(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 101902636 B
(45)授权公告日 2013.11.06

(21)申请号 201010110774.1
(22)申请日 2004.09.03
(30)优先权数据
60/501,081 2003.09.07 US
10/857,473 2004.05.27 US
10/933,958 2004.09.02 US
(62)分案原申请数据
200480025575.3 2004.09.03
(73)专利权人 微软公司
地址 美国华盛顿州
(72)发明人 T·W·赫尔科比
(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 顾嘉运
(51)Int.Cl.
H04N 7/26(2006.01)
H04N 7/36(2006.01)
H04N 7/46(2006.01)
H04N 7/50(2006.01)

(56)对比文件
CN 1226781 A,1999.08.25, 全文.
CN 1230855 A,1999.10.06, 全文.
US 5565922 A,1996.10.15, 全文.
Peter Borgwardt.Core Experiment on Macroblock Adaptive Frame/Field Encoding.《Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG》.2002, 全文.

审查员 荣芳

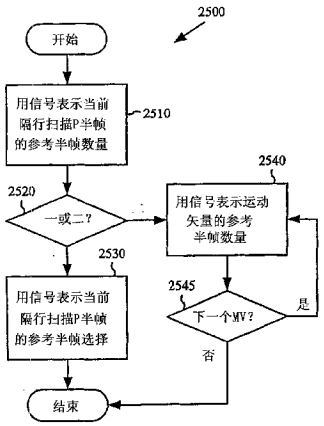
权利要求书3页 说明书67页 附图84页

(54)发明名称

隔行扫描视频的编码与解码

(57)摘要

本发明描述了编码和解码隔行扫描视频的各种技术和工具,包括(1)隔行扫描前向预测半帧的混合运动矢量预测,(2)使用运动矢量块模式,(3)在运动矢量预测值的主和非主极性之间选择,(4)参考半帧选择信息和差分运动矢量信息的联合编码和解码,(5)隔行扫描前向预测半帧的宏块的联合编码/解码,(6)使用可用于隔行扫描前向预测半帧的参考半帧数量的信号,以及(7)导出隔行扫描前向预测半帧的宏块的色度运动矢量。各种技术与工具可以组合或者独立地使用。



CN 101902636 B

1. 在视频解码器中的一种用于解码隔行扫描视频的方法，包括：

所述视频解码器处理指示隔行扫描前向预测半帧具有一个参考半帧还是两个可能的参考半帧用于运动补偿的第一半帧级信号；

如果所述第一半帧级信号指示隔行扫描前向预测半帧具有一个参考半帧，则所述视频解码器处理从两个可能的参考半帧中标识一个参考半帧的第二半帧级信号，所标识的参考半帧被用于所述隔行扫描前向预测半帧的所有运动补偿；以及

所述视频解码器对所述隔行扫描前向预测半帧执行所述运动补偿。

2. 如权利要求 1 所述的方法，其特征在于，所述第一半帧级信号是单个比特。

3. 如权利要求 1 所述的方法，其特征在于，所述第二半帧级信号是单个比特。

4. 如权利要求 1 所述的方法，其特征在于，还包括，如果所述第一半帧级信号指示所述隔行扫描前向预测半帧具有两个可能的参考半帧，则对于所述隔行扫描前向预测半帧的块和 / 或宏块的每一个运动矢量，所述视频解码器处理用于在所述两个可能的参考半帧之间进行选择的第三信号，所选择的参考半帧被用于运动补偿。

5. 如权利要求 4 所述的方法，其特征在于，所述第三信号在宏块级上。

6. 如权利要求 4 所述的方法，其特征在于，所述第三信号是在块级上。

7. 如权利要求 1 所述的方法，其特征在于，所述两个可能的参考半帧被约束为（1）时间上最近的先前隔行扫描帧内编码或前向预测半帧，以及（2）时间上第二最近的先前隔行扫描帧内编码或前向预测半帧。

8. 如权利要求 1 所述的方法，其特征在于，如果所述第一半帧级信号指示所述隔行扫描前向预测半帧具有两个可能的参考半帧，则所述视频解码器处理用于所述隔行扫描前向预测半帧的每运动矢量参考半帧选择的信号。

9. 在视频解码器中的一种用于解码隔行扫描视频的方法，包括：

所述视频解码器处理指示隔行扫描前向预测半帧具有一个参考半帧还是两个可能的参考半帧用于运动补偿的第一信号；

所述视频解码器确定所述第一信号指示隔行扫描前向预测半帧具有一个参考半帧还是两个可能的参考半帧，其中：

如果所述第一信号指示隔行扫描前向预测半帧具有一个参考半帧，处理用于所述隔行扫描前向预测半帧的参考半帧选择的信号；

如果所述第一信号指示隔行扫描前向预测半帧具有两个可能的参考半帧，处理用于所述隔行扫描前向预测半帧的每运动矢量的参考半帧选择的信号；

所述视频解码器对所述隔行扫描前向预测半帧执行运动补偿；以及

所述视频解码器更新后续运动补偿的参考半帧缓冲区，而不处理用于管理所述参考半帧缓冲区的附加信号。

10. 如权利要求 9 所述的方法，其特征在于，还包括，如果所述第一信号指示所述隔行扫描前向预测半帧具有一个参考半帧，则所述视频解码器处理从两个可能的参考半帧中标识一个参考半帧的第二信号。

11. 如权利要求 10 所述的方法，其特征在于，所述第一和第二信号各自是单个比特。

12. 如权利要求 10 所述的方法，其特征在于，所述第一和第二信号各自在所述隔行扫描前向预测半帧的图像级上。

13. 如权利要求 9 所述的方法，其特征在于，还包括，如果所述第一信号指示所述隔行扫描前向预测半帧具有两个可能的参考半帧，则对于所述隔行扫描前向预测半帧的块和 / 或宏块的每一个运动矢量，所述视频解码器处理用于在所述两个可能的参考半帧之间进行选择的第二信号。

14. 如权利要求 9 所述的方法，其特征在于，所述两个可能的参考半帧被约束为（1）在时间上最近的先前隔行扫描帧内编码或前向预测半帧，以及（2）在时间上第二最近的先前隔行扫描帧内编码或前向预测半帧。

15. 如权利要求 9 所述的方法，其特征在于，所述一个参考半帧被约束为（1）在时间上最近的先前隔行扫描帧内编码或前向预测半帧，或者为（2）在时间上第二最近的先前隔行扫描帧内编码或前向预测半帧。

16. 在解码器中的一种用于解码隔行扫描视频的方法，包括 ：

所述解码器处理第一信号，所述第一信号指示隔行扫描前向预测半帧具有一个参考半帧还是两个可能的参考半帧，其中（a）如果所述第一信号指示隔行扫描前向预测半帧具有一个参考半帧，处理用于所述隔行扫描前向预测半帧的参考半帧选择的信号，（b）如果所述第一信号指示隔行扫描前向预测半帧具有两个可能的参考半帧，处理用于所述隔行扫描前向预测半帧的每运动矢量的参考半帧选择的信号 ；

所述解码器处理第二信号，所述第二信号在所述第一信号指示隔行扫描前向预测半帧具有一个参考半帧时从两个可能的参考半帧中标识一个参考半帧 ；

所述解码器处理在所述第一信号指示隔行扫描前向预测半帧具有两个可能的参考半帧时用于多个运动矢量的每一个的第三信号，其中，所述第三信号的每一个用于在所述两个可能的参考半帧之间进行选择 ；以及

所述解码器对所述隔行扫描前向预测半帧执行运动补偿。

17. 如权利要求 16 所述的方法，其特征在于，所述第一信号是单个比特，并且所述第二信号是单个比特。

18. 如权利要求 16 所述的方法，其特征在于，所述第一信号和所述第二信号各自在所述隔行扫描前向预测半帧的图像级，并且其中，所述第三信号在宏块级上。

19. 如权利要求 16 所述的方法，其特征在于，所述两个可能的参考半帧被约束为（1）在时间上最近的先前隔行扫描帧内编码或前向预测半帧，以及（2）在时间上第二最近的先前隔行扫描帧内编码或前向预测半帧。

20. 如权利要求 16 所述的方法，其特征在于，进一步包括 ：所述解码器更新后续运动补偿的参考半帧缓冲区，而不处理用于管理参考半帧缓冲区的附加信号。

21. 在视频编码器中的一种用于编码隔行扫描视频的方法，包括 ：

所述视频编码器编码隔行扫描前向预测半帧，包括对所述隔行扫描前向预测半帧执行运动补偿 ；以及

所述视频编码器在比特流中输出编码的结果，包括 ：

用第一半帧级信号来指示隔行扫描前向预测半帧具有一个参考半帧还是两个可能的参考半帧用于运动补偿，其中如果所述第一半帧级信号指示隔行扫描前向预测半帧具有一个参考半帧，用第二半帧级信号表示所述隔行扫描前向预测半帧的参考半帧选择，如果所述第一半帧级信号指示隔行扫描前向预测半帧具有两个可能的参考半帧，用第三信号表示

所述隔行扫描前向预测半帧的每运动矢量的参考半帧选择，所述第二半帧级信号从两个可能的参考半帧中标识一个参考半帧，所标识的参考半帧被用于所述隔行扫描前向预测半帧的所有运动补偿。

22. 如权利要求 21 所述的方法，其特征在于，所述第三信号表示用于在所述两个可能的参考半帧之间的选择，所选择的参考半帧被用于运动补偿。

23. 如权利要求 21 所述的方法，其特征在于，所述第一半帧级信号是单个比特，并且所述第二半帧级信号是单个比特。

24. 如权利要求 21 所述的方法，其特征在于，所述两个可能的参考半帧被约束为（1）在时间上最近的先前隔行扫描帧内编码或前向预测半帧，以及（2）在时间上第二最近的先前隔行扫描帧内编码或前向预测半帧。

隔行扫描视频的编码与解码

[0001] 本申请是申请日为 2004.09.03，申请号为 200480025575.3（国际申请号为 PCT/US2004/029034），名为“隔行扫描视频的编码与解码”申请的分案申请。

[0002] 交叉引用

[0003] 本专利文档的部分公开内容包含受到版权保护的材料。版权所有者不妨碍任何人对出现在专利和商标局专利文件或记录中的本专利公开内容的复制，但无论如何保留所有版权。

技术领域

[0004] 描述了隔行扫描视频编码与解码的技术和工具。

背景技术

[0005] 数字视频消耗大量存储和传输能力。典型的原始数字视频序列每秒钟包括 15 或 30 帧。每帧可以包括成千上万个象素（也称为 pel），其中每个象素代表图像的一个微小元素。在原始格式中，计算机通常将一个象素表示为一组三个样本，总共 24 个比特。例如，一个象素可包括八位亮度样本（也称为 luma 样本，因为术语“luminance（亮度）”和“luma”在这里可交换使用），它定义了象素的灰度级分量，并包括两个八位色度样本（也称为 chroma 样本，因为术语“chrominance（色度）”和“chroma”在这里可交换使用），它们定义了象素的色彩分量。因而，典型的原始数字视频序列每秒钟的比特数，即比特率可以是每秒 5 百万个比特或更多。

[0006] 许多计算机和计算机网络缺乏处理原始数字视频的资源。因此，工程师使用压缩（也称为编码或译码）来降低数字视频的比特率。压缩通过将视频转换成较低比特率形式来降低存储和传输视频的成本。解压（也称为解码）从压缩格式重构原始视频的版本。“编解码器”是编码器 / 解码器系统。压缩可以是无损的，其中视频质量不受损害，但在比特率方面的降低受到视频数据的可变性的固有量（有时称为熵）限制。或者，压缩可以是有损的，其中视频质量受到损害，但在比特率方面可达到的降低是相当显著的。有损压缩经常结合无损压缩使用 - 有损压缩建立信息 的近似值，而应用无损压缩来表示该近似值。

[0007] 通常，视频压缩技术包括“图像内”压缩和“图像间”压缩，其中图像是例如逐行扫描的视频帧、隔行扫描视频帧（具有视频半帧的交替行）或隔行扫描视频半帧。对于逐行扫描帧，图像内压缩技术压缩独立的帧（一般称为 I 帧或关键帧），而图像间压缩技术参考前导和 / 或后续帧（一般称为参考或锚帧）或多个前导和 / 或后续帧（用于 B 帧）来压缩帧（一般称为预测帧，P 帧，或 B 帧）。

[0008] 图像间压缩技术经常使用运动估计和运动补偿。对于运动估计，例如，编码器将当前的预测帧划分成 8x8 或 16x16 个象素单元。对于当前帧的一个单元，找出参考帧中的相似单元用作预测值。运动矢量指示该预测值在参考帧中的位置。换言之，当前帧的一个单元的运动矢量指示在当前帧中该单元的空间位置与在参考帧中预测值的空间位置之间的位移。编码器计算当前单元与预测值之间的逐样本差，以确定残差（也称为误差信号）。如

果当前单元大小是 16x16，则将残差分成四个 8x8 块。对于每个 8x8 的残差，编码器应用一个可逆的频率变换运算，它产生一组频域（即频谱）系数。离散余弦变换 [“DCT”] 是一种类型的频率变换。对所得到的频谱系数块进行量化并且熵编码。如果将预测帧用作后续运动补偿的参考，则编码器重构预测帧。当重构残差时，编码器重构量化过的变换系数（例如，DCT 系数），并且执行反频率变换，如反 DCT[“IDCT”]。编码器执行运动补偿以计算预测值，并且将这些预测值与残差组合起来。在解码时，解码器一般对信息进行熵编码并且执行模拟运算来重构残差，执行运动补偿，并且将预测值与残差组合起来。

[0009] I. 在 Windows Media Video 版本 8 和 9 中的帧间压缩

[0010] 微软公司的 Windows Media Video(Windows 媒体视频）版本 8[“WMV8”] 包括视频编码器和视频解码器。WMV8 编码器使用帧内和帧间压缩，而 WMV8 解码器使用帧内和帧间解压。Windows Media Video 版本 9[“WMV9”] 对许多操作使用类似的体系结构。

[0011] WMV8 编码器中的帧间压缩使用基于块的经运动补偿的预测编码，之后是残留误差的变换编码。图 1 和 2 示出了用于 WMV8 编码器中的预测帧的基于块的帧间压缩。具体地，图 1 示出了用于预测帧 (110) 的运动估计，图 2 示出了用于预测帧的经运动补偿的块的预测残差的压缩。

[0012] 例如，在图 1 中，WMV8 编码器为预测帧 (110) 中的宏块 (115) 计算运动矢量。为计算运动矢量，编码器在参考帧 (130) 的搜索区域 (135) 中进行搜索。在 搜索区域 (135) 中，编码器将来自预测帧 (110) 的宏块 (115) 与各种候选宏块进行比较，以找出作为良好匹配的候选宏块。编码器输出指定匹配宏块的运动矢量（熵编码的）的信息。

[0013] 由于运动矢量值经常与空间围绕的运动矢量的值相关，因此用于发送运动矢量信息的数据的压缩可以通过从相邻宏块中确定或选择一个运动矢量预测值并且使用该运动矢量预测值预测当前宏块的运动矢量来完成。编码器可以编码运动矢量与运动矢量预测值之间的差分。例如，编码器计算运动矢量的水平分量与运动矢量预测值的水平分量之间的差，计算运动矢量的垂直分量与运动矢量预测值的垂直分量之间的差，以及编码这些差。

[0014] 在通过将差分加到预测值重构运动矢量之后，解码器使用运动矢量，利用来自参考帧 (130) 的信息来为宏块 (115) 计算预测宏块，参考帧 (130) 是在编码器和解码器处可用的先前重构的帧。预测很少是完美的，因此编码器通常对预测宏块和宏块 (115) 本身之间的像素差块（也称为误差或残差块）进行编码。

[0015] 图 2 示出了 WMV8 编码器中的误差块 (235) 的计算和编码的示例。误差块 (235) 是预测的块 (215) 和原始的当前块 (225) 之差。编码器向误差块 (235) 应用离散余弦变换 [“DCT”](240)，得到 8×8 的系数块 (245)。编码器然后量化 (250) 该 DCT 系数，得到 8×8 的经量化的 DCT 系数块 (255)。编码器将 8×8 的块 255 扫描 (260) 成一维数组 (265)，使得系数一般从最低频率到最高频率排序。编码器使用行程长度编码 (270) 的变更对扫描的系数进行熵编码。编码器从一个或多个“行程 / 等级 / 最后”表 (275) 中选择熵码，并输出该熵码。

[0016] 图 3 示出了用于帧间编码块的对应解码过程 (300) 的示例。在图 3 的概述中，解码器使用可变长度解码 (310)，利用一个或多个“行程 / 级别 / 最后”表 315 和行程长度解码 (320)，对表示预测残差的熵编码的信息进行解码 (310、320)。解码器将储存熵编码的信息的一维数组 325 反扫描 (330) 成二维块 (335)。解码器对该数据进行反量化和离散反余

弦变换（共同在 340 处），得到重构的误差块 (345)。在独立的运动补偿路径中，解码器对从参考帧的偏移使用运动矢量信息 (355) 计算预测块 (365)。解码器将预测块 (365) 与重构的误差块 (345) 组合 (370)，以形成重构块 (375)。

[0017] II. 隔行扫描视频和逐行扫描视频

[0018] 视频帧包含视频信号的空间信息的行。对于逐行扫描视频，这些行包含从一 个时刻开始继续以光栅扫描的方式通过连续的行直到帧底部的样值。逐行扫描的 I 帧是帧内编码的逐行扫描视频帧。逐行扫描的 P 帧是使用前向预测编码的逐行扫描视频帧，而逐行扫描的 B 帧是使用双向预测编码的逐行扫描视频帧。

[0019] 隔行扫描视频的主要方面是整个视频帧的光栅扫描是通过两遍完成的，其中每遍扫描交替的行。例如，第一扫描是由帧的偶数行构成，而第二扫描是由扫描的奇数行构成。这导致每个帧包含两个半帧，它们表示两个不同的时间点。图 4 示出包括上半帧 (410) 和下半帧 (420) 的隔行扫描的视频帧 (400)。在帧 (400) 中，偶数号的行（上半帧）是从一个时刻（例如，时刻 t）开始扫描的，而奇数号的行（下半帧）是从一个不同（通常稍晚）的时刻（例如，时刻 t+1）开始扫描的。当两个半帧在不同的时刻开始扫描时，这一时序可创建隔行扫描视频帧的存在运动的区域中看似锯齿状的特征。为此，可依照半帧结构对隔行扫描视频帧重新排列，使得奇数行被组合在一起成为一个半帧，而偶数行被组合在一起成为另一半帧。这一排列被称为半帧编码，它在高运动图像中对于降低这一锯齿状边缘的人为因素是非常有用的。另一方面，在静止区域中，隔行扫描视频帧中的图像细节可被更有效地保存，而无需这样的重新排列。因此，通常在静止或低运动的隔行扫描视频帧中使用帧编码，在这样的视频帧中，保存了原始的交错半帧行排列。

[0020] 典型的逐行扫描视频帧由具有非交错行的内容的一帧构成。与隔行扫描视频相反，逐行扫描视频不将视频帧划分成独立的半帧，且从单个时刻开始，从左到右、从上到下扫描整个帧。

[0021] III. 在 WMV 编码器和解码器中先前的编码与解码

[0022] 以可执行形式发布的用于 WMV 编码器和解码器的早先的软件已经使用了逐行扫描和隔行扫描 P 帧的编码与解码。尽管编码器和解码器对于许多不同的编码 / 解码情形和内容类型是有效的，但是在若干地方还有改进空间。

[0023] A. 运动补偿的参考图像

[0024] 编码器和解码器使用运动补偿用于逐行扫描和隔行扫描前向预测帧。对于逐行扫描 P 帧，运动补偿是相对于单个参考帧的，后者是先前重构的 I 帧或 P 帧，当前 P 帧紧跟在该帧之后。由于当前 P 帧的参考帧是已知的，并且只有一个参考帧是可能的，因此不需要用于在多个参考帧之间选择的信息。

[0025] 隔行扫描 P 帧的宏块可以是半帧编码的，或者是帧编码的。在半帧编码的宏 块中，最多两个运动矢量与该宏块相关联，一个用于上半帧而一个用于下半帧。在帧编码的宏块中，最多一个运动矢量与该宏块相关联。对于在隔行扫描 P 帧中的帧编码宏块，运动补偿是相对于单个参考帧的，后者是先前重构的 I 帧或 P 帧，当前 P 帧紧跟在该帧之后。对于在隔行扫描 P 帧中的半帧编码宏块，运动补偿仍是相对于单个参考帧的，但对于半帧编码宏块的上半帧的运动矢量只考虑参考帧的上半帧行，并且对于半帧编码宏块的下半帧的运动矢量只考虑参考帧的下半帧行。再一次，由于参考帧是已知的并且只有一个参考帧是可能

的，因此不需要用于在多个参考帧之间选择的信息。

[0026] 在某些编码 / 解码情形（例如，带有许多运动的高比特率隔行扫描视频）中，限制对前向预测的运动补偿相对于单个参考可能会损害整个压缩效率。

[0027] B. 用信号表示宏块信息

[0028] 编码器和解码器使用用于逐行扫描或隔行扫描 P 帧的宏块信息的信号表示。

[0029] 1. 用信号表示用于逐行扫描 P 帧的宏块信息

[0030] 逐行扫描 P 帧可以是 1MV 或混合 MV 帧。1MVP 逐行扫描帧包括 1MV 宏块。1MV 宏块具有一个运动矢量，它指示在该宏块中所有六个块的预测块位移。混合 MV 逐行扫描 P 帧包括 1MV 和 / 或 4MV 宏块。4MV 宏块具有从 0 到 4 个运动矢量，其中每个运动矢量用于该宏块的最多四个亮度块之一。在逐行扫描 P 帧中的宏块可以是三种可能类型之一 ：1MV，4MV 和跳过。另外，1MV 和 4MV 宏块可以是帧内编码的。宏块类型是由图像与宏块层元素的组合来指示的。

[0031] 因而，1MV 宏块可以在 1MV 和混合 MV 逐行扫描帧中出现。单个运动矢量数据 MVDATA 元素与 1MV 宏块中的所有块相关联。MVDATA 用信号表示这些块被编码为帧内编码类型还是被编码为帧间编码类型。如果它们被编码为帧间编码，则 MVDATA 还指示运动矢量差分。

[0032] 如果逐行扫描 P 帧是 1MV，则在该帧中的所有运动矢量是 1MV 宏块，因此没有必要单独地用信号表示宏块类型。如果逐行扫描 P 帧是混合 MV，则在该帧中的宏块可以是 1MV 或 4MV。在这种情况下，通过比特流中图像层处的位平面，为帧中的每个宏块用信号表示宏块类型 (1MV 或 4MV)。经解码的位平面将宏块的 1MV/4MV 状态以上左至下右的光栅扫描顺序表示为 1 比特值的平面。值 0 指示相应的宏块是以 1MV 模式编码的。值 1 指示相应的宏块是以 4MV 模式编码的。在一个编码模式中，在比特流的宏块层处的每个宏块用信号表示 1MV/4MV 状态信息 （而非表示为逐行扫描 P 帧的平面 ）。

[0033] 4MV 宏块出现在混合 MV 逐行扫描帧中。在 4MV 宏块内的各个块可以被编码为帧内编码块。对于 4MV 宏块的四个亮度块中的每一个，帧内编码 / 帧间编码状态是由与该块相关联的块运动矢量数据 BLKMVDATA 元素用信号表示的。对于 4MV 宏块，已编码块模式 CBPCY 元素指示哪些块在比特流中存在 BLKMVDATA 元素。色度块的帧间编码 / 帧内编码状态是从亮度帧间编码 / 帧内编码状态导出的。如果两个或多个亮度块被编码为帧内编码，则色度块也被编码为帧内编码。

[0034] 另外，帧中每个宏块的跳过 / 非跳过状态也是由逐行扫描帧的位平面用信号表示的。跳过的宏块仍可具有混合运动矢量预测的相关信息。

[0035] CBCPY 是可变长度代码 [“VLC”]，它被解码为一个 6 比特字段。CBPCY 出现在 1MV 和 4MV 宏块的比特流中的不同位置，并且对于 1MV 和 4MV 宏块具有不同的语义。

[0036] 如果 ：(1)MVDATA 指示该宏块是帧间编码的，以及 (2)MVDATA 指示 1MV 宏块的至少一个块包含系数信息 （由从 MVDATA 解码的“最后一个”值指示的 ），则 CBPCY 存在于 1MV 宏块层中。如果 CBPCY 存在，则它被解码为一个 6 比特字段，它指示相应的六个块中的哪一些包含至少一个非零的系数。

[0037] CBCPY 始终在 4MV 宏块层中存在。亮度块的 CBPCY 比特位置 （位 0-3) 与色度块的比特位置 （位 4 与 5) 相比，有轻微不同的意义。对于亮度块的一个比特位置，0 指示相应块

不包含运动矢量信息或任何非零系数。对于这样一个块，BLKMVDATA 不存在，预测运动矢量用作运动矢量，并且没有残差数据。如果运动矢量预测值指示使用混合运动矢量预测，则存在一个单一比特，它指示要使用的候选运动矢量预测值。在亮度块的一个比特位置中的 1 指示该块存在 BLKMVDATA。BLKMVDATA 指示该块是帧间编码还是帧内编码，并且如果是帧间编码，则指示运动矢量差分。BLKMVDATA 还指示是否存在该块的系数数据（用从 BLKMVDATA 解码的“最后一个”值）。对于色度块的一个比特位置，0 或 1 指示相应块是否包含非零系数信息。

[0038] 编码器和解码器使用 VLC 表的代码表，分别用于 MVDATA、BLKMVDATA 和 CBCPY。

[0039] 2. 用信号表示用于隔行扫描 P 帧的宏块信息

[0040] 隔行扫描 P 帧可具有帧编码与半帧编码宏块的混合。在半帧编码宏块中，最 多两个运动矢量与该宏块相关联。在帧编码宏块中，最多一个运动矢量与该宏块相关联。如果序列层元素 INTERLACE 是 1，则图像层元素 INTRLCF 存在于比特流中。INTRLCF 是 1 比特的元素，它指示用于在该帧中编码宏块的模式。如果 INTRLCF ＝ 0，则在帧中的所有宏块以帧模式编码。如果 INTRLCF ＝ 1，则宏块可以用半帧或帧模式编码，并且在图像层中存在的位平面 INTRLCMB 指示隔行扫描 P 帧中每个宏块的半帧 / 帧编码状态。

[0041] 在隔行扫描 P 帧中的宏块可以是三种可能类型之一 :帧编码，半帧编码和跳过。宏块类型是由图像与宏块层元素的组合指示的。

[0042] 单个 MVDATA 与帧编码宏块中的所有块相关联。MVDATA 用信号表示将块编码为帧内编码还是帧间编码类型。如果它们被编码为帧间编码，则 MVDATA 还指示运动矢量差分。

[0043] 在半帧编码宏块中，上半帧运动矢量数据 TOPMVDATA 元素与上半帧块相关联，而下半帧运动矢量数据 BOTMVDATA 元素与下半帧块相关联。在每个半帧的第一块处用信号表示这些元素。更明确地说，TOPMVDATA 连同左上半帧块一起用信号表示，而 BOTMVDATA 连同左下半帧块一起用信号表示。TOPMVDATA 指示上半帧块是帧内编码还是帧间编码。如果它们是帧间编码，则 TOPMVDATA 还指示上半帧块的运动矢量差分。同样，BOTMVDATA 用信号表示下半帧块的帧间编码 / 帧内编码状态，以及可能的下半帧块的运动矢量差分信息。CBPCY 指示哪些半帧具有在比特流中存在的运动矢量数据元素。

[0044] 跳过的宏块是由图像层中的 SKIPMB 位平面用信号表示的。CBPCY 和运动矢量数据元素用于指定块是否具有 AC 系数。如果从 MVDATA 解码的“最后一个”值指示在运动矢量之后有数据要解码，则对隔行扫描帧的帧编码宏块存在 CBPCY。如果 CBPCY 存在，则它被解码为一个 6 比特字段，一个比特用于四个 Y 块中的每一个，一个比特用于两个 U 块（上半帧和下半帧），以及一个比特用于两个 V 块（上半帧和下半帧）。

[0045] 对半帧编码宏块始终存在 CBPCY。CBPCY 和两个半帧运动矢量数据元素用于确定宏块的块中存在 AC 系数。CBPCY 的意义与帧编码宏块的比特 1、3、4 和 5 相同。即，它们指示 AC 系数在右上半帧 Y 块、右下半帧 Y 块、上 / 下 U 块和上 / 下 V 块中的存在和不存在。对于比特位置 0 和 2，意义轻微不同。在比特位置 0 中的 0 指示 TOPMVDATA 不存在，并且运动矢量预测值用作上半帧块的运动矢量。它还指示左上半帧块不包含任何非零系数。在比特位置 0 中的 1 指示 TOPMVDATA 存在。TOPMVDATA 指示上半帧块是帧间编码还是帧内编码，并且如果它们是帧间编码，还指示运动矢量差分。如果从 TOPMVDATA 解码的“最后一个”值解码为 1，则对左上半帧块不存在 AC 系数，否则，不存在左上半帧块的非零 AC 系数。同样，

上述规则应用于 BOTMVDATA 的比特位置 2 和左下半帧块。

[0046] 编码器和解码器使用 VLC 表的代码表选择，分别用于 MVDATA、TOPMVDATA、BOTMVDATA 和 CBPCY。

[0047] 3. 与早先的宏块信息信号表示有关的问题

[0048] 总之，逐行扫描帧和隔行扫描帧的宏块的各种信息是用在帧和宏块层处的独立代码（或者代码的组合）来用信号表示的。该独立地用信号表示的信息包括运动矢量的数量、宏块帧内编码 / 帧间编码状态、CBPCY 存在或不存在（例如，通过用于 1MV 和帧编码宏块的“最后一个”值）以及运动矢量数据存在或不存在（例如，通过用于 4MV 和半帧编码宏块的 CBPCY）。尽管该信号表示在许多情况下提供良好的总体性能，但是它没有充分地利用各种普通情况下不同的用信号表示的信息之间的统计相关性。而且，它不允许和不处理各种有用的配置，诸如用于 4MV 宏块的 CBPCY 的存在 / 不存在，或者用于 1MV 宏块的运动矢量数据的存在 / 不存在。

[0049] 而且，就用信号表示运动矢量数据的存在 / 不存在（例如，通过用于 4MV 和半帧编码宏块的 CBPCY）而言，它要求 CBPCY 元素的常规角色的混乱的重新定义。这进而要求通过不同的元素（例如 BLKMVDATA、TOPMVDATA、BOTMVDATA）（它们按照惯例不用于该目的）来用信号表示常规的 CBCPY 信息。并且，信号表示不允许和不处理各种有用的配置，诸如当运动矢量数据不存在时系数信息的存在。

[0050] C. 运动矢量预测

[0051] 对于在隔行扫描或逐行扫描 P 帧中的宏块（或者块，或者宏块的半帧等）的运动矢量，编码器通过基于相邻的运动矢量计算运动矢量预测值、计算运动矢量与运动矢量预测值之间的差分和编码该差分来编码运动矢量。解码器通过计算运动矢量预测值（再次基于相邻的运动矢量）、解码运动矢量差分和将运动矢量差分加到运动矢量预测值来重构运动矢量。

[0052] 图 5A 和 5B 示出了对 1MV 逐行扫描 P 帧中的 1MV 宏块的候选运动矢量预测值考虑的宏块的位置。候选预测值取自左侧、顶部和右上角的宏块，除了在宏块是 行中的最后一个宏块的情况之外。在这一情况下，预测值 B 取自左上角的宏块，而非右上角。对于其中该帧是一个宏块宽的特殊情况，预测值总是为预测值 A（顶部预测值）。当由于宏块在顶行中而使预测值 A 位于带外时，预测值为预测值 C。各种其它规则解决诸如帧内编码预测值等其它特殊情况。

[0053] 图 6A-10 示出了为混合 MV 逐行扫描 P 帧中的 1MV 或 4MV 宏块的运动矢量的多达三个候选运动矢量考虑的块或宏块的位置。在附图中，较大的方块是宏块边界，而较小的方块是块边界。对于其中帧是一个宏块宽的特殊情况，预测值总是预测值 A（顶部预测值）。各种其它规则解决诸如对顶行的 4MV 宏块的顶行块、顶行的 1MV 宏块以及帧内编码预测值等其它特殊情况。

[0054] 具体地，图 6A 和 6B 示出了为混合 MV 逐行扫描 P 帧中的 1MV 当前宏块的候选运动矢量预测值考虑的块位置。相邻宏块可以是 1MV 或 4MV 宏块。图 6A 和 6B 示出了假定邻块是 4MV 时候选运动矢量的位置（即，预测值 A 是对于在当前宏块上方的宏块中的块 2 的运动矢量，而预测值 C 是直接在当前宏块左边的宏块中的块 1 的运动矢量）。如果邻块中的任一块是 1MV 宏块，则图 5A 和 5B 所示的运动矢量预测值被带到整个宏块的运动矢量预测值。

如图 6B 所示，如果宏块是行中的最后一个宏块，则预测值 B 来自左上角的宏块的块 3，而非其它情况下的右上角宏块中的块 2。

[0055] 图 7A-10 示出了为混合 MV 逐行扫描 P 帧的 4MV 宏块中的 4 个亮度块的每一个的候选运动矢量预测值考虑的块位置。图 7A 和 7B 示出了为位置 0 处的块的候选运动矢量预测值考虑的块位置；图 8A 和 8B 示出了为位置 1 处的块的候选运动矢量预测值考虑的块位置的图示；图 9 示出了为位置 2 处的块的候选运动矢量预测值考虑的块位置；图 10 示出了为位置 3 处的块的候选运动矢量预测值考虑的块位置。再一次，如果邻块是 1MV 宏块，则该宏块的运动矢量预测值用于该宏块的各块。

[0056] 对于其中宏块是行中的第一个宏块的情况，块 0 的预测值 B 与该行中其余宏块的块 0 不同地处理（见图 7A 和 7B）。在这一情况下，预测值 B 取自直接在当前宏块上方的宏块中的块 3，而非如其它情况下取自在当前宏块左上方的宏块中的块 3。类似地，对于其中宏块是行中的最后一个宏块的情况，块 1 的预测值 B 加以不同的处理（图 8A 和 8B）。在这一情况下，预测值取自直接在当前宏块上方的宏块中的块 2，而非如其它情况下取自当前宏块右上方的宏块中的块 2。一般而言，如果宏块是在第一个宏块列中，块 0 和 2 的预测值 C 被设为等于 0。

[0057] 如果逐行扫描帧的宏块被编码为跳过的，则它的运动矢量预测值用作该宏块的运动矢量（或者它的块的预测值用于这些块，等等）。仍存在单个比特，以指示在混合运动矢量预测中要使用哪个预测值。

[0058] 图 11 和 12A-B 示出了隔行扫描 P 帧中，分别用于帧编码的宏块和半帧编码的宏块的运动矢量预测的候选预测值的示例。图 11 示出了在隔行扫描的 P 帧中用于内部位置的当前帧编码的宏块（不是宏块行中的第一个或最后一个宏块，不在顶行中）的候选预测值 A、B 和 C。预测值可以从除标记为 A、B 和 C 之外的不同候选方向上获得（例如，在诸如当前宏块是行中的第一个宏块或最后一个宏块，或在顶行中的特殊情况下，由于某些预测值对于这些情况是不可用的）。对于当前帧编码的宏块，候选预测值是根据相邻的宏块是半帧编码还是帧编码的来不同地计算的。对于相邻的帧编码的宏块，仅仅取其运动矢量作为候选预测值。对于相邻的半帧编码的宏块，通过对上半帧和下半帧运动矢量求平均值来确定候选运动矢量。

[0059] 图 12A-B 示出了用于半帧中的内部位置的半帧编码的宏块中的当前半帧的候选预测值 A、B 和 C。在图 12A 中，当前半帧是下半帧，且相邻宏块中的下半帧运动矢量用作候选预测值。在图 12B 中，当前半帧是上半帧，且相邻宏块中的上半帧运动矢量用作候选预测值。对于当前半帧编码的宏块中的每一半帧，每一半帧的运动矢量候选预测值的数目最多是 3，其中每一候选预测值来自与当前半帧相同的半帧类型（例如，上半帧或下半帧）。如果相邻宏块是帧编码的，则其运动矢量用作上半帧预测值和下半帧预测值。再一次，当当前宏块是行中的第一个宏块或最后一个宏块，或在顶行中时，由于某些预测值对于这些情况是不可用的，因此应用各种特殊情况（未示出）。如果帧是一个宏块宽，则运动矢量预测值是预测值 A。如果相邻宏块是帧内编码的，则它的运动矢量预测值是 0。

[0060] 图 13A 和 13B 示出在给出一组预测值 A、B 和 C 时计算运动矢量预测值的伪代码。要从一组候选预测值中选择一个预测值，编码器和解码器使用一选择算法，诸如图 13C 中所示的求三个数的中值 (median-of-three) 算法。

[0061] D. 逐行扫描 P- 帧的混合运动矢量预测

[0062] 允许混合运动矢量预测用于逐行扫描 P 帧的运动矢量。对于宏块或块的运动矢量，无论逐行扫描 P 帧是 1MV 还是混合 MV，相对于 A 和 C 预测值测试在前面章节中计算的运动矢量预测值，以确定预测值选择是否显式地在比特流中编码。如果是，则解码一个比特，它指示使用预测值 A 还是预测值 C 作为运动矢量的运动 矢量预测值（而非使用在上面章节 C 中计算的运动矢量预测值）。混合运动矢量预测不在隔行扫描 P 帧或者隔行扫描视频的任何表示的运动矢量预测中使用。

[0063] 图 14A 与 14B 中的伪代码示出逐行扫描 P 帧的运动矢量的混合运动矢量预测。在该伪代码中，变量 predictor_pre_x 和 predictor_pre_y 分别是水平和垂直运动矢量预测值，如在前面章节中计算的。变量 predictor_post_x 和 predictor_post_y 分别是在检查混合运动矢量预测之后的水平和垂直运动矢量预测值。

[0064] E. 解码运动矢量差分

[0065] 对于逐行扫描 P 帧的宏块或块，MVDATA 或 BLKMVDATA 元素用信号表示运动矢量差分信息。1MV 宏块具有单个 MVDATA。4MV 宏块具有零与四个之间的 BLKMVDATA 元素（其存在由 CBPCY 指示）。

[0066] MVDATA 或 BLKMVDATA 对以下三项联合编码：(1) 水平运动矢量差分分量；(2) 垂直运动矢量差分分量；以及 (3) 二进制的“最后”标志，它一般指示变换系数是否存在。将宏块（或块，对于 4MV）是帧内编码还是帧间编码用信号表示为运动矢量差分可能性之一。在图 15A 与 15B 中的伪代码示出如何为 MVDATA 或 BLKMVDATA 解码运动矢量差分信息、帧间编码 / 帧内编码类型和最后标志信息。在该伪代码中，变量 last_flag 是一个二进制标志，其用途在关于用信号表示宏块信息的章节中描述。变量 infra_flag 是一个二进制标志，它指示块或宏块是否为帧内编码的。变量 dmv_x 和 dmv_y 分别是差分水平与垂直运动矢量分量。变量 k_x 和 k_y 对于扩展范围运动矢量是定长的，它们的值如图 15C 的表格所示地变化。变量 halfpel_flag 是一个二进制值，它指示用于运动矢量的是二分之一象素还是四分之一象素，并且其值是基于图像层句法元素设置的。最后，表 size_tale 和 offset_table 是如下定义的数组：

[0067] size_table[6] = {0,2,3,4,5,8}，以及

[0068] offset_table[6] = {0,1,3,7,15,31}.

[0069] 对于隔行扫描 P 帧的帧编码或半帧编码宏块，以相同的方法解码 MVDATA、TOPMVDATA 和 BLTMVDATA。

[0070] F. 重构和导出运动矢量

[0071] 亮度运动矢量是从经编码的运动矢量差分信息和运动矢量预测值重构的，而色度运动矢量是从重构的亮度运动矢量中导出的。

[0072] 对于逐行扫描 P 帧的 1MV 和 4MV 宏块，亮度运动矢量是通过将差分加到运动矢量预测值来重构的，如下：

[0073] mv_x = (dmv_x+predictor_x)smod range_x,

[0074] mv_y = (dmv_y+predictor_y)smod range_y,

[0075] 其中 smod 是如下定义的有符号模运算：

[0076] A smod b = ((A+b)% 2b)-b,

[0077] 它保证重构的矢量是有效的。

[0078] 在 1MV 宏块中，对于构成宏块的亮度分量的四个块，有单个运动矢量。如果宏块是帧内编码的，则没有运动矢量与该宏块相关联。如果宏块是跳过的，则 dmv_x = 0 且 dmv_y = 0，因此 mv_x = predictor_x 且 mv_y = predictor_y。

[0079] 在 4MV 宏块中的每个帧间编码亮度块具有它自己的运动矢量。因此，在一个 4MV 宏块中，将存在 0 到 4 个亮度运动矢量。如果 4MV 宏块是跳过的或者如果 4MV 宏块的 CBPCY 指示块是非编码的，则 4MV 宏块中非编码的块可以出现。如果块不是编码的，则 dmv_x = 0 且 dmv_y = 0，因此 mv_x = predictor_x 且 mv_y = predictor_y。

[0080] 对于逐行扫描帧，色度运动矢量是从亮度运动矢量中导出的。而且，对于 4MV 宏块，基于亮度块的状态作出将色度块编码为帧间编码还是帧内编码的决定。色度矢量是在两个步骤中重构的。

[0081] 在第一个步骤中，名义上的色度运动矢量是通过适当地组合与比例缩放亮度运动矢量来获得的。比例缩放是以这样一种方法执行的，即二分之一象素偏移比四分之一偏移更优选。图 16A 示出伪代码，用于在从 1MV 宏块的亮度运动矢量中导出色度运动矢量时进行比例缩放。图 16B 示出伪代码，用于在导出 4MV 宏块的色度运动矢量时组合至多四个亮度运动矢量和比例缩放。图 13C 示出 median3() 函数的伪代码，以及图 16C 示出用于 median4() 函数的伪代码。

[0082] 在第二个步骤中，使用序列级的 1 比特元素来确定是否需要色度运动矢量的进一步舍入。如果需要，则将四分之一象素偏移处的色度运动矢量舍入到最近的整象素位置。

[0083] 对于隔行扫描帧的帧编码和半帧编码宏块，如对逐行扫描 P 帧所做的一样来重构亮度运动矢量。在帧编码宏块中，对于构成宏块的亮度分量的四个块，有单个运动矢量。如果宏块是帧内编码的，则没有运动矢量与该宏块相关联。如果宏块是跳过的，则 dmv_x = 0 且 dmv_y = 0，因此 mv_x = predictor_x 且 mv_y = predictor_y。 在帧编码宏块中，每个半帧可具有它自己的运动矢量。因此，在一个帧编码宏块中将有 0 到 2 个亮度运动矢量。如果半帧编码宏块是跳过的或者如果半帧编码宏块的 CBPCY 指示半帧是非编码的，则半帧编码宏块中的非编码半帧可以出现。如果半帧是非编码的，则 dmv_x = 0 且 dmv_y = 0，因此 mv_x = predictor_x 且 mv_y = predictor_y。

[0084] 对于隔行扫描 P 帧，色度运动矢量是从亮度运动矢量导出的。对于帧编码宏块，有一个对应于单个亮度运动矢量的色度运动矢量。对于半帧编码宏块，有两个色度运动矢量。一个是上半帧的，而一个是下半帧的，分别对应于上半帧与下半帧亮度运动矢量。导出色度运动矢量的规则对于半帧编码和帧编码宏块是相同的。它们取决于亮度运动矢量，而不是宏块的类型。图 17 示出伪代码，用于从隔行扫描 P 帧的帧编码或半帧编码宏块的亮度运动矢量中导出色度运动矢量。基本上，色度运动矢量的 x 分量被放大 4 倍，而色度运动矢量的 y 分量保持不变（由于 4:1:1 宏块色度二次采样）。色度运动矢量的放大的 x 分量还舍入到相邻的四分之一象素位置。如果 cmv_x 或 cmv_y 在边界外，则将它拉回到有效范围。

[0085] G. 强度补偿

[0086] 对于逐行扫描 P 帧，图像层包含句法元素，它控制该帧的运动补偿模式和强度补偿。如果用信号表示强度补偿，则在图像层中跟随着 LUMSCALE 和 LUMSHIFT 元素。LUMSCALE 和 LUMSHIFT 是 6 比特值，它指定在强度补偿过程中使用的参数。

[0087] 当强度补偿用于逐行扫描 P 帧时，在参考帧中的象素在它们用于 P 帧的运动补偿预测之前被重新映射。图 18 中的伪代码示出 LUMSCALE 和 LUMSHIFT 元素用于构造查找表，该查找表用于重新映射参考帧象素。使用 LUTY[] 表重新映射参考帧的 Y 分量，并且使用 LUTUV[] 表重新映射 U 和 V 分量，如下：

[0088] $\bar{p}_Y = LUTY[p_Y]$，且

[0089] $\bar{p}_{UV} = LUTUV[p_{UV}]$

[0090] 其中 pY 是参考帧中的原始亮度象素值，$\bar{p}_Y$是参考帧中重新映射的亮度象素值，PUV 是参考帧中原始的 U 或 V 象素值，而 $\bar{p}_{UV}$是参考帧中重新映射的 U 或 V 象素值。

[0091] 对于隔行扫描 P 帧，一个 1 比特图像层 INTCOMP 值用信号表示是否对帧使用强度补偿。如果使用强度补偿，则在图像层中跟随着 LUMSCALE 和 LUMSHIFT 元素，其中 LUMSCALE 和 LUMSHIFT 是 6 比特值，它们指定在强度补偿过程中为整个隔行扫描 P 帧使用的参数。强度补偿本身与用于逐行扫描 P 帧的相同。

[0092] VI. 视频压缩与解压的标准

[0093] 除早先的 WMV 编码器和解码器以外，若干国际标准与视频压缩和解压有关。这些标准包括运动图像专家组 [“MPEG”]1，2 和 4 标准以及来自国际电信联盟 [“ITU”] 的 H.261、H.262(MPEG2 的另一个名字)、H.263 和 H.264 标准。遵循这些标准之一的编码器和解码器一般使用运动估计和补偿来减少图像之间的时间冗余。

[0094] A. 运动补偿的参考图像

[0095] 对于若干标准，前向预测帧的运动补偿是相对于单个参考帧的，后者是先前重构的 I 或 P 帧，当前前向预测帧紧跟该帧之后。由于当前前向预测帧的参考帧是已知的，并且只有一个参考帧是可能的，因此不需要用于在多个参考帧之间选择的信息。例如，见 H.261 和 MPEG 1 标准。在某些编码 / 解码情形 (例如，带有许多运动的高比特率隔行扫描视频) 中，将对前向预测的运动补偿限制为相对于单个参考会损害总体压缩效率。

[0096] H.262 标准允许将隔行扫描视频帧编码为单个帧或两个半帧，其中帧编码或半帧编码可以在逐帧的基础上自适应地选择。对于当前半帧的基于半帧的预测，运动补偿使用先前重构的上半帧或下半帧。[H.262 标准，章节 7.6.1 和 7.6.2.1。]H.262 标准描述在两个参考半帧之间选择以用于对当前半帧的运动矢量的运动补偿。[H.262 标准，章节 6.2.5.2，6.3.17.2 和 7.6.4。] 对于 16x16 宏块 (或者宏块的上半部分 16x8，或者宏块的下半部分 16x8) 的给定运动矢量，用信号表示单个比特来指示要将该运动矢量应用于上参考半帧还是应用于下参考半帧。[同前] 对于其它细节，见 H.262 标准。

[0097] 尽管这样的参考半帧选择在某些情况下提供某种灵活性和预测改进，但它有若干与比特率有关的缺点。运动矢量的参考半帧选择信号可以消耗许多比特。例如，对于具有 810 个宏块的单个 720x288 半帧，每个宏块具有 0、1 或 2 个运动矢量，运动矢量的参考半帧选择比特消耗多达 1620 个比特。没有作出通过预测将为相应的运动矢量选择哪些参考半帧来减少参考半帧选择信息的比特率的尝试。参考半帧选择信息的信号表示在纯编码效率方面是没有效率的。而且，对于某些情况，尽管 编码了信息，但参考半帧选择信息可消耗如此多的比特，超过了在运动补偿中具有多个可用参考而得到的预测改进的好处。没有给出选项来禁止参考半帧选择以处理这类情况。

[0098] H.262 标准还描述双主 (dual-prime) 预测，这是一种预测模式，其中对两个基于前向半帧的预测求平均值，用于一个隔行扫描 P 图像中的 16x16 块。[H.262 标准，章节 7.6.3.6。]

[0099] MPEG-4 标准允许隔行扫描视频帧的宏块是帧编码或半帧编码的。[MPEG-4 标准，章节 6.1.3.8。] 对于半帧编码的宏块的上半帧或下半帧的基于半帧的预测，运动补偿使用先前重构的上半帧或下半帧。[MPEG-4 标准，章节 6.3.7.3 和 7.6.2。]MPEG-4 标准描述在两个参考半帧之间的选择以用于运动补偿。[MPEG-4 标准，章节 6.3.7.3。] 对于宏块的上半帧行或下半帧行的给定运动矢量，用信号表示单个比特以指示要将该运动矢量应用于上参考半帧还是应用于下参考半帧。[同前] 对于其它细节，见 MPEG-4 标准。这样的参考半帧选择信息的信号表示具有与上面对于 H.262 描述的相似的问题。

[0100] H.263 标准描述逐行扫描帧的运动补偿，包括可任选的参考图像选择模式。[H.263 标准，章节 3.4.12，附录 N。] 通常，时间上最近的先前的定位图像用于运动补偿。然而，当使用参考图像选择模式时，允许来自除最近参考图像以外的图像的时间预测。[同前] 通过允许编码器优化它对信道条件的视频编码 (例如，由于在帧间编码中参考所需要的信息损耗而停止错误传播)，这可以改进通过易于出错的信道的实时视频通信的性能。[同前] 当使用时，对于一个图像内给定的一组块或片，一个 10 比特值指示用于该组块或片的预测的参考。[同前] 在 H.263 中描述的参考图像选择机制用于逐行扫描视频并且适合于处理在易于错误的信道中的错误传播问题，本质上没有提高压缩效率。

[0101] 在 H.264 标准的草案 JVT-D157 中，块的运动补偿预测的帧间预测过程可以包括从许多存储的、先前解码的图像中选择参考图像。[JVT-D157，章节 0.4.3。] 在图像级上，一个或多个参数指定用于解码图像的参考图像数量。[JVT-D157，章节 7.3.2.2 和 7.4.2.2。] 在片级，可用参考图像的数量可改变，并且可接收附加的参数以重新排序和管理哪些参考图像在列表中。[JVT-D157，章节 7.3.3 和 7.4.3。] 对于一个给定的运动矢量 (用于宏块或者子宏块部分)，参考索引 (当存在时) 指示要用于预测的参考图像。[JVT-D157，章节 7.3.5.1 和 7.4.5.1。] 参考索引指示列表中的第一、第二、第三帧或半帧等。[同前] 如果在列表中只有一个活动的参考图像，则参考索引不存在。[同前] 如果列表中只有两个活动参考图像，则使用单个经编码的比特来表示参考索引。[同前] 对于其它细节，见 H.264 标准的草案 JVT-D157。

[0102] JVT-D157 的参考图像选择提供了灵活性，并且因而可以改进运动补偿的预测。然而，管理参考图像列表和用信号表示参考图像选择的过程是复杂的，并且在某些情况下消耗许多没有效率的比特。

[0103] B. 用信号表示宏块模式

[0104] 各种标准使用不同的机制来用信号表示宏块信息。例如，在 H.261 标准中，宏块的宏块头部包括宏块类型 MTYPE 元素，它用信号表示为 VLC。[H.261 标准，章节 4.2.3。]MTYPE 元素指示预测模式 (帧内编码、帧间编码、帧间编码 +MC、帧间编码 +MC+ 循环滤波)，对该宏块是否存在量化器 MQUANT 元素，对该宏块是否存在运动矢量数据 MVD 元素，对该宏块是否存在已编码块模式 CBP 元素，以及对该宏块的块是否存在变换系数 TCOEFF 元素。[同前] 对于每个经运动补偿的宏块，都存在 MVD 元素。[同前]

[0105] 在 MPEG-1 标准中，宏块具有 macroblock_type(宏块类型) 元素，它被用信号表示

为 VLC。[MPEG-1 标准，章节 2.4.2.6，表 B.2a 至 B.2d，D.6.4.2。] 对于在前向预测图像中的宏块，macroblock_type 元素指示对该宏块是否存在量化器比例元素，对该宏块是否存在前向运动矢量数据，对该宏块是否存在已编码块模式元素，以及该宏块是否为帧内编码的。[同前] 如果宏块使用前向运动补偿，则前向运动矢量数据总是存在的。[同前]

[0106] 在 H.262 标准中，宏块具有 macroblock_type 元素，它用信号表示为 VLC。[H.261 标准，章节 6.2.5.1，6.3.17.1，以及表 B.2 至 B.B。] 对于在前向预测图像中的宏块，macroblock_type 元素指示对该宏块是否存在 quantizer_scale_code（量化器比例代码）元素，对该宏块是否存在前向运动矢量数据，对该宏块是否存在已编码块模式元素，该宏块是否为帧内编码的，以及该宏块的可缩放性选项。[同前] 如果宏块使用前向运动补偿，则前向运动矢量数据总是存在的。[同前] 独立的代码 (frame_motion_type（帧运动类型）或者 field_motion_type（半帧运动类型）) 可进一步指示宏块预测类型，包括运动矢量的计数和宏块的运动矢量格式。[同前]

[0107] 在 H.263 标准中，宏块具有用于色度 MCBPC 元素的宏块类型和已编码块模式，它用信号表示为 VLC。[H.263 标准，章节 5.3.2，表 8 和 9 以及 F.2。] 宏块类型给出有关该宏块的信息（例如，帧间编码、帧间 4V、帧内编码）。[同前] 对于 帧间编码的图像中经编码的宏块，用于亮度的 MCBPC 和已编码块模式总是存在的，并且宏块类型指示对该宏块是否存在量化器信息元素。前向运动补偿的宏块总是有宏块（或者帧间 4V 类型的块）的运动矢量数据存在。[同前]MPEG-4 标准相似地指定 MCBPC 元素，它用信号表示为 VLC 的。[MPEG-4 标准，章节 6.2.7，6.3.7，11.1.1。]

[0108] 在 JVT-D157 中，mb_type（宏块类型）元素是宏块层的一部分。[JVT-D157，章节 7.3.5 和 7.4.5。]mb_type 指示宏块类型和各种相关联的信息。[同前] 例如，对于 P 片，mb_type 元素指示预测的类型（帧内编码或者前向），当宏块是帧内编码时指示各种帧内模式编码参数，当宏块是前向预测时指示宏块分区（例如，16x16、16x8、8x16、或者 8x8）并且因此指示运动矢量的数量，以及指示参考图像选择信息是否存在（如果分区是 8x8）。[同前] 预测的类型和 mb_type 还合在一起指示宏块是否存在已编码块模式元素。[同前] 对于前向运动补偿的宏块中的每个 16x16、16x8 或 8x16 分区，用信号表示运动矢量数据。[同前] 对于具有 8x8 分区的前向预测宏块，每 8x8 分区一个 sub_mb_type（子宏块类型）元素指示其预测类型（帧内编码或者前向）。[同前] 如果 8x8 分区是前向预测的，则 sub_mb_type 指示子分区（例如，8x8、8x4、4x8 或者 4x4），并且因此指示该 8x8 分区的运动矢量数量。[同前] 对于在前向运动补偿的 8x8 分区中的每个子分区，用信号表示运动矢量数据。[同前]

