

ICS 35.040

L71

声明：本稿是 AVS 工作组上报国家标准主管部门的最终稿的技术部分，供工作组会员参考。带有官方规定的正式标准请向中国标准化出版社购买(标准号 GB/T 200090.2-2006)，预计 2006 年 5 月出版。



中华人民共和国国家标准

GB/T 20090.2—2006

信息技术 先进音视频编码

第 2 部分：视频

Information technology - Advanced coding
of audio and video - Part 2: Video

(报批稿)

说明：本稿是 AVS 工作组上报国家标准主管部门的最终稿，供工作组会员参考。
正式官方版本请向中国标准化出版社购买。

××××-××-××发布

××××-××-××实施

国家质量监督检验检疫总局 发布

目 次

前 言	IV
引 言	V
0.1 目标	V
0.2 应用	V
0.3 档次和级别	V
0.4 技术概述	V
0.4.1 预测技术	V
0.4.2 图像分块	V
0.4.3 变换和量化	V
0.5 如何阅读本部分	V
0.6 相关专利情况说明	VI
1 范围	1
2 规范性引用文件	1
3 术语和定义	1
4 缩略语	7
5 约定	7
5.1 算术运算符	7
5.2 逻辑运算符	8
5.3 关系运算符	8
5.4 位运算符	8
5.5 赋值	8
5.6 数学函数	8
5.7 比特流语法、解析过程和解码过程的描述	9
5.7.1 描述方法	9
5.7.2 函数	10
5.7.3 描述符	11
5.7.4 保留、禁止和标记	12
6 编码比特流的结构	12
6.1 视频序列	12
6.1.1 逐行和隔行视频序列	12
6.1.2 序列头	12
6.2 图像	12
6.2.1 图像格式	13
6.2.2 图像类型	14
6.2.3 图像间的顺序	14
6.2.4 参考图像	14
6.3 条带	14
6.4 宏块	15
6.5 8×8 块	15

7 比特流的语法和语义	16
7.1 语法描述	16
7.1.1 起始码	16
7.1.2 视频序列定义	16
7.1.3 图像定义	20
7.1.4 条带定义	22
7.1.5 宏块定义	23
7.1.6 块定义	24
7.2 语义描述	25
7.2.1 视频扩展	25
7.2.2 视频序列	25
7.2.3 图像	34
7.2.4 条带	38
7.2.5 宏块	39
7.2.6 块	40
8 解析过程	40
8.1 k 阶指数哥伦布码	40
8.2 $ue(v)$, $se(v)$ 和 $me(v)$	41
8.3 $ce(v)$	44
9 解码过程	45
9.1 高层语法结构	45
9.2 图像头解码	45
9.3 条带解码	45
9.4 宏块解码	45
9.4.1 宏块类型	45
9.4.2 帧内预测模式	47
9.4.3 参考图像选择	49
9.4.4 运动矢量	51
9.4.5 宏块编码模板	53
9.4.6 量化参数	53
9.5 块解码	53
9.5.1 变长码解码	53
9.5.2 逆扫描	54
9.6 反量化	55
9.6.1 确定量化参数	55
9.6.2 反量化	56
9.7 反变换	56
9.8 帧内预测	57
9.8.1 参考样本的获得	57
9.8.2 亮度块帧内预测	57
9.8.3 色度块帧内预测	58
9.9 帧间预测	59
9.9.1 亮度运动矢量导出	59
9.9.2 参考样本的导出过程	63

9.9.3 加权预测	65
9.10 重建	65
9.11 环路滤波	65
9.11.1 边界滤波强度的推导过程	66
9.11.2 块边界阈值的推导过程	67
9.11.3 B_s 等于 2 时的边界滤波过程	68
9.11.4 B_s 等于 1 时的边界滤波过程	68
附录 A (规范性附录) 变长码表	70
附录 B (规范性附录) 档次和级别	79
B.1 档次	79
B.2 级别	79
B.2.1 本部分定义的级别	79
B.2.2 与档次无关的级别限制	80
附录 C (规范性附录) 伪起始码	83
附录 D (规范性附录) 比特流虚拟参考解码器	84
D.1 约定	84
D.1.1 约定一	84
D.1.2 约定二	84
D.1.3 约定三	84
D.2 基本操作	84
D.2.1 数据输入	84
D.2.1.1 方法一	84
D.2.1.2 方法二	85
D.2.2 数据移出	85
D.2.2.1 非低延迟	85
D.2.2.2 低延迟	86
D.3 缓冲区检测时间间隔	86
D.3.1 非低延迟	86
D.3.2 低延迟	86

前 言

GB/T 20090在《信息技术 先进音视频编码》的总标题下,包括以下九个部分:

第1部分:系统;

第2部分:视频;

第3部分:音频;

第4部分:一致性测试;

第5部分:参考软件;

第6部分:数字媒体版权管理;

第7部分:移动视频;

第8部分:在IP网络上传输AVS;

第9部分:AVS文件格式。

本部分为GB/T 20090的第2部分。

本部分的附录A~附录D为规范性附录。

本部分由中华人民共和国信息产业部提出。

本部分由全国信息技术标准化技术委员会归口。

本部分由信息产业部批准的数字音视频编解码技术标准工作组组织起草。本部分起草单位(技术提案被采纳的单位):中国科学院计算技术研究所、清华大学、浙江大学、华中科技大学、北京工业大学、中山大学、华为技术有限公司、上海广电(集团)有限公司中央研究院、北京长信嘉信息技术有限公司。

本部分主要起草人:高文(工作组组长)、黄铁军(工作组秘书长)、吴枫(视频专题组组长)、何芸(视频专题组联合组长)、虞露(视频专题组联合组长)、梁凡(编辑主笔)、赵海武、马思伟、吕岩、李国平、张志明、沈燕飞、周敏华、贾云卫、郭红星、楼剑、熊联欢。

引 言

0.1 目标

GB/T 20090.2是为了适应数字电视广播、数字存储媒体、网络流媒体、多媒体通信等应用中对运动图像压缩技术的需要而制定的。

0.2 应用

本部分适用的范围包括但不限于下述领域：

数字地面电视广播（DTTB, Digital terrestrial television broadcasting）

有线电视（CATV, Cable TV）

交互存储媒体

直播卫星视频业务（DBS, Direct broadcast satellite video services）

宽带视频业务

多媒体邮件

分组网络的多媒体业务（MSPN, Multimedia services on packet networks）

实时通信业务（视频会议，可视电话等）

远程视频监控

0.3 档次和级别

本部分能支持多种比特率、分辨率和质量的视频压缩。考虑到应用本部分时的互操作性，定义档次和级别。

档次是本部分规定的语法、语义及算法的子集。

级别是在某一档次下对语法元素和语法元素参数值的限定集合。

0.4 技术概述

本部分采用了一系列技术来达到高效率的视频编码，包括帧内预测、帧间预测、变换、量化和熵编码等。帧间预测使用基于块的运动矢量来消除图像间的冗余；帧内预测使用空间预测模式来消除图像内的冗余。再通过对预测残差进行变换和量化消除图像内的视觉冗余。最后，运动矢量、预测模式、量化参数和变换系数用熵编码进行压缩。

0.4.1 预测技术

帧内预测不需要参考其他图像，采用帧内预测编码的图像可作为编码后序列的随机访问点。

帧间预测需要参考先前已解码的图像，解码的顺序可与编码器中源图像捕获处理的顺序或从解码器输出用于显示的顺序不同。帧间预测中运动矢量的精度能达到1/4像素，运动矢量采用预测编码。

0.4.2 图像分块

本部分中视频解码过程的基本处理单元是宏块。一个宏块包括一个16×16的亮度样值块和对应的色度样值块。宏块可进一步划分到最小8×8的样本块来进行预测。

0.4.3 变换和量化

变换的单元是8×8的样本块。变换系数进行标量量化。

0.5 如何阅读本部分

建议读者从第1章（范围）开始，然后转到第3章（术语和定义）。第6章定义了编码比特流结构。第7章（语法和语义）定义了比特流的语法和语义，7.1是语法描述，定义了比特流中语法元素出现顺序，7.2是语义描述，也即语法元素的范围、限制和条件。第8章定义了语法元素的解析过程。最后，第9章

(解码过程)定义了语法元素如何映射到解码样值。在阅读本部分的过程中,读者还可参考第2章(规范性引用文件)、第4章(缩略语)、第5章(约定)及附录。

0.6 相关专利情况说明

本部分的发布机构提请注意如下事实,声明符合本部分时,可以使用涉及8.2, 9.3, 9.4.1, 9.4.3, 9.4.4.1, 9.5.1, 9.5.2, 9.6.2, 9.7, 9.8.2, 9.9.1, 9.9.2.1, 9.11条和附录C中有关内容的相关专利。

本部分的发布机构对于专利的范围、有效性和验证资料不提出任何看法。

专利持有人已向本部分的发布机构保证,他愿意同任何申请人在合理和非歧视的条款和条件下,就使用授权许可证进行谈判。这方面,该专利持有人的声明已在本部分的发布机构备案。

在本部分起草过程中,起草组织者数字音视频编解码技术标准工作组根据会员签署同意的工作组章程和有关知识产权规定以及会员在提案、审阅等期间提出的专利披露与许可声明等对标准可能涉及的专利进行了识别。已经确知下表列出的专利权人持有本标准的本部分的专利:

专利持有人	联系地址
中国科学院计算技术研究所	北京市海淀区中关村科学院南路6号(100080)
浙江大学	浙江省杭州市浙江大学信息与通信工程研究所(310027)
华中科技大学	湖北省武汉市洪山区珞瑜路1037号电子与信息工程系(430074)
清华大学	北京市海淀区清华大学电子工程系(100084)
北京工业大学	北京市朝阳区平乐园100号计算机学院(100022)
华为技术有限公司	广东省深圳市龙岗区坂田华为基地多媒体业务部(518057)
上海广电(集团)有限公司	上海市斜土路1646号上广电中央研究院(200233)

上述专利权人同意对所持有的本标准的本部分的必要专利在合理和非歧视的条款和条件基础上,通过AVS专利池进行许可。由数字音视频编解码技术标准工作组推动成立的AVS专利池管理委员会是决定专利池具体许可条款的独立机构。对于专利池中的所有专利,标准实施者可通过专利池管理委员会认可的授权机构获得许可。有关资料可从数字音视频编解码技术标准工作组秘书处获得,联系方法如下:

联系人:黄铁军(数字音视频编解码技术标准工作组秘书长)

通讯地址:北京2704信箱31分箱

邮政编码:100080

电子邮件:tjhuang@ict.ac.cn

电话:+10-58858303, +10-58858300-303

传真:+10-58858301

网址:<http://www.avs.org.cn>

请注意除上述已经识别出的专利外,本部分的某些内容有可能涉及专利。本部分的发布机构不应承担识别这些专利的责任。

信息技术 先进音视频编码 第2部分:视频

1 范围

GB/T 20090规定了数字音视频的压缩、解压缩、处理和表示的技术方案,适用于高分辨率和标准分辨率数字电视广播、激光数字存储媒体、互联网宽带流媒体、多媒体通信等应用。

GB/T 20090.2规定了多种比特率、分辨率和质量的视频压缩方法,适用于数字电视广播、交互式存储媒体、直播卫星视频业务、多媒体邮件、分组网络的多媒体业务、实时通信业务、远程视频监控等应用,并且规定了解码过程。

2 规范性引用文件

下列文件中的条款通过GB/T 20090的本部分的引用而成为本部分的条款。凡是注日期的引用文件,其随后所有的修改单(不包括勘误的内容)或修订版均不适用于本部分,然而,鼓励根据本部分达成协议的各方研究是否可使用这些文件的最新版本。凡是不注日期的引用文件,其最新版本适用于本部分。

GB/T 14857-1993 演播室数字电视编码参数规范(eqv CCIR 601-3)

3 术语和定义

下列术语和定义适用于GB/T 20090的本部分。

3.1

保留 reserved

定义了一些特定语法元素值用于将来对本部分的扩展。

注:这些值不应出现在符合本部分的比特流中。

3.2

比特串 bit string

有限位比特的有序序列,其最左边比特是最高有效位(MSB),最右边比特是最低有效位(LSB)。

3.3

比特流 bitstream

编码图像所形成的二进制数据流。

3.4

比特流缓冲区 bitstream buffer

存储比特流的缓冲区。

3.5

比特流顺序 bitstream order

编码图像在比特流中的排列顺序,与图像解码的顺序相同。

3.6

变长编码 variable length coding

一个可逆的熵编码过程,它将短的码字分配给出现频率较高的符号,将长的码字分配给出现频率较低的符号。

3.7

变换系数 transform coefficient

变换域上的一个标量。

- 3.8
编码表示 encoding presentation
数据编码后的形式。
- 3.9
编码过程 encoding process
产生符合本部分比特流的过程。
注：本部分不规定该过程。
- 3.10
编码器 encoder
编码过程的实现。
- 3.11
编码图像 coded picture
一帧图像的编码表示。
- 3.12
标志 flag
一个二值变量。
- 3.13
补偿 compensation
求由语法元素解码得到的样本残差与其对应的预测值之和。
- 3.14
残差 residual
样本或数据元素的重建值与其预测值之差。
- 3.15
参考索引 reference index
解码图像缓冲区中参考图像或其中场的编号。
- 3.16
参考图像 reference picture
解码过程中用于后续图像帧间预测的图像。
- 3.17
层 layer
比特流中的分级结构，高层包含低层。编码层由高到低依次为：序列、图像、条带、宏块和块。
- 3.18
场 field
由构成帧的三个样本矩阵中相间的行构成。
- 3.19
档次 profile
本部分规定的语法、语义及算法的子集。
- 3.20
非参考图像 non-reference picture
解码过程中不用于后续图像帧间预测的图像。
- 3.21
分量 component
图像的三个样值矩阵（亮度和两个色度）中的一个矩阵或矩阵中的单个样值。
- 3.22

反变换 inverse transform

将变换系数矩阵转换成空域样值矩阵的过程。

3.23**反量化 dequantization**

对量化系数缩放后得到变换系数的过程。

3.24**块 block**

一个 $M \times N$ 的样值矩阵或者变换系数矩阵（ M 列 N 行）。

3.25**块扫描 block scan**

量化系数的特定串行排序方式。

3.26**亮度 luma**

表示亮度信号的样值矩阵或单个样值。

注：亮度的符号是 Y 。

3.27**量化参数 quantization parameter**

在解码过程对量化系数进行反量化的参数。

3.28**量化系数 quantization coefficient**

反量化前变换系数的值。

3.29**光栅扫描 raster scan**

将二维矩形光栅映射到一维光栅，一维光栅的入口从二维光栅的第一行开始，然后接着扫描第二行、第三行，依次类推。光栅中的行从左到右扫描。

3.30**宏块 macroblock**

包括一个 16×16 的亮度样值块和对应的色度样值块。

3.31**宏块地址 macroblock address**

从图像左上角的宏块开始，沿光栅扫描的顺序编号，起始号为0。

3.32**宏块行 macroblock line**

从编码的图像左边界到右边界在相同的垂直位置连续的宏块，宏块行的高度是16个样本。

3.33**宏块位置 macroblock position**

图像中一个宏块的二维坐标，表示为 (x, y) 。如果当前图像的两场的编码数据交融出现，图像左上角的宏块 $(x, y) = (0, 0)$ ，对每个宏块列，从左到右 x 依次加1，对每个宏块行，从上到下 y 依次加1。如果当前图像的两场的编码数据依次出现，顶场左上角的宏块 $(x, y) = (0, 0)$ ，对顶场的每个宏块列，从左到右 x 依次加1，对顶场的每个宏块行，从上到下 y 依次加1；底场左上角的宏块 $(x, y) = (0, (H+31) \gg 5)$ ， H 是图像垂直方向扫描行数，对底场的每个宏块列，从左到右 x 依次加1，对底场的每个宏块行，从上到下 y 依次加1。

3.34**后向预测 backward prediction**

用显示顺序上将来的参考图像对当前图像进行预测。

3.35

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。

3.36

级别 level

在某一档次下对语法元素和语法元素参数值的限定集合。

3.37

交流系数 AC coefficient

AC系数

二维变换域上索引号不全为0的变换系数。

3.38

解码处理 decode processing

包括解析过程和解码过程。

3.39

解码过程 decoding process

由语法元素产生解码图像的过程。

3.40

解码器 decoder

完成解码处理的实体。

3.41

解码顺序 decoding order

解码过程根据图像之间的预测关系，对每帧图像解码的顺序。

3.42

解码图像 decoded picture

解码器根据比特流重建的图像。

3.43

解码图像缓冲区 decoded picture buffer

保存解码图像并用于预测、输出重排序和输出定时的缓冲区。

3.44

解析过程 parse

由比特流获得语法元素的过程。

3.45

禁止 forbidden

定义了一些特定语法元素值，这些值不应出现在符合本部分的比特流中。禁止某些值的目的是为了在比特流中出现伪起始码。

3.46

某档次解码器 x-profile decoder

能够解码符合某档次规定的比特流的解码器。

3.47

起始码 start code

长度为32比特的码字，其形式在整个比特流中是唯一的。起始码有多种用途，其中之一是用来标识比特流语法结构的开始。

3.48

- 前向预测 forward prediction**
用显示顺序上过去的参考图像对当前图像进行预测。
- 3.49 前向帧间解码图像 forward inter decoded picture**
P帧
帧间预测中只使用前向预测解码的图像。
- 3.50 色度 chroma**
两种色差信号的一种的样值矩阵或单个样值。
注：色度的符号是Cr和Cb。
- 3.51 视频序列 sequence**
编码比特流的最高层语法结构，包括一个或多个连续的编码图像。
- 3.52 输出重排序延迟 output reorder delay**
解码比特流中一帧图像到输出该解码图像之间的延迟。这是由于图像的显示顺序和解码顺序不同造成的。
- 3.53 输出处理过程 output processing**
由解码图像得到输出帧或场的过程。
- 3.54 输出顺序 output order**
输出解码图像的顺序，与显示顺序相同。
- 3.55 双向预测 bidirectional prediction**
用显示顺序上过去和将来的参考图像对当前图像进行预测。
- 3.56 双向帧间解码图像 bidirectional inter decoded picture**
B帧
帧间预测中使用双向预测解码的图像。
- 3.57 随机访问 random access**
从某一点而非比特流起始点开始对比特流解码并恢复出解码图像的能力。
- 3.58 随机访问点 random access point**
比特流中能进行随机访问的点。
- 3.59 填充比特 stuffing bits**
编码时插入比特流中的比特串，在解码时应被丢弃。
- 3.60 条带 slice**
按光栅扫描顺序的若干连续宏块行。
- 3.61 条带头 slice header**

编码的条带的一部分，是条带中宏块公用数据元素的编码表示。

3.62

跳过的宏块 **skipped macroblock**

除‘跳过’指示外，无其他编码数据的宏块。

3.63

图像重排序 **picture reordering**

如果解码顺序和输出顺序不同，解码图像进行重排序的过程。

3.64

显示顺序 **display order**

显示解码图像的顺序。

3.65

样本 **sample**

构成图像的基本元素。

3.66

样本宽高比 **width height ratio**

一帧图像中亮度样本列间的水平距离与行间的垂直距离之比。

表示为 $h:v$ ，其中 h 是水平宽度， v 是垂直高度。

3.67

样值 **sample value**

样本的幅值。

3.68

游程 **run**

在解码过程中若干连续的相同数据元素。一方面指在块扫描中一个非0系数前值为0的系数的个数；另一方面指跳过的宏块的数目。

3.69

预测 **prediction**

预测过程的具体实现。

3.70

预测过程 **prediction process**

使用预测器对当前解码样值或者数据元素进行估计。

3.71

预测值 **prediction value**

在后续样值或数据元素的解码过程中，使用的先前解码的样值或数据元素的组合。

3.72

语法元素 **syntax element**

比特流中的数据单元解析后的结果。

3.73

源 **source**

描述编码前视频素材或其某些属性的术语。

3.74

运动矢量 **motion vector**

用于帧间预测的二维矢量，由当前图像指向参考图像，其值为当前块和参考块在图像中坐标的偏移。

3.75

直流系数 **DC coefficient**

DC系数

二维变换域上索引号全为0的变换系数。

3.76**帧 frame**

视频信号空间信息的表示，由一个亮度样本矩阵（Y）和两个色度样本矩阵（Cb和Cr）构成。

3.77**帧间编码 inter coding**

使用帧间预测对宏块或图像进行编码。

3.78**帧间预测 inter prediction**

使用先前解码图像（或场）生成当前图像（或场）样本预测值的过程。

3.79**帧内编码 intra coding**

使用帧内预测对宏块或图像进行编码。

3.80**帧内解码图像 intra decoded picture****I帧**

只使用帧内预测解码的图像。如果I帧采用场编码，则第一场只使用帧内预测编码。

3.81**帧内预测 intra prediction**

在相同解码图像（或场）中使用先前解码的样值生成当前样本预测值的过程。

3.82**字节 byte**

8位的比特串。

3.83**字节对齐 byte alignment**

从比特流的第一个比特开始，如果某比特的位置是8的整数倍，则该比特是字节对齐的。

4 缩略语

BBV: 比特流参考解码器 (Bitstream Buffer Verifier)

CBR: 恒定比特率 (Constant Bit Rate)

CIF: 通用中间格式 (Common Intermediate Format)

LSB: 最低有效位 (Least Significant Bit)

MB: 宏块 (Macroblock)

MSB: 最高有效位 (Most Significant Bit)

QCIF: 四分之一通用中间格式 (Quarter Common Intermediate Format)

VBR: 可变比特率 (Variable Bit Rate)

VLC: 变长编码 (Variable Length Coding)

5 约定

GB/T 20090的本部分中使用的数学运算符和优先级与C语言使用的类似。但对整型除法和算术移位操作进行了特定定义。除特别说明外，约定编号和计数从0开始。

5.1 算术运算符

算术运算符定义如下：

+	加法运算
-	减法运算（二元运算符）或取反（一元前缀运算符）
×	乘法运算
a^b	幂运算，表示 a 的 b 次幂。也可表示上标
/	整除运算，沿向0的取值方向截断。例如，7/4和-7/-4截断至1，-7/4和7/-4截断至-1
÷	除法运算，不做截断或四舍五入
$\frac{a}{b}$	除法运算，不做截断或四舍五入
$\sum_{i=a}^b f(i)$	自变量 i 取由 a 到 b （含 b ）的所有整数值时，函数 $f(i)$ 的累加和
$a \% b$	模运算， a 除以 b 的余数，其中 a 与 b 都是正整数

5.2 逻辑运算符

逻辑运算符定义如下：

$a \ \&\& \ b$	a 和 b 之间的与逻辑运算
$a \ \ \ \ b$	a 和 b 之间的或逻辑运算
!	逻辑非运算

5.3 关系运算符

关系运算符定义如下：

>	大于
>=	大于或等于
<	小于
<=	小于或等于
==	等于
!=	不等于

5.4 位运算符

位运算符定义如下：

&	与运算
	或运算
~	取反运算
$a \gg b$	将 a 以2的补码整数表示的形式向右移 b 位。仅当 b 取正数时定义此运算
$a \ll b$	将 a 以2的补码整数表示的形式向左移 b 位。仅当 b 取正数时定义此运算

5.5 赋值

赋值运算定义如下：

=	赋值运算符
++	递增， $x++$ 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
--	递减， $x--$ 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
+=	自加指定值，例如 $x += 3$ 相当于 $x = x + 3$ ， $x += (-3)$ 相当于 $x = x + (-3)$
-=	自减指定值，例如 $x -= 3$ 相当于 $x = x - 3$ ， $x -= (-3)$ 相当于 $x = x - (-3)$

5.6 数学函数

数学函数定义如下：

$$\text{Abs}(x) = \begin{cases} x & ; \ x \geq 0 \\ -x & ; \ x < 0 \end{cases} \quad (1)$$

$\text{Ceil}(x)$ 取不小于 x 的最小整数 (2)

$\text{Clip1}(x) = \text{Clip3}(0, 255, x)$ (3)

$$\text{Clip3}(a, b, c) = \begin{cases} a & ; c < a \\ b & ; c > b \\ c & ; \text{其它} \end{cases} \quad (4)$$

$\text{Floor}(x)$ 取不大于 x 的最大整数 (5)

$\text{Log}_2(x)$ 取以2为底的 x 的对数

$\text{Log}_{10}(x)$ 取以10为底的 x 的对数 (6)

$\text{Median}(x, y, z) = x + y + z - \text{Min}(x, \text{Min}(y, z)) - \text{Max}(x, \text{Max}(y, z))$ (7)

$$\text{Min}(x, y) = \begin{cases} x & ; x \leq y \\ y & ; x > y \end{cases} \quad (8)$$

$$\text{Max}(x, y) = \begin{cases} x & ; x \geq y \\ y & ; x < y \end{cases} \quad (9)$$

$\text{Round}(x) = \text{Sign}(x) \times \text{Floor}(\text{Abs}(x) + 0.5)$

$$\text{Sign}(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases} \quad (10)$$

5.7 比特流语法、解析过程和解码过程的描述

5.7.1 描述方法

比特流语法描述方法类似C语言。比特流的语法元素使用粗体字表示，每个语法元素通过名字（用下划线分割的英文字母组，所有字母都是小写）、语法和语义来描述。语法表和正文中语法元素的值用常规字体表示。

某些情况下，可在语法表中应用从语法元素导出的其他变量值，这样的变量在语法表或正文中用不带下划线的小写字母和大写字母混合命名。大写字母开头的变量用于解码当前以及相关的语法结构，也可用于解码后续的语法结构。小写字母开头的变量只在它们所在的小节内使用。

语法元素值的助记符和变量值的助记符与它们的值之间的关系在正文中说明。在某些情况下，二者等同使用。助记符由一个或多个使用下划线分隔的字母组表示，每个字母组以大写字母开始，也可包括多个大写字母。

