(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 113508597 B
(45)授权公告日 2023.11.21

(21)申请号 202080018115.7
(22)申请日 2020.03.02
(65)同一申请的已公布的文献号
申请公布号 CN 113508597 A
(43)申请公布日 2021.10.15
(66)本国优先权数据
PCT/CN2019/076695 2019.03.01 CN
(85)PCT国际申请进入国家阶段日
2021.09.01
(86)PCT国际申请的申请数据
PCT/CN2020/077415 2020.03.02
(87)PCT国际申请的公布数据
WO2020/177659 EN 2020.09.10
(73)专利权人 北京字节跳动网络技术有限公司
地址 100041 北京市石景山区实兴大街30号院3号楼2层B-0035房间
专利权人 字节跳动有限公司
(72)发明人 许继征 张莉 张凯 刘鸿彬 王悦
(74)专利代理机构 北京市柳沈律师事务所 11105
专利代理师 张亮
(51)Int.Cl.
H04N 19/593(2006.01)
H04N 19/52(2006.01)
(56)对比文件
CN 105532000 A,2016.04.27
CN 107925773 A,2018.04.17
US 2005129115 A1,2005.06.16
US 2016219298 A1,2016.07.28
WO 2012167713 A1,2012.12.13
WO 2013128010 A2,2013.09.06
WO 2016150343 A1,2016.09.29
WO 2017041692 A1,2017.03.16 (续)
审查员 杜乾敏

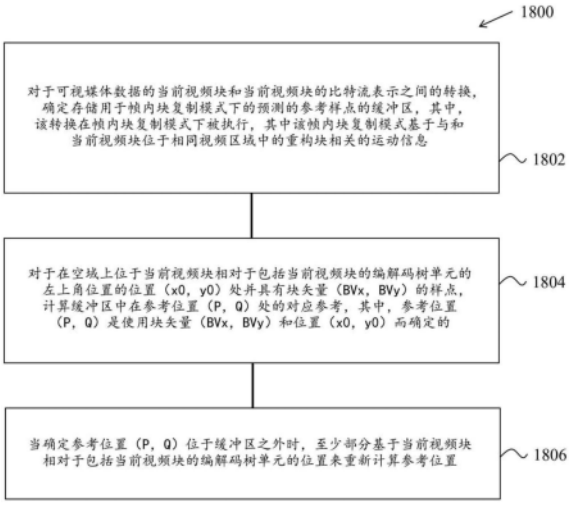
权利要求书3页 说明书77页 附图28页

(54)发明名称

用于视频编解码中的帧内块复制的基于方向的预测

(57)摘要

一种可视媒体处理的方法，包括：对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换，确定存储用于帧内块复制模式下的预测的参考样点的缓冲区；对于在空域上位于当前视频块相对于包括当前视频块的编解码树单元的左上角位置的位置处并具有块矢量的样点，计算缓冲区中在参考位置处的对应参考，其中，参考位置是使用块矢量和位置而确定的；以及当确定参考位置位于缓冲区之外时，至少部分基于当前视频块相对于包括当前视频块的编解码树单元的位置来重新计算参考位置。



CN 113508597 B

[转续页]

[接上页]

(56)对比文件

Rajan Joshi等.High Efficiency Video Coding (HEVC) Screen Content Coding.《Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 23rd Meeting: San Diego, USA, 19-26 February 2016,JCTVC-W1005-v4》.2016,全文.

1.一种可视媒体处理的方法,包括:

对于视频的当前视频块和视频的比特流的转换,确定第一编解码模式被应用于所述当前视频块;

为所述当前视频块推导第一块矢量(BVx,BVy);

基于所述第一块矢量和样点缓冲区来生成所述当前视频块的预测样点,其中没有被应用滤波操作的先前视频块的重构样点被存储在所述样点缓冲区中,并且其中在所述第一编解码模式下,所述预测样点是从包括所述当前视频块的相同图片推导的,并且所述样点缓冲区的比特深度与在所述转换期间使用的重构缓冲区的比特深度相同;以及

基于所述预测样点来执行所述转换,

其中,为了生成所述当前视频块中的第一样点(x0,y0)的预测样点,位置转换操作被应用于(x0+BVx,y0+BVy),以推导所述样点缓冲区中第一预测样点的位置,

其中,所述位置转换操作是取模操作,并且所述第一预测样点的位置由((x0+BVx)mod M,(y0+BVy)mod N)表示,并且mod是模函数,

其中,M和N与包括所述当前视频块的编解码树块的尺寸有关。

2.根据权利要求1所述的方法,其中,所述样点缓冲区以第一顺序被更新。

3.根据权利要求2所述的方法,其中,所述第一顺序是在所述转换期间样点重构的顺序。

4.根据权利要求2所述的方法,其中,所述第一顺序基于先进先出规则。

5.根据权利要求4所述的方法,在所述样点缓冲区已满的情况下,最早添加在所述样点缓冲区中的样点被最新的重构样点替换。

6.根据权利要求1所述的方法,其中,所述样点缓冲区是矩形区域,并且所述样点缓冲区的尺寸基于被包括在所述比特流中的字段来指示。

7.根据权利要求1所述的方法,其中,所述先前视频块的部分位于不同于包括所述当前视频块的当前编解码树块的编解码树块中。

8.根据权利要求1所述的方法,其中,所述转换包括将所述当前视频块编码为所述比特流。

9.根据权利要求1所述的方法,其中,所述转换包括从所述比特流解码所述当前视频块。

10.一种用于处理视频数据的装置,包括处理器和其上具有指令的非暂时性存储器,其中所述指令在由所述处理器执行时使得所述处理器:

对于视频的当前视频块和视频的比特流的转换,确定第一编解码模式被应用于所述当前视频块;

为所述当前视频块推导第一块矢量(BVx,BVy);

基于所述第一块矢量和样点缓冲区来生成所述当前视频块的预测样点,其中没有被应用滤波操作的先前视频块的重构样点被存储在所述样点缓冲区中,并且其中在所述第一编解码模式下,所述预测样点是从包括所述当前视频块的相同图片推导的,并且所述样点缓冲区的比特深度与在所述转换期间使用的重构缓冲区的比特深度相同;以及

基于所述预测样点来执行所述转换,

其中,为了生成所述当前视频块中的第一样点(x0,y0)的预测样点,位置转换操作被应

用于(x0+BVx,y0+BVy),以推导所述样点缓冲区中第一预测样点的位置,

其中,所述位置转换操作是取模操作,并且所述第一预测样点的位置由((x0+BVx)mod M,(y0+BVy)mod N)表示,并且mod是模函数,

其中,M和N与包括所述当前视频块的编解码树块的尺寸有关。

11.根据权利要求10所述的装置,其中,所述样点缓冲区以第一顺序被更新。

12.根据权利要求11所述的装置,其中,所述第一顺序是在所述转换期间样点重构的顺序。

13.根据权利要求11所述的装置,其中,所述第一顺序基于先进先出规则。

14.根据权利要求13所述的装置,在所述样点缓冲区已满的情况下,最早添加在所述样点缓冲区中的样点被最新的重构样点替换。

15.根据权利要求10所述的装置,其中,所述样点缓冲区是矩形区域,并且所述样点缓冲区的尺寸基于被包括在所述比特流中的字段来指示。

16.根据权利要求10所述的装置,其中,所述先前视频块的部分位于不同于包括所述当前视频块的当前编解码树块的编解码树块中。

17.一种存储指令的非暂时性计算机可读存储介质,其中所述指令使得处理器:

对于视频的当前视频块和视频的比特流的转换,确定第一编解码模式被应用于所述当前视频块;

为所述当前视频块推导第一块矢量(BVx,BVy);

基于所述第一块矢量和样点缓冲区来生成所述当前视频块的预测样点,其中没有被应用滤波操作的先前视频块的重构样点被存储在所述样点缓冲区中,并且其中在所述第一编解码模式下,所述预测样点是从包括所述当前视频块的相同图片推导的,并且所述样点缓冲区的比特深度与在所述转换期间使用的重构缓冲区的比特深度相同;以及