[0109] 各种标准使用多种多样的信号表示机制用于宏块信息。无论这些信号表示机制有什么优点，它们也都具有下列缺点。首先，它们有时没有有效地用信号表示宏块类型、已编码块模式信息的存在 / 不存在以及运动补偿宏块的运动矢量差分信息的存在 / 不存在。实际上，这些标准一般根本没有用信号表示经运动补偿的宏块（或者其块或半帧）的运动矢量差分信息的存在 / 不存在，代之以假定如果使用运动补偿则用信号表示运动矢量差分信息。最后，这些标准在它们决定要为宏块模式信息使用哪些代码表方面是不灵活的。

[0110] C. 运动矢量预测

[0111] H.261、H.262、H.263、MPEG-1、MPEG-4 和 JVT-D157 的每一个指定某种形式的运动

矢量预测，尽管运动矢量预测的细节在这些标准之间变化很大。例如，在 H.261 标准中运动矢量预测是最简单的，其中当前宏块的运动矢量的运动矢量预测值是先前编码 / 解码的宏块的运动矢量。[H.261 标准，章节 4.2.3.4。] 对于各种特殊 情况（例如，当前宏块是行中的第一个），运动矢量预测值是 0。运动矢量预测与在 MPEG-1 标准中的相似。[MPEG-1 标准，章节 2.4.4.2 和 D.6.2.3。]

[0112] 其它标准（诸如 H.262）指定更复杂的运动矢量预测，但一般仍从单个相邻帧来确定运动矢量预测值。[H.262 标准，章节 7.6.3。] 在运动是一致的时候从单个相邻帧确定运动矢量预测值足够了，但在许多其它情况下是无效的。

[0113] 因此，还有其它标准（诸如 H.263、MPEG-4、JVT-D157）从具有不同候选运动矢量预测值的多个不同的相邻帧来确定运动矢量预测值。[H.263 标准，章节 6.1.1 ;MPEG-4 标准，章节 7.5.5 和 7.6.2 ;以及 F.2 ;JVT-D157，章节 8.4.1。] 这些对多种运动是有效的，但仍不足以处理这样的情况，其中在不同的候选运动矢量预测值之间有高度的不同，指示在运动模式方面的不连续性。

[0114] 对于其它细节，见相应的标准。

[0115] D. 解码运动矢量差分

[0116] H.261、H.262、H.263、MPEG-1、MPEG-4 和 JVT-D157 的每一个指定某种形式的差分运动矢量编码和解码，尽管编码和解码的细节在这些标准之间变化很大。例如，在 H.261 标准中运动矢量编码与解码是最简单的，其中一个 VLC 表示水平差分分量，而另一个 VLC 表示垂直差分分量。[H.261 标准，章节 4.2.3.4。] 其它标准为运动矢量差分信息指定更复杂的编码与解码。对于其它细节，见相应的标准。

[0117] E. 重构与导出运动矢量

[0118] 通常，在 H.261、H.262、H.263、MPEG-1、MPEG-4 或 JVT-D157 中的运动矢量是通过组合运动矢量预测值与运动矢量差分来重构的。再一次，重构的细节在标准之间变化。

[0119] 色度运动矢量（它们未用信号表示）一般是从亮度运动矢量（它们用信号表示）导出的。例如，在 H.261 标准中，二等分亮度运动矢量朝向零截断以导出色度运动矢量。[H.261 标准，章节 3.2.2。] 同样，在 MPEG-1 标准和 JVT-D157 中，二等分亮度运动矢量以导出色度运动矢量。[MPEG-1 标准，章节 2.4.4.2 ;JVT-D157，章节 8.4.1.4。]

[0120] 在 H.262 标准中，按照取决于色度二次采样模式的因子将亮度运动矢量比例缩小至色度运动矢量（例如，4:2:0、4:2:2 或 4:4:4）。[H.262 标准，章节 7.6.3.7。]

[0121] 在 H.263 标准中，对于将单个亮度运动矢量用于全部四个亮度块的宏块，色 度运动矢量是通过将亮度运动矢量除以二并舍入到二分之一象素位置来导出的。[H.263 标准，章节 6.1.1。] 对于具有四个亮度运动矢量（每块一个）的宏块，色度运动矢量是通过累加这四个亮度运动矢量，除以八，并且舍入到二分之一象素位置来导出的。[H.263 标准，章节 F.2。] 在 MPEG-4 标准中相似地导出色度运动矢量。[MPEG- 标准，章节 7.5.5 和 7.6.2。]

[0122] F. 加权的预测

[0123] H.264 标准的草案 JVT-D157 描述加权的预测。图像的加权预测标志指示该图像中的预测片是否使用加权的预测。[JVT-D157，章节 7.3.2.2 和 7.4.2.2。] 如果一个图像使用加权的预测，则该图像中的每个预测片具有一个预测权重表。[JVT-D157，章节 7.3.3，7.3.3.2 和 10.4.1。] 对于该表，用信号表示亮度权重参数的分母和色度权重参数的分母。

[同前]然后，对于可用于该片的每个参考图像，亮度权重标志指示是否为该图像用信号表示亮度参数和亮度偏移分子参数（在用信号表示时，跟随在后面的是这些参数），而色度权重标志指示是否为该图像用信号表示色度权重和色度偏移分子参数（当用信号表示时，跟随在后面的是这些参数）。[同前]将与用信号表示的分子值有关的默认值给予未用信号表示的分子权重参数。[同前]尽管 JVT-D157 在用信号表示加权的预测参数时提供某种灵活性，但这种信号表示机制在各种情况下是无效的。

[0124] 给出了数字视频的视频压缩与解压的关键重要性，就不奇怪视频压缩与解压是被大力开发的领域。然而，无论以前的视频压缩与解压技术有什么好处，它们都没有下面的技术与工具的优点。

发明内容

[0125] 概言之，详细描述针对的是用于编码与解码隔行扫描视频的各种技术与工具。所述的各种技术与工具可以组合地或者独立地使用。

[0126] 详细描述的一些部分针对的是用于隔行扫描前向预测半帧的混合运动矢量预测的各种技术和工具。所述的技术和工具包括但不限于下列：

[0127] 诸如视频编码器或解码器等工具至少部分地基于可应用于运动矢量预测值的预测值极性，来检查混合运动矢量预测条件。例如，预测值极性信号是用于选择运动矢量预测值的主极性或非主极性。该工具随后确定运动矢量预测值。

[0128] 或者，诸如视频编码器或解码器等工具为隔行扫描前向预测半帧的运动矢量 确定初始的、导出的运动矢量预测值。该工具随后至少部分地基于该初始的、导出的运动矢量预测值以及一个或多个相邻的运动矢量来检查变化条件。如果满足变化条件，则该工具使用一个或多个相邻运动矢量之一作为运动矢量的最终运动矢量预测值。否则，该工具使用初始的、导出的运动矢量预测值作为最终的运动矢量预测值。

[0129] 详细描述的一些部分针对的是用于使用运动矢量块模式的各种技术与工具，所述运动矢量块模式用信号表示具有多个运动矢量的宏块的运动矢量数据的存在或不存在。所述的技术与工具包括但不限于下列：

[0130] 诸如视频编码器或解码器等工具处理第一可变长度代码，它表示具有多个亮度运动矢量的宏块的第一信息。第一信息包括宏块的每亮度运动矢量的一个运动矢量数据存在指示符。该工具还处理第二可变长度代码，它表示宏块的第二信息。第二信息包括宏块的多个块的多个变换系数数据存在指示符。

[0131] 或者，诸如视频编码器或解码器等工具对于具有第一数量的亮度运动矢量的宏块（其中第一数量大于 1）处理运动矢量块模式，它由第二数量的比特组成（其中第二数量＝第一数量）。这些比特的每一个指示亮度运动矢量中相应的一个是否已经关联于比特流中用信号表示的运动矢量数据。该工具还对亮度运动矢量的每一个处理相关联的运动矢量数据，对这些亮度运动矢量，指示要在比特流中用信号表示相关联运动矢量数据。

[0132] 详细描述的一些部分针对的是用于在运动矢量预测值的主极性与非主极性之间选择的各种技术与工具。所述技术与工具包括但不限于下列：

[0133] 诸如视频编码器或解码器等工具确定运动矢量预测值的主极性。该工具至少部分地基于主极性处理运动矢量预测值，并且至少部分地基于运动矢量预测值处理运动矢量。

例如，运动矢量是隔行扫描前向预测半帧的当前块或宏块的，并且主极性至少部分地基于相邻块或宏块的多个先前的运动矢量的每一个的极性。

[0134] 或者，诸如视频编码器或解码器等工具处理指示在运动矢量预测值的主极性与非主极性之间选择的信息，并且至少部分地基于运动矢量预测值处理运动矢量。例如，解码器确定主极性和非主极性，随后至少部分地基于主极性和非主极性和指示在它们之间的选择的信息来确定运动矢量预测值。

[0135] 详细描述的一些部分针对的是用于对参考半帧选择信息与差分运动矢量信息进行联合编码和解码的各种技术与工具。所述的技术与工具包括但不限于下列：

[0136] 诸如视频解码器等工具解码可变长度代码，它联合地表示运动矢量的差分运 动矢量信息与运动矢量预测值选择。解码器随后至少部分地基于差分运动矢量信息与运动矢量预测值选择来重构运动矢量。

[0137] 或者，诸如视频编码器等工具确定运动矢量的主 / 非主预测值选择。编码器确定运动矢量的差分运动矢量信息，并且主 / 非主极性选择与差分运动矢量信息一起进行联合编码。

[0138] 详细描述的一些部分针对的是用于代码表选择和隔行扫描前向预测半帧的宏块的宏块模式信息的联合编码 / 解码的各种技术与工具。所述的技术与工具包括但不限于下列：

[0139] 诸如视频编码器或解码器等工具处理可变长度代码，它联合地用信号表示宏块的宏块模式信息。宏块是经运动补偿的，并且联合地用信号表示的宏块模式信息包括：(1) 宏块类型，(2) 已编码块模式存在还是不存在，以及 (3) 经运动补偿的宏块的运动矢量数据存在还是不存在。

[0140] 或者，诸如视频编码器或解码器等工具从隔行扫描前向预测半帧的宏块模式信息的多个可用代码表中选择一个代码表。该工具使用所选择的代码表来处理可变长度代码，它指示宏块的宏块模式信息。宏块模式信息包括 (1) 宏块类型，(2) 已编码块模式存在还是不存在，以及 (3) 当可应用于宏块类型时，运动矢量数据存在还是不存在。

[0141] 详细描述的一些部分针对的是用于使用可用于隔行扫描前向预测半帧的参考半帧数量的信号的各种技术与工具。所述的技术与工具包括但不限于下列：

[0142] 诸如视频编码器或解码器等工具处理第一信号，它指示隔行扫描前向预测半帧是具有一个参考半帧还是两个可能的参考半帧用于运动补偿。如果第一信号指示隔行扫描前向预测半帧具有一个参考半帧，则该工具处理第二信号，它从两个可能的参考半帧中标识一个参考半帧。另一方面，如果第一信号指示隔行扫描前向预测半帧具有两个可能的参考半帧，则对于隔行扫描前向预测半帧的块和 / 或宏块的多个运动矢量的每一个，该工具可处理用于在两个可能的参考半帧之间选择的第三信号。该工具随后执行对隔行扫描前向预测半帧的运动补偿。

[0143] 或者，诸如视频编码器或解码器等工具处理指示隔行扫描前向预测半帧具有一个参考半帧还是两个可能的参考半帧用于运动补偿的信号。该工具执行隔行扫描前向预测半帧的运动补偿。该工具还更新参考半帧缓冲器用于后续运动补偿，而不处理用于管理参考半帧缓冲器的附加信号。

[0144] 详细描述的一些部分针对的是用于导出隔行扫描前向预测半帧的宏块的色度 运

动矢量的各种技术与工具。所述技术与工具包括但不限于下列：

[0145] 诸如视频编码器或解码器等工具对于具有一个或多个亮度运动矢量的宏块，至少部分地基于一个或多个亮度运动矢量的极性估计来导出色度运动矢量。例如，一个或多个亮度运动矢量的每一个是奇或偶极性，并且极性估计包括确定哪个极性在一个或多个亮度运动矢量中是最共同的。

[0146] 或者，诸如视频编码器或解码器等工具确定宏块的多个亮度运动矢量之中的占优势极性。该工具随后至少部分地基于具有占优势极性的多个亮度运动矢量中的一个或多个来导出宏块的色度运动矢量。

[0147] 其它特征与优点将通过下列参考附图进行的不同实施例的详细描述而变得明显。

附图说明

[0148] 图 1 是示出依照现有技术的视频编码器中的运动估计的图示。

[0149] 图 2 是示出依照现有技术的视频解码器中的 8×8 的预测残差块的基于块的压缩的图示。

[0150] 图 3 是示出依照现有技术的视频编码器中的 8×8 的预测残差块的基于块的解压的图示。

[0151] 图 4 是示出依照现有技术的隔行扫描帧的图示。

[0152] 图 5A 和 5B 是示出依照现有技术，用于逐行扫描 P 帧中的 1MV 宏块的候选运动矢量预测值的宏块位置的图示。

[0153] 图 6A 和 6B 是示出依照现有技术，用于混合的 1MV/4MV 逐行扫描 P 帧中的 1MV 宏块的候选运动矢量预测值的块位置的图示。

[0154] 图 7A、7B、8A、8B、9 和 10 是示出依照现有技术，用于混合的 1MV/4MV 逐行扫描 P 帧中的 4MV 宏块中的各种位置处的块的候选运动数量预测值的块位置的图示。

[0155] 图 11 是示出依照现有技术，用于隔行扫描 P 帧中的当前帧编码宏块的候选运动矢量预测值的图示。

[0156] 图 12A-12B 是示出依照现有技术，用于隔行扫描 P 帧中的当前半帧编码宏块的候选运动矢量预测值的图示。

[0157] 图 13A-13C 是按照现有技术计算运动矢量预测值的伪代码。

[0158] 图 14A 和 14B 是示出按照现有技术用于逐行扫描 P 帧的混合运动矢量预测的 伪代码。

[0159] 图 15A-15C 是示出按照现有技术的运动矢量差分信息的解码的伪代码和表。

[0160] 图 16A-16C 和 13C 是示出按照现有技术用于逐行扫描 P 帧的色度运动矢量的导出的伪代码。

[0161] 图 17 是示出按照现有技术用于隔行扫描 P 帧的色度运动矢量的导出的伪代码。

[0162] 图 18 是示出按照现有技术用于逐行扫描 P 帧的强度补偿的伪代码。

[0163] 图 19 是可结合其来实现若干描述的实施例的合适计算环境的框图。

[0164] 图 20 是结合其可实现若干描述的实施例的广义视频编码器系统的框图。

[0165] 图 21 是结合其可实现若干描述的实施例的广义视频解码器系统的框图。

[0166] 图 22 是在若干描述的实施例中使用的宏块格式的图。

[0167] 图 23A 是隔行扫描视频帧的部分图，示出上半帧和下半帧的交替行。图 23B 是为编码 / 解码为帧而组织的隔行扫描视频帧的图，以及图 23C 是为编码 / 解码为半帧而组织的隔行扫描视频帧的图。

[0168] 图 24A-24F 是示出隔行扫描 P 半帧的参考半帧的示例的图。

[0169] 图 25A 和 25B 是分别示出用于参考半帧数量和选择信息的编码和解码的技术的流程图。

[0170] 图 26 和 27 是示出 MBMODE 值的表。

[0171] 图 28A 和 28B 是分别示出隔行扫描 P 半帧的宏块的宏块模式信息的编码和解码的技术的流程图。

[0172] 图 29 是用于确定主和非主参考半帧的伪代码。

[0173] 图 30 是用于用信号表示使用主还是非主参考半帧用于运动矢量的伪代码。

[0174] 图 31A 和 31B 是分别示出用于在编码和解码中确定两个参考半帧隔行扫描 P 半帧的运动矢量的运动矢量预测的主和非主极性的技术的流程图。

[0175] 图 32 是在解码期间的混合运动矢量预测的伪代码。

[0176] 图 33A 和 33B 是分别示出在编码和解码期间的混合运动矢量预测的技术的流程图。

[0177] 图 34 是示出亮度块与 4MVBP 元素之间的关联的图。

[0178] 图 35A 和 35B 是分别示出使用运动矢量块模式的编码和解码的技术的流程图。

[0179] 图 36 是用于编码两个参考半帧隔行扫描 P 半帧的运动矢量差分信息和主 / 非主预测值选择的伪代码。

[0180] 图 37A 和 37B 是分别示出两个参考半帧隔行扫描 P 半帧的运动矢量差分信息和主 / 非主预测值选择的编码和解码的技术的流程图。

[0181] 图 38 是 4:2:0 宏块的色度二次采样模式的图。

[0182] 图 39 是示出垂直运动矢量分量的当前与参考半帧之间的关系的图。

[0183] 图 40 是用于选择为隔行扫描 P 半帧的经运动补偿的宏块的色度运动矢量作出贡献的亮度运动矢量的伪代码。

[0184] 图 41 是示出从隔行扫描 P 半帧的宏块的亮度运动矢量中导出色度运动矢量的技术的流程图。

[0185] 图 42 和 43 分别是编码器框架和解码器框架的图，其中为隔行扫描 P 半帧执行强度补偿。

[0186] 图 44 是示出用于用信号表示隔行扫描 P 半帧的强度补偿参考半帧模式的句法元素的表。

[0187] 图 45A 和 45B 是分别示出对隔行扫描 P 半帧在编码中执行褪色估计和在解码中执行褪色补偿的技术的流程图。

[0188] 图 46A-46E 是按照第一组合实现的比特流的层的句法图。

[0189] 图 47A-47K 是在第一组合实现中的代码表。

[0190] 图 48 是示出在第一组合实现中的垂直运动矢量分量的当前与参考半帧之间的关系的图。

[0191] 图 49A 和 49B 分别是用于第一组合实现中的 1 参考半帧隔行扫描 P 半帧的运动矢

量差分解码的伪代码和表。

[0192] 图 50 是用于解码第一组合实现中的 2 参考半帧隔行扫描 P 半帧的运动矢量差分信息和主 / 非主预测值选择的伪代码。

[0193] 图 51A 和 51B 是用于第一组合实现中的 1 参考半帧隔行扫描 P 半帧的运动矢量预测的伪代码。

[0194] 图 52A-52J 是用于第一组合实现中的 2 参考半帧隔行扫描 P 半帧的运动矢量预测的伪代码和表。图 52K 至 52N 是用于比例缩放操作的伪代码和表，它们是图 52H 至 52J 所示操作的替换操作。

[0195] 图 53 是用于第一组合实现中的隔行扫描 P 半帧的混合运动矢量预测的伪代码。

[0196] 图 54 是用于第一组合实现中的 2 参考半帧隔行扫描 P 半帧的运动矢量重构的 伪代码。

[0197] 图 55A 和 55B 是用于第一组合实现中的隔行扫描 P 半帧的色度运动矢量导出的伪代码。

[0198] 图 56 是用于第一组合实现中的隔行扫描 P 半帧的强度补偿的伪代码。

[0199] 图 57A-57C 是用于按照第二组合实现的比特流的层的句法图。

[0200] 图 58A 和 58B 分别是用于第二组合实现中的 1 参考半帧隔行扫描 P 半帧的运动矢量差分解码的伪代码和表。

[0201] 图 59 是用于解码第二组合实现中的 2 参考半帧隔行扫描 P 半帧的运动矢量差分信息和主 / 非主预测值选择的伪代码。

[0202] 图 60A 和 60B 是用于第二组合实现中的 1 参考半帧隔行扫描 P 半帧的运动矢量预测的伪代码。

[0203] 图 61A-61F 是用于第二组合实现中的 2 参考半帧隔行扫描 P 半帧的运动矢量预测的伪代码。

具体实施方式

[0204] 本申请涉及用于有效地压缩和解压隔行扫描视频的技术与工具。隔行扫描视频内容的压缩与解压随着各种技术与工具一起改进，这些技术与工具明确地被设计成处理隔行扫描视频表示的特定属性。在各种描述的实施例中，视频编码器和解码器结合了用于编码和解码隔行扫描前向预测半帧的技术，以及用于包括不同层或级（例如，序列级、帧级、半帧级、片级、宏块级和 / 或块级）的比特流格式或句法的相应信号表示技术。

[0205] 隔行扫描视频内容一般在通过电缆、卫星或 DSL 的数字视频广播系统中使用。用于压缩和解压隔行扫描视频内容的有效技术与工具是视频编解码器的重要部分。

[0206] 在此描述的实现的各种替换方案是可能的。例如，参考流程图描述的技术可以通过改变流程图中所示步骤的顺序、通过重复或省略某些步骤等来改变。作为另一个示例，尽管参考特定的宏块格式描述了某些实现，但也可以使用其它格式。而且，参考隔行扫描前向预测半帧描述的技术与工具也可应用于其它类型的图像。

[0207] 在各种实施例中，编码器和解码器比特流中使用了标志和 / 或信号。尽管描述了特定的标志与信号，但应该理解，这种描述方式包括了标志与信号的不同约定（例如，0 而非 1）。

[0208] 各种技术与工具可以组合或独立使用。不同实施例实现所述技术与工具中的 一个或多个。在此描述的有些技术与工具可以在视频编码器或解码器中使用，或者在不是明确限于视频编码或解码的某种其它系统中使用。

[0209] I. 计算环境

[0210] 图 19 示出了适合在其中实现所描述的若干实施例的计算环境 (1900) 的一个广义示例。计算环境 (1900) 并非对使用范围或功能提出任何局限，因为这些技术和工具可以在完全不同的通用或专用计算环境中实现。

[0211] 参考图 19，计算环境 (1900) 包括至少一个处理单元 (1910) 和存储器 (1920)。在图 19 中，这一最基本配置 (1930) 包括在虚线内。处理单元 (1910) 执行计算机可执行指令，且可以是真实或虚拟处理器。在多处理系统中，多个处理单元执行计算机可执行指令以提高处理能力。存储器 (1920) 可以是易失性存储器 (例如，寄存器、高速缓存、RAM)、非易失性存储器 (例如，ROM、EEPROM、闪存等) 或两者的某一组合。存储器 (1920) 储存实现视频编码器或解码器的软件 (1980)。

[0212] 计算环境可具有额外的特征。例如，计算环境 (1900) 包括存储 (1940)、一个或多个输入设备 (1950)、一个或多个输出设备 (1960) 以及一个或多个通信连接 (1970)。诸如总线、控制器或网络等互连机制 (未示出) 将计算环境 (1900) 的组件互连。通常，操作系统软件 (未示出) 为在计算环境 (1900) 中执行的其它软件提供了操作环境，并协调计算环境 (1900) 的组件的活动。

[0213] 存储 (1940) 可以是可移动或不可移动的，且包括磁盘、磁带或磁带盒、CD-ROM、DVD 或可用于储存信息并可在计算环境 (1900) 内访问的任何其它介质。存储 (1940) 储存用于软件 (1980) 实现视频编码器或解码器的指令。

[0214] 输入设备 (1950) 可以是诸如键盘、鼠标、笔或跟踪球等触摸输入设备、语音输入设备、扫描设备或可向计算环境 (1900) 提供输入的另一设备。对于音频或视频编码，输入设备 (1950) 可以是声卡、显卡、TV 调谐卡、或接受模拟或数字格式的音频或视频输入的类似的设备、或将音频或视频样值读入计算环境 (1900) 的 CD-ROM 或 CD-RW。输出设备 (1960) 可以是显示器、打印机、扬声器、CD 刻录机、或从计算环境 (1900) 提供输出的另一设备。

[0215] 通信连接 (1970) 允许通过通信介质到另一计算实体的通信。通信介质传达诸如计算机可执行指令、音频或视频输入或输出、或已调制数据信号形式的其它数据等信息。已调制数据信号是其一个或多个特征以在信号中编码信息的方式设置或改变的信号。作为示例而非局限，通信介质包括以电、光、RF、红外、声学或其 它载波实现的有线或无线技术。

[0216] 各种技术和工具可以在计算机可读介质的一般上下文中描述。计算机可读介质可以是可在计算环境内访问的任何可用介质。作为示例而非局限，对于计算环境 (1900)，计算机可读介质包括存储器 (1920)、存储 (1940)、通信介质以及上述任一个的组合。

[0217] 各种技术和工具可以在诸如程序模块中所包括的在计算环境中的目标真实或虚拟处理器上执行的计算机可执行指令的一般上下文中描述。一般而言，程序模块包括例程、程序、库、对象、类、组件、数据结构等，它们执行特定任务或实现特定抽象数据类型。程序模块的功能可以如各实施例中所需的组合或在程序模块之间分离。用于程序模块的计算机可执行指令可以在本地或分布式计算环境中执行。

[0218] 为演示起见，详细描述使用了如“估计”、“补偿”、“预测”和“应用”等术语，来描述

计算环境中的计算机操作。这些术语是由计算机执行的操作的高级抽象，且不应与人类所执行的动作混淆。对应于这些术语的实际的计算机操作取决于实现而不同。

[0219] II. 广义的视频编码器和解码器

[0220] 图 20 是可结合其实现所描述的各种实施例的广义视频编码器系统 (2000) 的框图。图 21 是可结合其实现所描述的各种实施例的广义视频解码器 (2100) 的框图。

[0221] 编码器 (2000) 和解码器 (2100) 内的模块之间所示的关系指示了编码器和解码器中的主要信息流；为简明起见，未示出其它关系。具体地，图 20 和 21 一般不示出指示用于视频序列、帧、宏块、块等的编码器设置、模式、表等辅助信息。这一辅助信息通常在该辅助信息的熵编码之后在输出比特流中发送。输出比特流的格式可以是 Windows Media Video 版本 9 或其它格式。

[0222] 编码器 (2000) 和解码器 (2100) 处理视频图像，视频图像可以是视频帧、视频半帧或帧和半帧的组合。图像和宏块级的比特流句法和语法可取决于使用了帧还是半帧。也可以对宏块组织和总体时序有改变。编码器 (2000) 和解码器 (2100) 是基于块的，且对帧使用 4:2:0 的宏块格式，其中每一宏块包括四个 8×8 的亮度块（有时候作为一个 16×16 的宏块来对待）以及两个 8×8 的色度块。对于半帧，可使用相同或不同的宏块组织和格式。8×8 的块还可在不同的级细分，例如在频率变换和熵编码级。示例性视频帧组织在下一节中更详细描述

[0223] 取决于所需的实现和压缩类型，编码器或解码器的模块可被添加、省略、分成多个模块、与其它模块组合、和 / 或用相似的模块来替代。在替换实施例中，具有不同模块和 / 或其它模块配置的编码器或解码器执行一个或多个所描述的技术。

[0224] A. 视频帧组织

[0225] 在某些实现中，编码器 (2000) 和解码器 (2100) 处理如下组织的视频帧。帧包含视频信号的空间信息行。对于逐行扫描视频，这些行包含从一个时刻开始并继续通过连续的行直到帧底部的样值。逐行扫描视频帧被划分成诸如图 22 所示的宏块 (2200) 等宏块。宏块 (2200) 包括四个 8×8 的亮度块 (Y1 到 Y4) 以及两个 8×8 的色度块，这些色度块与四个亮度块共同定位，但是水平和垂直分辨率都是一半，遵循常规的 4:2:0 的宏块格式。8×8 的块还可以在不同的级上细分，例如在频率变换级（例如，8×4、4×8 或 4×4DCT）和熵编码级。逐行扫描 I 帧是帧内编码的逐行扫描视频帧。逐行扫描 P 帧是使用前向预测编码的逐行扫描视频帧，而逐行扫描 B 帧是使用双向预测编码的逐行扫描视频帧。逐行扫描 P 帧和 B 帧可包括帧内编码的宏块以及不同类型的预测宏块。

[0226] 隔行扫描视频帧由一帧的两次扫描构成 - 一次包括帧的偶数行（上半帧），另一次包括帧的奇数行（下半帧）。这两个半帧可表示两个不同的时间段，或者它们可以来自同一时间段。图 23A 示出了隔行扫描视频帧 (2300) 的一部分，包括位于隔行扫描视频帧 (2300) 的左上部分的上半帧和下半帧的交替行。

[0227] 图 23B 示出了为编码 / 解码组织为帧 (2330) 的图 23A 的隔行扫描视频帧 (2300)。隔行扫描视频帧 (2300) 被划分成诸如宏块 (2331) 和 (2332) 等宏块，它们使用如图 22 所示的 4:2:0 的格式。在亮度平面中，每一宏块 (2331、2332) 包括来自上半帧的 8 行，这 8 行与来自下半帧的 8 行交替，总共有 16 行，且每一行是 16 个像素长。（宏块 (2331、2332) 内亮度块和色度块的实际组织和布置未示出，且实际上可以对不同的编码决策不同。）在给定宏

块内，上半帧信息和下半帧信息可以联合编码或在各种阶段的任一个单独编码。隔行扫描 I 帧是隔行扫描视频帧的两个帧内编码的半帧，其中宏块包括关于这两个半帧的信息。隔行扫描 P 帧是使用前向预测编码的隔行扫描视频帧的两个半帧，而隔行扫描 B 帧是使用双向预测编码的隔行扫描视频帧的两个半帧，其中宏块包括关于这两个半帧的信息。隔行扫描 P 帧和 B 帧可包括帧内编码宏块以及不同类型的预测宏块。

[0228] 图 23C 示出了为编码 / 解码被组织成半帧 (2360) 的图 23A 的隔行扫描视频帧 (2300)。隔行扫描视频帧 (2300) 的两个半帧中的每一个被划分成宏块。上半帧被划分成诸如宏块 (2361) 等宏块，下半帧被划分成诸如宏块 (2362) 等宏块。(这些宏块也使用如图 22 所示的 4:2:0 格式，且宏块内亮度块和色度块的组织和布置未示出)。在亮度平面中，宏块 (2361) 包括来自上半帧的 16 行，且宏块 (2362) 包括来自下半帧的 16 行，且每一行是 16 个像素长。隔行扫描 I 半帧是隔行扫描视频帧的单个单独表示的半帧。隔行扫描 P 半帧是使用前向预测编码的隔行扫描视频帧的单个单独表示的半帧，隔行扫描 B 半帧是使用双向预测编码的隔行扫描视频帧的单个单独表示的半帧。隔行扫描 P 半帧和 B 半帧可包括帧内编码宏块以及不同类型的预测宏块。

[0229] 术语图像一般指的是源、已编码的或已重构的图像数据。对于逐行扫描视频，图像是逐行扫描视频帧。对于隔行扫描视频，图像可以指的是隔行扫描视频帧、帧的上半帧、或帧的下半帧，取决于上下文。

[0230] 或者，编码器 (2000) 和解码器 (2100) 是基于对象的，使用不同的宏块或块格式，或对与 8×8 的块和 16×16 的宏块不同大小或配置的像素集执行操作。

[0231] B. 视频编码器

[0232] 图 20 是广义的视频编码器系统 (2000) 的框图。编码器系统 (2000) 接收包括当前图像 (2005)(例如，逐行扫描视频帧、隔行扫描视频帧或隔行扫描视频帧的半帧) 的视频图像序列，并产生压缩的视频信息 (2095) 作为输出。视频编码器的具体实施例通常使用广义编码器 (2000) 的变化或补充版本。

[0233] 编码器系统 (2000) 压缩预测图像和关键图像。为演示起见，图 20 示出了关键图像通过编码器系统 (2000) 的路径以及用于前向预测图像的路径。编码器系统 (2000) 的许多组件用于同时压缩关键图像和预测图像两者。由这些组件执行的确切操作可以取决于所压缩的信息类型而变化。

[0234] 预测图像 (对双向预测也称为 P 图像或 B 图像，或帧间编码图像) 按照来自一个或多个其它图像的预测 (或差) 来表示。预测残差是所预测的和原始图像之差。相反，关键图像 (也称为 I 图像或帧内编码图像) 不参考其它图像来压缩。

[0235] 如果当前图像 (2005) 是前向预测图像，则运动估计器 (2010) 相对于参考图像来估计宏块的运动或者当前图像 (2005) 的其它象素集，参考图像是在图像存储器 (2020) 中缓冲的重构的先前图像 (2025)。在一替换实施例中，参考图像是后面的图像或者当前图像是双向预测的。运动估计器 (2010) 可以按象素、1/2 象 素、1/4 象素或其它增量进行估计，并且可以在逐个图像的基础上或者在其它基础上切换运动估计的精度。运动估计的精度在水平与垂直方向上可以相同或不同。运动估计器 (2010) 输出运动信息 (2015)，如运动矢量作为辅助信息。运动补偿器 (2030) 将运动信息 (2015) 应用于重构的先前图像 (2025) 以形成经运动补偿的当前图像 (2035)。然而预测很少是完美的，并且在经运动补偿的当前图

像 (2035) 与原始当前图像 (2005) 之间的差是预测残差 (2045)。可供替换地，运动估计器和运动补偿器应用另一类型的运动估计 / 补偿。

[0236] 频率变换器 (2060) 将空间域视频信息转换成频域 (即，频谱) 数据。对于基于块的视频图像，频率变换器 (2060) 向像素数据或预测残差数据的块应用 DCT、DCT 的变体，从而产生 DCT 换系数块。或者，频率变换器 (2060) 应用诸如傅立叶变换等另一常规频率变换或使用小波或子带分析。频率变换器 (2060) 向预测的图像的预测残差应用 8×8、8×4、4×8、4×4 或其它大小的频率变换 (例如，DCT)。

[0237] 量化器 (2070) 然后量化频谱数据系数块。量化器向频谱数据应用均匀的标量量化，其步长在逐图像的基础或其它基础上变化。或者，量化器向频谱数据系数应用另一类型的量化，例如非均匀的、矢量或非自适应量化，或直接在不使用频率变换的编码器系统中量化空间域数据。除自适应量化之外，编码器 (2000) 可使用帧丢弃、自适应滤波或其它技术用于速率控制。

[0238] 如果预测的图像中的给定宏块没有某些类型的信息 (例如，没有宏块的运动信息以及没有残差信息)，则编码器 (2000) 可将该宏块编码为跳过宏块。如果这样，则编码器在压缩的视频信息 (2095) 的输出比特流中用信号表示该跳过宏块。

[0239] 当需要重构的当前图像用于后续的运动估计 / 补偿时，反量化器 (2076) 在量化的频谱数据系数上执行反量化。反频率变换器 (2066) 然后执行频率变换器 (2060) 的逆运算，从而产生重构的预测残差 (对于预测图像) 或重构的样值 (对帧内编码的图像)。如果所编码的图像 (2005) 是预测的图像，则重构的预测残差被加到经运动补偿的预测 (2035) 以形成重构的当前图像。图像存储 (2020) 缓冲重构的当前图像，以预测下一图像中使用。在某些实施例中，编码器向重构的帧应用分块滤波器，以自适应地平滑帧的块之间的不连续性。

[0240] 熵编码器 (2080) 压缩量化器 (2070) 的输出以及某些辅助信息 (例如，运动信息 (2015)、量化步长)。典型的熵编码技术包括算术编码、差分编码、哈夫曼编码、行程长度编码、LZ 编码、字典式编码以及上述的组合。熵编码器 (2080) 通常对不同种类的信息 (例如，DC 系数、AC 系数、不同种类的辅助信息) 使用 不同的编码技术，并可从特定编码技术内的多个代码表中进行选择。

[0241] 熵编码器 (2080) 将压缩的视频信息 (2095) 置于缓冲器 (2090) 中。缓冲器级别指示符被反馈给比特率自适应模块。压缩的视频信息 (2095) 以恒定或相对恒定的比特率从缓冲器 (2090) 中消耗完并且存储用于以该比特率的后续流传送。因此，缓冲器 (2090) 的级别主要是经滤波的、量化的视频信息的熵的函数，它影响熵编码的效率。可供替换地，编码器系统 (2000) 紧接着压缩流传送压缩的视频信息，并且缓冲器 (2090) 的级别还依赖于哪些信息从缓冲器 (2090) 消耗完用于传输。

[0242] 在缓冲器 (2090) 之前或之后，压缩的视频信息 (2095) 可被信道编码用于通过网络发送。信道编码可向压缩的视频信息 (2095) 应用检错和纠错数据。

[0243] C. 视频解码器

[0244] 图 21 是通用视频解码器系统 (2100) 的框图。解码器系统 (2100) 接收关于压缩的视频图像序列的信息 (2205)，并产生包括重构的图像 (2105)(例如，逐行扫描视频帧、隔行扫描视频帧或隔行扫描视频帧的半帧) 的输出。视频解码器的具体实施例通常使用广义

解码器 (2100) 的变体或补充版本。

[0245] 解码器系统 (2100) 解压预测图像和关键图像。为演示起见，图 21 示出了关键图像通过解码器系统 (2100) 的路径以及用于前向预测图像的路径。解码器系统 (2100) 的许多组件用于解压关键图像和预测图像。由这些组件执行的确切操作可以取决于所解压的信息类型而变化。

[0246] 缓冲器 (2190) 接收关于压缩的视频序列的信息 (2195)，并使得所接收的信息对熵解码器 (2180) 可用。缓冲器 (2190) 通常以随时间相对恒定的速率接收该信息，并包括抖动缓冲器以平滑带宽或传输中的短期变化。缓冲器 (2190) 也可包括回放缓冲器。或者，缓冲器 (2190) 以变化的速率接收信息。在缓冲器 (2190) 之前或之后，压缩的视频信息可以被信道解码，并被处理用于检错和纠错。

[0247] 熵解码器 (2180) 对熵编码的量化数据以及熵编码的辅助信息 (例如，运动信息 (2115)、量化步长) 进行解码，通常应用编码器中执行的熵编码的逆运算。熵解码技术包括算术解码、差分解码、哈夫曼解码、行程长度解码、LZ 解码、字典式解码以及上述的组合。熵解码器 (2180) 通常对不同种类的信息 (例如，DC 系数、AC 系数、不同种类的辅助信息) 使用不同的解码技术，并可从特定的解码技术中的多个代码表之中进行选择。

[0248] 如果要重构的图像 (2105) 是前向预测图像，则运动补偿器 (2130) 向参考图像 (2125) 应用运动信息 (2115)，以形成所重构的图像 (2105) 的预测 (2135)。例如，运动补偿器 (2130) 使用宏块运动矢量以找出参考图像 (2125) 中的宏块。图像缓冲器 (2120) 储存先前重构的图像以用作参考图像。运动补偿器 (2130) 可以按像素、1/2 像素、1/4 像素或其它增量来补偿运动，并可在逐图像的基础或其它基础上切换运动补偿的精度。运动补偿的精度可以在水平和垂直上相同或不同。或者，运动补偿器应用另一类型的运动补偿。运动补偿器的预测很少是完美的，因此解码器 (2100) 也重构预测残差。

[0249] 当解码器需要重构的图像用户随后的运动补偿时，图像存储 (2120) 缓冲重构的图像以在预测下一图像时使用。在某些实施例中，编码器向重构的帧应用分块滤波器以自适应地平滑帧的块之间的不连续性。

[0250] 反量化器 (2170) 对熵解码的数据进行反量化。一般而言，反量化器向熵解码的数据应用均匀的标量反量化，其中步长在逐图像的基础或其它基础上变化。或者，反量化器向数据应用另一类型的反量化，例如非均匀矢量量化或非自适应反量化，或直接在不使用反频率变换的解码器系统中对空间域数据进行反量化。

[0251] 反频率变换器 (2160) 将量化的频域数据转换成空间域视频信息。对于基于块的视频图像，反频率变换器 (2160) 向 DCT 系数块应用 IDCT 或 IDCT 的变体，从而分别对关键图像或预测图像产生像素数据或预测残差数据。或者，反频率变换器 (2160) 应用另一常规的反频率变换，诸如傅立叶反变换或使用小波或子带合成。反频率变换器 (2160) 向预测的图像的预测残差应用 8×8、8×4、4×8、4×4 或其它大小的反频率变换 (例如，IDCT)。

[0252] III. 隔行扫描 P 帧

[0253] 典型的隔行扫描视频帧由在不同的时刻扫描的两个半帧 (例如，上半帧和下半帧) 构成。一般而言，通过将两个半帧一起编码 (“帧模式”编码) 来对隔行扫描视频帧的静止区域进行编码是更有效的。另一方面，通过单独对各半帧编码 (“半帧模式”编码) 来对隔行扫描视频帧的运动区域进行编码通常是更有效的，因为两个半帧往往具有不同的运

动。前向预测的隔行扫描视频帧可以被编码为两个单独的前向预测的半帧—隔行扫描 P 半帧。对前向预测的隔行扫描视频帧单独地编码半帧在例如贯穿该隔行扫描视频帧中有高运动，且因此在半帧之间有较多差异时可能是有效的。

[0254] 或者，前向预测的隔行扫描视频帧可使用半帧编码和帧编码的混合来编码，作为隔行扫描的 P 帧。对于隔行扫描 P 帧的宏块，宏块包括上半帧和下半帧的像素行，且这些行可以在帧编码模式中共同编码或者在半帧编码模式中单独编码。

[0255] 隔行扫描 P- 半帧参考一个或多个先前解码的半帧。例如，在一些实现中，隔行扫描 P 半帧参考一个或两个先前解码的半帧，然而隔行扫描 B 半帧参考至多两个先前和两个将来的参考半帧（即，最多总共四个参考半帧）。（隔行扫描 P 半帧的编码和解码技术在下面详细描述。）或者，按照一些实施例，尤其对于有关隔行扫描 P 半帧和 2 参考隔行扫描 P 半帧的更多信息，见 2004 年 5 月 27 日提交的美国专利申请第 10/857,473 号，其标题为“Predicting Motion Vectors for Fields ofForward-predicted Interlaced Video Frames（为前向预测的隔行扫描视频帧的半帧预测运动矢量）”。

[0256] IV. 隔行扫描 P 半帧中的参考半帧的数量

[0257] 在一些实施例中，当执行单个当前隔行扫描 P 半帧的运动补偿预测时，两个先前编码 / 解码的半帧可以用作参考半帧。通常，使用两个参考半帧的能力产生比当运动补偿的预测限于一个参考半帧时更好的压缩效率。然而，当两个参考半帧可用时，信号表示开销较高，因为发送额外的信息以指示两个半帧中的哪一个为具有运动矢量的每个宏块或块提供参考。

[0258] 在某些情况下，每运动矢量具有更多可能的运动补偿预测值（两个参考半帧与一个参考半帧相比）的好处不会超出用信号表示参考半帧选择所需要的开销。例如，当最佳的参考全部来自两个可能的参考半帧之一时，选择使用单个参考半帧代替两个可以是有利的。这通常是由于只引起两个参考半帧之一来自与当前半帧相同的场景的场景变化。或者，只有一个参考半帧可用，诸如在序列的开始处。在这些情况下，在当前 P 半帧的半帧级上用信号表示只使用一个参考半帧并且这一个参考半帧是什么，以及让这个决定应用于当前 P 半帧内的宏块和块是更有效的。参考半帧选择信息随后不再需要与每个具有运动矢量的宏块或块一起发送。

[0259] A. 在不同方案中参考半帧的数量

[0260] 一种方案允许两个先前编码 / 解码的半帧用作当前 P 半帧的参考半帧。对运动矢量用信号表示该运动矢量（用于宏块或块）使用的参考半帧，如同该运动矢量的其它信息一样。例如，对于运动矢量，用信号表示的信息指示 :(1) 参考半帧 ； 以及 (2) 在与该运动矢量相关联的当前块或宏块的块或宏块预测值的参考半帧中位置。或者，参考半帧信息和运动矢量信息如在章节 XII 的组合实现之一中描述的那样用信号表示。

[0261] 在另一个方案中，只有一个先前编码 / 解码的半帧用作当前 P 半帧的参考半帧。对于运动矢量，不需要指示该运动矢量参考的参考半帧。例如，对于运动矢量，用信号表示的信息只指示在与该运动矢量相关联的当前块或宏块的预测值的参考半帧中的位置。或者，运动矢量信息如在章节 XII 的组合实现之一中描述的那样用信号表示。在一个参考半帧方案中的运动矢量一般用比在两个参考半帧方案中相同的运动矢量较少的比特来编码。

[0262] 对于任一方案，用于后续运动补偿的参考半帧的缓冲器和图像存储器的更新是简

单的。当前 P 半帧的一个或多个参考半帧是在当前 P 半帧之前的最近和第二最近的 I 或 P 半帧之一或两者。由于候选参考半帧的位置是已知的，编码器和解码器可自动并且在不缓冲管理信号的情况下更新下一个 P 半帧的运动补偿的参考图像缓冲器。

[0263] 可供替换地，编码器和解码器使用用于隔行扫描 P- 半帧的多个参考半帧的一个或多个其它方案。

[0264] B. 信号表示示例

[0265] 在本章节和在章节 XII 的组合实现中描述的信号表示的特定示例用信号表示多少参考半帧用于当前 P 半帧，以及当使用一个参考半帧时，使用哪个候选参考半帧。例如，在 P 半帧头部中的 1 比特字段（称为 NUMREF）指示 P 半帧使用一个还是两个先前的半帧作为参考。如果 NUMREF ＝ 0，则只使用一个参考半帧。如果 NUMREF ＝ 1，则使用两个参考半帧。如果 NUMREF ＝ 0，则另一个 1 比特字段（称为 REFFIELD）存在，并且指示使用两个半帧中的哪一个作为参考。如果 REFFIELD ＝ 0，则使用在时间上更接近的半帧作为参考半帧。如果 REFFIELD ＝ 1，则使用两个候选参考半帧中在时间上更远的那一个作为当前 P 半帧的参考半帧。可供替换地，编码器和解码器使用其它和 / 或附加信号用于参考半帧选择。

[0266] C. 参考半帧的位置

[0267] 图 24A-24F 示出可用于在隔行扫描 P 半帧的运动补偿预测中使用的参考半帧的位置。P 半帧可以使用一个或两个先前编码 / 解码的半帧作为参考。具体地，图 24A-24F 示出 NUMREF ＝ 0 和 NUMREF ＝ 1 的参考半帧的示例。

[0268] 图 24A 和 24B 示出其中对当前 P 半帧使用两个参考半帧的示例。(NUMREF ＝ 1。) 在图 24A 中，当前半帧参考在时间上先前的的上半帧和下半帧。中间的隔行扫描 B 半帧不用作参考半帧。在图 24B 中，当前半帧参考隔行扫描视频帧中的上半帧与下半帧，该隔行扫描视频帧就在包含当前半帧的隔行扫描视频帧之前。

[0269] 图 24C 和 24D 示出示例，其中对当前 P 半帧使用一个参考半帧 (NUMREF ＝ 0)，并且这个参考半帧是时间上最近的参考半帧 (REFFIELD ＝ 0)。参考半帧的极性与当前 P 半帧的极性相反，这意味着，例如，如果当前 P 半帧来自偶数行，则参考半帧来自奇数行。在图 24C 中，当前半帧参考时间上先前的隔行扫描视频帧中的下半帧，并且不参考在该隔行扫描视频帧中的不太近的上半帧。再一次，中间的隔行扫描 B 半帧不允许作为参考半帧。在图 24D 中，当前半帧参考隔行扫描视频帧中的下半帧，它就在包含当前半帧的隔行扫描视频帧之前，而不是不太近的上半帧。

[0270] 图 24E 和 24F 示出示例，其中对当前 P 半帧使用一个参考半帧 (NUMREF ＝ 0)，并且该一个参考半帧是时间上第二最近的参考半帧 (REFFIELD ＝ 1)。参考半帧的极性与当前半帧的极性相同，意味着，例如，如果当前半帧来自偶数行，则参考半帧也来自偶数行。在图 24E 中，当前半帧参考一个时间上先前的隔行扫描视频帧中的上半帧，但不参考更近的下半帧。再一次，中间的隔行扫描 B 半帧不允许作为参考半帧。在图 24F 中，当前半帧参考上半帧而不是更近的下半帧。

[0271] 可供替换地，编码器和解码器使用在其它和 / 或附加位置或定时的参考半帧，用于隔行扫描 P 半帧的运动补偿预测。例如，允许同一帧内的参考半帧作为当前 P- 半帧。或者，帧的上半帧或下半帧可以首先被编码 / 解码。

[0272] D. 编码技术

[0273] 诸如图 20 的编码器 (2000) 等编码器用信号表示多个参考半帧方案中的哪一个用于编码隔行扫描 P 半帧。例如，编码器执行图 25A 中所示的技术 (2500)。

[0274] 对于给定的隔行扫描 P 半帧，编码器用信号表示 (2510) 在隔行扫描 P 半帧的运动补偿预测中使用的参考半帧的数量。例如，编码器使用单个比特来指示使用一个还是两个参考半帧。可供替换地，编码器使用用于参考半帧的数量的另一信号表示 / 编码机制。

[0275] 编码器确定 (2520) 使用一个还是两个参考半帧。如果使用一个参考半帧， 则编码器用信号表示 (2530) 隔行扫描 P 半帧的参考半帧选择。例如，编码器使用单个比特来指示使用时间上最近还是时间上第二最近的参考半帧 (先前的 I 或 P 半帧)。可供替换地，编码器使用用于 P 半帧的参考半帧选择的另一信号表示 / 编码机制。

[0276] 如果使用两个参考半帧，则编码器用信号表示 (2540) 隔行扫描 P 半帧的块、宏块或其它部分的运动矢量的参考半帧。例如，编码器对运动矢量的参考半帧选择与差分运动矢量信息一起进行联合编码。可供替换地，编码器使用用于运动矢量的参考半帧选择的另一信号表示 / 编码机制。编码器重复 (2545，2540) 下一运动矢量的信号表示，直到 P 半帧没有更多的运动矢量要用信号表示。(为了简单起见，图 25A 没有示出宏块与块编码和相应的信号表示 (2540) 的各级，它们可在参考半帧选择的信号表示 (2540) 之后或左右发生。相反，图 25A 集中在 P 半帧中多个运动矢量的参考半帧选择的重复信号表示。)

[0277] 可供替换地，编码器执行另一技术来指示使用多个参考半帧方案中的哪一个来编码隔行扫描 P 半帧。例如，编码器具有用于参考半帧的数量的更多和 / 或不同选项。

[0278] 为了简单起见，图 25A 没有示出可将该技术 (2500) 编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述了各种组合实现。

[0279] E. 解码技术

[0280] 诸如图 21 的解码器 (2100) 等解码器接收和解码指示要使用多个方案中的哪一个来解码隔行扫描 P 半帧的信号。例如，解码器执行图 25B 中所示的技术 (2550)。

[0281] 对于给定的隔行扫描 P 半帧，解码器接收和解码 (2560) 关于在隔行扫描 P 半帧的运动补偿预测中使用的参考半帧的数量的信号。例如，解码器接收和解码单个比特以指示使用一个还是两个参考半帧。可供替换地，解码器使用用于参考半帧的数量的另一解码机制。

[0282] 解码器确定 (2570) 使用一个还是两个参考半帧。如果使用一个参考半帧，则解码器接收和解码 (2580) 关于隔行扫描 P 半帧的参考半帧选择的信号。例如，解码器接收和解码单个比特以指示使用时间最近还是时间上第二最近的参考半帧 (先前的 I 或 P 半帧)。可供替换地，解码器使用用于 P 半帧的参考半帧选择的另一解码机制。

[0283] 如果使用两个参考半帧，则解码器接收和解码 (2590) 关于隔行扫描 P 半帧 的块、宏块或其它部分的运动矢量的参考半帧选择的信号。例如，解码器解码与运动矢量的差分运动矢量信息一起联合编码的参考半帧选择。可供替换地，解码器使用用于运动矢量的参考半帧选择的另一解码机制。解码器重复 (2595，2590) 下一运动矢量的接收与解码，直到没有更多的为 P 半帧用信号表示的运动矢量。(为了简单起见，图 25B 没有示出可以在参考半帧选择的接收与解码 (2590) 之后或左右发生的宏块和块解码的各级。相反，图 25B 集中在 P 半帧中的多个运动矢量的参考半帧选择的重复接收 / 解码。)

[0284] 可供替换地，解码器执行另一技术来确定使用多个参考半帧方案中的哪一个来解

码隔行扫描 P 半帧。例如，解码器具有用于参考半帧的数量的更多和 / 或不同选项。

[0285] 为了简单起见，图 25B 没有示出可将技术 (2550) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述了各种组合实现。

[0286] V. 用信号表示隔行扫描 P 半帧的宏块模式信息

[0287] 在一些实施例中，对隔行扫描 P 半帧的宏块的各种宏块模式信息进行联合分组以用于信号表示。隔行扫描 P 半帧的宏块可以用许多不同模式通过若干存在或不存在的不同句法元素中的任何元素来编码。具体地，对运动补偿的类型（例如 1MV、4MV 或帧内编码）、在宏块的比特流中是否存在已编码块模式、以及（对于 1MV 情况）宏块的比特流中是否存在运动矢量数据进行联合编码。可使用不同的代码表用于宏块模式信息的不同情况，这产生信息的更有效的整体压缩。

[0288] 在本章节和在章节 XII 的组合实现中描述的信号表示的特定示例用可变长度编码的 MBMODE 句法元素来用信号表示宏块模式信息。MBMODE 的表选择是通过定长编码的半帧级元素 MBMODETAB 来用信号表示的。可供替换地，编码器和解码器使用其它和 / 或附加信号来用信号表示宏块模式信息。

[0289] A. 不同类型的隔行扫描 P 半帧的宏块模式

[0290] 通常，宏块模式指示宏块类型 (1MV，4MV 或帧内编码）、宏块的已编码块模式的存在 / 不存在、以及宏块的运动矢量数据的存在 / 不存在。由宏块模式句法元素指示的信息取决于隔行扫描 P 半帧是被编码为 1MV 半帧（具有帧内编码和 / 或 1MV 宏块）还是混合 MV 半帧（具有帧内编码、1MV 和 / 或 4MV 宏块）。

[0291] 在 1MV 隔行扫描 P 半帧中，宏块的宏块模式元素联合地表示宏块类型（帧内 或 1MV)、宏块的已编码块模式元素的存在 / 不存在以及运动矢量数据的存在 / 不存在（当宏块类型是 1MV 时，但不是当它是帧内编码时）。图 26 中的表示出由 1MV 隔行扫描 P 半帧中的 MBMODE 用信号表示的宏块信息的完整事件空间。

[0292] 在混合 MV 隔行扫描 P 半帧中，宏块的宏块模式元素联合地表示宏块类型（帧内编码或 1MV 或 4MV)、宏块的已编码块模式的存在 / 不存在、以及运动矢量数据的存在 / 不存在（当宏块类型是 1MV 时，但不是当它是帧内编码或 4MV 时）。图 27 中的表示出由混合 MV 隔行扫描 P 半帧的 MBMODE 用信号表示的宏块信息的完整事件空间。