比特串的长度是4的整数倍时，可使用十六进制符号表示。十六进制的前缀是‘0x’，例如‘0x1a’表示比特串‘0001 1010’。

条件语句中0表示FALSE，非0表示TRUE。

语法表描述了所有符合本部分的比特流语法的超集，附加的语法限制在相关小节中说明。

下面给出了描述语法的伪代码例子。当语法元素出现时，表示从比特流中读一个数据单元。

	描述符
/*语句是一个语法元素的描述符，或者说明语法元素的存在、类型和数值，下面给出两个例子。*/	
syntax_element	uc(v)
conditioning statement	
/*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。*/	

{	
statement	
statement	
...	
}	
/* “while” 语句测试condition是否为TRUE, 如果为TRUE, 则重复执行循环体, 直到condition不为TRUE。*/	
while (condition)	
statement	
/* “do ... while”语句先执行循环体一次, 然后测试condition是否为TRUE, 如果为TRUE, 则重复执行循环体, 直到condition不为TRUE。*/	
do	
statement	
while (condition)	
/* “if ... else” 语句首先测试condition, 如果为TRUE, 则执行primary语句, 否则执行alternative语句。如果alternative语句不需要执行, 结构的“else”部分和相关的alternative语句可忽略。*/	
if (condition)	
primary statement	
else	
alternative statement	
/* “for” 语句首先执行initial语句, 然后测试condition, 如果condition为TRUE, 则重复执行primary语句和subsequent语句直到condition不为TRUE。*/	
for (initial statement; condition; subsequent statement)	
primary statement	

解析过程和解码过程用文字和类似C语言的伪代码描述。

5.7.2 函数

以下函数用于语法描述。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要读取的下一个比特的位置。函数由函数名及左右圆括号内的参数构成。函数也可没有参数。

byte_aligned()

如果比特流的当前位置是字节对齐的, 返回TRUE, 否则返回FALSE。

next_bits(n)

返回比特流的随后 n 个比特, MSB在前, 不改变比特流指针。如果剩余的比特少于 n , 则返回0。

byte_aligned_next_bits(n)

如果比特流当前位置不是字节对齐的, 返回比特流当前位置的下一个字节开始的 n 个比特, MSB在前, 不改变比特流指针; 如果比特流当前位置是字节对齐的, 返回比特流随后的 n 个比特, MSB在前, 不改变比特流指针。如果剩余的比特少于 n , 则返回0。

next_start_code()

在比特流中寻找下一个起始码, 函数定义如下。

next_start_code() {	描述符
stuffing_bit	'1'
while (! byte_aligned())	

stuffing_bit	'0'
while (next_bits(24) != '0000 0000 0000 0000 0000 0001')	
stuffing_byte	'0000 0000'
}	

stuffing_byte 应出现图像头之后和第一个条带起始码之前。

is_end_of_slice()

在比特流中检测是否是条带的结束，函数定义见如下。

is_end_of_slice() {	描述符
if(byte_aligned()) {	
if(next_bits(32) == 0x80000001	
return TRUE; // 条带结束	
}	
else {	
if((byte_aligned_next_bits(24) == 0x000001) && is_stuffing_pattern())	
return TRUE; // 条带结束	
}	
return FALSE;	
}	

is_stuffing_pattern()

在比特流中检测当前字节中剩下的比特或在字节对齐时下一个字节是否是填充的比特，函数定义见如下。

is_stuffing_pattern() {	描述符
if(next_bits(8-n) == (1 << (7-n)) // n: 0~7, 为码流指针在当前字节的位置偏移, n为0时码流指针指向当前字节最高位	
return TRUE;	
else	
return FALSE;	
}	

read_bits(n)

返回比特流的随后 n 个比特，MSB在前，同时比特流指针前移 n 个比特。如果 n 等于0，则返回0，比特流指针不前移。

函数也用于解析过程和解码过程的描述。

5.7.3 描述符

以下描述符表示不同语法元素的解析过程。

b(8)

一个任意取值的字节。解析过程由函数read_bits(8)的返回值规定。

ce(v)

变长编码的语法元素。解析过程在8.3中定义。

f(n)

取特定值的连续 n 个比特。解析过程由函数read_bits(n)的返回值规定。

i(n)

n 位整数。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，该返回值用高位在前的2的补码表示。

me(v)

用指数哥伦布码编码的语法元素。解析过程在8.2中定义。

r(n)

连续 n 个‘0’。解析过程由函数read_bits(n)的返回值规定。

se(v)

有符号整数语法元素，用指数哥伦布码编码。解析过程在8.2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，该返回值用高位在前的二进制表示。

ue(v)

无符号整数语法元素，用指数哥伦布码编码。解析过程在8.2中定义。

5.7.4 保留、禁止和标记

本部分定义的比特流语法中，某些语法元素的值被标注为‘保留’(reserved)或‘禁止’(forbidden)。

‘保留’定义了一些特定语法元素值用于将来对本部分的扩展。这些值不应出现在符合本部分的比特流中。

‘禁止’定义了一些特定语法元素值，这些值不应出现在符合本部分的比特流中。

‘标记’(marker_bit)指该比特的值应为‘1’。

比特流中的‘保留位’(reserved_bits)表明保留了一些语法单元用于将来对本部分的扩展，解码处理应忽略这些比特。

6 编码比特流的结构

本章说明编码后比特流的结构及其层次关系和处理顺序。

6.1 视频序列

视频序列是比特流的最高层语法结构。视频序列由序列头开始，后面跟着一个或多个编码图像，每帧图像之前应有图像头。编码图像在比特流中按比特流顺序排列，比特流顺序应与解码顺序相同。解码顺序可与显示顺序不相同。序列结束码表明了一个视频序列的结束。

6.1.1 逐行和隔行视频序列

GB/T 20090的本部分支持两种序列：逐行序列和隔行序列。

帧由三个样本矩阵构成，包括一个亮度样本矩阵(Y)和两个色度样本矩阵(Cb和Cr)。样本矩阵元素的值为整数。Y、Cb和Cr三个分量与原始的(模拟)红、绿和蓝色信号之间的关系，包括原始信号的色度和转移特性等可在比特流中定义，这些信息不影响解码过程。

场由构成帧的三个样本矩阵中相间的行构成，即帧样本矩阵的第一行、第三行、第五行，依次类推，构成一个场，称为顶场；第二行、第四行、第六行，依次类推，构成另一个场，称为底场。

解码器的输出是一系列帧或场，两帧之间存在着一个帧时间间隔。对隔行序列而言，每帧图像的两场之间存在着一个场时间间隔。对逐行序列而言，每帧图像的两场之间时间间隔为0。

6.1.2 序列头

视频序列头由视频序列起始码开始，后面跟着一串编码图像数据。

序列头可在比特流中重复出现，称为重复序列头。使用重复序列头的主要目的是支持对视频序列的随机访问。

序列头后的第一个编码图像应是I帧。序列头后出现的第一个P帧只能参考该序列头后出现的图像。在对比特流进行编辑或随机访问的情况下，重复序列头之前的全部数据可被丢弃，这样得到的一个新的比特流仍应符合本部分。

6.2 图像

一幅图像是一帧，其编码数据由图像起始码开始，到序列起始码、序列结束码或下一个图像起始码结束。

在比特流中，隔行扫描图像的两场的编码数据可依次出现，也可交融出现。两场数据的解码和显示顺序在图像头中规定。

图像的解码处理包括解析过程和解码过程。

6.2.1 图像格式

6.2.1.1 4:2:0 格式

对于4:2:0格式，Cb和Cr矩阵水平和垂直方向的尺寸都只有Y矩阵的一半。

亮度和色度样本位置如图1。

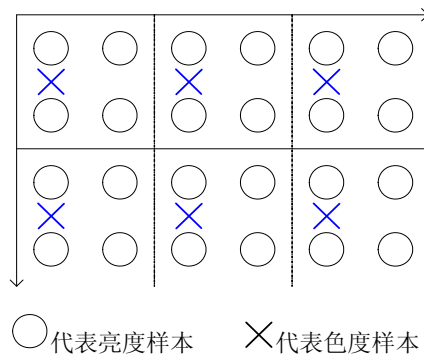


图1 4:2:0 格式下亮度和色度样本位置

6.2.1.2 4:2:2 格式

对于4:2:2格式，Cb和Cr矩阵在水平方向的尺寸只有Y矩阵的一半，在垂直方向的尺寸和Y相同。

亮度和色度样本位置如图2所示。

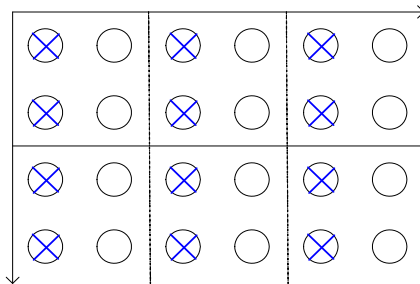


图2 4:2:2 格式下亮度和色度样本位置

6.2.1.3 4:4:4 格式

对于4:4:4格式，Cb和Cr矩阵在水平和垂直方向的尺寸都和Y矩阵一样。

亮度和色度样本位置如图3所示。

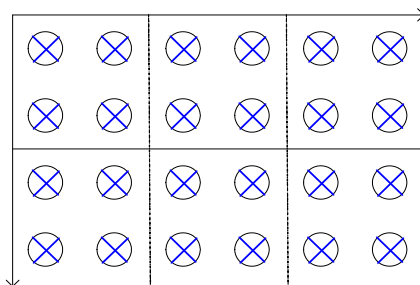


图3 4:4:4 格式下亮度和色度样本位置

6.2.2 图像类型

本部分定义了三种解码图像：

- 帧内解码图像（I 帧）；
- 前向帧间解码图像（P 帧）；
- 双向帧间解码图像（B 帧）。

6.2.3 图像间的顺序

如果视频序列中没有B帧，解码顺序与显示顺序相同。如果视频序列中包含B帧，解码顺序与显示顺序不同，解码图像输出显示前应进行图像重排序。图像重排序规则如下：

- 当前解码图像是 B 帧，输出由此 B 帧解码的图像；
- 当前解码图像是 I 帧或 P 帧，如果存在前一个 I 帧或 P 帧的解码图像，输出前一个解码图像。如果不存在前一个 I 帧或 P 帧的解码图像，不输出任何解码图像；
- 完成所有步骤后，如果缓冲区中还有未输出的解码图像，则输出该图像。

下面举例说明图像重排序：I 帧和 P 帧之间有两个 B 帧，两个连续的 P 帧之间也有两个 B 帧。用图像 1I 预测图像 4P，用图像 4P 和 1I 预测图像 2B 和 3B。解码顺序是 1I，4P，2B，3B；显示顺序是 1I，2B，3B，4P。

编码器输入顺序：

1	2	3	4	5	6	7	8	9	10	11	12	13
I	B	B	P	B	B	P	B	B	I	B	B	P

解码顺序：

1	4	2	3	7	5	6	10	8	9	13	11	12
I	P	B	B	P	B	B	I	B	B	P	B	B

解码器输出，即显示顺序：

1	2	3	4	5	6	7	8	9	10	11	12	13
I	B	B	P	B	B	P	B	B	I	B	B	P

6.2.4 参考图像

P 帧或 B 帧最多可有两帧参考图像。P 帧可参考前向的两帧。在一帧中，后解码的场还可参考当前帧的另外一场。B 帧可参考一前一后的两帧。

运动矢量所指的参考像素可超出参考图像的边界，在这种情况下对超出参考图像边界的整数样本应使用距离该整数参考样本所指位置最近的图像内的整数样本进行边界扩展。对亮度样本矩阵，参考块的像素在水平和垂直方向均不应超出参考图像边界外 16 个像素。对色度样本矩阵：

- 如果图像格式是 4:2:0，参考块的像素在水平和垂直方向均不应超出参考图像边界外 8 个像素。
- 如果图像格式是 4:2:2，参考块的像素在水平方向不应超出参考图像边界外 8 个像素，在垂直方向不应超出参考图像边界外 16 个像素。
- 如果图像格式是 4:4:4，参考块的像素在水平和垂直方向均不应超出参考图像边界外 16 个像素。

场边界扩展方法和参考图像边界扩展方法相同。

6.3 条带

条带是按光栅扫描顺序连续的若干宏块行，条带内的宏块行不应重叠，条带之间也不应重叠。条带内宏块的解码处理不应使用本图像其他条带的数据。

如果隔行图像的两场数据依次出现，这两场数据应属于不同的条带。
条带结构如图4所示。

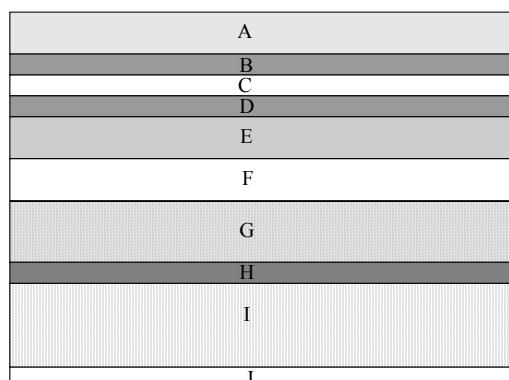


图4 条带结构

6.4 宏块

图像划分为宏块，宏块左上角的点不应超出图像边界。在比特流中，当隔行扫描图像的两场编码数据依次出现时，任一宏块的像素应来自同一场。

宏块的划分如图5所示，这种划分用于运动补偿。图5中矩形里的数字表示宏块划分后运动矢量和参考索引在码流中的顺序。

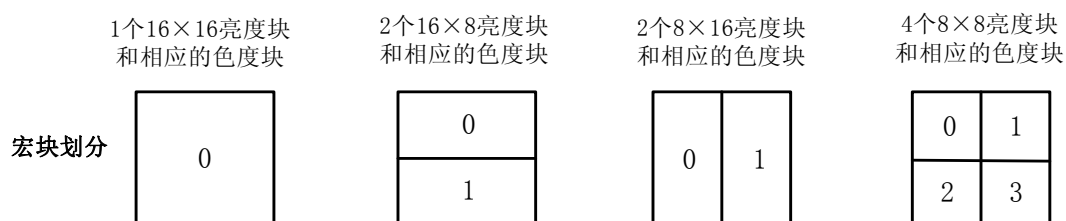


图5 宏块的划分

6.5 8×8 块

在4:2:0格式下，一个宏块包括4个8×8亮度块（Y）和2个8×8色度块（1个Cb，1个Cr）。如图6所示，图中数字为宏块中8×8块的顺序号。

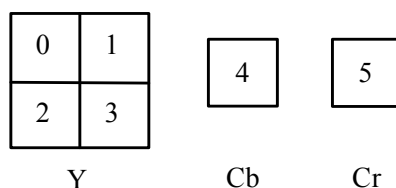


图6 宏块划分为8×8块（4:2:0格式）

在4:2:2格式下，一个宏块包括4个8×8亮度块（Y）和4个8×8色度块（2个Cb，2个Cr）。如图7所示，图中数字为宏块中8×8块的顺序号。

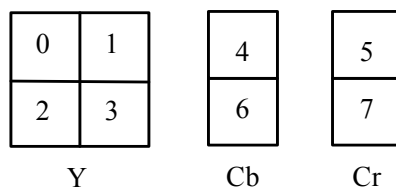


图7 宏块划分为8×8块（4:2:2格式）

在4:4:4格式下，一个宏块包括4个8×8亮度块（Y）和8个8×8色度块（4个Cb，4个Cr）。如图8所示，图中数字为宏块中8×8块的顺序号。

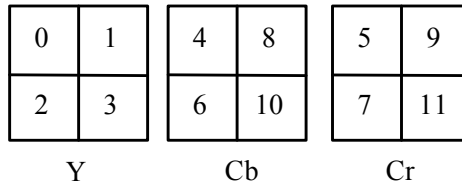


图8 宏块划分为8×8块（4:4:4格式）

宏块中的各个块在比特流中出现的顺序由图6到图8中的数字规定。

7 比特流的语法和语义

7.1 语法描述

7.1.1 起始码

起始码是一组特定的比特串。在符合GB/T 20090的本部分的比特流中，除起始码外的任何情况下都不应出现这些比特串。

起始码由起始码前缀和起始码值构成。起始码前缀是比特串‘0000 0000 0000 0000 0000 0001’。所有的起始码都应字节对齐。

起始码值是一个8比特整数，用来表示起始码的类型，见表1。

表1 起始码值

起始码类型	起始码值（十六进制）
条带起始码 (slice_start_code)	00~AF
视频序列起始码 (video_sequence_start_code)	B0
视频序列结束码 (video_sequence_end_code)	B1
用户数据起始码 (user_data_start_code)	B2
I图像起始码 (i_picture_start_code)	B3
保留	B4
视频扩展起始码 (extension_start_code)	B5
PB图像起始码 (pb_picture_start_code)	B6
视频编辑码 (video_edit_code)	B7
保留	B8
系统起始码	B9~FF

部分语法元素取特定值时可得到与起始码前缀相同的比特串，称为伪起始码。符合本部分的编码器和解码器应使用附录C定义的方法处理伪起始码问题。

7.1.2 视频序列定义

视频序列定义如下：

video_sequence() {	描述符
next_start_code()	
do {	
if (next_bits(32) == video_edit_code)	
video_edit_code	u(32)

sequence_header()	
extension_and_user_data(0)	
do {	
if (next_bits(32) == i_picture_start_code)	
i_picture_header()	
else	
pb_picture_header()	
extension_and_user_data(1)	
picture_data()	
} while ((next_bits(32) == pb_picture_start_code) (next_bits(32) == i_picture_start_code))	
} while (next_bits(32) != video_sequence_end_code)	
video_sequence_end_code	f(32)
}	

7.1.2.1 序列头定义

序列头定义如下：

sequence_header() {	描述符
video_sequence_start_code	f(32)
profile_id	u(8)
level_id	u(8)
progressive_sequence	u(1)
horizontal_size	u(14)
vertical_size	u(14)
chroma_format	u(2)
sample_precision	u(3)
aspect_ratio	u(4)
frame_rate_code	u(4)
bit_rate_lower	u(18)
marker_bit	f(1)
bit_rate_upper	u(12)
low_delay	u(1)
marker_bit	f(1)
bbv_buffer_size	u(18)
reserved_bits	r(3)
next_start_code()	
}	

7.1.2.2 扩展和用户数据定义

扩展和用户数据定义如下：

extension_and_user_data(i) {	描述符
while ((next_bits(32) == extension_start_code) (next_bits(32) == user_data_start_code)) {	
if (next_bits(32) == extension_start_code)	
extension_data(i)	
if (next_bits(32) == user_data_start_code)	

user_data()	
}	
}	

7.1.2.2.1 扩展数据定义

扩展数据定义如下：

extension_data(i) {	描述符
while (next_bits(32) == extension_start_code) {	
extension_start_code	f(32)
if (i == 0) { /* 序列头之后 */	
if (next_bits(4) == '0010') /* 序列显示扩展 */	
sequence_display_extension()	
else if (next_bits(4) == '0100') /* 版权扩展 */	
copyright_extension()	
else if (next_bits(4) == '1011') /* 摄像机参数扩展 */	
camera_parameters_extension()	
else {	
while (next_bits(24) != '0000 0000 0000 0000 0000 0001')	
reserved_extension_data_byte	u(8)
}	
}	
else { /* 图像头之后 */	
if (next_bits(4) == '0100') /* 版权扩展 */	
copyright_extension()	
else if (next_bits(4) == '0111') /* 图像显示扩展 */	
picture_display_extension()	
else if (next_bits(4) == '1011') /* 摄像机参数扩展 */	
camera_parameters_extension()	
else {	
while (next_bits(24) != '0000 0000 0000 0000 0000 0001')	
reserved_extension_data_byte	u(8)
}	
}	
}	

7.1.2.2.2 用户数据定义

用户数据定义如下：

user_data() {	描述符
user_data_start_code	f(32)
while (next_bits(24) != '0000 0000 0000 0000 0000 0001') {	
user_data	b(8)
}	
}	

7.1.2.3 序列显示扩展定义

序列显示扩展定义如下:

sequence_display_extension() {	描述符
extension_id	f(4)
video_format	u(3)
sample_range	u(1)
colour_description	u(1)
if (colour_description) {	
colour_primaries	u(8)
transfer_characteristics	u(8)
matrix_coefficients	u(8)
}	
display_horizontal_size	u(14)
marker_bit	f(1)
display_vertical_size	u(14)
reserved_bits	r(2)
next_start_code()	
}	

7.1.2.4 版权扩展定义

版权扩展定义如下:

copyright_extension() {	描述符
extension_id	f(4)
copyright_flag	u(1)
copyright_id	u(8)
original_or_copy	u(1)
reserved_bits	r(7)
marker_bit	f(1)
copyright_number_1	u(20)
marker_bit	f(1)
copyright_number_2	u(22)
marker_bit	f(1)
copyright_number_3	u(22)
next_start_code()	
}	

7.1.2.5 摄像机参数扩展定义

摄像机参数扩展定义如下:

camera_parameters_extension() {	描述符
extension_id	f(4)
reserved_bits	r(1)
camera_id	u(7)
marker_bit	f(1)
height_of_image_device	u(22)
marker_bit	f(1)

focal_length	u(22)
marker_bit	f(1)
f_number	u(22)
marker_bit	f(1)
vertical_angle_of_view	u(22)
marker_bit	f(1)
camera_position_x_upper	i(16)
marker_bit	f(1)
camera_position_x_lower	i(16)
marker_bit	f(1)
camera_position_y_upper	i(16)
marker_bit	f(1)
camera_position_y_lower	i(16)
marker_bit	f(1)
camera_position_z_upper	i(16)
marker_bit	f(1)
camera_position_z_lower	i(16)
marker_bit	f(1)
camera_direction_x	i(22)
marker_bit	f(1)
camera_direction_y	i(22)
marker_bit	f(1)
camera_direction_z	i(22)
marker_bit	f(1)
image_plane_vertical_x	i(22)
marker_bit	f(1)
image_plane_vertical_y	i(22)
marker_bit	f(1)
image_plane_vertical_z	i(22)
marker_bit	f(1)
reserved_bits	r(32)
next_start_code()	
}	

7.1.3 图像定义

7.1.3.1 I 图像头定义

I 图像头定义如下：

i_picture_header() {	描述符
i_picture_start_code	f(32)
bbv_delay	u(16)
time_code_flag	u(1)
if (time_code_flag == '1')	
time_code	u(24)

marker_bit	f(1)
picture_distance	u(8)
if (low_delay == '1')	
bbv_check_times	ue(v)
progressive_frame	u(1)
if (progressive_frame == '0')	
picture_structure	u(1)
top_field_first	u(1)
repeat_first_field	u(1)
fixed_picture_qp	u(1)
picture_qp	u(6)
if (progressive_frame == '0') {	
if (picture_structure == '0') {	
skip_mode_flag	u(1)
}	
}	
reserved_bits	r(4)
loop_filter_disable	u(1)
if (! loop_filter_disable) {	
loop_filter_parameter_flag	u(1)
if (loop_filter_parameter_flag) {	
alpha_c_offset	se(v)
beta_offset	se(v)
}	
}	
next_start_code()	
}	

7.1.3.2 PB 图像头定义

PB 图像头定义如下：

pb_picture_header() {	描述符
pb_picture_start_code	f(32)
bbv_delay	u(16)
picture_coding_type	u(2)
picture_distance	u(8)
if (low_delay == '1')	
bbv_check_times	ue(v)
progressive_frame	u(1)
if (progressive_frame == '0') {	
picture_structure	u(1)
if (picture_structure == '0')	
advanced_pred_mode_disable	u(1)
}	

top_field_first	u(1)
repeat_first_field	u(1)
fixed_picture_qp	u(1)
picture_qp	u(6)
if (! (picture_coding_type == '10' && PictureStructure == 1))	
picture_reference_flag	u(1)
no_forward_reference_flag	u(1)
reserved_bits	r(3)
skip_mode_flag	u(1)
loop_filter_disable	u(1)
if (! loop_filter_disable) {	
loop_filter_parameter_flag	u(1)
if (loop_filter_parameter_flag) {	
alpha_c_offset	se(v)
beta_offset	se(v)
}	
}	
next_start_code()	
}	

7.1.3.3 图像显示扩展定义

图像显示扩展定义如下：

picture_display_extension() {	描述符
extension_id	f(4)
for (i = 0; i < NumberOfFrameCentreOffsets; i ++) {	
frame_centre_horizontal_offset	i(16)
marker_bit	f(1)
frame_centre_vertical_offset	i(16)
marker_bit	f(1)
}	
next_start_code()	
}	

7.1.3.4 图像数据定义

图像数据定义如下：

picture_data() {	描述符
do {	
slice()	
} while (next_bits(32) == slice_start_code)	
next_start_code()	
}	

7.1.4 条带定义

条带定义如下：

slice() {	描述符
-----------	-----

slice_start_code	f(32)
if (vertical_size > 2800)	
slice_vertical_position_extension	u(3)
if (fixed_picture_qp == '0') {	
fixed_slice_qp	u(1)
slice_qp	u(6)
}	
if (PictureType != 0 (PictureStructure == 0 && MbIndex >= MbWidth × MbHeight / 2)) {	
slice_weighting_flag	u(1)
if (slice_weighting_flag == '1') {	
for (i=0; i<NumberOfReference; i++) {	
luma_scale	u(8)
luma_shift	i(8)
marker_bit	f(1)
chroma_scale	u(8)
chroma_shift	i(8)
marker_bit	f(1)
}	
mb_weighting_flag	u(1)
}	
}	
do {	
if (PictureType != 0 (PictureStructure == 0 && MbIndex >= MbWidth × MbHeight / 2)) {	
if (skip_mode_flag == '1')	
mb_skip_run	ue(v)
}	
if (MbIndex < MbWidth × MbHeight)	
macroblock()	
} while (! is_end_of_slice())	
next_start_code()	
}	