基于所述预测样点来执行所述转换,

其中,为了生成所述当前视频块中的第一样点(x0,y0)的预测样点,位置转换操作被应用于(x0+BVx,y0+BVy),以推导所述样点缓冲区中第一预测样点的位置,

其中,所述位置转换操作是取模操作,并且所述第一预测样点的位置由((x0+BVx)mod M,(y0+BVy)mod N)表示,并且mod是模函数,

其中,M和N与包括所述当前视频块的编解码树块的尺寸有关。

18.一种存储视频的比特流的方法,包括:

对于视频的当前视频块,确定第一编解码模式被应用于所述当前视频块;

为所述当前视频块推导第一块矢量(BVx,BVy);

基于所述第一块矢量和样点缓冲区来生成所述当前视频块的预测样点,其中没有被应用滤波操作的先前视频块的重构样点被存储在所述样点缓冲区中,并且其中在所述第一编解码模式下,所述预测样点是从包括所述当前视频块的相同图片推导的,并且所述样点缓冲区的比特深度与重构缓冲区的比特深度相同;

基于所述预测样点来生成所述比特流;以及

将所述比特流存储在非暂时性计算机可读记录介质中,

其中,为了生成所述当前视频块中的第一样点(x0,y0)的预测样点,位置转换操作被应用于(x0+BVx,y0+BVy),以推导所述样点缓冲区中第一预测样点的位置,

其中,所述位置转换操作是取模操作,并且所述第一预测样点的位置由((x0+BVx)mod M,(y0+BVy)mod N)表示,并且mod是模函数,并且

其中,M和N与包括所述当前视频块的编解码树块的尺寸有关。

用于视频编解码中的帧内块复制的基于方向的预测

[0001] 相关申请的交叉引用

[0002] 本申请是基于2020年3月2日提交的国际专利申请PCT/CN2020/077415,其要求2019年3月1日提交的国际专利申请PCT/CN2019/076695的优先权和利益,前述申请的全部公开通过引用而并入作为本申请的公开的一部分。

技术领域

[0003] 本专利文档涉及视频编解码和解码技术、设备以及系统。

背景技术

[0004] 尽管视频压缩有所进步,但数字视频仍占互联网和其它数字通信网络上的最大带宽使用。随着能够接收和显示视频的连接用户设备的数量增加,预计对数字视频使用的带宽需求将继续增长。

发明内容

[0005] 本文档描述了用于对视频或图像进行解码或编码的帧内块复制模式的缓冲区管理和块矢量编解码的各种实施例和技术。

[0006] 在一个示例方面,公开了一种视频或图像(可视数据)处理的方法。该方法包括:对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换,确定存储用于帧内块复制模式下的预测的参考样点的缓冲区,其中,该转换在帧内块复制模式下被执行,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息;对于在空域上位于当前视频块相对于包括当前视频块的编解码树单元的左上角位置的位置(x0,y0)处并具有块矢量(BVx,BVy)的样点,计算缓冲区中在参考位置(P,Q)处的对应参考,其中,参考位置(P,Q)是使用块矢量(BVx,BVy)和位置(x0,y0)而确定的;以及当确定参考位置(P,Q)位于缓冲区之外时,至少部分基于当前视频块相对于包括当前视频块的编解码树单元的位置来重新计算参考位置。

[0007] 在另一示例方面,公开了另一种可视数据处理的方法。该方法包括:对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换,确定存储用于帧内块复制模式下的预测的参考样点的缓冲区,其中,该转换在帧内块复制模式下被执行,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息;对于在空域上位于当前视频块相对于包括当前视频块的图片的左上角位置的位置(x,y)处并具有块矢量(BVx,BVy)的样点,至少部分基于满足与以下中的至少一个相关联的一个或多个条件来将块矢量(BVx,BVy)指定为有效: 当前视频块的位置(x,y)、当前视频块的大小、图片的大小、包括当前视频块的编解码树单元的大小、或缓冲区的大小;执行检查以确定块矢量(BVx,BVy)有效;以及当识别出块矢量(BVx,BVy)有效时,计算缓冲区中在参考位置(P,Q)处的对应参考,其中,参考位置(P,Q)是使用块矢量(BVx,BVy)、位置(x,y)以及缓冲区的大小而确定的。

[0008] 在又一示例方面，公开了另一种可视数据处理的方法。该方法包括：对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换，确定当前视频块的块矢量(BVx，BVy)或块矢量差(BVDx，BVDy)，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息；以及将块矢量(BVx，BVy)的至少一个分量或块矢量差(BVDx，BVDy)的至少一个分量归一化为位于一范围内。

[0009] 在又一示例方面，公开了另一种视频处理的方法。该方法包括：对于当前视频块和当前视频块的比特流表示之间的转换，确定用于存储用于帧内块复制模式下的预测的重构样点的缓冲区，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息；以及根据顺序来更新存储在缓冲区中的重构样点。

[0010] 在另一示例方面，公开了另一种视频处理的方法。该方法包括：执行当前视频块和当前视频块的比特流表示之间的转换，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，用于预测计算的第一精确度低于用于重构计算的第二精确度。

[0011] 在又一示例方面，公开了另一种视频处理的方法。该方法包括：使用帧内块复制模式来执行当前视频块和当前视频块的比特流表示之间的转换，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，尺寸为nM×nM的参考区域被使用，其中n和M为整数，并且其中，当前视频块位于编解码树单元中，并且其中，参考区域包括来自对应于当前视频块的编解码树单元行中的n×n个最近的可用编解码树单元的样点。

[0012] 在又一示例方面，公开了另一种视频处理的方法。该方法包括：使用帧内块复制模式来执行当前视频块和当前视频块的比特流表示之间的转换，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，尺寸为nM×pM的参考区域被使用，其中n、p和M为整数，并且其中，当前视频块位于编解码树单元中，并且其中，参考区域包括来自对应于当前视频块的编解码树单元行中的n×p-1个最近的可用编解码树单元的样点。

[0013] 在又一示例方面，公开了另一种视频处理的方法。该方法包括：使用帧内块复制模式来执行视频区域的虚拟管道数据单元(VPDU)的当前视频块和当前视频块的比特流表示之间的转换，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，尺寸为nM×nM的参考区域被使用，VPDU的尺寸为kM×kM，其中k、n和M为整数，并且其中，当前视频块位于编解码树单元中，并且其中，参考区域包括来自对应于当前视频块的编解码树单元行中的n×n-k个最近的可用编解码树单元的样点。

[0014] 在又一示例方面，公开了另一种视频处理的方法。该方法包括：对于可视媒体数据的尺寸为w×h的当前视频块和当前视频块的比特流表示之间的转换，确定存储用于帧内块复制模式下的预测的参考样点的缓冲区，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息；对于在空域上位于当前视频块相对于包括当前视频块的尺寸为M×M的编解码树单元(CTU)的左

上角位置的位置(x0,y0)处并具有块矢量(BVx,BVy)的样点,计算在缓冲区中的参考位置(P,Q)处开始的对应参考区域,其中,参考位置(P,Q)是使用块矢量(BVx,BVy)和/或位置(x0,y0)而确定的;以及将一个或多个基于规则的约束应用于参考区域和/或参考位置(P,Q),以限制参考区域与视频区域的重叠。

[0015] 在又一示例方面,公开了另一种视频处理的方法。该方法包括:对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换,确定存储用于帧内块复制模式下的预测的参考样点的缓冲区,其中,该转换在帧内块复制模式下被执行,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息;对于在空域上位于当前视频块相对于包括当前视频块的编解码单元(CU)的位置(x0,y0)处的样点,计算在缓冲区中的参考位置处开始的对应参考区域;以及调整参考区域和参考位置以确定先前处理的块中的哪些被用于预测。