[0293] 如果宏块模式指示运动矢量数据存在，则运动矢量数据存在于宏块层中并且用信号表示运动矢量差分，它与运动矢量预测值组合以重构运动矢量。如果宏块模式元素指示运动矢量数据不存在，则运动矢量差分假定为零，并且因此运动矢量等于运动矢量预测值。宏块模式元素因而有效地用信号表示何时要使用只带有一个运动矢量预测值（未被任何运动矢量差分修改）的运动补偿。

[0294] 使用多个不同 VLC 表之一来用信号表示隔行扫描 P 半帧的宏块模式元素。例如，在图 47H 中示出混合 MV 隔行扫描 P 半帧的宏块的 MBMODE 的八个代码表，并且在图 47I 中示出 1MV 隔行扫描 P 半帧的 MBMODE 的八个不同代码表。表选择是由在半帧层用信号表示的 MBMODETAB 指示的。可供替换地，编码器和解码器使用其它和 / 或附加代码来用信号表示宏块模式信息和表选择。

[0295] B. 编码技术

[0296] 诸如图 20 的编码器 (2000) 等编码器编码隔行扫描 P 半帧的宏块模式信息。例如，

编码器执行图 28A 中所示的技术 (2800)。

[0297] 对于给定的隔行扫描 P 半帧，编码器选择 (2810) 用于编码隔行扫描 P 半帧的宏块的宏块模式信息的代码表。例如，编码器选择图 47H 或 47I 中所示的 VLC 表之一。可供替换地，编码器从其它和 / 或附加表中选择。

[0298] 编码器在比特流中用信号表示 (2820) 所选择的代码表。例如，编码器在给定隔行扫描 P 半帧的类型时，用信号表示指示所选择的代码表的 FLC。可供替换地，编码器使用用于代码表选择的不同信号表示机制，例如，使用 VLC 用于代码表选择。

[0299] 编码器从多个可用的宏块模式选择 (2830) 宏块的宏块模式。例如，编码器选择指示宏块类型、已编码块模式是否存在、以及 (如果可应用于宏块类型) 运动 矢量数据是否存在的宏块模式。MBMODE 的各种选项组合在图 26 与 27 中列出。可供替换地，编码器从其它和 / 或附加宏块组合选项的其它和 / 或附加宏块模式中选择。

[0300] 编码器使用所选择的代码表用信号表示 (2840) 所选择的宏块模式。一般地，编码器使用选择的 VLC 表将宏块模式用信号表示为 VLC。编码器重复 (2845，2830，2840) 宏块模式的选择和信号表示，直到 P 半帧没有更多的宏块模式要用信号表示。(为了简单起见，图 28A 没有示出可以在选择的宏块模式的信号表示 (2840) 之后或左右发生的宏块和块编码与相应信号表示的各级。相反，图 28A 集中在使用为 P 半帧选择的代码表来重复用信号表示宏块的宏块模式。)

[0301] 可供替换地，编码器执行另一技术来编码隔行扫描 P 半帧的宏块的宏块模式信息。例如，尽管图 28A 在模式选择之前示出代码表选择，但在许多常见的编码情况下，编码器首先为宏块选择宏块模式，然后选择用于有效地用信号表示那些选择的宏块模式的代码表，然后用信号表示代码表选择和这些模式。而且，尽管图 28A 示出每隔行扫描 P 半帧发生的代码表选择，但可供替换地，在更频繁、更不频繁或者非周期基础上选择代码表，或者编码器完全跳过代码表选择 (总是使用同一代码表)。或者，编码器可从上下文信息选择代码表 (使用信号表示代码表选择没有必要)。

[0302] 为了简单起见，图 28A 没有示出可将技术 (2800) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述了各种组合实现。

[0303] C. 解码技术

[0304] 诸如图 21 的解码器 (2100) 等解码器接收和解码隔行扫描 P 半帧的宏块的宏块模式信息。例如，解码器执行图 28B 所示的技术 (2850)。

[0305] 对于给定的隔行扫描 P 半帧，解码器接收和解码 (2860) 对要用于解码隔行扫描 P 半帧的宏块的宏块模式信息的代码表的代码表选择。例如，解码器在给定隔行扫描 P 半帧的类型时，接收和解码指示选择的代码表的 FLC。可供替换地，解码器与用于代码表选择的不同信号表示机制一起工作，例如，为代码表选择使用 VLC 的信号表示机制。

[0306] 解码器基于解码的代码表选择 (以及可能的其它信息) 来选择 (2870) 代码表。例如，解码器选择图 47H 或 47I 中所示的 MBMODE 的 VLC 表之一。可供替换地，解码器从其它和或附加表中选择。

[0307] 解码器接收和解码 (2880) 宏块的宏块模式选择。例如，宏块模式选择指示宏块类型、已编码块模式是否存在、以及 (如果可应用于宏块类型) 运动矢量数据是否存在。MBMODE 的这些选项的各种组合在图 26 与 27 中列出。可供替换地，宏块模式是其它和 / 或

附加宏块组合选项的其它和 / 或附加宏块模式之一。解码器重复 (2885,2880) 下一宏块的宏块模式的接收与解码,直到 P 半帧没有更多要接收与解码的宏块模式。(为了简单起见,图 28B 没有示出可以在宏块模式选择的接收与解码 (2880) 之后或左右发生的宏块与块解码的各级。相反,图 28B 集中在使用为 P 半帧选择的代码表来重复接收 / 解码 P 半帧中宏块的宏块模式选择。)

[0308] 可供替换地,解码器执行另一技术来解码隔行扫描 P 半帧的宏块的宏块模式信息。例如,尽管图 28B 示出每隔行扫描 P 半帧代码表选择发生,但可供替换地,在更频繁、更不频繁或非周期基础上选择代码表,或者解码器完全跳过代码表选择 (总是使用同一代码表)。或者,解码器可从上下文信息选择代码表 (使代码表选择的接收与解码没有必要)。

[0309] 为了简单起见,图 28B 没有示出可将技术 (2850) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述了各种组合实现。

[0310] VI. 在两个参考半帧隔行扫描 P 半帧中的参考半帧选择

[0311] 在一些实施例中,当为单个当前隔行扫描 P 半帧执行运动补偿预测时,使用两个先前编码 / 解码的半帧作为参考半帧。(例如,见章节 IV。) 用信号表示的信息指示两个半帧中的哪一个为具有运动矢量的每个宏块 (或块) 提供参考。

[0312] 在本节中,描述了各种技术与工具,它们用于有效地用信号表示在编码或解码当前宏块或块时多个先前编码 / 解码的参考半帧中的哪一些用于提供运动补偿预测信息。例如,编码器和解码器隐含地基于隔行扫描 P 半帧中的先前编码的运动矢量来导出当前宏块或块的主和非主参考半帧。(或者,相应地,编码器和解码器导出主和非主运动矢量预测值极性。) 用信号表示的信息随后指示对当前宏块或块的运动补偿使用主或非主参考半帧。

[0313] A. 主和非主参考半帧和预测值

[0314] 隔行扫描半帧可以不使用运动补偿 (I 半帧)、前向运动补偿 (P 半帧)、或者前向与后向运动补偿 (B 半帧) 来编码。隔行扫描 P 半帧可参考两个参考半帧,它们是先前编码 / 解码的 I 或 P 半帧。图 24A 和 24B 示出示例,其中对当前 P 半帧 使用两个参考半帧。这两个参考半帧是相反极性的。一个参考半帧表示视频帧的奇数行,而另一个参考半帧表示视频帧 (没必要是包括奇数行参考半帧的同一帧) 的偶数行。当前正在编码或解码的 P 半帧可以在运动补偿中使用两个先前编码 / 解码的半帧之一或两者作为参考。因而,P 半帧的宏块或块的运动矢量数据以某种方式指示 :(1) 在运动补偿中要使用哪一个半帧作为参考半帧 ;以及 (2) 在运动补偿中要使用的样本值的参考半帧所在的位移 / 位置。

[0315] 用信号表示参考半帧选择信息会消耗低效率的比特数。然而对于给定的运动矢量,通过预测将使用哪一个参考半帧用于运动矢量并且随后用信号表示实际上是否使用预测参考半帧作为运动矢量的参考半帧,可减少比特数。

[0316] 例如,对于在隔行扫描 P- 半帧中使用运动补偿的每个宏块或块,编码器或解码器分析最多三个来自相邻宏块或块的先前编码 / 解码的运动矢量。从它们中,编码器或解码器导出主和非主参考半帧。实际上,编码器或解码器确定两个可能的参考半帧中的哪一个是由相邻宏块或块的运动矢量的大多数使用的。由相邻宏块或块中较多的运动矢量参考的半帧是主参考半帧,并且另一个参考半帧是非主参考半帧。同样,主参考半帧的极性是主运动矢量预测值极性,而非主参考半帧的极性是非主运动矢量预测值极性。

[0317] 图 29 中的伪代码示出编码器或解码器确定主和非主参考半帧的一种技术。在该

伪代码中，术语“相同半帧”和“相反半帧”相对于当前隔行扫描 P 半帧。例如，如果当前 P 半帧是偶数半帧，则“相同半帧”是偶数行参考半帧，并且“相反半帧”是奇数行参考半帧。图 5A 至 10 示出从其中取得预测值 A、B 和 C 的相邻宏块和块的位置。在图 29 的伪代码中，主半帧是由候选运动矢量预测值中的大多数参考的半帧。在平分的情况下，从相反半帧导出的运动矢量视为主运动矢量预测值。帧内编码的宏块在主 / 非主预测值的计算中不考虑。如果所有候选预测值宏块是帧内编码的，则主和非主运动矢量预测值设置为零，并且主预测值取自相反半帧。

[0318] 可供替换地，编码器和解码器分析来自相邻宏块或块的其它和 / 或附加运动矢量，和 / 或应用不同的决策逻辑以确定主和非主参考半帧。或者，编码器和解码器使用不同的机制来预测将为隔行扫描 P 半帧中的给定运动矢量选择哪一个参考半帧。

[0319] 在某些情况下，指示使用主还是非主半帧的 1 比特信息与差分运动矢量信息一起进行联合编码。因此，用于该 1 比特信息的比特 / 码元可以更准确地匹配真正 的码元熵。例如，用信号表示主 / 非主选择器，作为图 30 的伪代码中所示的运动矢量差分的垂直分量的一部分。其中，MVY 是运动矢量的垂直分量，PMVY 是运动矢量预测值的垂直分量。实际上，垂直运动矢量差分对参考半帧选择器和垂直偏移差分进行联合编码，如下：

[0320] DMVY = (MVY-PMVY)*2+p，其中，如果使用主参考半帧则 p = 0，而如果使用非主参考半帧则 p = 1。作为一个数字示例：假定当前块是偶极性，则运动矢量的实际参考半帧是偶极性，并且主预测值是相反半帧（换言之，主参考半帧是奇极性参考半帧）。还假定运动矢量的垂直位移是 7 单位 (MVY = 7) 并且运动矢量预测值的垂直分量是 4 单位 (PMVY = 4)。由于当前参考半帧和主预测值是相反极性的，DMVY = (7-4)*2+1 = 7。

[0321] 可供替换地，主 / 非主选择器以某种其它方式与运动矢量差分信息一起联合编码。或者，用另一机制用信号表示主 / 非主选择器。

[0322] B. 编码技术

[0323] 诸如图 20 的编码器 (2000) 等编码器在两个参考半帧隔行扫描 P 半帧的运动矢量的编码期间，确定候选运动矢量预测值的主和非主参考半帧。例如，编码器对当前宏块或块的运动矢量执行图 31A 所示的技术 (3100)。一般地，编码器在两个参考半帧中执行某种形式的运动估计以获得运动矢量和参考半帧。运动矢量随后按照技术 (3100) 编码。

[0324] 编码器确定 (3110) 与运动矢量相同参考半帧极性的运动矢量预测值。例如，编码器确定在章节 VII 中描述运动矢量预测值用于与运动矢量相关联的参考半帧。可供替换地，编码器用另一种机制来确定运动矢量预测值。

[0325] 编码器确定 (3120) 运动矢量的主和非主参考半帧极性。例如，编码器遵循图 29 所示的伪代码。可供替换地，编码器使用另一种技术来确定主和非主极性。

[0326] 编码器用信号表示 (3125) 比特流中的主 / 非主极性选择器，它指示应该对与运动矢量相关联的运动矢量预测值和参考半帧使用主还是非主极性。例如，编码器使用联合 VLC 对主 / 非主极性选择器与其它信息一起进行联合编码。可供替换地，编码器使用另一种机制来用信号表示选择器，例如，指示选择器的一个比特的算术编码。运动矢量预测值的参考半帧极性的预测降低了选择器信息的熵，这允许更有效地编码选择器信息。

[0327] 编码器根据运动矢量预测值和运动矢量计算 (3130) 运动矢量差分，并且用信号表示 (3140) 运动矢量差分信息的信息。

[0328] 可供替换地，编码器执行另一种机制在两个参考半帧隔行扫描 P 半帧的运动矢量的编码期间确定运动矢量预测的主和非主极性。而且，尽管图 31A 示出主和非主选择器和运动矢量差分信息的单独信号表示，但在各种实施例中，这个精确的信息是联合地用信号表示的。各种其它重新排序是可能的，包括在确定主 / 非主极性之后确定运动矢量（以便在运动矢量选择过程中将选择器信号表示的开销成本考虑进去）。

[0329] 为了简单起见，图 31A 没有示出可将技术 (3100) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述了各种组合实现。

[0330] C. 解码技术

[0331] 诸如图 21 的解码器 (2100) 等解码器在两个参考半帧隔行扫描 P 半帧的运动矢量的解码期间确定运动矢量预测值候选的主和非主参考半帧极性。例如，解码器执行图 31B 所示的技术 (3150)。

[0332] 解码器确定 (3160) 当前宏块或块的运动矢量的主和非主参考半帧极性。例如，解码器遵循图 29 所示的伪代码。可供替换地，解码器使用另一种技术来确定主和非主极性。

[0333] 解码器接收和解码 (3165) 比特流中的主 / 非主极性选择器，它指示应该对与运动矢量相关联的运动矢量预测值和参考半帧使用主还是非主极性。例如，解码器接收和解码已经使用联合 VLC 与其它信息一起联合编码的主 / 非主极性选择器。可供替换地，解码器接收和解码使用另一种机制用信号表示的选择器，例如，指示选择器的一个比特的算术解码。

[0334] 解码器确定 (3170) 要与运动矢量一起使用的参考半帧的运动矢量预测值。例如，解码器确定在章节 VII 中描述的运动矢量预测值用于用信号表示的极性。可供替换地，解码器用另一种机制来确定运动矢量预测值。

[0335] 解码器接收和解码 (3180) 运动矢量差分的信息，并且从运动矢量差分和运动矢量预测值中重构 (3190) 运动矢量。

[0336] 可供替换地，解码器执行另一种技术在两个参考半帧隔行扫描 P 半帧的运动矢量的解码期间确定运动矢量预测值的主和非主极性。例如，尽管图 31B 示出主 / 非主选择器和运动矢量差分信息的单独信号表示，但可供替换地，该信息是联合地用信号表示的。各种其它重新排序也是可能的。

[0337] 为了简单起见，图 31B 没有示出可将技术 (3150) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述了各种组合实现。

[0338] VII. 隔行扫描 P 半帧的混合运动矢量预测

[0339] 在一些实施例中，将运动矢量用信号表示为相对于运动矢量预测值的差分，以便减少与用信号表示运动矢量相关联的比特率。运动矢量差分信号表示的性能部分地取决于运动矢量预测的质量，这通常在从当前宏块、块等周围区域考虑多个候选运动矢量预测值时得到改进。然而在有些情况下，多个候选预测值的使用损害了运动矢量预测的质量。例如，这是在将运动矢量预测值计算为一组相异的候选预测值的中值时发生的（例如，在运动矢量预测值之间具有很大的变化）。

[0340] 因此，在一些实施例中，编码器和解码器对隔行扫描 P 半帧的运动矢量执行混合运动矢量预测。当构成当前宏块或块的因果相邻帧的矢量按照某些准则是相异的时候，使用混合运动矢量预测模式。在这个模式中，并非使用候选预测值组的中值作为运动矢量预测值，而是由一个选择器比特或码字来用信号表示来自该组的特定运动矢量（例如，上预

测值，左预测值）。这有助于改进在隔行扫描 P 半帧中运动不连续处的运动矢量预测。对于 2 参考半帧隔行扫描 P 半帧，在检查混合运动矢量预测条件时也将主极性考虑在内。

[0341] A. 隔行扫描 P- 半帧的运动矢量预测

[0342] 混合运动矢量预测是隔行扫描 P 半帧的正常运动矢量预测的特殊情况。如前面说明的，运动矢量是通过将运动矢量差分（它是在比特流中用信号表示的）加到运动矢量预测值来重构的。预测值是从最多三个相邻运动矢量计算的。图 5A 至 10 示出从其取得运动矢量预测的预测值 A、B 和 C 的相邻宏块和块的位置。（这些图示出逐行扫描 P 半帧的宏块和块，但也应用于隔行扫描 P 半帧的宏块和块，如在章节 VI 中所述。）

[0343] 如果隔行扫描 P 半帧只参考一个先前半帧，则为 P 半帧的每个运动矢量计算单个运动矢量预测值。例如，图 51A 和 51B（或者可供替换地，图 60A 和 60B）示出如何为 1 参考半帧隔行扫描 P 半帧的运动矢量计算运动矢量预测值，如在章节 XII 中讨论的。

[0344] 如果对隔行扫描 P 半帧使用两个参考半帧，则两个运动矢量预测值有可能用于 P 半帧的每个运动矢量。可计算两个运动矢量预测值，然后选择一个，或者可 通过首先确定预测值选择，只计算一个运动矢量预测值。例如，一个可能的运动矢量预测值来自主参考半帧，而另一个可能的运动矢量预测值来自非主参考半帧，其中术语主和非主如在章节 VI 中描述的那样。主和非主参考半帧具有相反极性，因此一个运动矢量预测值来自与当前 P 半帧相同极性的参考半帧，而另一个运动矢量预测值来自具有相反极性的参考半帧。例如，图 52A 至 52N 中的伪代码和表示出了为 2 参考半帧 P 半帧的运动矢量计算运动矢量预测值的过程，如在章节 XII 中详细讨论的。变量 samefieldpred_x 和 samefieldpred_y 分别表示来自相同半帧的运动矢量预测值的水平和垂直分量，而变量 oppositefieldpred_x 和 oppositefieldpred_y 分别表示来自相反半帧的运动矢量预测值的水平和垂直分量。变量 dominantpredictor 指示哪一个半帧包含主预测值。predictor_flag 指示使用对运动矢量主还是非主预测值。可供替换地，使用图 61A 至 61F 中的伪代码。

[0345] B. 隔行扫描 P 半帧的混合运动矢量预测

[0346] 对于运动矢量的混合运动矢量预测，编码器和解码器检查运动矢量的混合运动矢量预测条件。通常，条件与运动矢量预测值中的变化程度有关。估计的预测值可以是候选运动矢量预测值和 / 或使用正常的运动矢量预测计算的运动矢量预测值。如果满足条件（例如，变化程度高），则一般使用原始候选运动矢量预测值之一代替正常的运动矢量预测值。编码器用信号表示要使用哪一个混合运动矢量预测值，而解码器接收和解码该信号。当预测值之间的变化很小时（这是通常的情况），不使用混合运动矢量预测值。

[0347] 编码器和解码器检查隔行扫描 P 半帧的每个运动矢量的混合运动矢量条件、运动矢量是否用于宏块、块等。换言之，编码器和解码器为每个运动矢量确定条件是否被触发并且因而预期一预测值选择信号。可供替换地，编码器和解码器只为隔行扫描 P 半帧的一些运动矢量检查混合运动矢量条件。

[0348] 隔行扫描 P 半帧的混合运动矢量预测的优点是，它使用计算出的预测值和主极性来选择良好的运动矢量预测值。大量实验结果表明混合运动矢量如下所述地提供显著的压缩 / 质量改进，这超过了没有它的运动矢量预测，并且也超过了混合运动矢量预测的早期实现。而且，混合矢量预测检查的附加计算代价不是很大。

[0349] 在一些实施例中，编码器或解码器对照原始候选运动矢量预测值集来测试正常的

运动矢量预测值（如由在章节 VII.A 中描述的技术所确定的）。正常的运动矢量预测值是预测值 A、B 和 / 或 C 的分量级中值，并且编码器和解码器相对于预测 值 A 和预测值 C 来测试它。该测试检查正常运动矢量预测值与候选之间的差异是否高。如果高，则真正的运动矢量有可能更接近这些候选预测值 (A，B 或 C) 之一，而不是更接近从中值运算导出的预测值。当候选预测值分开很远时，它们的分量级中值不提供良好的预测，并且发送一个指示真正的运动矢量更接近 A 还是 C 的附加信号更有效。如果预测值 A 是更接近的预测值，则使用它作为当前运动矢量的运动矢量预测值，并且如果预测值 C 是更接近的预测值，则使用它作为当前运动矢量的运动矢量预测值。

[0350] 图 32 中的伪代码示出在解码期间这样的混合运动矢量预测。变量 predictor_pre_x 和 predictor_pre_y 分别是水平和垂直运动矢量预测值，如使用正常的混合运动矢量预测来计算的。变量 predictor_psst_x 和 predictor_post_y 分别是在混合运动矢量预测之后水平和垂直运动矢量预测值。在伪代码中，相对于预测值 A 和 C 测试正常的运动矢量预测值，以查看运动矢量预测值选择是否明确地编码在比特流中。如果是，则在比特流中存在指示使用预测值 A 还是预测值 C 作为运动矢量预测值的单个比特。否则，使用正常的运动矢量预测值。也可检查各种其它条件（例如，如果 A 或 C 是帧内编码的，则检查正常运动矢量的量值）。当 A 或 C 是帧内编码时，分别对应于 A 或 C 的运动被视为零。

[0351] 对于两个参考半帧 P 半帧的运动矢量，所有预测值是相同极性的。在一些实施例中，由在差分运动矢量解码过程中获得的主 / 非主预测值极性和选择器信号来确定参考半帧极性。例如，如果使用相反半帧预测值，则 :predictor_pre_x = oppositefieldpred_x，predictor_pre_x = oppositefieldpred_y，predictorA_x = oppositefieldpredA_x，predictorA_y = oppositefieldpredA_y，predictorC_x = oppositefieldpredC_x，以及 predictorC_y = oppositefieldpredC_y。如果使用相同半帧极性则 :predictor_pre_x = samefieldpred_x，predictoryre_x = samefieldpred_y，predictorA_x = samefieldpredA_x，predictorA_y = samefieldpredA_y，predictorC_x = samefieldpredC_x，以及 predictorC_y = samefieldpredC_y。例如，oppositefieldpred 和 samefieldpred 的值如在图 52A 至 52J 或者 61A 至 61F 的伪代码中那样计算。图 53 示出在组合实现（见章节 XII）中的混合运动矢量预测的替换伪代码。

[0352] 可供替换地，编码器和解码器测试不同的混合运动矢量预测条件，例如，考虑其它和 / 或附加预测值的条件，使用不同的决策逻辑来检测运动不连续性的条件，和 / 或对变化使用不同的阈值（除了 32 以外）的条件。

[0353] 在两个候选预测值（例如 A 与 C) 之间选择的简单信号是每运动矢量单个比 特。可供替换地，编码器和解码器使用不同的信号表示机制，例如，与诸如运动矢量数据等其它信息一起联合地用信号表示选择器比特。

[0354] C. 编码技术

[0355] 诸如图 20 的编码器 (2000) 等编码器在隔行扫描 P 半帧的运动矢量的编码期间执行混合运动矢量预测。例如，编码器对当前宏块或块的运动矢量执行图 33A 所示的技术 (3300)。

[0356] 编码器确定 (3310) 运动矢量的运动矢量预测值。例如，编码器使用章节 VII.A 中描述的技术来确定运动矢量预测值。可供替换地，编码器用另一种技术来确定运动矢量预

测值。

[0357] 编码器随后检查 (3320) 运动矢量预测值的混合运动矢量预测条件。例如，编码器使用反映图 32 中所示的解码器侧的伪代码的技术。可供替换地，编码器检查不同的混合运动矢量预测条件。(相应的解码器象编码器一样检查混合运动矢量预测条件，而无论条件是什么，因为预测值信号信息的存在 / 不存在是隐含地由编码器和相应的解码器导出的。)

[0358] 如果没有触发混合运动矢量条件 (从判定 3325 出来的“否”路径)，编码器使用最初确定的运动矢量预测值。

[0359] 另一方面，如果触发混合运动矢量条件 (判定 3325 出来的“是”路径)，则编码器选择 (3330) 要使用的混合运动矢量预测值。例如，编码器在上候选预测值与左候选预测值之间选择，它们是相邻的运动矢量。可供替换地，编码器在其它和 / 或附加预测值之间选择。

[0360] 编码器随后用信号表示 (3340) 所选择的混合运动矢量预测值。例如，编码器传输单个比特，它指示要使用上候选预测值还是左候选预测值作为运动矢量预测值。可供替换地，编码器使用另一个信号表示机制。

[0361] 编码器对隔行扫描 P 半帧的每个运动矢量，或者只对隔行扫描 P 半帧的某些运动矢量 (例如，取决于宏块类型) 执行技术 (3300)。为了简单起见，图 33A 没有显示可将技术 (3300) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述了各种组合实现。

[0362] D. 解码技术

[0363] 诸如图 21 的解码器 (2100) 等解码器在隔行扫描 P 半帧的运动矢量的解码期 间执行混合运动矢量预测。例如，解码器对当前宏块或块的运动矢量执行图 33B 所示的技术 (3350)。

[0364] 解码器确定 (3360) 运动矢量的运动矢量预测值。例如，解码器使用章节 VII.A 中描述的技术来确定运动矢量预测值。可供替换地，解码器用另一种技术来确定运动矢量预测值。

[0365] 解码器随后检查 (3370) 运动矢量预测值的混合运动矢量预测条件。例如，解码器遵循图 32 所示的伪代码。可供替换地，解码器检查不同的混合运动矢量预测条件。(解码器检查与相应的编码器相同的运动矢量预测条件，而无论条件是什么。)

[0366] 如果没有触发混合运动矢量条件 (判定 3375 出来的“否”路径)，则解码器使用最初确定的运动矢量预测值。

[0367] 另一方面，如果触发混合运动矢量条件 (判定 3375 出来的“是”路径)，则解码器接收和解码 (3380) 指示所选择的混合运动矢量预测值的信号。例如，解码器取得单个比特，它指示要使用上候选预测值还是左候选预测值作为运动矢量预测值。可供替换地，解码器结合另一种信号表示机制操作。

[0368] 解码器随后选择要使用的混合运动矢量预测值。例如，解码器在上候选预测值与左候选预测值之间选择，它们是相邻的运动矢量。可供替换地，解码器在其它和 / 或附加预测值之间选择。

[0369] 解码器对隔行扫描 P 半帧的每个运动矢量，或者只对隔行扫描 P 半帧的某些运动矢量 (例如，取决于宏块类型) 执行技术 (3350)。为了简单起见，图 33B 没有示出可将技

术 (3350) 与编码和解码器的其它方面集成的各种方法。在章节 XII 中详细地描述各种组合实现。

[0370] VII. 运动矢量块模式

[0371] 在一些实施例中，宏块可具有多个运动矢量。例如，混合 MV 隔行扫描 P 半帧的宏块可具有一个运动矢量、四个运动矢量（宏块的每个亮度块一个）或者是帧内编码的（无运动矢量）。同样，隔行扫描 P 半帧的半帧编码宏块可具有两个运动矢量（每半帧一个）或者四个运动矢量（每半帧两个），并且隔行扫描 P 半帧的帧编码宏块可具有一个运动矢量或四个运动矢量（每亮度块一个）。

[0372] 如果宏块没有相关联的运动矢量数据（例如差分）要用信号表示，2MV 或 4MV 宏块可用信号表示为“跳过”。如果这样，则运动矢量预测值一般用作宏块的运动 矢量。或者，宏块可具有非零的运动矢量数据要为一个运动矢量用信号表示，但不是为另一个运动矢量（它具有 (0,0) 运动矢量差分）。对于至少一个但非全部运动矢量具有 (0,0) 差分的 2MV 或 4MV 宏块，用信号表示运动矢量数据会消耗低效率的比特数。

[0373] 因此，在一些实施例中，编码器和解码器使用一种信号表示机制，它有效地用信号表示具有多个运动矢量的宏块的运动矢量数据存在或不存在。宏块的运动矢量已编码块模式（或者简称“运动矢量块模式”）在逐个运动矢量的基础上指示，哪些块、半帧、二分之一半帧等具有在比特流中用信号表示的运动矢量数据，以及哪些没有。运动矢量块模式联合地用信号表示宏块的运动矢量数据的模式，它允许编码器和解码器利用块之间存在的空间相关性。而且，用运动矢量块模式用信号表示运动矢量数据的存在 / 不存在提供一个简单的方法，以一种与用信号表示有关变换系数数据的存在 / 不存在（诸如通过 CBCPY 元素）分离的方式用信号表示该信息。

[0374] 在本章节和在章节 XII 的组合实现中描述的信号表示的特定示例用可变长度编码的 2MVBP 和 4MVBP 句法元素用信号表示运动矢量块模式。2MVBP 和 4MVBP 的表选择分别通过定长编码的 2MVBPTAB 和 4MVBPTAB 来用信号表示。可供替换地，编码器和解码器使用其它和 / 或附加信号来用信号表示运动矢量块模式。

[0375] A. 运动矢量块模式

[0376] 运动矢量块模式指示对具有多个运动矢量的宏块，哪些运动矢量被“编码”和哪些没有被“编码”。如果运动矢量的差分运动矢量非零（即，要用信号表示的运动矢量不同于它的运动矢量预测值），则编码该运动矢量。否则，不编码该运动矢量。

[0377] 如果宏块具有四个运动矢量，则运动矢量块模式具有 4 个比特，每一个比特分别用于四个运动矢量的每一个。在运动矢量块模式中的比特的顺序遵循图 34 中为隔行扫描 P 半帧的 4MV 宏块或者隔行扫描帧的 4MV 帧编码宏块所示的块顺序。对于隔行扫描帧的 4MV 半帧编码宏块，运动矢量块模式的比特顺序是左上半帧运动矢量、右上半帧运动矢量、左下半帧运动矢量和右下半帧运动矢量。

[0378] 如果宏块具有两个运动矢量，则运动矢量块模式有 2 个比特，每个比特用于两个运动矢量的每一个。对于隔行扫描 P 帧的 2MV 半帧编码宏块，运动矢量块模 式的比特顺序简单地为上半帧运动矢量然后下半帧运动矢量。

[0379] 可使用多个不同 VLC 表之一来用信号表示运动矢量块模式元素。例如，在图 47J 中示出 4MVBP 的四个不同代码表，并且在图 47K 中示出 2MVBP 的四个不同的代码表。表选

择由在图像层用信号表示的 4MVBPTAB 或 2MVBPTAB 指示。可供替换地，编码器和解码器使用其它和 / 或附加代码用于用信号表示运动矢量块模式信息和表选择。

[0380] 应用附加的规则，以确定为 2 参考半帧隔行扫描 P 半帧的宏块编码哪些运动矢量。“未编码”的运动矢量具有主预测值，如在章节 VI 中所述。“编码”的运动矢量可具有零值运动矢量差分，但用信号表示非主预测值。或者，“编码”的运动矢量可具有非零差分运动矢量并且用信号表示主或非主预测值。

[0381] 可供替换地，编码器和解码器使用运动矢量块模式用于其它和 / 或附加种类的图像、用于其它和 / 或附加种类的宏块、用于其它和 / 或附加数量的运动矢量，和 / 或具有不同的比特位置。

[0382] B. 编码技术

[0383] 诸如图 20 的编码器 (2000) 等编码器使用运动矢量块模式编码宏块的运动矢量数据。例如，编码器执行图 35A 所示的技术 (3500)。

[0384] 对于给定的具有多个运动矢量的宏块，编码器确定 (3510) 宏块的运动矢量块模式。例如，编码器为隔行扫描 P 半帧中 4MV 宏块或者为隔行扫描 P 帧中 4MV 半帧编码或帧编码宏块确定四个块模式。或者，编码器为隔行扫描 P 帧中 2MV 半帧编码宏块确定两个运动矢量块模式。可供替换地，编码器为其它种类的宏块和 / 或其它数量的运动矢量确定运动矢量块模式。

[0385] 然后编码器用信号表示 (3520) 运动矢量块模式。通常，编码器使用诸如图 47J 和 47K 所示的代码表用信号表示运动矢量块模式的 VLC。可供替换地，编码器使用另一种机制来用信号表示运动矢量块模式。

[0386] 如果有要为其用信号表示运动矢量数据的至少一个运动矢量（判定 3525 出来的“是”路径），则编码器用信号表示 (3530) 运动矢量的运动矢量数据。例如，编码器使用章节 IX 中描述的技术将运动矢量数据编码为 BLKMVDATA、TOPMVDATA 或 BOTMVDATA 元素。可供替换地，编码器使用不同的信号表示技术。

[0387] 编码器重复 (3525，3530) 运动矢量数据的编码，直到没有要为其用信号表示运动矢量数据的更多运动矢量（判定 3525 出来的“否”路径）。

[0388] 编码器可在多个代码表之间选择以编码运动矢量块模式（未在图 35A 中示出）。例如，编码器为隔行扫描 P 半帧或 P 帧选择一个代码表，然后使用该表来编码图像中宏块的运动矢量块模式。可供替换地，编码器在更频繁、较不频繁或非周期的基础上选择代码表，或者编码器完全跳过代码表选择（总是使用同一代码表）。或者，编码器可从上下文信息中选择代码表（使得用信号表示代码表选择没有必要）。代码表可以是图 47J 和 47K 中所示的表、其它表和 / 或附加表。例如，编码器用指示所选择的代码表的 FLC，用指示所选择的代码表的 VLC，或者用一种不同的信号表示机制，在比特流中用信号表示所选择的代码表。

[0389] 可供替换地，编码器执行另一种技术来使用运动矢量块模式编码宏块的运动矢量数据。为了简单，图 35A 没有示出可将技术 (3500) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述各种组合实现。

[0390] C. 解码技术

[0391] 诸如图 21 的解码器 (2100) 等解码器接收和解码使用运动矢量块模式的隔行扫描 P 半帧或隔行扫描 P 帧的运动矢量数据。例如，解码器执行图 35B 所示的技术 (3550)。

[0392] 对于具有多个运动矢量的给定宏块，解码器接收和解码 (3560) 宏块的运动矢量块模式。例如，解码器接收和解码 4 运动矢量块模式，或者在前面章节中描述的其它运动矢量块模式。通常，解码器接收运动矢量块模式的 VLC 并且使用图 47J 和 47I 所示的代码表来对它进行解码。可供替换地，解码器结合另一种信号表示机制接收和解码运动矢量块模式。

[0393] 如果有要为其用信号表示运动矢量数据（判定 3565 出来的“是”路径）的至少一个运动矢量，则解码器接收和解码 (3570) 该运动矢量的运动矢量数据。例如，解码器接收和解码使用在章节 IX 中描述的技术编码为 BLTMVDATA、TOPMVDATA 或 BOTMVDATA 元素的运动矢量数据。可供替换地，解码器使用不同的解码技术。

[0394] 解码器重复 (3565，3570) 运动矢量数据的接收和解码，直到没有要为其用信号表示运动矢量数据的更多运动矢量（判定 3565 出来的“否”路径）。

[0395] 解码器可在多个代码表之间选择以解码运动矢量块模式（未在图 3513 中示出）。例如，表选择和表选择信号表示选项反映出在前面章节中为编码器描述的那些技术。

[0396] 可供替换地，解码器执行另一种技术来使用运动矢量块模式解码宏块的运动矢量数据。为了简单起见，图 35B 没有示出可将技术 (3550) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述各种组合实现。

[0397] IX. 隔行扫描 P- 半帧中的运动矢量差分

[0398] 在一些实施例中，当为单个当前隔行扫描 P 半帧执行运动补偿预测时，使用两个先前编码 / 解码的半帧作为参考半帧。（例如，见章节 IV，VI 和 VII。）在 P 半帧中为运动矢量用信号表示的信息指示 :(1) 两个半帧中的哪一个为运动矢量提供参考 ;以及 (2) 运动矢量值。运动矢量值一般用信号表示为相对于运动矢量预测值的差分。在两个可能的参考半帧之间的选择可用单个附加比特来为运动矢量用信号表示，但在许多情况下这种信号表示方式是低效率的。通常，两个参考半帧对于一个给定的运动矢量不是相等可能的，并且运动矢量的选择并不是与其它（例如相邻）运动矢量的选择无关的。因而，实际上，用每选择单个比特来用信号表示参考半帧选择通常是低效率的。

[0399] 因此，在一些实施例中，编码器对运动矢量差分信息和参考半帧选择信息进行联合编码。解码器执行联合编码信息的相应解码。

[0400] A. 理论与实验结果

[0401] 对于 2 参考半帧隔行扫描 P 半帧，两个参考半帧具有与 P- 半帧的下列空间和时间关系。在时间上最接近的参考半帧的极性与当前 P 半帧的极性相反。例如，如果当前 P 半帧是偶半帧（由隔行扫描帧的偶数行构成），则在时间顺序上最接近的参考半帧是奇半帧，而另一参考半帧（在时间顺序上较远的）是偶半帧。

[0402] 编码器和解码器使用因果信息来预测当前运动矢量的参考半帧选择。例如，来自相邻的先前编码的运动矢量的参考半帧选择信息用于预测用于当前运动矢量的参考半帧。然后，一二进制值指示是否使用该预测的参考半帧。一个值指示当前运动矢量的实际参考半帧是预测的参考半帧，而另一个值指示当前运动矢量的实际参考半帧是另一参考半帧。在一些实现中，参考半帧预测是按照先前使用的参考半帧和当前运动矢量的预期参考半帧的极性来表示的（例如，如主或非主极性，见章节 VI）。在大多数情况下，通过这样的预测，二进制值参考半帧选择器的概率分布是一致的，并且朝向预测的参考半帧偏斜。在实验中，

预测的参考半帧用于大约 70％的运动矢量，而大约 30％的运动矢量使用另一参考半帧。

[0403] 传输单个比特以用信号表示具有这样一种概率分布的参考半帧选择信息是低效率的。更有效的方法是将参考半帧选择信息与差分运动矢量信息一起联合编码。

[0404] B. 信号表示机制的示例

[0405] 提供了用于运动矢量差分信息和参考半帧选择信息进行联合编码和解码的信号表示机制的各种示例。可供替换地，编码器和解码器结合另一种机制来对信息进行联合编码和解码。

[0406] 图 36 中的伪代码示出按照一般的信号表示机制对运动矢量差分信息和参考半帧选择信息进行联合编码。在该伪代码中，变量 DMVX 和 DMVY 分别是水平和垂直差分运动矢量分量。变量 AX 和 AY 是差分分量的绝对值，而变量 SX 和 SY 是差分分量的符号。水平运动矢量范围从 -RX 到 RX+1，而垂直运动矢量范围从 -RY 到 RY+1。RX 和 RY 是二的幂，分别具有指数 MX 和 MY。变量 ESCX 和 ESCY（它们分别是指数为 KX 和 KY 的二的幂）指示了阈值，超过该阈值则使用转义码。变量 R 是参考半帧选择的二进制值。

[0407] 当触发转义条件时 (AX ＞ ESCX 或 AY ＞ ESCY)，编码器发送联合地表示转义模式信号和 R 的 VLC。编码器随后发送 DMVX 和 DMVY，分别作为长度 MX+1 和 MY+1 的定长代码。因而，使用 VLC 表中的两个元素来用信号表示 (1) 合在一起使用 (MX+MY+2) 个比特来编码 DMVX 和 DMVY，以及 (2) 相关联的 R 值。换言之，两个元素是对应于 R ＝ 0 和 R ＝ 1 的转义码。

[0408] 对于其它事件，变更 NX 和 NY 指示要使用多少比特来分别用信号表示 AX 和 AY 的不同值。AX 在时间间隔 (2NX ＜＝ AX ＜ 2NX+1) 中，其中 NX ＝ 0，1，2，…KX-1，而当 NX ＝ -1 时 AX ＝ 0。AXY 在时间间隔 (2NY ＜＝ AY ＜ 2NY+1) 中，其中 NY ＝ 0，1，2，…KY-1，而在 NY ＝ -1 时 AY ＝ 0。

[0409] 用于编码大小信息 NX 和 NY 以及半帧参考信息 R 的 VLC 表是具有 (KX+1)*(KY+1)*2+1 个元素的表，其中每个元素是一个（码字，代码大小）对。在表的元素中，除了两个之外全部用于联合地用信号表示 NX、NY 和 R 的值。这其它两个元素是转义码。

[0410] 对于与 NX 和 NY 一起用信号表示的事件，编码器发送 VLC 来指示 NX、NY 和 R 值的组合。编码器随后发送 AX 为 NX 个比特，发送 SX 为一个比特，发送 AY 为 NY 个比特，并且发送 SY 为一个比特。如果 NX 为 0 或 -1，则 AX 不需要 发送，并且对于 NY 和 AY 也是如此，因为 AX 或 AY 的值在这些情况下可直接从 NX 或 NY 中导出。

[0411] 在 AX ＝ 0、AY ＝ 0 和 R ＝ 0 的事件是由另一种机制用信号表示的，诸如跳过宏块机制或运动矢量块模式（见章节 VIII）。图 36 中伪代码的 VLC 表不存在 [0,0,0] 元素，或者在该伪代码中不处理。

[0412] 相应的解码器执行联合解码，它反映出图 36 所示的编码。例如，解码接收比特代替发送比特，执行可变长度解码代替可变长度编码等等。

[0413] 图 50 中的伪代码示出运动矢量差分信息和参考半帧选择信息的解码，它们已经按照一个组合实现中的信号表示机制联合编码。图 59 中的伪代码示出运动矢量差分信息和参考半帧选择信息的解码，它们已经按照另一个组合实现的信号表示机制联合编码。图 50 和 59 中的伪代码在章节 XII 中详细地说明。具体地，该伪代码示出具有垂直差分值或者具有垂直和水平差分值的大小的预测选择器的联合编码与解码。

[0414] 相应的编码器执行联合编码，它反映出图 50 或 59 中所示的解码。例如，编码器发送比特代替接收比特，执行可变长度编码代替可变长度解码等等。

[0415] C. 编码技术

[0416] 诸如图 20 的编码器 (2000) 等编码器对参考半帧预测选择器信息和差分运动矢量信息进行联合编码。例如，编码器执行图 37A 所示的技术 (3700) 来对该信息进行联合编码。通常，编码器在两个参考半帧中执行某种形式的运动估计以获得运动矢量和参考半帧。运动矢量随后按照技术 (3700) 编码，在该点处通过对选择器信息进行联合编码，例如与垂直运动矢量差分一起联合编码，将两个可能的参考半帧之一与运动矢量相关联。

[0417] 编码器确定 (3710) 运动矢量的运动矢量预测值。例如，编码器如章节 VII 中所述地确定运动矢量预测值。可供替换地，编码器用另一种机制确定运动矢量预测值。

[0418] 编码器为运动矢量确定 (3720) 相对于运动矢量预测值的运动矢量差分。通常，差分是运动矢量与运动矢量预测值之间的分量级差。

[0419] 编码器还确定 (3730) 参考半帧预测选择器信息。例如，编码器确定运动矢量的主和非主极性 (并且因此确定运动矢量预测值的主参考半帧、主极性等，见章节 IV)，在这种情况下选择器指示是否使用主极性。可供替换地，编码器使用不 同的技术来确定参考半帧预测选择器信息。例如，编码器使用不同类型的参考半帧预测。

[0420] 编码器随后随运动矢量的运动矢量差分信息和参考半帧预测选择器信息进行联合编码 (3740)。例如，编码器使用在前面章节中描述的机制来编码该信息。可供替换地，编码器使用另一种机制。

[0421] 为了简单起见，图 37A 没有示出可将技术 (3700) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述各种组合实现。

[0422] D. 解码技术

[0423] 诸如图 21 的解码器 (2100) 等解码联合编码的参考半帧预测选择器信息和差分运动矢量信息。例如，解码器执行图 37B 中所示的技术 (3750) 来解码这样的联合编码的信息。

[0424] 解码器解码 (3760) 对运动矢量进行联合编码的运动矢量差分信息和参考半帧预测选择器信息。例如，解码器解码使用在章节 IX.B 中描述的机制之一用信号表示的信息。可供替换地，解码器解码使用另一种机制用信号表示的信息。

[0425] 解码器随后确定 (3770) 运动矢量的运动矢量预测值。例如，解码器确定运动矢量的主和非主极性 (见章节 VI)，应用选择器信息，并且如在章节 VII 中所述确定所选择极性的运动矢量预测值。可供替换地，解码器使用不同的机制来确定运动矢量预测值。例如，解码器使用不同类型的参考半帧预测。

[0426] 最后，解码器通过将运动矢量差分与运动矢量预测值组合来重构 (3750) 运动矢量。

[0427] 为了简单起见，图 37B 没有示出可将技术 (3750) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述各种组合实现。

[0428] X. 在隔行扫描 P- 半帧中导出色度运动矢量

[0429] 在一些实施例中，编码器和解码器从为隔行扫描 P 半帧的宏块用信号表示的亮度运动矢量中导出色度运动矢量。色度运动矢量没有在比特流中明确地用信号表示。相反，

它们是从宏块的亮度运动矢量确定的。编码器和解码器可使用适用于逐行扫描 P 帧或隔行扫描 P 帧的色度运动矢量导出，但这一般不能为隔行扫描 P 半帧提供足够的性能。因此，编码器和解码器使用适合于隔行扫描 P 半帧的参考半帧组织的色度运动矢量导出。

[0430] 色度运动矢量导出有两个阶段：(1) 选择，和 (2) 二次采样和色度舍入。这些阶段中，选择阶段特别适合于隔行扫描 P 半帧中的色度运动矢量导出。选择阶段的输出是初始的色度运动矢量，它取决于宏块的亮度运动矢量的数量（并且有可能极性）。如果没有亮度运动用于宏块（帧内编码宏块），则没有色度运动矢量导出。如果单个亮度运动矢量用于宏块 (1MV 宏块)，则选择单个亮度运动矢量用于第二和第三阶段。如果四个亮度运动矢量用于宏块 (4MV 宏块)，则使用支持四个亮度运动矢量中最共同极性的逻辑来选择初始色度运动矢量。

[0431] A. 色度二次采样和运动矢量表示

[0432] 隔行扫描 P 半帧的宏块的色度运动矢量导出取决于用于宏块的色度二次采样类型，并且还取决于运动矢量表示。

[0433] 一些共同的色度二次采样格式是 4:2:0 和 4:1:1。图 38 示出 YUV 4:2:0 宏块的采样网格，按照它相对于亮度样本以规则的 4:1 模式对色度样本进行二次采样。图 38 示出 16x16 宏块的亮度与色度样本之间的空间关系，其中有四个 8x8 亮度块，一个 8x8 色度“U”块，和一个 8x8 色度“V”块（诸如在图 22 中所示）。总的来讲，色度网格的分辨率在 x 和 y 两个方向都是亮度网格分辨率的一半，这是在色度运动矢量导出中下采样的基础。为了将亮度网格的运动矢量距离比例缩放到色度网格的相应距离，运动矢量值除以系数 2。在此描述的选择阶段技术可应用于 YUV 4:2:0 宏块或者应用于具有另一种色度二次采样格式的宏块。

[0434] 隔行扫描 P 半帧的亮度和色度运动矢量的表示部分地取决于运动矢量和运动补偿的精度。典型的运动矢量精度是 1/2 象素和 1/4 象素，它们分别与运动补偿中的 1/2 象素和 1/4 象素内插一起工作。

[0435] 在一些实施例中，隔行扫描 P 半帧的运动矢量可参考上或下参考半帧，它们或者是相同或者是相反极性。由运动矢量值指定的垂直位移取决于当前 P 半帧和参考半帧的极性。运动矢量单位一般用半帧图像单位来表示。例如，如果运动矢量的垂直分量是 +6（以 1/4 象素为单位），这通常指示 1/2 半帧图像行的垂直位移（在为当前 P 半帧或参考半帧的不同极性调整之前，如果必要的话）。

[0436] 对于各种运动矢量分量值和半帧极性组合，图 39 示出按照第一种约定的当前和参考半帧中相应的空间位置。半帧极性的每种组合具有一对列，一个（左列）用于当前半帧中行的象素（编号的行 N = 0,1,2,等等），而另一个（右列）用于参考半帧中行的象素（也是编号的行 N = 0,1,2,等等）。圆圈表示在整数象素位置处的样 本，而 x 表示在子象素位置处的内插样本。在这种约定下，为 0 的垂直运动矢量分量值参考参考半帧中的整数象素位置（即，在实际行上的样本）。如果当前半帧和参考半帧具有相同极性，则来自当前半帧的行 N 的为 0 的垂直分量值参考参考半帧中的行 N，它在一个帧的同一实际偏移处。如果当前半帧和参考半帧具有相反极性，则来自当前半帧中行 N 的为 0 的垂直分量值仍参考参考帧中的行 N，但参考位置在帧的 1/2 象素实际偏移处，这是由于奇数与偶数行的交替。

[0437] 图 48 示出按照第二种约定的当前和参考半帧的相应空间位置。在这种约定下，为

0 的垂直运动矢量分量值参考在隔行扫描帧中同一实际偏移处的样本。参考的样本在相同极性参考半帧中的整数象素位置处，或者在相反参考半帧中的 1/2 象素位置处。

[0438] 可供替换地，隔行扫描 P 半帧的运动矢量使用另一种表示和 / 或遵循用于处理极性的垂直位移的另一种约定。

[0439] B. 选择阶段示例

[0440] 在一些实施例中，色度运动矢量导出的选择阶段适合于在具有一个或两个参考半帧的隔行扫描 P 半帧的运动矢量中使用的参考半帧模式。例如，宏块的选择阶段的结果取决于宏块的亮度运动矢量的数量和极性。

[0441] 最简单的情况是当整个宏块是帧内编码的。在这种情况下，没有色度运动矢量，并且跳过色度运动矢量导出的第二和第三阶段。宏块的色度块是帧内编码 / 解码的，没有运动补偿。

[0442] 下一个最简单的情况是当宏块具有单个亮度运动矢量用于所有四个亮度块。无论当前 P 半帧具有一个参考半帧还是两个参考半帧，本质上没有选择操作，因为简单地向前传送单个亮度运动矢量以进行舍入和二次采样。

[0443] 当宏块具有多达四个亮度运动矢量时，选择阶段更复杂。总的来说，选择阶段支持宏块的亮度运动矢量中的主极性。如果 P 半帧只有一个参考半帧，则极性与宏块的所有亮度运动矢量的相同。然而如果 P 半帧具有两个参考半帧，则宏块的不同亮度运动矢量可指向不同的参考半帧。例如，如果当前 P 半帧的极性是奇，则宏块可具有两个相反极性的亮度运动矢量（参考偶极性参考半帧）和两个相同极性的亮度运动矢量（参考奇极性参考半帧）。编码器或解码器确定宏块的亮度运动矢量的主极性，并且根据主极性的亮度运动矢量确定初始色度运动矢量。

[0444] 在一些实施例中，4MV 宏块具有零至四个运动矢量。这样一个 4MV 宏块的 亮度块是帧内编码的，或者具有相关联的相同极性亮度运动矢量，或者具有相关联的相反极性亮度运动矢量。在其它实现中，4MV 宏块总是具有四个亮度运动矢量，即使没有用信号表示它们中的一些（例如因为它们具有 (0,0) 差分）。这样一个 4MV 宏块的亮度块具有相反极性运动矢量或者具有相同极性运动矢量。选择阶段逻辑对于这些不同实现略有不同。

[0445] 1. 具有 0 至 4 个亮度运动矢量的 4MV 宏块

[0446] 图 40 中的伪代码示出选择阶段逻辑的一个示例，它应用于具有 0 和 4 个亮度运动矢量的 4MV 宏块。亮度运动矢量中，如果参考相同极性参考半帧的亮度运动矢量数量大于参考相反极性参考半帧的数量，则编码器 / 解码器从参考相同极性参考半帧的亮度运动矢量导出初始色度运动矢量。否则编码器 / 解码器从参考相反极性参考半帧的亮度运动矢量导出初始色度运动矢量。

[0447] 如果四个亮度运动矢量具有主极性（例如全部奇参考半帧或者全部偶参考半帧），编码器 / 解码器计算四个亮度运动矢量的中值。如果只有三个亮度运动矢量具有主极性（例如，因为一个亮度块是帧内编码的或者具有非主极性运动矢量），则编码器 / 解码器计算三个亮度运动矢量的中值。如果两个亮度运动矢量具有主极性，则编码器 / 解码器计算两个亮度运动矢量的平均值。（在相同和相反极性计数平分的情况下，支持（与当前 P 半帧）相同的极性。）最后，如果只一个主极性的亮度运动矢量（例如，因为三个亮度块是帧内编码的），取这个亮度运动矢量作为选择阶段的输出。如果所有亮度块是帧内编码的，

则宏块是帧内编码的，并且不应用图 40 中的伪代码。

[0448] 2. 具有 4 个亮度运动矢量的 4MV 宏块

[0449] 图 55A 和 55B 中的伪代码示出选择阶段逻辑的另一个示例，它应用于总是有 4 个亮度运动矢量的 4MV 宏块（例如，因为不允许帧内编码的亮度块。）图 55A 为 1 参考半帧隔行扫描 P 半帧中这样的 4MV 宏块处理色度运动矢量导出，并且图 55B 为 2 参考半帧隔行扫描 P 半帧中这样的 4MV 宏块处理色度运动矢量导出。

[0450] 参考图 55B，编码器 / 解码器确定在 4MV 宏块的四个亮度运动矢量中哪个极性占优势（例如奇或偶）。如果所有四个亮度运动矢量都来自相同的半帧（例如，全部奇或全部偶），则确定四个亮度运动矢量的中值。如果四个中的三个来自相同的半帧，则确定三个亮度运动矢量的中值。最后，如果每种极性有两个亮度运动矢量， 则支持具有与当前 P 半帧相同极性的两个亮度运动矢量，并且确定它们的平均值。（如果 4MV 宏块总是有四个亮度运动矢量，只一个亮度运动矢量和没有亮度运动矢量具有主极性的情况是不可能的。）

[0451] 可供替换地，编码器或解码器在从隔行扫描 P 半帧的宏块的多个亮度运动矢量中导出色度运动矢量时使用不同的选择逻辑。或者，编码器或解码器在另一类型的宏块的色度运动矢量导出中考虑亮度运动矢量极性（例如，具有不同数量的亮度运动矢量和 / 或在不同于隔行扫描 P 半帧的图像类型中的宏块）。