7.1.5 宏块定义

宏块定义如下：

macroblock() {	描述符
if (PictureType != 0 (PictureStructure == 0 && MbIndex >= MbWidth × MbHeight / 2))	
mb_type	ue(v)
if (MbType != 'P_Skip' && MbType != 'B_Skip') {	
if (MbType == 'B_8x8') {	
for (i=0; i<4; i++)	
mb_part_type	u(2)
}	
if (MbType == 'I_8x8') {	

for (i=0; i<4; i++) {	
pred_mode_flag	u(1)
if (! pred_mode_flag)	
intra_luma_pred_mode	u(2)
}	
intra_chroma_pred_mode	ue(v)
if (chroma_format == '10')	
intra_chroma_pred_mode_422	ue(v)
}	
if ((PictureType==1 (PictureType==2&&PictureStructure==0))&&picture_reference_flag=='0'){	
for (i = 0; i < MvNum; i++)	
mb_reference_index	u(1)/u(2)
}	
for (i = 0; i < MvNum; i++) {	
mv_diff_x	se(v)
mv_diff_y	se(v)
}	
if (MbWeightingFlag == 1)	
weighting_prediction	u(1)
if (! ((MbTypeIndex >= 24 && PictureType == 2) (MbTypeIndex >= 5 && PictureType != 2)))	
cbp	me(v)
if (chroma_format == '10')	
cbp_422	me(v)
if ((MbCBP > 0 (MbCBP422 > 0 && chroma_format == '10')) && ! FixedQP)	
mb_qp_delta	se(v)
for (i = 0; i < 6; i++)	
block(i)	
if (chroma_format == '10') {	
for (i = 6; i < 8; i++)	
block(i)	
}	
}	
}	

7.1.6 块定义

块定义如下：

block(i) {	描述符
if ((i < 6 && (MbCBP & (1 << i))) (i >= 6 && (MbCBP422 & (1 << (i - 6))))) {	
do {	
trans_coefficient	ce(v)
if (trans_coefficient >= 59)	
escape_level_diff	ce(v)
} while (trans_coefficient != 'EOB')	
}	

}	
---	--

7.2 语义描述

7.2.1 视频扩展

本部分定义了若干视频扩展，在语法的不同位置，可出现的视频扩展是不同的。每一种视频扩展都有一个唯一的视频扩展标号，见表2。

表2 视频扩展标号

视频扩展标号	含义
0000	保留
0001	保留
0010	序列显示扩展
0011	保留
0100	版权扩展
0101	保留
0110	保留
0111	图像显示扩展
1000 ~ 1010	保留
1011	摄像机参数扩展
1100 ~ 1111	保留

7.2.2 视频序列

视频编辑码 `video_edit_code`

比特串‘0x000001B7’。说明紧跟`video_edit_code`的第一个I帧后续的P帧或B帧可能缺少参考帧，不能正确解码。

视频序列结束码 `video_sequence_end_code`

比特串‘0x000001B1’。标识视频序列的结束。

7.2.2.1 序列头

视频序列起始码 `video_sequence_start_code`

比特串‘0x000001B0’。标识视频序列的开始。

档次标号 `profile_id`

8位无符号整数。表示比特流的档次。

级别标号 `level_id`

8位无符号整数。表示比特流的级别。

档次和级别见附录B。

逐行序列标志 `progressive_sequence`

标志。规定视频序列的扫描格式。值为‘1’表示编码视频序列只包含逐行扫描的帧图像；值为‘0’表示编码视频序列可包含逐行扫描和隔行扫描帧图像。

水平尺寸 `horizontal_size`

14位无符号整数。规定图像亮度分量可显示区域（该区域与图像的左侧边缘对齐）的宽度，即水平方向样本数。

以宏块为单位计算的显示区域宽度是

$$\text{MbWidth} = (\text{horizontal_size} + 15) / 16。$$

horizontal_size不应为0。horizontal_size的单位应是编码图像每行样本数。

垂直尺寸 vertical_size

14位无符号整数。规定图像亮度分量可显示区域（该区域与图像的顶部边缘对齐）的高度，即垂直方向扫描行数。

在视频序列比特流中，存在隔行扫描图像的两场编码数据依次出现时，以宏块为单位计算的显示区域高度是

$$\text{MbHeight} = 2 \times ((\text{vertical_size} + 31) / 32)$$

在其他情况下，以宏块为单位计算的显示区域高度是

$$\text{MbHeight} = (\text{vertical_size} + 15) / 16$$

vertical_size不应为0。vertical_size的单位应是编码图像的行数。

色度格式 chroma_format

2位无符号整数。规定色度分量的格式，见表3。

表3 色度格式

chroma_format	含义
00	保留
01	4:2:0
10	4:2:2
11	保留

样本精度 sample_precision

3位无符号整数。规定亮度和色度样本的精度，见表4。

表4 样本精度

sample_precision	含义
000	禁止
001	亮度和色度均为8 bit精度
010 ~ 111	保留

宽高比 aspect_ratio

4位无符号整数。规定重建图像的样本宽高比（SAR）或显示宽高比（DAR）。见表5。

表5 宽高比

aspect_ratio	样本宽高比（SAR）	显示宽高比（DAR）
0000	禁止	禁止
0001	1.0	-
0010	-	4 ÷ 3
0011	-	16 ÷ 9
0100	-	2.21 ÷ 1
0101 ~ 1111	-	保留

如果比特流中没有序列显示扩展，那么整个重建图像将要映射到整个活动显示区域。样本宽高比按下式计算。

$$\text{SAR} = \text{DAR} \times \text{vertical_size} \div \text{horizontal_size}$$

注：在这种情况下，horizontal_size和vertical_size受源图像的样本宽高比和选定的显示宽高比限制。
如果比特流中有序列显示扩展出现，样本宽高比按下式计算。

$$\text{SAR} = \text{DAR} \times \text{display_vertical_size} \div \text{display_horizontal_size}$$

帧率代码 **frame_rate_code**

4位无符号整数。规定帧率，见表6。

表 6 帧率代码

frame_rate_code	帧率
0000	禁止
0001	24000 ÷ 1001 (23.976...)
0010	24
0011	25
0100	30000 ÷ 1001 (29.97...)
0101	30
0110	50
0111	60000 ÷ 1001 (59.94...)
1000	60
1001 ~ 1111	保留

连续两帧之间的时间间隔是帧率的倒数。隔行扫描帧中两场之间的时间间隔是帧率的倒数的二分之一。

比特流速率低位 **bit_rate_lower**

BitRate的低18位。

比特流速率高位 **bit_rate_upper**

BitRate的高12位。

$$\text{BitRate} = (\text{bit_rate_upper} \ll 18) + \text{bit_rate_lower}$$

BitRate以400bit/s为单位计算视频比特流的比特率，并向上取整。BitRate不应为0。

低延迟 **low_delay**

标志。值为‘1’说明视频序列不包含B帧，不存在图像重排序延时，比特流中可能包含所谓“大图像”（见附录D）；值为‘0’说明视频序列可包含B帧，存在图像重排序延时，比特流中不包含所谓“大图像”（见附录D）。

比特流缓冲区尺寸 **bbv_buffer_size**

18位无符号整数。规定了比特流参考解码器对视频序列解码的比特流缓冲区尺寸（见附录D）。BBS是比特流参考解码器对视频序列解码所需的比特流缓冲区最小尺寸（按比特计算），其定义如下：

$$\text{BBS} = 16 \times 1024 \times \text{bbv_buffer_size}$$

7.2.2.2 扩展和用户数据

7.2.2.2.1 扩展数据

视频扩展起始码 **extension_start_code**

比特串‘0x000001B5’。标识视频扩展数据的开始。

视频扩展数据保留字节 **reserved_extension_data_byte**

8位无符号整数。保留位。解码器应丢弃这些数据。

7.2.2.2.2 用户数据

用户数据起始码 **user_data_start_code**

比特串‘0x000001B2’。标识用户数据的开始。用户数据连续存放，直到下一个起始码。

用户数据字节 user_data

8位整数。用户数据的含义由用户自行定义。用户数据中不应出现21个以上连续的‘0’。

7.2.2.3 序列显示扩展

本部分不定义显示过程。这一扩展中的信息对解码过程没有影响，解码器可忽略这些信息。

视频扩展标号 extension_id

比特串‘0010’。标识序列显示扩展。

视频格式 video_format

3位无符号整数。说明视频在按本部分进行编码之前的格式，见表7。如果比特流中没有出现序列显示扩展，可假设视频格式为“未作规定的视频格式”。

表7 视频格式

video_format	含义
000	分量信号
001	PAL
010	NTSC
011	SECAM
100	MAC
101	未作规定的视频格式
110	保留
111	保留

样值范围 sample_range

标志。说明亮度和色度信号样值的范围。如果比特流中没有出现序列显示扩展，设sample_range为‘0’。

彩色信息描述 colour_description

标志。值为‘1’说明比特流中有colour_primaries、transfer_characteristics和matrix_coefficients；值为‘0’说明比特流中没有colour_primaries、transfer_characteristics和matrix_coefficients。

彩色三基色 colour_primaries

8位无符号整数。说明源图像三基色的色度坐标，见表8。

表8 彩色三基色

colour_primaries	彩色三基色
0	禁止
1	ITU-R建议 BT. 709 基色 x y 绿 0.300 0.600 蓝 0.150 0.060 红 0.640 0.330 白 D65 0.3127 0.3290
2	未作规定的视频 图像特性未知

表 8 (续)

colour_primaries	彩色三基色
3	保留
4	ITU-R建议 BT. 470-2 System M 基色 x y 绿 0.21 0.71 蓝 0.14 0.08 红 0.67 0.33 白 C 0.310 0.316
5	ITU-R建议 BT. 470-2 System B, G 基色 x y 绿 0.29 0.60 蓝 0.15 0.06 红 0.64 0.33 白 D65 0.313 0.329
6	SMPTE 170M 基色 x y 绿 0.310 0.595 蓝 0.155 0.070 红 0.630 0.340 白 D65 0.3127 0.3290
7	SMPTE 240M (1987) 基色 x y 绿 0.310 0.595 蓝 0.155 0.070 红 0.630 0.340 白 D65 0.3127 0.3291
8	普通胶片 (彩色滤光镜, C光源) 基色 x y 绿 0.243 0.692 (Wratten 58) 蓝 0.145 0.049 (Wratten 47) 红 0.681 0.319 (Wratten 25)
9 ~ 255	保留

如果比特流中没有序列显示扩展, 或者colour_description的值是‘0’, 假设色度已由应用本身隐含定义。

光电转移特性 transfer_characteristics

8位无符号整数。说明源图像的光电转移特性, 见表9。

表 9 光电转移特性

transfer_characteristics	光电转移特性
0	禁止
1	ITU-R建议 BT. 709 $V = 1.099 Lc^{0.45} - 0.099, 1 \geq Lc \geq 0.018$ $V = 4.500 Lc, 0.018 > Lc \geq 0$
2	未作规定的视频 图像特性未知

表 9 (续)

transfer_characteristics	光电转移特性
3	保留
4	ITU-R建议 BT. 470-2 System M 假设显示伽玛值为2.2
5	ITU-R建议 BT. 470-2 System B, G 假设显示伽玛值为2.8
6	SMPTE 170M $V = 1.099 L_c^{0.45} - 0.099, 1 \geq L_c \geq 0.018$ $V = 4.500 L_c, 0.018 > L_c \geq 0$
7	SMPTE 240M (1987) $V = 1.1115 L_c^{0.45} - 0.1115, L_c \geq 0.0228$ $V = 4.0 L_c, 0.0228 > L_c$
8	线性转移特性 即 $V = L_c$
9	对数转移特性 (范围100:1) $V = 1.0 - \text{Log}_{10}(L_c)/2, 1 = L_c = 0.01$ $V = 0.0, 0.01 > L_c$
10	对数转移特性 (范围316.22777:1) $V = 1.0 - \text{Log}_{10}(L_c)/2.5, 1 = L_c = 0.0031622777$ $V = 0.0, 0.0031622777 > L_c$
11 ~ 255	保留

如果比特流中没有序列显示扩展，或者colour_description的值是‘0’，假设光电转移特性已由应用本身隐含定义。

彩色信号转换矩阵 **matrix_coefficients**

8位无符号整数。说明从红绿蓝三基色转换为亮度和色度信号时采用的转换矩阵，见表10。

表 10 彩色信号转换矩阵

matrix_coefficients	彩色信号转换矩阵
0	禁止
1	ITU-R建议 BT. 709 $E'_Y = 0.7154 E'_G + 0.0721 E'_B + 0.2125 E'_R$ $E'_{PB} = -0.386 E'_G + 0.500 E'_B - 0.115 E'_R$ $E'_{PR} = -0.454 E'_G - 0.046 E'_B + 0.500 E'_R$
2	未作规定的视频 图像特性未知
3	保留
4	FCC $E'_Y = 0.59 E'_G + 0.11 E'_B + 0.30 E'_R$ $E'_{PB} = -0.331 E'_G + 0.500 E'_B - 0.169 E'_R$ $E'_{PR} = -0.421 E'_G - 0.079 E'_B + 0.500 E'_R$
5	ITU-R建议 BT. 470-2 System B, G $E'_Y = 0.587 E'_G + 0.114 E'_B + 0.299 E'_R$ $E'_{PB} = -0.331 E'_G + 0.500 E'_B - 0.169 E'_R$ $E'_{PR} = -0.419 E'_G - 0.081 E'_B + 0.500 E'_R$

表 10 (续)

matrix_coefficients	彩色信号转换矩阵
6	SMPTE 170M $E'_Y = 0.587 E'_G + 0.114 E'_B + 0.299 E'_R$ $E'_{PB} = -0.331 E'_G + 0.500 E'_B - 0.169 E'_R$ $E'_{PR} = -0.419 E'_G - 0.081 E'_B + 0.500 E'_R$
7	SMPTE 240M (1987) $E'_Y = 0.701 E'_G + 0.087 E'_B + 0.212 E'_R$ $E'_{PB} = -0.384 E'_G + 0.500 E'_B - 0.116 E'_R$ $E'_{PR} = -0.445 E'_G - 0.055 E'_B + 0.500 E'_R$
8 ~ 255	保留

在表10中

- E'_Y 是值在 0 和 1 之间的模拟量；
- E'_{PB} 和 E'_{PR} 是值在-0.5 和 0.5 之间的模拟量；
- E'_R 、 E'_G 和 E'_B 是值在 0 和 1 之间的模拟量；
- Y 、 C_b 和 C_r 与 E'_Y 、 E'_{PB} 和 E'_{PR} 的关系如下：

如果sample_range为 '0'：

$$Y = (219 \times 2^{n-8} \times E'_Y) + 2^{n-4}$$

$$C_b = (224 \times 2^{n-8} \times E'_{PB}) + 2^{n-1}$$

$$C_r = (224 \times 2^{n-8} \times E'_{PR}) + 2^{n-1}$$

如果sample_range为 '1'：

$$Y = ((2^n - 1) \times E'_Y)$$

$$C_b = ((2^n - 1) \times E'_{PB}) + 2^{n-1}$$

$$C_r = ((2^n - 1) \times E'_{PR}) + 2^{n-1}$$

其中n是样本点精度。例如：

n = 8, sample_range为 '0' 时：

$$Y = (219 \times E'_Y) + 16$$

$$C_b = (224 \times E'_{PB}) + 128$$

$$C_r = (224 \times E'_{PR}) + 128$$

Y 的取值范围是16~235, C_b 和 C_r 的取值范围是16~240。

n = 8, sample_range为 '1' 时：

$$Y = (255 \times E'_Y)$$

$$C_b = (255 \times E'_{PB}) + 128$$

$$C_r = (255 \times E'_{PR}) + 128$$

Y 、 C_b 和 C_r 的取值范围都是0~255。

注1：本部分规定的解码过程将输出的 Y 、 C_b 和 C_r 的样值范围限制在0~255。如果比特流中没有出现序列显示扩展，或者colour_description的值是 '0'，假设转换矩阵已由应用本身隐含定义。

注2：某些应用可能有多个不同的视频信号，而不同的视频信号又可能具有不同的彩色三基色、转移特性和/或转换矩阵。在这种情况下建议应用首先将这些不同的参数集转换到一个统一的参数集。

水平显示尺寸 **display_horizontal_size**

垂直显示尺寸 **display_vertical_size**

display_horizontal_size和**display_vertical_size**都是14位无符号整数。它们共同定义了一个矩形，如果该矩形的尺寸比编码图像的尺寸小，宜只显示编码图像的一部分；如果该矩形的尺寸比编码图像的尺寸大，宜只在显示设备的一部分上显示重建图像。

display_horizontal_size的单位应是编码图像每行样本数。display_vertical_size的单位应是编码图像的行数。

display_horizontal_size和display_vertical_size对解码过程没有影响。它们可被显示过程使用。本部分不定义显示过程。

7.2.2.4 版权扩展

视频扩展标号 extension_id

比特串‘0100’。标识版权扩展。

版权标志 copyright_flag

标志。值为‘1’说明该版权扩展定义的版权信息有效期直到下一个版权扩展或视频序列结束码；值为‘0’说明该版权扩展没有定义版权信息。

版权信息由copyright_id和CopyrightNumber进一步说明。

版权标号 copyright_id

8位无符号整数。版权所有者的代码，由版权注册机构统一分配。如果为0，说明没有相关版权信息。

如果copyright_id的值为0，CopyrightNumber应为0。

如果copyright_flag的值为0，copyright_id应为0。

原创或拷贝 original_or_copy

标志。值为‘1’说明源视频的内容是原创的；值为‘0’说明源视频的内容是拷贝的。

版权号1 copyright_number_1

20位无符号整数。CopyrightNumber的第44到第63位。

版权号2 copyright_number_2

22位无符号整数。CopyrightNumber的第22到第43位。

版权号3 copyright_number_3

22位无符号整数。CopyrightNumber的第0到第21位。

CopyrightNumber是64位无符号整数，定义如下：

$$\text{CopyrightNumber} = (\text{copyright_number_1} \ll 44) + (\text{copyright_number_2} \ll 22) + \text{copyright_number_3}$$

如果copyright_flag的值是‘1’，CopyrightNumber和该版权扩展说明的源视频内容一一对应，CopyrightNumber为0说明没有相关信息；如果copyright_flag的值是‘0’，CopyrightNumber也应为0。

7.2.2.5 摄像机参数扩展

视频扩展标号 extension_id

比特串‘1011’。标识摄像机参数扩展。

摄像机标号 camera_id

7位无符号整数。摄像机标识符。

图像设备高度 height_of_image_device

22位无符号整数。给出图像设备的高度，以0.001mm为单位，范围从0到4,194.303mm。

焦距 focal_length

22位无符号整数。给出摄像机的焦距，以0.001mm为单位，范围从0到4,194.303mm。

光圈 f_number

22位无符号整数。给出摄像机的光圈（光圈 = 焦距 ÷ 镜头的有效孔径），以0.001为单位，范围从0到4,194.303。

垂直视角 vertical_angle_of_view

22位无符号整数。给出由图像设备顶端和底端决定的垂直视角，以0.0001°为单位，范围从0°到180°。

摄像机坐标X高位, 摄像机坐标Y高位, 摄像机坐标Z高位 `camera_position_x_upper,`
`camera_position_y_upper, camera_position_z_upper`

分别表示CameraPositionX, CameraPositionY和CameraPositionZ的高16位。

摄像机坐标X低位, 摄像机坐标Y低位, 摄像机坐标Z低位 `camera_position_x_lower,`
`camera_position_y_lower, camera_position_z_lower`

分别表示CameraPositionX, CameraPositionY和CameraPositionZ的低16位。

CameraPositionX, CameraPositionY和CameraPositionZ是一组32位整数, 用2的补码表示。说明摄像机光学原点在由用户定义的全局坐标系中的坐标值, 每个坐标值都以0.001mm为单位, 范围从-2, 147, 483.648mm到2, 147, 483.647mm。

摄像机方向矢量X, 摄像机方向矢量Y, 摄像机方向矢量Z `camera_direction_x,`
`camera_direction_y, camera_direction_z`

一组22位整数, 用2的补码表示。说明摄像机的方向, 每个值范围从-2, 097, 152到2, 097, 151。摄像机的方向用从摄像机光学原点到摄像机前面位于摄像机光轴上的某点的矢量表示。

图像平面垂直矢量X, 图像平面垂直矢量Y, 图像平面垂直矢量Z `image_plane_vertical_x,`
`image_plane_vertical_y, image_plane_vertical_z`

一组22位整数, 用2的补码表示。说明摄像机向上的方向, 每个值的范围从-2, 097, 152到2, 097, 151。摄像机向上的方向用平行于设备的边缘, 方向从底到顶的矢量表示。

本条内容如图9和图10所示。

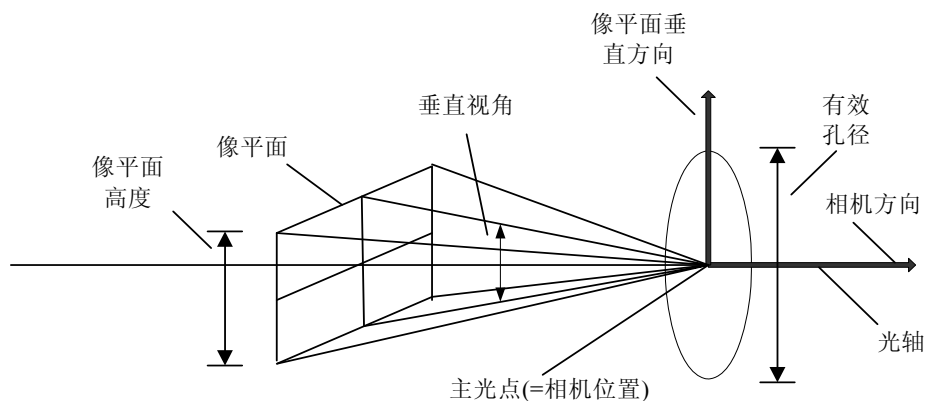


图9 摄像机原理示意图

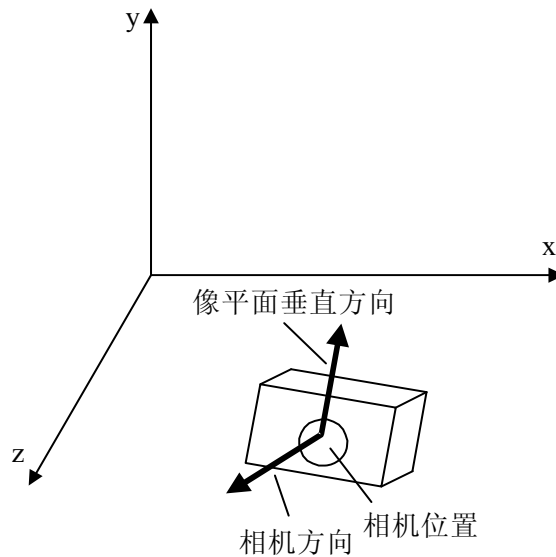


图10 摄像机坐标系示意图

7.2.3 图像

7.2.3.1 I 图像头

I图像起始码 `i_picture_start_code`

比特串 ‘0x000001B3’。I帧起始码，标识I帧的开始。

BBV延时 `bbv_delay`

16位无符号整数。如果`bbv_delay`不等于0xFFFF，它规定了BBV从收到图像起始码的最后一个字节到开始解码图像之间要等待的时间。这个时间用从27 MHz系统时钟导出的90 kHz时钟周期数来表示。如果视频序列中某一帧图像的`bbv_delay`等于0xFFFF，那么整个视频序列中所有图像的`bbv_delay`都必须等于0xFFFF。见附录D。

时间编码标志 `time_code_flag`

标志。值为 ‘1’ 表示比特流中包含`time_code`；值为 ‘0’ 表示比特流中没有`time_code`。

时间编码 `time_code`

24位比特串，包括以下字段：`DropFrameFlag`，`TimeCodeHours`，`TimeCodeMinutes`，`TimeCodeSeconds` 和 `TimeCodePictures`，见表11。其中`TimeCodeHours`，`TimeCodeMinutes`，`TimeCodeSeconds` 和 `TimeCodePictures`字段是以反码表示的无符号整数。这些参数与IEC 60461标准《视频磁带录像机的时间和控制编码》中定义的参数相对应。`time_code`描述从当前帧开始第一个`picture_distance`为0的帧的显示时间。

表 11 时间编码

<code>time_code</code> 的字段	取值	描述符
<code>DropFrameFlag</code>		u(1)
~ <code>TimeCodeHours</code>	0~23	u(5)
~ <code>TimeCodeMinutes</code>	0~59	u(6)
~ <code>TimeCodeSeconds</code>	0~59	u(6)
~ <code>TimeCodePictures</code>	0~59	u(6)

图像间距 `picture_distance`

8位无符号整数。`picture_distance`等于前一帧（显示顺序）的`picture_distance`加1，再加上当前帧和前一帧之间被跳过的图像帧数（连续两帧间被跳过的帧数应小于32，相邻的两个非双向帧间解码图

像间被跳过的帧数和B帧数之和应小于127)，最后模256。一个视频序列的第一帧的picture_distance应为0。

BBV检测次数 **bbv_check_times**

如果low_delay的值为‘0’，比特流中不应出现bbv_check_times，此时BbvCheckTimes等于0。如果比特流中出现bbv_check_times，由bbv_check_times解析得到BbvCheckTimes，解析过程见8.2。bbv_check_times的值应小于 $2^{16}-1$ 。

检测BBV缓冲区的次数为BbvCheckTimes加1次，BbvCheckTimes大于0说明当前图像是一个大图像(见附录D)。

逐行帧标志 **progressive_frame**

标志。值为‘0’表示该帧的两场是隔行场，这两场之间存在一个场时间间隔，在这种情况下，repeat_first_field的值应是‘0’。

如果progressive_frame的值是‘1’，表示该帧的两场实际上来自同一时刻，在这种情况下，PictureStructure的值应是‘1’。

图像编码结构标志 **picture_structure**

标志。picture_structre的值为‘0’表示当前图像的两场的编码数据依次出现；值为‘1’表示当前图像的两场的编码数据交融出现。如果progressive_sequence的值为‘1’，则picture_structre的值也应为‘1’。PictureStructure的值等于picture_structure。