[0016] 在又一示例方面,公开了另一种视频处理的方法。该方法包括:对于视频的当前视频块和当前视频块的比特流表示之间的转换,使用视频的分量X来确定与视频的分量c的当前视频块相对应的块矢量的有效性,其中,分量X不同于视频的亮度分量;以及当确定块矢量对于当前视频块有效时,使用块矢量来执行该转换,其中,该转换在帧内块复制(IBC)模式下被执行,其中该帧内块复制(IBC)模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息。

[0017] 在又一示例方面,公开了一种视频编码器或解码器装置,包括被配置为实施上述方法的处理器。

[0018] 在另一示例方面,公开了一种计算机可读程序介质。该介质存储体现用于实施所公开的方法中的一种的处理器可执行指令的代码。

[0019] 在本文档中更详细地描述了这些和其它方面。

附图说明

[0020] 图1示出了当前图片参考或帧内块复制视频或图像编解码技术的示例。

[0021] 图2示出了动态参考区域的示例。

[0022] 图3示出了对从(x,y)开始的块的编解码的示例。

[0023] 图4示出了选择先前编解码的64×64块的可能的替代方式的示例。

[0024] 图5示出了改变64×64块的编解码/解码顺序的可能的替代方式的示例。

[0025] 图6是视频或图像处理的示例方法的流程图。

[0026] 图7是用于视频或图像编解码或解码的硬件平台的框图。

[0027] 图8示出了在64×64块的解码顺序是从顶部到底部、从左到右时选择先前编解码的64×64块的另一种可能的替代方式。

[0028] 图9示出了选择先前编解码的64×64块的另一种可能的替代方式。

[0029] 图10示出了具有整形的解码过程的示例流程图。

[0030] 图11示出了在64×64块的解码顺序是从左到右、从顶部到底部时选择先前编解码的64×64块的另一种可能的替代方式。

[0031] 图12是IBC参考缓冲区状态的图示,其中块表示64×64 CTU。

[0032] 图13示出了用于IBC的参考区域的一种布置。

[0033] 图14示出了用于IBC的参考区域的另一种布置。

[0034] 图15示出了当当前虚拟管道数据单元(Virtual Pipeline Data Unit,VPDU)在图片边界的右侧时的用于IBC的参考区域的另一种布置。

[0035] 图16示出了当CTU行中的VPDU被顺序地解码时的虚拟缓冲区的状态的示例。

[0036] 图17是可以在其中实施所公开的技术的示例视频处理系统的框图。

[0037] 图18是可视数据处理的示例方法的流程图。

[0038] 图19是可视数据处理的示例方法的流程图。

[0039] 图20是可视数据处理的示例方法的流程图。

[0040] 图21是可视数据处理的示例方法的流程图。

[0041] 图22是可视数据处理的示例方法的流程图。

[0042] 图23是可视数据处理的示例方法的流程图。

[0043] 图24是可视数据处理的示例方法的流程图。

[0044] 图25是可视数据处理的示例方法的流程图。

[0045] 图26是可视数据处理的示例方法的流程图。

[0046] 图27是可视数据处理的示例方法的流程图。

[0047] 图28是可视数据处理的示例方法的流程图。

具体实施方式

[0048] 为了便于理解,在本文档中使用了章节标题,并且不将每个章节中所公开的实施例的范围仅限制于该章节。本文档描述了用于对视频或图像进行解码或编码的帧内块复制模式的缓冲区管理和块矢量编解码的各种实施例和技术。

[0049] 1. 概要

[0050] 本专利文档涉及视频编解码技术。具体地,它涉及视频编解码中的帧内块复制。它可以被应用于正在开发的标准,例如,多功能视频编解码。它也可以适用于未来的视频编解码标准或视频编解码器。

[0051] 2. 简短讨论

[0052] 视频编解码标准主要是通过众所周知的ITU-T和ISO/IEC标准的发展而演变的。ITU-T产生了H.261和H.263,ISO/IEC产生了MPEG-1和MPEG-4Visual,并且这两个组织联合产生了H.262/MPEG-2视频和264/MPEG-4高级视频编解码(Advanced Video Coding,AVC)标准和H.265/HEVC标准。自H.262以来,视频编解码标准基于混合视频编解码结构,其中利用了时域预测加变换编解码。为了探索HEVC以外的未来的视频编解码技术,VCEG和MPEG于2015年联合成立了联合视频探索小组(Joint Video Exploration Team,JVET)。此后,JVET采用了许多新方法,并将其放入到命名为联合探索模型(Joint Exploration Model,JEM)的参考软件中。2018年4月,建立了VCEG(Q6/16)和ISO/IEC JTC1 SC29/WG11(MPEG)之间的联合视频专家小组(Joint Video Expert Team,JVET)以致力于VVC标准, 目标是与HEVC相比的50%比特率降低。

[0053] 2.1 HEVC/H.265中的帧间预测

[0054] 每个帧间预测PU具有一个或两个参考图片列表的运动参数。运动参数包括运动矢量和参考图片索引。也可以使用inter_pred_idc来信令通知对两个参考图片列表之一的使

用。运动矢量可以被显式地编解码为相对于预测值的增量(delta)。

[0055] 当以跳过模式对CU进行编解码时,一个PU与CU相关联,并且没有显著的残差系数,没有编解码的运动矢量增量或参考图片索引。指定了Merge模式,由此从包括空域和时域候选的邻近PU获得当前PU的运动参数。Merge模式可以被应用于任何帧间预测PU,而不仅是针对跳过模式。Merge模式的替代方案是运动参数的显式传输,其中,运动矢量(更确切地说,与运动矢量预测值相比的运动矢量差(Motion Vector Difference,MVD))、每个参考图片列表的对应参考图片索引和参考图片列表使用按每PU被显式地信令通知。这样的模式在本公开中被称为高级运动矢量预测(Advanced Motion VectorPrediction,AMVP)。

[0056] 当信令指示将使用两个参考图片列表之一时,从一个样点块产生PU。这被称为“单向预测”。单向预测适用于P条带和B条带两者。

[0057] 当信令指示要使用两个参考图片列表时,从两个样点块产生PU。这被称为“双向预测”。双向预测仅适用于B条带。

[0058] 下文提供了关于在HEVC中指定的帧间预测模式的细节。描述将以Merge模式开始。

[0059] 2.2当前图片参考

[0060] 已经在HEVC屏幕内容编解码扩展(HEVC Screen Content Coding extension,HEVC-SCC)和当前的VVC测试模型中采用了当前图片参考(Current Picture Referencing,CPR),或一度命名为帧内块复制(Intra Block Copy,IBC)。IBC将运动补偿的概念从帧间编解码扩展到帧内编解码。如图1所示,当应用CPR时,当前块通过相同图片中的参考块进行预测。在对当前块进行编解码或解码之前,参考块中的样点必须已经被重构。尽管CPR对大多数相机捕捉的序列来说效率不高,但它示出了对屏幕内容的显著编解码增益。原因是屏幕内容图片中有许多重复图案,诸如图标和文本字符。CPR可以有效地移除这些重复图案之间的冗余。在HEVC-SCC中,如果帧间编解码的编解码单元(Coding Unit,CU)选择当前图片作为其参考图片,则它可以应用CPR。在这种情况下,MV被重命名为块矢量(Block Vector,BV),并且BV总是具有整数像素精确度。为了与主要简表(profile)HEVC兼容,当前图片在解码图片缓冲区(Decoded Picture Buffer,DPB)中被标记为“长期”参考图片。应当注意,类似地,在多视图/3D视频编解码标准中,视图间参考图片也被标记为“长期”参考图片。

[0061] 在BV找到其参考块之后,可以通过复制参考块来生成预测。残差可以通过从原始信令中减去参考像素而得到。那么变换和量化可以如在其它编解码模式中被应用。

[0062] 图1是当前图片参考的示例图示。

[0063] 然而,当参考块在图片之外、或与当前块重叠、或在重构区域之外、或在受某些约束限制的有效区域之外时,部分或全部像素值未被定义。基本上,有两种解决方案解决这样的问题。一种是不允许这种情况,例如,在比特流一致性中。另一种是对那些未定义的像素值应用填充。以下子段详细描述了解决方案。