[0452] C. 二次采样 / 舍入阶段

[0453] 对于色度运动矢量导出的第二阶段，编码器和解码器一般应用舍入逻辑从初始色度运动矢量中消除某些象素位置（例如，要舍入到 3/4 象素位置，因此这样的色度运动矢量在下采样之后不指示 1/4 象素位移）。可调整舍入的使用以在预测质量与内插复杂性之间折衷。在更主动的舍入的情况下，例如，编码器或解码器消除得到的色度运动矢量中的所有 1/4 象素色度位移，因此只允许整数象素和 1/2 象素位移，这简化了在色度块的运动补偿中的内插。

[0454] 在第二阶段，编码器和解码器还对初始色度运动矢量进行下采样，来以适合于色度分辨率的比例获得色度运动矢量。例如，如果色度分辨率在水平和垂直方向上都是亮度分辨率的 1/2，则水平和垂直运动矢量分量按照因子 2 下采样。

[0455] 可供替换地，编码器或解码器应用其它和 / 或附加机制用于色度运动矢量的舍入、二次采样、回拉 (pullback) 或其它调整。

[0456] D. 导出技术

[0457] 诸如图 20 的编码器 (2000) 等编码器导出隔行扫描 P 半帧的宏块的色度运动矢量。或者诸如图 21 的解码器 (2100) 等解码器导出隔行扫描 P 半帧的宏块的色度运动矢量。例如，编码器 / 解码器执行图 41 所示的技术 (4100) 来导出色度运动矢量。

[0458] 编码器 / 解码器确定 (4110) 当前宏块是否为帧内编码宏块。如果是，则编码器 / 解码器跳过运动矢量导出，改为对该宏块使用运动补偿、帧内编码 / 解码。

[0459] 如果宏块不是帧内宏块，则编码器 / 解码器确定 (4120) 宏块是否为 1MV 宏块。如果是，则编码器 / 解码器使用宏块的单个亮度运动矢量作为传递给技术 (4100) 后面的调整阶段 (4150) 的初始色度运动矢量。

[0460] 如果宏块不是 1MV 宏块，则编码器 / 解码器确定 (4130) 宏块的亮度运动矢量中的主极性。例如，编码器 / 解码器确定宏块的一个或多个亮度运动矢量中的占优势极性，如图

40 或 55A 和 55B 所述。可供替换地，编码器 / 解码器应用其它和 / 或附加判定逻辑来确定占优势极性。如果包括宏块的 P 半帧只有一个参考半帧，则亮度运动矢量中的主极性简单地为该参考半帧的极性。

[0461] 编码器 / 解码器随后根据具有主极性的宏块的那些亮度运动矢量确定 (4140) 初始色度运动矢量。例如，编码器 / 解码器确定初始色度运动矢量，如图 40 或 55A 和 55B 所示。可供替换地，编码器 / 解码器使用其它和 / 或附加逻辑确定初始色度运动矢量为主极性运动矢量的中值、平均值或其它组合。

[0462] 最后，编码器 / 解码器调整 (4150) 由前面阶段之一产生的初始色度运动矢量。例如，编码器 / 解码器执行如上所述的舍入和二次采样。可供替换地，编码器 / 解码器执行其它和 / 或附加调整。

[0463] 可供替换地，编码器 / 解码器以不同的顺序检查各种宏块类型和极性条件。或者，编码器 / 解码器为隔行扫描 P 半帧或其它类型图像的其它和 / 或附加类型的宏块导出色度运动矢量。

[0464] 为了简单起见，图 41 没有示出可将技术 (4100) 与编码和解码器的其它方面集成的各种方法。在章节 XII 中详细描述了各种组合实现。

[0465] XI. 隔行扫描 P 半帧的强度补偿

[0466] 褪色、变形和混色在视频内容的创建和编辑中广泛使用。这些技术平滑视频在经过内容过渡时的视觉演变。另外，某些视频序列包括因照明而改变的自然褪色。对于受到褪色、变形、混色等影响的预测图像，在亮度方面的全局改变与参考图像相比减少了常规的运动估计和补偿的效果。结果，运动补偿的预测很糟糕，并且预测的图像需要更多的比特来表示它。这个问题对于具有一个参考半帧或者具有多个参考半帧的隔行扫描 P 半帧更复杂。

[0467] 在某些实施例中，编码器和解码器对隔行扫描 P 半帧的参考半帧执行褪色补偿 (也称为强度补偿)。编码器执行相应的褪色估计。褪色估计和补偿，以及用于褪色补偿参数的信号表示机制适合于隔行扫描 P 半帧的参考半帧组织。例如，对于具有一个参考半帧或具有两个参考半帧的隔行扫描 P 半帧，分别为每个参考半帧作出执行褪色补偿的判定。使用褪色补偿的每个参考半帧可具有它自己的褪色补偿参数。用于褪色补偿判定的信号表示机制和参数有效地表示该信息。结果，隔行 扫描视频的质量改进和 / 或减少了比特率。

[0468] A. 对于参考半帧的褪色估计和补偿

[0469] 褪色补偿涉及对一个或多个参考半帧执行改变以补偿褪色、混色、变形等等。通常，褪色补偿包括任何用于褪色 (即向黑色褪色或从黑色褪色)、混色、变形或其它影响象素值强度的自然或综合灯光效果的补偿。例如，全局亮度改变可表示为在场景的明亮度和 / 或对比度方面的改变。通常改变是线性的，但也可以被定义为包括在同一框架内的任何平滑的、非线性的映射。当前 P 半帧随后由根据经调整的一个或多个参考半帧的运动估计 / 补偿来预测。

[0470] 对于 YUV 色彩空间中的参考半帧，调整通过调整亮度和色度通道中的样本而发生。调整可包括比例缩放与平移亮度值和比例缩放与平移色度值。可供替换地，色彩空间是不同的 (例如 YIQ 或 RGB)，和 / 或补偿使用其它调整技术。

[0471] 编码器 / 解码器在逐个半帧的基础上执行褪色估计 / 补偿。可供替换地，编码器 / 解码器在某种其它基础上执行褪色估计 / 补偿。因此，褪色补偿调整影响定义的区域，它

可以是半帧或者半帧的一部分（例如，单独的块或宏块，或者一组宏块），并且褪色补偿参数用于该定义的区域。或者，褪色补偿参数用于整个半帧，但可选择地应用以及在半帧内区域需要的时候应用。

[0472] B. 隔行扫描 P 半帧的参考半帧组织

[0473] 在一些实施例中，隔行扫描 P 半帧具有一个或两个参考半帧用于运动补偿。（例如，见章节 IV。）图 24A-24F 示出可用于在隔行扫描 P 半帧的运动补偿预测中使用的参考半帧的位置。编码器和解码器可对 P 半帧的运动补偿预测使用在其它和 / 或附加位置或定时的参考半帧。例如，允许在与当前 P 半帧相同帧内的参考半帧。或者，帧的上半帧或下半帧可以首先被编码 / 解码。

[0474] 对于具有一个或两个参考半帧用于运动补偿的隔行扫描 P 半帧，P 半帧只有一个参考半帧。或者，P 半帧可具有两个参考半帧，并且在两个参考半帧之间切换用于不同的运动矢量或者在某种其它基础上切换。

[0475] 可供替换地，P 半帧具有更多的参考半帧和 / 或在不同位置的参考半帧。

[0476] C. 编码器和解码器

[0477] 图 42 示出一个示例性编码器框架 (4200)，用于为具有一个或两个参考半帧 的隔行扫描 P 半帧执行强度估计和补偿。在这个框架 (4200) 中，编码器有条件地使用通过褪色估计获得的参数来重新映射参考半帧。当编码器检测到在半帧上具有良好程度的确定性与一致性的褪色时，编码器执行重新映射或者褪色补偿。否则，褪色补偿是完全相同的操作（即输出＝输入）。

[0478] 参考图 42，编码器使用褪色检测模块 (4230) 将当前 P 半帧 (4210) 与第一参考半帧 (4220) 比较，以确定半帧 (4220，4210) 之间是否发生褪色。编码器使用褪色检测模块 (4230) 独立地将当前 P 半帧 (4210) 与第二参考半帧 (4225) 比较以确定那些半帧 (4225，4210) 之间是否发生褪色。编码器基于褪色检测的结果生成一个或多个“褪色开”或“褪色关”信号 (4240)。信号指示是否要使用褪色补偿，并且如果是，是仅在参考半帧 (4220，4225) 的第一、仅在第二还是在两个上使用褪色补偿。

[0479] 如果要对第一参考半帧 (4220) 进行褪色补偿，则褪色估计模块 (4250) 估计第一参考半帧 (4220) 的褪色参数 (4260)。（褪色估计细节在下面讨论。）同样，如果要对第二参考半帧 (4225) 进行褪色补偿，则褪色估计模块 (4250) 独立地估计第二参考半帧的褪色参数。

[0480] 褪色补偿模块 (4270，4275) 使用褪色参数 (4260) 来重新映射一个或两个参考半帧 (4220)。尽管图 42 示出两个褪色补偿模块 (4270，4275)（每参考半帧一个），但可供替换地，编码器框架 (4200) 包括单个褪色补偿模块，它在任一参考半帧 (4220，4225) 上操作。

[0481] 其它编码器模块 (4280)（例如，运动估计和补偿，频率变换器和量化模块）压缩当前 P 半帧 (4210)。编码器输出运动矢量、残差和其它信息 (4290)，它们定义编码的 P 半帧 (4210)。除了具有移位运动矢量的运动估计 / 补偿之外，框架 (4200) 可在多种多样的基于运动补偿的视频编解码器中应用。

[0482] 图 43 示出执行强度补偿的示例性解码器框架 (4300)。解码器产生解码的 P 半帧 (4310)。要解码经编码的褪色补偿 P 半帧，解码器使用褪色补偿模块 (4370，4375) 在一个或两个先前解码的参考半帧 (4320，4325) 上执行褪色补偿。可供替换地，解码器框架 (4300)

包括单个褪色补偿模块，它在任一参考半帧 (4320，4325) 上操作。

[0483] 如果褪色开 / 关信号 (4340) 指示为第一参考半帧 (4320) 和 P 半帧 (4310) 使用褪色补偿，则解码器在第一参考半帧 (4320) 上执行褪色补偿。同样，如果褪色开 / 关信号 (4340) 指示要为第二参考半帧 (4325) 和 P 半帧 (4310) 使用褪色 补偿，则解码器在第二参考半帧 (4325) 上执行褪色补偿。解码器使用在第一和第二参考半帧 (4320，4325) 的褪色估计期间获得的相应褪色参数集来执行褪色补偿 (如在编码器中所做的)。如果褪色补偿是关，则褪色补偿是完全相同的操作 (即输出＝输入)。

[0484] 其它解码器模块 (4360)(例如，运动补偿、反频率变换器和反量化模块) 使用由编码器提供的运动矢量、残差和其它信息 (4390) 来解压缩经编码的 P 半帧 (4310)。

[0485] D. 参数化和补偿

[0486] 在 P- 半帧与第一参考半帧之间和 / 或在 P- 半帧与第二参考半帧之间，参数表示褪色、混色、变形或其它改变。随后在褪色补偿中应用这些参数。

[0487] 在视频编辑中，合成褪色有时是通过应用简单的象素级线性变换于亮度和色度通道来实现的。同样，交叉褪色有时作为两个视频序列的线性和来实现，其中分量随着时间过去而改变。因此，在一些实施例中，褪色或其它强度补偿调整是作为象素级线性变换来参数化的，而交叉褪色是作为线性和来参数化的。

[0488] 假定 I(n) 是 P 半帧 n，而 I(n-1) 是一个参考半帧。在运动小的地方，通过下列方程中的一阶关系来模拟简单的褪色。由于视频序列中可能的运动，在该方程中的关系是近似的。

[0489] I(n) ≈ C1I(n-1)+B1，其中褪色参数 B1 和 C1 分别对应于参考半帧的明亮度与对比度变化。(参数 B2 和 C2 分别对应于其它参考半帧的明亮度和对比度变化。) 当非线性褪色发生时，一阶分量一般是大部分变化的主要原因。

[0490] 从图象序列 U(n) 到图象序列 V(n) 的交叉褪色可以由下列方程中的关系来模拟。再一次，由于序列中可能的运动，在该方程中的关系是近似的。

[0491] $$I(n) \approx \alpha n V + (1-\alpha n) U$$

[0492] $$\approx I(n-1) + \alpha (V-U)$$

[0493] $$\approx \begin{cases} (1-\alpha) I(n-1) & n \approx 0 \\ (1+\alpha) I(n-1) & n \approx 1/\alpha \end{cases}$$

[0494] 其中 n ≈ 0 表示交叉褪色的开始，而 n ≈ 1/α 表示交叉褪色的结束。对于跨若干半帧的交叉褪色，α 小。在交叉褪色的开始处，第 n 个半帧接近第 n-1 个半帧的衰减 (对比度＜ 1) 版本。到结束时，第 n 个半帧是第 n-1 个半帧的放大 (对比度＞ 1) 版本。

[0495] 编码器通过重新映射参考半帧来完成强度补偿。编码器在逐个象素的基础上或者在某种其它基础上重新映射参考半帧。原始的未重新映射的参考半帧实际上被丢弃 (尽管在某些实现中，未重新映射的参考半帧仍可用于运动补偿)。

[0496] 下面的线性规则按照两个参数 B1 和 C1 将参考半帧 R 的亮度值重新映射到经重新映射的参考半帧 $\hat{R}$：

[0497]

$$\hat{R} \approx C1R + B1,$$

[0498] 参考半帧的亮度值是按对比度值比例缩放 (或“加权”) 的，并且按照明亮度值平

移（即通过加一个偏移）。对于色度，重新映射遵循以下规则：

[0499]

$$\hat{R} \approx C1(R-\mu)+\mu,$$

[0500] 其中 μ 是色度值的平均值。在一个实施例中，假定 128 是色度值的无符号八比特的平均值表示。这个用于色度重新映射的规则不使用明亮度分量。在一些实施例中，两个参考的线性重新映射扩展到较高阶项。例如，将 R 的亮度值重新映射到 $\hat{R}$的二次方程是：

[0501]

$$\hat{R} \approx C1_1 R^2 + C1_2 R + B1.$$

[0502] 其它实施例使用其它重新映射规则。在这样的重新映射规则的一个类别中，对于非线性褪色，用非线性映射代替线性映射。

[0503] 褪色补偿可在运动补偿之前应用于参考半帧。或者，可在运动补偿期间需要的时候应用于参考半帧，例如，仅应用于运动矢量实际参考的参考半帧的那些区域。

[0504] E. 参数的估计

[0505] 估计是在编码过程中的计算补偿参数的过程。诸如图 42 的框架 (4200) 中的编码器等编码器在编码期间计算明亮度 (B1，B2) 和对比度 (C1，C2)。可供替换地，这样一个编码器计算其它补偿参数。

[0506] 为加快估计，编码器独立地为每个残留误差考虑和估计参数。而且，编码器只分析亮度通道。可供替换地，当更多的计算资源可用时，编码器在分析中包括色度。例如，编码器求解第一参考半帧的亮度和色度重新映射方程中的 C1（或 C2），不只是亮度，以使 C1（或 C2）更健壮。

[0507] 在褪色估计期间忽略场景中的运动。这基于以下观察结果：(a) 褪色和交叉褪色一般在静止或低运动场景中发生，以及 (b) 在高运动场景中的强度补偿的效用非常低。可供替换地，编码器联合地求解褪色补偿参数和运动信息。运动信息随后用于在该技术的后面阶段或在某个其它时间进一步提高褪色补偿参数的准确性。 使用运动信息的一种方法是从褪色估计计算中省略检测到移动的参考帧的那些部分。

[0508] Σ abs(I(n)-R) 或 $\sum abs(I(n)-\hat{R})$的绝对误差和用作确定褪色的存在和参数的度量。可供替换地，编码器使用其它或附加度量，诸如相同误差项上的均方误差或平均均方误差和，或者编码器使用不同的误差项。

[0509] 编码器可在满足退出条件，诸如下面描述的条件时结束估计。对于另一个退出条件，编码器检查对比度参数 C1（或 C2）在开始时或者在估计的中间阶段是否接近 1.0（在一个实现中，.99 ＜ C ＜ 1.02），并且如果是，则结束该技术。

[0510] 编码器通过下采样当前半帧和所选择的参考半帧（第一或第二）来开始估计。在一个实现中，编码器在水平和垂直方向按照因子 4 进行下采样。可供替换地，编码器按照另一个因子进行下取样，或者根本就不进行下采样。

[0511] 编码器随后计算 Σ abs(I_d(n)-R_d) 在当前和参考半帧的较低分辨率版本 Id(n) 和 Rd 上的绝对误差和。绝对误差和测量下采样的当前半帧与下采样的参考半帧之间在值方面的差异。如果绝对误差和比某个阈值（例如预定的差异测量）小，则编码器推定没有褪色发生过并且不使用褪色补偿。

[0512] 否则，编码器估计明亮度 B1（或 B2）和对比度 C1（或 C2）参数。第一截除估计 (cut

estimates) 是通过按照不同参数值的 Rd 模拟 Id(n) 而获得的。例如，明亮度和对比度参数是通过在整个下采样的半帧上的线性回归获得的。或者，编码器使用其它形式的统计分析，诸如总最小二乘方、最小平方中值等，用于更健壮的分析。例如，编码器最小化误差项 Id(n)-Rd 的 MSE 或 SSE。在有些环境下，MSE 和 SSE 不是健壮的，因此编码器还测试误差项的绝对误差和。编码器丢弃特定点的高误差值（这可能是由于运动而不是褪色）。

[0513] 量化和反量化第一截除参数以保证它们位于许可范围内并且测试顺应性。在一些实施例中，对于典型的八比特深度成象，这些参数各自量化为 6 比特。B1（或 B2）取从 -32 到 31 的整数值（表示为有符号的六比特整数）。C1（或 C2）从 0.5 到 1.484375 变化，以均匀的步长 0.015625(1/64)，对应于 C1（或 C2）的 0 至 63 量化值。量化是通过舍入 B1（或 B2）和 C1（或 C2）到最接收的有效反量化值并且选取合适的二进制索引来执行的。

[0514] 编码器计算原始的有界的绝对误差和 (SOrgBnd) 和重新映射的有界的绝对误差和 (SRmpBnd)。在一些实施例中，编码器使用适合度分析来计算这些和。对于原始分辨率的随机或伪随机象素集，编码器计算重新映射的有界的绝对误差和 Σ babs(I(n)-C_fR-B_f)，其中 babs(x) = min(abs(x),M) 用于某个界限 M，诸如正在编码的半帧的量化参数的倍数。界限 M 在量化参数粗略时较高，而在量化参数精细时较低。编码器还累计原始的有界的绝对误差和 Σ babs(I(n)-R)。如果计算资源可用，则编码器可计算整个半帧上的有界的误差和。

[0515] 基于原始和重新映射的有界的绝对误差和的相对值，编码器确定是否使用褪色补偿。例如，在一些实施例中，编码器不执行褪色补偿，除非重新映射的有界的绝对误差和小于或等于原始的有界的绝对误差和的某个阈值百分比 σ。在一个实现中，σ = .95。

[0516] 如果使用褪色补偿，则编码器重新计算褪色参数，这次基于 I(n) 和 R 之间的线性回归但在全分辨率上。为节省计算时间，编码器可以执行在半帧的随机或伪随机采样上的重复的线性回归。再一次，可供替换地，编码器可以使用其它形式的统计分析（例如，总最小二乘方、最小平方中值等）用于更健壮的分析。

[0517] 在一些实现中，编码器允许一种特殊情况，其中 C1（或 C2）的重构值是 -1。该特殊情况是由等于 0 的 C1（或 2）的句法元素来用信号表示的。在这个“倒置”模式中，在按 B1（或 B2）平移之间先倒置参考半帧，并且 B1（或 B2）的范围是以均匀步长 2 的 193 到 319。可供替换地，部分或全部褪色补偿参数使用另一种表示，或者使用其它和 / 或附加参数。

[0518] F. 信号表示

[0519] 在高层上，用信号表示的褪色补偿信息包括 (1) 补偿开 / 关信息和 (2) 补偿参数。开 / 关信息可进一步包括 :(a) 整体允许还是不允许褪色补偿（例如，对于整个序列）;(b) 如果允许褪色补偿，褪色补偿是否用于一个特定的 P 半帧 ;以及 (c) 如果褪色补偿用于一个特定的 P 半帧，则应该由褪色补偿调整哪些参考半帧。当褪色补偿用于参考半帧时，要应用的褪色补偿参数如下。

[0520] 1. 整体开 / 关信号表示

[0521] 在序列级上，一个比特指示是否对序列允许褪色补偿。如果允许褪色补偿，则后面的元素指示何时和如何执行它。可供替换地，在某个其它句法级上允许 / 禁止褪色补偿。或者，总是允许褪色补偿和跳过整体开 / 关信号表示。

[0522] 2.P 半帧开 / 关信号表示

[0523] 如果允许褪色补偿，则一个或多个附加信号指示何时使用褪色补偿。在典型的隔行扫描视频序列的半帧中，强度补偿的发生是很少的。有可能通过每半帧加一个比特来用信号表示P半帧的褪色补偿的使用（例如在半帧级上用信号表示的一个比特）。然而，更经济的是与其它信息一起联合地用信号表示褪色补偿的使用。

[0524] 一个选项是与运动矢量模式（例如，运动矢量的数量和配置、子象素内插方案等）一起联合地用信号表示P半帧的褪色补偿的使用。例如，VLC联合地指示最不常用的运动矢量模式和P半帧的褪色补偿的激活。对于其它细节，见美国专利申请公开号2003-0206593-A1，标题为“Fading Estimation/Compensation（褪色估计/补偿）”。或者，P半帧的褪色补偿的使用/不使用与运动矢量模式信息一起用信号表示，如在下面的若干组合实现中描述的。见章节XII，MVMODE和MVMODE2元素。可供替换地，使用用于用信号表示P半帧褪色补偿开/关信息的另一种机制。

[0525] 3. 参考半帧开/关信号表示

[0526] 如果褪色补偿用于P半帧，则可以有若干选项让参考半帧用于经受褪色补偿。当P半帧使用褪色补偿并且具有两个参考半帧时，有三种情况。执行褪色补偿用于：(1) 两个参考半帧；(2) 仅第一参考半帧（例如，时间上第二最近的参考半帧）；或者 (3) 仅第二参考半帧（例如，时间上最近的参考半帧）。褪色补偿参考半帧模式信息可对每P半帧用信号表示为一个FLC或VLC。图44中的表显示一组用于元素INTCOMPFIELD的模式信息的VLC，这在P半帧头部中用信号表示。可供替换地，图47G中的表或其它表在半帧级或另一句法级上使用。

[0527] 在一些实现中，褪色补偿的参考半帧模式是为全部P半帧用信号表示的。可供替换地，对于使用褪色补偿的1参考半帧P半帧，跳过参考半帧模式的信号表示，因为褪色补偿自动地应用于单个参考半帧。

[0528] 4. 褪色补偿参数信号表示

[0529] 如果褪色补偿用于参考半帧，则用信号表示参考半帧的褪色补偿参数。例如，第一组褪色补偿参数存在于P半帧的头部中。如果褪色补偿只用于一个参考半帧，则第一组参数用于那个参考半帧。然而如果褪色补偿用于P半帧的两个参考半帧，则第一组参数用于一个参考半帧，而第二组褪色补偿参数存在于其它参考半帧的褪色补偿的头部中。

[0530] 例如，每组褪色补偿参数包括对比度参数和明亮度参数。在一个组合实现中，第一组参数包括LUMSCALE1和LUMSHIFT1元素，当为P半帧用信号表示强度补偿时，它们存在于P半帧头部中。如果INTCOMPFIELD指示两个参考半帧或者只有第二最近的参考半帧使用褪色补偿，则LUMSCALE1和LUMSHIFT1应用于第二最近参考半帧。否则（INTCOMPFIELD指示只有最近的参考半帧使用褪色补偿），LUMSCALE1和LUMSHIFT1应用于最近参考半帧。当为P半帧用信号表示强度补偿并且INTCOMPFIELD指示两个参考半帧使用褪色补偿时，包括LUMSCALE2和LUMSHIFT2元素的第二组参数存在于P半帧头部。LUMSCALE2和LUMSHIFT2应用于较近的参考半帧。

[0531] LUMSHIFT1、LUMSCALE1、LUMSHIFT2和LUMSCALE2对应于参数B1、C1、B2和C2。LUMSCALE1、LUMSCALE2、LUMSHIFT1和LUMSHIFT2各自使用一个6比特FLC来用信号表示。可供替换地，这些参数是使用VLC用信号表示的。图56示出用于基于LUMSHIFT1和LUMSCALE1在第一参考半帧上执行褪色补偿的伪代码。为基于LUMSHIFT2和LUMSCALE2在第二参考半

帧上的褪色补偿执行类似的过程。

[0532] 可供替换地，褪色补偿参数具有不同表示和 / 或用不同的信号表示机制来用信号表示。

[0533] G. 估计和信号表示技术

[0534] 诸如图 20 的编码器 (2000) 或图 42 的框架 (4200) 中的编码器等编码器执行具有两个参考半帧的隔行扫描 P 半帧的褪色估计和相应的信号表示。例如，编码器执行图 45A 中所示的技术 (4500)。

[0535] 编码器在 P 半帧的两个参考半帧的第一个上执行褪色检测 (4510)。如果检测到褪色 (判定 4512 出来的“是”路径)，则编码器相对于第一参考半帧执行 P 半帧的褪色估计 (4514)，这产生第一参考半帧的褪色补偿参数。编码器还在 P 半帧的两个参考半帧的第二个上执行褪色检测 (4520)。如果检测到褪色 (判定 4522 出来的“是”路径)，则编码器相对于第二参考半帧执行 P 半帧的褪色估计 (4524)，这产生第二参考半帧的褪色补偿参数。例如，编码器执行褪色检测和估计，如在标题为“褪色参数的估计”一节中描述的。可供替换地，编码器使用不同的技术来检测褪色和 / 或获得褪色补偿参数。如果当前 P 半帧只有一个参考半帧，则第二参考半帧的操作可被跳过。

[0536] 编码器用信号表示 (4530) 褪色补偿对于 P 半帧是开还是关。例如，编码器 将信息与 P 半帧的运动矢量模式信息一起联合编码。可供替换地，编码器使用其它和 / 或附加信号来指示褪色补偿对于 P 半帧是开还是关。如果褪色补偿对于当前 P 半帧不是开 (判定 4532 出来的“否”路径)，则技术 (4500) 结束。

[0537] 否则 (判定 4532 出来的“是”路径)，编码器用信号表示 (4540) 褪色补偿的参考半帧模式。例如，编码器用信号表示一个 VLC，它指示褪色补偿用于两个参考半帧、仅第一参考半帧或仅第二参考半帧。可供替换地，编码器使用另一种信号表示机制 (例如 FLC) 来指示参考半帧模式。在这个路径中，编码器还用信号表示 (4542) 在褪色估计中计算的第一组和 / 或第二组褪色补偿参数。例如，编码器使用如在章节 XI.F 中描述的信号表示。可供替换地，编码器使用其它信号表示。

[0538] 尽管编码器一般还执行褪色补偿、运动估计和运动补偿，但为了简单起见，图 45A 没有示出这些操作。而且，褪色估计可在运动估计之前或同时执行。图 45A 没有示出可将技术 (4500) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细描述各种组合实现。

[0539] H. 解码和补偿技术

[0540] 诸如图 21 的解码器 (2100) 或者图 43 的框架 (4300) 中的解码器等解码器为具有两个参考半帧的隔行扫描 P 半帧执行解码和褪色补偿。例如，解码器执行图 45B 中所示的技术 (4550)。

[0541] 解码器接收和解码 (4560) 指示褪色补偿对于 P 半帧是开还是关的一个或多个信号。例如，将信息与 P 半帧的运动矢量模式信息一起联合编码。可供替换地，解码器接收和解码指示褪色补偿对于 P 半帧是开还是关的其它和 / 或附加的信号。如果褪色补偿对于 P 半帧不是开 (判定 4562 出来的“否”路径)，则技术 (4550) 结束。

[0542] 否则 (判定 4562 出来的“是”路径)，解码器接收和解码 (4570) 褪色补偿的参考半帧模式。例如，解码器接收和解码指示褪色补偿用于两个参考半帧、仅第一参考半帧还是

仅第二参考半帧的 VLC。可供替换地，解码器结合另一种信号表示机制（例如 FLC）操作以确定参考半帧模式。

[0543] 在这个路径中，解码器还接收和解码 (4572) 第一组褪色补偿参数。例如，解码器与如章节 XI.F 中描述的信号表示一起工作。可供替换地，解码器与其它信号表示一起工作。

[0544] 如果褪色补偿是仅为两个参考半帧之一执行的（判定 4575 出来的“否”路径），则第一组参数用于第一或第二参考半帧，如由参考半帧模式指示。解码器用第一组褪色补偿参数在所指示的参考半帧上执行褪色补偿 (4592)，并且技术 (4500) 结束。

[0545] 否则，褪色补偿是为全部两个参考半帧执行的（判定 4575 出来的“是”路径），并且解码器接收和解码 (4580) 第二组褪色补偿参数。例如，解码器与如在章节 XI.F 中描述的信号表示一起工作。可供替换地，解码器与其它信号表示一起工作。在这种情况下，第一组参数用于两个参考半帧之一，并且第二组参数用于另一个。解码器用第一组参数在一个参考半帧上执行褪色补偿 (4592)，并且用第二组参数在另一参考半帧上执行褪色补偿 (4582)。

[0546] 为了简单起见，图 45B 没有示出可将技术 (4550) 与编码和解码的其它方面集成的各种方法。在章节 XII 中详细地描述了各种组合实现。

[0547] XII. 组合实现

[0548] 现在描述比特流句法的详细组合实现，其中重点在于隔行扫描 P 半帧。下面的描述包括第一组合实现和一个替换的第二组合实现。另外，于 2004 年五月 27 日提交的美国专利申请序列号 10/857,473 揭示了第三组合实现的诸方面。

[0549] 尽管重点在于隔行扫描 P 半帧，但在本节的各种位置中，着眼于句法元素、语义和解码对于其它图像类型（例如，隔行扫描 P 和 B 帧、隔行扫描 I、BI、PI 和 B 半帧）的适用性。

[0550] A. 第一组合实现中的序列和语义

[0551] 在第一组合实现中，压缩的视频序列是由结构化成分等级的层中的数据构成的：图像层，宏块层和块层。序列层先于序列，并且入口点层可散布在序列中。图 46A 至 46E 示出构成各种层的比特流元素。

[0552] 1. 序列层句法和语义

[0553] 序列级头部包含序列级参数，用于解码压缩图像的序列。在有些简档 (profile) 中，与序列有关的元数据通过传输层或其它手段传送到解码器。然而对于具有隔行扫描 P 半帧的简档（高级简档），这个头部句法是视频数据比特流的一部分。

[0554] 图 46A 示出构成高级简档的序列头部的句法元素。PROFILE(4601) 和 LEVEL(4602) 元素分别指定用于编码序列的简档和简档中的编码等级。特别感兴趣的隔 行扫描 P 半帧的 INTERLACE(4603) 元素是 1 比特的句法元素，它用信号表示源内容是逐行扫描 (INTERLACE = 0) 还是隔行扫描 (INTERLACE = 1)。独立的帧在 INTERLACE = 1 时仍可使用逐行扫描或隔行扫描句法来编码。

[0555] 2. 入口点层句法和语义

[0556] 入口点头部存在于高级简档中。入口点有两个目的。首先，它用于用信号表示比特流内的随机访问点。基次，它用于用信号表示在编码控制参数方面的变化。

[0557] 图 46B 示出构成入口点层的句法元素。特别感兴趣的隔行扫描 P 半帧的参考

帧距离标志 REFDIST_FLAG(4611) 元素是 1 比特的句法元素。REFDIST_FLAG = 1 指示 REFDIST(4624) 元素存在于 I/I、I/P、P/I 或 P/P 半帧图像头部中。REFDIST_FLAG = 0 指示 REFDIST(4624) 元素不存在于 I/I、I/P、P/I 或 P/P 半帧图像头部。

[0558] 扩展的运动矢量标志 EXTENDED_MV(4612) 元素是 1 比特的元素，它指示扩展的运动矢量能力是开 (EXTENDED_MV = 1) 还是关 (EXTENDED_MV = 0)。扩展的差分运动矢量范围标志 EXTENDED_DMV(4613) 元素是 1 比特元素，如果 EXTENDED_MV = 1 则它存在。如果 EXTENDED_DMV = 1，则在扩展差分运动矢量范围内的运动矢量差分是在入口点段内的图像级上用信号表示的。如果 EXTENDED_DMV = 0，则不用信号表示在扩展的差分运动矢量范围内的运动矢量差分。扩展的差分运动矢量范围是用于隔行扫描 P 和 B 图像的选项，包括隔行扫描 P 半帧和 P 帧以及隔行扫描 B 半帧和 B 帧。

[0559] 3. 图像级句法和语义

[0560] 图像的数据由图像头部和跟随其后的宏块层数据组成。图 46C 示出构成隔行扫描半帧图像的帧头部的比特流元素。在下列描述中，重点放在与隔行扫描 P 半帧一起使用的元素上，但图 46C 所示的头部可应用于隔行扫描 I、P、B 和 BI 半帧的各种组合。

[0561] 帧编码模式 FCM(4621) 元素只存在于高级简档中，并且仅当序列层 INTERLACE(4603) 具有值 1 时存在。FCM(4621) 指示图像是被编码为逐行扫描、隔行扫描半帧还是隔行扫描帧。图 47A 中的表包括用于指示图像编码类型与 FCM 的 VLC。

[0562] 半帧图像类型 FPTYPE(4622) 元素是 3 比特句法元素，它存在于隔行扫描半 帧图像的图像头部中。FPTYPE 是按照图 47B 中的表解码的。如表所示，隔行扫描帧可包括两个隔行扫描 I 半帧、一个隔行扫描 I 半帧和一个隔行扫描 P 半帧、两个隔行扫描 P 半帧、两个隔行扫描 B 半帧、一个隔行扫描 B 半帧和一个隔行扫描 BI 半帧或者两个隔行扫描 BI 半帧。

[0563] 上半帧第一 TFF(4623) 元素是 1 比特元素，如果序列头部元素 PULLDOWN = 1 并且序列头部元素 INTERLACE = 1，则存在于高级简档图像头部中。TFF = 1 指上半帧是第一解码的半帧。如果 TFF = 0，则下半帧是第一解码的半帧。

[0564] P 参考距离 REFDIST(4624) 元素是可变大小句法元素，如果入口级标志 REFDIST_FLAG = 1 并且如果图像类型不是 BB、BBI、BI/B、BI/BI，则存在于隔行扫描半帧图像头部中。如果 REFDIST_FLAG = 0，REFDIST(4624) 设置为默认值 0。REFDIST(4624) 指示当前帧与参考帧之间的帧数。图 47C 中的表包括用于 REFDIST(4624) 值的 VLC。表中最后一行指示用于表示大于 2 的参考帧距离的码字。这些编码为 (二进制)11，后面是 N-3 个 1，其中 N 是参考帧距离。码字中的最后一个比特是 0。REFDIST(4624) 的值小于或等于 16。例如 :

[0565] N = 3，VLC 码字 = 110，VLC 大小 = 3，

[0566] N = 4，VLC 码字 = 1110，VLC 大小 = 4，以及

[0567] N = 5，VLC 码字 = 11110，VLC 大小 = 5。

[0568] 半帧图像层 FIELDPICLAYER(4625) 元素是用于隔行扫描帧的各个隔行扫描半帧之一的数据。如果隔行扫描帧是 P/P 帧 (FPTYPE = 011)，则比特流包括两个 FIELDPICLAYER(4625) 元素用于两个隔行扫描 P 半帧。图 46D 示出构成隔行扫描 P 半帧图像的半帧图像头部的比特流元素。

[0569] 参考图像数量 NUMREF(4631) 元素是存在于隔行扫描 P 半帧头部中的 1 比特句法元素。它指示隔行扫描 P 半帧具有 1(NUMREF = 0) 或 2(NUMREF = 1) 个参考图像。参考半

帧图像指示符 REFFIELD(4632) 是如果 NUMREF ＝ 0 则存在于隔行扫描 P 半帧中的 1 比特句法元素。它指示两个可能的参考图像中的哪一个由隔行扫描 P 半帧使用。

[0570] 扩展的 MV 范围标志 MVRANGE(4633) 是可变大小句法元素，它通常指示运动矢量的扩展范围（即，运动矢量的较长的可能水平和 / 或垂直位移）。扩展的差分 MV 范围标志 DMVRANGE(4634) 是可变大小句法元素，如果 EXTENDED_DMV ＝ 1 则存在。图 47D 中的表用于 DMVRANGE(4634) 元素。两个 MVRANGE(4633) 和 DMVRANGE(4634) 用于解码运动矢量差分，而且扩 展的差分运动矢量范围是隔行扫描 P 半帧、隔行扫描帧、隔行扫描 B 半帧和隔行扫描 B 帧的选项。

[0571] 运动矢量模式 MVMODE(4635) 元素是可变大小句法元素，它用信号表示四个运动矢量编码模式之一或者一个强度补偿模式。运动矢量编码模式包括具有运动补偿的不同子象素内插规则的三个“1MV”模式。1MV 表示图像中每个宏块具有至少一个运动矢量。在“混合 MV”模式中，图像中的每个宏块可具有一个或四个运动矢量，或者跳过。取决于 PQUANT（图像的量化系数）的值，图 47E 中所示的表之一用于 MVMODE(4635) 元素。

[0572] 运动矢量模式 2MVMODE2(4636) 元素是可变大小句法元素，如果 MVMODE(4635) 用信号表示强度补偿则存在于隔行扫描 P 半帧头部中。取决于 PQUANT 的值，图 47F 中所示的任一表用于 MVMODE(4635) 元素。

[0573] 强度补偿半帧 INTCOMPFIELD(4637) 是存储于隔行扫描 P 半帧图像头部中的可变大小句法元素。如图 47G 的表中所示，INTCOMPFIELD(4637) 用于指示哪个（些）参考半帧受到强度补偿。即使 NUMREF ＝ 0，INTCOMPFIELD(4637) 也存在。半帧图像亮度比例 1LUMSCALE1(4638)、半帧图像亮度平移 1LUMSHIFT1(4639)、半帧图像亮度比例 2LUMSCALE2(4640) 和半帧图像亮度平移 2LUMSHIFT2(4641) 元素各自是在强度补偿中的 6 比特值。如果 MVMODE(4635) 用信号表示强度补偿，则 LUMSCALE1(4638) 和 LUMSHIFT1(4639) 元素存在。如果 INTCOMPFIELD(4637) 元素是“1”或“00”，则 LUMSCALE1(4638) 和 LUMSHIFT1(4639) 应用于上半帧。否则，LUMSCALE1(4638) 和 LUMSHIFT1(4639) 应用于下半帧。如果 MVMODE(4635) 用信号表示强度补偿并且 INTCOMPFILED(4637) 元素是“1”，则 LUMSCALE2(4640) 和 LUMSHIFT2(4641) 元素存在。LUMSCALE2(4640) 和 LUMSHIFT2(4641) 应用于下半帧。

[0574] 宏块模式表 MBMODETAB(4642) 元素是固定长度域，其中 3 比特值用于隔行扫描 P 半帧头部。MBMODETAB(4642) 指示八个代码表中的哪一个（如用 3 比特值指定的表 0 至 7）用于在宏块层中编码 / 解码宏块模式 MBMODE(4661) 句法元素。有两组八个代码表，并且使用的组取决于 4MV 宏块在图像中是否可能。图 47H 示出可用于混合 MV 模式的隔行扫描 P 半帧中 MBMODE(4661) 的八个表。图 47I 示出可用于 1MV 模式的隔行扫描 P 半帧中的 MBMODE(4661) 的八个表。

[0575] 运动矢量表 MVTAB(4643) 元素是固定长度域。对于其中 NUMREF ＝ 0 的隔 行扫描 P 半帧，MVTAB(4643) 是 2 比特句法元素，它指示四个代码表中哪一个（如用 2 比特值指定的表 0 至 3）用于解码运动矢量数据。对于其中 NUMREF ＝ 1 的隔行扫描 P 半帧，MVTAB(4643) 是 3 比特句法元素，它指示八个代码表中的哪一个（如用三比特值指定的表 0 至 7）用于编码 / 解码运动矢量数据。

[0576] 在隔行扫描 P 半帧头部中，如果 MVMODE(4635)（或者 MVMODE2(4636)，如

果 MVMODE(4635) 设置为强度补偿）指示图像是混合 MV 类型的，则 4MV 块模式表 4MVBPTAB(4644) 元素是 2 比特值。4MVBPTAB(4644) 句法元素用信号表示四个表中哪一个（如用 2 比特值指定的表 0 至 3）用于 4MV 宏块中的 4MV 块模式 4MVBP(4664) 句法元素。图 47J 示出可用于 4MVBP(4664) 的四个表。

[0577] 隔行扫描 P 帧头部（未示出）具有许多与图 46C 所示的半帧编码的隔行扫描帧头部以及图 46D 所示的隔行扫描 P 半帧头部相同的元素。这些包括 FCM(4621)、MVRANGE(4633)、DMVRANGE(4634)、MBMODETAB(4642) 和 MVTAB(4643)，尽管隔行扫描 P 帧的精确句法和语义可不同于隔行扫描 P 半帧。隔行扫描帧头部还包括图像类型的不同元素、在 1MV 和 4MV 模式之间切换和强度补偿信号表示。

[0578] 由于隔行扫描 P 帧可包括具有每宏块两个运动矢量的半帧编码宏块，隔行扫描帧头部包括 2 运动矢量块模式表 2MVBPTAB 元素。2MVBPTAB 是存在于隔行扫描 P 帧中的两个 2 比特值。这个句法元素用信号表示四个表中哪一个（用两比特值指定的表 0 至 3）用于解码 2MV 半帧编码宏块中的 2MV 块模式 (2MVBP) 元素。图 47K 示出可用于 2MVBP 的四个表。

[0579] 隔行扫描 B 半帧和隔行扫描 B 帧具有许多隔行扫描 P 半帧与隔行扫描帧的相同元素。具体地，隔行扫描 B 半帧可包括 4MVBPTAB(4644) 句法元素。隔行扫描 B 帧包括 2MVBPTAB 和 4MVBPTAB(4644) 两个句法元素，尽管这些元素的语义可以不同。

[0580] 4. 宏块层句法和语义

[0581] 宏块的数据由宏块头部和跟随其后的块层组成。图 46E 示出隔行扫描 P 半帧的宏块层结构。

[0582] 宏块模式 MBMODE(4661) 元素是可变大小元素。它联合地指示信息，诸如宏块 (1MV、4MV 或帧内编码）的运动矢量的数量、对该宏块是否存在已编码块模式 CBPCY(4662) 元素以及（在有些情况下）对该宏块是否存在运动矢量差分 数据。图 47H 和 47I 示出可用于隔行扫描 P 半帧的 MBMODE(4661) 的表。

[0583] 运动矢量数据 MVDATA(4663) 元素是可变大小元素，它编码运动矢量的运动矢量信息（例如水平和垂直差分）。对于具有两个参考半帧的隔行扫描 P 半帧，MVDATA(4663) 还编码用于在运动矢量的多个可能运动矢量预测值之间选择的信息。

[0584] 四个运动矢量块模式 4MVBP(4664) 元素是可变大小句法元素，它可存在于隔行扫描 P 半帧、B 半帧、帧和 B 帧的宏块中。在隔行扫描 P 半帧、B 半帧和 P 帧的宏块中，如果 MBMODE(4661) 指示该宏块具有 4 个运动矢量，则 4MVBP(4664) 元素存在。在这种情况下，4MVBP(4664) 指示 4 个亮度块中哪一个包含非零运动矢量差分。

[0585] 在隔行扫描 B 帧的宏块中，如果 MBMODE(4661) 指示宏块包含 2 个半帧运动矢量并且如果宏块是内插的宏块，则 4MVBP(4664) 存在。在这种情况下，4MVBP(4664) 指示四个运动矢量（上和下半帧前向运动矢量和上和下半帧后向运动矢量）中哪一个存在。

[0586] 两个运动矢量块模式 2MVBP 元素（未示出）是可变大小句法元素，它存在于隔行扫描 P 帧和 B 帧中。在隔行扫描 P 帧宏块中，如果 MBMODE(4661) 指示宏块具有 2 个半帧运动矢量，则 2MVBP 存在。在这种情况下，2MVBP 指示 2 个半帧（上和下）中哪一个包含非零运动矢量差分。在隔行扫描 B 帧宏块中，如果宏块包含 1 个运动矢量并且宏块是内插的宏块，则 2MVBP 存在。在这种情况下，2MVBP 指示两个运动矢量（前向和后向运动矢量）中哪一个存在。

[0587] 块级运动矢量数据 BLKMVDATA(4665) 元素是在某些情况下存在的可变大小元素。它包含宏块的块的运动信息。

[0588] 混合运动矢量预测 HYBRIDPRED(4666) 元素是每运动矢量 1 比特句法元素，它可存在于隔行扫描 P 半帧的宏块中。当使用混合运动矢量预测时，HYBRIDPRED(4666) 指示两个运动矢量预测值中哪一个要使用。

[0589] 5. 块层句法和语义

[0590] 隔行扫描图像的块层遵循逐行扫描图像的块层的句法和语义。通常，块或子块的 DC 和 AC 系数的信息是在块层上用信号表示的。

[0591] B. 第一组合实现中的解码

[0592] 当视频序列由隔行扫描视频帧组成或者包括隔行扫描和逐行扫描帧的混合时，FCM(4621) 元素指示给定的图像编码为逐行扫描帧、隔行扫描半帧还是隔行扫描帧。对于编码为隔行扫描半帧的帧，FPTYPE(4622) 指示帧包括两个隔行扫描 I 半帧、一个隔行扫描 I 半帧和一个隔行扫描 P 半帧、两个隔行扫描 P 半帧、两个隔行扫描 B 半帧、一个隔行扫描 B 半帧和一个隔行扫描 BI 半帧或者两隔行扫描 BI 半帧。隔行扫描半帧的解码如下。下面的章节集中于隔行扫描 P 半帧的解码过程。

[0593] 1. 隔行扫描 P 半帧解码的参考

[0594] 隔行扫描 P 半帧可在运动补偿中参考一个或两个先前解码的半帧。NUMREF(4631) 元素指示当前 P 半帧可参考一个还是两个先前的参考半帧。如果 NUMREF ＝ 0，则当前 P 半帧只可参考一个半帧。在这种情况下，REFFIELD(4632) 元素在比特流中跟随其后。REFFIELD(4632) 指示哪一个解码的半帧用作参考。如果 REFFIELD ＝ 0，则时间上最近（在显示顺序中）的 I 半帧或 P- 半帧用作参考。如果 REFFIELD ＝ 1，则第二时间上最近的 I 半帧或 P 半帧用作参考。如果 NUMREF ＝ 1，则当前 P 半帧使用两个时间上最近（在显示顺序中）的 I 半帧或 P 半帧作为参考。如上所述示于图 24A-24F 的 NUMREF ＝ 0 和 NUMREF ＝ 1 的参考半帧图像的示例应用于第一组合实现。

[0595] 2. 图像类型

[0596] 隔行扫描 P 半帧可以是两种类型之一 :1MV 或混合 MV。在 1MVP 半帧中，每个宏块是 1MV 宏块。在混合 MVP 半帧中，每个宏块可被编码为 1MV 或 4MV 宏块，如由在每个宏块上的 MBMODE(4661) 指示的。1MV 或混合 MV 模式是由 MVMODE(4635) 或 MVMODE2(4636) 元素为隔行扫描 P 半帧用信号表示的。

[0597] 3. 宏块模式

[0598] 在隔行扫描 P 半帧中的宏块可以是 3 种可能类型之一 :1MV、4MV 和帧内。MBMODE(4661) 元素指示宏块类型 (1MV、4MV 或帧内)，并且还指示 CBP 和 MV 数据的存在。取决于 MVMODE(4635)/MVMODE2(4636) 句法元素指示隔行扫描 P 半帧是混合 MV 还是全部 1MV，MBMODE(4661) 如下用信号表示 信息。

[0599] 图 26 中的表显示 MBMODE(4661) 如何用信号表示有关在全部 1MVP 半帧中的宏块的信息。如在图 47I 中所示，8 个表之一用于编码 / 解码 1MVP 半帧的 MBMODE(4661)。图 27 中的表显示 MBMODE(4661) 用信号表示有关混合 MVP 半帧中宏块的信息。如图 47H 所示，8 个表之一用于编码 / 解码混合 MVP 半帧的 MBMODE(4661)。

[0600] 因而，1MV 宏块可在 1MV 和混合 MV 隔行扫描 P 半帧中出现。在 1MV 宏块中，单个运

动矢量表示宏块中全部6个块的当前和参考图像之间的位移。对于1MV宏块，MBMODE(4661)元素指示以下三项：(1) 宏块类型是1MV；(2) 对该宏块是否存在CBPCY(4662)元素；以及(3) 对该宏块是否存在MVDATA(4663)元素。

[0601] 如果MBMODE(4661)元素指示CBPCY(4662)元素存在，则CBPCY(4662)元素存在于相应位置的宏块层中。CBPCY(4662)指示6个块中哪一个在块层中编码。如果MBMODE(4661)元素指示CBPCY(4662)不存在，则CBPCY(4662)假定等于0并且对于宏块中6个块的任何一个不存在块数据。

[0602] 如果MBMODE(4661)元素指示MVDATA(4663)元素存在，则MVDATA(4663)元素存在于相应位置的宏块层中。MVDATA(4663)元素编码运动矢量差分，它与运动矢量预测值组合起来重构运动矢量。如果MBMODE(4661)元素指示MVDATA(4663)元素不存在，则运动矢量差分假定为零并且因此运动矢量等于运动矢量预测值。

[0603] 4MV宏块出现在混合MVP半帧中。在4MV宏块中，宏块中的4个亮度块的每一个可具有相关联的运动矢量，它指示该块的当前和参考图像之间的位移。色度块的位移是从4个亮度运动矢量导出的。当前和参考块之间的差在块层中编码。对于4MV宏块，MBMODE(4661)元素指示以下两项：(1) 宏块类型是4MV；以及(2)CBPCY(4662)元素是否存在。

[0604] 帧内编码宏块可出现在1MV或混合MVP半帧中。在帧内编码宏块中，在不参考任何先前的图像数据的情况下编码全部六个块。对于帧内编码宏块，MBMODE(4661)元素指示以下两项：(1) 宏块类型是帧内编码；以及(2)CBPCY(4662)元素是否存在。对于帧内编码宏块，当CBPCY(4662)元素存在时，指示6个块中哪一个具有在块层中编码的AC系数数据。在所有情况下，DC系数对于每个块仍存在。

[0605] 4. 运动矢量块模式

[0606] 4MVBP(4664)元素指示4个亮度块中哪一个包含非零运动矢量差分。4MVBP(4664)被解码为0与15之间的值，它在表示为二进制值时表示1比特句法元素，它指示相应亮度块的运动矢量是否存在。图34中的表显示亮度块与4MVBP(4664)中比特的关联。如在图47J中所示，4个表之一用于编码/解码4MVBP(4664)。

[0607] 对于4MVBP(4664)中4个比特位置的每一个，值0指示对相应位置中的块不存在运动矢量差分(在BLKMVDATA中)，并且运动矢量差分假定为0。值1指示对相应位置中的块存在运动矢量差分(在BLKMVDATA中)。例如，如果4MVBP(4664)解码为二进制值1100，则比特流包含块0和1的BLKMVDATA(4665)，并且块2和3没有BLKMVDATA(4665)。4MVBP(4664)相似地用于指示隔行扫描B半帧和隔行扫描P帧中4MV宏块的运动矢量差分信息的存在/不存在。

[0608] 在隔行扫描P帧或隔行扫描B中的半帧编码的宏块可包括2个运动矢量。在2个半帧MV宏块的情况下，2MVBP元素指示两个半帧中哪一个具有非零的差分运动矢量。如图47K所示，4个表之一用于编码/解码2MVBP。

[0609] 5. 半帧图像坐标系统

[0610] 在下列章节中，运动矢量单位是以半帧图像单位表示的。例如，如果运动矢量指示位移为+6(以四分之一象素为单位)的垂直分量，则这指示1/2的半帧图像行的位移。

[0611] 图48示出当前和参考半帧极性(相反和相同)的两个组合的运动矢量的垂直分

量与空间位置之间的关系。图 48 示出当前和参考半帧中一个垂直象素列。圆圈表示整数象素位置并且 x 表示四分之一象素位置。值 0 指示当前和参考半帧位置之间没有垂直位移。如果当前和参考半帧是相反极性的，则 0 垂直矢量指向参考半帧中半帧行之间中间的位置 (1/2 象素平移)。如果当前和参考半帧是相同极性的，则 0 垂直矢量指向参考半帧中相应的半帧行。

[0612] 6. 解码运动矢量差分

[0613] MVDATA(4663) 和 BLKMVDATA(4665) 元素编码宏块中宏块或块的运动信息。1MV 宏块具有单个 MVDATA(4663) 元素，而 4MV 宏块可具有零到四 个 BLMMVDATA(4665)。根据 MVDATA(4663) 或 BLKMVDATA(4665) 计算运动矢量差分的过程对于一个参考 (NUMREF = 0) 情况和两个参考 (NUMREF = 1) 情况不同。

[0614] 在具有一个参考半帧的半帧图像中，每个 MVDATA(4663) 或 BLKMVDATA(4665) 句法元素联合地编码以下两项 :(1) 水平运动矢量差分分量 ;和 (2) 垂直运动矢量差分分量。MVDATA(4663) 或 BLKMVDATA(4665) 元素是一个 VLC，之后跟随一个 FLC。VLC 的值确定 FLC 的尺寸。MVTAB(4643) 句法元素指定用于解码 VLC 的表。

[0615] 图 49A 示出伪代码，它示出对具有一个参考半帧的半帧图像中块或宏块的运动矢量的运动矢量差分解码。在该伪代码中，计算值 dmv_x 和 dmv_y，其中 dmv_x 是差分水平运动矢量分量而 dmv_y 是差分垂直运动矢量分量。变量 k_x 和 k_y 是固定长度值，它们取决于如由 MVRANGE(4633) 按照图 49B 所示的表定义的运动矢量范围。

[0616] 变量 extend_x 用于扩展的范围水平运动矢量差分，并且变量 extend_y 用于扩展的范围垂直运动矢量差分。变量 extend_x 和 extend_y 是从 DMVRANGE(4634) 句法元素导出的。如果 DMVRANGE(4634) 指示使用水平分量的扩展的范围，则 extend_x = 1。否则 extend_x = 0。同样，如果 DMVRANGE(4634) 指示使用垂直分量的扩展的范围，则 extend_y = 1。否则，extend_y = 0。偏移表是如下定义的数组 :

[0617] offset_table1[9] = {0,1,2,4,8,16,32,64,128}，以及

[0618] offset_table2[9] = {0,1,3,7,15,31,63,127,255}，

[0619] 其中当为水平或垂直分量扩展差分范围时，offset_table[] 用于该水平或垂直分量。尽管图 49A 和 49B 示出隔行扫描 P 半帧的扩展的差分运动矢量解码，但扩展的差分运动矢量解码还用于第一组合实现中的隔行扫描 B 半帧、隔行扫描 P 帧和隔行扫描 B 帧。

[0620] 在具有两个参考半帧的半帧图像中，每个 MVDATA(4663) 或 BLKMVDATA(4665) 句法元素联合地编码以下三项 :(1) 水平运动矢量差分分量 ;(2) 垂直运动矢量差分分量 ;以及 (3) 使用主还是非主预测值，即两个半帧中哪一个是由运动矢量参考的。如在一个参考半帧情况中，MVDATA(4663) 或 BLKMVDATA(4665) 元素是一个 VLC，其后跟着一个 FLC，并且 MVTAB(4643) 句法元素指定用于解码 VLC 的表。

[0621] 图 50 示出伪代码，它示出对具有两个参考半帧的半帧图像中块或宏块的运动 矢量的运动矢量差分和主 / 非主预测值解码。在该伪代码中，值预测值标志是一个二进制标志，它指示使用主还是非主运动矢量预测值。如果 predictor_flag = 0，则使用主预测值，并且如果 predictor_flag = 1，则使用非主预测值。各种其它变量 (包括 dmv_x，dmv_y，k_x，k_y，extend_x，extend_y，offset_table[] 和 offset_table2[]) 是为一个参考半帧情况描述的。表 size_table 是如下定义的数组 :

[0622] size-table[16] = {0,0,1,1,2,2,3,3,4,4,5,5,6,6,7,7}.