顶场在先 **top_field_first**

标志。其含义由progressive_sequence、progressive_frame、picture_structure和repeat_first_field决定。

——如果progressive_sequence的值是‘0’，top_field_first说明解码场的输出显示顺序。

1) 如果PictureStructure的值是‘0’，top_field_first的值应是‘1’，说明顶场的编码数据首先出现在比特流中，顶场在底场之前输出显示。

2) 如果PictureStructure的值是‘1’，解码处理首先解码整帧。如果top_field_first的值是‘1’则顶场在底场之前输出，top_field_first的值是‘0’则底场在顶场之前输出。

——如果progressive_sequence的值是‘1’，top_field_first和repeat_first_field一起说明解码处理输出帧的次数(1次，2次或3次)。

1) 如果repeat_first_field的值是‘0’，top_field_first的值也应是‘0’，解码处理输出一个帧。

2) 如果top_field_first的值是‘0’，repeat_first_field的值是‘1’，解码处理输出两个完全一样的帧。

3) 如果top_field_first和repeat_first_field的值都是‘1’，解码处理输出三个完全一样的帧。

重复首场 **repeat_first_field**

标志。只在progressive_frame的值为‘1’时起作用；否则repeat_first_field的值应为‘0’。

——如果progressive_sequence和progressive_frame的值都是‘0’，repeat_first_field的值也应是‘0’，解码处理输出两个场，第一场(由top_field_first决定是顶场还是底场)，后面跟着第二场。

——如果progressive_sequence的值是‘0’，progressive_frame的值是‘1’，那么：

1) 如果repeat_first_field的值是‘0’，解码处理输出两个场，第一场(由top_field_first决定是顶场还是底场)，后面跟着第二场。

2) 如果repeat_first_field的值是‘1’，解码处理输出三个场，第一场(由top_field_first决定是顶场还是底场)，后面跟着第二场，最后重复输出第一场。

——如果progressive_sequence的值是‘1’，那么：

- 1) 如果 repeat_first_field 的值是 ‘0’，解码处理输出一帧。
- 2) 如果 repeat_first_field 的值是 ‘1’，解码处理输出两或三帧，由 top_field_first 决定。

固定图像量化因子 **fixed_picture_qp**

标志。值为 ‘1’ 说明在该帧图像内量化因子不变； ‘0’ 说明在该帧图像内量化因子可变。

图像量化因子 **picture_qp**

6位无符号整数。给出图像的量化因子。量化因子取值范围是0~63。

宏块跳过模式标志 **skip_mode_flag**

标志。值为 ‘1’ 表示宏块跳过模式使用游程编码；值为 ‘0’ 表示宏块跳过模式的编码由mb_type 决定，见9.4.1。

环路滤波禁用标志 **loop_filter_disable**

标志。值为 ‘1’ 表示不应使用环路滤波；值为 ‘0’ 表示应使用环路滤波。

环路滤波参数标志 **loop_filter_parameter_flag**

标志。值为 ‘1’ 表示比特流中包含alpha_c_offset和beta_offset；值为 ‘0’ 表示比特流中没有alpha_c_offset和beta_offset。

α 和C索引的偏移 **alpha_c_offset**

当前图像环路滤波 α 和C索引的偏移，alpha_c_offset取值范围是-8~8，环路滤波参数AlphaCOffset 等于alpha_c_offset。如果比特流中没有alpha_c_offset，AlphaCOffset等于0。

β 索引的偏移 **beta_offset**

当前图像环路滤波 β 索引的偏移，beta_offset取值范围是-8~8，环路滤波参数BetaOffset 等于beta_offset。如果比特流中没有beta_offset，BetaOffset等于0。

7.2.3.2 PB 图像头

PB图像起始码 **pb_picture_start_code**

比特串 ‘0x000001B6’。P帧或B帧起始码，标识P帧或B帧的开始。

图像编码方式 **picture_coding_type**

2位无符号整数。规定图像的编码方式，见表 12。

表 12 图像编码方式

picture_coding_type	编码方式
00	禁止
01	前向预测编码 (P)
10	双向预测编码 (B)
11	保留

高级预测模式禁用标志 **advanced_pred_mode_disable**

标志。值应为 ‘1’，说明禁止使用高级预测模式。值为 ‘0’ 保留。

图像参考标志 **picture_reference_flag**

标志。值为 ‘1’ 表示所有宏块都使用缺省参考图像；值为 ‘0’ 表示由每个宏块在宏块层自行确定参考图像。确定参考图像的方法见9.4.3。

无前向参考标志 **no_forward_reference_flag**

标志。值为 ‘1’ 表示当前图像不参考前向参考图像；值为 ‘0’ 表示当前图像可参考前向参考图像。

PB图像头的其他语法元素见7.2.3.1。

7.2.3.3 图像显示扩展

本部分不定义显示过程。这一扩展中的信息对解码处理没有影响，解码器可忽略这些信息。

图像显示扩展允许显示矩形（其尺寸由序列显示扩展定义）按图像移动。其中一项应用是实现全景扫描。

视频扩展标号 `extension_id`

比特串‘0111’。标识图像显示扩展。

帧中心水平偏移 `frame_centre_horizontal_offset`

16位整数。以1/16样本为单位给出水平偏移。正值表示重建图像的中心位置在显示矩形中心的右侧。

帧中心垂直偏移 `frame_centre_vertical_offset`

16位整数。以1/16样本为单位给出垂直偏移。正值表示重建图像的中心位置在显示矩形中心的下方。显示矩形区域的尺寸在序列显示扩展中定义。编码图像内区域的坐标由图像显示扩展定义。

重建图像的中心指由horizontal_size和vertical_size定义的矩形的中心。

在隔行序列中，一幅编码图像可能与一个、两个或三个解码场有关，因此图像显示扩展最多可以定义三组偏移量。

7.1.3.3中NumberOfFrameCentreOffsets的值按以下方式定义：

```
if ( progressive_sequence == '1' ) {
    if ( repeat_first_field == '1' ) {
        if ( top_field_first == '1' )
            NumberOfFrameCentreOffsets = 3
        else
            NumberOfFrameCentreOffsets = 2
    } else {
        NumberOfFrameCentreOffsets = 1
    }
} else {
    if ( picture_structure == '0' ) {
        NumberOfFrameCentreOffsets = 1
    } else {
        if ( repeat_first_field == '1' )
            NumberOfFrameCentreOffsets = 3
        else
            NumberOfFrameCentreOffsets = 2
    }
}
```

如果前面的序列头之后没有序列显示扩展，那么比特流中不应出现图像显示扩展。

如果一帧图像没有图像显示扩展，使用最近的解码图像的中心偏移量。注意丢失帧的中心偏移量的值与前一非丢失帧的值相同。在序列头后，所有图像的中心偏移量置为0，直到出现图像显示扩展。

利用图像中心偏移量，可定义一个矩形区域，这个区域在整个重建图像范围内移动，从而实现全景扫描。

图像中心偏移量参数见图11。

注1：显示矩形的尺寸可能大于重建图像。

注2：在场图像中，frame_centre_vertical_offset表示的中心偏移量以1/16帧行为单位。

注3：图11中，frame_centre_horizontal_offset和frame_centre_vertical_offset均为负值。

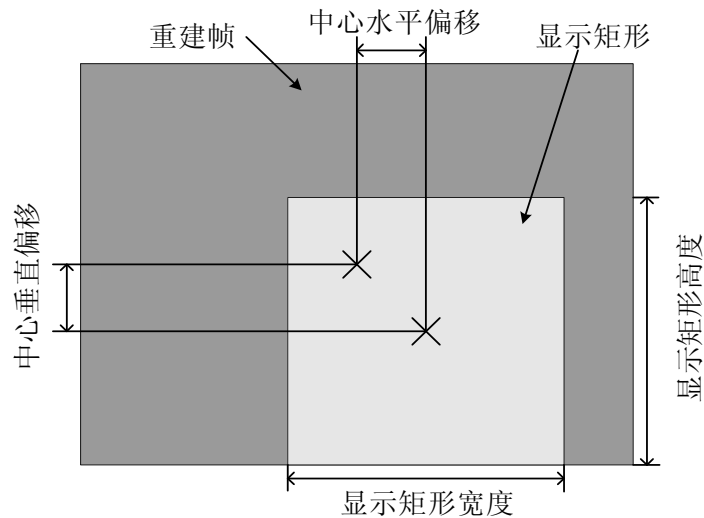


图11 帧中心偏移量参数

7.2.4 条带

条带起始码 `slice_start_code`

32位比特串，前24位是0x000001。后8位为`slice_vertical_position`，其范围是0x00~0xAF。

条带垂直位置 `slice_vertical_position`

8位无符号整数，给出条带的第一个宏块在图像中的垂直位置，以宏块为单位。`slice_vertical_position`的值的范围应为0x00~0xAF。

如果图像的`vertical_size`大于2800，条带垂直位置由`slice_vertical_position`和`slice_vertical_position_extension`决定。

条带垂直位置扩展 `slice_vertical_position_extension`

3位无符号整数，如果图像的`vertical_size`小于或等于2800，比特流中不应出现`slice_vertical_position_extension`。

MbRow按下面的方法计算：

```
if ( vertical_size > 2800 )
    MbRow = ( slice_vertical_position_extension << 7 ) + slice_vertical_position
else
    MbRow = slice_vertical_position
```

固定条带量化因子 `fixed_slice_qp`

标志。值为‘1’说明在该条带内量化因子不变；‘0’说明在该条带内量化因子可变。

条带量化因子 `slice_qp`

6位无符号整数。给出条带的量化因子。量化因子取值范围是0~63。

条带加权预测标志 `slice_weighting_flag`

标志。值为‘1’表示宏块的运动补偿可使用加权预测；值为‘0’表示宏块的运动补偿不应使用加权预测。

亮度缩放参数 `luma_scale`

8位无符号整数，表示预测亮度时的缩放参数。

亮度平移参数 `luma_shift`

8位有符号整数，表示预测亮度时的平移参数。

色度缩放参数 `chroma_scale`

8位无符号整数，表示预测色度时的缩放参数。

色度平移参数 `chroma_shift`

8位有符号整数，表示预测色度时的平移参数。

宏块加权预测标志 `mb_weighting_flag`

标志。`mb_weighting_flag`的值为‘0’表示所有非帧内预测宏块都应采用加权运动补偿；值为‘1’表示每个非帧内预测宏块由`WeightingPrediction`决定是否采用加权预测。

如果比特流中存在`mb_weighting_flag`，则`MbWeightingFlag`等于`mb_weighting_flag`；否则`MbWeightingFlag`等于0。

跳过宏块计数 `mb_skip_run`

跳过宏块计数。解析过程见8.2，解码过程见9.3。

7.2.5 宏块

宏块类型 `mb_type`

宏块的类型，其语义由图像类型、`PictureStructure`和`skip_mode_flag`决定。解析过程见8.2，解码过程见9.4.1。

宏块子类型 `mb_part_type`

2位无符号整数。宏块的子类型。解码过程见9.4.1。

预测模式标志 `pred_mode_flag`

标志。值为‘1’表示根据预测模式的预测值确定帧内亮度预测模式；值为‘0’表示根据`intra_luma_pred_mode`确定帧内亮度预测模式。

帧内亮度预测模式 `intra_luma_pred_mode`

2位无符号整数。用于确定亮度块的帧内预测模式，解码过程见9.4.2。

帧内色度预测模式 `intra_chroma_pred_mode`

用于确定宏块中序号为4、5的两个色度块的帧内预测模式，解析过程见8.2，解码过程见9.4.2。

4:2:2帧内色度预测模式 `intra_chroma_pred_mode_422`

用于确定4:2:2格式下宏块中序号为6、7的两个色度块的帧内预测模式，解析过程见8.2，解码过程见9.4.2。

宏块参考索引 `mb_reference_index`

如果`PictureStructure`的值为1或者`PictureType`的值为2，`mb_reference_index`是1位无符号整数。如果`PictureStructure`的值为0并且`PictureType`的值为1，`mb_reference_index`是2位无符号整数。先解码全部前向参考索引，然后解码全部后向参考索引，解码过程见9.4.3。

运动矢量水平分量差值 `mv_diff_x`

运动矢量垂直分量差值 `mv_diff_y`

运动矢量差值，单位为1/4样本，取值范围为-4096~4095（按亮度像素样本为-1024~1023.75），先解码全部前向运动矢量，然后解码全部后向运动矢量，解码过程见9.4.4。

加权预测 `weighting_prediction`

标志。如果比特流中存在`weighting_prediction`，则`WeightingPrediction`等于`weighting_prediction`；否则`WeightingPrediction`的值为‘0’。`WeightingPrediction`的值为‘0’表示该宏块不应采用加权预测；值为‘1’表示该宏块应采用加权预测。

宏块编码模板 `cbp`

`cbp`确定宏块中序号为0到5的亮度块和色度块是否包含编码数据。由`cbp`解析得到一个6位无符号整数`MbCBP`，解析过程见8.2。`MbCBP`的解码过程见9.4.5。

4:2:2宏块编码模板 `cbp_422`

`cbp_422`确定4:2:2格式下宏块中序号为6、7的两个色度块是否包含编码数据。由`cbp_422`解析得到一个2位无符号整数`MbCBP422`，解析过程见8.2。`MbCBP422`的解码过程见9.4.5。

宏块量化参数增量 `mb_qp_delta`

给出当前宏块的量化参数相对预测量化参数的增量，取值范围是-32~31。

7.2.6 块

变换系数 `trans_coefficient`

用于确定游程长度和量化非零系数值的联合索引，或用于确定转逸游程长度和转逸系数值的符号，解析过程见8.3，解码过程见9.5.1。

转逸系数差值 `escape_level_diff`

当`trans_coefficient`不能确定游程长度和量化非零系数值的联合索引时，`escape_level_diff`用于确定转逸系数的绝对值，解析过程见8.3，解码过程见9.5.1。

8 解析过程

本章定义语法元素的解析过程。语法元素描述符 `ue(v)`，`se(v)`和 `me(v)`的解析过程见 8.2；`ce(v)`的解析过程见 8.3。

8.1 k阶指数哥伦布码

解析k阶指数哥伦布码时，首先从比特流的当前位置开始寻找第一个非零比特，并将找到的零比特个数记为`leadingZeroBits`，然后根据`leadingZeroBits`计算`CodeNum`。用伪代码描述如下：

```

leadingZeroBits = -1;
for ( b = 0; ! b; leadingZeroBits++ )
    b = read_bits(1)
CodeNum = 2leadingZeroBits + k - 2k + read_bits(leadingZeroBits + k)
    
```

表13给出了0阶、1阶、2阶和3阶指数哥伦布码的结构。指数哥伦布码的比特串分为“前缀”和“后缀”两部分。前缀由`leadingZeroBits`个连续的‘0’和一个‘1’构成。后缀由`leadingZeroBits + k`个比特构成，即表中的 x_i 串， x_i 的值为‘0’或‘1’。

表 13 k阶指数哥伦布码表

阶数	码字结构	CodeNum取值范围
k = 0	1	0
	0 1 x_0	1~2
	0 0 1 $x_1 x_0$	3~6
	0 0 0 1 $x_2 x_1 x_0$	7~14

k = 1	1 x_0	0~1
	0 1 $x_1 x_0$	2~5
	0 0 1 $x_2 x_1 x_0$	6~13
	0 0 0 1 $x_3 x_2 x_1 x_0$	14~29

k = 2	1 $x_1 x_0$	0~3
	0 1 $x_2 x_1 x_0$	4~11
	0 0 1 $x_3 x_2 x_1 x_0$	12~27
	0 0 0 1 $x_4 x_3 x_2 x_1 x_0$	28~59

表 13 (续)

阶数	码字结构	CodeNum取值范围
k = 3	1 x ₂ x ₁ x ₀	0~7
	0 1 x ₃ x ₂ x ₁ x ₀	8~23
	0 0 1 x ₄ x ₃ x ₂ x ₁ x ₀	24~55
	0 0 0 1 x ₅ x ₄ x ₃ x ₂ x ₁ x ₀	56~119

8.2 ue(v), se(v)和 me(v)

ue(v), se(v)和me(v)描述的语法元素使用0阶指数哥伦布码, 其解析过程为:

——ue(v): 语法元素的值等于 CodeNum;

——se(v): 根据表 14 给出的有符号指数哥伦布码的映射关系求语法元素的值;

me(v): 分别根据

表 15 和

——表 16 求 MbCBP 和 MbCBP422 的值 (见 9.4.1 和 9.4.5)

表 14 se(v) 与 CodeNum 的映射关系

CodeNum	语法元素值
0	0
1	1
2	-1
3	2
4	-2
5	3
6	-3
k	$(-1)^{k+1} \times \text{Ceil}(k \div 2)$

表 15 MbCBP 与 CodeNum 的映射关系

CodeNum	MbCBP	
	xxxxxx (543210)	宏块帧内编码模式

0	63	0
1	15	15
2	31	63
3	47	31
4	0	16
5	14	32
6	13	47
7	11	13

表 15 (续)

CodeNum	MbCBP xxxxxx (543210)	
	宏块帧内编码模式	宏块帧间编码模式
8	7	14
9	5	11
10	10	12
11	8	5
12	12	10
13	61	7
14	4	48
15	55	3
16	1	2
17	2	8
18	59	4
19	3	1
20	62	61
21	9	55
22	6	59
23	29	62
24	45	29
25	51	27
26	23	23
27	39	19
28	27	30
29	46	28

30	53	9
31	30	6
32	43	60
33	37	21
34	60	44
35	16	26
36	21	51
37	28	35
38	19	18

表 15 (续)

CodeNum	MbCBP xxxxxx (543210)	
	宏块帧内编码模式	宏块帧间编码模式
39	35	20
40	42	24
41	26	53
42	44	17
43	32	37
44	58	39
45	24	45
46	20	58
47	17	43
48	18	42
49	48	46
50	22	36
51	33	33
52	25	34
53	49	40
54	40	52
55	36	49
56	34	50
57	50	56
58	52	25
59	54	22
60	41	54

61	56	57
62	38	41
63	57	38

表 16 MbCBP422 与 CodeNum 的映射关系

CodeNum	MbCBP422	
	xx (10)	
	宏块帧内编码模式	宏块帧间编码模式
0	0	0
1	1	1
2	2	2
3	3	3

8.3 ce(v)

ce(v)描述的语法元素采用0阶、1阶、2阶或3阶指数哥伦布码进行解析，其阶数由下面的规则决定：

- 帧内编码块亮度系数的 escape_level_diff 采用 1 阶指数哥伦布码。
- 帧间编码块亮度系数的 escape_level_diff 采用 0 阶指数哥伦布码。
- 色度系数的 escape_level_diff 采用 0 阶指数哥伦布码。
- GB/T 20090 的本部分定义了 19 个与 ce(v)有关的变长码表，即 VLC0_Intra, VLC1_Intra, VLC2_Intra, VLC3_Intra, VLC4_Intra, VLC5_Intra, VLC6_Intra, VLC0_Inter, VLC1_Inter, VLC2_Inter, VLC3_Inter, VLC4_Inter, VLC5_Inter, VLC6_Inter, VLC0_Chroma, VLC1_Chroma, VLC2_Chroma, VLC3_Chroma, VLC4_Chroma, 见附录 A。不同的码表决定了 ce(v)所用的指数哥伦布码的阶数。其中 VLC0_Inter 采用 3 阶指数哥伦布码，VLC2_Chroma 及 VLC3_Chroma 采用 1 阶指数哥伦布码，VLC1_Chroma 及 VLC4_Chroma 采用 0 阶指数哥伦布码，其他码表均采用 2 阶指数哥伦布码。

本条中上述19个码表的选择按照以下规则进行：

- 编码块第 1 个解码量化系数的码表选择：
 - 1) 帧内预测编码块的亮度系数，CurrentVLCTable = VLC0_Intra, 见表 A. 1。
 - 2) 帧间预测编码块的亮度系数，CurrentVLCTable = VLC0_Inter, 见表 A. 8。
 色度系数，CurrentVLCTable = VLC0_Chroma, 见表 A. 15。
 maxAbsLevel 等于 0。
 absLevel 等于第 1 个解码量化系数值的绝对值。
- 其他解码量化系数的码表选择：

- 1) 对于帧内预测编码块的亮度系数, 如果 absLevel 大于 maxAbsLevel , 按如下方式进行码表切换:
 - ◆ 如果 absLevel 等于 1, 选择 $\text{CurrentVLCTable} = \text{VLC1_Intra}$, 见表 A. 2。
 - ◆ 如果 absLevel 等于 2, 选择 $\text{CurrentVLCTable} = \text{VLC2_Intra}$, 见表 A. 3。
 - ◆ 如果 absLevel 等于 3 或 4, 选择 $\text{CurrentVLCTable} = \text{VLC3_Intra}$, 见表 A. 4。
 - ◆ 如果 absLevel 等于 5、6 或 7, 选择 $\text{CurrentVLCTable} = \text{VLC4_Intra}$, 见表 A. 5。
 - ◆ 如果 absLevel 等于 8、9 或 10, 选择 $\text{CurrentVLCTable} = \text{VLC5_Intra}$, 见表 A. 6。
 - ◆ 如果 absLevel 大于 10, 选择 $\text{CurrentVLCTable} = \text{VLC6_Intra}$, 见表 A. 7。
- 2) 对于帧间预测编码块的亮度系数, 如果 absLevel 大于 maxAbsLevel , 按如下方式进行码表切换:
 - ◆ 如果 absLevel 等于 1, 选择 $\text{CurrentVLCTable} = \text{VLC1_Inter}$, 见表 A. 9。
 - ◆ 如果 absLevel 等于 2, 选择 $\text{CurrentVLCTable} = \text{VLC2_Inter}$, 见表 A. 10。
 - ◆ 如果 absLevel 等于 3, 选择 $\text{CurrentVLCTable} = \text{VLC3_Inter}$, 见表 A. 11。
 - ◆ 如果 absLevel 等于 4、5 或 6, 选择 $\text{CurrentVLCTable} = \text{VLC4_Inter}$, 见表 A. 12。
 - ◆ 如果 absLevel 等于 7、8 或 9, 选择 $\text{CurrentVLCTable} = \text{VLC5_Inter}$, 见表 A. 13。
 - ◆ 如果 absLevel 大于 9, 选择 $\text{CurrentVLCTable} = \text{VLC6_Inter}$, 见表 A. 14。
- 3) 对于色度块的系数, 如果 absLevel 大于 maxAbsLevel , 按如下方式进行码表切换:
 - ◆ 如果 absLevel 等于 1, 选择 $\text{CurrentVLCTable} = \text{VLC1_Chroma}$, 见表 A. 16。
 - ◆ 如果 absLevel 等于 2, 选择 $\text{CurrentVLCTable} = \text{VLC2_Chroma}$, 见表 A. 17。
 - ◆ 如果 absLevel 等于 3 或 4, 选择 $\text{CurrentVLCTable} = \text{VLC3_Chroma}$, 见表 A. 18。
 - ◆ 如果 absLevel 大于 4, 选择 $\text{CurrentVLCTable} = \text{VLC4_Chroma}$, 见表 A. 19。
- 4) 如果 absLevel 大于 maxAbsLevel , 则 maxAbsLevel 等于 absLevel 。
- 5) absLevel 等于当前解码量化系数值的绝对值。

$\text{ce}(v)$ 描述的语法元素解析过程如下:

——语法元素 trans_coefficient 等于 CodeNum 。

——如果 trans_coefficient 大于或等于 59, 解析下一个 $\text{ce}(v)$ 语法元素, 得到一个新的 CodeNum , escape_level_diff 等于 CodeNum 。

9 解码过程

本章定义解码过程。

9.1 高层语法结构

如果 $\text{progressive_sequence}$ 的值为 ‘1’, 解码器以帧周期的整数倍为时间间隔输出重建图像。

如果 $\text{progressive_sequence}$ 的值为 ‘0’, 如果 progressive_frame 的值为 ‘1’, 则解码器以帧周期的整数倍为时间间隔输出重建图像; 否则, 解码器将重建图像拆成两场, 以场周期为时间间隔输出。

9.2 图像头解码

图像头解码过程如下:

如果当前图像起始码是 $0x00001B3$, 则表示图像是 I 帧, PictureType 等于 0。

如果当前图像起始码是 $0x00001B6$, 并且 picture_code_type 等于 ‘01’, 则表示图像是 P 帧, PictureType 等于 1。

如果当前图像起始码是 $0x00001B6$, 并且 picture_code_type 等于 ‘10’, 则表示图像是 B 帧, PictureType 等于 2。

预测量化参数 PreviousQP 初始化为 picture_qp 。固定量化因子标志 FixedQP 等于 fixed_picture_qp 。当前图像的宏块索引 MbIndex 初始化为 0。

9.3 条带解码

条带解码过程如下：

宏块索引MbIndex等于MbRow × MbWidth。

如果fixed_picture_qp等于‘0’，预测量化参数PreviousQP等于slice_qp。固定量化因子标志FixedQP等于fixed_slice_qp。

如果mb_skip_run存在，由mb_skip_run解析得到SkipMbCount，否则SkipMbCount等于0。若SkipMbCount不等于0，从MbIndex至MbIndex+SkipMbCount-1的所有宏块的MbType根据图像类型设为P_Skip或B_Skip，这些宏块按照所设置的宏块类型进行处理。处理完成后，MbIndex应加上SkipMbCount。