[0064] 2.3 HEVC屏幕内容编解码扩展中的CPR

[0065] 在HEVC的屏幕内容编解码扩展中,当块使用当前图片作为参考时,它应该保证整个参考块在可用的重构区域内,如在下规格文本所指示的:

[0066] 变量offsetX和offsetY被推导如下:

[0067] offsetX＝(Chroma,ArrayType＝＝0)?0:(mvCLX[0]&0x7?2:0) (8-104)

[0068] offsetX＝(Chroma,ArrayType＝＝0)?0:(mvCLX[1]&0x7?2:0) (8-105)

[0069] 比特流一致性的一个要求是,当参考图片是当前图片时,亮度运动矢量mvLX应当遵守以下约束:

[0070] -当以设置为等于(xCb,yCb)的(xCurr,yCurr)和设置为等于(xPb+(mvLX[0]>>2)-offsetX,yPb+(mvLX[1]>>2)-offsetY)的邻近亮度位置(xNbY,yNbY)作为输入来调用如在条款6.4.1中指定的z扫描顺序块可用性的推导过程时,输出应当等于TRUE(真)。

[0071] -当以设置为等于(xCb,yCb)的(xCurr,yCurr)和设置为等于(xPb+(mvLX[0]>>2)+nPbW-1+offsetX,yPb+(mvLX[1]>>2)+nPbH-1+offsetY)的邻近亮度位置(xNbY,yNbY)作为输入来调用如在条款6.4.1中指定的z扫描顺序块可用性的推导过程时,输出应当等于TRUE。

[0072] -以下条件中的一个或两个应当为真:

[0073] -(mvLX[0]>>2)+nPbW+xB1+offsetX的值小于或等于0。

[0074] -(mvLX[1]>>2)+nPbH+yB1+offsetY的值小于或等于0。

[0075] -以下条件应当为真:

[0076] (xPb+(mvLX[0]>>2)+nPbSw-1+offsetX)/CtbSizeY-xCb/

[0077] CtbSizeY<＝yCb/CtbSizeY-(yPb+(mvLX[1]>>2)+nPbSh-1+

[0078] offsetY)/CtbSizeY (8-106)

[0079] 因此,不会发生参考块与当前块重叠或者参考块在图片之外的情况。不需要填充参考或预测块。

[0080] 2.4 CPR/IBC的示例

[0081] 在VVC测试模型中,整个参考块应该具有当前编解码树单元(Coding Tree Unit,CTU),并且不与当前块重叠。因此,不需要填充参考或预测块。

[0082] 当启用双树(dual tree)时,从亮度CTU到色度CTU,分割结构可以不同。因此,对于4:2:0颜色格式,一个色度块(例如,CU)可以对应于已经被划分为多个亮度CU的一个并置亮度区域。

[0083] 当以下条件应当为真时,色度块只能以CPR模式进行编解码:

[0084] 1)并置亮度块内的每个亮度CU都应当以CPR模式进行编解码

[0085] 2)亮度4×4块的BV中的每一个首先被转换为色度块的BV,并且色度块的BV是有效BV。

[0086] 如果两个条件中的任何一个为假(false),色度块不应以CPR模式进行编解码。

[0087] 注意,“有效BV”的定义具有以下约束:

[0088] 1)由BV标识的参考块内的所有样点应当在受限的搜索范围内(例如,在当前VVC设计中,应当在相同CTU内)。

[0089] 2)由BV标识的参考块内的所有样点都已经被重构。

[0090] 2.5 CPR/IBC的示例

[0091] 在一些示例中,用于CPR/IBC的参考区域被限制到当前CTU,其多达128×128。参考区域被动态地改变以重用内存来存储用于CPR/IBC的参考样点,使得CPR/IBC块可以具有更多的参考候选,而用于CPR/IBC的参考缓冲区可以保持或从一个CTU减少。

[0092] 图2示出了一种方法,其中块为64×64,并且CTU包含4个64×64块。当对64×64块进行编解码时,先前的3个64×64块可以被用作参考。通过这样做,解码器只需要存储4个64

×64块来支持CPR/IBC。

[0093] 假设当前亮度CU相对于图片的左上角的位置为(x,y),并且块矢量为(BVx,BVy)。在当前设计中,BV是否有效可以通过亮度位置((x+BVx)>>6<<6+(1<<7),(y+BVy)>>6<<6)还未被重构并且((x+BVx)>>6<<6+(1<<7),(y+BVy)>>6<<6)不等于(x>>6<<6,y>>6<<6)进行判断。

[0094] 2.6环路整形(In-loop Reshaping,ILR)

[0095] 环路整形(ILR)的基本思想是将原始(在第一域)信号(预测/重构信号)转换到第二域(整形域)。

[0096] 环路亮度整形器被实施为一对查找表(Look-Up Table,LUT),但是两个LUT中只有一个需要被信令通知,因为另一个LUT可以从信令通知的LUT计算。每个LUT都是一维的、10比特的、1024条目的映射表(1D-LUT)。一个LUT是正向LUT,FwdLUT,其将输入亮度码值Y_i映射到变更值Y_r:Y_r=FwdLUT[Y_i]。另一个LUT是逆向LUT,InvLUT,其将变更码值Y_r映射到$\hat{Y}_i$:

$\hat{Y}_i = InvLUT[Y_r]$($\hat{Y}_i$表示Y_i的重构值)。

[0097] 2.6.1 PWL模型

[0098] 概念性地,分段线性(Piece-Wise Linear,PWL)以以下方式被实施:

[0099] 假定x1、x2为两个输入枢轴点,并且y1、y2为它们对于一个段(piece)的对应的输出枢轴点。x1和x2之间的任何输入值x的输出值y可以通过以下等式进行插值:

[0100] y=((y2-y1)/(x2-x1))*(x-x1)+y1

[0101] 在定点实施方式中,等式可以改写为:

[0102] $y=((m*x+2^{FP\_PREC-1})>>FP\_PREC)+c$

[0103] 其中m是标量,c是偏移量,FP_PREC是指定精确度的恒定值。

[0104] 在一些示例中,PWL模型用于预先计算1024条目的FwdLUT和InvLUT映射表;但是PWL模型也允许在不预先计算LUT的情况下即时计算相同的映射值的实施方式。

[0105] 2.6.2.1亮度整形

[0106] 一种环路亮度整形的方法提供了更低复杂度的管道,该管道还消除了帧间条带重构中逐块帧内预测的解码时延。帧内预测在帧间和帧内条带两者的整形域中执行。

[0107] 无论条带类型如何,帧内预测总是在整形域中执行。在这种布置下,帧内预测可以在先前的TU重构完成之后立即开始。这样的布置还可以为帧内模式提供统一的过程,而不是依赖于条带。图10示出了基于模式的CE12-2解码过程的框图。

[0108] 针对亮度和色度残差缩放测试了16段分段线性(PWL)模型,而不是32段PWL模型。

[0109] 利用环路亮度整形器的帧间条带重构(浅绿色阴影块指示整形域中的信令:亮度残差;帧内亮度预测的;和帧内亮度重构的)。

[0110] 2.6.2.2依赖于亮度的色度残差缩放

[0111] 依赖于亮度的色度残差缩放是一个用定点整数运算实施的乘法过程。色度残差缩放补偿亮度信号与色度信号的相互作用。色度残差缩放被应用于TU级别。更具体地,以下适用:

[0112] -对于帧内,对重构亮度进行平均。

[0113] -对于帧间,对预测亮度进行平均。

[0114] 平均用于标识PWL模型中的索引。该索引标识缩放因子cScaleInv。色度残差乘以该数。

[0115] 注意，色度缩放因子是根据正向映射的预测亮度值而不是重构亮度值来计算的。

[0116] 2.6.2.3 ILR边信息的信令通知

[0117] 参数(当前)在片组头中传送(类似于ALF)。这些据称需要40-100比特。

[0118] 在一些示例中，添加的语法以斜体进行突出显示。

[0119] 7.3.2.1中序列参数集RBSP语法

[0120]

seq_parameter_set_rbsp( ) {	描述符
sps_seq_parameter_set_id	• ue(v)
…	•
sps_triangle_enabled_flag	• u(1)
sps_ladf_enabled_flag	• u(1)
if (sps_ladf_enabled_flag) {	•
sps_num_ladf_intervals_minus2	• u(2)
sps_ladf_lowest_interval_qp_offset	• se(v)
for(i = 0; i < sps_num_ladf_intervals_minus2 + 1; i++) {	•
sps_ladf_qp_offset[i]	• se(v)
sps_ladf_delta_threshold_minus1[i]	• ue(v)
}	•
}	•
sps_reshaper_enabled_flag	• ***u(1)***
rbsp_trailing_bits( )	•
}	•

[0121] 7.3.3.1中通用片组头语法

[0122]

tile_group_header( ) {	描述符
…	
if(num_tiles_in_tile_group_minus1 > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < num_tiles_in_tile_group_minus1; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if (sps_reshaper_enabled_flag) {	•
tile_group_reshaper_model_present_flag	• *u(1)*
if (tile_group_reshaper_model_present_flag)	•
tile_group_reshaper_model ( )	•

[0123]

tile_group_reshaper_enable_flag	• *u(1)*
if (tile_group_reshaper_enable_flag && (!(qtbtt_dual_tree_intra_flag && tile_group_type == I)))	•
tile_group_reshaper_chr oma_residual_scale_flag	• *u(1)*
}	•
byte_alignment( )	•
}	•

[0124] 添加新的语法表片组整形器模型：

[0125]

tile_group_reshaper_model () {	***描述符***
reshaper_model_min_bin_idx	*ue(v)*
reshaper_model_delta_max_bin_idx	*ue(v)*
reshaper_model_bin_delta_abs_cw_prec_minus1	*ue(v)*
for (i = reshaper_model_min_bin_idx; i <= reshaper_model_max_bin_idx; i++) {	
reshape_model_bin_delta_abs_CW [i]	*u(v)*
if (reshaper_model_bin_delta_abs_CW[i]) > 0)	
reshaper_model_bin_delta_sign_CW_flag[i]	*u(1)*
}	
}	

[0126] 在通用序列参数集RBSP语义中,添加以下语义：

[0127] sps_reshaper_enabled_flag等于1指定在编解码视频序列(Coded Video Sequence,CVS)中使用整形器。sps_reshaper_enabled_flag等于0指定在CVS中不使用整形器。

[0128] 在片组头语法中,添加以下语义

[0129] tile_group_reshaper_model_present_flag等于1指定tile_group_reshaper_model()存在于片组头中。tile_group_reshaper_model_present_flag等于0指定tile_group_reshaper_model()不存在于片组头中。当tile_group_reshaper_model_present_flag不存在时,它被推断为等于0。

[0130] tile_group_reshaper_enabled_flag等于1指定对当前片组启用整形器。tile_group_reshaper_enabled_flag等于0指定不对当前片组启用整形器。当tile_group_reshaper_enable_flag不存在时,它被推断为等于0。

[0131] tile_group_reshaper_chroma_residual_scale_flag等于指定对当前片组启用色度残差缩放。tile_group_reshaper_chroma_residual_scale_flag等于0指定不对当前片组启用色度残差缩放。当tile_group_reshaper_chroma_residual_scale_flag不存在时,它被推断为等于0。添加tile_group_reshaper_model()语法reshape_model_min_bin_idx指定要在整形器构建过程中使用的最小二进制位(或段)索引。reshape_model_min_bin_idx的值应在0至MaxBinIdx的范围内,包括0和MaxBinIdx。MaxBinIdx的值应当等于15。

[0132] reshape_model_delta_max_bin_idx指定最大允许的二进制位(或段)索引MaxBinIdx减去要在整形器构建过程中使用的最大二进制位索引。reshape_model_max_bin_idx的值被设置为等于MaxBinIdx-reshape_model_delta_max_bin_idx。

[0133] reshaper_model_bin_delta_abs_cw_prec_minus1加1指定用于表示语法reshape_model_bin_delta_abs_CW[i]的比特数。

[0134] reshape_model_bin_delta_abs_CW[i]指定第i个二进制位的绝对增量码字值。reshaper_model_bin_delta_sign_CW_flag[i]指定reshape_model_bin_delta_abs_CW[i]的符号,如下所示:

[0135] -如果reshape_model_bin_delta_sign_CW_flag[i]等于0,则对应的变量RspDeltaCW[i]为正值。

[0136] -否则(reshape_model_bin_delta_sign_CW_flag[i]不等于0),对应的变量RspDeltaCW[i]为负值。

[0137] 当reshape_model_bin_delta_sign_CW_flag[i]不存在时,它被推断为等于0。变量RspDeltaCW[i]＝(1 2*reshape_model_bin_delta_sign_CW[i])*reshape_model_bin_delta_abs_CW[i];

[0138] 变量RspCW[i]如以下步骤而推导:

[0139] 变量OrgCW被设置为等于(1<<$BitDepth_Y$)/(MaxBinIdx+1)。

[0140] -如果reshaper_model_min_bin_idx<＝i<＝reshaper_model_max_bin_idxRspCW[i]＝OrgCW+RspDeltaCW[i]。

[0141] -否则,RspCW[i]＝0。

[0142] 如果$BitDepth_Y$的值等于10,则RspCW[i]的值应当在32至2*OrgCW-1的范围内。

[0143] 变量InputPivot[i](i在0至MaxBinIdx+1的范围内,包括0和MaxBinIdx+1)被推导如下

[0144] InputPivot[i]＝i*OrgCW

[0145] 变量ReshapePivot[i](i在0至MaxBinIdx+1的范围内,包括0和MaxBinIdx+1)、变量ScaleCoef[i]和InvScaleCoeff[i](i在0至MaxBinIdx的范围内,包括0和MaxBinIdx)被推导如下:

```
shiftY = 14
ReshapePivot[ 0 ] = 0;
for( i = 0; i <= MaxBinIdx ; i++) {
ReshapePivot[ i + 1 ] = ReshapePivot[ i ] + RspCW[ i ]
    ScaleCoef[ i ] = ( RspCW[ i ] * (1 << shiftY) + (1 <<
```

[0146]

```
(Log2(OrgCW) - 1))) >> (Log2(OrgCW))
    if ( RspCW[ i ] == 0 )
        InvScaleCoeff[ i ] = 0
    else
        InvScaleCoeff[ i ] = OrgCW * (1 << shiftY) / RspCW[ i ]
}
```

[0147] 变量ChromaScaleCoef[i](i在0至MaxBinIdx的范围内,包括0和MaxBinIdx)被推

导如下：

[0148] ChromaResidualScaleLut[64]＝{16384,16384,16384,16384,16384,16384,16384,8192,8192,8192,8192,5461,5461,5461,5461,4096,4096,4096,4096,3277,3277,3277,3277,2731,2731,2731,2731,2341,2341,2341,2048,2048,2048,1820,1820,1820,1638,1638,1638,1638,1489,1489,1489,1489,1365,1365,1365,1365,1260,1260,1260,1260,1170,1170,1170,1170,1092,1092,1092,1092,1024,1024,1024,1024}；

[0149] shiftC＝11

[0150] -如果(RspCW[i]＝＝0)

[0151] ChromaScaleCoef[i]＝(1<<shiftC)

[0152] -否则(RspCW[i]!＝0),ChromaScaleCoef[i]＝ChromaResidualScaleLut[RspCW[i]>>1]

[0153] 2.6.2.4 ILR的使用

[0154] 在编码器侧,每个图片(或片组)首先被转换到整形域。并且所有的编解码过程都是在整形域中执行的。对于帧内预测,邻近块在整形域中;对于帧间预测,(从来自解码图片缓冲区的原始域生成的)参考块首先被转换到整形域。然后残差被生成并被编解码成比特流。