[0623] 7. 运动矢量预测值

[0624] 运动矢量是通过将在前面章节中计算的运动矢量差分加到运动矢量预测值来计算的。预测值是根据最多三个相邻运动矢量计算的。运动矢量预测值的计算是以 1/4 象素单位完成的，即使运动矢量模式是半象素。

[0625] 在 1MV 隔行扫描 P 半帧中，最多三个相邻运动矢量用于计算当前宏块的预测值。具有考虑的运动矢量的相邻宏块的位置在图 5A 和 5B 中所示，并且是为 1MV 逐行扫描 P 帧描述的。

[0626] 在混合运动矢量隔行扫描 P 半帧中，最多三个相邻运动矢量用于计算当前块或宏块的预测值。具有考虑的运动矢量的相邻块和 / 或宏块的位置在图 6A-10 中所示，并且是为混合 MV 逐行扫描 P 帧描述的。

[0627] 如果图像头部中的 NUMREF(4631) 句法元素为 0，则当前隔行扫描 P 半帧可只参考一个先前编码的半帧。如果 NUMREF = 1，则当前隔行扫描 P 半帧可参考两个最近的参考半帧图像。在前一情况中，为每个运动矢量计算单个预测值。在后一种情况下，计算两个运动矢量预测值。图 51A 和 51B 中的伪代码描述如何为一个参考半帧情况计算运动矢量预测值。伪代码中的变量 fieldpred_x 和 fieldpred_y 表示运动矢量预测值的水平和垂直分量。

[0628] 在两个参考半帧隔行扫描 P 半帧 (NUMREF = 1) 中，当前半帧可参考两个最近的参考半帧。在这种情况下，为每个帧间编码的宏块计算两个运动矢量预测值。一个预测值来自相同极性的参考半帧，而另一个来自相反极性的参考半帧。相同极性半帧与相反极性半帧中，一个是主半帧而另一个是非主半帧。主半帧是包含大多数候选运动矢量预测值的半帧。在平分的情况下，从相反半帧导出的运动矢量视为主预测值。帧内编码的宏块在主 / 非主预测值的计算中不考虑。如果所有候选预测值宏块是帧内编码的，则主和非主运动矢量预测值设置为零，并且从相反半帧中取得主预测值。

[0629] 图 52A-52F 中的伪代码描述在给出 3 个候选运动矢量预测值时如何为两个参考半帧情况计算运动矢量预测值。变量 samefieldpred_x 和 samefieldpred_y 表示来自相同半帧的运动矢量预测值的水平和垂直分量，并且变量 oppositefieldpred_x 和 oppositefieldpred_y 表示来自相反半帧的运动矢量预测值的水平和垂直分量。变量 samecount 和 oppositecount 被初始化为 0。变量 dominatpredictor 指示哪一个半帧包含主预测值。值 predictor_flag(从运动矢量差分解码的) 指示使用主还是非主预测值。

[0630] 图 52G 和 52H 中的伪代码示出图 52A-52F 中伪代码引用的比例缩放操作，它们用于从一个半帧的预测值导出另一个半帧的预测值。SCALEOPP、SCALESAME1、SCALESAME2、SCALEZONE1_X、SCALEZONE1_Y、ZONE1OFFSET_X 和 ZONE1OFFSET_Y 的值对当前半帧是第一半帧的情况在图 52I 的表中示出，而对当前半帧是第二半帧的情况在图 52J 的表中示出。参考帧距离是在图像头部的 REFDIST(4624) 中编码的。参考帧距离是 REFDIST+1。

[0631] 图 52K 至 52N 是伪代码和表，用于可替换图 52H 至 52J 中所示的比例缩放操作。代替图 52H 至 52J 中的比例缩放伪代码和表 (但仍使用图 52A 至 52G 中的伪代码)，使用图 52K 至 52N 中的比例缩放伪代码和表。从半帧层头部的元素获得参考帧距离。N 值取决于运动矢量范围，如在图 52N 的表中所示。

[0632] 8. 混合运动矢量预测

[0633] 相对于A(上)和C(左)预测值测试在前面章节中计算的运动预测值,以确定参考图像是否明确地编码在比特流中。如果是,则存在指示使用预测值A还预测值C作为运动矢量预测值的一个比特。图53中的伪代码示出混合运动矢量预测解码。在该伪代码中,变量predictor_pre_x和predictor_pre_y分别是如在前面章节中计算的水平和垂直运动矢量预测值。变量predictor_post_x和predictor_post_y分别是在检查混合运动矢量预测之后的水平和垂直运动矢量预测值。变量predictor_pre、predictor_post、predictorA、predictorB和predictorC全部表示由预测值标志的值指示的极性的半帧。例如,如果predictor_flag指示使用相反半帧预测值,则:

[0634] predictor_pre_x = oppositefieldpred_x

[0635] predictor_pre_x = oppositefieldpred_y

[0636] predictorA_x = oppositefieldpredA_x

[0637] predictorA_y = oppositefieldpredA_y

[0638] predictorB_x = oppositefieldpredB_x

[0639] predictorB_y = oppositefieldpredB_y

[0640] predictorC_x = oppositefieldpredC_x

[0641] predictorC_y = oppositefieldpredC_y

[0642] 同样,如果predictor_flag指示使用相同半帧预测值,则:

[0643] predictor_pre_x = samefieldpred_x

[0644] predictor_pre_x = samefieldpred_y

[0645] predictorA_x = samefieldpredA_x

[0646] predictorA_y = samefieldpredA_y

[0647] predictorB_x = samefieldpredB_x

[0648] predictorB_y = sainefieldpredB_y

[0649] predictorC_x = samefieldpredC_x

[0650] predictorC_y = samefieldpredC_y

[0651] 其中oppositefieldpred和samefieldpred的值是如在前面章节中描述的那样计算的。

[0652] 9. 重构亮度运动矢量

[0653] 对于1MV和4MV宏块两者,通过如下那样将差分加到预测值来重构亮度运动矢量,其中变量range_x和range_y取决于MVRANGE(4633)并且在图49B所示的表中指定。对于NUMREF = 0(1参考半帧隔行扫描P半帧):

[0654] my_x = (dmv_x+predictor_x)smod range_x,以及

[0655] my_y = (dmv_y+predictor_y)smod(range_y).

[0656] 对于NUMREF = 1(2参考半帧隔行扫描P半帧):

[0657] my_x = (dmv_x+predictor_x)smod range_x,以及

[0658] mv_y = (dmv_y+predictor_y)smod(range_y/2).

[0659] 如果隔行扫描P半帧使用两个参考图像(NUMREF = 1),则predictor_flag(在解码运动矢量差分中导出的)与dominantpredictor(在运动矢量预测中导出的)的值相组合以确定使用哪些半帧作为参考,如图54所示。

[0660] 在 1MV 宏块中，对用于构成宏块的亮度分量的 4 个块存在单个运动矢量。如果 MBMODE(4661) 句法元素指示在宏块层中不存在 MV 数据，则 dmv_x ＝ 0 且 dmv_y ＝ 0(mv_x ＝ predictor_x 且 mv_y ＝ predictor_y)。

[0661] 在 4MV 宏块中，宏块中的帧间编码的亮度块的每一个具有它自己的运动矢量。因此，在每个 4MV 宏块中有 4 个亮度运动矢量。如果 4MVBP(4664) 句法元素指示一个块没有运动矢量信息，则该块的 dmv_x ＝ 0 且 dmv_y(mv_x ＝ predictor_x 且 mv_y ＝ predicotr_y)。

[0662] 10. 导出色度运动矢量

[0663] 色度运动矢量是从亮度运动矢量导出的。色度运动矢量在两个步骤中重构。作为第一步骤，通过适当地组合和比例缩放亮度运动矢量获得名义的色度运动矢量。比例缩放是以这样一种方法进行的，即与四分之一象素偏移相比，二分之一偏移是较佳的。在第二步骤中，使用 1 比特 FASTUVMC 句法元素来确定色度运动矢量的进一步舍入是否必要。如果 FASTUVMC ＝ 0，则在第二步骤中不进行舍入。如果 FASTUVMC ＝ 1，则在四分之一象素偏移处的色度运动矢量将舍入到最接近的二分之一和整象素位置。只有双线性滤波用于全部色度内插。变量 cmv_x 和 cmv_y 分别表示色度运动矢量分量，而 lmv_x 和 lmv_y 分别表示亮度运动矢量分量。

[0664] 在 1MV 宏块中，色度运动矢量是从亮度运动矢量导出的，如下：

[0665] cmv_x ＝ (lmv_x+round[lmv_x&3]) ＞＞ 1，以及

[0666] cmv_y ＝ (lmv_y+round[lmv_y&31) ＞＞ 1，

[0667] 其中 round[0] ＝ 0，round[1] ＝ 0，round[2] ＝ 0，round[3] ＝ 1。

[0668] 图 55A 和 55B 中的伪代码示出色度运动矢量是如何从 4MV 宏块的四个亮度块中的运动信息导出的第一阶段。在该伪代码中，ix 和 iy 是临时变量。图 55A 是用于 1 参考半帧隔行扫描 P 半帧的色度运动矢量导出的伪代码，而图 55B 是用于 2 参考半帧隔行扫描 P 半帧的色度运动矢量导出的伪代码。

[0669] 11. 强度补偿

[0670] 如果 MVMODE(4635) 指示对隔行扫描 P 半帧使用强度补偿，则参考半帧之一或两个中的象素在使用它们作为当前 P 半帧的预测值之前被重新映射。当使用强度补偿时，LUMSCALE1(4638) 和 LUMSHIFT1(4639) 句法元素存在于第一参考半帧的比特流中，而 LUMSCALE2(4640) 和 LUMSHIFT2(4641) 元素也可存在于第二参考半帧的比特流中。图 56 中的伪代码示出如何使用 LUMSCALE1(4698) 和 LUMSHIFT1(4639) 值来建立用于重新映射第一参考半帧的参考半帧 象素的查找表。(该伪代码可相似地应用于第二参考半帧的 LUMSCALE2(4640) 和 LUMSHIFT2(4641)。)

[0671] 参考半帧的 Y 分量是使用 LUTY[] 表重新映射的，而 Cb/Cr 分量是使用 LUTUV[] 表重新映射的，如下：

[0672] $\overline{p}_Y = LUTY[p_Y]$，以及

[0673] $\overline{p}_{UV} = LUTUV[p_{UV}]$，

[0674] 其中 p_Y 是参考半帧中的原始亮度象素值，$\overline{p}_Y$ 是参考半帧中重新映射的亮度象素值，p_{UV} 是参考半帧中原始的 Cb 或 Cr 象素值，而 $\overline{p}_{UV}$ 是参考半帧中重新映射的 Cb 或 Cr 象素值。

[0675] 12. 剩余的解码

[0676] 当CBPCY(4662)元素存在时，解码器解码宏块的该元素，其中CBPCY(4662)元素指示系数数据的存在/不存在。在块层上，解码器解码帧间编码的块和帧内编码的块（除了4MV宏块）的系数数据。为重构帧间编码的块，解码器：(1)选择一个变换类型(8x8,4x4,4x8,或者4x4)，(2)解码子块模式，(3)解码系数，(4)执行反变换，(5)执行反量化，(6)获得块的预测，以及(7)将预测和误差块相加。

[0677] C. 第二组合实现中的序列和语义

[0678] 在第二组合实现中，压缩的视频序列是由结构化成分等级的层中的数据构成的。从上到下的层是：图像层，宏块层和块层。序列层先于序列。图57A至57C示出构成各种层的比特流元素。

[0679] 1. 序列层句法和语义

[0680] 序列级头部包含序列级参数，用于解码压缩图像的序列。使这个头部可用于解码器，或者作为外部传送的解码器配置信息，或者作为视频数据比特流的一部分。图57A是序列层比特流的句法图，它示出构成序列层的元素。剪辑简档PROFILE(5701)元素指定用于产生剪辑的编码简档。如果PROFILE是“高级”简档，则剪辑级LEVEL(5702)元素指定剪辑的编码等级。可供替换地（例如，对于其它简档），通过外部手段将剪辑等级发送到解码器。

[0681] INTERLACE(5703)元素是1比特字段，如果PROFILE是高级简档，则它 存在。INTERFACE(5703)指定在逐行扫描还是在隔行扫描模式中编码视频。如果INTERLACE = 0，则视频帧是在逐行扫描模式中编码的。如果INTERLACE = 1，则视频帧是在隔行扫描模式中编码的。如果PROFILE(5701)不是高级简档，则视频是在逐行扫描模式中编码的。

[0682] 扩展的运动矢量EXTENDED_MV(5704)元素是1比特字段，它指示扩展的运动矢量能力是开还是关。如果EXTENDED_MV = 1，则运动矢量具有扩展的范围。如果EXTENDED_MV = 0，则运动矢量没有扩展的范围。

[0683] 2. 图像层句法和语义

[0684] 图像的数据由图像头部和跟随其后的宏块层数据构成。图57B是图像层比特流的句法图，它示出构成隔行扫描P半帧的图像层的元素。

[0685] 图像类型PTYPE(5722)元素是1比特字段，或者是可变大小字段。如果没有B图像，则在序列中只有I和P图像，并且PTYPE是用单个比特编码的。如果PTYPE = 0，则图像类型是I。如果PTYPE = 1，则图像类型是P。如果B图像的数量大于0，则PTYPE(5722)是可变大小字段，指示帧的图像类型。如果PTYPE = 1，则图像类型是P。如果PTYPE = 01(二进制)，则图像类型是I。而且，如果PTYPE = 00(二进制)，则图像类型是B。

[0686] 参考图像的数量NUMREF(5731)元素是存在于隔行扫描P半帧头部中的1比特句法元素。它指示隔行扫描P半帧具有1个(NUMREF = 0)还是具有2个(NUMREF = 1)参考图像。参考半帧图像指示符REFFIELD(5732)是如果NUMREF = 0则存在于隔行扫描P半帧头部中的1比特句法元素。它指示两个可能的参考图像中哪一个是隔行扫描P半帧使用的。外部MV范围标志MVRANGE(5733)是可变大小句法元素，它存在于使用特定简档（“主”简档）编码的序列的P图像中，并且其BROADCAST元素设置为1。通常，MVRANGE(5733)指示运动矢量的扩展的范围（即，运动矢量的较长的可能水平和/或垂直位移）。MVRANGE(5733)

用于解码运动矢量差分。

[0687] 运动矢量模式 MVMODE(5735) 元素是可变大小句法元素，它用信号表示四个运动矢量编码模式之一或者一个强度补偿模式。运动矢量编码模式包括三个具有用于运动补偿的不同子象素内插规则的“1MV”模式。1MV 表示图像中每个宏块具有至少一个运动矢量。在“混合 MV”模式中，在图像中的每个宏块可具有一个或四个运动矢量，或者可被跳过。取决于 PQUANT（图像的量化系数）的值，图 47E 中所示的任意一个表用于 MVMODE(5735) 元素。

[0688] 运动矢量模式 2MVMODE2(5736) 元素是可变大小句法元素，如果 MVMODE(5735) 用信号表示强度补偿，则它存在于隔行扫描 P 半帧头部中。前面的表（减去强度补偿的代码）可用于 MVMODE2(5736)。

[0689] 亮度比例 LUMSCALE(5738) 和亮度平移 LUMSHIFT(5739) 元素各自是在强度补偿中使用的 6 比特值。如果 MVMODE(5735) 用信号表示强度补偿，则 LUMSCALE(5738) 和 LUMSHIFT(5739) 存在于隔行扫描 P 半帧头部中。

[0690] 宏块模式表 MBMODETAB(5742) 元素是隔行扫描 P 半帧头部的 2 比特字段。MBMODETAB(5742) 指示四个代码表中哪一个（用 2 比特值指定的表 0 至 3）用于编码 / 解码宏块层中的宏块模式 MBMODE(5761) 句法元素。

[0691] 运动矢量表 MVTAB(5743) 元素是隔行扫描 P 半帧的 2 比特字段。MVTAB(5743) 指示四个代码表中哪一个（用两比特值指定的表 0 至 3）用于编码 / 解码运动矢量数据。

[0692] 4MV 块模式表 4MVBPTAB(5744) 元素是 2 比特值，如果 MVMODE(5735)（或者 MVMODE2(5736)，如果 MVMODE(5735) 设置为强度补偿）指示图像是混合 MV 类型的，则它存在于隔行扫描 P 半帧中。4MVBPTAB(5744) 用信号表示四个代码表中哪一个（用两比特值指定的表 0 至 3）用于编码 / 解码 4MV 宏块中的 4MV 块模式 4MVBP(5764) 字段。

[0693] 隔行扫描帧头部（未示出）具有许多与图 57B 所示的隔行扫描 P 半帧头部相同的元素。这些包括 PTYPE(5722)、MBMODETAB(5742)、MVTAB(5743) 和 4MVBPTAB(5744)，尽管隔行扫描帧的精确句法和语义可不同于隔行扫描 P 半帧。例如，4MVBPTAB 也是一个 2 比特字段，它指示四个代码表中哪一个（两比特值指定的表 0 至 3）用于编码 / 解码 4MV 宏块中的 4MV 块模式 4MVBP 元素。隔行扫描帧头部还包括不同的元素，用于在 1MV 与 4MV 模式之间切换和用于强度补偿信号表示。

[0694] 由于隔行扫描帧可包括具有每宏块两个运动矢量的半帧编码宏块，因此隔行扫描帧头部包括两个运动矢量块模式表 2MVBPTAB 元素。2MVBPTAB 是隔行扫描 P 帧中存在的 2 比特字段。该句法元素用信号表示四个表之一（用两比特值指定的表 0 至 3）用于 2MV 半帧编码宏块中的编码 / 解码 2MV 块模式 (2MVBP) 元素。图 47K 示出可用于 2MVBP 的四个表。

[0695] 隔行扫描 B 半帧和隔行扫描 B 帧具有许多与隔行扫描 P 半帧和隔行扫描 P 帧 相同的元素。具体地，隔行扫描 B 帧包括 2MVBPTAB 和 4MVBPTAB(5721) 句法元素，尽管元素的语义可以与隔行扫描 P 半帧和帧不同。

[0696] 3. 宏块层句法和语义

[0697] 宏块的数据由宏块头部和跟随其后的块层组成。图 57C 是宏块层比特流的句法图，它示出构成隔行扫描 P 半帧的宏块的宏块层的元素。

[0698] 宏块模式 MBMODE(5761) 元素是可变大小元素。它联合地指示信息，诸如宏块

(1MV、4MV 或帧内编码）的运动矢量数量、对该宏块是否存在已编码块模式 CBPCY(5762) 以及（在某些情况下）对该宏块是否存在运动矢量差分数据。

[0699] 运动矢量数据 MVDATA(5763) 元素是可变大小元素，它编码宏块的运动矢量的运动矢量信息（例如，水平和垂直差分）。对于具有两个参考半帧的隔行扫描 P 半帧，MVDATA(5763) 还编码用于在运动矢量的主和非主运动矢量预测值之间进行选择的信息。

[0700] 如果 MBMODE(5761) 指示宏块具有四个运动矢量，则四个运动矢量块模式 4MVBP(5764) 元素存在。4MVBP(5764) 元素指示四个亮度块中哪些包含非零运动矢量差分。使用一个代码表将 4MVBP(5764) 元素解码成 0 与 14 之间的值。当表示为二进制值时，这个解码的值表示指示相应亮度块的运动矢量是否存在的 1 比特字段，如图 34 所示。

[0701] 两个运动矢量块模式 2MVBP 元素（未示出）是存在于隔行扫描 P 帧的宏块中的可变大小句法元素。在隔行扫描帧宏块中，如果 MBMODE(5761) 指示宏块具有 2 个半帧运动矢量，则 2MVBP 存在。在这种情况下，2MVBP 指示 2 个半帧（上和下）中哪一个包含非零运动矢量差分。

[0702] 块级运动矢量数据 BLKMVDATA(5765) 元素是在某些情况下存在的可变大小元素。它包含宏块的块的运动信息。

[0703] 混合运动矢量预测 HYBRIDPRED(5766) 元素是每运动矢量 1 比特句法元素，它可存在于隔行扫描 P 半帧的宏块中。当使用混合运动矢量预测时，HYBRIDPRED(5766) 指示要使用两个运动矢量预测值中的哪一个。

[0704] 4. 块层句法和语义

[0705] 隔行扫描图像的块层遵循逐行扫描图像的块层的句法和语义。通常，块和子块的 DC 和 AC 系数的信息是在块层上用信号表示的。

[0706] D. 在第二组合实现中解码

[0707] 下面的章节集中于隔行扫描 P 半帧的解码过程。

[0708] 1. 隔行扫描 P 半帧解码的参考

[0709] 隔行扫描 P 半帧可以在运动补偿中参考一个或两个先前解码的半帧。在图像层中的 NUMREF(5731) 半帧指示当前半帧可以参考一个还是两个先前的参考半帧图像。如果 NUMREF ＝ 0，则当前隔行扫描 P 半帧可以只参考一个半帧。在这种情况下，REFFIELD(5732) 元素跟随在图像层比特流中并且指示哪一个半帧用作参考。如果 REFFIELD ＝ 0，则时间上最近（在显示顺序中）的 I 或 P 半帧用作参考。如果 REFFIELD ＝ 1，则时间上第二最近的 I 或 P 半帧图像用作参考。如果 NUMREF ＝ 1，则当前隔行扫描 P 半帧图像使用两个时间上最近（在显示顺序中）的 I 或 P 半帧图像作为参考。如上所述，图 24A-24F 所示的 NUMREF ＝ 0 和 NUMREF ＝ 1 的参考半帧图像的示例应用于第二组合实现。

[0710] 2. 图像类型和图像层表选择

[0711] 隔行扫描 P 半帧可以是两种类型之一：1MV 或混合 MV。在 1MVP 半帧中，对于 1MV 宏块，使用单个运动矢量来指示宏块中全部 6 个块的预测块的位移。在混合 MVP 半帧中，宏块可以被编码为 1MV 或 4MV 宏块。对于 4MV 宏块，四个亮度块的每一个可具有与它相关联的运动矢量。1MV 模式或混合 MV 模式是由 MVMODE(5735) 和 MVMODE2(5736) 图像层块用信号表示的。

[0712] 对于隔行扫描 P 半帧，图像层包含控制半帧的运动补偿模式和强度补偿的句法元

素。MVMODE(5735) 用信号表示下面任一个 :1) 半帧的四个运动矢量模式之一,或者 2) 在半帧中使用强度补偿。如果用信号表示强度补偿,则在图像层中接着有 MVMODE2(5736)、LUMSCALE(5738) 和 LUMSHIFT(5739) 字段。图 47E 中的两个表之一用于解码 MVMODE(5735) 和 MVMODE2(5736) 字段,取决于 PQUANT 是否大于 12。

[0713] 如果运动矢量模式是混合 MV 模式,则 MBMODETAB(5742) 用信号表示四个混合 MV MBMODE 表中哪一个用于用信号表示半帧中每个宏块的模式。如果运动矢量模式不是混合 MV(在这种情况下全部帧间编码的宏块使用 1 个运动矢量),则 MBMODETAB(5742) 用信号表示四个 1MV MBMODE 表中哪一个用于用信 号表示半帧中每个宏块的模式。

[0714] MVTAB(5742) 指示用于解码隔行扫描 P 半帧中宏块的运动矢量差分的代码表。4MVBPTAB(5744) 指示用于解码隔行扫描 P 半帧中 4MV 宏块的 4MVBP(5764) 的代码表。

[0715] 3. 宏块模式和运动矢量块模式

[0716] 在隔行扫描 P- 半帧中的宏块可以是 3 种可能类型之一 :1MV,4MV 和帧内编码。宏块类型是由宏块层中的 MBMODE(5761) 用信号表示的。

[0717] 1MV 宏块可以在 1MV 和混合 MVP 半帧中出现。在 1MV 宏块中,单个运动矢量表示宏块的全部 6 个块的当前与参考图像之间的位移。当前与参考块之间的差异在块层中编码。对于 1MV 宏块,MBMODE(5761) 指示以下三项 :(1) 宏块类型是 1MV ;(2)CBPCY(5762) 是否存在 ;以及 (3)MVDATA(5763) 是否存在。

[0718] 如果 MBMODE(5761) 指示 CBPCY(5762) 存在,则 CBPCY(5762) 存在于宏块层并且指示 6 个块中哪些编码在块层中。如果 MBMODE(5761) 指示 CBPCY(5762) 不存在,则 CBPCY(5762) 假定等于 0,并且宏块中的 6 个块中任何一个都不存在块数据。

[0719] 如果 MBMODE(5761) 指示 MVDATA(5763) 存在,则 MVDATA(5763) 存在于宏块层中并且编码运动矢量差分,它与运动矢量预测值组合来重构运动矢量。如果 MBMODE(5761) 指示 MVDATA(5763) 不存在,则运动矢量差分假定为零,并且因此运动矢量等于运动矢量预测值。

[0720] 4MV 宏块只出现在混合 MVP 半帧中。在 4MV 宏块中,宏块中四个亮度块的每一个可具有一个相关联的运动矢量,它指示该块的当前与参考图像之间的位移。色度块的位移是从四个亮度运动矢量导出的。当前与参考块之间的差异在块层中编码。对于 4MV 宏块,MBMODE(5761) 指示以下三项 :(1) 宏块类型是 4MV ;(2)CBPCY(5762) 是否存在 ;以及 (3)4MVBP(5764) 是否存在。

[0721] 如果 MBMODE(5761) 指示 4MVBP(5764) 存在,则 4MVBP(5764) 存在于宏块层中并且指示四个亮度块中的哪些包含非零运动矢量差分。4MVBP(5764) 被解码成 0 与 14 之间的值,它在表示为二进制值时表示 1 比特字段,它指示相应亮度块的运动矢量数据是否存在,如图 27 所示。对于 4MVBP(5764) 中 4 比特位置的每一个,值 0 指示该块不存在运动矢量差分 (BLKMVDATA (5765)),并且运动矢量差分假定为 0。值 1 指示该块存在运动矢量差分 (BLKMVDATA(5765))。如果 MBMODE(5761) 指示 4MVBP(5764) 不存在,则假定全部四个亮度块存在运动矢量差分数据 (BLKMVDATA(5765))。

[0722] 在隔行扫描帧中的半帧编码宏块可包括 2 个运动矢量。在 2 个半帧 MV 宏块的情况下,2MVBP 元素指示两个半帧中哪些具有非零差分运动矢量。

[0723] 帧内编码宏块可以出现在 1MV 或混合 MVP 半帧中。在帧内宏块中,在不参考任何先前图像数据的情况下,编码全部六个块。当前块象素与常数值 128 之间的差异在块层中编

码。对于帧内编码宏块，MBMODE(5761) 指示以下两项：(1) 宏块类型是帧内编码；以及 (2) CBPCY(5762) 是否存在。对于帧内编码宏块，CBPCY(5762) 当存在时，指示六个块中的哪一个具有在块层中编码的 AC 系数数据。

[0724] 4. 解码运动矢量差分

[0725] MVDATA(5763) 和 BLKDATA(5765) 字段编码宏块或宏块中的块的运动信息。1MV 宏块具有单个 MVDATA(5763) 字段，而 4MV 宏块可以具有零到四个 BLKMVDATA(5765)。计算运动矢量差分对于一个参考 (NUMREF = 0) 情况和两个参考 (NUMREF = 1) 情况是不同地进行的。

[0726] 在只有一个参考半帧的半帧图像中，宏块层中的每个 MVDATA(5763) 或 BLKMVDATA(5765) 字段联合地编码以下两项：(1) 水平运动矢量差分分量；以及 (2) 垂直运动矢量差分分量。MVDATA(5763) 或 BLKMVDATA(5765) 字段是一个哈夫曼 VLC，后面跟着一个 FLC。VLC 的值确定 FLC 的大小。图像层中的 MVTAB(5743) 字段指定用于解码该 VLC 的表。

[0727] 图 58A 示出伪代码，它示出具有一个参考半帧的半帧图像中的块或宏块的运动矢量的运动矢量差分解码。在该伪代码中，计算值 dmv_x 和 dmv_y。值 dmv_x 是差分水平运动矢量分量，而值 dmv_y 是差分垂直运动矢量分量。变量 k_x 和 k_y 是用于长运动矢量的固定长度值，并且取决于如由 MVRANGE(5733) 定义的运动矢量范围，如在图 58B 的表中所示。值 halfpel_flag 是二进制值，指示使用二分之一象素还是四分之一象素精度用于图像的运动补偿。halfpel_flag 的值是由运动矢量模式确定的。如果模式是 1MV 或混合 MV，则 halfpel_flag = 0 和四分之一象素精度用于运动补偿。如果模式是 1MV 象素或 1MV 二分之一象素双线性，则 halfpel_flag = 1 并且使用二分之一象素精度。offset_table 是如下定义的数组：

[0728] offset_table[9] = {0,1,2,4,8,16,32,64,128}.

[0729] 在具有两个参考半帧的半帧图像中，宏块层中的每个 MVDATA(5763) 或 BLKMVDATA(5765) 字段联合地编码以下三项：(1) 水平运动矢量差分分量；(2) 垂直运动矢量差分分量；以及 (3) 使用主还是非主运动矢量预测值。MVDATA(5763) 或 BLKMVDATA(5765) 字段是一个哈夫曼 VLC，后面跟着一个 FLC，并且 VLC 的值确定 FLC 的大小。MVTAB(5743) 字段指定用于解码 VLC 的表。

[0730] 图 59 示出伪代码，它示出具有两个参考半帧的半帧图像中块或宏块的运动矢量的运动矢量差分和主 / 非主预测值解码。在该伪代码中，值 predictor_flag 是二进制标志，它指示使用主还是非主运动矢量预测值 (0 = 使用主预测值，1 = 使用非主预测值)。各种其它变量 (包括 dmv_x，dmv_y，k_x，k_y，halfpel_flag 和 offset_table[] 如对一个参考半帧情况一样地描述。表 size_table 是如下定义的数组：

[0731] size-table[14] = {10,0,1,1,2,2,3,3,4,4,5,5,6,6}.

[0732] 5. 运动矢量预测值

[0733] 运动矢量是通过将在前面章节中计算出的运动矢量差分加到运动矢量预测值来计算的。预测值是从最多三个相邻运动矢量计算出的。

[0734] 在 1MV 隔行扫描 P 半帧中，最多三个运动矢量用于计算当前宏块的预测值。相邻预测值 A、B 和 C 的位置在图 5A 和 5B 中示出。如对逐行扫描 P 帧所述，相邻预测值是从左、

上和右上宏块取得的，除了在当前宏块是行中最后一个宏块的情况下。在这种情况下，预测值 B 是从左上（代替右上）宏块取得的。对于帧是一个宏块宽的特殊情况，则预测值总是预测值 A（上预测值）。

[0735] 在混合 MV 隔行扫描 P 半帧中，最多三个运动矢量用于计算当前块或宏块的预测值。图 6A-10 示出混合 MVP 半帧中 1MV 和 4MV 宏块的三个候选运动矢量，如对逐行扫描 P 帧描述的。对于帧是一个宏块宽的特殊情况，则预测值总是预测值 A（上预测值）。

[0736] 如果图像头部中的 NUMREF(5731) 字段是 0，则当前隔行扫描 P 半帧可以只参考一个先前编码的图像。如果 NUMREF ＝ 1，则当前隔行扫描 P 半帧可以参考两个最近的参考半帧图像。在前一情况下，为每个运动矢量计算单个预测值。在后一种情况下，计算两个运动矢量预测值。图 60A 和 60B 中的伪代码示出如何为一个参考半帧情况计算运动矢量预测值。变量 fieldpred_x 和 fieldpred_y 表示运动矢量预测值的水平和垂直分量。

[0737] 在 2 参考半帧隔行扫描 P 半帧 (NUMREF ＝ 1) 中，当前半帧可以参考两个最近的参考半帧。在这种情况下，为每个帧间编码宏块计算两个运动矢量预测值。一个预测值来自相同极性的参考半帧，而另一个来自相反极性的参考半帧。

[0738] 图 61A-61F 中的伪代码描述如何在给出 3 个候选运动矢量预测值的情况下为两个参考半帧情况计算运动矢量预测值。变量 samefiledpred_x 和 samefieldpred_y 表示来自相同半帧的运动矢量预测值的水平和垂直分量，而变量 oppositefieldpred_x 和 oppositefieldpred_y 表示来自相反半帧的运动矢量预测值的水平和垂直分量。变量 dominantpredictor 指示哪一个半帧包含主预测值。值 predictor_flag（从运动矢量差分解码的）指示使用主还是非主预测值。

[0739] 6. 混合运动矢量预测

[0740] 如果隔行扫描 P 半帧是 1MV 或混合 MV，则相对于 A（上）和 C（左）预测值测试在前面章节中计算的运动矢量预测值来确定在比特流中是否明确编码了预测值。如果是，则存在指示是使用预测值 A 还是预测值 C 作为运动矢量预测值的一个比特。图 14A 和 14B 中的伪代码示出混合运动矢量预测解码，使用如下变量 :变量 predictor_pre_x 和 predictor_pre_y 和候选预测值 A、B 和 C 是如在前面章节中计算的（即，它们是相反半帧预测值或者它们是相同的半帧预测值，如由 predictor_flag 指示的）。变量 predictor_post_x 和 predictor_post_y 分别是在检查混合运动矢量预测之后的水平和垂直运动矢量预测值。

[0741] 7. 重构运动矢量

[0742] 对于 1MV 和 4MV 宏块两者，亮度运动矢量是通过将差分加到预测值来重构的，如下 :

[0743] my_x ＝ (dmv_x+predictor_x)-smod range_x，以及

[0744] mv_y ＝ (dmv_y+predictor_y)smod range_y，

[0745] 其中变量 range_x 和 range_y 取决于 MVRANGE(5733)，并且在图 58B 所示的表中指定的，并且其中运算“smod”是如下定义的有符号模。

[0746] A smod b ＝ ((A+b)％ 2b)-b，

[0747] 其中保证重构的矢量是有效的。(A smod b) 位于 b 与 b-1 内 .

[0748] 在 1MV 宏块中，对用于构成宏块的亮度分量的四个块将存在单个运动矢量。如果 dmv_x 指示宏块是帧内编码的，则没有运动矢量与宏块相关联。如果宏块是 跳过的，则

dmv_x = 0 且 dmv_y = 0，因此 mv_x = predictor_x 且 mv_y = predictor_y。

[0749] 在 4MV 宏块中，宏块中帧间编码的亮度块的每一个具有它自己的运动矢量。因此，将有 0 至 4 个亮度运动矢量用于每个 4MV 宏块。4MV 宏块中未编码的块可以用下面两种方法之一出现：(1) 如果宏块是跳过的并且宏块是 4MV（在这种情况下宏块中的全部块是跳过的）；或者 (2) 如果宏块的 CBPCY(5762) 指示该块是未编码的。如果块没有被编码，则 dmv_x = 0 且 dmv_y = 0，因此 mv_x = predictor_x 且 mv_y = predicotr_y。

[0750] 8. 导出色度运动矢量

[0751] 色度运动矢量是从亮度运动矢量导出的。而且，对于 4MV 宏块，将色度块编码为帧间编码还是帧内编码的判定是基于亮度块的状态作出的。色度运动矢量是在两个步骤中重构的。作为第一步骤，通过适当地组合和比例缩放亮度运动矢量获得名义的色度运动矢量。比例缩放是以这样一种方法进行的，即与四分之一象素偏移相比，二分之一象素是较佳的。在第二阶段中，序列级 1 比特字段 FASTUVMC 字段用于确定色度运动矢量的进一步舍入是否必须。如果 FASTUVMC = 0，在第二阶段中不进行舍入。如果 FASTUVMC = 1，在四分之一象素偏移处的色度运动矢量将舍入到最接近的整象素位置。另外，当 FASTUVMC = 1 时，将只使用双线性滤波用于全部色度内插。

[0752] 在 1MV 宏块中，色度运动矢量是从亮度运动矢量中导出的，如下：

[0753] //s_RndTb1[0] = 0，s_RndTb1[1] = 0，s_RndTb1[2] = 0，s_RndTb1[3] = 1

[0754] cmv_x = (lmv_x+s_RndTb1[lmv_x&3]) >> 1

[0755] cmv_y = (lmv_y+s_RndTb1[lmv_y&3]) >> 1

[0756] 图 16B 中的伪代码出第一阶段，即使用如下的变量，色度运动矢量是如何从 4MV 宏块中四个亮度块的运动信息中导出的。确定 4MV 宏块的最多四个亮度运动矢量中的主极性，并且色度运动矢量是从具有主极性的亮度运动矢量确定的，（但不是从其它极性的亮度运动矢量）。

[0757] 9. 强度补偿

[0758] 如果强度补偿用于参考半帧，则在使用参考帧中的象素作为预测值之前重新映射它们。当使用强度补偿时，LUMSCALE(5738) 和 LUMSHIFT(5739) 存在于图像比特流中。图 18 或 56 中的伪代码示出 LUMSCALE(5738) 和 LUMSHIFT (5739) 用于重新映射参考半帧象素。使用 LUTY[] 表重新映射参考的 Y 分量，并且使用 LUTUV[] 表重新映射 U 和 V 分量，如下：

[0759] $\overline{p}_Y = LUTY[p_Y]$，以及

[0760] $\overline{p}_{UV} = LUTUV[p_{UV}]$，

[0761] 其中 p_Y 是参考半帧中的原始亮度象素值，$\overline{p}_Y$ 是参考半帧中重新映射的亮度象素值，p_{UV} 是参考半帧中原始的 U 或 V 象素值，并且 $\overline{p}_{UV}$ 是参考半帧中重新映射的 U 或 V 象素值。

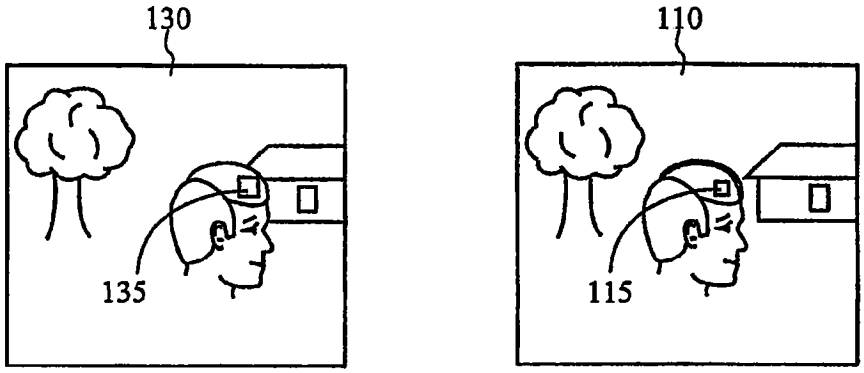
[0762] 10. 剩余的解码

[0763] 当 CBPCY(5762) 元素存在时，解码器解码宏块的 CBPCY(5762)，其中 CBPCY(5762) 元素指示系数数据的存在 / 不存在。在块层上，解码器解码帧间编码的块和帧内编码的块的系数数据。为重构帧间编码的块，解码器：(1) 选择变换类型 (8x8，8x4，4x8 或 4x4)，(2) 解码子块模式，(3) 解码系数，(4) 执行反变换，(5) 执行反量化，(6) 获得块的预测，和 (7)

将预测和误差块相加。

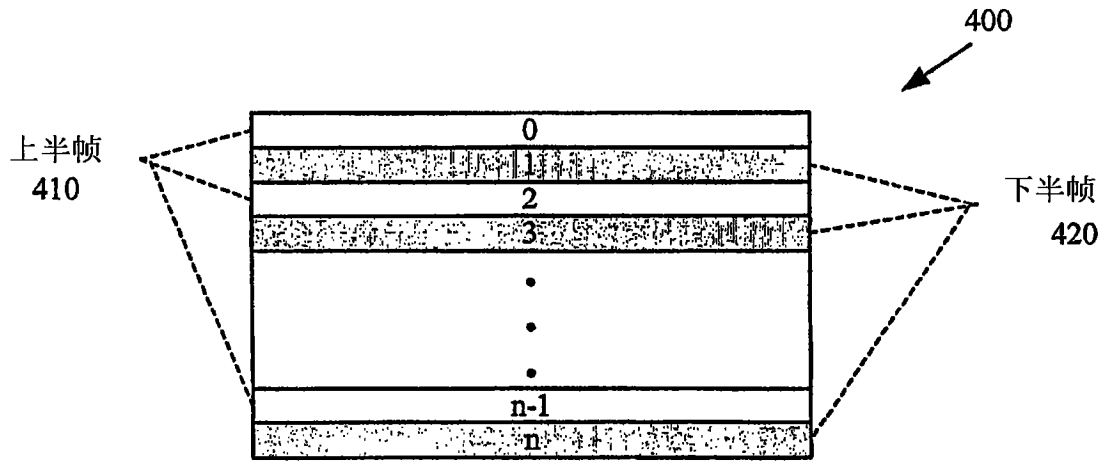
[0764] 已经参考各种实施例描述和示出了本发明的原理，将认识到各种实施例可以在不脱离这样的原理的情况下在排列和细节方面进行修改。应该理解，在此描述的程序、过程或方法与任何特定类型的计算环境相关或受限于它，除非另外指示。可在各种类型的通用或专用的计算环境下使用或者按照在此描述的教导执行操作。在软件中所示的实施例元素可用硬件实现，反之亦然。

[0765] 鉴于许多可应用本发明原理的可能实施例，因此要求保护落在所附的权利要求书及其等价技术方案的范围和精神内的所有这样的实施例作为本发明。



现有技术

图 1



现有技术

图 4

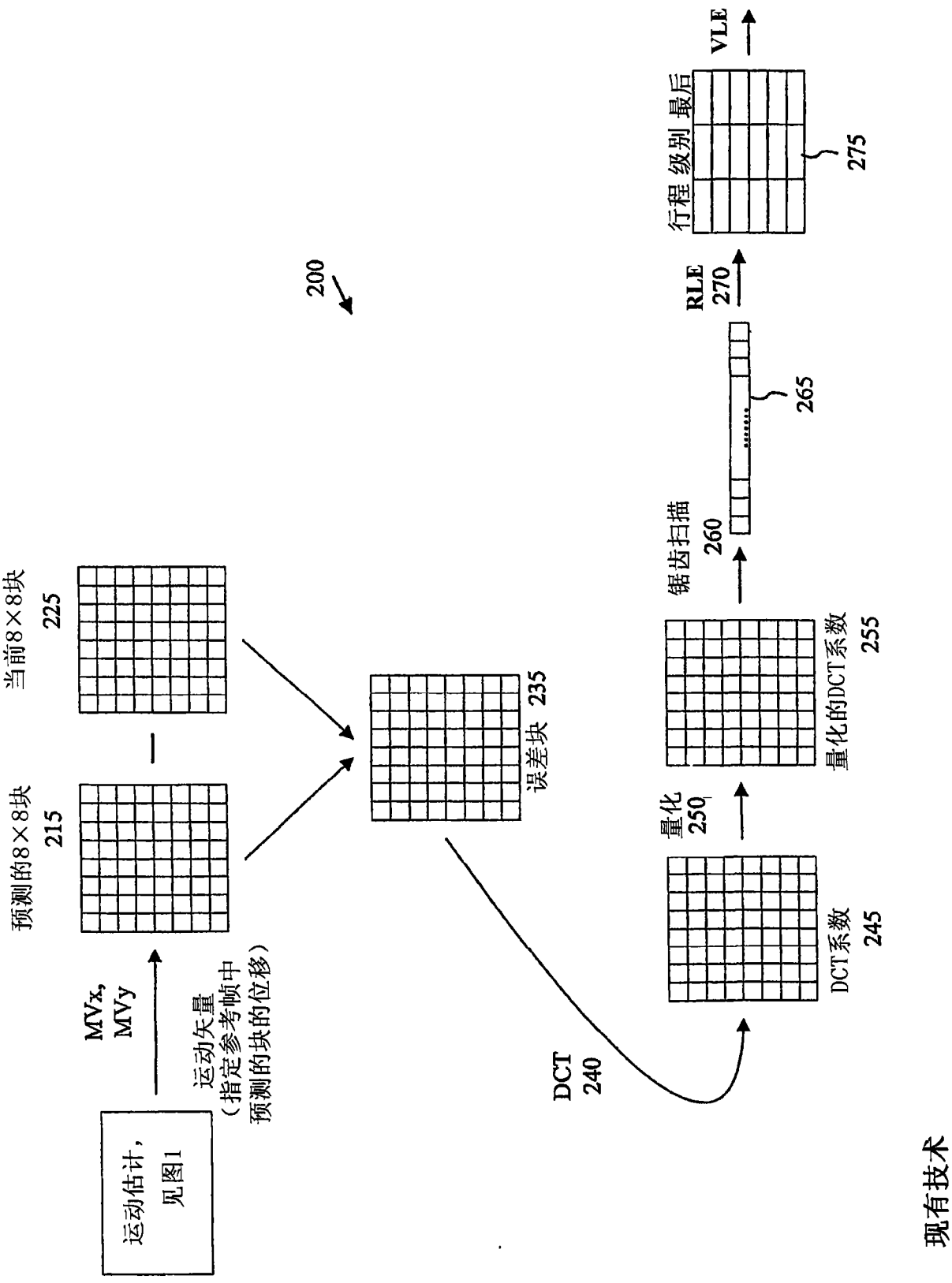

200
运动估计，
见图1
MVx,
MVy
运动矢量
（指定参考帧中
预测的块的位移）
预测的8×8块
215
当前8×8块
225
误差块 235
DCT
240
DCT系数
245
量化
250
量化的DCT系数
255
锯齿扫描
260
265
RLE
270
行程 级别 最后
275
VLE
现有技术

图 2

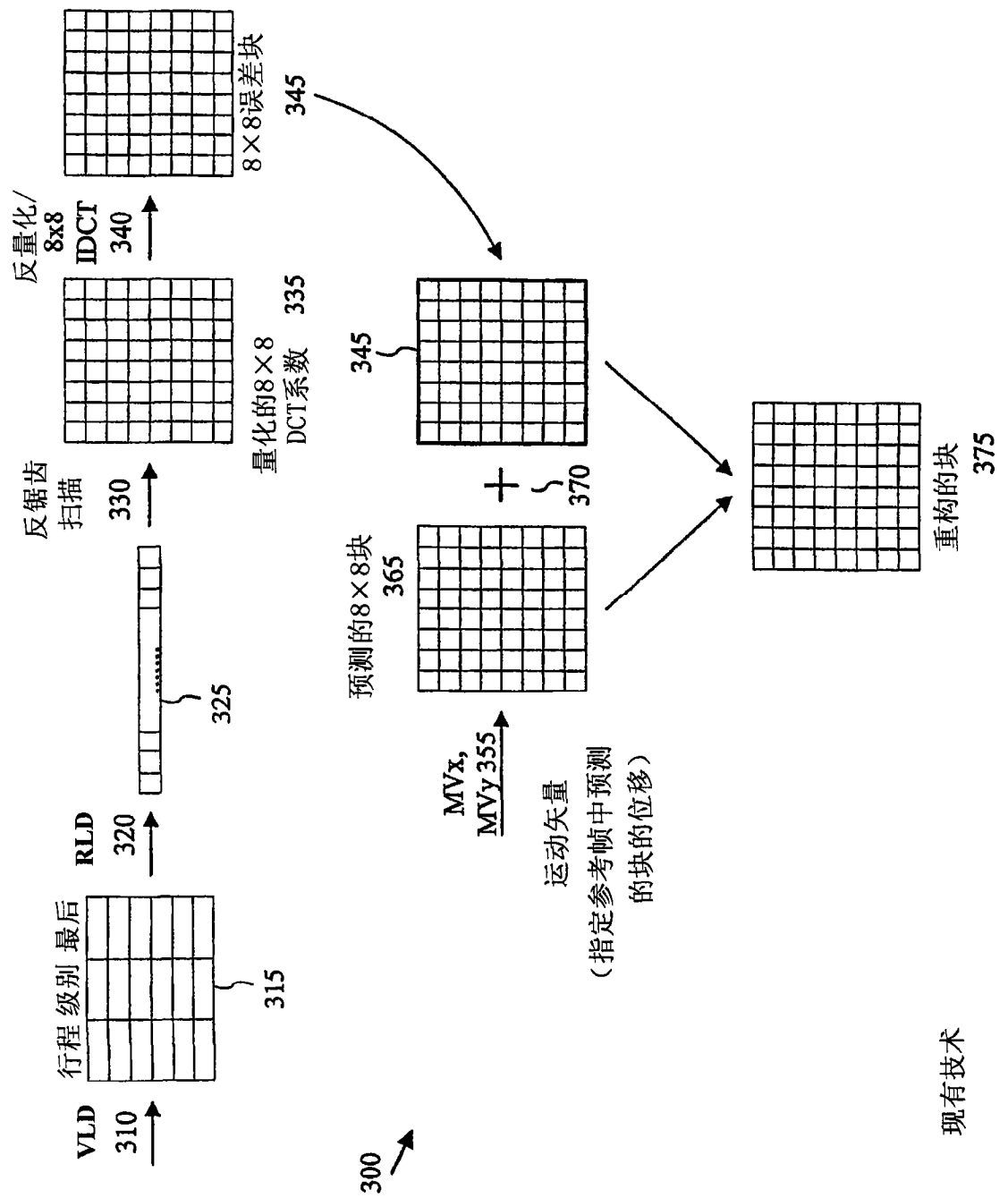
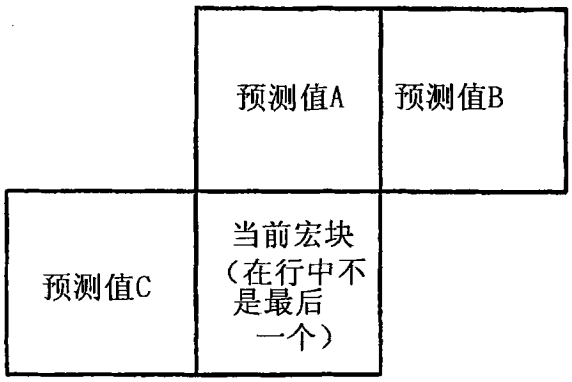
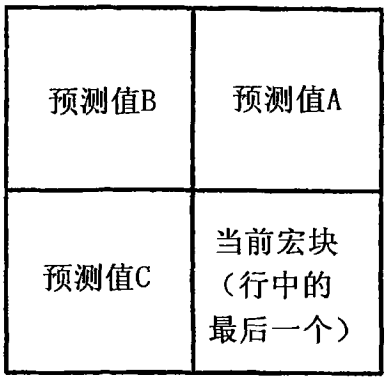

300
VLD
310
行程 级别 最后
315
RLD
320
325
反锯齿
扫描
330
量化的8×8
DCT系数
335
反量化/
8x8
IDCT
340
8×8误差块
345
345
预测的8×8块
365
+
370
MVx,
MVy 355
运动矢量
（指定参考帧中预测
的块的位移）
重构的块
375
现有技术