9.4 宏块解码

解码索引值为MbIndex的宏块，完成后MbIndex加1。

9.4.1 宏块类型

宏块类型MbType和宏块子类型MbPartType的解码过程如下：

——如果当前图像是I帧

- 1) 如果PictureStructure的值为1，则MbTypeIndex等于0，MbType等于‘I_8x8’，MvNum等于0。
- 2) 如果PictureStructure的值为0并且MbIndex < MbWidth × MbHeight / 2，则MbTypeIndex等于0，MbType等于‘I_8x8’，MvNum等于0。
- 3) 如果PictureStructure的值为0并且MbIndex ≥ MbWidth × MbHeight / 2
 - ◆ 如果skip_mode_flag的值为1，MbTypeIndex等于mb_type加1；否则MbTypeIndex等于mb_type。
 - 如果MbTypeIndex大于或等于5，则mb_type中包含的CBP信息CBPCodeNum等于MbTypeIndex减5，然后令MbTypeIndex等于5。MbType和MvNum的值见表17。

——如果当前图像是P帧

如果skip_mode_flag的值为1，MbTypeIndex等于mb_type加1；否则MbTypeIndex等于mb_type。如果MbTypeIndex大于或等于5，则mb_type中包含的CBP信息CBPCodeNum等于MbTypeIndex减5，然后令MbTypeIndex等于5。MbType和MvNum的值见表17。

——如果当前图像是B帧

如果skip_mode_flag的值为1，MbTypeIndex等于mb_type加1；否则MbTypeIndex等于mb_type。如果MbTypeIndex大于或等于24，则mb_type中包含的CBP信息CBPCodeNum等于MbTypeIndex减24，然后令MbTypeIndex等于24。

MbType和MvNum的值见表18。如果MbType等于‘B_8x8’，MbPartType和MbPartMvNum的值用mb_part_type查表19，MvNum是该宏块所有块的MbPartMvNum的和。

表 17 P 宏块类型

MbTypeIndex	MbType	MvNum	MbPredMode
0	P_Skip	0	前向
1	P_16x16	1	前向
2	P_16x8	2	前向
3	P_8x16	2	前向
4	P_8x8	4	前向

5	I_8x8	0	无
---	-------	---	---

表 18 B 宏块类型

MbTypeIndex	MbType	MvNum	MbPredMode
0	B_Skip	0	双向
1	B_Direct_16x16	0	双向
2	B_Fwd_16x16	1	前向
3	B_Bck_16x16	1	后向
4	B_Sym_16x16	1	双向
5	B_Fwd_Fwd_16x8	2	上前、下前
6	B_Fwd_Fwd_8x16	2	左前、右前

表 18 (续)

MbTypeIndex	MbType	MvNum	MbPredMode
7	B_Bck_Bck_16x8	2	上后、下后
8	B_Bck_Bck_8x16	2	左后、右后
9	B_Fwd_Bck_16x8	2	上前、下后
10	B_Fwd_Bck_8x16	2	左前、右后
11	B_Bck_Fwd_16x8	2	上后、下前
12	B_Bck_Fwd_8x16	2	左后、右前
13	B_Fwd_Sym_16x8	2	上前、下双
14	B_Fwd_Sym_8x16	2	左前、右双
15	B_Bck_Sym_16x8	2	上后、下双
16	B_Bck_Sym_8x16	2	左后、右双
17	B_Sym_Fwd_16x8	2	上双、下前
18	B_Sym_Fwd_8x16	2	左双、右前
19	B_Sym_Bck_16x8	2	上双、下后
20	B_Sym_Bck_8x16	2	左双、右后
21	B_Sym_Sym_16x8	2	上双、下双
22	B_Sym_Sym_8x16	2	左双、右双
23	B_8x8	0~4	前向、后向、双向
24	I_8x8	0	无

表 19 B_8x8 子模式

mb_part_type	MbPartType	MbPartMvNum	MbPartPredMode
0	SB_Direct_8x8	0	双向
1	SB_Fwd_8x8	1	前向
2	SB_Bck_8x8	1	后向
3	SB_Sym_8x8	1	双向

表17、表18和表19定义的MbType和MbPartType中，有Skip字样的称为跳过模式，有Direct字样的称为直接模式，有Sym字样的称为对称模式。表17到表19中的MvNum表示宏块在比特流中的运动矢量数。对

称模式是一种双向预测模式，此时比特流中只包含前向参考索引和前向运动矢量，对称模式的后向参考索引和后向运动矢量的推导见9.9.1。

9.4.2 帧内预测模式

当前宏块的每一个8×8块采用以下方法确定其预测模式：

——如果当前块是亮度块

1) 计算当前块预测模式的预测值

- ◆ 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 `intraPredModeA`；否则 `intraPredModeA` 等于-1。
- ◆ 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 `intraPredModeB`；否则 `intraPredModeB` 等于-1。
- ◆ 如果 `intraPredModeA` 或 `intraPredModeB` 等于-1，则 `predIntraPredMode` 等于 2；否则 `predIntraPredMode` 等于 $\text{Min}(\text{intraPredModeA}, \text{intraPredModeB})$ 。

2) 如果 `pred_mode_flag` 的值为‘1’，则预测模式 `IntraLumaPredMode` 等于 `predIntraPredMode`。

3) 如果 `pred_mode_flag` 的值为‘0’，并且 `intra_luma_pred_mode` 小于 `predIntraPredMode`，则 `IntraLumaPredMode` 等于 `intra_luma_pred_mode`；否则 `IntraLumaPredMode` 等于 `intra_luma_pred_mode` 加 1。

左边块或上边块‘存在’指该块应在图像内，并且该块应与当前块属于同一条带。

`IntraLumaPredMode`的值与亮度块预测模式的关系见表20。

表 20 8×8 亮度块帧内预测模式

IntraLumaPredMode	名称
0	Intra_8x8_Vertical
1	Intra_8x8_Horizontal
2	Intra_8x8_DC
3	Intra_8x8_Down_Left
4	Intra_8x8_Down_Right

——如果当前块是色度块，色度块的顺序号为 4、5 时预测模式 `IntraChromaPredMode` 等于 `intra_chroma_pred_mode`；色度块的顺序号为 6、7 时 `IntraChromaPredMode` 等于 `intra_chroma_pred_mode_422`。

`IntraChromaPredMode`的值与色度块预测模式的关系见表21。

表 21 8×8 色度块帧内预测模式

IntraChromaPredMode	名称
0	Intra_Chroma_DC
1	Intra_Chroma_Horizontal
2	Intra_Chroma_Vertical
3	Intra_Chroma_Plane

表20所示的8×8亮度块帧内预测模式见图12。

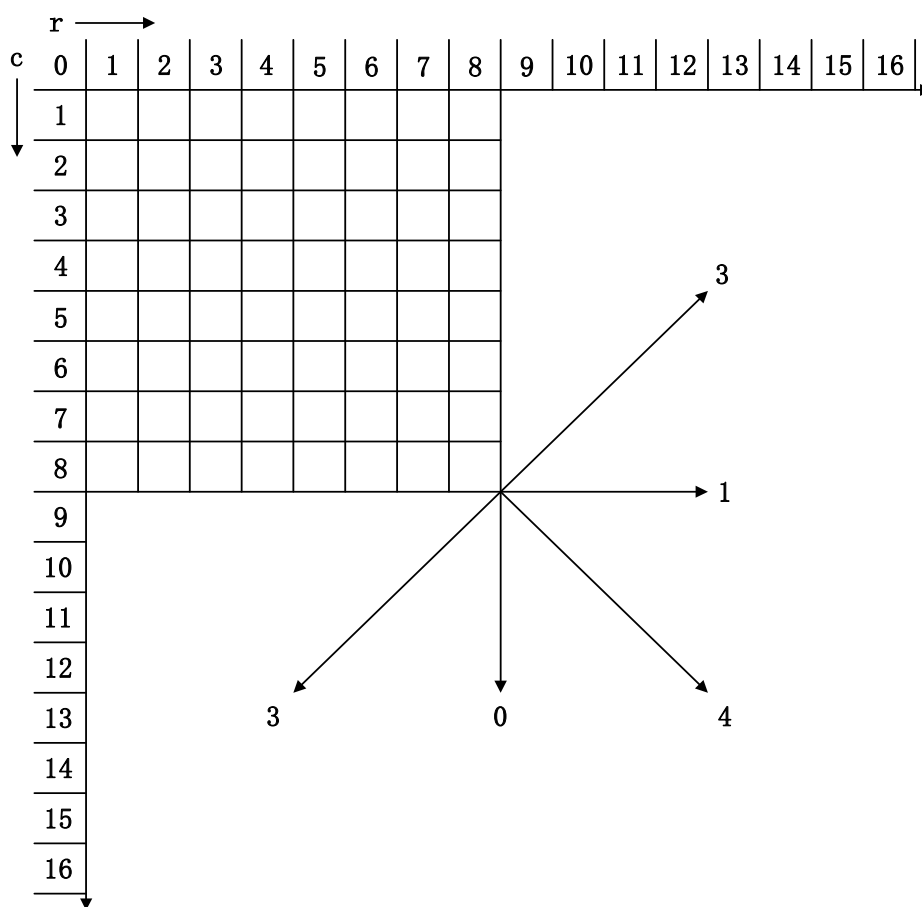


图 12 8×8 亮度块帧内预测模式

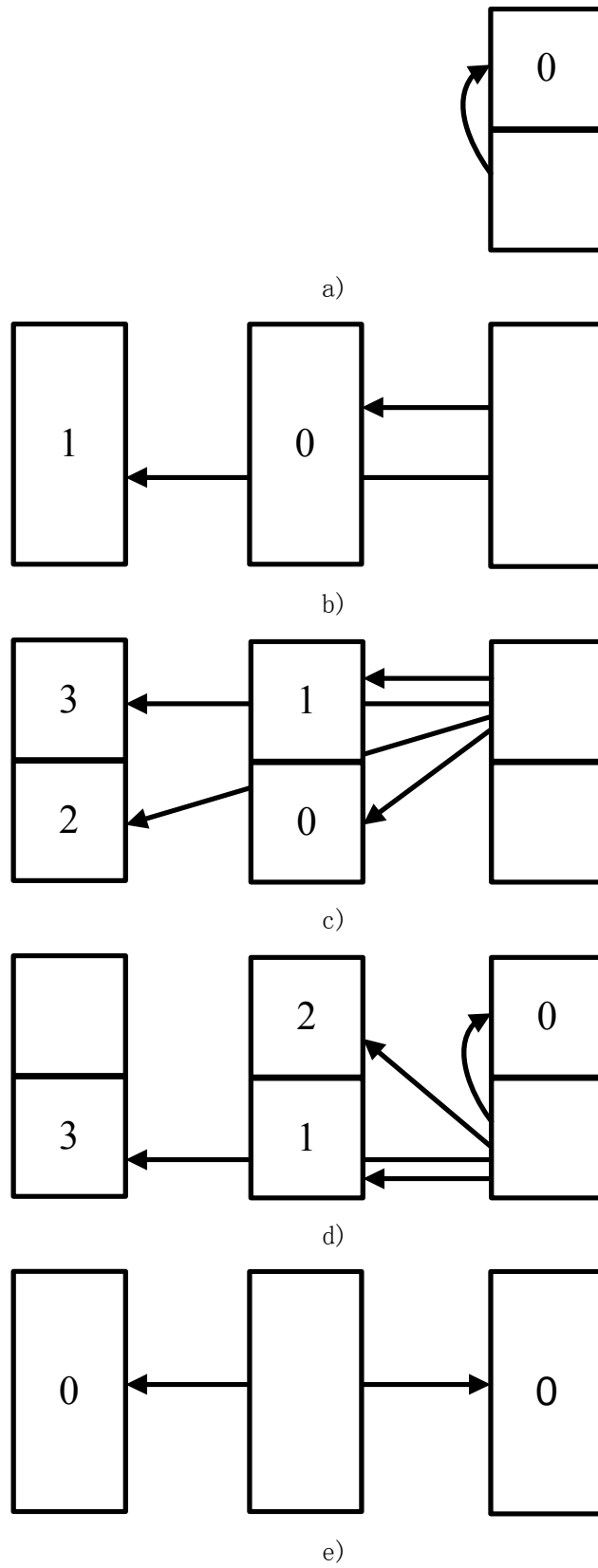
9.4.3 参考图像选择

每帧图像使用的参考图像不应超过两个，它们应是最近解码的I帧或P帧。

参考索引值用来确定对当前图像进行解码处理所用的参考图像，参考图像的两场可有不同的参考索引值，因此参考索引的取值范围是0~3。参考索引值随着参考图像与当前图像距离（显示顺序）的增加而增加，索引值为0的参考图像与当前图像的距离最近，索引值为1的图像距离其次，索引值为3的图像距离最远。两个方向单独处理。参考图像（或场）的参考索引值的标记方法如下（图13中的数字表示参考索引值，箭头所指的是参考图像（或场））：

- 如果当前图像是 I 帧并且 PictureStructure 等于 0，同时当前解码场在显示顺序上是第二场，标记方法如图 13a) 所示。此时参考图像（或场）的个数 NumberOfReference 等于 1。
- 如果当前图像是 P 帧并且 PictureStructure 等于 1，标记方法如图 13b) 所示。此时 NumberOfReference 等于 2。
- 如果当前图像是 P 帧并且 PictureStructure 等于 0，同时当前解码场在显示顺序上是第一场，标记方法如图 13c) 所示。此时 NumberOfReference 等于 4。
- 如果当前图像是 P 帧并且 PictureStructure 等于 0，同时当前解码场在显示顺序上是第二场，标记方法如图 13d) 所示。此时 NumberOfReference 等于 4。
- 如果当前图像是 B 帧并且 PictureStructure 等于 1，标记方法如图 13e) 所示。此时变量 NumberOfReference 等于 2。
- 如果当前图像是 B 帧并且 PictureStructure 等于 0，同时当前解码场在显示顺序上是第一场，标记方法如图 13f) 所示。此时 NumberOfReference 等于 4。
- 如果当前图像是 B 帧并且 PictureStructure 等于 0，同时当前解码场在显示顺序上是第二场，

标记方法如图 13g) 所示。此时 NumberOfReference 等于 4。



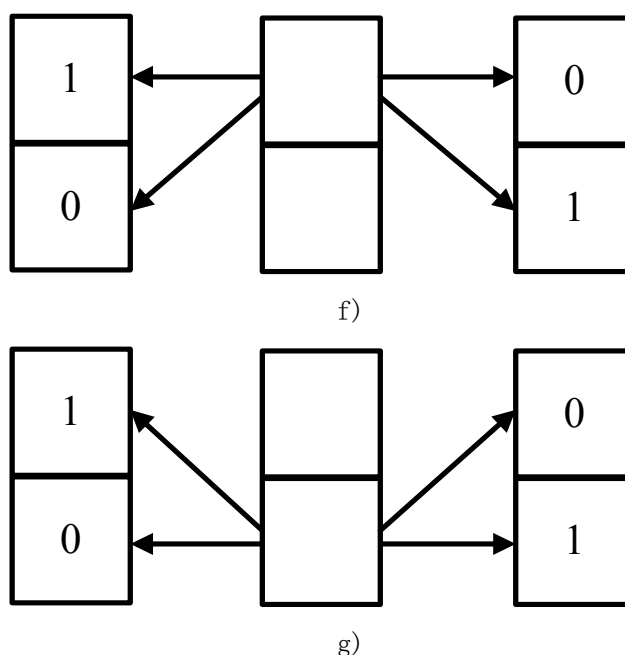


图 13 参考索引标记方法

如果当前图像是I帧，或者当前图像是B帧并且PictureStructure等于1，或者当前图像是P帧或B帧同时picture_reference_flag存在并等于‘1’，则参考索引不出现在比特流中，缺省参考图像（或场）是图13中标记为0的图像（或场）。否则参考索引值等于语法元素mb_reference_index的值。

如果使用加权预测，解码的加权预测参数按参考索引值由小到大依次分配给各参考图像（或场）。如果当前图像是B帧，对参考索引值相同的前后参考图像（或场），解码的加权预测参数先分配给前向参考图像（或场），再分配给后向参考图像（或场）。

9.4.4 运动矢量

一个亮度块E和它的相邻亮度8×8块A、B、C和D之间的空间位置如图14所示。E的大小可以是16×16、16×8、8×16或者8×8。A、B、D都是与E的左上角样本紧邻的块，C是与E的右上角样本紧邻的块。

块的距离索引DistanceIndex定义如下：如果块的所有像素都属于所在隔行扫描图像的第二场（显示顺序）或者都属于所在逐行扫描图像的底场，DistanceIndex等于picture_distance乘2加1；否则DistanceIndex等于picture_distance乘2。

当前块（属于当前图像）和它的运动矢量所指向的参考块（属于参考图像）之间的距离BlockDistance计算如下：

- 如果参考块在当前块之前（显示顺序），BlockDistance 等于当前块的 DistanceIndex 减去参考块的 DistanceIndex 的差加上 512 的和模 512。
- 如果参考块在当前块之后（显示顺序），BlockDistance 等于参考块的 DistanceIndex 减去当前块的 DistanceIndex 的差加上 512 的和模 512。

9.4.4.1 亮度运动矢量预测

图14中块A、B、C、D的原始运动矢量为mvA、mvB、mvC、mvD，对应的BlockDistance为BlockDistanceA、BlockDistanceB、BlockDistanceC、BlockDistanceD。

- 如果A‘不可用’或者采用帧内预测模式，或者与当前块E没有同一预测方向的运动矢量，mvA为零矢量，BlockDistanceA等于1，A的参考索引值为-1。
- 如果B‘不可用’或者采用帧内预测模式，或者与当前块E没有同一预测方向的运动矢量，mvB为零矢量，BlockDistanceB等于1，B的参考索引值为-1。

- 如果 D ‘不可用’ 或者采用帧内预测模式，或者与当前块 E 没有同一预测方向的运动矢量， mvD 为零矢量， $BlockDistanceD$ 等于 1，D 的参考索引值为-1。
 - 如果 C ‘不可用’，那么 mvC 等于 mvD ， $BlockDistanceC$ 等于 $BlockDistanceD$ ，C 的参考索引值等于 D 的参考索引值。
 - 如果 C 采用帧内预测模式，或者与当前块 E 没有同一预测方向的运动矢量， mvC 为零矢量， $BlockDistanceC$ 等于 1，C 的参考索引值为-1。
- 块 ‘不可用’ 指该块不存在，或者尚未解码；否则该块 ‘可用’。

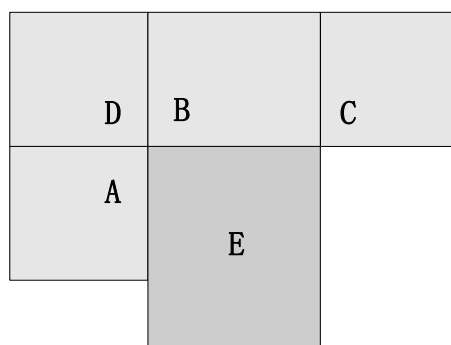


图 14 亮度块 E 和相邻亮度块的空间位置关系

当前块 E 的运动矢量预测值 $MVEPred$ 计算过程如下：

第一步，如果 A、B、C 三者中只有一个块 X 的参考索引值不为-1，那么 $MVEPred$ 等于 mvX (X 为 A、B 或 C)；否则进行第二步。

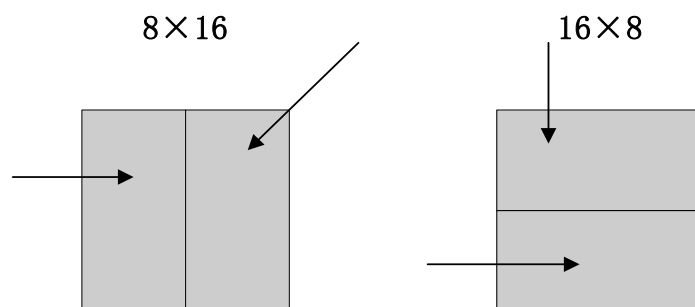
第二步，如果 E 所在宏块按 16×8 或 8×16 模式编码，计算过程如下（见图 15）：

—— 8×16 模式：

- 1) E 为左块：如果 A 和 E 的参考索引值相同， $MVEPred$ 等于 mvA ；否则进行第三步。
- 2) E 为右块：如果 C 和 E 的参考索引值相同， $MVEPred$ 等于 mvC ；否则进行第三步。

—— 16×8 模式：

- 1) E 为上块：如果 B 和 E 的参考索引值相同， $MVEPred$ 等于 mvB ；否则进行第三步。
- 2) E 为下块：如果 A 和 E 的参考索引值相同， $MVEPred$ 等于 mvA ；否则进行第三步。

图 15 8×16 或 16×8 模式预测

第三步，首先根据 $BlockDistanceA$ 、 $BlockDistanceB$ 、 $BlockDistanceC$ 、 $BlockDistanceE$ 对 $mvA(mvA_x, mvA_y)$ 、 $mvB(mvB_x, mvB_y)$ 、 $mvC(mvC_x, mvC_y)$ 进行缩放，得到 $MVA(MVA_x, MVA_y)$ 、 $MVB(MVB_x, MVB_y)$ 、 $MVC(MVC_x, MVC_y)$ ，然后计算 $MVEPred$ 。

$$MVA_x = \text{Sign}(mvA_x) \times ((\text{Abs}(mvA_x) \times BlockDistanceE \times (512 / BlockDistanceA) + 256)$$

$$\begin{aligned} & \gg 9) \\ MVA_y &= \text{Sign}(mvA_y) \times ((\text{Abs}(mvA_y) \times \text{BlockDistanceE} \times (512 / \text{BlockDistanceA}) + 256) \\ & \gg 9) \\ MVB_x &= \text{Sign}(mvB_x) \times ((\text{Abs}(mvB_x) \times \text{BlockDistanceE} \times (512 / \text{BlockDistanceB}) + 256) \\ & \gg 9) \\ MVB_y &= \text{Sign}(mvB_y) \times ((\text{Abs}(mvB_y) \times \text{BlockDistanceE} \times (512 / \text{BlockDistanceB}) + 256) \\ & \gg 9) \\ MVC_x &= \text{Sign}(mvC_x) \times ((\text{Abs}(mvC_x) \times \text{BlockDistanceE} \times (512 / \text{BlockDistanceC}) + 256) \\ & \gg 9) \\ MVC_y &= \text{Sign}(mvC_y) \times ((\text{Abs}(mvC_y) \times \text{BlockDistanceE} \times (512 / \text{BlockDistanceC}) + 256) \\ & \gg 9) \end{aligned}$$

其中BlockDistanceE是当前块E对应的BlockDistance。

定义距离 $\text{Dist}(MV1, MV2) = \text{Abs}(x1 - x2) + \text{Abs}(y1 - y2)$ ，其中运动矢量 $MV1=[x1, y1]$ ， $MV2=[x2, y2]$ 。定义VAB等于 $\text{Dist}(MVA, MVB)$ ，VBC等于 $\text{Dist}(MVB, MVC)$ ，VCA等于 $\text{Dist}(MVC, MVA)$ 。MVEPred的计算如下：

- 计算FMV等于 $\text{Median}(VAB, VBC, VCA)$ ；
- 如果FMV和VAB相等，MVEPred等于MVC；
- 否则，如果FMV和VBC相等，MVEPred等于MVA；
- 否则，MVEPred等于MVB。

9.4.4.2 亮度运动矢量解码

当前块E的运动矢量mvE等于MVEPred加上由mv_diff_x和mv_diff_y解码得到的运动矢量增量的和，其基本单位为四分之一样本。

9.4.5 宏块编码模板

如果当前宏块的cbp在比特流中，

表15中CodeNum的值为cbp；否则，如果当前宏块的类型不等于P_Skip或B_Skip，

表15中CodeNum的值等于CBPCodeNum（见9.4.1）。由CodeNum得到6位无符号整数MbCBP（见8.2）。MbCBP表示宏块中顺序号为0到5的4个8×8亮度块和2个8×8色度块是否包含非零变换系数。MbCBP的第n位为‘0’表示顺序号为n的8×8块没有非零系数，等于‘1’表示该8×8块至少有一个非零系数（顺序号见6.5）。

如果当前宏块的cbp_422在比特流中，

表16中CodeNum的值为cbp_422。由CodeNum得到2位无符号整数MbCBP422（见8.2）。MbCBP422表示4:2:2格式的宏块中顺序号为6和7的2个8×8色度块是否包含非零变换系数。MbCBP422的第n位为‘0’表示顺序号为n+6的8×8色度块没有非零系数，等于‘1’表示该8×8色度块至少有一个非零系数（顺序号见6.5）。

9.4.6 量化参数

如果当前宏块的mb_qp_delta在比特流中，当前宏块的量化参数CurrentQP等于PreviousQP加上mb_qp_delta；否则CurrentQP等于PreviousQP。宏块解码完成后，将PreviousQP的值设为CurrentQP。

9.5 块解码

9.5.1 变长码解码

本条定义由语法元素生成量化系数值数组（Level数组）和量化系数游程数组（Run数组）的过程。Level数组包含非0量化系数的幅值，Run数组包含当前非0量化系数前的连续0的个数。生成量化系数数组所需语法元素的解析过程见8.3。

生成Run数组和Level数组的过程如下（数组地址初始值为0）：

——如果 `trans_coefficient` 小于 59：

- 1) 如果 `CurrentVLCTable` 中包含此 `trans_coefficient` 值的索引项，则以 `trans_coefficient` 为索引查 `CurrentVLCTable`，得到量化系数和游程，把它们分别放入 Level 数组和 Run 数组；
- 2) 如果表中不包含此 `trans_coefficient` 值的索引项，则以 $(\text{trans_coefficient}-1)$ 为索引查 `CurrentVLCTable`，得到量化系数和游程，并将量化系数符号取反后放入 Level 数组，将游程放入 Run 数组。