[0155] 在整个图片(或片组)完成编码/解码之后,整形域中的样点被转换到原始域,然后应用去方块滤波器和其它滤波器。

[0156] 对于以下情况,禁用对预测信号进行正向整形：

[0157] 当前块是帧内编解码的

[0158] 当前块被编解码为CPR(当前图片参考,也被称为帧内块复制,IBC)

[0159] 当前块被编解码为组合的帧间帧内模式(Combined Inter-Intra Mode,CIIP),并且对帧内预测块禁用正向整形。

[0160] 3.由各种实施例解决的问题的示例

[0161] 在CPR/IBC的当前设计中,存在一些问题。

[0162] 1)参考区域动态改变,这使得编码器/解码器处理复杂。

[0163] 2)容易生成无效块矢量并且难以检查,这使得编码器和解码器都复杂。

[0164] 3)不规则的参考区域导致块矢量的低效编解码。

[0165] 4)如何处理小于128×128的CTU尺寸不清楚。

[0166] 5)在BV是有效还是无效的确定过程中,对于色度块,判决基于亮度样点的可用性,这可能导致由于双树分割结构的错误判决。

[0167] 4.示例实施例

[0168] 在一些实施例中,常规缓冲区可以用于CPR/IBC块以得到参考。

[0169] 函数isRec(x,y)被定义为指示像素(x,y)是否已经被重构并由IBC模式参考。当(x,y)在图片外、在不同的条带/片/区块(brick)外时,isRec(x,y)返回假(false);当(x,y)未被重构时,isRec(x,y)返回假。在另一示例中,当样点(x,y)已经被重构但满足一些其它条件时,它也可以被标记为不可用,诸如在参考区域外/在不同的VPDU中,并且isRec(x,y)返回假。

[0170] 函数isRec(c,x,y)被定义为指示分量c的样点(x,y)是否可用。例如,如果样点(x,

y)还未被重构,则它被标记为不可用。在另一示例中,当样点(x,y)已经被重构但满足一些其它条件时,它也可以被标记为不可用,诸如在图片外/在不同的条带/片/区块中/在不同的VPDU中、在允许的参考区域外。当样点(x,y)不可用时,isRec(c,x,y)返回假,否则返回真(true)。

[0171] 在下面的讨论中,参考样点可以是重构样点。注意,“像素缓冲区”可以响应于“一个颜色分量的缓冲区”或“多个颜色分量的缓冲区”。

[0172] 用于CPR/IBC的参考缓冲区

[0173] 1.提出使用M×N像素缓冲区来存储用于CPR/IBC的亮度参考样点。

[0174] a.在一个示例中,缓冲区尺寸为64×64。

[0175] b.在一个示例中,缓冲区尺寸为128×128。

[0176] c.在一个示例中,缓冲区尺寸为64×128。

[0177] d.在一个示例中,缓冲区尺寸为128×64。

[0178] e.在一个示例中,N等于CTU的高度。

[0179] f.在一个示例中,N=nH,其中H是CTU的高度,n是正整数。

[0180] g.在一个示例中,M等于CTU的宽度。

[0181] h.在一个示例中,M=mW,其中W是CTU的宽度,m是正整数。i.在一个示例中,缓冲区尺寸不等于CTU尺寸,诸如96×128或128×96。

[0182] j.在一个示例中,缓冲区尺寸等于CTU尺寸。

[0183] k.在一个示例中,M=mW并且N=H,其中W和H是CTU的宽度和高度,m是正整数。

[0184] l.在一个示例中,M=W并且N=nH,其中W和H是CTU的宽度和高度,n是正整数。

[0185] m.在一个示例中,M=mW并且N=nH,其中W和H是CTU的宽度和高度,m和n是正整数。

[0186] n.在以上示例中,m和n可以取决于CTU尺寸。

[0187] i.在一个示例中,当CTU尺寸为128×128时,m=1并且n=1。

[0188] ii.在一个示例中,当CTU尺寸为64×64时,m=4并且n=1。

[0189] iii.在一个示例中,当CTU尺寸为32×32时,m=16并且n=1。

[0190] iv.在一个示例中,当CTU尺寸为16×16时,m=64并且n=1。

[0191] o.可替代地,缓冲区尺寸对应于CTU尺寸。

[0192] p.可替代地,缓冲区尺寸对应于虚拟管道数据单元(VPDU)尺寸。

[0193] q.可以从编码器向解码器信令通知M和/或N,诸如在VPS/SPS/PPS/

[0194] 图片头/条带头/片组头中。

[0195] 2.在标准中定义的不同简表/级别/层级中,M和/或N可以不同。提出使用另一个Mc×Nc像素缓冲区来存储用于CPR/IBC的色度参考样点。

[0196] a.在一个示例中,对于4:2:0视频,Mc=M/2并且Nc=N/2

[0197] b.在一个示例中,对于4:4:4视频,Mc=M并且Nc=N

[0198] c.在一个示例中,对于4:2:2视频,Mc=M并且Nc=N/2

[0199] d.可替代地,Mc和Nc可以独立于M和N。

[0200] e.在一个示例中,色度缓冲区包括两个通道,对应于Cb和Cr。

[0201] f.在一个示例中,Mc=M并且Nc=N。

[0202] 3.提出使用M×N样点缓冲区来存储用于CPR/IBC的RGB参考样点。

[0203] a.在一个示例中,缓冲区尺寸为64×64。

[0204] b.在一个示例中,缓冲区尺寸为128×128。

[0205] c.在一个示例中,缓冲区尺寸为64×128。

[0206] d.在一个示例中,缓冲区尺寸为128×64。

[0207] e.可替代地,缓冲区尺寸对应于CTU尺寸。

[0208] f.可替代地,缓冲区尺寸对应于虚拟管道数据单元(VPDU)尺寸。

[0209] 4.提出缓冲区可以存储环路滤波之前的重构像素。环路滤波可以指去方块滤波器、自适应环路滤波器(Adaptive Loop Filter,ALF)、样点自适应偏移(Sample Adaptive Offset,SAO)、跨分量ALF或任何其它滤波器。

[0210] a.在一个示例中,缓冲区可以存储当前CTU中的样点。

[0211] b.在一个示例中,缓冲区可以存储当前CTU之外的样点。

[0212] c.在一个示例中,缓冲区可以存储来自当前图片的任何部分的样点。

[0213] d.在一个示例中,缓冲区可以存储来自其它图片的样点。

[0214] 5.提出缓冲区可以存储环路滤波之后的重构像素。环路滤波可以指去方块滤波器、自适应环路滤波器(ALF)、样点自适应偏移(SAO)、跨分量ALF或任何其它滤波器。

[0215] a.在一个示例中,缓冲区可以存储当前CTU中的样点。

[0216] b.在一个示例中,缓冲区可以存储当前CTU之外的样点。

[0217] c.在一个示例中,缓冲区可以存储来自当前图片的任何部分的样点。

[0218] d.在一个示例中,缓冲区可以存储来自其它图片的样点。

[0219] 6.提出缓冲区可以存储环路滤波之前的重构样点和环路滤波之后的重构样点两者。环路滤波可以指去方块滤波器、自适应环路滤波器(ALF)、样点自适应偏移(SAO)、跨分量ALF或任何其它滤波器。

[0220] a.在一个示例中,缓冲区可以存储来自当前图片的样点和来自其它图片的样点两者,取决于这些样点的可用性。

[0221] b.在一个示例中,来自其它图片的参考样点来自环路滤波之后的重构样点。

[0222] c.在一个示例中,来自其它图片的参考样点来自环路滤波之前的重构样点。

[0223] 7.提出缓冲区存储具有给定的比特深度(bit-depth)的样点,其中该给定的比特深度可以不同于编解码视频数据的比特深度。

[0224] a.在一个示例中,重构缓冲区/编解码视频数据的比特深度大于存储在缓冲区中的IBC参考样点的比特深度。

[0225] b.在一个示例中,即使当内部比特深度不同于视频序列的输入比特深度时,诸如(10比特vs 8比特),IBC参考样点也被存储为与输入比特深度对齐。

[0226] c.在一个示例中,比特深度与重构缓冲区的比特深度相同。

[0227] d.在一个示例中,比特深度与输入图像/视频的比特深度相同。

[0228] e.在一个示例中,比特深度与预定义数字相同。

[0229] f.在一个示例中,比特深度取决于标准的简表。

[0230] g.在一个示例中,比特深度或与输出比特深度/输入比特深度/内部比特深度相比的比特深度差可以在SPS/PPS/序列头/图片头/条带头/片组头/片头或其它种类的视频数据单元中被信令通知。