图 3



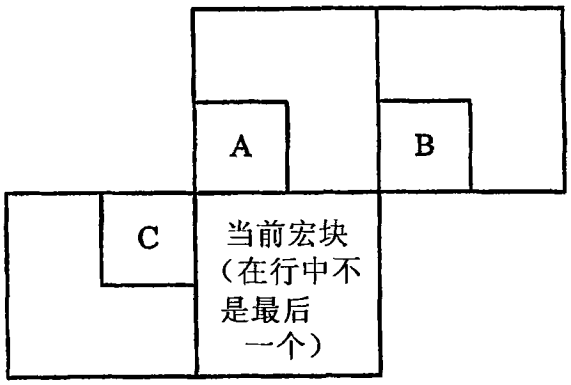
现有技术

图 5A



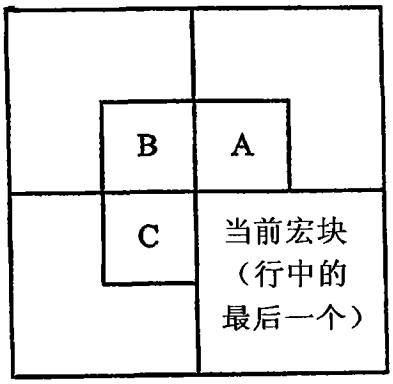
现有技术

图 5B



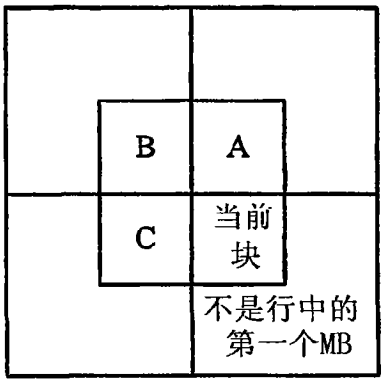
现有技术

图 6A



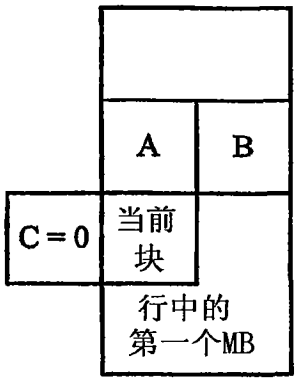
现有技术

图 6B



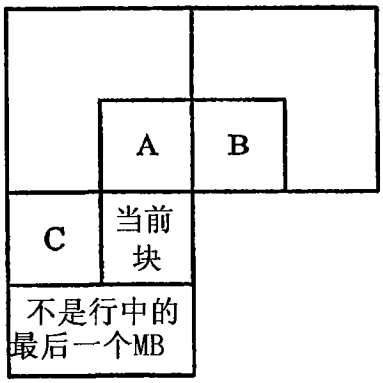
现有技术

图 7A



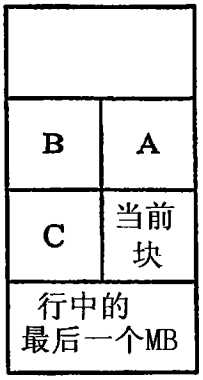
现有技术

图 7B



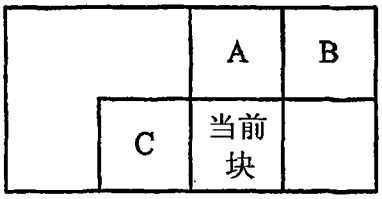
现有技术

图 8A



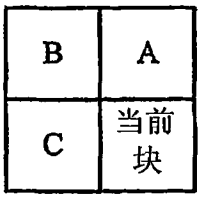
现有技术

图 8B



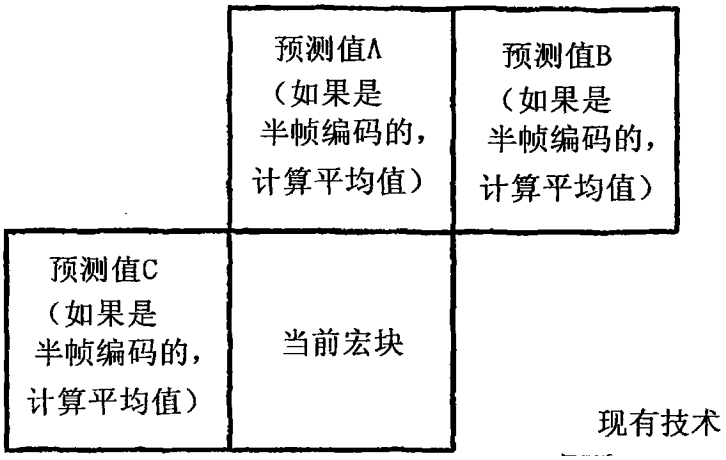
现有技术

图 9



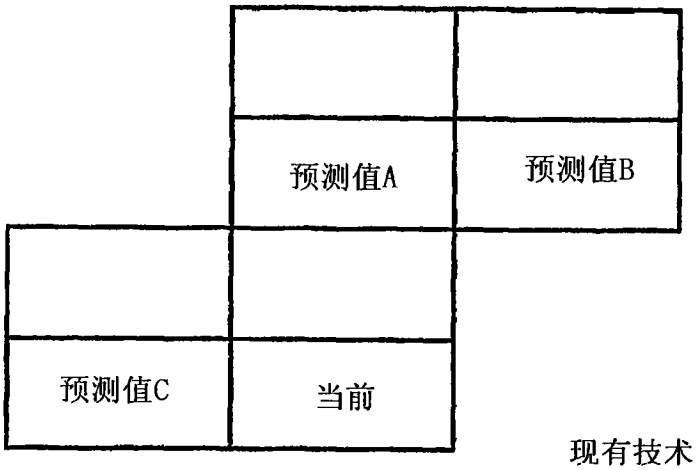
现有技术

图 10



现有技术

图 11



现有技术

图 12A

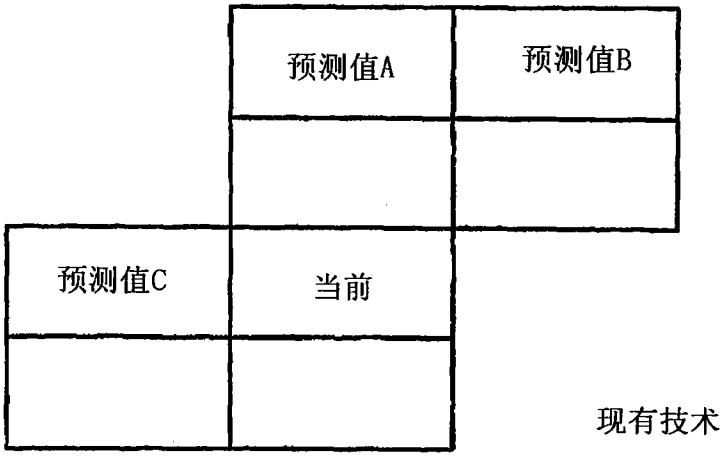


图 12B

```
if (预测值A在边界外&&预测值C在边界外)
{
      predictor_x = 0
      predictor_y = 0
}
else
{
      if (预测值C在边界外)
      {
            if (预测值A是帧内编码的)
            {
                  predictor_x = 0
                  predictor_y = 0
            }
            else
            {
                  //使用上预测值
                  predictor_x = predictorA_x
                  predictor_y = predictorA_y
            }
      }
       else if (预测值A在边界外)
      {
            if (预测值C是帧内编码的)
            {
                  predictor_x = 0
                  predictor_y = 0
            }
            else
            {
                  //使用左预测值
                  predictor_x = predictorC_x
                  predictor_y = predictorC_y
            }
      }
      else
      {
            if (预测值B在边界外)
            {
                  //设置predictorB为帧间编码类型并且设置运动矢量为零
                  将预测值B设为帧间编码的
                  predictorB_x = 0
                  predictorB_y = 0
            }                                    在13B继续
```

现有技术

图 13A

从13A继续

```
num_intra = 0
if (预测值A是帧内编码的)
{
      predictorA_x = 0
      predictorA_y = 0
      num_intra = num_intra + 1
}
 if (预测值B是帧内编码的)
{
      predictorB_x = 0
      predictorB_y = 0
      num_intra = num_intra + 1
}
 if (预测值C是帧内编码的)
{
      predictorC_x = 0
      predictorC_y = 0
      num_intra = num_intra + 1
}

if (num_intra > 1)
{
      predictor_x = 0
      predictor_y = 0
}
else
{
      // 根据A、B和C候选预测值计算预测值

      predictor_x = median3(predictorA_x, predictorB_x,
predictorC_x)
      predictor_y = median3(predictorA_y, predictorB_y,
predictorC_y)
 }
   }
}
```

现有技术

图 13B

```
median3 (a, b, c) {
      if (a > b) {
            if (b > c)
                  median = b
            else if (a > c)
                  median = c
            else
                  median = a
      }
      else if (a > c)
            median = a
      else if (b > c)
            median = c
      else median = b
      return median
}
```

现有技术

图 13C

```
median4 (a, b, c, d)
{
      max = min = a
      if (b > max)
            max = b
      else if (b < min)
            min = b
      if (c > max)
            max = c
      else if (c < min)
            min = c
      if (d > max)
            max = d
      else if (d < min)
            min = d
   median =  (a + b + c + d - max - min) / 2
   return median
}
```

现有技术

图 16C

```
if((预测值A在边界外)) || ((预测值C在边界外))
{
    predictor_post_x = predictor_pre_x
    predictor_post_y = predictor_pre_y
}
else
{
    if(预测值A是帧内编码的)
        sum = abs(predictor_pre_x) + abs(predictor_pre_y)
    else
        sum = abs(predictor_pre_x - predictorA_x) + abs(predictor_pre_y - predictorA_y)
    if (sum > 32)
    {
        //读一下个比特以查看哪个候选预测值要使用
        if (get_bits(1) == 0)          //HYBRIDPRED字段
        {
            //HYBRIDPRED字段
            predictor_post_x = predictorA_x
            predictor_post_y = predictorA_y
        }
        else
        {
            //使用左预测值
            predictor_post_x = predictorC_x
            predictor_post_y = predictorC_y
        }
    }
                                    在14B继续
```

现有技术

图 14A

从14A继续

```
else
{
    if(预测值C是帧内编码的)
        sum = abs(predictor_pre_x) + abs(predictor_pre_y)
    else
        sum = abs(predictor_pre_x - predictorC_x) + abs(predictor_pre_y - predictorC_y)
    if (sum > 32)
    {
        //读一下个比特以查看哪个候选预测值要使用
        if (get_bits(1) == 0)
        {
            //使用上预测值
            predictor_post_x = predictorA_x
            predictor_post_y = predictorA_y
        }
        else
        {
            //使用左预测值
            predictor_post_x = predictorC_x
            predictor_post_y = predictorC_y
        }
    }
}
```

现有技术

图 14B

```
index = vlc_decode()      //使用由图像层中MVTAB指示的哈夫曼表

index = index + 1
if (index >= 37)
{
      last_flag = 1
      index = index - 37
}
else
      last_flag = 0

intra_flag = 0
if (index == 0)
{
      dmv_x = 0
      dmv_y = 0
}
else if (index == 35)
{
      dmv_x = get_bits(k_x - halfpel_flag)
      dmv_y = get_bits(k_y - halfpel_flag)
}
else if (index == 36)
{
      intra_flag = 1
      dmv_x = 0
      dmv_y = 0
}
else
{
    index1 = index % 6
    if (halfpel_flag == 1 && index1 == 5)
      hpel = 1
    else
      hpel = 0
   val = get_bits (size_table[index1] - hpel)
   sign = 0 - (val & 1)
   dmv_x = sign ^ ((val >> 1) + offset_table[index1])
   dmv_x = dmv_x - sign
```

在15B继续

现有技术

图 15A

从15A继续

```
    index1 = index / 6
    if (halfpel_flag == 1 && index1 == 5)
      hpel = 1
    else
      hpel = 0
    val = get_bits (size_table[index1] - hpel)
    sign = 0 - (val & 1)
    dmv_y = sign ^ ((val >> 1) + offset_table[index1])
    dmv_y = dmv_y- sign
}
```

现有技术

图 15B

MVRANGE	k_x	k_y	range_x	range_y
0（默认值）	9	8	256	128
10	10	9	512	256
110	12	10	2048	512
111	13	11	4096	1024

现有技术

图 15C

```
// s_RndTbl[0] = 0, s_RndTbl[1] = 0, s_RndTbl[2] = 0, s_RndTbl[3] = 1
cmv_x = (lmv_x + s_RndTbl[1mv_x & 3]) >> 1
cmv_y = (lmv_y + s_RndTbl[1mv_y & 3]) >> 1
```

现有技术

图 16A

```
if(所有4个亮度块是帧间编码的)
{
      // lmv0_x, lmv0_y是块0的运动矢量
      // lmv1_x, lmv1_y是块1的运动矢量
      // lmv2_x, lmv2_y是块2的运动矢量
      // lmv3_x, lmv3_y是块3的运动矢量
      ix = median4(lmv0_x, lmv1_x, lmv2_x, lmv3_x)
      iy = median4(lmv0_y, lmv1_y, lmv2_y, lmv3_y)
}
 else if (亮度块中的3个是帧间编码的)
{
      //lmv0_x, lmv0_y是第一帧间编码块的运动矢量
      //lmv1_x, lmv1_y是第二帧间编码块的运动矢量
      //lmv2_x, lmv2_y是第三帧间编码块的运动矢量
      ix = median3(lmv0_x, lmv1_x, lmv2_x)
      iy = median3(lmv0_y, lmv1_y, lmv2_y)
}
  else if (亮度块中的2个是帧间编码的)
{
      //lmv0_x, lmv0_y是第一帧间编码块的运动矢量
      //lmv1_x, lmv1_y是第二帧间编码块的运动矢量
      ix = (lmv0_x + lmv1_x) / 2
      iy = (lmv0_y + lmv1_y) / 2
}
else
      色度块被编码为帧内编码
// s_RndTbl[0] = 0, s_RndTbl[1] =  0, s_RndTbl[2] = 0, s_RndTbl[3] = 1
cmv_x = (ix + s_RndTbl[ix & 3]) >> 1
cmv_y = (iy + s_RndTbl[iy & 3]) >> 1
```

现有技术

图 16B

```
frac_x4 = (lmv_x << 2) % 16;
int_x4= (lmv_x << 2) - frac_x;

ChromaMvRound [16] = {0, 0, 0, .25, .25, .25, .5, .5, .5, .5, .5,
.75, .75, .75, 1, 1};
cmv_y = lmv_y;
cmv_x = Sign (lmv_x) * (int_x4 >> 2) + ChromaMvRound [frac_x4];
```

现有技术

图 17

```
if (LUMSCALE == 0)
{
      iScale = - 64
      iShift = 255 * 64 + 32 - LUMSHIFT *2 * 64
}
else {
      iScale = LUMSCALE + 32
      if (LUMSHIFT > 31)
            iShift = LUMSHIFT * 64 - 64 * 64;
            else
            iShift = LUMSHIFT * 64;
        }

//建立LUT
for (i = 0; i < 256; i++)
{
      j = (iScale * i + iShift + 32) >> 6
      if (j > 255)
            j = 255
      else if (j < 0)
                  j = 0
      LUTY[i] = j
      j = (iScale * (i - 128) + 128 * 64 + 32) >>6
      if (j > 255)
            j = 255
      else if (j < 0)
            j = 0
      LUTUV[i] =  j
}
```

现有技术

图 18

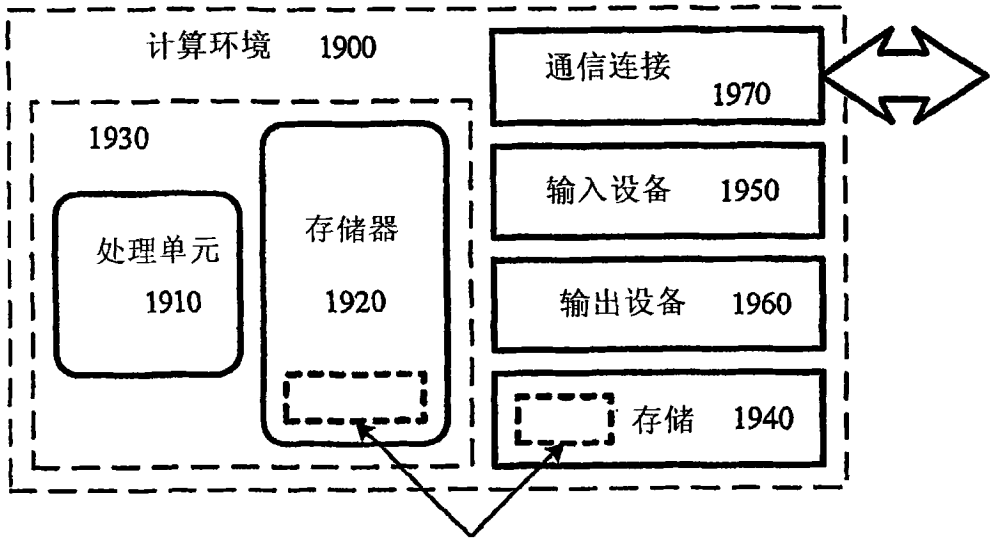

计算环境 1900
1930
处理单元
1910
存储器
1920
通信连接
1970
输入设备 1950
输出设备 1960
存储 1940
实现视频编码器和/或解码器的软件1980

图 19

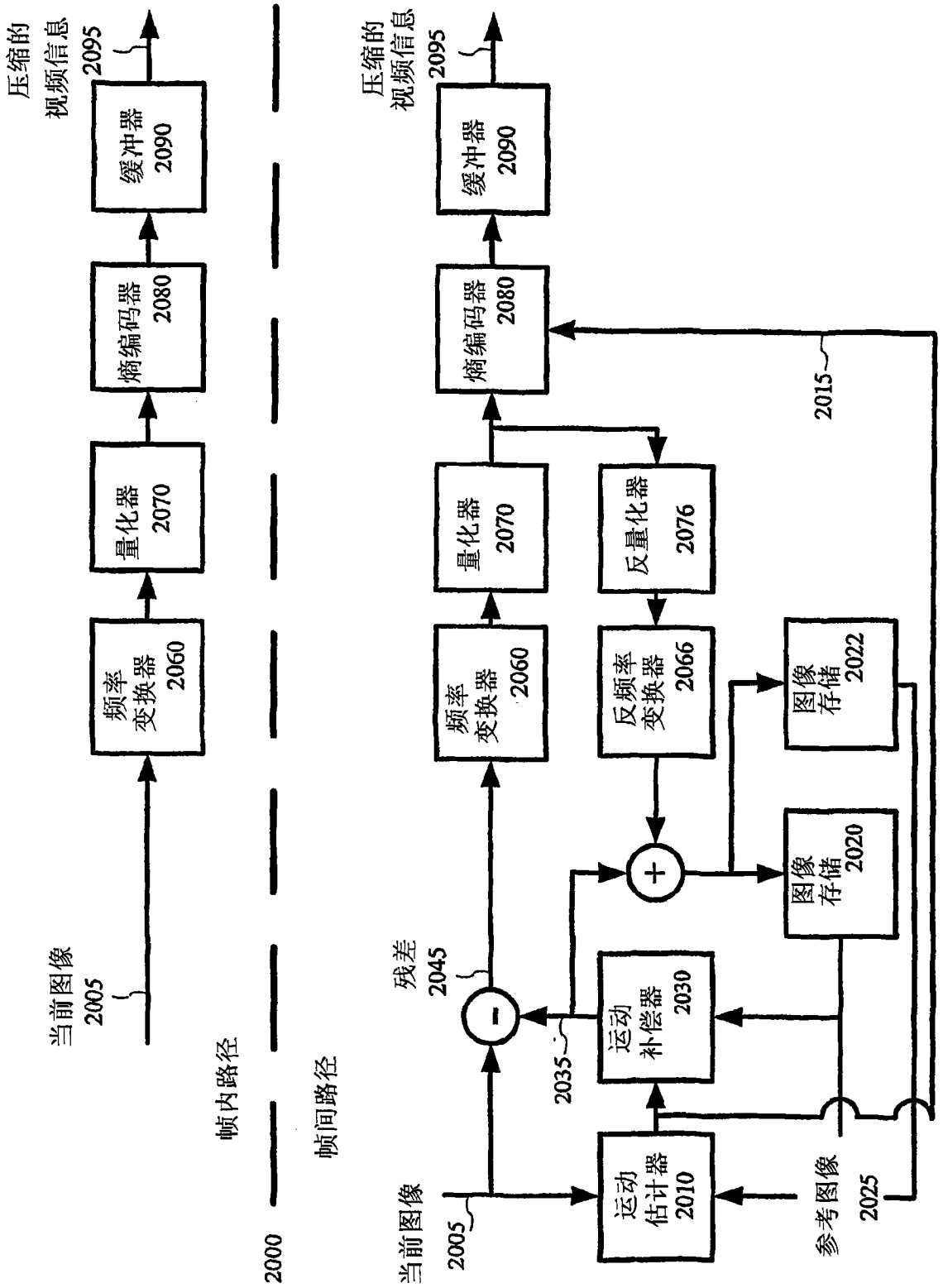

当前图像
2005
频率变换器
2060
量化器
2070
熵编码器
2080
缓冲器
2090
压缩的视频信息
2095
帧内路径
帧间路径
2000
残差
2045
运动估计器
2010
运动补偿器
2030
2035
反频率变换器
2066
反量化器
2076
图像存储
2020
图像存储
2022
参考图像
2025
2015

图 20

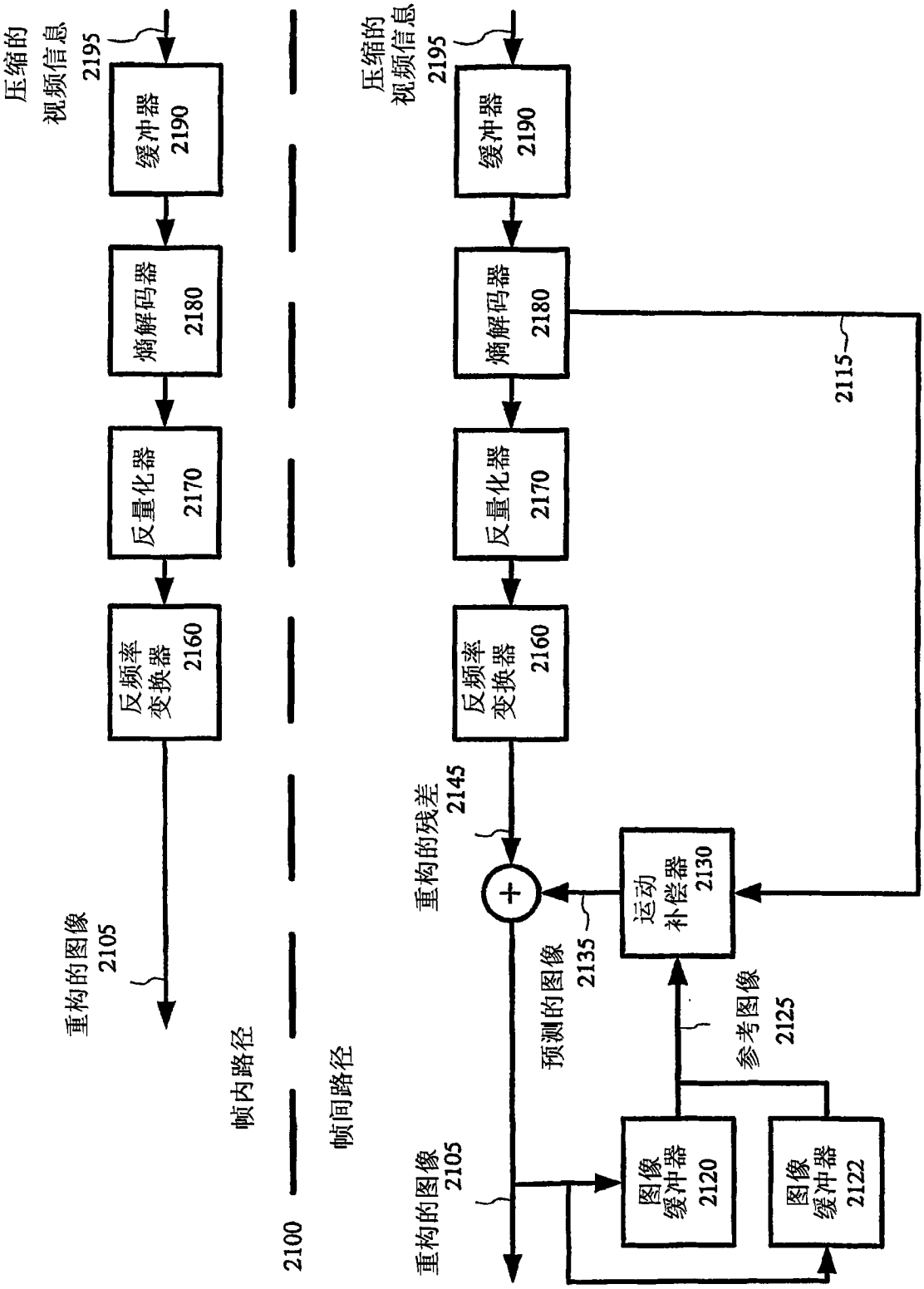

压缩的
视频信息
2195
缓冲器
2190
熵解码器
2180
反量化器
2170
反频率
变换器
2160
重构的图像
2105
帧内路径
帧间路径
2100
压缩的
视频信息
2195
缓冲器
2190
熵解码器
2180
反量化器
2170
反频率
变换器
2160
2115
重构的残差
2145
+
运动
补偿器
2130
预测的图像
2135
参考图像
2125
重构的图像
2105
图像
缓冲器
2120
图像
缓冲器
2122

图 21

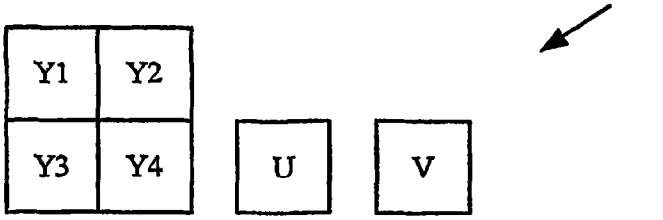


图 22

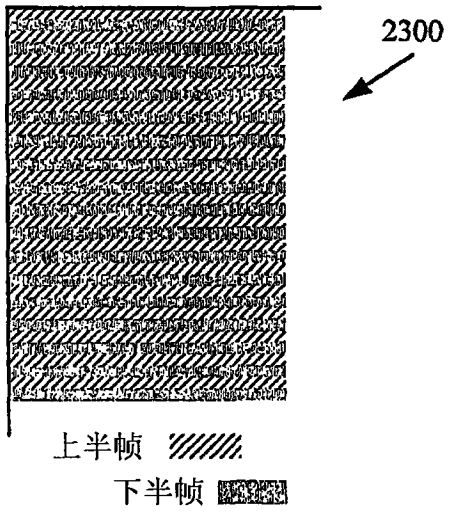


图 23A

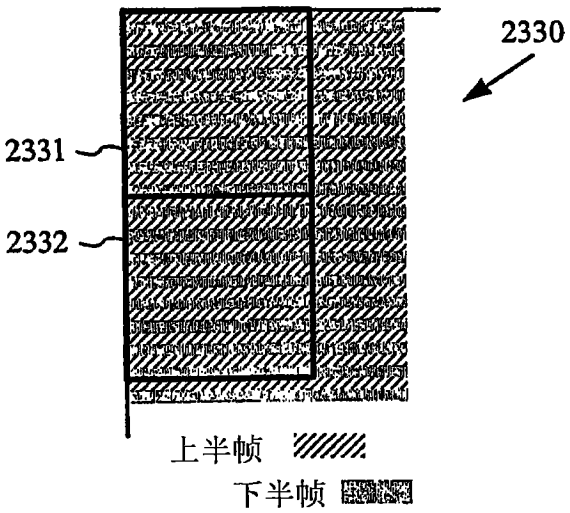


图 23B

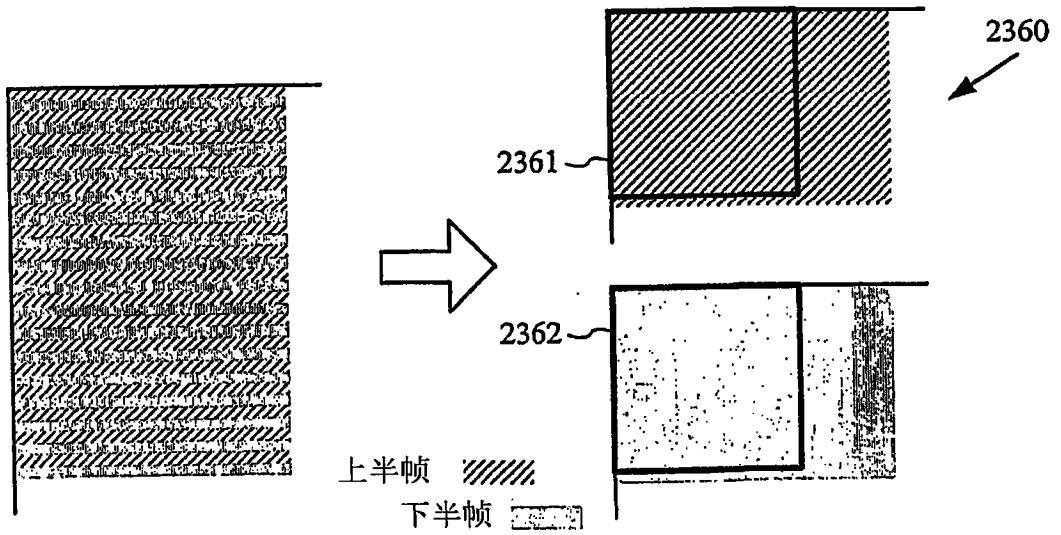


图 23C

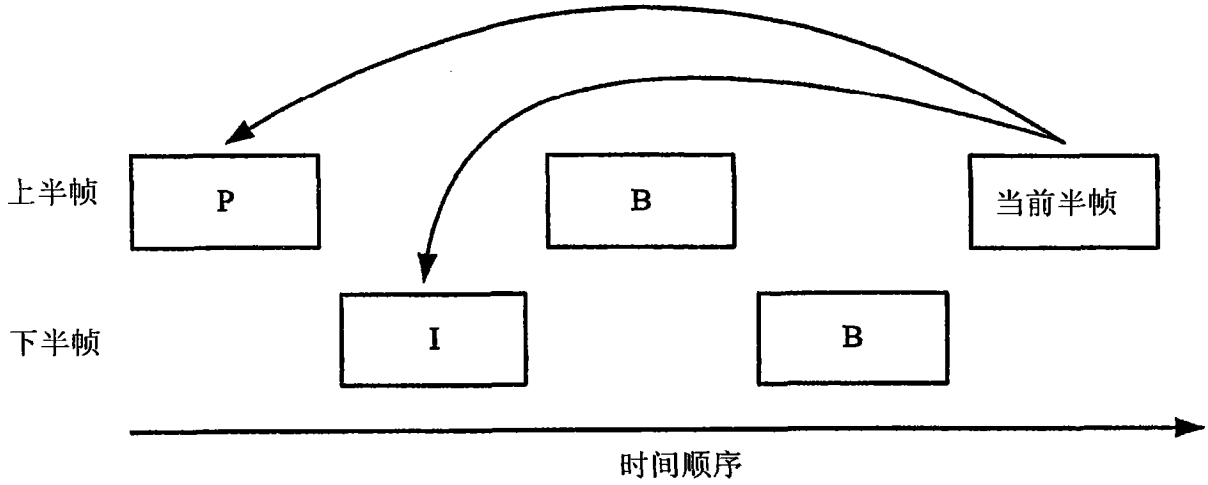


图 24A

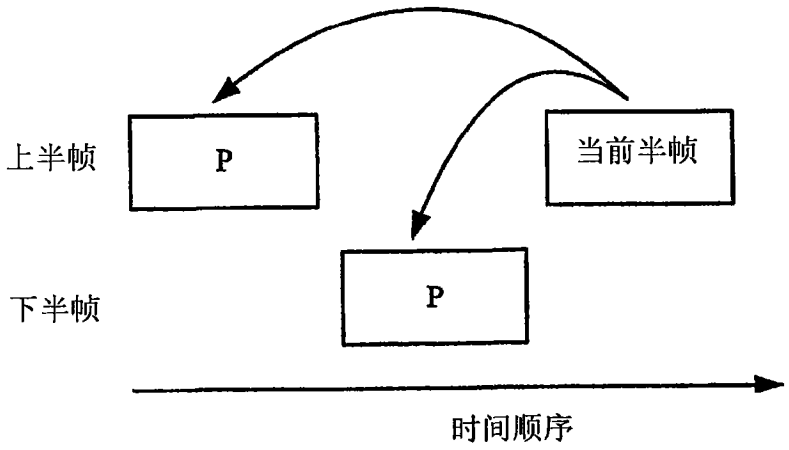


图 24B

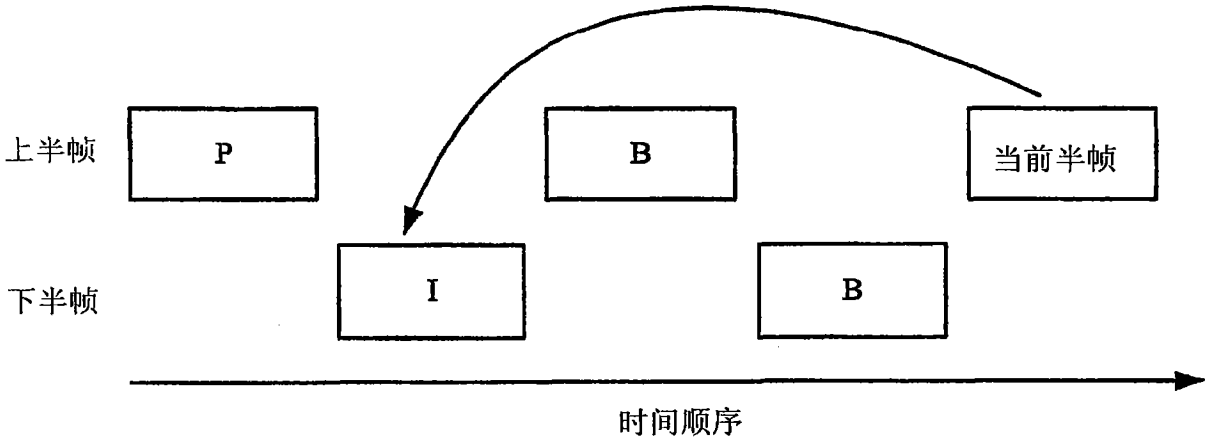


图 24C

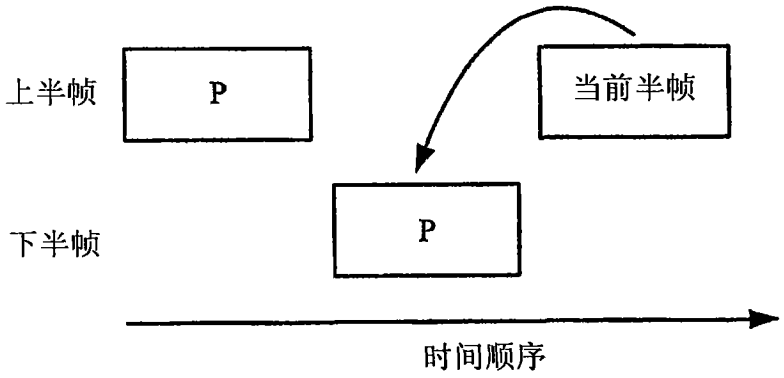


图 24D

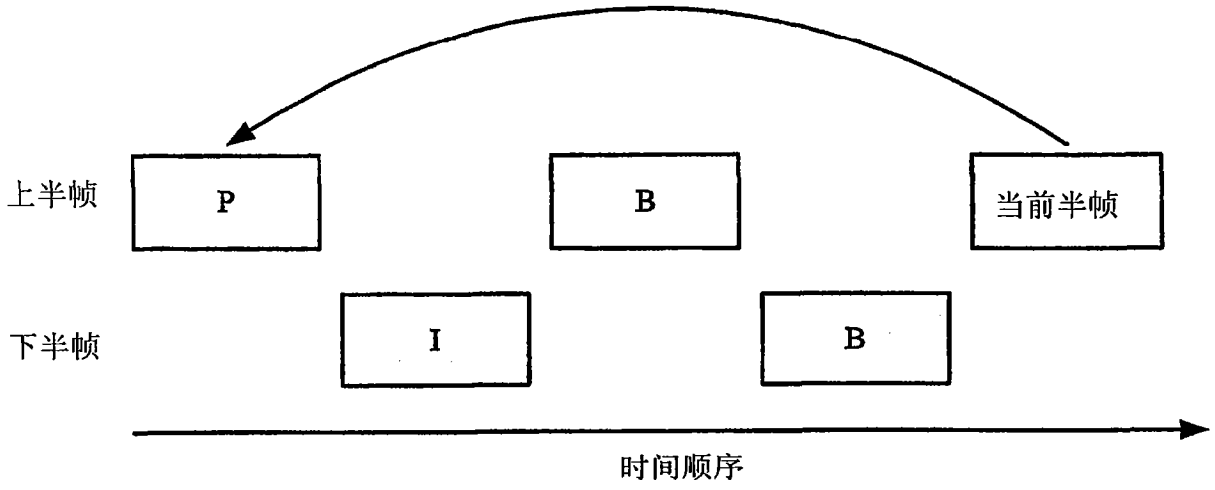


图 24E

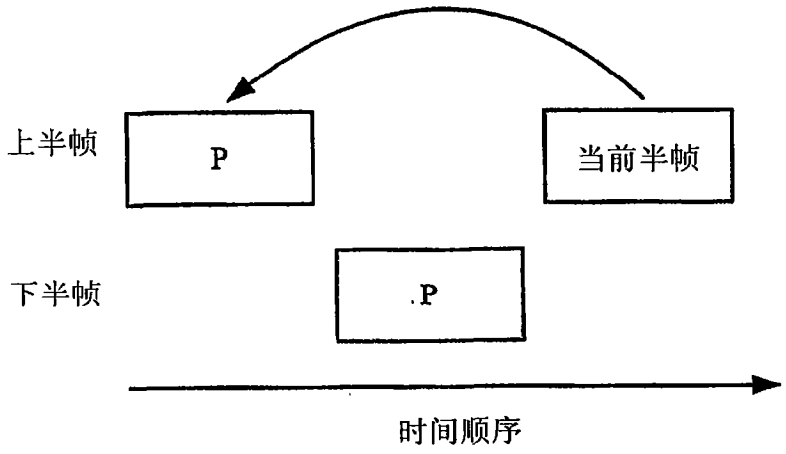


图 24F

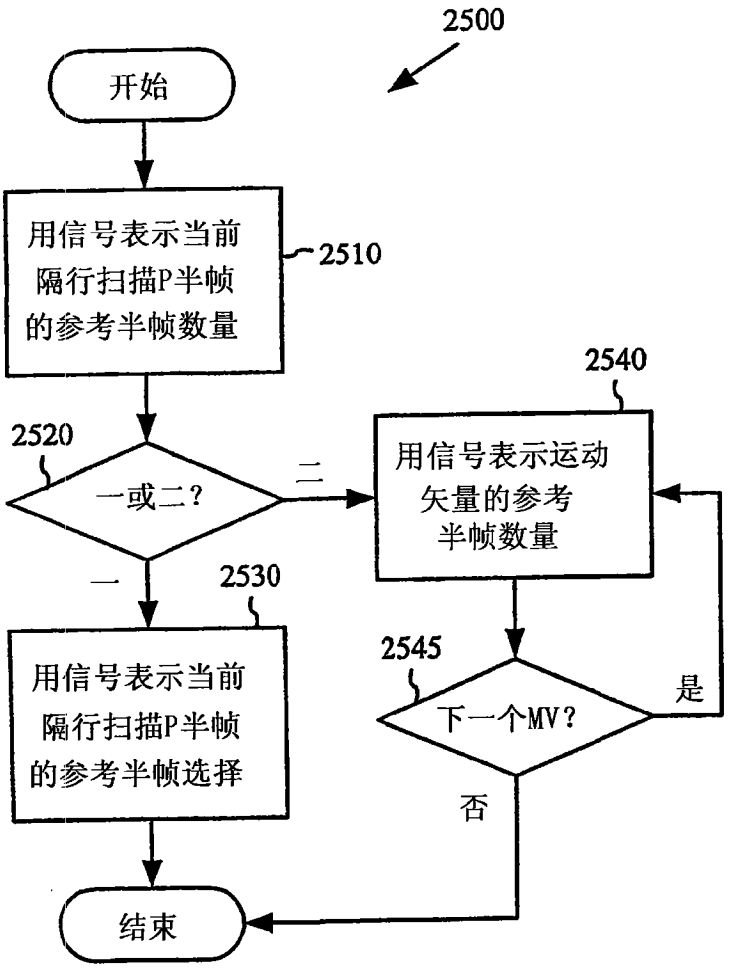

2500
开始
用信号表示当前隔行扫描P半帧的参考半帧数量
2510
2520
一或二?
二
一
2530
用信号表示当前隔行扫描P半帧的参考半帧选择
2540
用信号表示运动矢量的参考半帧数量
2545
下一个MV?
是
否
结束

图 25A

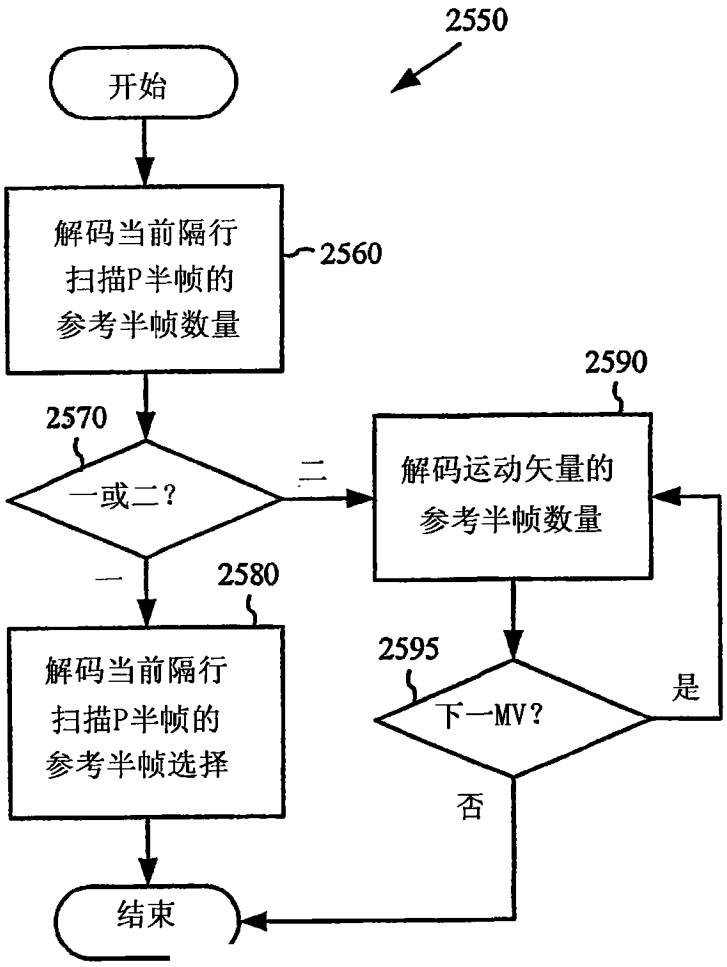


图 25B

索引	宏块类型	CBP 存在	MV 存在
0	帧内编码	否	NA
1	帧内编码	是	NA
2	1MV	否	否
3	1MV	否	是
4	1MV	是	否
5	1MV	是	是

图 26

索引	宏块类型	CBP 存在	MV 存在
0	帧内编码	否	NA
1	帧内编码	是	NA
2	1MV	否	否
3	1MV	否	是
4	1MV	是	否
5	1MV	是	是
6	4MV	否	NA
7	4MV	是	NA

图 27

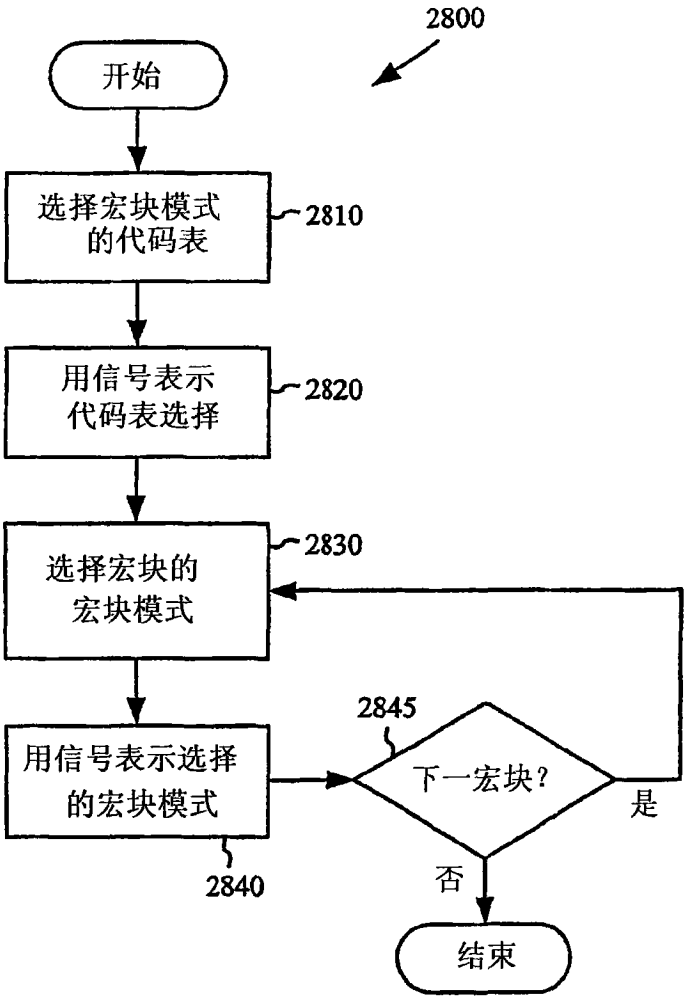

2800
开始
选择宏块模式的代码表
2810
用信号表示代码表选择
2820
选择宏块的宏块模式
2830
用信号表示选择的宏块模式
2840
2845
下一宏块?
是
否
结束

图 28A

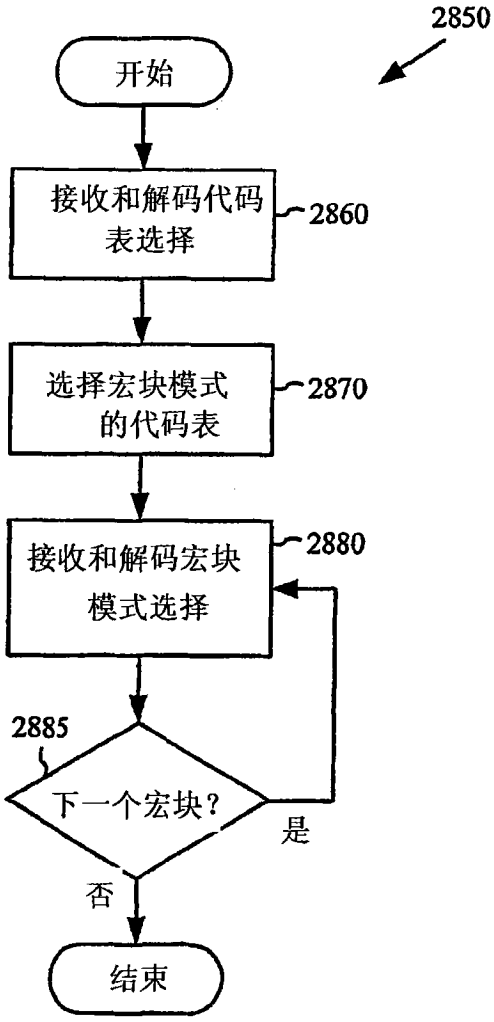

2850
开始
接收和解码代码表选择
2860
选择宏块模式的代码表
2870
接收和解码宏块模式选择
2880
2885
下一个宏块?
是
否
结束

图 28B

```
samefieldcount = 0
oppfieldcount = 0
if (预测值A不在边界外且预测值A不是帧内编码的)                    {
    if (预测值A来自相同的半帧)
    {
        samefieldcount = samefieldcount + 1
    }
    else
    {
        oppositefieldcount = oppfieldcount + 1
    }
}
 if (预测值B不在边界外且预测值B不是帧内编码的)                    {
    if (预测值B来自相同的半帧)
    {
        samefieldcount = samefieldcount + 1
    }
    else
    {
        oppositefieldcount = oppfieldcount + 1
    }
}
 if (预测值C不在边界外且预测值C不是帧内编码的)                    {
    if (预测值C来自相同的半帧)
    {
        samefieldcount = samefieldcount + 1
    }
    else
    {
        oppositefieldcount = oppfieldcount + 1
    }
}

If (samfieldcount > oppfieldcount) {
    dominantpredictor = samefield
    nondominantpredictor = oppfield
}
else
{
    dominantpredictor = oppfield
    nondominantpredictor = samefield
}
```

图 29

```
if ((主预测值是相同的半帧且当前参考是相同的半帧))
   或((主预测值是相反的半帧且当前参考是相反的半帧))
{
     DMVY = (MVY - PMVY) * 2
}
else {
     DMVY = (MVY - PMVY) * 2 + 1
}
```

图 30

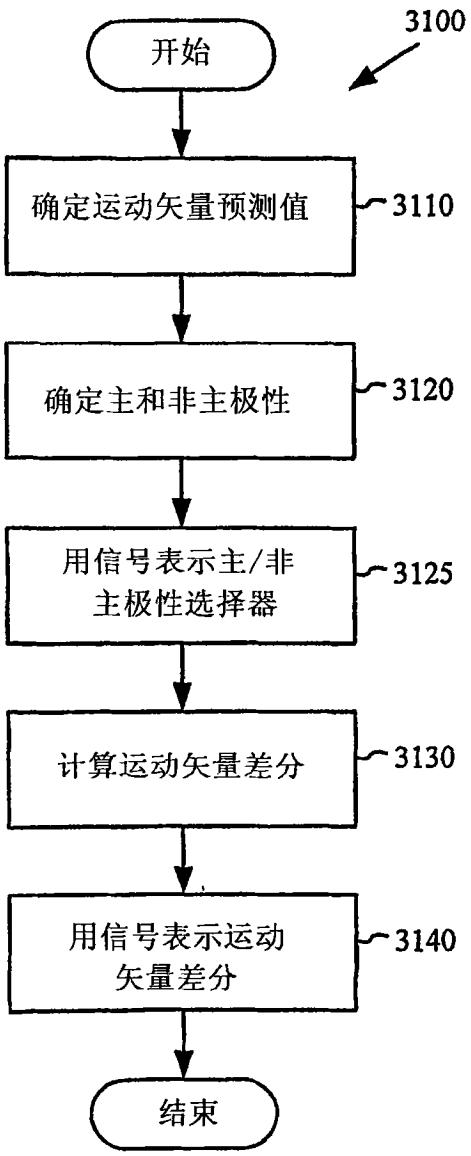


图 31A

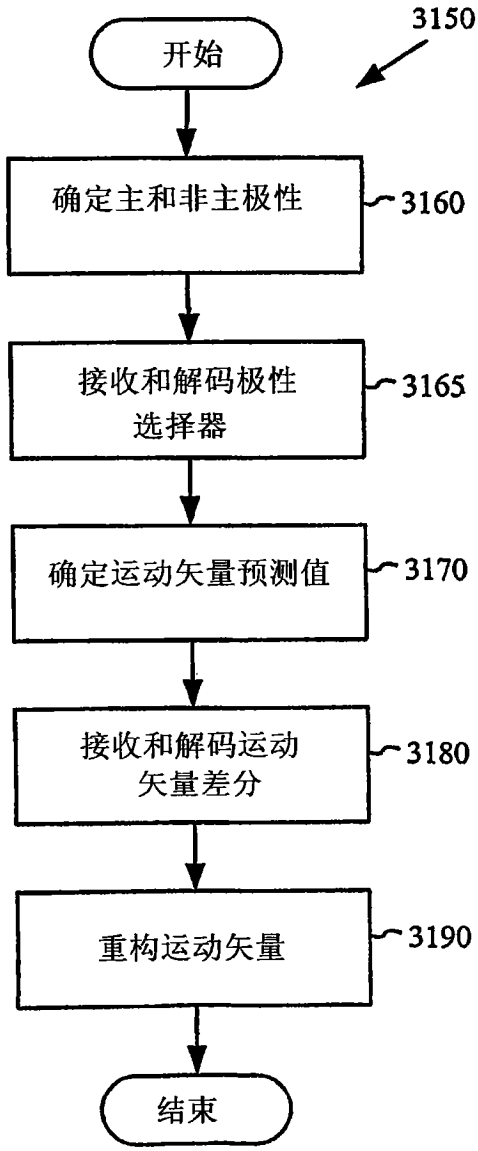


图 31B

```
if ((预测值A在边界外)) || ((预测值C在边界外))
    predictor_post_x = predictor_pre_x
    predictor_post_y = predictor_pre_y
}
else {
    if (预测值A是帧内编码的)
        sum = abs(predictor_pre_x) + abs(predictor_pre_y)
    else
        sum = abs(predictor_pre_x - predictorA_x) + abs(predictor_pre_y -
predictorA_y)
    if (sum > 32) {
        //读下一比特以查看哪个候选预测值要使用
        if (ReadTheHybridBit == 1) { //HYBRIDPRED字段
            //使用上预测值（预测值A）
            predictor_post_x = predictorA_x
            predictor_post_y = predictorA_y
        }
        else {
            //使用左预测值（预测值C）
            predictor_post_x = predictorC_x
            predictor_post_y = predictorC_y
        }
    }
    else {
        if(预测值C是帧内编码的)
            sum = abs(predictor_pre_x) + abs(predictor_pre_y)
        else
            sum = abs(predictor_pre_x - predictorC_x) + abs(predictor_pre_y -
predictorC_y)
        if (sum > 32) {
            //读下一比特以查看哪个候选预测值要使用
        if (ReadTheHybridBit == 1) { //HYBRIDPRED字段
                //使用上预测值（预测值A）
                predictor_post_x = predictorA_x
                predictor_post_y = predictorA_y
            }
            else {
                //使用左预测值（预测值C）
                predictor_post_x = predictorC_x
                predictor_post_y = predictorC_y
            }
        }
    }
}
```

图 32

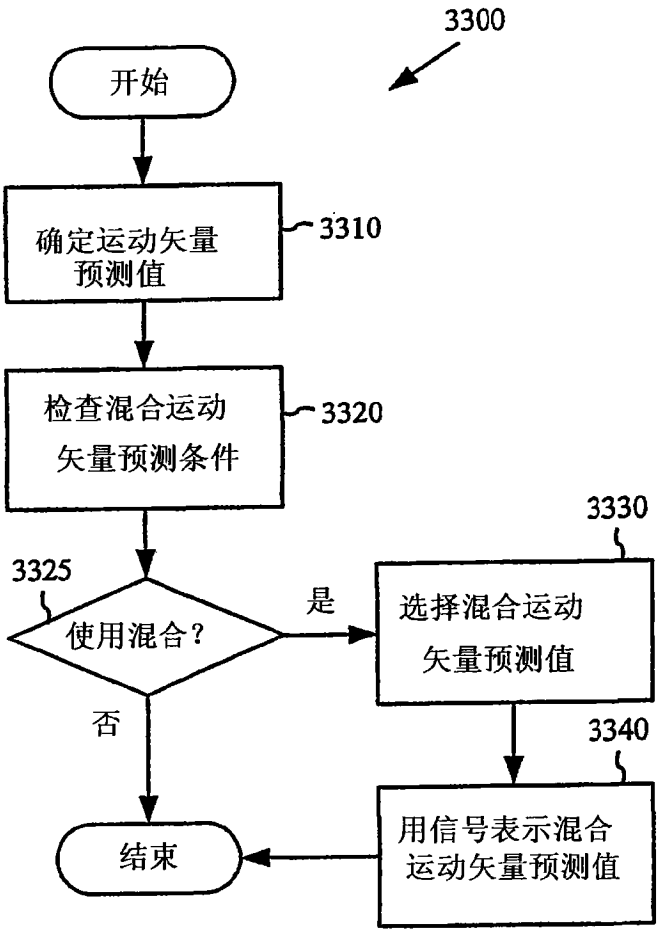

3300
开始
确定运动矢量预测值
3310
检查混合运动矢量预测条件
3320
3325
使用混合？
是
否
3330
选择混合运动矢量预测值
3340
用信号表示混合运动矢量预测值
结束

图 33A

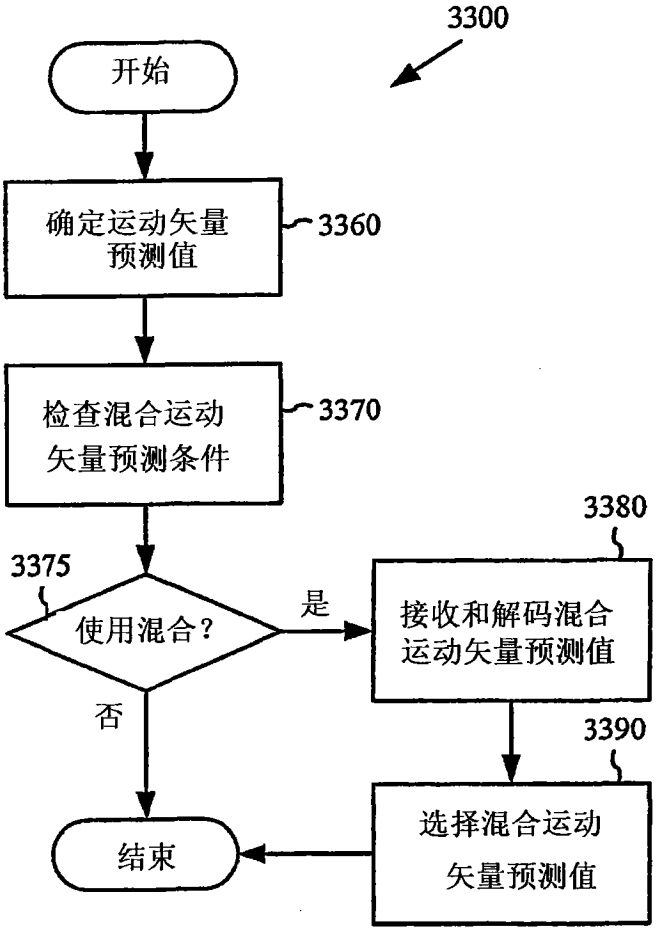

3300
开始
确定运动矢量预测值
3360
检查混合运动矢量预测条件
3370
3375
使用混合?
是
否
3380
接收和解码混合运动矢量预测值
3390
选择混合运动矢量预测值
结束