——如果 `trans_coefficient` 大于或等于 59：

- 1) 将 $(\text{trans_coefficient}-59)/2$ 放入 Run 数组。
- 2) 由 `CurrentVLCTable` 查表 A.20 得到 `MaxRun`。如果 $(\text{trans_coefficient}-59)/2$ 大于 `MaxRun`，`RefAbsLevel` 等于 1；否则以 $(\text{trans_coefficient}-59)/2$ 为索引查 `CurrentVLCTable` 得到 `RefAbsLevel`。
- 3) 如果 `trans_coefficient` 是奇数，将 $-(\text{RefAbsLevel}+\text{escape_level_diff})$ 放入 Level 数组；否则，将 $(\text{RefAbsLevel}+\text{escape_level_diff})$ 放入 Level 数组。

——如果 `trans_coefficient` 等于 EOB，结束块系数解码；否则数组地址加 1，按 8.3 定义的方法更新 `CurrentVLCTable`，解码下一个系数值和游程。

9.5.2 逆扫描

由解码的Level数组和Run数组生成数组`QuantCoeffArray`（包含64个量化系数）的步骤如下：

——首先将 `QuantCoeffArray` 数组的所有元素初始化为 0；

——第二步将非 0 量化系数的值赋给 `QuantCoeffArray` 数组中相应的元素。定义 `j` 和 `coeffNum`，令 `j` 等于 9.5.1 中 Run 数组或 Level 数组中最后一个赋值元素的地址索引，`coeffNum` 等于 -1，`while (j >= 0)`

```
{
    coeffNum += ( Run[j] + 1 )
    QuantCoeffArray[coeffNum] = Level[j]
    j--
}
```

——第三步通过逆块扫描将 `QuantCoeffArray` 数组映射为 `QuantCoeffMatrix` 数组。

- 1) 如果 `progressive_frame` 的值为 ‘1’，或者 `progressive_frame` 的值为 ‘0’ 并且 `picture_structure` 的值为 ‘1’，逆块扫描方法见图 16 a)。
- 2) 如果 `progressive_frame` 的值为 ‘0’ 并且 `picture_structure` 的值为 ‘0’，逆块扫描方法见图 16 b)。

设 `QuantCoeffArray` 数组中某元素的地址为 `k`，在图 16 a) 或图 16 b) 中找到值为 `k` 的单元对应的列号 `i` 和行号 `j`，然后将 `QuantCoeffArray[k]` 赋给 `QuantCoeffMatrix[i, j]`。

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

j

a)

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

j

b)

图 16 逆块扫描

9.6 反量化

9.6.1 确定量化参数

亮度量化参数和色度量化参数的取值范围是0~63。

如果当前块是亮度块，其量化参数QP等于CurrentQP。如果当前块是色度块，以CurrentQP为索引查表22得到色度块的QP。

表 22 色度块的 QP 与 CurrentQP 的映射关系

CurrentQP	<43	43	44	45	46	47	48	49	50	51	52
QP	CurrentQP	42	43	43	44	44	45	45	46	46	47
CurrentQP	53	54	55	56	57	58	59	60	61	62	63
QP	47	48	48	48	49	49	49	50	50	50	51

9.6.2 反量化

本条定义根据量化参数QP将二维量化系数矩阵QuantCoeffMatrix转换为二维变换系数矩阵CoeffMatrix的过程。其中，量化系数的取值范围是 $-2^{11} \sim 2^{11}-1$ 。

二维变换系数矩阵CoeffMatrix的元素 w_{ij} 由下式得到：

$$w_{ij} = (\text{QuantCoeffMatrix}[i,j] \times \text{DequantTable}(\text{QP}) + 2^{\text{ShiftTable}(\text{QP})-1}) \gg \text{ShiftTable}(\text{QP}) \quad i,j=0 \sim 7$$

其中QP与DequantTable和ShiftTable的关系见表23。

表 23 QP 与 DequantTable 和 ShiftTable 的关系

QP	0	1	2	3	4	5	6	7
DequantTable(QP)	32768	36061	38968	42495	46341	50535	55437	60424
ShiftTable(QP)	14	14	14	14	14	14	14	14
QP	8	9	10	11	12	13	14	15
DequantTable(QP)	32932	35734	38968	42495	46177	50535	55109	59933
ShiftTable(QP)	13	13	13	13	13	13	13	13
QP	16	17	18	19	20	21	22	23
DequantTable(QP)	65535	35734	38968	42577	46341	50617	55027	60097
ShiftTable(QP)	13	12	12	12	12	12	12	12
QP	24	25	26	27	28	29	30	31
DequantTable(QP)	32809	35734	38968	42454	46382	50576	55109	60056
ShiftTable(QP)	11	11	11	11	11	11	11	11
QP	32	33	34	35	36	37	38	39
DequantTable(QP)	65535	35734	38968	42495	46320	50515	55109	60076
ShiftTable(QP)	11	10	10	10	10	10	10	10
QP	40	41	42	43	44	45	46	47
DequantTable(QP)	65535	35744	38968	42495	46341	50535	55099	60087
ShiftTable(QP)	10	9	9	9	9	9	9	9
QP	48	49	50	51	52	53	54	55
DequantTable(QP)	65535	35734	38973	42500	46341	50535	55109	60097
ShiftTable(QP)	9	8	8	8	8	8	8	8
QP	56	57	58	59	60	61	62	63
DequantTable(QP)	32771	35734	38965	42497	46341	50535	55109	60099
ShiftTable(QP)	7	7	7	7	7	7	7	7

从符合GB/T 20090的本部分的比特流中解码得到的变换系数的取值范围应为 $-2^{13} \sim 2^{13}-1$ 。

9.7 反变换

本条定义将8×8变换系数矩阵CoeffMatrix转换为8×8残差样值矩阵ResidueMatrix的过程，步骤如下：

——首先，对变换系数矩阵进行如下水平反变换：

$$H' = \text{CoeffMatrix} \times T_8^T$$

其中, T_8 是8×8反变换矩阵, T_8^T 是 T_8 的转置矩阵, H' 表示水平反变换后的中间结果。

$$T_8 = \begin{bmatrix} 8 & 10 & 10 & 9 & 8 & 6 & 4 & 2 \\ 8 & 9 & 4 & -2 & -8 & -10 & -10 & -6 \\ 8 & 6 & -4 & -10 & -8 & 2 & 10 & 9 \\ 8 & 2 & -10 & -6 & 8 & 9 & -4 & -10 \\ 8 & -2 & -10 & 6 & 8 & -9 & -4 & 10 \\ 8 & -6 & -4 & 10 & -8 & -2 & 10 & -9 \\ 8 & -9 & 4 & 2 & -8 & 10 & -10 & 6 \\ 8 & -10 & 10 & -9 & 8 & -6 & 4 & -2 \end{bmatrix}$$

——第二步, 矩阵 H' 的元素 h'_{ij} 计算如下:

$$h''_{ij} = (\text{Clip3}(-2^{15}, 2^{15}-1, (h'_{ij} + 4))) \gg 3 \quad i, j = 0 \sim 7$$

——第三步, 对矩阵 H'' 进行如下垂直反变换:

$$H = T_8 \times H''$$

其中, H 表示反变换后的8×8矩阵。

——第四步, 残差样值矩阵ResidueMatrix的元素 r_{ij} 计算如下:

$$r_{ij} = (\text{Clip3}(-2^{15}, 2^{15}-1, (h_{ij} + 2^6))) \gg 7 \quad i, j = 0 \sim 7$$

其中 h_{ij} 是 H 矩阵的元素。

9.8 帧内预测

当前帧内预测块由其上边和左边的参考样本 $r[i]$ ($i=0 \sim 16$)和 $c[i]$ ($i=0 \sim 16$)来预测(r 、 c 可表示亮度或色度参考样本), 见图12, 其中 $r[0]$ 等于 $c[0]$ 。如果帧内预测需要用到 i 大于16的上边和左边的参考样本, 则 $r[i]=r[16]$, $c[i]=c[16]$ ($i>16$)。本条定义帧内预测的具体方法, 预测完成后得到一个8×8预测样本矩阵predMatrix。

9.8.1 参考样本的获得

设当前块所属的图像样本矩阵为 I , I 可表示亮度或色度矩阵。如果某个图像样本所在的块“不存在”或此样本尚未解码, 则此样本“不可用”; 反之此样本“可用”。

设当前块左上角样本的坐标是 (x_0, y_0) , 其参考样本按以下规则获得:

- 如果坐标为 (x_0+i-1, y_0-1) ($i=1 \sim 8$)的样本“可用”, 则 $r[i]$ 等于 $I[x_0+i-1, y_0-1]$, $r[i]$ “可用”; 否则 $r[i]$ “不可用”;
- 如果坐标为 (x_0+i-1, y_0-1) ($i=9 \sim 16$)的样本“可用”, 则 $r[i]$ 等于 $I[x_0+i-1, y_0-1]$, $r[i]$ “可用”; 否则 $r[i]$ 等于 $r[8]$, $r[i]$ 是否“可用”由 $r[8]$ 是否“可用”决定;
- 如果坐标为 (x_0-1, y_0+i-1) ($i=1 \sim 8$)的样本“可用”, 则 $c[i]$ 等于 $I[x_0-1, y_0+i-1]$, $c[i]$ “可用”; 否则 $c[i]$ “不可用”;
- 如果坐标为 (x_0-1, y_0+i-1) ($i=9 \sim 16$)的样本“可用”, 则 $c[i]$ 等于 $I[x_0-1, y_0+i-1]$, $c[i]$ “可用”; 否则 $c[i]$ 等于 $c[8]$, $c[i]$ 是否“可用”由 $c[8]$ 是否“可用”决定;
- 如果坐标为 (x_0-1, y_0-1) 的样本“可用”, 则 $r[0]$ 等于 $I[x_0-1, y_0-1]$, $r[0]$ “可用”; 否则
 - 1) 如果 $r[1]$ “可用”并且 $c[1]$ “不可用”, 则 $r[0]$ 等于 $r[1]$, $r[0]$ “可用”; 否则
 - 2) 如果 $c[1]$ “可用”并且 $r[1]$ “不可用”, 则 $r[0]$ 等于 $c[1]$, $r[0]$ “可用”; 否则 $r[0]$ “不可用”。

9.8.2 亮度块帧内预测

根据IntraLumaPredMode决定亮度块帧内预测方法。

——IntraLumaPredMode 等于 0 (Intra_8x8_Vertical 预测)

当 $r[i]$ ($i=1\sim 8$) ‘可用’时, 该模式才被使用, 此时

$$\text{predMatrix}[x, y] = r[x+1] \quad (x, y=0\sim 7)$$

——IntraLumaPredMode 等于 1 (Intra_8x8_Horizontal 预测)

当 $c[i]$ ($i=1\sim 8$) ‘可用’时, 该模式才被使用, 此时

$$\text{predMatrix}[x, y] = c[y+1] \quad (x, y=0\sim 7)$$

——IntraLumaPredMode 等于 2 (Intra_8x8_DC 预测)

1) 如果 $r[i]$ 、 $c[i]$ ($i=0\sim 9$) 都 ‘可用’, 则

$$\text{predMatrix}[x, y] = ((r[x]+2\times r[x+1]+r[x+2]+2)\gg 2+(c[y]+2\times c[y+1]+c[y+2]+2)\gg 2)\gg 1 \quad (x, y=0\sim 7); \text{ 否则}$$

2) 如果 $r[i]$ ($i=0\sim 9$) ‘可用’, 则

$$\text{predMatrix}[x, y] = (r[x]+2\times r[x+1]+r[x+2]+2)\gg 2 \quad (x, y=0\sim 7); \text{ 否则}$$

3) 如果 $c[i]$ ($i=0\sim 9$) ‘可用’, 则

$$\text{predMatrix}[x, y] = (c[y]+2\times c[y+1]+c[y+2]+2)\gg 2 \quad (x, y=0\sim 7); \text{ 否则}$$

4) $\text{predMatrix}[x, y] = 128 \quad (x, y=0\sim 7)$ 。

——IntraLumaPredMode 等于 3 (Intra_8x8_Down_Left 预测)

当 $r[i]$ 、 $c[i]$ ($i=1\sim 16$) 均 ‘可用’时, 该模式才被使用, 此时

$$\text{predMatrix}[x, y] = ((r[x+y+1]+2\times r[x+y+2]+r[x+y+3]+2)\gg 2+(c[x+y+1]+2\times c[x+y+2]+c[x+y+3]+2)\gg 2)\gg 1 \quad (x, y=0\sim 7)。$$

——IntraLumaPredMode 等于 4 (Intra_8x8_Down_Right 预测)

当 $r[i]$ 、 $c[i]$ ($i=0\sim 16$) 均 ‘可用’时, 该模式才被使用, 此时

1) 如果 x 等于 y , $\text{predMatrix}[x, y] = (c[1]+2\times r[0]+r[1]+2)\gg 2 \quad (x, y=0\sim 7)$; 否则

2) 如果 x 大于 y , $\text{predMatrix}[x, y] = (r[x-y+1]+2\times r[x-y]+r[x-y-1]+2)\gg 2 \quad (x, y=0\sim 7)$; 否则

3) 如果 y 大于 x , $\text{predMatrix}[x, y] = (c[y-x+1]+2\times c[y-x]+c[y-x-1]+2)\gg 2 \quad (x, y=0\sim 7)$ 。

9.8.3 色度块帧内预测

根据IntraChromaPredMode决定色度块帧内预测方法。

——IntraChromaPredMode 等于 0 (Intra_Chroma_DC 预测)

1) 如果 $r[i]$ 、 $c[i]$ ($i=0\sim 9$) 都 ‘可用’, 则

$$\text{predMatrix}[x, y] = ((r[x]+2\times r[x+1]+r[x+2]+2)\gg 2+(c[y]+2\times c[y+1]+c[y+2]+2)\gg 2)\gg 1 \quad (x, y=0\sim 7); \text{ 否则}$$

2) 如果 $r[i]$ ($i=0\sim 9$) ‘可用’, 则

$$\text{predMatrix}[x, y] = (r[x]+2\times r[x+1]+r[x+2]+2)\gg 2 \quad (x, y=0\sim 7); \text{ 否则}$$

3) 如果 $c[i]$ ($i=0\sim 9$) ‘可用’, 则

$$\text{predMatrix}[x, y] = (c[y]+2\times c[y+1]+c[y+2]+2)\gg 2 \quad (x, y=0\sim 7); \text{ 否则}$$

4) $\text{predMatrix}[x, y] = 128 \quad (x, y=0\sim 7)$ 。

——IntraChromaPredMode 等于 1 (Intra_Chroma_Horizontal 预测)

当 $c[i]$ ($i=1\sim 8$) ‘可用’时, 该模式才被使用, 此时

$\text{predMatrix}[x, y] = c[y+1] \quad (x, y=0\sim7)$ 。

——IntraChromaPredMode 等于 2 (Intra_Chroma_Vertical 预测)

当 $r[i] \quad (i=1\sim8)$ ‘可用’ 时, 该模式才被使用, 此时

$\text{predMatrix}[x, y] = r[x+1] \quad (x, y=0\sim7)$ 。

——IntraChromaPredMode 等于 3 (Intra_Chroma_Plane 预测)

当 $r[i]$ 、 $c[i] \quad (i=1\sim8)$ 均 ‘可用’ 时, 该模式才被使用, 此时

$\text{predMatrix}[x, y] = \text{Clip1}((ia+(x-3)\times ib+(y-3)\times ic+16)\gg 5) \quad (x, y=0\sim7)$ 。

其中, $ia = (r[8]+c[8])\ll 4$, $ib = (17\times ih+16)\gg 5$, $ic = (17\times iv+16)\gg 5$,

$$ih = \sum_{i=0}^3 (i+1)\times (r[5+i]-r[3-i]), \quad iv = \sum_{i=0}^3 (i+1)\times (c[5+i]-c[3-i])。$$

9.9 帧间预测

9.9.1 亮度运动矢量导出

如果当前宏块类型或者当前子块类型是跳过模式、直接模式或对称模式, 其运动矢量按照下面定义的方法导出 (具体见图17、图18), 否则将9.4.4解码得到的运动矢量, 按照宏块划分顺序(在图5中定义)分配给相应的子块。

——如果当前宏块类型为 P_Skip:

- 1) 当前块的前向参考图像为缺省参考图像, 即图13中标记为0的图像;
- 2) 如果当前宏块的上面宏块B或左面的宏块A ‘不可用’, mvE 等于零矢量;
- 3) 如果 mvA 为零矢量并且宏块A的参考索引值为0, 或 mvB 为零矢量并且宏块B的参考索引值为0, 则 mvE 等于零矢量;
- 4) 对于其他情况, $mvE = MVEPred$ 。

——如果当前宏块类型为 B_Skip 或 B_Direct_16x16, 或当前块类型为 SB_Direct_8x8:

第一步:

- 1) 如果后向参考图像中与当前块的左上角样本位置对应的块所在的宏块类型为 ‘I_8x8’, 则当前块的前后向参考图像均为缺省参考图像, 即图13中标记为0的图像, 该块的运动矢量是根据9.4.4得到的运动矢量预测值 $MVEPred$, 此时当前块E的大小应为 16×16 。
- 2) 否则, 如果当前块所在图像的 PictureStructure 等于1, 此时当前块E的大小应为 8×8 ,
 - ◆ 如果后向参考图像的 PictureStructure 等于1, 当前块的前后向参考图像为图13e)中标记为0的图像, 其前后向距离索引分别为 $DistanceIndexFw$ 和 $DistanceIndexBw$; 当前块的前后向 BlockDistance 分别为 $BlockDistanceFw$ 和 $BlockDistanceBw$ 。在后向参考图像中与当前块的左上角样本位置对应的块的运动矢量为 $mvRef(mvRef_x, mvRef_y)$, 该运动矢量指向的参考图像的距离索引为 $DistanceIndexRef$;
 - ◆ 否则 (后向参考图像的 PictureStructure 等于0), 分别将前后向参考图像的两场合并, 如图13e)所示。当前块的前后向参考图像为图13e)中标记为0的图像。前后向参考图像显示顺序上第一场的距离索引分别为 $DistanceIndexFw$ 和 $DistanceIndexBw$; 当前块的前后向 BlockDistance 分别为 $BlockDistanceFw$ 和 $BlockDistanceBw$ 。在后向参考图像中与当前块的左上角样本位置对应的块的运动矢量为 $mvRef(mvRef_x, mvRef_y)$, 该运动矢量指向的参考图像的显示顺序上第一场的距离索引为 $DistanceIndexRef$ 。
- 3) 否则 (当前块所在图像的 PictureStructure 等于0), 此时当前块E的大小应为 8×8 ,
 - ◆ 在后向参考图像中与当前块的左上角样本位置对应的块的运动矢量为 $mvRef(mvRef_x, mvRef_y)$, 该运动矢量指向的参考图像的距离索引为 $DistanceIndexRef$; 当后向参考图像的 PictureStructure 等于1时, 如果后向参考图像中与当前块的左

上角样本位置对应的块的 $MbIndex < MbWidth \times MbHeight / 2$ 时, $DistanceIndexRef$ 等于 $picture_distance \times 2$; 否则 $DistanceIndexRef$ 等于 $picture_distance \times 2 + 1$ 。 $picture_distance$ 为 $mvRef$ 所指向的参考图像的 $picture_distance$;

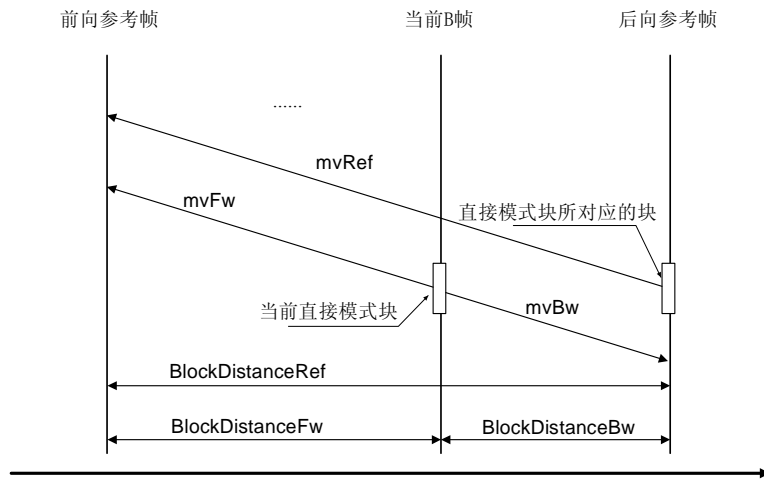
- ◆ 当前块的前向参考图像中标记为 0 和 1 的图像的距离索引分别为 $DistanceIndexFw0$ 和 $DistanceIndexFw1$ 。如果 $DistanceIndexRef$ 等于 $DistanceIndexFw0$, 则当前块的前向参考图像是参考索引标记为 0 的图像, $DistanceIndexFw$ 等于 $DistanceIndexFw0$; 否则当前块的前向参考图像是参考索引标记为 1 的图像, $DistanceIndexFw$ 等于 $DistanceIndexFw1$;
- ◆ 当前块的后向参考图像中标记为 0 和 1 的图像的距离索引分别为 $DistanceIndexBw0$ 和 $DistanceIndexBw1$ 。如果 $MbIndex < MbWidth \times MbHeight / 2$, 则当前块的后向参考图像是参考索引标记为 0 的图像, $DistanceIndexBw$ 等于 $DistanceIndexBw0$; 否则当前块的后向参考图像是参考索引标记为 1 的图像, $DistanceIndexBw$ 等于 $DistanceIndexBw1$ 。

第二步:

- 1) $BlockDistanceRef = (DistanceIndexBw - DistanceIndexRef + 512) \% 512$
当前块的距离索引为 $DistanceIndexCur$, 则
 $BlockDistanceFw = (DistanceIndexCur - DistanceIndexFw + 512) \% 512$
 $BlockDistanceBw = (DistanceIndexBw - DistanceIndexCur + 512) \% 512$
- 2) 如果当前块所在图像的 $PictureStructure$ 等于 1, 后向参考图像的 $PictureStructure$ 等于 0, 则 $mvRef_y = mvRef_y \times 2$;
- 3) 如果当前块所在图像的 $PictureStructure$ 等于 0, 后向参考图像的 $PictureStructure$ 等于 1, 则 $mvRef_y = mvRef_y / 2$ 。

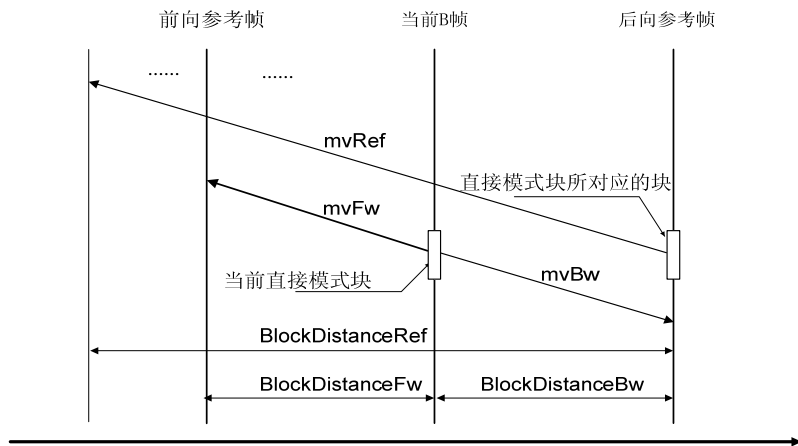
第三步:

- 1) 当前块的前向运动矢量 $mvFw(mvFw_x, mvFw_y)$ 为:
 - ◆ 如果 $mvRef_x$ 小于 0, 则 $mvFw_x = -(((16384/BlockDistanceRef) \times (1 - mvRef_x \times BlockDistanceFw) - 1) \gg 14)$; 否则 $mvFw_x = ((16384/BlockDistanceRef) \times (1 + mvRef_x \times BlockDistanceFw) - 1) \gg 14$ 。
 - ◆ 如果 $mvRef_y$ 小于 0, 则 $mvFw_y = -(((16384/BlockDistanceRef) \times (1 - mvRef_y \times BlockDistanceFw) - 1) \gg 14)$; 否则 $mvFw_y = ((16384/BlockDistanceRef) \times (1 + mvRef_y \times BlockDistanceFw) - 1) \gg 14$ 。
- 2) 当前块的后向运动矢量 $mvBw(mvBw_x, mvBw_y)$ 为:
 - ◆ 如果 $mvRef_x$ 小于 0, 则 $mvBw_x = ((16384/BlockDistanceRef) \times (1 - mvRef_x \times BlockDistanceBw) - 1) \gg 14$; 否则 $mvBw_x = -(((16384/BlockDistanceRef) \times (1 + mvRef_x \times BlockDistanceBw) - 1) \gg 14)$;
 - ◆ 如果 $mvRef_y$ 小于 0, 则 $mvBw_y = ((16384/BlockDistanceRef) \times (1 - mvRef_y \times BlockDistanceBw) - 1) \gg 14$; 否则 $mvBw_y = -(((16384/BlockDistanceRef) \times (1 + mvRef_y \times BlockDistanceBw) - 1) \gg 14)$;



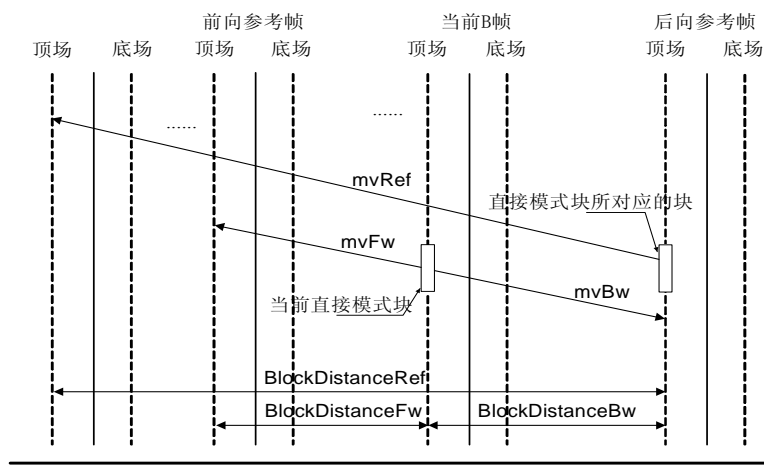
当前帧的PictureStructure等于1时直接或跳过模式块所对应的块指向标记为0的参考图像

a)