[0231] h.提出的方法可以与在其它项目符合中提到的提出的缓冲区定义一起被应用,可替代地,它们也可以适用于IBC的现有设计。

[0232] i.缓冲区的每个颜色分量的比特深度可以不同。

[0233] 缓冲区初始化

[0234] 8.提出用给定值初始化缓冲区

[0235] a.在一个示例中,缓冲区用给定值进行初始化。

[0236] i.在一个示例中,给定值可以取决于输入比特深度和/或内部比特深度。

[0237] ii.在一个示例中,缓冲区用中间灰度(mid-grey)值(例如,8比特信号为128,或者10比特信号为512)进行初始化。

[0238] iii.在一个示例中,当ILR被使用时,缓冲区用forwardLUT(m)进行初始化。例如m=1<<(Bitdepth-1)。

[0239] b.可替代地,缓冲区用在SPS/VPS/APS/PPS/序列头/片组头/图片头/片/CTU/编解码单元/VPDU/区域中信令通知的值进行初始化。

[0240] c.在一个示例中,可以从先前解码的图片或条带或CTU行或CTU或CU的样点推导给定值。

[0241] d.对于不同的颜色分量,给定值可以不同。

[0242] 9.可替代地,提出用来自先前编解码的块的解码像素初始化缓冲区。

[0243] a.在一个示例中,解码像素是环路滤波之前的那些解码像素。

[0244] b.在一个示例中,当缓冲区尺寸为CTU时,如果可用,则缓冲区用先前解码的CTU的解码像素进行初始化。

[0245] c.在一个示例中,当缓冲区尺寸为64×64时,如果可用,则其缓冲区尺寸用先前解码的64×64块的解码像素进行初始化。

[0246] d.此外,可替代地,如果没有先前编解码的块可用,则可以应用项目符号8中的方法。

[0247] 对缓冲区的参考

[0248] 10.对于使用缓冲区中的像素作为参考的块,它可以使用缓冲区内的位置(x,y)来指示在哪里得到参考,其中,x=0,1,2,…,M-1;y=0,1,2,…,N-1。

[0249] 11.可替代地,参考位置可以被表示为l=y*M+x,l=0,1,…,M*N-1。

[0250] 12.将与当前CTU相关的块的左上角位置表示为(x0,y0),可以将块矢量(BVx,BVy)=(x-x0,y-y0)传送到解码器,以指示在缓冲区中在哪里得到参考。

[0251] 13.可替代地,块矢量(BVx,BVy)可以被定义为(x-x0+Tx,y-y0+Ty),其中Tx和Ty是预定义的偏移。

[0252] 14.对于任何像素(x0,y0)和(BVx,BVy),其在缓冲区中的参考可以在(x0+BVx,y0+BVy)处被找到。

[0253] a.在一个示例中,当(x0+BVx,y0+BVy)在缓冲区之外时,它将被裁剪(clip)到边界。

[0254] b.可替代地,当(x0+BVx,y0+BVy)在缓冲区之外时,其参考值被预定义为给定值,例如中间灰度。

[0255] c.可替代地,参考位置被定义为((x0+BVx)mod M,(y0+BVy)modN),使得它总是在

缓冲区内。

[0256] 15.对于任何像素(x0,y0)和(BVx,BVy),当(x0+BVx,y0+BVy)在缓冲区之外时,可以从缓冲区中的值推导其参考值。

[0257] a.在一个示例中,该值是从缓冲区中的样点((x0+BVx)mod M,(y0+BVy)mod N)推导的。

[0258] b.在一个示例中,该值是从缓冲区中的样点((x0+BVx)mod M,clip(y0+BVy,0,N-1))推导的。

[0259] c.在一个示例中,该值是从缓冲区中的样点(clip(x0+BVx,0,M-1),(y0+BVy)mod N)推导的。

[0260] d.在一个示例中,该值是从缓冲区中的样点(clip(x0+BVx,0,M-1),clip(y0+BVy,0,N-1))推导的。

[0261] 16.它可能不允许缓冲区范围之外的特定坐标。

[0262] a.在一个示例中,对于相对于CTU的左上角的任何像素(x0,y0)和块矢量(BVx,BVy),比特流约束是y0+BVy应该在范围[0,…,N-1]中。

[0263] b.在一个示例中,对于相对于CTU的左上角的任何像素(x0,y0)和块矢量(BVx,BVy),比特流约束是x0+BVx应该在范围[0,…,M-1]中。

[0264] c.在一个示例中,对于相对于CTU的左上角的任何像素(x0,y0)和块矢量(BVx,BVy),比特流约束是y0+BVy应该在范围[0,…,N-1]中,并且x0+BVx应该在范围[0,…,M-1]中。

[0265] 17.当一个块的信令通知的或推导的块矢量指向缓冲区之外的某处时,可以根据缓冲区来应用填充。

[0266] a.在一个示例中,缓冲区之外的任何样点的值用预定义值进行定义。

[0267] i.在一个示例中,该值可以是1<<(Bitdepth-1),例如,8比特信号为128,并且10比特信号为512。

[0268] ii.在一个示例中,当ILR被使用时,该值可以是forwardLUT(m)。例如m＝1<<(Bitdepth-1)。

[0269] iii.可替代地,预定义值的指示可以在SPS/PPS/序列头/图片头/条带头/片组/片/CTU/CU级别被信令通知或指示。

[0270] b.在一个示例中,缓冲区之外的任何样点被定义为缓冲区中的最近样点的值。

[0271] 18.处理缓冲区外参考的方法可以在水平和垂直方向上不同,或者可以根据当前块的位置(例如,是否更靠近图片边界)而不同。

[0272] a.在一个示例中,当y0+BVy在[0,N-1]之外时,(x0+BVx,y0+BVy)的样点值被指定为预定义值。

[0273] b.在一个示例中,当x0+BVx在[0,M-1]之外时,(x0+BVx,y0+BVy)的样点值被指定为预定义值。

[0274] c.可替代地,(x0+BVx,y0+BVy)的样点值被指定为((x0+BVx)mod M,y0+BVy)的样点值,如果((x0+BVx)mod M,y0+BVy)仍然在缓冲区之外,则其可以调用其它方法来进一步推导该值。

[0275] d.可替代地,(x0+BVx,y0+BVy)的样点值被指定为(x0+BVx,(y0+BVy)mod N)的样

点值,如果(x0+BVx,(y0+BVy)modN)仍然在缓冲区之外,则其可以调用其它方法来进一步推导该值。

[0276] 块矢量表示

[0277] 19.块矢量(BVx,BVy)的每个分量或分量中的一个可以被归一化到特定范围。

[0278] a.在一个示例中,BVx可以由(BVx mod M)替换。

[0279] b.可替代地,BVx可以由((BVx+X)mod M)-X替换,其中X是预定义值。

[0280] i.在一个示例中,X是64。

[0281] ii.在一个示例中,X是M/2;

[0282] iii.在一个示例中,X是块相对于当前CTU的水平坐标。

[0283] c.在一个示例中,BVy可以由(BVy mod N)替换。

[0284] d.可替代地,BVy可以由((BVy+Y)mod N)-Y替换,其中Y是预定义值。

[0285] i.在一个示例中,Y是64。

[0286] ii.在一个示例中,Y是N/2;