图 33B

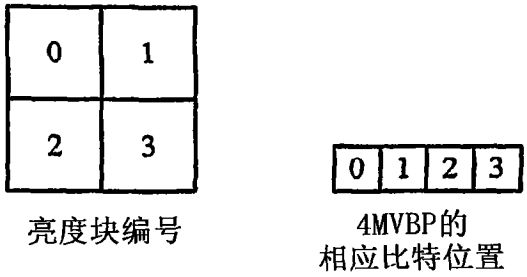

0
1
2
3
亮度块编号
0 1 2 3
4MVBP的相应比特位置

图 34

```
//编码DMVX、DMVY运动矢量差分对；
//水平范围是-RX到+RX-1
//垂直范围是-RY到+RY-1
//RX的绝对值=2^MX
//RY的绝对值=2^MY
//AX=DMVX的绝对值
//AY=DMVY的绝对值
//SX=DMVX的符号：如果SX=1则SX=1，并且如果DMVX大于或等于0则SX=0
//SY=DMVY的符号：如果SY=1则SY=1，并且如果DMVY大于或等于0则SY=0
//ESCX=2^KX
//ESCY=2^KY
//R=参考半帧相对于预测参考半帧的极性（0=与预测值相同，1=与预测值相反）
if ( AX > ESCX或AY > ESCY)
{
    SendBits (VLC_CODE [ESCAPE,R], VLC_SIZE [ESCAPE,R])
    SendBits (DMVX, MX+1)
    SendBits (DMVY, MY+1)
}
else
{
    SendBits (VLC_CODE [NX, NY,R], VLC_SIZE [NX, NY,R])
    SendBits (AX, NX)
    SendBits (SX, 1)
    SendBits (AY NY)
    SendBits (SY, 1)
}
```

图 36

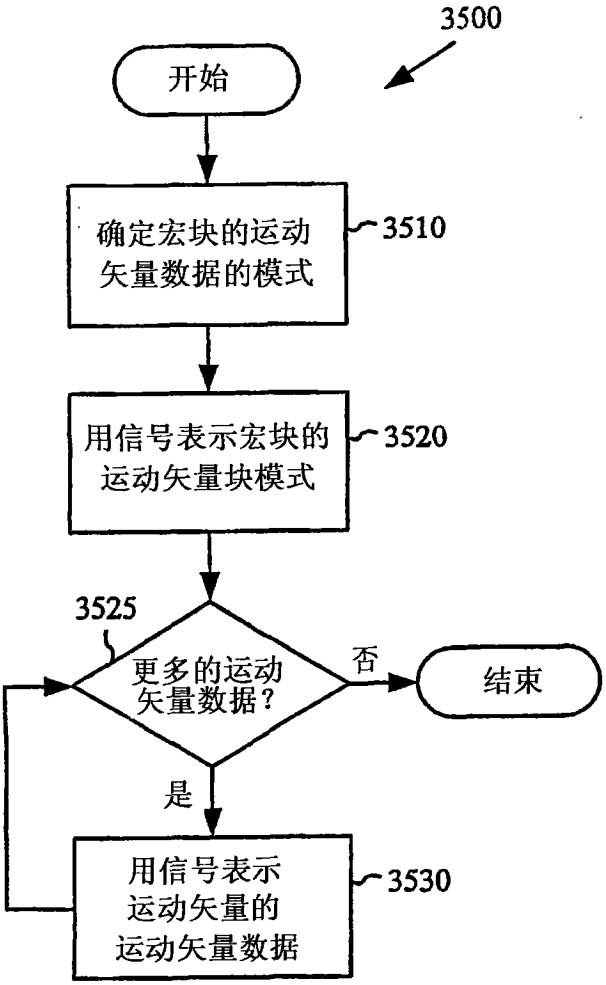

3500
开始
确定宏块的运动矢量数据的模式
3510
用信号表示宏块的运动矢量块模式
3520
3525
更多的运动矢量数据？
否
结束
是
用信号表示运动矢量的运动矢量数据
3530

图 35A

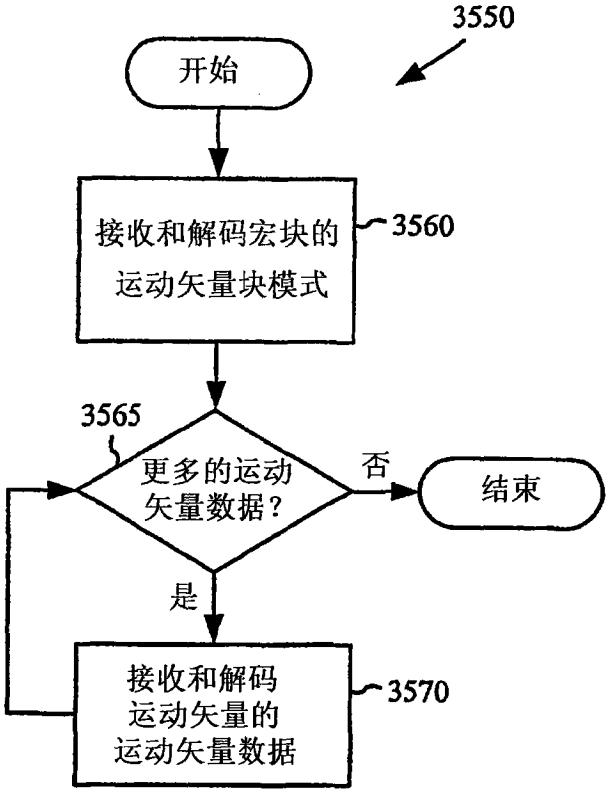

3550
开始
接收和解码宏块的
运动矢量块模式
3560
3565
更多的运动
矢量数据？
否
结束
是
接收和解码
运动矢量的
运动矢量数据
3570

图 35B

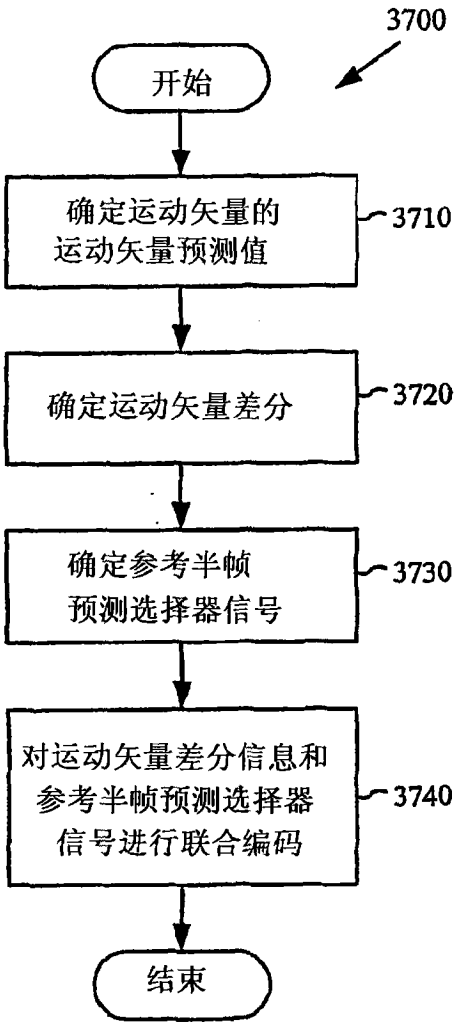

3700
开始
确定运动矢量的
运动矢量预测值
3710
确定运动矢量差分
3720
确定参考半帧
预测选择器信号
3730
对运动矢量差分信息和
参考半帧预测选择器
信号进行联合编码
3740
结束

图 37A

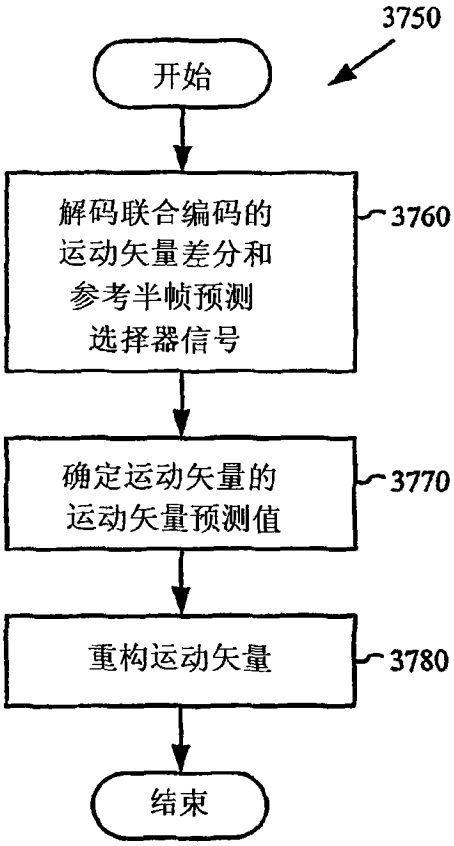

3750
开始
解码联合编码的
运动矢量差分和
参考半帧预测
选择器信号
3760
确定运动矢量的
运动矢量预测值
3770
重构运动矢量
3780
结束

图 37B

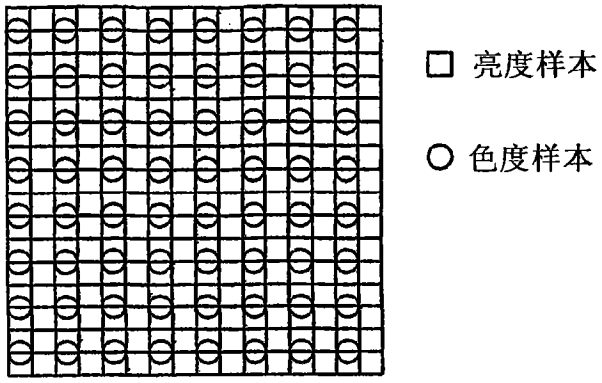

亮度样本
色度样本

图 38

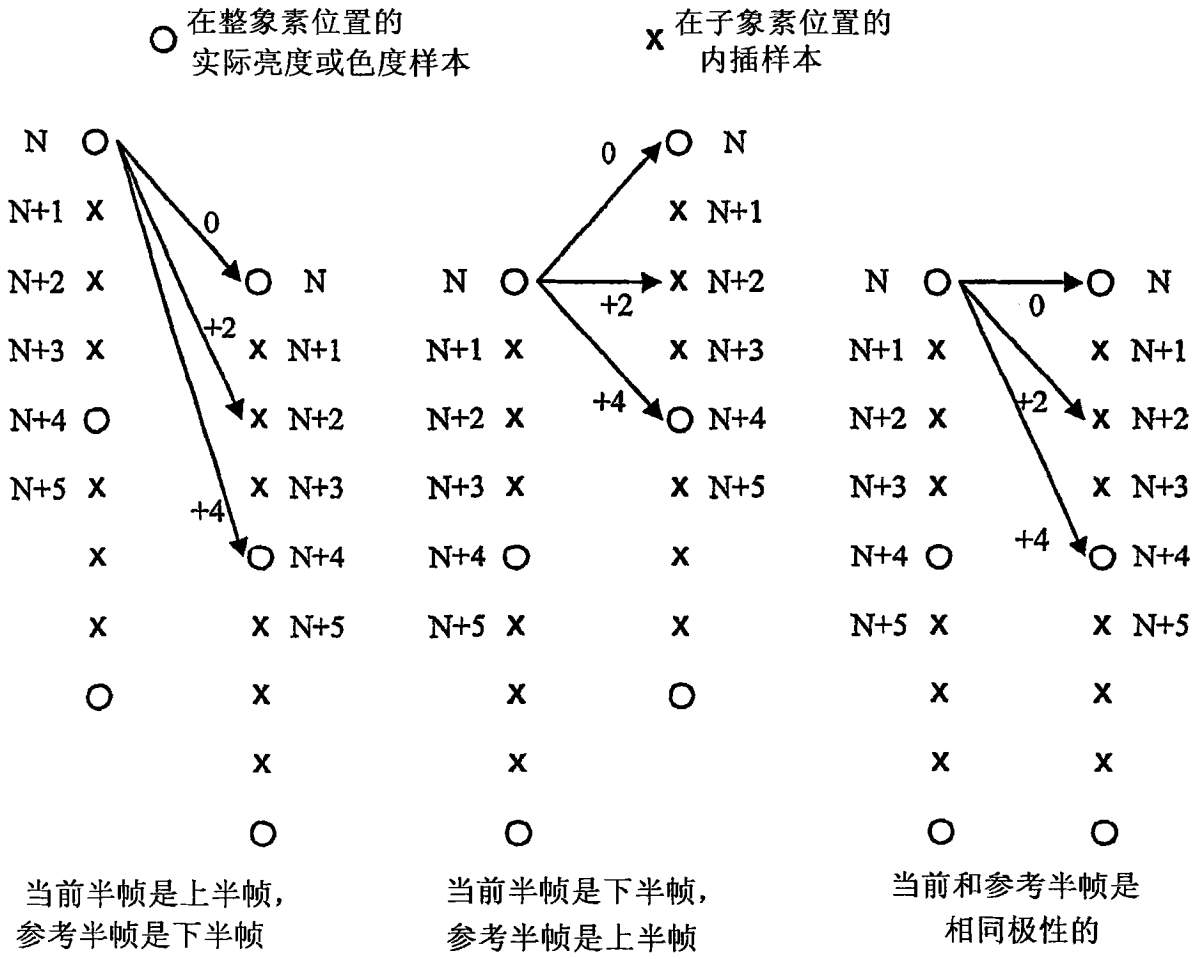

在整象素位置的
实际亮度或色度样本
在子象素位置的
内插样本
N
N+1
N+2
N+3
N+4
N+5
0
+2
+4
当前半帧是上半帧，
参考半帧是下半帧
当前半帧是下半帧，
参考半帧是上半帧
当前和参考半帧是
相同极性的

图 39

```
MotionVector SelectChromaMVFrom4MV ()
{
    MotionVector SelectedMV;

    如果对应的MB使用了1 MV
    则使用该MV
    else     //要从4个MV选取
    {

            对相同半帧和相反半帧MV的数目进行计数

            if (OppFieldCount > SameFieldCount)
            则在下一步中仅使用相反半帧MV
            //即，相反是选择的极性
            else在下一步中仅使用相同半帧MV
            //即，相同是选择的极性
            对所选择极性的MV的数目进行计数
            if (选择的MV = 3)   {
               SelectedMV = 选择的3个MV的中值
            }
             else if (选择的MV =2)   {
               SelectedMV = 选择的2个MV的平均值
            }
            else if (选择的MV = 1)   {
                SelectedMV = 选择的MV
            }
            else {  //全部4个是所选择极性的
                SelectedMV = 选择的4个MV的中值
            }
        }
    }

    return (SelectedMV);
}
```

图 40

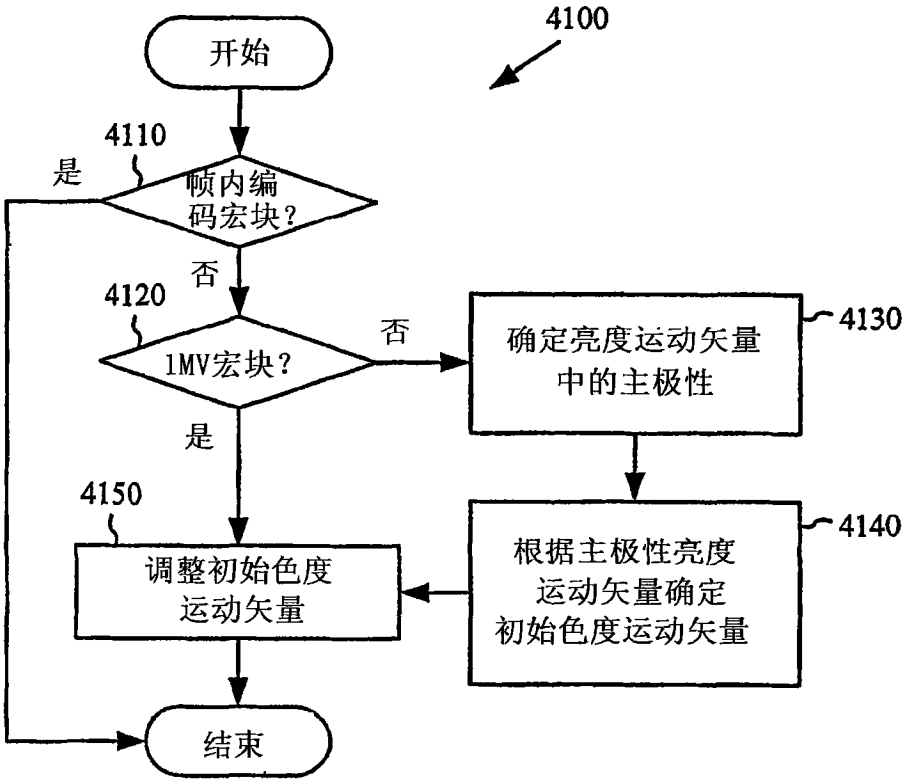

4100
开始
4110
是
帧内编码宏块?
否
4120
1MV宏块?
否
确定亮度运动矢量中的主极性
4130
是
4150
调整初始色度运动矢量
根据主极性亮度运动矢量确定初始色度运动矢量
4140
结束

图 41

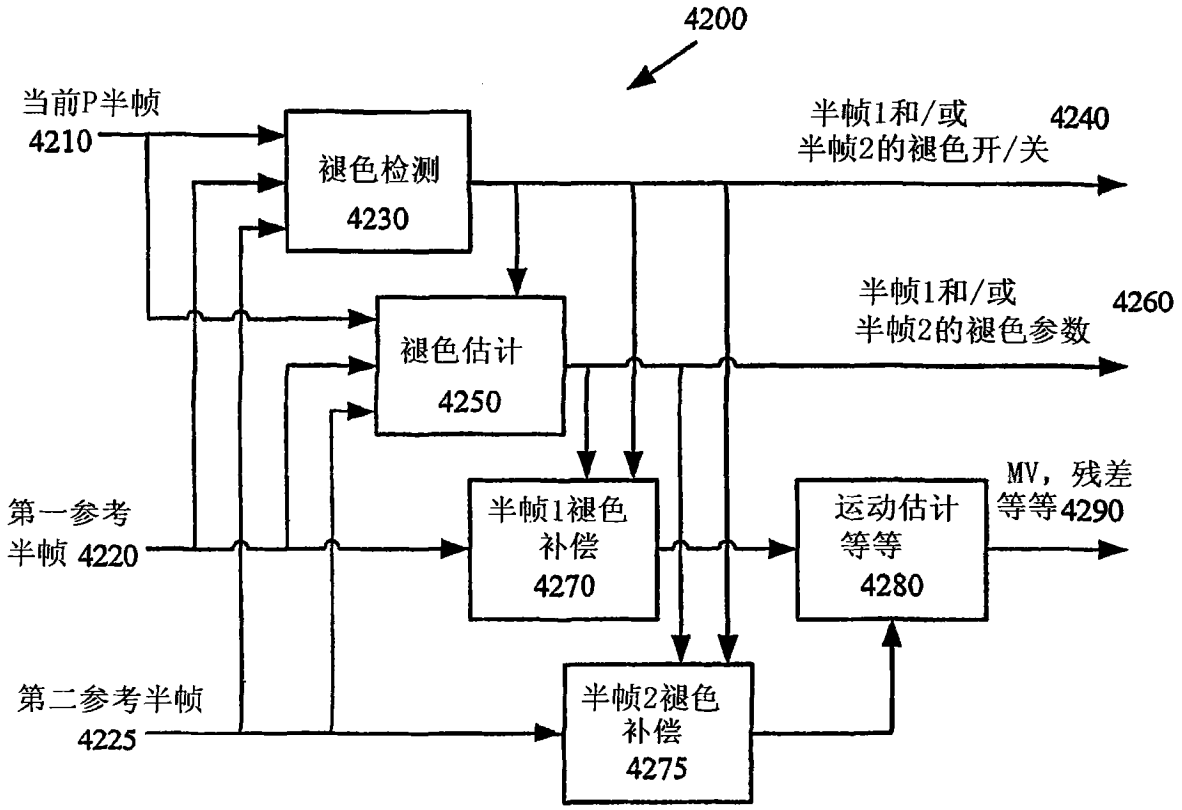

4200
当前P半帧
4210
褪色检测
4230
半帧1和/或
半帧2的褪色开/关
4240
褪色估计
4250
半帧1和/或
半帧2的褪色参数
4260
第一参考
半帧 4220
半帧1褪色
补偿
4270
运动估计
等等
4280
MV，残差
等等4290
第二参考半帧
4225
半帧2褪色
补偿
4275

图 42

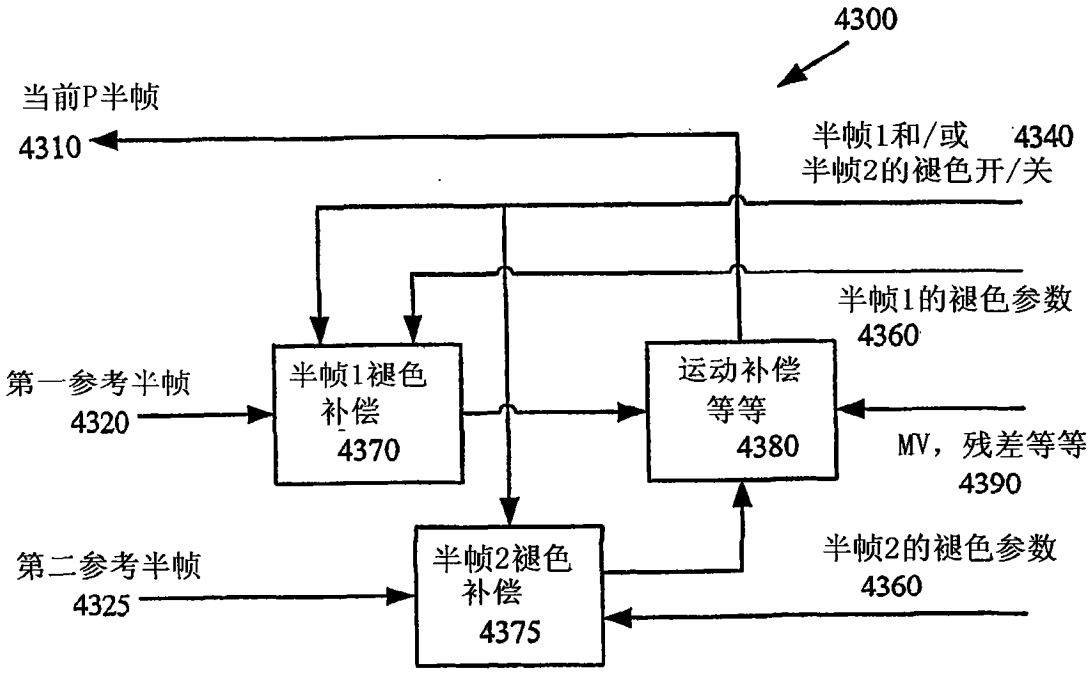

4300
当前P半帧
4310
半帧1和/或
半帧2的褪色开/关
4340
半帧1的褪色参数
4360
第一参考半帧
4320
半帧1褪色
补偿
4370
运动补偿
等等
4380
MV，残差等等
4390
第二参考半帧
4325
半帧2褪色
补偿
4375
半帧2的褪色参数
4360

图 43

INTCOMPFIELD VLC	全局亮度补偿应用于：
1	两个参考半帧
00	第二最近参考半帧
01	最近参考半帧

图 44

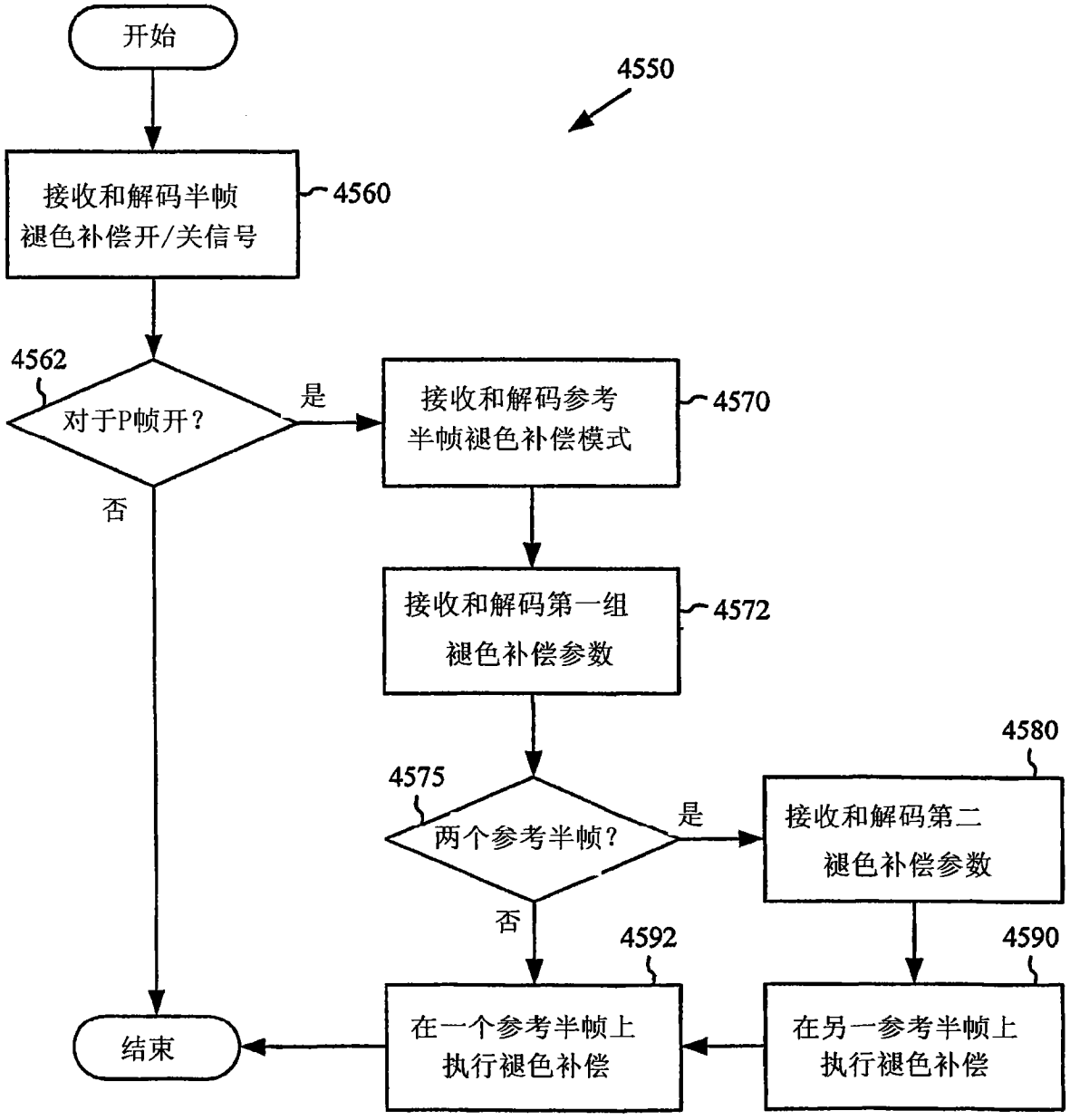


图 45B

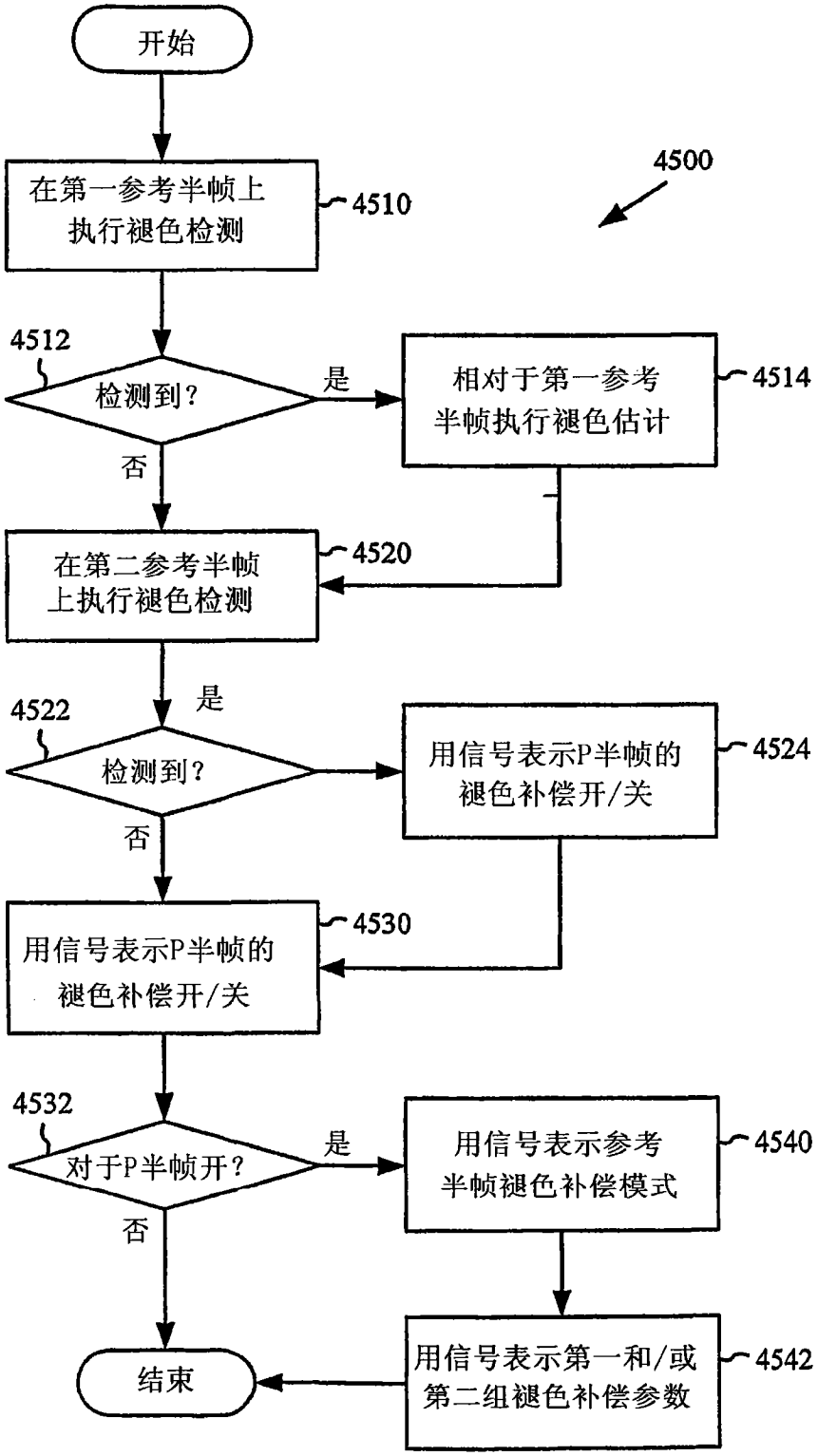

开始
在第一参考半帧上执行褪色检测
4510
4500
4512
检测到？
是
相对于第一参考半帧执行褪色估计
4514
否
在第二参考半帧上执行褪色检测
4520
4522
是
检测到？
用信号表示P半帧的褪色补偿开/关
4524
否
用信号表示P半帧的褪色补偿开/关
4530
4532
对于P半帧开？
是
用信号表示参考半帧褪色补偿模式
4540
否
用信号表示第一和/或第二组褪色补偿参数
4542
结束

图 45A

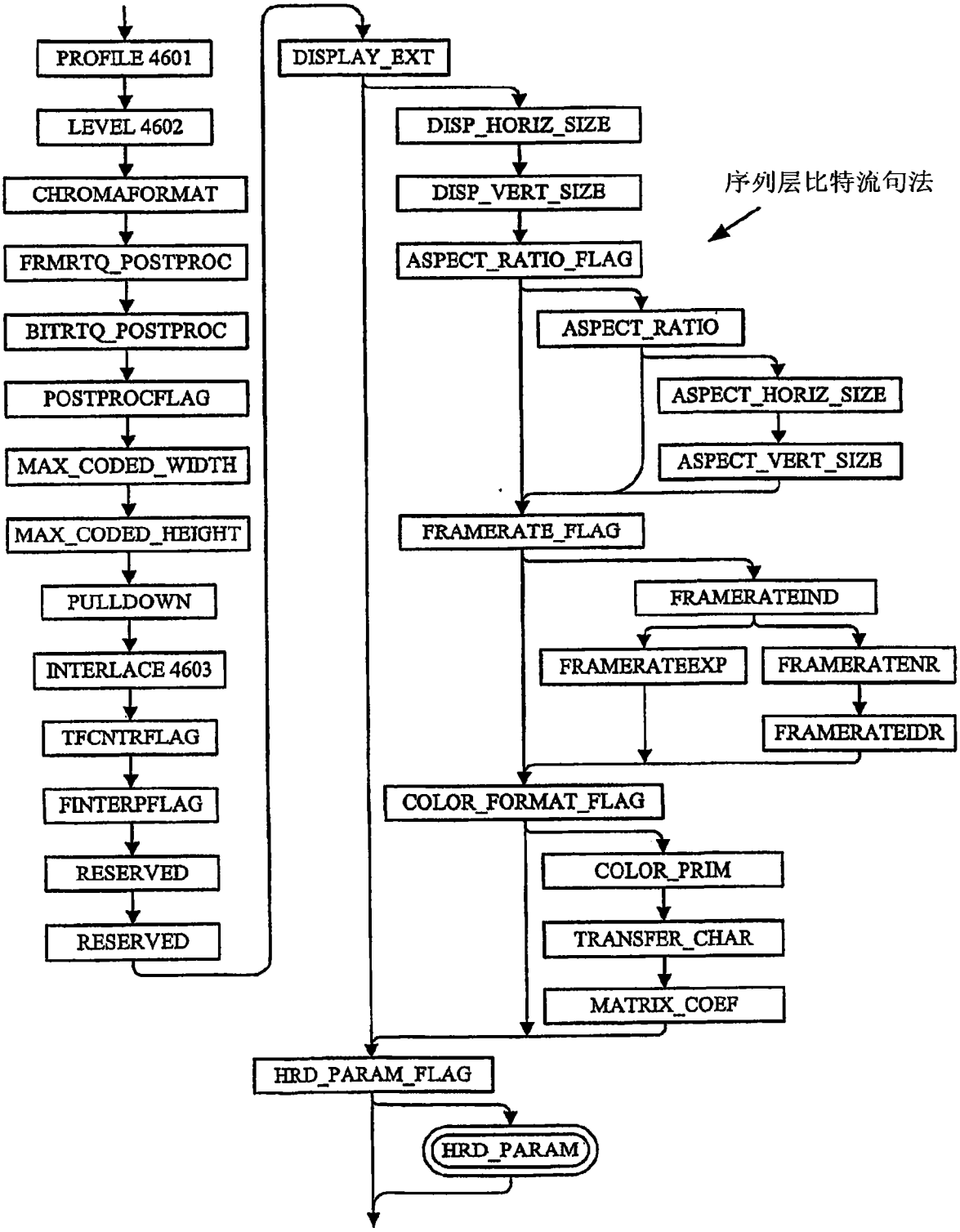

PROFILE 4601
LEVEL 4602
CHROMAFORMAT
FRMRTQ_POSTPROC
BITRTQ_POSTPROC
POSTPROCFLAG
MAX_CODED_WIDTH
MAX_CODED_HEIGHT
PULLDOWN
INTERLACE 4603
TFCNTRFLAG
FINTERPFLAG
RESERVED
RESERVED
DISPLAY_EXT
DISP_HORIZ_SIZE
DISP_VERT_SIZE
ASPECT_RATIO_FLAG
ASPECT_RATIO
ASPECT_HORIZ_SIZE
ASPECT_VERT_SIZE
FRAMERATE_FLAG
FRAMERATEIND
FRAMERATEEXP
FRAMERATENR
FRAMERATEIDR
COLOR_FORMAT_FLAG
COLOR_PRIM
TRANSFER_CHAR
MATRIX_COEF
HRD_PARAM_FLAG
HRD_PARAM
序列层比特流句法

图 46A

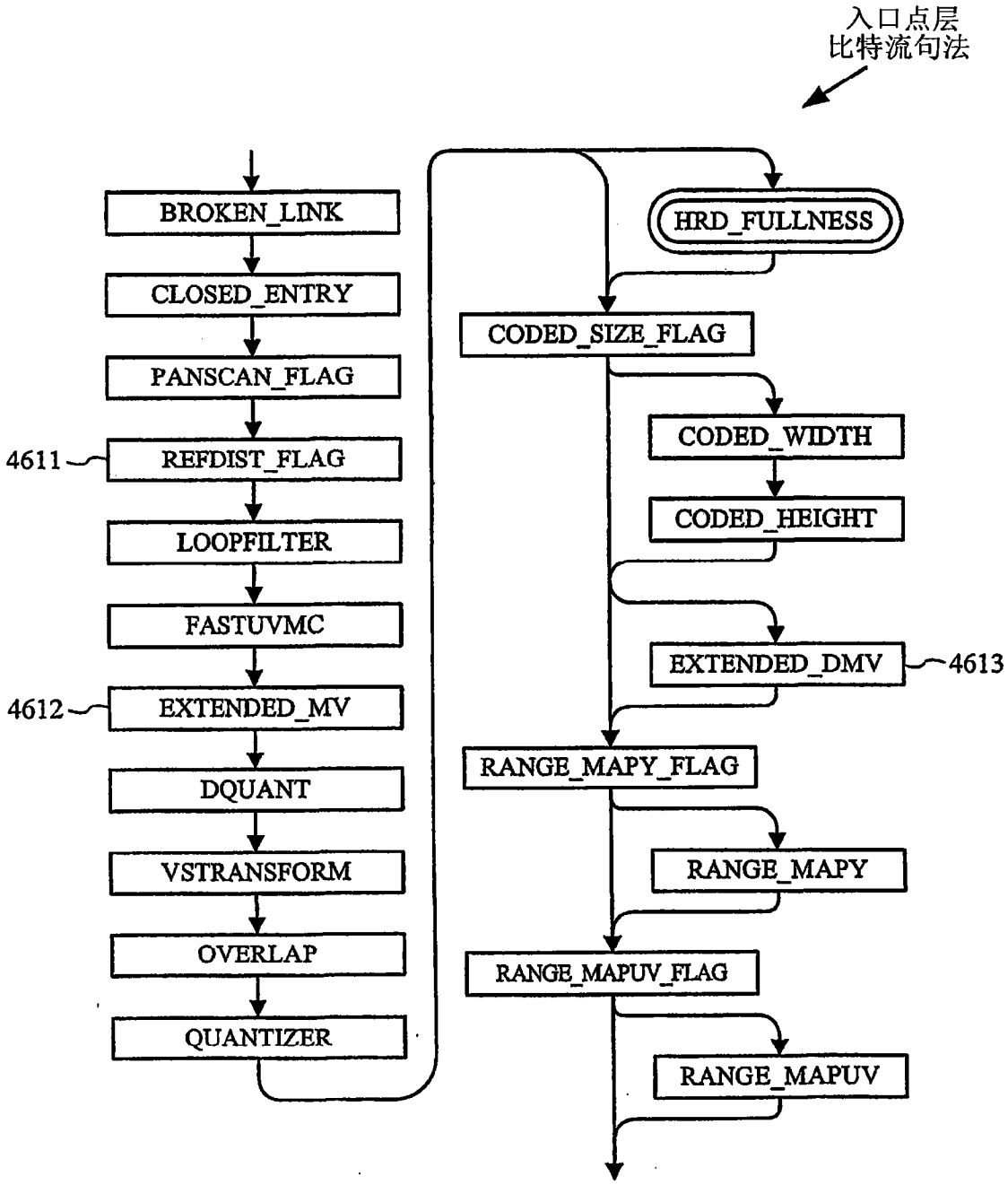

入口点层
比特流句法
BROKEN_LINK
CLOSED_ENTRY
PANSCAN_FLAG
4611
REFDIST_FLAG
LOOPFILTER
FASTUVMC
4612
EXTENDED_MV
DQUANT
VSTRANSFORM
OVERLAP
QUANTIZER
HRD_FULLNESS
CODED_SIZE_FLAG
CODED_WIDTH
CODED_HEIGHT
EXTENDED_DMV
4613
RANGE_MAPY_FLAG
RANGE_MAPY
RANGE_MAPUV_FLAG
RANGE_MAPUV

图 46B

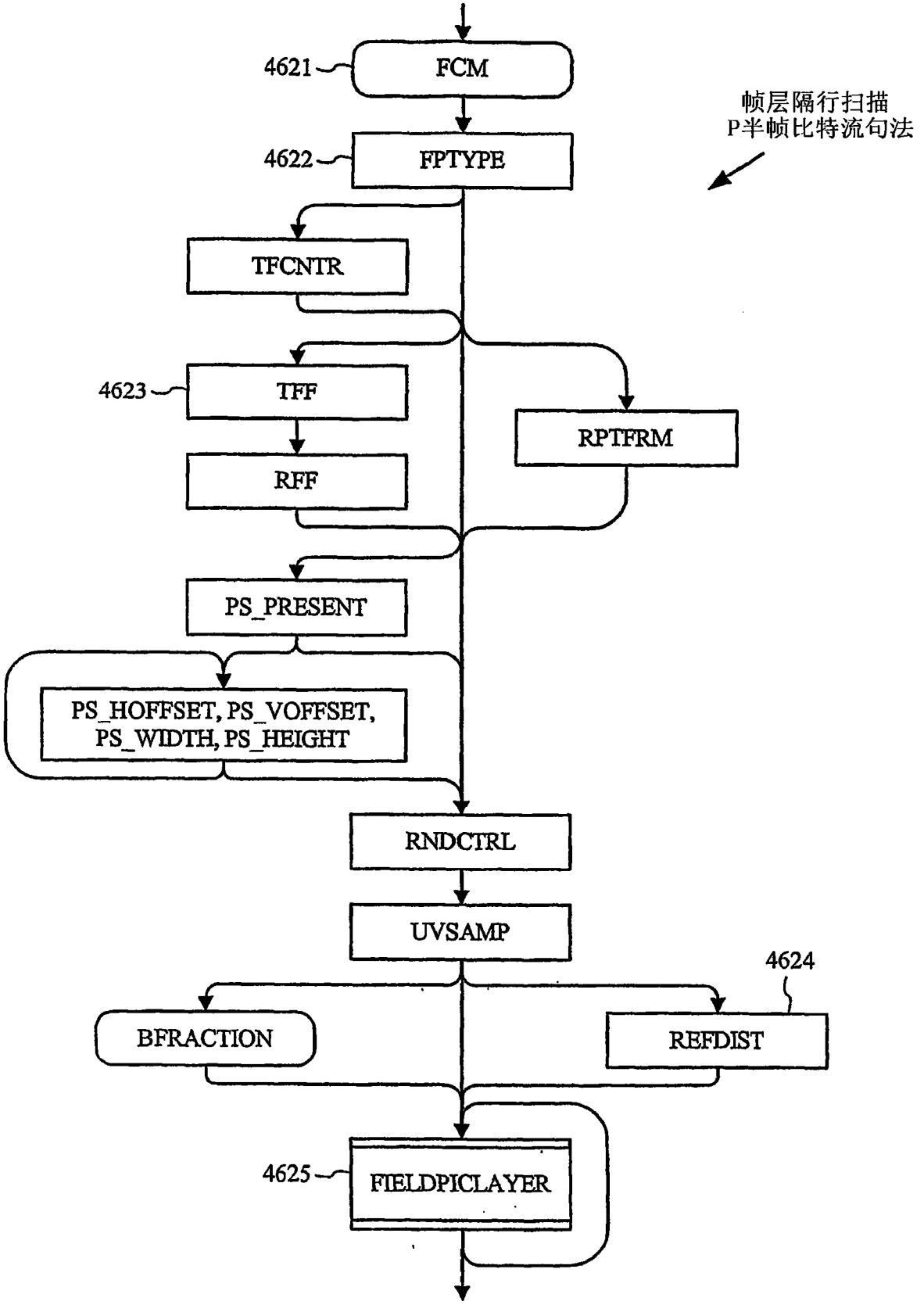

4621
FCM
帧层隔行扫描
P半帧比特流句法
4622
FPTYPE
TFCNTR
4623
TFF
RPTFRM
RFF
PS_PRESENT
PS_HOFFSET, PS_VOFFSET,
PS_WIDTH, PS_HEIGHT
RNDCTRL
UVSAMP
4624
BFRACTION
REFDIST
4625
FIELDPICLAYER

图 46C

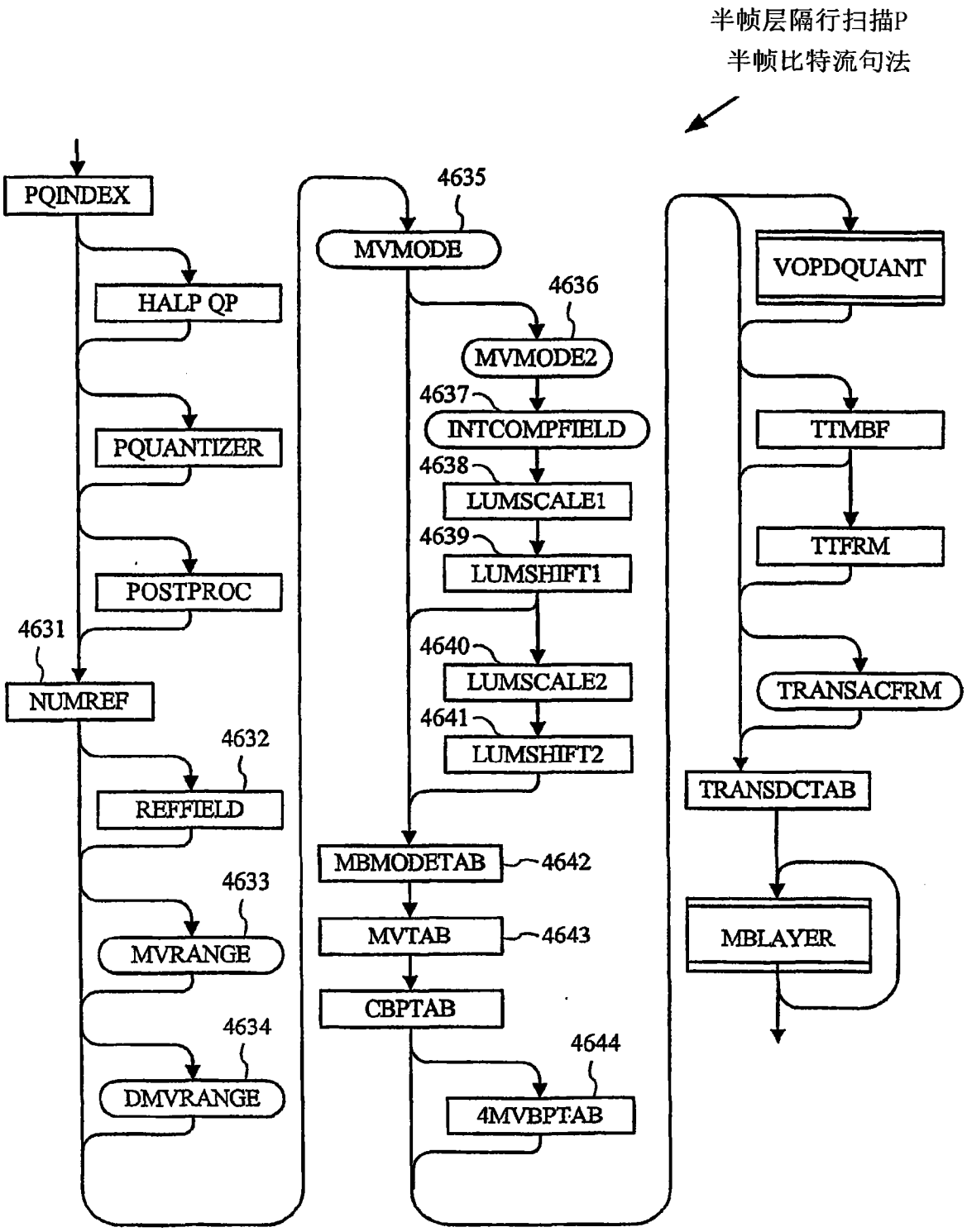

半帧层隔行扫描P
半帧比特流句法
PQINDEX
HALP QP
PQUANTIZER
POSTPROC
4631
NUMREF
4632
REFFIELD
4633
MVRANGE
4634
DMVRANGE
4635
MVMODE
4636
MVMODE2
4637
INTCOMPFIELD
4638
LUMSCALE1
4639
LUMSHIFT1
4640
LUMSCALE2
4641
LUMSHIFT2
MBMODETAB
4642
MVTAB
4643
CBPTAB
4644
4MVBPTAB
VOPDQUANT
TTMBF
TTFRM
TRANSACFRM
TRANSDCTAB
MBLAYER

图 46D

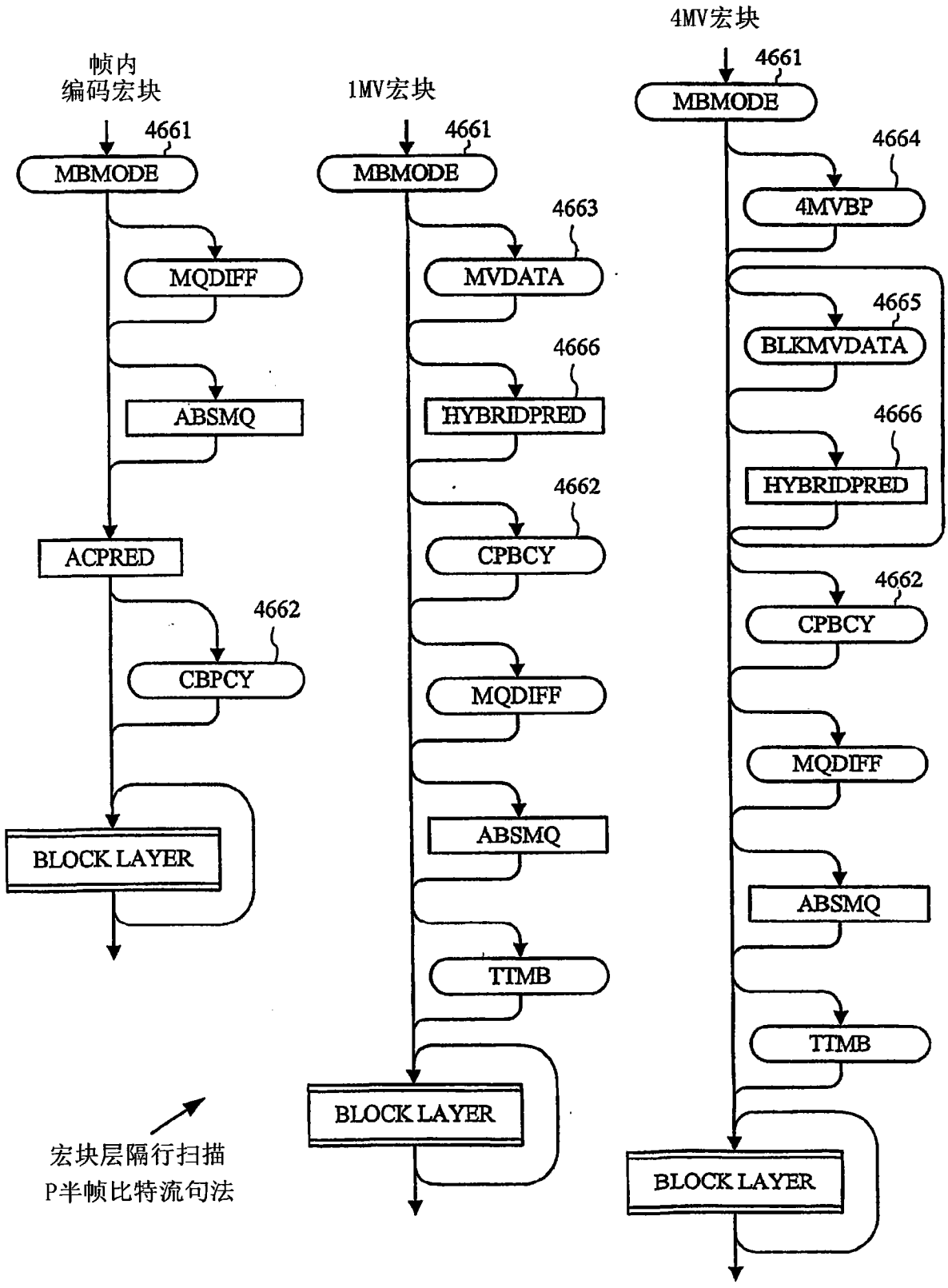

帧内
编码宏块
4661
MBMODE
MQDIFF
ABSMQ
ACPRED
4662
CBPCY
BLOCK LAYER
1MV宏块
4661
MBMODE
4663
MVDATA
4666
HYBRIDPRED
4662
CPBCY
MQDIFF
ABSMQ
TTMB
BLOCK LAYER
4MV宏块
4661
MBMODE
4664
4MVBP
4665
BLKMVDATA
4666
HYBRIDPRED
4662
CPBCY
MQDIFF
ABSMQ
TTMB
BLOCK LAYER
宏块层隔行扫描
P半帧比特流句法

图 46E

FCM	图像编码类型
0	逐行扫描
10	帧-隔行扫描
11	半帧-隔行扫描

图 47A

FPTYPE FLC	第一半帧图像类型	第二半帧图像类型
000	I	I
001	I	P
010	P	I
011	P	P
100	B	B
101	B	BI
110	BI	B
111	BI	BI