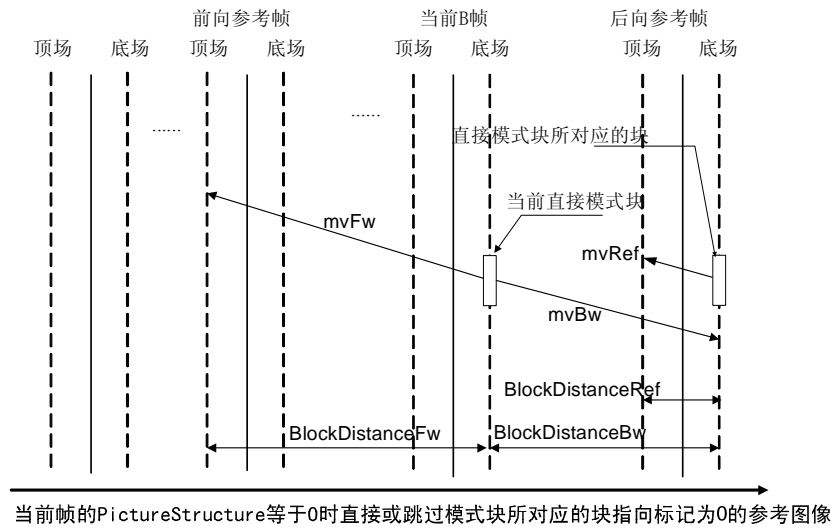
当前帧的PictureStructure等于1时直接或跳过模式块所对应的块指向标记为1的参考图像

b)



当前帧的PictureStructure等于0时直接或跳过模式块所对应的块指向标记为3的参考图像

c)



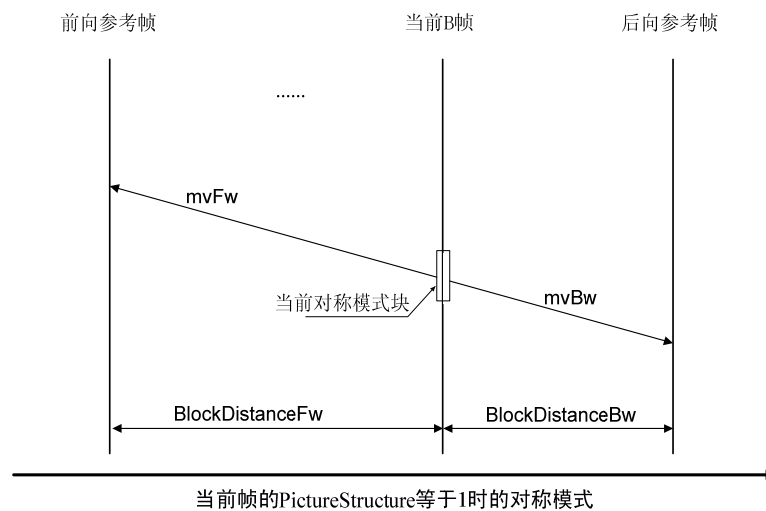
d)

图 17 直接模式的运动矢量推导过程

——如果当前块类型为对称模式，运动矢量按照下面定义的方法导出：
 前向参考索引由比特流中的 mb_reference_index 解码得到。如果 PictureStructure 等于 1，
 则后向参考索引与前向参考索引相等；否则，后向参考索引等于 1 减去前向参考索引。对称模
 式块的前向运动矢量 mvFw 按照 9.4.4 的方法得到，当前块的后向运动矢量 mvBw (mvBw_x,
 mvBw_y) 为：

$$mvBw_x = -((mvFw_x \times BlockDistanceBw \times (512 / BlockDistanceFw) + 256) \gg 9)$$

$$mvBw_y = -((mvFw_y \times BlockDistanceBw \times (512 / BlockDistanceFw) + 256) \gg 9)$$



a)

$b' = (-C + 5D + 5E - F)$; 最终的预测值 $b = \text{Clip1}((b' + 4) \gg 3)$ 。

——二分之一样本 h : 首先用 F_1 对垂直方向上最近的 4 个整数样本滤波, 得到中间值 $h' = (-A + 5D + 5H - K)$; 最终的预测值 $h = \text{Clip1}((h' + 4) \gg 3)$ 。

——二分之一样本 j : 首先用 F_1 在水平或垂直方向上对最近的 4 个二分之一样本中间值滤波, 得到中间值 $j' = (-bb' + 5h' + 5m' - cc')$, 或者 $j' = (-aa' + 5b' + 5s' - dd')$ 。其中 aa' , dd' 和 s' 是相应位置二分之一样本中间值 (用 F_1 在水平方向滤波得到), bb' , cc' 和 m' 是相应位置二分之一样本中间值 (用 F_1 在垂直方向滤波得到)。最终的预测值 $j = \text{Clip1}((j' + 32) \gg 6)$ 。采用水平方向或垂直方向滤波得到的值相同。

四分之一样本的计算过程如下:

——四分之一样本 a : 首先用 F_2 在水平方向上对 ee' , D , b 和 E 四个值滤波, 得到中间值 $a' = (ee' + 7D + 7b + E)$; 最终的预测值 $a = \text{Clip1}((a' + 64) \gg 7)$ 。其中 ee' 和 b 是相应位置二分之一样本中间值, D 和 E 是相应位置整数样本放大 8 倍的值。四分之一样本 c 的插值过程与 a 的插值过程相同。

——四分之一样本 d : 首先用 F_2 在垂直方向上对 ff' , D' , h 和 H 四个值滤波, 得到中间值 $d' = (ff' + 7D' + 7h + H)$; 最终的预测值 $d = \text{Clip1}((d' + 64) \gg 7)$ 。其中 ff' 和 h 是相应位置二分之一样本中间值, D' 和 H 是相应位置整数样本放大 8 倍的值。四分之一样本 n 的插值过程与 d 的插值过程相同。

——四分之一样本 i : 首先用 F_2 在水平方向上对 gg' , h' , j 和 m' 四个值滤波, 得到中间值 $i' = (gg' + 7h' + 7j + m')$; 最终的预测值 $i = \text{Clip1}((i' + 512) \gg 10)$ 。其中 gg' 和 j 是相应位置二分之一中间值, h' 和 m' 是相应位置二分之一样本中间值放大 8 倍的值。四分之一样本 k 的插值过程与 i 的插值过程相同。

——四分之一样本 f : 首先用 F_2 在垂直方向上对 hh' , b' , j 和 s'' 四个值滤波, 得到中间值 $f' = (hh' + 7b' + 7j + s'')$; 最终的预测值 $f = \text{Clip1}((f' + 512) \gg 10)$ 。其中 hh' 和 j 是相应位置二分之一中间值, b' 和 s'' 是相应位置二分之一样本中间值放大 8 倍的值。四分之一样本 q 的插值过程与 f 的插值过程相同。

——四分之一样本 e , g , p 和 r :

$$e = \text{Clip}((D' + j + 64) \gg 7)$$

$$g = \text{Clip}((E' + j + 64) \gg 7)$$

$$p = \text{Clip}((H' + j + 64) \gg 7)$$

$$r = \text{Clip}((I' + j + 64) \gg 7)$$

其中 D' , E' , H' 和 I' 是相应位置整数样本放大 64 倍的值, j 是相应位置二分之一样本中间值。

图19中当前亮度块E的左上角整像素位置为 (x, y) , 预测样本矩阵的元素 $\text{predMatrix}[x, y]$ 根据表24赋值。

表 24 预测样本矩阵元素

xFracL	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
yFracL	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
predMatrix [x, y]	D	d	h	n	a	e	i	p	b	f	j	q	c	g	k	r

表24中xFracL等于mvE的水平分量mvE_x & 3, yFracL等于mvE的垂直分量mvE_y & 3。

9.9.2.2 色度样本插值过程

色度样本插值使用与对应亮度块的运动矢量mvE (mvE的水平分量为mvE_x, 垂直分量为mvE_y) 对应的运动矢量mvC。mvC的水平分量为mvC_x, 垂直分量为mvC_y, mvC的基本单位为八分之一样本。如果

chroma_format的值为‘01’，则mvC_x的值等于mvE_x，mvC_y的值等于mvE_y；如果chroma_format的值为‘10’，则mvC_x的值等于mvE_x，mvC_y的值等于mvE_y × 2。色度样本插值如图20所示，A，B，C，D是被插值样本周围的整数样本值，dx与dy分别是预测样本与A的水平距离，dx等于mvC_x & 7，dy等于mvC_y & 7，这里各变量与参考样本的位置关系如图20所示。

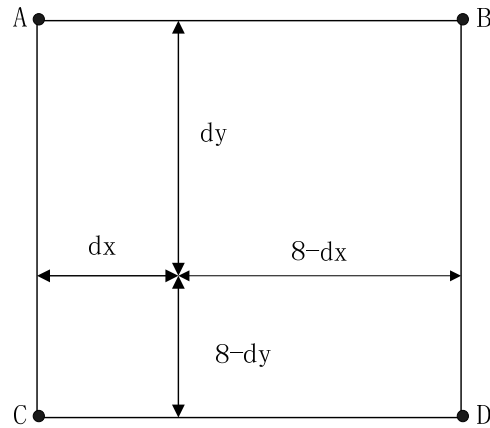


图 20 八分之一色度插值

预测样本矩阵的元素predMatrix[x, y]根据下式计算：

$$\text{predMatrix}[x, y] = \text{Clip1}(((8-dx) \times (8-dy) \times A + dx \times (8-dy) \times B + (8-dx) \times dy \times C + dx \times dy \times D + 32) \gg 6)$$

6)

9.9.3 加权预测

亮度样本加权预测如下：

$$\text{predMatrix}[x, y] = \text{Clip1}((\text{predMatrix}[x, y] \times \text{luma_scale} + 16) \gg 5 + \text{luma_shift})$$

色度样本加权预测如下：

$$\text{predMatrix}[x, y] = \text{Clip1}((\text{predMatrix}[x, y] \times \text{chroma_scale} + 16) \gg 5 + \text{chroma_shift})$$

9.10 重建

如果当前块是I或P宏块类型，重建系数矩阵RecMatrix计算如下：

$$\text{RecMatrix}[x, y] = \text{Clip1}(\text{predMatrix}[x, y] + \text{ResidueMatrix}[x, y])$$

如果当前块是B宏块类型，并且有两帧参考图像，重建系数矩阵RecMatrix计算如下：

$$\text{RecMatrix}[x, y] = \text{Clip1}((\text{predMatrixFw}[x, y] + \text{predMatrixBw}[x, y] + 1) \gg 1 + \text{ResidueMatrix}[x, y])$$

如果当前块是B宏块类型，并且只有前向参考图像，重建系数矩阵RecMatrix计算如下：

$$\text{RecMatrix}[x, y] = \text{Clip1}((\text{predMatrixFw}[x, y] + \text{ResidueMatrix}[x, y])$$

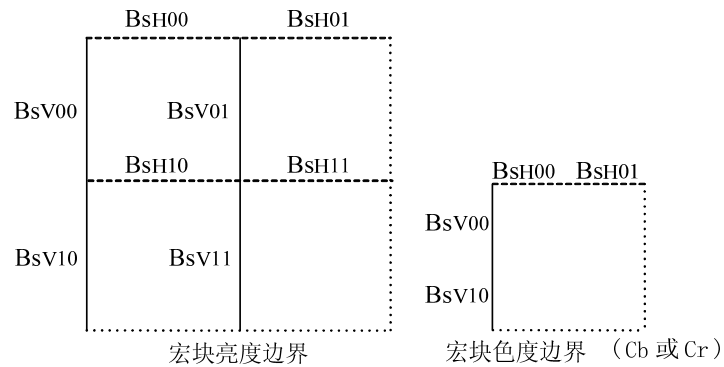
如果当前块是B宏块类型，并且只有后向参考图像，重建系数矩阵RecMatrix计算如下：

$$\text{RecMatrix}[x, y] = \text{Clip1}((\text{predMatrixBw}[x, y] + \text{ResidueMatrix}[x, y])$$

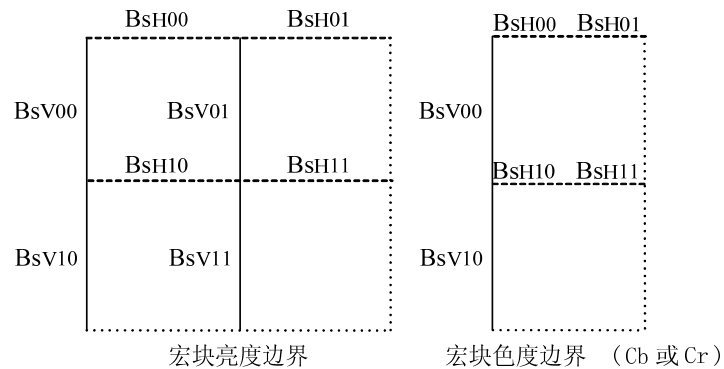
其中predMatrixFw是前向参考图像的预测样本矩阵，predMatrixBw是后向参考图像的预测样本矩阵。

9.11 环路滤波

每个8×8亮度块之间的边界有一个“边界强度”Bs，色度块的边界强度用对应位置亮度块边界的Bs代替，如图21所示。如果Bs等于0则不要对边界滤波，否则根据局部样本值的特性和Bs值对边界滤波。



a) 4:2:0格式



b) 4:2:2格式

注：粗实线为垂直边界，粗虚线为水平边界。

图 21 宏块中需要滤波的边界示意

除图像边界及条带的边界之外，宏块的所有边界都应进行滤波。此处宏块边界定义为宏块内部各个8×8块的边界，以及当前宏块与相邻宏块的上边界和左边界。

环路滤波以宏块为单位，按照光栅扫描顺序依次处理。图像中每个宏块的滤波过程如下：

对亮度和色度分别做环路滤波，如图21所示，首先从左到右对垂直边界滤波，然后从上到下对水平边界滤波。当前宏块的上边或者左边的样本值可能在以前的宏块环路滤波过程中已经被修改，当前宏块的环路滤波的输入为这些可能被修改的样本值，并且当前宏块环路滤波可能进一步修改这些样本值。当前宏块垂直边界滤波过程中修改的样本值作为水平边界滤波过程的输入。

帧内预测使用环路滤波前的重建图像样本值。

9.11.1 边界滤波强度的推导过程

根据宏块类型、宏块中8×8亮度块的运动矢量，按如下方法计算Bs的值：

- 如果边界两边的两个8×8块有一个或两个都属于帧内预测宏块，则Bs等于2；
 - 否则，如果picture_coding_type的值为‘01’并且满足以下两个条件中的任一个，则Bs等于1；如果picture_coding_type的值为‘01’并且以下两个条件都不满足，则Bs等于0；
- 1) 两个块的参考图像不同。

- 2) 两个块的参考图像相同,但是两个运动矢量分量中任一个分量的差值大于或等于一个整像素。

——否则,按以下方法计算Bs的值:

- 1) 如果两个块p和q的前向索引参考值和后向索引参考值分别相等:
 - ◆ 以下两个条件中任一个成立,则Bs等于1:
 - I) 两个块的前向运动矢量分量中任一个分量的差值大于或等于一个整像素。
 - II) 两个块的后向运动矢量分量中任一个分量的差值大于或等于一个整像素。
 - ◆ 否则,Bs等于0;
- 2) 否则,Bs等于1。

9.11.2 块边界阈值的推导过程

图22表示块p和块q在水平或垂直边界两侧的6个样本点(边界用黑色粗线表示)。用P₀、P₁、Q₀和Q₁分别表示p₀、p₁、q₀和q₁滤波后的样本值。

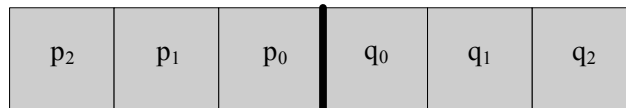


图 22 8×8 块水平或垂直边界样本

如果下式为真,对边界样本滤波:

$$Bs \neq 0 \ \&\& \ Abs(p_0 - q_0) < \alpha \ \&\& \ Abs(p_1 - p_0) < \beta \ \&\& \ Abs(q_1 - q_0) < \beta$$

其中α和β为块边界阈值,可以根据两个块的QP平均值QP_{av},以及AlphaOffset和BetaOffset计算查表索引IndexA和IndexB。

两个块的QP平均值QP_{av}为:

$$QP_{av} = (QP_p + QP_q + 1) \gg 1$$

索引IndexA和IndexB为:

$$IndexA = Clip3(0, 63, QP_{av} + AlphaOffset)$$

$$IndexB = Clip3(0, 63, QP_{av} + BetaOffset)$$

根据索引IndexA和IndexB与阈值α和β间的对应关系,由表25得到α、β的取值,根据IndexA查表得到α,根据IndexB查表得到β。

表 25 块边界阈值α和β与 IndexA 和 IndexB 的关系

索引	α	β	索引	α	β	索引	α	β	索引	α	β
0	0	0	16	4	2	32	22	6	48	46	15
1	0	0	17	4	2	33	24	7	49	48	16
2	0	0	18	5	3	34	26	7	50	50	17
3	0	0	19	5	3	35	28	7	51	52	18
4	0	0	20	6	3	36	30	8	52	53	19
5	0	0	21	7	3	37	33	8	53	54	20
6	1	1	22	8	4	38	33	8	54	55	21
7	1	1	23	9	4	39	35	9	55	56	22
8	1	1	24	10	4	40	35	9	56	57	23
9	1	1	25	11	4	41	36	10	57	58	23

表 25 (续)

索引	α	β	索引	α	β	索引	α	β	索引	α	β
10	1	1	26	12	5	42	37	10	58	59	24
11	2	1	27	13	5	43	37	11	59	60	24
12	2	1	28	15	5	44	39	11	60	61	25
13	2	2	29	16	5	45	39	12	61	62	25
14	3	2	30	18	6	46	42	13	62	63	26
15	3	2	31	20	6	47	44	14	63	64	27

9.11.3 B_s 等于 2 时的边界滤波过程

首先定义 $ap = \text{Abs}(p_2 - p_0)$, $aq = \text{Abs}(q_2 - q_0)$ 。

对亮度块边界两边的样本 p_0 、 p_1 、 q_0 和 q_1 的滤波过程如下：

```

if ( ap <  $\beta$  && Abs( p0 - q0 ) < (( $\alpha$  >> 2) + 2) ) {
    P0 = ( p1 + 2 × p0 + q0 + 2 ) >> 2
    P1 = ( 2 × p1 + p0 + q0 + 2 ) >> 2
}
else
    P0 = ( 2 × p1 + p0 + q0 + 2 ) >> 2
if ( aq <  $\beta$  && Abs( p0 - q0 ) < (( $\alpha$  >> 2) + 2) ) {
    Q0 = ( q1 + 2 × q0 + p0 + 2 ) >> 2
    Q1 = ( 2 × q1 + q0 + p0 + 2 ) >> 2
}
else
    Q0 = ( 2 × q1 + q0 + p0 + 2 ) >> 2
    
```

其中 P_0 和 Q_0 分别为 p_0 和 q_0 滤波后的值。如果在上面的滤波过程中 P_1 (Q_1) 不被赋值，则不对 p_1 (q_1) 滤波；否则， P_1 或 Q_1 为 p_1 或 q_1 滤波后的值。

色度块边界两边的样本 p_0 和 q_0 的采用同样的方法滤波，不对 p_1 和 q_1 滤波。

9.11.4 B_s 等于 1 时的边界滤波过程

边界滤波强度 B_s 的值为 1 时，对 p_0 和 q_0 滤波的计算过程如下 (P_0 和 Q_0 分别为 p_0 和 q_0 滤波后的值)：

```

delta = Clip3(-C, C, (((q0 - p0) × 3 + (p1 - q1) + 4) >> 3))
P0 = Clip1(p0 + delta)
Q0 = Clip1(q0 - delta)
    
```

然后判断是否需要 p_1 和 q_1 滤波，计算过程如下：

——如果为色度边界，不对 p_1 和 q_1 滤波。

——如果在亮度边界处有 ap 小于 β ，则对 p_1 滤波，滤波后的值为

```

P1 = Clip1(p1 + Clip3(-C, C, (((P0 - p1) × 3 + (p2 - Q0) + 4) >> 3)))
    
```

——如果在亮度边界处有 aq 小于 β ，则对 q_1 滤波，滤波后的值为

```

Q1 = Clip1(q1 - Clip3(-C, C, (((q1 - Q0) × 3 + (P0 - q2) + 4) >> 3)))
    
```

上述滤波过程中， ap 和 aq 的定义见 9.11.3， C 称为滤波裁减参数， C 与 $IndexA$ 之间的关系见

表 26。

表 26 滤波裁减参数 C 与 IndexA 的关系

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

附 录 A
(规范性附录)
变长码表

本附录给出了变长码表。表A.1~A.20中Run表示量化系数游程，Level表示量化系数值，EOB栏及Level栏的各列数据表示语法元素trans_coefficient的值。解码时根据trans_coefficient可以查表得到Level和Run。表中EOB栏对应的数字表示代表EOB的trans_coefficient的值。

表 A.1 VLC0_Intra (用于解码帧内编码亮度块的游程和非零量化系数值的映射表)

Run	Level>0			RefAbsLevel	Run	Level>0			RefAbsLevel
	1	2	3			1	2	3	
0	0	22	38	4	12	26	-	-	2
1	2	32	-	3	13	28	-	-	2
2	4	44	-	3	14	30	-	-	2
3	6	50	-	3	15	34	-	-	2
4	8	54	-	3	16	36	-	-	2
5	10	-	-	2	17	40	-	-	2
6	12	-	-	2	18	42	-	-	2
7	14	-	-	2	19	46	-	-	2
8	16	-	-	2	20	48	-	-	2
9	18	-	-	2	21	52	-	-	2
10	20	-	-	2	22	56	-	-	2
11	24	-	-	2					

表 A.2 VLC1_Intra (用于解码帧内编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0						RefAbsLevel
		1	2	3	4	5	6	
	8	-	-	-	-	-	-	
0	-	0	4	15	27	41	55	7
1	-	2	17	35	-	-	-	4
2	-	6	25	53	-	-	-	4
3	-	9	33	-	-	-	-	3
4	-	11	39	-	-	-	-	3
5	-	13	45	-	-	-	-	3
6	-	19	49	-	-	-	-	3
7	-	21	51	-	-	-	-	3
8	-	23	-	-	-	-	-	2
9	-	29	-	-	-	-	-	2

表 A.2 (续)

Run	EOB	Level > 0						RefAbsLevel
		1	2	3	4	5	6	
	8	-	-	-	-	-	-	
10	-	31	-	-	-	-	-	2
11	-	37	-	-	-	-	-	2
12	-	43	-	-	-	-	-	2
13	-	47	-	-	-	-	-	2
14	-	57	-	-	-	-	-	2

表 A.3 VLC2_Intra (用于解码帧内编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0									RefAbsLevel
		1	2	3	4	5	6	7	8	9	
	8	-	-	-	-	-	-	-	-	-	
0	-	0	2	6	13	17	27	35	45	55	10
1	-	4	11	21	33	49	-	-	-	-	6
2	-	9	23	37	-	-	-	-	-	-	4
3	-	15	29	51	-	-	-	-	-	-	4
4	-	19	39	-	-	-	-	-	-	-	3
5	-	25	43	-	-	-	-	-	-	-	3
6	-	31	53	-	-	-	-	-	-	-	3
7	-	41	-	-	-	-	-	-	-	-	2
8	-	47	-	-	-	-	-	-	-	-	2
9	-	57	-	-	-	-	-	-	-	-	2

表 A.4 VLC3_Intra (用于解码帧内编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0												RefAbsLevel
		1	2	3	4	5	6	7	8	9	10	11	12	
	8	-	-	-	-	-	-	-	-	-	-	-	-	
0	-	0	2	4	9	11	17	21	25	33	39	45	55	13
1	-	6	13	19	29	35	47	-	-	-	-	-	-	7
2	-	15	27	41	57	-	-	-	-	-	-	-	-	5
3	-	23	37	53	-	-	-	-	-	-	-	-	-	4
4	-	31	51	-	-	-	-	-	-	-	-	-	-	3
5	-	43	-	-	-	-	-	-	-	-	-	-	-	2
6	-	49	-	-	-	-	-	-	-	-	-	-	-	2

表 A.5 VLC4_Intra (用于解码帧内编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0																	RefAbsLevel
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
	6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
0	-	0	2	4	7	9	11	15	17	21	23	29	33	35	43	47	49	57	18
1	-	13	19	27	31	37	45	55	-	-	-	-	-	-	-	-	-	-	8
2	-	25	41	51	-	-	-	-	-	-	-	-	-	-	-	-	-	-	4
3	-	39	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2
4	-	53	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2

表 A.6 VLC5_Intra (用于解码帧内编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0																					RefAbsLevel
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	
	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
0	-	1	3	5	7	9	11	13	15	17	19	23	25	27	31	33	37	41	45	49	51	55	22
1	-	21	29	35	43	47	53	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	7
2	-	39	57	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3

表 A.7 VLC6_Intra (用于解码帧内编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0																					RefAbsLevel
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	
	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
0	-	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	35	37	39	41	43	27
1	-	33	45	55	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	4
		Level > 0																					
		22	23	24	25	26	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
0	-	47	49	51	53	57	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	27
1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

表 A.8 VLC0_Inter (用于解码帧间编码亮度块的游程和非零量化系数值的映射表)

Run	Level>0			RefAbsLevel	Run	Level>0			RefAbsLevel
	1	2	3			1	2	3	
0	0	26	40	4	4	8	-	-	2
1	2	46	-	3	5	10	-	-	2
2	4	-	-	2	6	12	-	-	2
3	6	-	-	2	7	14	-	-	2

