716	685	654	592	561	
19	530	468	437	406	344	313	282	220	189	158	996	965	934	903	872	841	810	779	748	717	686	655	624	593	562	531	500	469	438	407	376	345	
20	314	283	252	221	190	159	997	966	935	873	842	811	749	718	687	625	594	563	501	470	439	377	346	315	253	222	191	998	967	874	843	750	
21	719	626	595	502	471	378	347	254	223	999	875	751	627	503	379	255	904	780	656	532	408	284	936	905	812	781	688	657	564	533	440	409	
22	316	285	968	937	906	844	813	782	720	689	658	596	565	534	472	441	410	348	317	286	1000	969	938	907	876	845	814	783	752	721	690	659	
23	628	597	566	535	504	473	442	411	380	349	318	287	1001	970	939	877	846	815	753	722	691	629	598	567	505	474	443	381	350	319	1002	971	
24	878	847	754	723	630	599	506	475	382	351	1003	879	755	631	507	383	908	784	660	536	412	940	909	816	785	692	661	568	537	444	413	972	
25	941	910	848	817	786	724	693	662	600	569	538	476	445	414	1004	973	942	911	880	849	818	787	756	725	694	663	632	601	570	539	508	477	
26	446	415	1005	974	943	881	850	819	757	726	695	633	602	571	509	478	447	1006	975	882	851	758	727	634	603	510	479	1007	883	759	635	511	
27	912	788	664	540	944	913	820	789	696	665	572	541	976	945	914	852	821	790	728	697	666	604	573	542	1008	977	946	915	884	853	822	791	
28	760	729	698	667	636	605	574	543	1009	978	947	885	854	823	761	730	699	637	606	575	1010	979	886	855	762	731	638	607	1011	887	763	639	
29	916	792	668	948	917	824	793	700	669	980	949	918	856	825	794	732	701	670	1012	981	950	919	888	857	826	795	764	733	702	671	1013	982	
30	951	889	858	827	765	734	703	1014	983	890	859	766	735	1015	891	767	920	796	952	921	828	797	984	953	922	860	829	798	1016	985	954	923	
31	892	861	830	799	1017	986	955	893	862	831	1018	987	894	863	1019	895	924	956	925	988	957	926	1020	989	958	927	1021	990	959	1022	991	1023	

j

j) 32×32 扫描

图 14 (续)