图 47B

参考帧距离	VLC（二进制）	VLC大小
0	00	2
1	01	2
2	10	2
N	11[(N-3)个1]0	N

图 47C

扩展的水平差分运动矢量范围	扩展的垂直差分运动矢量范围	VLC（二进制）	VLC大小
否	否	0	1
是	否	10	2
否	是	110	3
是	是	111	3

图 47D

P图像低速率（PQUANT>12）MVMODE表

MVMODE VLC	模式
1	1MV二分之一象素双线性
01	1 MV
001	1MV二分之一象素
0000	混合的MV
0001	强度补偿

P图像高速率（PQUANT<=12）MVMODE表

MVMODE VLC	模式
1	1 MV
01	混合的MV
001	1MV二分之一象素
0000	1MV二分之一象素双线性
0001	强度补偿

图 47E

P图像低速率（PQUANT>12）MVMODE2表

MVMODE2 VLC	模式
1	1MV二分之一象素双线性
01	1 MV
001	1MV二分之一象素
000	混合的MV

P图像高速率（PQUANT<=12）MVMODE2表

MVMODE2 VLC	模式
1	1 MV
01	混合的MV
001	1MV二分之一象素
000	1MV二分之一象素双线性

图 47F

INTCOMPFIELD VLC	强度补偿应用于：
1	两个半帧
00	上半帧
01	下半帧

图 47G

混合MV宏块模式表0

宏块模式	VLC	VLC大小
0	16	6
1	17	6
2	3	2
3	3	3
4	0	2
5	5	4
6	9	5
7	2	2

混合MV宏块模式表3

宏块模式	VLC	VLC大小
0	56	6
1	57	6
2	15	4
3	4	3
4	5	3
5	6	3
6	29	5
7	0	1

混合MV宏块模式表6

宏块模式	VLC	VLC大小
0	16	5
1	17	5
2	6	3
3	7	3
4	0	2
5	1	2
6	9	4
7	5	3

混合MV宏块模式表1

宏块模式	VLC	VLC大小
0	8	5
1	9	5
2	3	3
3	6	3
4	7	3
5	0	2
6	5	4
7	2	2

混合MV宏块模式表4

宏块模式	VLC	VLC大小
0	52	6
1	53	6
2	27	5
3	14	4
4	15	4
5	2	2
6	12	4
7	0	1

混合MV宏块模式表7

宏块模式	VLC	VLC大小
0	56	6
1	57	6
2	0	1
3	5	3
4	6	3
5	29	5
6	4	3
7	15	4

混合MV宏块模式表2

宏块模式	VLC	VLC大小
0	16	6
1	17	6
2	5	4
3	3	3
4	0	2
5	3	2
6	9	5
7	2	2

混合MV宏块模式表5

宏块模式	VLC	VLC大小
0	56	6
1	57	6
2	29	5
3	5	3
4	6	3
5	0	1
6	15	4
7	4	3

图 47H

1MV宏块模式表0

宏块模式	VLC	VLC大小
0	0	5
1	1	5
2	1	1
3	1	3
4	1	2
5	1	4

1MV宏块模式表1

宏块模式	VLC	VLC大小
0	0	5
1	1	5
2	1	1
3	1	2
4	1	3
5	1	4

1MV宏块模式表2

宏块模式	VLC	VLC大小
0	16	5
1	17	5
2	3	2
3	0	1
4	9	4
5	5	3

1MV宏块模式表3

宏块模式	VLC	VLC大小
0	20	5
1	21	5
2	3	2
3	11	4
4	0	1
5	4	3

1MV宏块模式表4

宏块模式	VLC	VLC大小
0	4	4
1	5	4
2	2	2
3	3	3
4	3	2
5	0	2

1MV宏块模式表5

宏块模式	VLC	VLC大小
0	4	4
1	5	4
2	3	3
3	2	2
4	0	2
5	3	2

1MV宏块模式表6

宏块模式	VLC	VLC大小
0	0	5
1	1	5
2	1	3
3	1	4
4	1	1
5	1	2

1MV宏块模式表7

宏块模式	VLC	VLC大小
0	16	5
1	17	5
2	9	4
3	5	3
4	3	2
5	0	1

图 47I

4MV块模式表 0

4MV 编码的模式	VLC	VLC 大小
0	14	5
1	58	6
2	59	6
3	25	5
4	12	5
5	26	5
6	15	5
7	15	4
8	13	5
9	24	5
10	27	5
11	0	3
12	28	5
13	1	3
14	2	3
15	2	2

4MV块模式表 1

4MV 编码的模式	VLC	VLC 大小
0	8	4
1	18	5
2	19	5
3	4	4
4	20	5
5	5	4
6	30	5
7	11	4
8	21	5
9	31	5
10	6	4
11	12	4
12	7	4
13	13	4
14	14	4
15	0	2

4MV块模式表 2

4MV 编码的模式	VLC	VLC 大小
0	15	4
1	6	4
2	7	4
3	2	4
4	8	4
5	3	4
6	28	5
7	9	4
8	10	4
9	29	5
10	4	4
11	11	4
12	5	4
13	12	4
14	13	4
15	0	3

4MV块模式表 3

4MV 编码的模式	VLC	VLC 大小
0	0	2
1	11	4
2	12	4
3	4	4
4	13	4
5	5	4
6	30	5
7	16	5
8	14	4
9	31	5
10	6	4
11	17	5
12	7	4
13	18	5
14	19	5
15	10	4

图 47J

隔行扫描帧2 MVP块模式表0

上	下	VLC	VLC 大小
0	0	2	2
0	1	1	2
1	0	0	2
1	1	3	2

隔行扫描帧2 MVP块模式表1

上	下	VLC	VLC 大小
0	0	1	1
0	1	0	2
1	0	2	3
1	1	3	3

隔行扫描帧2 MVP块模式表2

上	下	VLC	VLC 大小
0	0	2	3
0	1	0	2
1	0	3	3
1	1	1	1

隔行扫描帧2 MVP块模式表3

上	下	VLC	VLC 大小
0	0	1	1
0	1	3	3
1	0	2	3
1	1	0	2

图 47K

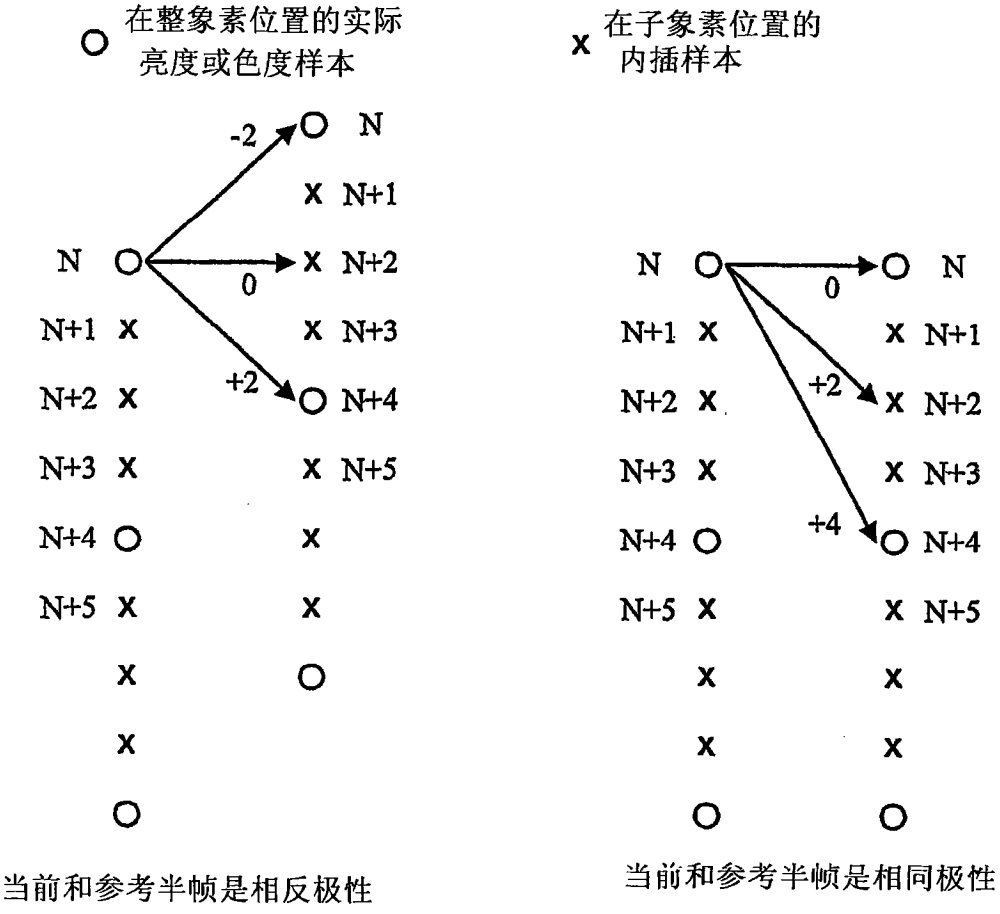

在整象素位置的实际亮度或色度样本
在子象素位置的内插样本
-2
0
+2
N
N+1
N+2
N+3
N+4
N+5
当前和参考半帧是相反极性
当前和参考半帧是相同极性

图 48

```
index = vlc_decode()    //使用由图像层中的MVTAB指示的表

if (index == 71)
{
      dmv_x = get_bits(k_x)
      dmv_y = get_bits(k_y)
}
else
{
        if (extend_x == 1)
      offset_table = offset_table2
        else
      offset_table = offset_table1
     index1 = (index + 1) % 9
        if (index1 != 0)
        {
            val = get_bits (index1 + extend_x)
            sign = 0 - (val & 1)
            dmv_x = sign ^ ((val >> 1) + offset_table[index1])
            dmv_x = dmv_x - sign
        }
        else
            dmv_x = 0

        if (extend_y == 1)
      offset_table = offset_table2
        else
      offset_table = offset_table1
     index1 = (index + 1) / 9
        if (index1 != 0)
       {
            val = get_bits (index1 + extend_y)
            sign = 0 - (val & 1)
            dmv_y = sign ^ ((val >> 1) + offset_table[index1])
            dmv_y = dmv_y - sign
        }
        else
            dmv_y = 0
}
```

图 49A

MVRANGE	k_x	k_y	range_x	range_y
0 (默认值)	9	8	256	128
10	10	9	512	256
110	12	10	2048	512
111	13	11	4096	1024

图 49B

```
index = vlc_decode()    //使用图像层中的MVTAB指示的表
picture layer
if (index == 125)
{
      dmv_x = get_bits(k_x)
      dmv_y = get_bits(k_y)
      predictor_flag = dmv_y & 1
      dmv_y = (dmv_y + predictor_flag) >> 1
}
else
{
        if (extend_x == 1)
      offset_table = offset_table2
        else
      offset_table = offset_table1
        index1 = (index + 1) % 9
        if (index1 != 0)
        {
            val = get_bits (index1 + extend_x)
            sign = 0 - (val & 1)
            dmv_x = sign ^ ((val >> 1) + offset_table[index1])
            dmv_x = dmv_x - sign
        }
        else
            dmv_x = 0

        if (extend_y == 1)
      offset_table = offset_table2
        else
      offset_table = offset_table1
        index1 = (index + 1) / 9
        if (index1 != 0)
        {
            val = get_bits (size_table[index1 + 2 * extend_y])
            sign = 0 - (val & 1)
            dmv_y = sign ^ ((val >> 1) + offset_table[index1 >> 1])
            dmv_y = dmv_y - sign
            predictor_flag = index1 & 1
        }
        else
        {
            dmv_y = 0
            predictor_flag = 0
        }
}
```

图 50

```
if (预测值A不在边界外)                    {
    if (预测值A不在边界外)                    {
    if (预测值A是帧内编码的)          {
            predictorA_x = 0
            predictorA_y = 0
        }
     if (预测值B是帧内编码的)          {
            predictorB_x = 0
            predictorB_y = 0
        }
    if (预测值C是帧内编码的)          {
            predictorC_x = 0
            predictorC_y = 0
        }
        fieldpred_x = median (predictorA_x, predictorB_x, predictorC_x)
        fieldpred_y = median (predictorA_y, predictorB_y, predictorC_y)
    }
    else {
        //预测值C在边界外
        if (每行只有1个宏块)                {
            if (预测值A是帧内编码的) {
                fieldpred_x = 0
                fieldpred_y = 0
            }
            else {
                //使用预测值A
                fieldpred_x = predictorA_x
                fieldpred_y = predictorA_y
            }
        }
        else {
            //预测值C在边界外，使用预测值和预测值B
            predictorC_x = 0
            predictorC_y = 0
         if (预测值A是帧内编码的)          {
                predictorA_x = 0
                predictorA_y = 0
            }
         if (预测值B是帧内编码的)          {
                predictorB_x = 0
                predictorB_y = 0
            }                                   在51B继续
```

图 51A

从51A继续

```
            if (预测值C是帧内编码的)  {
                predictorC_x = 0
                predictorC_y = 0
            }
            fieldpred_x = median (predictorA_x, predictorB_x, predictorC_x)
            fieldpred_y = median (predictorA_y, predictorB_y, predictorC_y)
        }
    }
}
else {
      //预测值A在边界外
      if (预测值C在边界外)          {
        fieldpred_x = 0
        fieldpred_y = 0
    }
    else {
        //使用预测值C
        fieldpred_x = predictorC_x
        fieldpred_y = predictorC_y
    }
}
```

图 51B

```
samecount = 0;
oppositecount = 0;
                    if (预测值A不在边界外) {
                        if (预测值C不在边界外) {
                if (预测值A是帧内编码的) {
                 samefieldpred_x = oppositefieldpred_x = samefieldpredA_x = oppositefieldpredA_x = 0
                 samefieldpred_y = oppositefieldpred_y = samefieldpredA_y = oppositefieldpredA_y = 0
               }
             if (预测值B是帧内编码的)          {
                    samefieldpred_x = oppositefieldpred_x = samefieldpredB_x = oppositefieldpredB_x = 0
                    samefieldpred_y = oppositefieldpred_y = samefieldpredB_y = oppositefieldpredB_y = 0
               }
            if (预测值C是帧内编码的)          {
                    samefieldpred_x = oppositefieldpred_x = samefieldpredC_x = oppositefieldpredC_x = 0
                    samefieldpred_x = oppositefieldpred_x = samefieldpredC_y = oppositefieldpredC_y = 0
               }
             if (预测值A不是帧内编码的)          {
               if (预测值A来自相同半帧)                    {
                    samecount = samecount + 1
                    samefieldpred_x = samefieldpredA_x = predictorA_x
                    samefieldpred_y = samefieldpredA_y = predictorA_y
                    oppositefieldpred_x = oppositefieldpredA_x = scaleforopposite_x(predictorA_x)
                    oppositefieldpred_y = oppositefieldpredA_y = scaleforopposite_y(predictorA_y)
                }
                else {
                    oppositecount = oppositecount + 1
                    oppositefieldpred_x = oppositefieldpredA_x = predictorA_x
                    oppositefieldpred_y = oppositefieldpredA_y = predictorA_y
                    samefieldpred_x = samefieldpredA_x = scaleforsame_x(predictorA_x)
                    samefieldpred_y = samefieldpredA_y = scaleforsame_y(predictorA_y)
                }
               }
```

在52B继续

图 52A

```
if (预测值B不是帧内编码的)        {                    从52A继续
    if (预测值B来自相同半帧)                {
        samecount = samecount + 1
        samefieldpred_x = samefieldpredB_x = predictorB_x
        samefieldpred_y = samefieldpredB_y = predictorB_y
        oppositefieldpred_x = oppositefieldpredB_x = scaleforopposite_x(predictorB_x)
        oppositefieldpred_y = oppositefieldpredB_y = scaleforopposite_y(predictorB_y)
    }
    else {
        oppositecount  = oppositecount + 1
        oppositefieldpred_x = oppositefieldpredB_x = predictorB_x
        oppositefieldpred_y = oppositefieldpredB _y = predictorB_y
        samefieldpred_x = samefieldpredB_x = scaleforsame_x(predictorB_x)
        samefieldpred_y = samefieldpredB_y = scaleforsame_y(predictorB_y)
    }
}
if (预测值C不是帧内编码的)        {
    if (预测值C来自相同半帧)                {
        samecount = samecount + 1
        samefieldpred_x = samefieldpredC_x = predictorC_x
        samefieldpred_y = samefieldpredC_y = predictorC_y
        oppositefieldpred_x = oppositefieldpredC_x = scaleforopposite_x(predictorC_x)
        oppositefieldpred_y = oppositefieldpredC_y = scaleforopposite_y(predictorC_y)
    }
    else {
        oppositecount = oppositecount + 1
        oppositefieldpred_x  = oppositefieldpredC_x = predictorC_x
        oppositefieldpred_y = oppositefieldpredC _y = predictorC_y
        samefieldpred_x = samefieldpredC _x = scaleforsame_x(predictorC_x)
        samefieldpred_y = samefieldpredC _y = scaleforsame_y(predictorC_y)
    }
}                                                    在52C继续
```

图 52B

从52B继续

```
    if ((samecount + oppositecount) > 1) {
        samefieldpred_x = median (samefieldpredA_x, samefieldpredB_x, samefieldpredC_x)
        samefieldpred_y = median (samefieldpredA_y, samefieldpredA_y, samefieldpredC_y)
        oppositefieldpred_x = median (oppositefieldpredA_x, oppositefieldpredB_x, oppositefieldpredC_x)
        oppositefieldpred_y = median (oppositefieldpredA_y, oppositefieldpredB_y, oppositefieldpredC_y)
    }

    if (samecount > oppositecount)
        dominantpredictor = samefield
    else
        dominantpredictor = oppositefield
}
else {
   //预测值C在边界外
   if (每行仅有1个宏块)                    {
        if (预测值A是帧内编码的)          {
            samefieldpred_x = oppositefieldpred_x = 0
            samefieldpred_y = oppositefieldpred_y = 0
            dominantpredictor = oppositefield
        }
        else {
            //使用预测值A
            if (预测值A来自相同半帧)                    {
               samefieldpred_x = predictorA_x
               samefieldpred_y = predictorA_y
               oppositefieldpred_x = scaleforopposite_x(predictorA_x)
               oppositefieldpred_y = scaleforopposite_y(predictorA_y)
               dominantpredictor = samefield
    }
```

在52D继续

图 52C

```
        else {                                                    从52C继续
            oppositefieldpred_x = predictorA_x
            oppositefieldpred_y = predictorA_y
            samefieldpred_x = scaleforsame_x(predictorA_x)
            samefieldpred_y = scaleforsame_y(predictorA_y)
            dominantpredictor = oppositefield
        }
    }
}
else {
    //预测值C出了边界，使用预测值和预测值B
    predictorC_x = 0
    predictorC_y = 0
    if (预测值A是帧内编码的)      {
        samefieldpred_x = oppositefieldpred_x = samefieldpredA_x = oppositefieldpredA_x  = 0
        samefieldpred_y = oppositefieldpred_y = samefieldpredA_y = oppositefieldpredA_y  = 0
    }
    if (预测值B是帧内编码的)      {
        samefieldpred_x = oppositefieldpred_x = samefieldpredB_x = oppositefieldpredB_x  = 0
        samefieldpred_y = oppositefieldpred_y = samefieldpredB_y = oppositefieldpredB_y  = 0
    }
    if (预测值C是帧内编码的)      {
        samefieldpred_x = oppositefieldpred_x = samefieldpredC_x = oppositefieldpredC_x = 0
        samefieldpred_x = oppositefieldpred_x = samefieldpredC_y = oppositefieldpredC_y  = 0
    }
    if (预测值A不是帧内编码的)        {
        if (预测值A来自相同半帧)                {
            samecount = samecount + 1
            samefieldpred_x = samefieldpredA_x = predictorA_x
            samefieldpred_y = samefieldpredA_y = predictorA_y
            oppositefieldpred_x = oppositefieldpredA_x = scaleforopposite_x(predictorA_x)
            oppositefieldpred_y = oppositefieldpredA_y = scaleforopposite_y(predictorA_y)
        }
                                                                  在52E继续
```

图 52D

从52D继续

```
else {
    oppositecount = oppositecount + 1
    oppositefieldpred_x = oppositefieldpredA_x = predictorA_x
    oppositefieldpred_y = oppositefieldpredA _y = predictorA_y
    samefieldpred_x = samefieldpredA_x = scaleforsame_x(predictorA_x)
    samefieldpred_y = samefieldpredA_y = scaleforsame_y(predictorA_y)
  }
}
if (预测值B不是帧内编码的)      {
  if (预测值B来自相同半帧)
    samecount  = samecount + 1
    samefieldpred_x = samefieldpredB_x = predictorB_x
    samefieldpred_y = samefieldpredB_y = predictorB_y
    oppositefieldpred_x = oppositefieldpredB_x = scaleforopposite_x(predictorB_x)
    oppositefieldpred_y = oppositefieldpredB_y = scaleforopposite_y(predictorB_y)
  }
  else {
    oppositecount = oppositecount + 1
    oppositefieldpred_x = oppositefieldpredB_x = predictorB_x
    oppositefieldpred_y = oppositefieldpredB_y = predictorB_y
    samefieldpred_x = samefieldpredB_x = scaleforsame_x(predictorB_x)
    samefieldpred_y = samefieldpredB_y = scaleforsame_y(predictorB_y)
  }
}
if ((samecount + oppositecount) > 1) {
  samefieldpred_x = median (samefieldpredA_x, samefieldpredB_x, samefieldpredC_x)
  samefieldpred_y = median (samefieldpredA_y, samefieldpredA_y, samefieldpredC_y)
  oppositefieldpred_x = median (oppositefieldpredA_x, oppositefieldpredB_x,
oppositefieldpredC_x)
    oppositefieldpred_y = median (oppositefieldpredA_y, oppositefieldpredB_y,
oppositefieldpredC_y)
  }
```

在52F继续

图 52E

从52E继续

```
            if (samecount > oppositecount)
                dominantpredictor = samefield
            else
                dominantpredictor = oppsositefield
        }
    }
}
else {
     //预测值A在边界外
    if (预测值C在边界外)                    {
        samefieldpred_x = oppositefieldpred_x = 0
        samefieldpred_y = oppositefieldpred_y = 0
        dominantpredictor = oppositefield
    }
    else {
       //使用预测值C
        if (预测值C来自相同半帧)                    {
            samefieldpred_x = predictorC_x
            samefieldpred_y = predictorC_y
            oppositefieldpred_x = scaleforopposite_x(predictorC_x)
            oppositefieldpred_y = scaleforopposite_y(predictorC_y)
            dominantpredictor = samefield
        }
        else {
            oppositefieldpred_x = predictorC_x
            oppositefieldpred_y = predictorC_y
            samefieldpred_x = scaleforsame_x(predictorC_x)
            samefieldpred_y = scaleforsame_y(predictorC_y)
            dominantpredictor = oppositefield
        }
    }
}
```

图 52F

```
scaleforopposite_x (n)  {
      int scaledvalue
      scaledvalue = (n * SCALEOPP) >> 8
      return scaledvalue
}

scaleforopposite_y (n)  {
      int scaledvalue
      if (当前半帧是上半帧)
            scaledvalue = ((n * SCALEOPP) >> 8) - 2
      else  //当前半帧是下半帧
            scaledvalue = ((n * SCALEOPP) >> 8) + 2
      return scaledvalue
}

scaleforsame_x (n)  {
      if (abs (n) < SCALEZONE1_X)
            scaledvalue = (n * SCALESAME1) >> 8
      else {
            if (n < 0)
                  scaledvalue = ((n * SCALESAME2) >> 8) - ZONE1OFFSET_X
            else
                  scaledvalue = ((n * SCALESAME2) >> 8) + ZONE1OFFSET_X
      }
      return scaledvalue
}
```

图 52G

```
scaleforsame_y (n)  {
      if (当前半帧是上半帧)          {
      if (abs (n) < SCALEZONE1_Y)
            scaledvalue = (n * SCALESAME1) >> 8
      else {
            if (n < 0)
                  scaledvalue = ((n * SCALESAME2) >> 8) - ZONE1OFFSET_Y
            else
                  scaledvalue = ((n * SCALESAME2) >> 8) + ZONE1OFFSET_Y
      }
    }
      else {       //当前半帧是下半帧
      if (abs (n) < SCALEZONE1_Y)
            scaledvalue = (n * SCALESAME1) >> 8
      else {
            if (n < 0)
                  scaledvalue = ((n * SCALESAME2) >> 8) - ZONE1OFFSET_Y
            else
                  scaledvalue = ((n * SCALESAME2) >> 8) + ZONE1OFFSET_Y
      }
    }

      return scaledvalue
}
```

图 52H

在当前半帧是第一半帧时的P半帧运动矢量预测值比例值

	参考帧距离			
	1	2	3	4或更大
SCALEOPP	128	192	213	224
SCALESAME1	512	341	307	293
SCALESAME2	219	236	242	245
SCALEZONE1_X	32	48	53	56
SCALEZONE1_Y	8	12	13	14
ZONE1OFFSET_X	37	20	14	11
ZONE1OFFSET_Y	10	5	4	3

图 52I

在当前半帧是第二半帧时的P半帧运动矢量预测值比例值

	参考帧距离			
	1	2	3	4或更大
SCALEOPP	128	64	43	32
SCALESAME1	512	1024	1536	2048
SCALESAME2	219	204	200	198
SCALEZONE1_X	32	16	11	8
SCALEZONE1_Y	8	4	3	2
ZONE1OFFSET_X	37	52	56	11
ZONE1OFFSET_Y	10	5	4	3

图 52J

```
scaleforsame_y (n)  {
    if (当前半帧是上半帧)          {
    if (abs (n) < SCALEZONE1_Y)
        scaledvalue = ((n + 2) * SCALESAME1) >> 8
    else {
        if (n < 0)
            scaledvalue = (((n + 2) * SCALESAME2) >> 8) - ZONE1OFFSET_Y
        else
            scaledvalue = (((n + 2) * SCALESAME2) >> 8) + ZONE1OFFSET_Y
    }
}
    else {      //当前半帧是下半帧
    if (abs (n) < SCALEZONE1_Y)
        scaledvalue = ((n - 2) * SCALESAME1) >> 8
    else {
        if (n < 0)
            scaledvalue = (((n - 2) * SCALESAME2) >> 8) - ZONE1OFFSET_Y
        else
            scaledvalue = (((n - 2) * SCALESAME2) >> 8) + ZONE1OFFSET_Y
    }
}
    return scaledvalue
}
```

图 52K

在当前P半帧是第一半帧时的比例参数

	参考帧距离			
	1	2	3	4或更大
SCALEOPP	128	192	213	224
SCALESAME1	512	341	307	293
SCALESAME2	219	236	242	245
SCALEZONE1_X	32 * N	48 * N	53 * N	56 * N
SCALEZONE1_Y	8 * N	12 * N	13 * N	14 * N
ZONE1OFFSET_X	37 * N	20 * N	14 * N	11 * N
ZONE1OFFSET_Y	10 * N	5 * N	4 * N	3 * N

图 52L

在当前P半帧是第二半帧时的比例参数

	参考帧距离			
	1	2	3	4或更大
SCALEOPP	128	64	43	32
SCALESAME1	512	1024	1536	2048
SCALESAME2	219	204	200	198
SCALEZONE1_X	32 * N	16 * N	11 * N	8 * N
SCALEZONE1_Y	8 * N	4 * N	3 * N	2 * N
ZONE1OFFSET_X	37 * N	52 * N	56 * N	11 * N
ZONE1OFFSET_Y	10 * N	5 * N	4 * N	3 * N

图 52M

MVRANGE	N
0或默认值	1
10	2
110	8
111	16

图 52N

```
if ((预测值A在边界外) || (预测值C在边界外)
      || (预测值A是帧内编码的) || (预测值C是帧内编码的)) {
     predictor_post_x = predictor_pre_x
     predictor_post_y = predictor_pre_y
 }
 else {
     sumA = abs(predictor_pre_x - predictorA_x) + abs(predictor_pre_y -
predictorA_y)
     sumC = abs(predictor_pre_x - predictorC_x) + abs(predictor_pre_y -
predictorC_y)
     if (sumA > 32) {
          //读下一比特以查看哪个候选预测值要使用
          if (get_bits(1) == 1) { //HYBRIDPRED字段
              //使用上预测值（预测值A）
              predictor_post_x = predictorA_x
              predictor_post_y = predictorA_y
          }
          else {
              //使用左预测值（预测值C）
              predictor_post_x = predictorC_x
              predictor_post_y = predictorC_y
          }
     }
     else if (sumC > 32){
          if (get_bits(1) == 1) {
                //使用上预测值（预测值A）
               predictor_post_x = predictorA_x
               predictor_post_y = predictorA_y
           }
           else {
                //使用左预测值（预测值C）
               predictor_post_x = predictorC_x
               predictor_post_y = predictorC_y
           }
     }
     else {
         predictor_post_x = predictor_pre_x
         predictor_post_y = predictor_pre_y
      }
 }
```

图 53

```
if (predictor_flag == 0) {
      if (dominantpredictor == samefield)
            参考来自与当前半帧相同的半帧
      else
            参考来自与当前半帧相反的半帧
}
else {
      // predictor_flag == 1
      if (dominantpredictor == samefield);
            参考来自与当前半帧相反的半帧
      else
            参考来自与当前半帧相同的半帧
}
```

图 54

```
//lmv0_x, lmv0_y是块0的运动矢量
//lmv1_x, lmv1_y是块1的运动矢量
//lmv2_x, lmv2_y是块2的运动矢量
//lmv3_x, lmv3_y是块3的运动矢量
ix = median4(lmv0_x, lmv1_x, lmv2_x, lmv3_x)
iy = median4(lmv0_y, lmv1_y, lmv2_y, lmv3_y)
cmv_x = (ix + round[ix & 3]) >> 1
cmv_y = (iy + round[iy & 3]) >> 1

其中  round[0] = 0, round[1] = 0, round[2] = 0 以及 round[3] = 1
```

图 55A

```
if (所有4个亮度块运动矢量都来自相同的半帧)
{
      //lmv0_x, lmv0_y是块0的运动矢量
      //lmv1_x, lmv1_y是块1的运动矢量
      //lmv2_x, lmv2_y是块2的运动矢量
      //lmv3_x, lmv3_y是块3的运动矢量
      ix = median4(lmv0_x, lmv1_x, lmv2_x, lmv3_x)
      iy = median4(lmv0_y, lmv1_y, lmv2_y, lmv3_y)
}
 else if (亮度块运动矢量中的3个来自相同的半帧)
{
      // lmv0_x, lmv0_y,
      // lmv1_x, lmv1_y,
      //lmv2_x, lmv2_y是来自相同半帧的3个运动矢量
      ix = median3(lmv0_x, lmv1_x, lmv2_x)
      iy = median3(lmv0_y, lmv1_y, lmv2_y)
}
else if (亮度块运动矢量中的2个来自相同的半帧)
{
      //使用来自具有与当前半帧相同极性的半帧的2个运动矢量

      // lmv0_x, lmv0_y,
      //lmv1_x, lmv1_y是具有与当前半帧相同极性的运动矢量

      ix = (lmv0_x + lmv1_x) / 2
      iy = (lmv0_y + lmv1_y) / 2
}

cmv_x = (ix + round[ix & 3]) >> 1
cmv_y = (iy + round[iy & 3]) >> 1

 其中 round[0] = 0, round[1] = 0, round[2] = 0 以及 round[3] = 1
```

图 55B

```
if (LUMSCALE1 == 0)
{
      iScale = - 64
      iShift = 255 * 64  - LUMSHIFT1 *2 * 64
             if (LUMSHIFT1 > 31)
                 iShift += 128 * 64;
}
else {
      iScale = LUMSCALE1 + 32
      if (LUMSHIFT1 > 31)
            iShift = LUMSHIFT1 * 64 - 64 * 64;
            else
            iShift = LUMSHIFT1 * 64;
        }

//建立LUT
for (i = 0; i < 256; i++)
{
      j = (iScale * i + iShift + 32) >> 6
      if (j > 255)
            j = 255
      else if (j < 0)
                  j = 0
      LUTY[i] = j
      j = (iScale * (i - 128) + 128 * 64 + 32) >>6
      if (j > 255)
            j = 255
      else if (j < 0)
            j = 0
      LUTUV[i] =  j
}
```

图 56

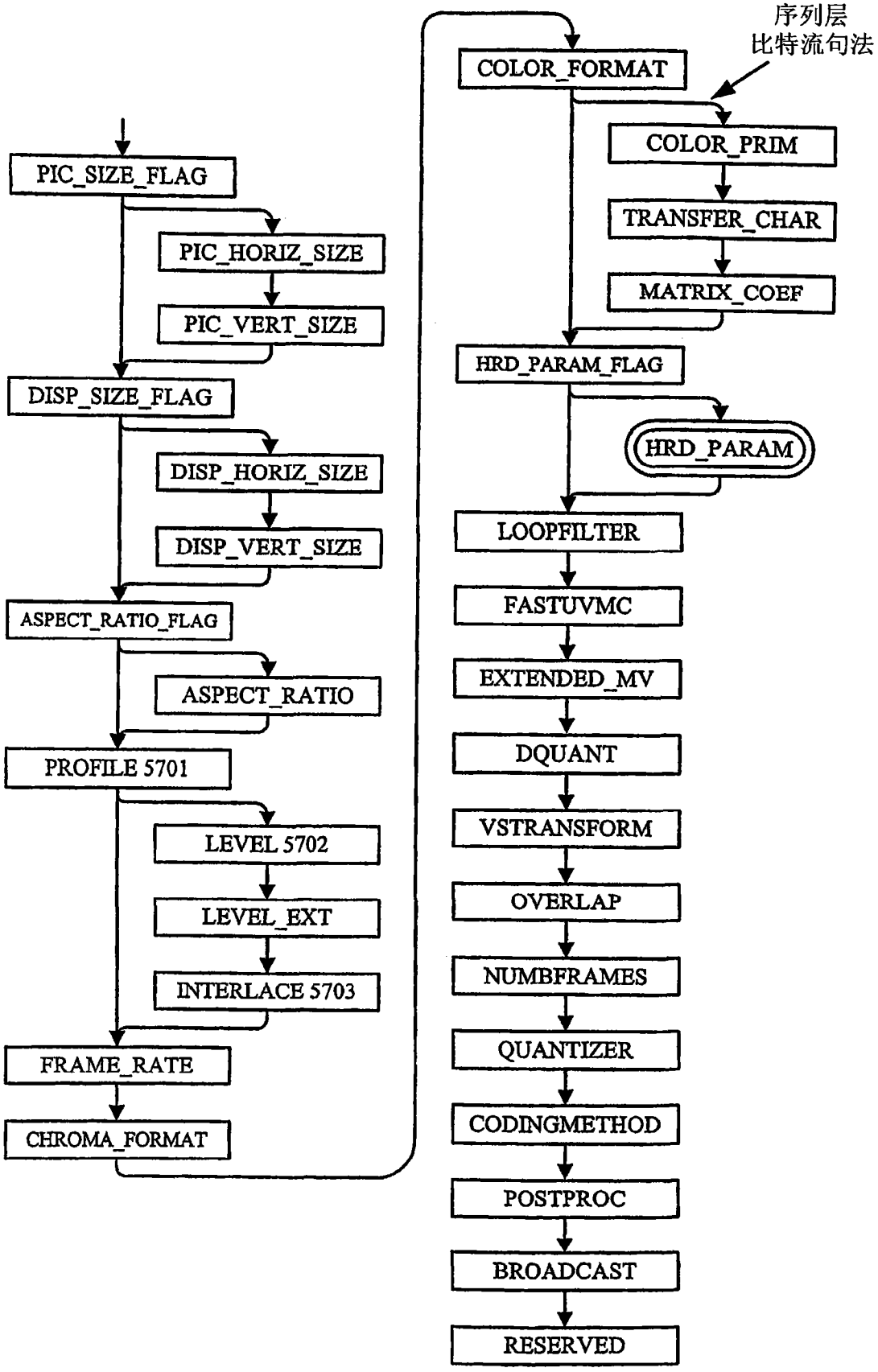

序列层
比特流句法
PIC_SIZE_FLAG
PIC_HORIZ_SIZE
PIC_VERT_SIZE
DISP_SIZE_FLAG
DISP_HORIZ_SIZE
DISP_VERT_SIZE
ASPECT_RATIO_FLAG
ASPECT_RATIO
PROFILE 5701
LEVEL 5702
LEVEL_EXT
INTERLACE 5703
FRAME_RATE
CHROMA_FORMAT
COLOR_FORMAT
COLOR_PRIM
TRANSFER_CHAR
MATRIX_COEF
HRD_PARAM_FLAG
HRD_PARAM
LOOPFILTER
FASTUVMC
EXTENDED_MV
DQUANT
VSTRANSFORM
OVERLAP
NUMBFRAMES
QUANTIZER
CODINGMETHOD
POSTPROC
BROADCAST
RESERVED

图 57A

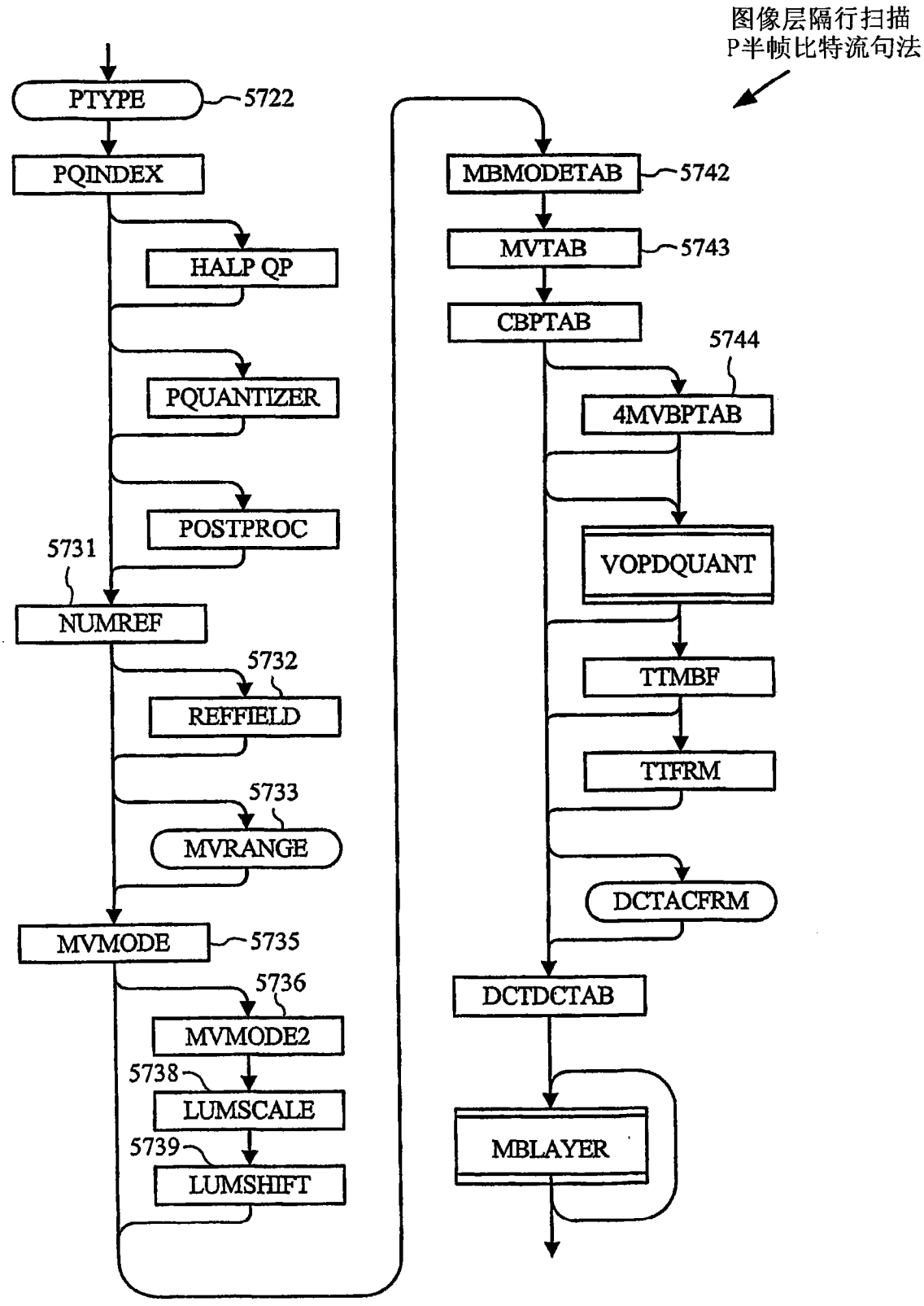

图像层隔行扫描
P半帧比特流句法
PTYPE
5722
PQINDEX
HALP QP
PQUANTIZER
POSTPROC
5731
NUMREF
5732
REFFIELD
5733
MVRANGE
MVMODE
5735
5736
MVMODE2
5738
LUMSCALE
5739
LUMSHIFT
MBMODETAB
5742
MVTAB
5743
CBPTAB
5744
4MVBPTAB
VOPDQUANT
TTMBF
TTFRM
DCTACFRM
DCTDCTAB
MBLAYER

图 57B

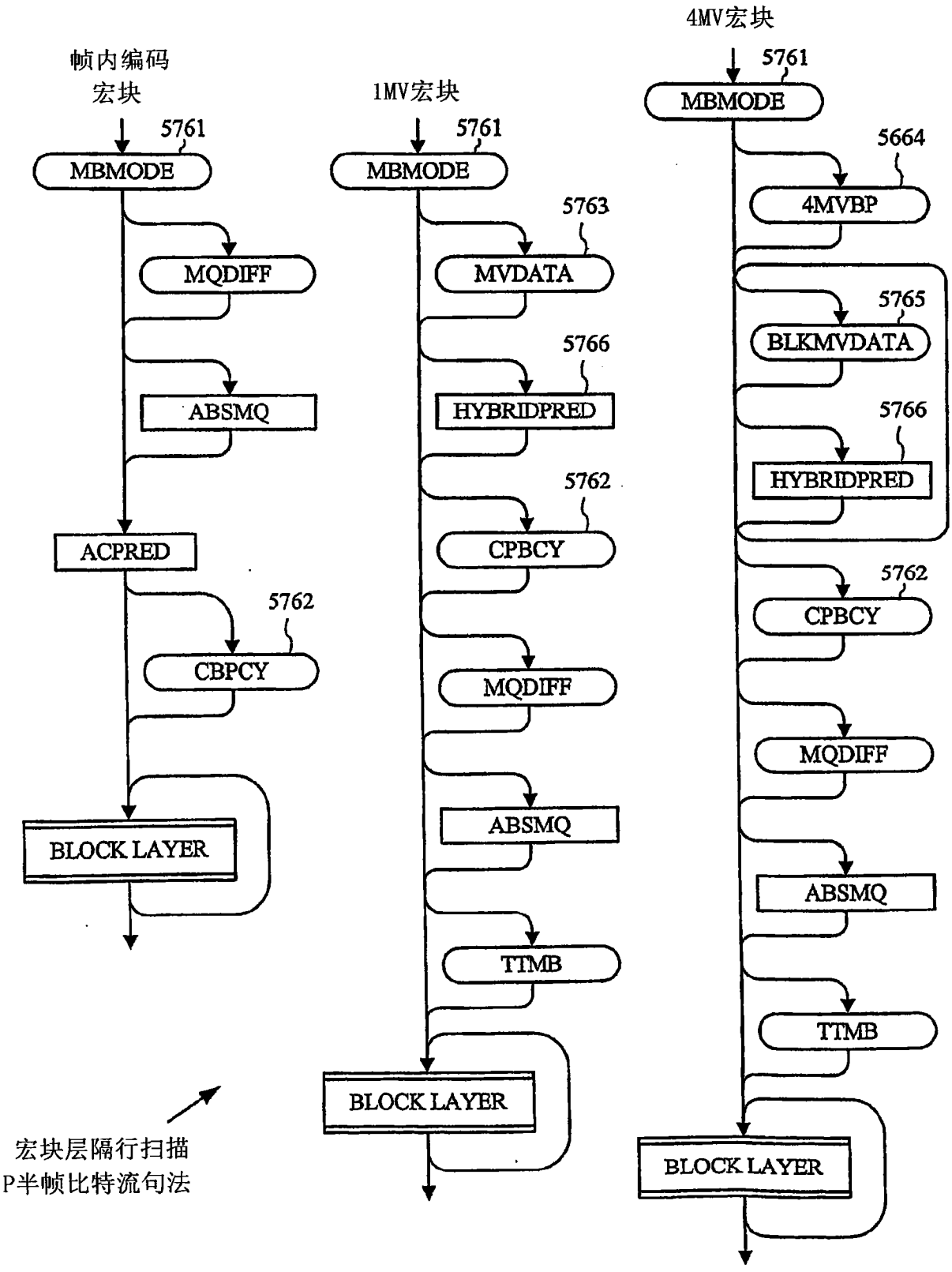

帧内编码
宏块
5761
MBMODE
MQDIFF
ABSMQ
ACPRED
5762
CBPCY
BLOCK LAYER
1MV宏块
5761
MBMODE
5763
MVDATA
5766
HYBRIDPRED
5762
CPBCY
MQDIFF
ABSMQ
TTMB
BLOCK LAYER
4MV宏块
5761
MBMODE
5664
4MVBP
5765
BLKMVDATA
5766
HYBRIDPRED
5762
CPBCY
MQDIFF
ABSMQ
TTMB
BLOCK LAYER
宏块层隔行扫描
P半帧比特流句法

图 57C

```
index = vlc_decode()   //使用图像层中的MVTAB指示的哈夫曼表

if (index == 0) {
      dmv_x = 1 - 2 * get_bits(1)
      dmv_y = 0
}
if (index == 125)
{
      dmv_x = get_bits(k_x - halfpel_flag)
      dmv_y = get_bits(k_y - halfpel_flag)
}
else
{
    index1 = (index + 1) % 9
         val = get_bits (index1)
         sign = 0 - (val & 1)
         dmv_x = sign ^ ((val >> 1) + offset_table[index1])
         dmv_x = dmv_x - sign

    index1 = (index + 1) / 9
         val = get_bits (index1)
         sign = 0 - (val & 1)
         dmv_y = sign ^ ((val >> 1) + offset_table[index1])
         dmv_y = dmv_y - sign
}
```

图 58A

MVRANGE	k_x	k_y	range_x	range_y
0 (默认值)	9	8	256	128
10	10	9	512	256
110	12	10	2048	512
111	13	11	4096	1024

图 58B

```
index = vlc_decode()   //使用图像层中的MVTAB指示的哈夫曼表

if (index == 0) {
      dmv_x = 1 - 2 * get_bits(1)
      dmv_y = 0
      predictor_flag = 0
}
if (index == 125)
{
      dmv_x = get_bits(k_x - halfpel_flag)
      dmv_y = get_bits(k_y - halfpel_flag)
      predictor_flag = dmv_y & 1
      dmv_y = dmv_y >> 1
}
else
{
    index1 = (index + 1) % 9
         val = get_bits (index1)
         sign = 0 - (val & 1)
         dmv_x = sign ^ ((val >> 1) + offset_table[index1])
         dmv_x = dmv_x - sign

    index1 = (index + 1) / 9
         val = get_bits (size_table[index1])
         sign = 0 - (val & 1)
         dmv_y = sign ^ ((val >> 1) + offset_table[index1])
         dmv_y = dmv_y - sign
         predictor_flag = index1 & 1
}
```

图 59

```
if (预测值A不在边界外)                    {
   if (预测值C不在边界外)                      {
       if (预测值A是帧内编码的)       {
            predictorA_x = 0
            predictorA_y = 0
       }
        if (预测值B是帧内编码的)      {
           predictorB_x = 0
           predictorB_y = 0
       }
        if (预测值C是帧内编码的)      {
           predictorC_x = 0
           predictorC_y = 0
       }
       fieldpred_x = median (predictorA_x, predictorB_x, predictorC_x)
       fieldpred_y = median (predictorA_y, predictorA_y, predictorC_y)
   }
   else {
       //预测值C在边界外
       if (预测值C是帧内编码的)              {
           if (预测值A是帧内编码的)       {
               fieldpred_x = oppositefieldpred_x = 0
               fieldpred_y = oppositefieldpred_y = 0
           }
           else {
               //使用预测值A
               fieldpred_x = predictorA_x
               fieldpred_y = predictorA_y
           }
       }
       else {
           //预测值C在边界外，使用预测值和预测值B
           predictorC_x = 0
           predictorC_y = 0
           if (预测值A是帧内编码的)       {
               predictorA_x = 0
               predictorA_y = 0
           }                                          在60B继续
```

图 60A

从60A继续

```
            if (预测值B是帧内编码的)      {
                predictorB_x = 0
                predictorB_y = 0
            }
            if (预测值C是帧内编码的)      {
                predictorC_x = 0
                predictorC_y = 0
            }
            fieldpred_x = median(predictorA_x, predictorB_x, predictorC_x)
            fieldpred_y = median(predictorA_y, predictorB_y, predictorC_y)
        }
    }
}
else {
    //预测值A在边界外
    if (预测值C在边界外)                  {
        fieldpred_x = 0
        fieldpred_y = 0
    }
    else {
        //使用预测值C
        samefieldpred_x = predictorC_x
        samefieldpred_y = predictorC_y
    }
}
```

图 60B

```
if (预测值A不在边界外)                    {
    if (预测值C不在边界外)                    {
        if (预测值A是帧内编码的)      {
            predictorA_x = 0
            predictorA_y = 0
        }
        if (预测值B是帧内编码的)      {
            predictorB_x = 0
            predictorB_y = 0
        }
        if (预测值C是帧内编码的)      {
            predictorC_x = 0
            predictorC_y = 0
        }
      if (预测值A来自相同的半帧)                {
            samecount = samecount + 1
            samefieldpredA_x = predictorA_x
            samefieldpredA_y = predictorA_y
            oppositefieldpredA_x = scaleforopposite_x(predictorA_x)
            oppositefieldpredA_y = scaleforopposite_y(predictorA_y)
        }
        else {
            oppositecount = oppositecount + 1
            oppositefieldpredA_x = predictorA_x
            oppositefieldpredA_y = predictorA_y
            samefieldpredA_x = scaleforsame_x(predictorA_x)
            samefieldpredA_y = scaleforsame_y(predictorA_y)
        }
```

在61B继续

图 61A

从61A继续

```
if (预测值B来自相同的半帧)                    {
    samecount = samecount + 1
    samefieldpredB_x = predictorB_x
    samefieldpredB_y = predictorB_y
    oppositefieldpredB_x = scaleforopposite_x(predictorB_x)
    oppositefieldpredB_y = scaleforopposite_y(predictorB_y)
}
else {
    oppositecount  = oppositecount + 1
    oppositefieldpredB_x = predictorB_x
    oppositefieldpredB _y = predictorB_y
    samefieldpredB_x = scaleforsame_x(predictorB_x)
    samefieldpredB_y = scaleforsame_y(predictorB_y)
}
if (预测值C来自相同的半帧)                    {
    samecount = samecount + 1
    samefieldpredC_x = predictorC_x
    samefieldpredC_y = predictorC_y
    oppositefieldpredC_x = scaleforopposite_x(predictorC_x)
    oppositefieldpredC_y = scaleforopposite_y(predictorC_y)
}
else {
    oppositecount = oppositecount + 1
    oppositefieldpredC_x = predictorC_x
    oppositefieldpredC _y = predictorC_y
    samefieldpredC _x = scaleforsame_x(predictorC_x)
    samefieldpredC _y = scaleforsame_y(predictorC_y)
}
```

在61C继续

图 61B

从61B继续

```
    samefieldpred_x = median(samefieldpredA_x, samefieldpredB_x, samefieldpredC_x)
    samefieldpred_y = median(samefieldpredA_y, samefieldpredA_y, samefieldpredC_y)
    oppositefieldpred_x = median(oppositefieldpredA_x, oppositefieldpredB_x, oppositefieldpredC_x)
    oppositefieldpred_y = median(oppositefieldpredA_y, oppositefieldpredA_y, oppositefieldpredC_y)

    if (samecount > oppositecount)
        dominantpredictor = samefield
    else
        dominantpredictor = oppsositefield
}
else {
    //预测值C在边界外
    if (每行仅1个宏块)                    {
        if (预测值A是帧内编码的)     {
            samefieldpred_x = oppositefieldpred_x = 0
            samefieldpred_y = oppositefieldpred_y = 0
            dominantpredictor = oppositefield
        }
        else {
            //使用预测值A
            if (预测值A来自相同的半帧)              {
                samefieldpred_x = predictorA_x
                samefieldpred_y = predictorA_y
                oppositefieldpred_x = scaleforopposite_x(predictorA_x)
                oppositefieldpred_y = scaleforopposite_x(predictorA_y)
                dominantpredictor = samefield
            }
            else {
                oppositefieldpred_x = predictorA_x
                oppositefieldpred_y = predictorA_y
```

在61D继续

图 61C

从61C继续

```
            samefieldpred_x = scaleforsame_x(predictorA_x)
            samefieldpred_y = scaleforsame_x(predictorA_y)
            dominantpredictor = oppositefield
        }
    }
}
else {
    //预测值C在边界外，使用预测值和预测值B
    predictorC_x = 0
    predictorC_y = 0
    if (预测值A是帧内编码的)      {
        predictorA_x = 0
        predictorA_y = 0
    }
    if (预测值B是帧内编码的)      {
        predictorB_x = 0
        predictorB_y = 0
    }
    if (预测值C是帧内编码的)      {
        predictorC_x = 0
        predictorC_y = 0
    }
    if (预测值A来自相同的半帧)              {
        samecount = samecount + 1
        samefieldpredA_x = predictorA_x
        samefieldpredA_y = predictorA_y
        oppositefieldpredA_x = scaleforopposite_x(predictorA_x)
        oppositefieldpredA_y = scaleforopposite_y(predictorA_y)
    }
```

在61E继续

图 61D

从61D继续

```
else {
    oppositecount = oppositecount + 1
    oppositefieldpredA_x = predictorA_x
    oppostiefieldpredA _y = predictorA_y
    samefieldpredA_x = scaleforsame_x(predictorA_x)
    samefieldpredA_y = scaleforsame_y(predictorA_y)
}
if (预测值B来自相同的半帧)          {
    samecount  = samecount + 1
    samefieldpredB_x = predictorB_x
    samefieldpredB_y = predictorB_y
    oppositefieldpredB_x = scaleforopposite_x(predictorB_x)
    oppostiefieldpredB_y = scaleforopposite_y(predictorB_y)
}
else {
    oppositecount = oppositecount + 1
    oppositefieldpredB_x = predictorB_x
    oppositefieldpredB_y = predictorB_y
    samefieldpredB_x = scaleforsame_x(predictorB_x)
    samefieldpredB_y = scaleforsame_y(predictorB_y)
}
samefieldpred_x = median(samefieldpredA_x, samefieldpredB_x, samefieldpredC_x)
samefieldpred_y = median(samefieldpredA_y, samefieldpredA_y, samefieldpredC_y)
oppositefieldpred_x = median(oppositefieldpredA_x,oppositefieldpredB_x,oppositefieldpredC_x)
oppositefieldpred_y = median(oppositefieldpredA_y,oppositefieldpredA_y,oppositefieldpredC_y)
if (samecount > oppositecount)
    dominantpredictor = samefield
else
    dominantpredictor = oppositefield
}
}
}
```

在61F继续

图 61E

从61E继续

```
else {
    //预测值A在边界外
    if (预测值C在边界外)                    {
        samefieldpred_x = oppositefieldpred_x = 0
        samefieldpred_y = oppositefieldpred_y = 0
        dominantpredictor = oppositefield
    }
    else {
        //使用预测值C
        if (predictorC is from same field) {
            samefieldpred_x = predictorC_x
            samefieldpred_y = predictorC_y
            oppositefieldpred_x = scaleforopposite_x(predictorC_x)
            oppositefieldpred_y = scaleforopposite_x(predictorC_y)
            dominantpredictor = samefield
        }
        else {
            oppositefieldpred_x = predictorC_x
            oppositefieldpred_y = predictorC_y
            samefieldpred_x = scaleforsame_x(predictorC_x)
            samefieldpred_y = scaleforsame_x(predictorC_y)
            dominantpredictor = oppositefield
        }
    }
}
```

图 61F