表 A. 8 (续)

Run	Level>0			RefAbsLevel	Run	Level>0			RefAbsLevel
	1	2	3			1	2	3	
8	16	-	-	2	17	36	-	-	2
9	18	-	-	2	18	38	-	-	2
10	20	-	-	2	19	42	-	-	2
11	22	-	-	2	20	44	-	-	2
12	24	-	-	2	21	48	-	-	2
13	28	-	-	2	22	50	-	-	2
14	30	-	-	2	23	52	-	-	2
15	32	-	-	2	24	54	-	-	2
16	34	-	-	2	25	56	-	-	2

表 A. 9 VLC1_Inter (用于解码帧间编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0				RefAbsLevel
		1	2	3	4	
	2	-	-	-	-	
0	-	0	13	29	47	5
1	-	3	23	57	-	4
2	-	5	35	-	-	3
3	-	7	39	-	-	3
4	-	9	43	-	-	3
5	-	11	49	-	-	3
6	-	15	55	-	-	3
7	-	17	-	-	-	2
8	-	19	-	-	-	2
9	-	21	-	-	-	2
10	-	25	-	-	-	2
11	-	27	-	-	-	2
12	-	31	-	-	-	2
13	-	33	-	-	-	2
14	-	37	-	-	-	2
15	-	41	-	-	-	2
16	-	45	-	-	-	2
17	-	51	-	-	-	2
18	-	53	-	-	-	2

表 A.10 VLC2_Inter (用于解码帧间编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0						RefAbsLevel
		1	2	3	4	5	6	
	2	-	-	-	-	-	-	
0	-	0	5	11	23	35	47	7
1	-	3	13	27	49	-	-	5
2	-	7	21	45	-	-	-	4
3	-	9	29	55	-	-	-	4
4	-	15	37	-	-	-	-	3
5	-	17	41	-	-	-	-	3
6	-	19	53	-	-	-	-	3
7	-	25	-	-	-	-	-	2
8	-	31	-	-	-	-	-	2
9	-	33	-	-	-	-	-	2
10	-	39	-	-	-	-	-	2
11	-	43	-	-	-	-	-	2
12	-	51	-	-	-	-	-	2
13	-	57	-	-	-	-	-	2

表 A.11 VLC3_Inter (用于解码帧间编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0									RefAbsLevel
		1	2	3	4	5	6	7	8	9	
	2	-	-	-	-	-	-	-	-	-	
0	-	0	3	7	13	17	27	35	43	55	10
1	-	5	11	21	33	51	-	-	-	-	6
2	-	9	23	37	57	-	-	-	-	-	5
3	-	15	29	47	-	-	-	-	-	-	4
4	-	19	41	-	-	-	-	-	-	-	3
5	-	25	49	-	-	-	-	-	-	-	3
6	-	31	-	-	-	-	-	-	-	-	2
7	-	39	-	-	-	-	-	-	-	-	2
8	-	45	-	-	-	-	-	-	-	-	2
9	-	53	-	-	-	-	-	-	-	-	2

表 A.12 VLC4_Inter (用于解码帧间编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0												RefAbsLevel
		1	2	3	4	5	6	7	8	9	10	11	12	
	2	-	-	-	-	-	-	-	-	-	-	-	-	
0	-	0	3	5	9	11	17	21	25	33	41	45	55	13
1	-	7	13	19	29	35	49	-	-	-	-	-	-	7
2	-	15	27	43	57	-	-	-	-	-	-	-	-	5

表 A.12 (续)

Run	EOB	Level > 0												RefAbsLevel	
		1	2	3	4	5	6	7	8	9	10	11	12		
	2	-	-	-	-	-	-	-	-	-	-	-	-		-
3	-	23	37	51	-	-	-	-	-	-	-	-	-	-	4
4	-	31	53	-	-	-	-	-	-	-	-	-	-	-	3
5	-	39	-	-	-	-	-	-	-	-	-	-	-	-	2
6	-	47	-	-	-	-	-	-	-	-	-	-	-	-	2

表 A.13 VLC5_Inter (用于解码帧间编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0																RefAbsLevel
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
0	-	1	3	5	7	9	13	15	17	21	25	29	33	39	43	49	53	17
1	-	11	19	27	31	41	45	57	-	-	-	-	-	-	-	-	-	8
2	-	23	37	51	-	-	-	-	-	-	-	-	-	-	-	-	-	4
3	-	35	55	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3
4	-	47	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2

表 A.14 VLC6_Inter (用于解码帧间编码亮度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0																				RefAbsLevel	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		21
	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-		-
0	-	1	3	5	7	9	11	13	17	19	21	23	25	29	33	35	39	41	43	47	49	57	22
1	-	15	27	37	45	55	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	6
2	-	31	51	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3
3	-	53	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2

表 A.15 VLC0_Chroma (用于解码色度块的游程和非零量化系数值的映射表)

Run	Level>0				RefAbsLevel	Run	Level>0				RefAbsLevel
	1	2	3	4			1	2	3	4	
0	0	14	32	56	5	6	12	-	-	-	2
1	2	48	-	-	3	7	16	-	-	-	2
2	4	-	-	-	2	8	18	-	-	-	2
3	6	-	-	-	2	9	20	-	-	-	2
4	8	-	-	-	2	10	22	-	-	-	2
5	10	-	-	-	2	11	24	-	-	-	2

表 A. 15 (续)

Run	Level>0				RefAbsLevel	Run	Level>0				RefAbsLevel
	1	2	3	4			1	2	3	4	
12	26	-	-	-	2	19	42	-	-	-	2
13	28	-	-	-	2	20	44	-	-	-	2
14	30	-	-	-	2	21	46	-	-	-	2
15	34	-	-	-	2	22	50	-	-	-	2
16	36	-	-	-	2	23	52	-	-	-	2
17	38	-	-	-	2	24	54	-	-	-	2
18	40	-	-	-	2						

表 A. 16 VLC1_Chroma (用于解码色度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0					RefAbsLevel
		1	2	3	4	5	
	0	-	-	-	-	-	
0	-	1	5	15	29	43	6
1	-	3	21	45	-	-	4
2	-	7	37	-	-	-	3
3	-	9	41	-	-	-	3
4	-	11	53	-	-	-	3
5	-	13	-	-	-	-	2
6	-	17	-	-	-	-	2
7	-	19	-	-	-	-	2
8	-	23	-	-	-	-	2
9	-	25	-	-	-	-	2
10	-	27	-	-	-	-	2
11	-	31	-	-	-	-	2
12	-	33	-	-	-	-	2
13	-	35	-	-	-	-	2
14	-	39	-	-	-	-	2
15	-	47	-	-	-	-	2
16	-	49	-	-	-	-	2
17	-	51	-	-	-	-	2
18	-	55	-	-	-	-	2
19	-	57	-	-	-	-	2

表 A.17 VLC2_Chroma (用于解码色度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0									RefAbsLevel
		1	2	3	4	5	6	7	8	9	
	2	-	-	-	-	-	-	-	-	-	
0	-	0	3	7	11	17	27	33	47	53	10
1	-	5	13	21	37	55	-	-	-	-	6
2	-	9	23	41	-	-	-	-	-	-	4
3	-	15	31	57	-	-	-	-	-	-	4
4	-	19	43	-	-	-	-	-	-	-	3
5	-	25	45	-	-	-	-	-	-	-	3
6	-	29	-	-	-	-	-	-	-	-	2
7	-	35	-	-	-	-	-	-	-	-	2
8	-	39	-	-	-	-	-	-	-	-	2
9	-	49	-	-	-	-	-	-	-	-	2
10	-	51	-	-	-	-	-	-	-	-	2

表 A.18 VLC3_Chroma (用于解码色度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0													RefAbsLevel
		1	2	3	4	5	6	7	8	9	10	11	12	13	
	0	-	-	-	-	-	-	-	-	-	-	-	-	-	
0	-	1	3	5	7	11	15	19	23	29	35	43	47	53	14
1	-	9	13	21	31	39	51	-	-	-	-	-	-	-	7
2	-	17	27	37	-	-	-	-	-	-	-	-	-	-	4
3	-	25	41	-	-	-	-	-	-	-	-	-	-	-	3
4	-	33	55	-	-	-	-	-	-	-	-	-	-	-	3
5	-	45	-	-	-	-	-	-	-	-	-	-	-	-	2
6	-	49	-	-	-	-	-	-	-	-	-	-	-	-	2
7	-	57	-	-	-	-	-	-	-	-	-	-	-	-	2

表 A.19 VLC4_Chroma (用于解码色度块的游程和非零量化系数值的映射表)

Run	EOB	Level > 0																		RefAbsLevel	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18		19
	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-		-
0	-	1	3	5	7	9	11	13	15	19	21	23	27	29	33	37	41	43	51	55	20
1	-	17	25	31	39	45	53	-	-	-	-	-	-	-	-	-	-	-	-	-	7
2	-	35	49	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3
3	-	47	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2
4	-	57	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2

表 A. 20 CurrentVLCTable 和 MaxRun 的映射关系

CurrentVLCTable	MaxRun	CurrentVLCTable	MaxRun	CurrentVLCTable	MaxRun
VLC0_Intra	22	VLC0_Inter	25	VLC0_Chroma	24
VLC1_Intra	14	VLC1_Inter	18	VLC1_Chroma	19
VLC2_Intra	9	VLC2_Inter	13	VLC2_Chroma	10
VLC3_Intra	6	VLC3_Inter	9	VLC3_Chroma	7
VLC4_Intra	4	VLC4_Inter	6	VLC4_Chroma	4
VLC5_Intra	2	VLC5_Inter	4		
VLC6_Intra	1	VLC6_Inter	3		

附 录 B

(规范性附录)

档次和级别

档次和级别提供了一种定义GB/T 20090的本部分的语法和语义的子集的手段。档次和级别对比特流进行了各种限制，同时也就规定了对某一特定比特流解码所需要的解码器能力。档次是本部分规定的语法、语义及算法的子集。符合某个档次规定的解码器必须完全支持该档次定义的子集。级别是在某一档次下对语法元素和语法元素参数值的限定集合。在给定档次的情况下，不同级别往往意味着对解码器能力和存储器容量的不同要求。

本附录描述了不同档次和级别所对应的各种限制。所有未被限定的语法元素和参数可以取任何本部分所允许的值。如果一个解码器能对某个档次和级别所规定的语法元素的所有允许值正确解码，则称此解码器在这个档次和级别上符合本部分。如果一个比特流中不存在某个档次和级别所不允许的语法元素，并且其所含有的语法元素的值不超过此档次和级别所允许的范围，则认为此比特流在这个档次和级别上符合本部分。

profile_id和level_id定义了比特流的档次和级别。

B.1 档次

本部分定义的档次见表B.1。

表 B.1 档次

profile_id	档次
0x00	禁止
0x20	基准档次 (Jizhun profile)
其他	保留

对于一个给定的档次，不同的级别支持相同的语法子集。

基准档次的比特流应满足以下条件：

- profile_id 的值应为 0x20。
- advanced_pred_mode_disable 的值应为 ‘1’。
- chroma_format 的值为 ‘01’ 或 ‘10’。
- B.2.2 规定的级别限制。

基准档次支持的级别包括：4.0、4.2、6.0和6.2。

B.2 级别

B.2.1 本部分定义的级别

本部分定义的级别见表B.2。

表 B.2 级别

level_id	级别
0x00	禁止
0x10	2.0
0x20	4.0

表 B.2 (续)

level_id	级别
0x22	4.2
0x40	6.0
0x42	6.2
其他	保留

B.2.2 与档次无关的级别限制

对于所有档次，每个宏块编码后最大比特数的限制见表B.3。

表 B.3 宏块编码后最大比特数

图像格式	宏块编码后最大比特数
4:2:0	$128 + 256 \times 8 \times 1.5 = 3200$
4:2:2	$128 + 256 \times 8 \times 2 = 4224$

表B.4、B.5和B.6给出了其他限制。

表 B.4 级别的参数限制

参 数	级 别
	2.0
每行最大样本数	352
每帧最大行数	288
每秒最大帧数	30
亮度样本速率	2,534,400
最大比特率 (比特每秒)	1,000,000
BBV缓冲区大小 (比特)	122,880
每帧最大宏块个数	396
每秒最大宏块个数	11,880
帧编码时最大垂直运动矢量范围 (亮度样本数)	[-128, +127.75]
场编码时最大垂直运动矢量范围 (亮度样本数)	-
最大水平运动矢量范围 (亮度样本数)	[-2048, +2047.75]
图像格式	4:2:0

表 B.5 级别的参数限制

参 数	级 别	
	4.0	4.2
每行最大样本数	720	720
每帧最大行数	576	576
每秒最大帧数	30	30
亮度样本速率	10,368,000	10,368,000
最大比特率（比特每秒）	10,000,000	15,000,000
BBV缓冲区大小（比特）	1,228,800	1,851,392
每帧最大宏块个数	1,620	1,620
每秒最大宏块个数	40,500	40,500
帧编码时最大垂直运动矢量范围（亮度样本数）	[-256, +255.75]	[-256, +255.75]
场编码时最大垂直运动矢量范围（亮度样本数）	[-128, +127.75]	[-128, +127.75]
最大水平运动矢量范围（亮度样本数）	[-2048, +2047.75]	[-2048, +2047.75]
图像格式	4:2:0	4:2:0或4:2:2

表 B.6 级别的参数限制

参 数	级 别	
	6.0	6.2
每行最大样本数	1,920	1,920
每帧最大行数	1,152	1,152
每秒最大帧数	60	60
亮度样本速率	62,668,800	62,668,800
最大比特率（比特每秒）	20,000,000	30,000,000
BBV缓冲区大小（比特）	2,457,600	3,686,400
每帧最大宏块个数	8,160	8,160
每秒最大宏块个数	244,800	244,800
帧编码时最大垂直运动矢量范围（亮度样本数）	[-512, +511.75]	[-512, +511.75]
场编码时最大垂直运动矢量范围（亮度样本数）	[-256, +255.75]	[-256, +255.75]
最大水平运动矢量范围	[-2048, +2047.75]	[-2048, +2047.75]
图像格式	4:2:0	4:2:0或4:2:2

注1:某一级别下BBV缓冲区大小与该级别所允许的最大比特率成正比,向上取整为最接近16384比特的倍数的数。变

化的参照值为基准档次4.0级别的缓冲区大小。例如基准档次6.0级别的缓冲区大小为
 $\text{Ceil}((1228800 \times 20 \div 10) \div 16384) \times 16384 = 2457600$ 比特。

注2：与表B.4、表B.5和表B.6有关的语法元素包括：`horizontal_size`、`vertical_size`、`frame_rate_code`、`bbv_buffer_size`、`chroma_format`。

附 录 C
(规范性附录)
伪起始码

本附录定义防止在比特流中出现伪起始码的方法。起始码的形式、含义，以及为了使起始码字节对齐而进行填充的方法见7.1.1和5.7.2。

为了防止出现伪起始码，编码时应按照以下方法处理：写入一位时，如果该位是一个字节的第二最低有效位，检查该位之前写入的22位，如果这22位都是‘0’，在该位之前插入‘10’，该位成为下一个字节的最高有效位。

解码时应按以下方法处理：每读入一个字节时，检查前面读入的两个字节和当前字节，如果这三个字节构成位串‘0000 0000 0000 0000 0000 0010’，丢弃当前字节的最低两个有效位。丢弃一个字节最低两个有效位可采用任意等效的方式，GB/T 20090的本部分不做规定。

在编码和解码时对于序列头、序列显示扩展、版权扩展、用户数据、摄像机参数扩展中的数据不应采用上述方法。

附 录 D
(规范性附录)
比特流虚拟参考解码器

本附录定义了比特流虚拟参考解码器（以下简称BBV）。

BBV有一个输入缓冲区，称为BBV缓冲区。编码数据按照D. 2. 1定义的方式进入BBV缓冲区，按照D. 2. 2定义的方式移出BBV缓冲区。符合GB/T 20090的本部分的比特流不应导致BBV缓冲区上溢。如果low_delay的值为‘0’，符合本部分的比特流不应导致BBV缓冲区下溢；如果low_delay的值为‘1’，符合本部分的比特流可能导致BBV缓冲区下溢，此时应按D. 2. 2. 2定义的方式处理。

本附录中所有的运算都是实数运算，不存在舍入误差，例如BBV缓冲区中的比特数不必是整数。

D. 1 约定

D. 1. 1 约定一

BBV和视频编码器的时钟频率和帧率相同，并且同步操作。

D. 1. 2 约定二

BBV缓冲区的大小为BBS。

D. 1. 3 约定三

编码数据输入BBV缓冲区的最大速率 R_{\max} （单位为比特每秒）按下式计算：

$$R_{\max} = \text{BitRate} \times 400 \quad (\text{D. 1})$$

D. 2 基本操作

D. 2. 1 数据输入

本附录定义了两种方法来计算编码数据进入BBV缓冲区的速率。这两种方法不应同时使用。

D. 2. 1. 1 方法一

如果bbv_delay的值不等于0xFFFF，第 n 帧图像进入BBV缓冲区的速率 $R(n)$ 按下式计算

$$R(n) = d_n^* \div (\tau(n) - \tau(n+1) + t(n+1) - t(n)) \quad (\text{D. 2})$$

式中：

d_n^* ——从第 n 帧的起始码后第1个比特到第 $n+1$ 帧的起始码后第1个比特之间所有的比特数；

$\tau(n)$ ——第 n 帧的bbv_delay的值，单位为秒（s）；

$t(n)$ ——第 n 帧图像的编码数据从BBV缓冲区移出的时间，单位为秒（s）；

$t(n+1) - t(n)$ ——第 $n+1$ 帧和第 n 帧图像的解码时间间隔，单位为秒（s）。

在视频序列开始和结束时，可能缺少参数来无歧义地确定这个时间间隔，此时可采用下面的方法来处理。

D. 2. 1. 1. 1 视频序列开始时的歧义性

对视频序列进行随机访问时，序列头后的第1帧图像缺少在此之前的I帧或P帧。在这种情况下，如果比特流是系统流的一部分，可从系统流来确定解码时间间隔。

如果还不能无歧义地确定解码时间间隔，就无法确定 $R(n)$ 。此时在一段有限的时间内（这个时间总是小于bbv_delay的最大值）BBV不能准确地确定充满度的变化轨迹，从而无法在整个比特流中严格地验证BBV缓冲区。编码器总是知道在每个重复序列头后 $t(n+1) - t(n)$ 的值，因此也知道如何生成不违反BBV限制的比特流。

D. 2. 1. 1. 2 视频序列结束时的歧义性

如果一帧图像的编码数据后第1个起始码是视频序列结束码，此时无法确定该帧图像编码数据的比特数。在这种情况下，应存在一个输入速率，这个速率不应导致BBV缓冲区上溢；在low_delay的值为‘1’时，这个速率也不应导致缓冲区下溢。该速率应小于在视频序列头中定义的最大速率。

视频序列的第1帧图像的起始码前的所有数据和这个起始码输入BBV缓冲区后，每一帧图像的编码数据应在由比特流中的bbv_delay规定的时间内输入BBV缓冲区，并在这个时间开始解码处理。输入速率由公式D.2确定。

所有比特流的 $R(n)$ 均不应大于 R_{max} 。

在CBR情况下，视频序列中的 $R(n)$ 在bbv_delay允许的精度范围内是一个常数。

D.2.1.2 方法二

如果bbv_delay的值为0xFFFF，编码数据按以下方式输入BBV缓冲区：

如果BBV缓冲区没有充满，数据以速率 R_{max} 输入缓冲区；如果BBV缓冲区充满，则数据不能进入该缓冲区直到缓冲区中的部分数据被移出。

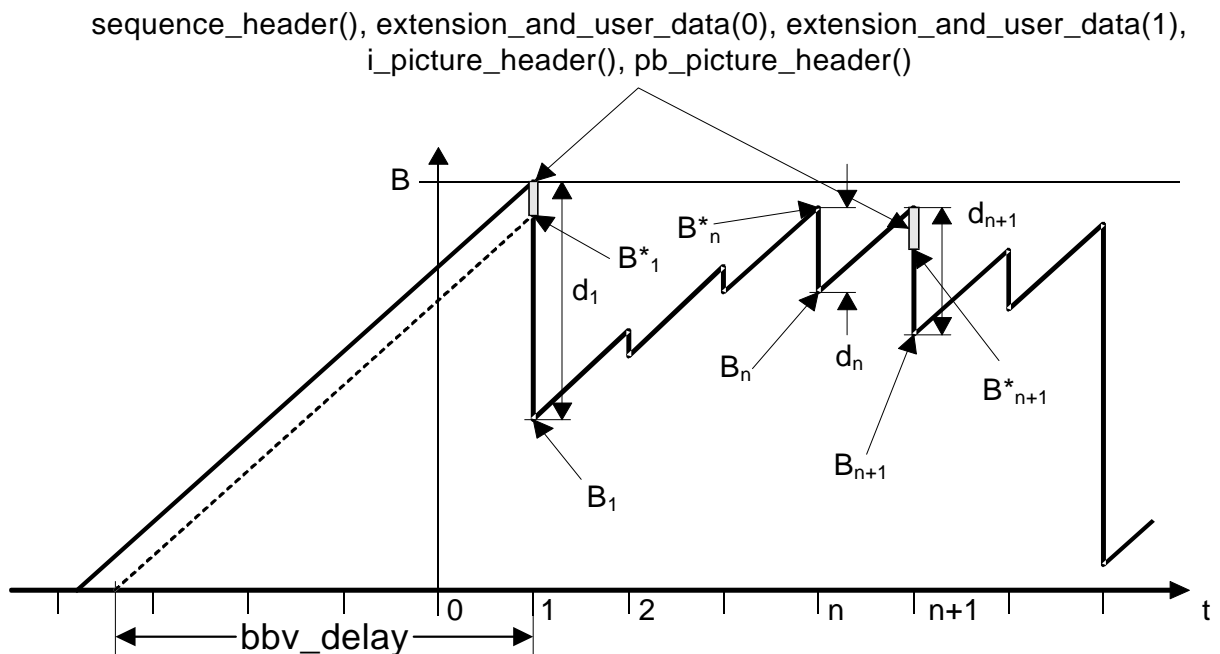
视频序列的第1帧图像的起始码前的所有数据和这个起始码输入BBV缓冲区后，数据继续输入BBV缓冲区直到缓冲区充满，并在这个时间开始解码处理。

D.2.2 数据移出

在BBV中，第 n 帧图像的编码数据 $f(n)$ 包括以下数据（如果存在）：

- 序列头、跟在序列头之后的扩展和用户数据、视频编辑码。这些数据与本帧图像起始码之间不应有其他图像的数据。
- 本帧图像的所有编码数据。
- 跟在本帧图像头后的图像显示扩展。
- 跟在本帧图像后的填充数据、视频序列结束码。

具体见图 D.1 。



图D.1 BBV缓冲区占用情况（CBR）

D.2.2.1 非低延迟

low_delay的值为‘0’时，数据按照下面的方式从BBV缓冲区中移出：在每帧图像的解码时刻 $t(n)$ ，如果BBV缓冲区充满度 $B(n)$ 小于 $f(n)$ 则发生下溢，否则移出该帧图像编码数据并解码。

如果low_delay的值为‘0’，BBV缓冲区不应发生下溢。

D.2.2.2 低延迟

low_delay的值为‘1’时，在移出一帧图像的编码数据之前，要检测BBV缓冲区BbvCheckTimes加1次。如果BbvCheckTimes大于0，当前解码图像定义为大图像。

数据按照下面的方式从BBV缓冲区中移出：每隔一个检测时间间隔检测一次BBV缓冲区，完成BbvCheckTimes加1次检测后移出图像编码数据。

如果当前解码图像不是大图像，检测时间间隔在D.3定义。

如果当前解码图像是大图像，检测时间间隔是一个帧时间间隔。移出大图像之前BBV缓冲区充满度应小于BBS。一个视频序列的最后一帧不应是大图像。

在低延迟模式下，移出大图像时BBV缓冲区可能发生下溢。

D.3 缓冲区检测时间间隔

本章定义BBV缓冲区检测时间间隔。

D.3.1 非低延迟

当low_delay的值为‘0’时，BBV缓冲区检测时间间隔 $t(n+1)-t(n)$ 是帧率的倒数 T 的倍数。

如果第 n 帧图像是repeat_first_field的值为‘0’的B帧，检测时间间隔等于 T 。

如果第 n 帧图像是repeat_first_field的值为‘1’的B帧，并且top_field_first的值为‘0’，检测时间间隔等于 $2T$ 。

如果第 n 帧图像是repeat_first_field的值为‘1’的B帧，并且top_field_first的值为‘1’，检测时间间隔等于 $3T$ 。

如果第 n 帧图像是P帧或I帧，并且前一个P帧或I帧的repeat_first_field的值为‘0’，检测时间间隔等于 T 。

如果第 n 帧图像是P帧或I帧，并且前一个P帧或I帧的repeat_first_field的值为‘1’，top_field_first的值为‘0’，检测时间间隔等于 $2T$ 。

如果第 n 帧图像是P帧或I帧，并且前一个P帧或I帧的repeat_first_field的值为‘1’，top_field_first的值为‘1’，检测时间间隔等于 $3T$ 。

D.3.2 低延迟

当low_delay的值为‘1’时，BBV缓冲区检测时间间隔 $t(n+1)-t(n)$ 是帧率的倒数 T 的倍数。

如果第 n 帧图像是repeat_first_field的值为‘0’的P帧或I帧，检测时间间隔等于 T 。

如果第 n 帧图像是repeat_first_field的值为‘1’并且top_field_first等于‘0’的P帧或I帧，检测时间间隔等于 $2T$ 。

如果第 n 帧图像是repeat_first_field的值为‘1’并且top_field_first等于‘1’的P帧或I帧，检测时间间隔等于 $3T$ 。