[0287] iii.在一个示例中,Y是块相对于当前CTU的垂直坐标。

[0288] 20.BVx和BVy可以具有不同的归一化范围。

[0289] 21.块矢量差(BVDx,BVDy)可以被归一化到特定范围。

[0290] a.在一个示例中,BVDx可以由(BVDx mod M)替换,其中函数mod返回余数。

[0291] b.可替代地,BVDx可以由((BVDx+X)mod M)-X替换,其中X是预定义值。

[0292] i.在一个示例中,X是64。

[0293] ii.在一个示例中,X是M/2;

[0294] c.在一个示例中,BVy可以由(BVDy mod N)替换。

[0295] d.可替代地,BVy可以由((BVDy+Y)mod N)-Y替换,其中Y是预定义值。

[0296] i.在一个示例中,Y是64。

[0297] ii.在一个示例中,Y是N/2;

[0298] 22.BVDx和BVDy可以具有不同的归一化范围。

[0299] 块矢量的有效性检查

[0300] 将IBC缓冲区的宽度和高度表示为W_{buf}和H_{buf}。对于从相对于图片的左上角的(X,Y)开始的W×H块(可以是亮度块、色度块、CU、TU、4×4、2×2或其它子块),以下可以适用于判断块矢量(BVx,BVy)是否有效。假定W_{pic}和H_{pic}为图片的宽度和高度;并且W_{ctu}和H_{ctu}是CTU的宽度和高度。函数floor(x)返回不大于x的最大整数。函数isRec(x,y)返回样点(x,y)是否已经被重构。

[0301] 23.即使任何参考位置在图片边界之外,块矢量(BVx,BVy)也可以被设置为有效。

[0302] a.在一个示例中,即使X+BVx<0,块矢量也可以被设置为有效。

[0303] b.在一个示例中,即使X+W+BVx>W_{pic},块矢量也可以被设置为有效。

[0304] c.在一个示例中,即使Y+BVy<0,块矢量也可以被设置为有效。

[0305] d.在一个示例中,即使Y+H+BVy>H_{pic},块矢量也可以被设置为有效。

[0306] 24.即使任何参考位置在当前CTU行之外,块矢量(BVx,BVy)也可以被设置为有效。

[0307] a.在一个示例中,即使Y+BVy<floor(Y/H_{ctu})*H_{ctu},块矢量也可以被设置为有效。

[0308] b.在一个示例中,即使Y+H+BVy>=floor(Y/H_{ctu})*H_{ctu}+H_{ctu},块矢量也可以被设置

为有效。

[0309] 25.即使任何参考位置在当前CTU和左侧(n-1)个CTU之外,块矢量(BVx,BVy)也可以被设置为有效,其中n是可以被用作用于IBC的参考区域的CTU(包括或不包括当前CTU)的数量。

[0310] a.在一个示例中,即使X+BVx<floor(X/W_{ctu})*W_{ctu}-(n-1)*W_{ctu},块矢量也可以被设置为有效。

[0311] b.在一个示例中,即使X+W+BVx>floor(X/W_{ctu})*W_{ctu}+W_{ctu},块矢量也可以被设置为有效。

[0312] 26.即使特定样点还未被重构,块矢量(BVx,BVy)也可以被设置为有效。

[0313] a.在一个示例中,即使isRec(X+BVx,Y+BVy)为假,块矢量也可以被设置为有效。

[0314] b.在一个示例中,即使isRec(X+BVx+W-1,Y+BVy)为假,块矢量也可以被设置为有效。

[0315] c.在一个示例中,即使isRec(X+BVx,Y+BVy+H-1)为假,块矢量也可以被设置为有效。

[0316] d.在一个示例中,即使isRec(X+BVx+W-1,Y+BVy+H-1)为假,块矢量也可以被设置为有效。

[0317] 27.当块不是CTU行中的第一个CTU时,块矢量(BVx,BVy)可以总是被设置为有效。

[0318] a.可替代地,块矢量可以总是被设置为有效。

[0319] 28.当满足所有以下3个条件时,块矢量(BVx,BVy)可以总是被设置为有效

[0320] • X+BVx>=0

[0321] • Y+BVy>=floor(Y/H_{ctu})

[0322] • isRec(X+BVx+W-1,Y+BVy+H-1)==真

[0323] a.可替代地,当针对CTU行中的第一个CTU的块满足所有三个条件时,块矢量可以总是被设置为有效。

[0324] 29.当块矢量(BVx,BVy)有效时,块的样点复制可以基于块矢量。

[0325] a.在一个示例中,样点(X,Y)的预测可以根据((X+BVx)%W_{buf},(Y+BVy)%H_{buf})。

[0326] 缓冲区更新

[0327] 30.当编解码新图片或片时,缓冲区可以被重置。

[0328] a.术语“重置”可以指缓冲区被初始化。

[0329] b.术语“重置”可以指缓冲区中的所有样点/像素被设置为给定值(例如,0或-1)。

[0330] 31.当完成VPDU的编解码时,可以用VPDU的重构值更新缓冲区。

[0331] 32.当完成CTU的编解码时,可以用CTU的重构值更新缓冲区。

[0332] a.在一个示例中,当缓冲区未满时,可以顺序地CTU接CTU地更新缓冲区。

[0333] b.在一个示例中,当缓冲区已满时,对应于最老(oldest)的CTU的缓冲区域将被更新。

[0334] c.在一个示例中,当M=mW且N=H(W和H是CTU尺寸;M和N是缓冲区尺寸)并且先前更新的区域从(kW,0)开始时,要更新的下一个起始位置将是((k+1)W mod M,0)。

[0335] 33.缓冲区可以在每个CTU行的开始处被重置。

[0336] a.可替代地,缓冲区可以在开始解码每个CTU时被重置。

[0337] b.可替代地,缓冲区可以在开始解码一个片时被重置。

[0338] c.可替代地,缓冲区可以在开始解码一个片组/图片时被重置。

[0339] 34.当完成对从(x,y)开始的块进行编解码时,将用根据块的重构更新从(x,y)开始的缓冲区的对应区域。

[0340] a.在一个示例中,(x,y)是相对于CTU的左上角的位置。

[0341] 35.当完成对相对于图片的块进行编解码时,将用根据块的重构更新缓冲区的对应区域。

[0342] a.在一个示例中,可以用相对于图片的左上角的位置(x,y)的重构像素值更新缓冲区中的位置(x mod M,y mod N)处的值。

[0343] b.在一个示例中,可以用相对于当前片的左上角的位置(x,y)的重构像素值更新缓冲区中的位置(x mod M,y mod N)处的值。

[0344] c.在一个示例中,可以用相对于当前CTU行的左上角的位置(x,y)的重构像素值更新缓冲区中的位置(x mod M,y mod N)处的值。

[0345] d.在一个示例中,可以用比特深度对齐之后的重构像素值更新缓冲区中的值。

[0346] 36.当完成对从(x,y)开始的块进行编解码时,将用根据块的重构更新从(xb,yb)开始的缓冲区的对应区域,其中(xb,yb)和(x,y)

[0347] 是两个不同坐标。

[0348] a.在一个示例中,(x,y)是与CTU的左上角相关的位置,并且(xb,yb)是(x+update_x,y+update_y),其中update_x和update_y指向缓冲区中的可更新位置。

[0349] 37.对于以上示例,块的重构值可以指示应用滤波器(例如,去方块滤波器)之前的重构值。

[0350] a.可替代地,块的重构值可以指示应用滤波器(例如,去方块滤波器)之后的重构值。

[0351] 38.当缓冲区根据重构样点而更新时,重构样点可以在被存储之前首先被修改,诸如样点比特深度可以被改变。

[0352] a.在一个示例中,缓冲区用与缓冲区的比特深度进行比特深度对齐之后的重构样点值进行更新。

[0353] b.在一个示例中,缓冲区值根据值{p+[1<<(b-1)]}>>b而更新,其中p是重构样点值,b是预定义的比特移位值。

[0354] c.在一个示例中,缓冲区值根据值clip({p+[1<<(b-1)]}>>b,0,(1<<bitdepth)-1)而更新,其中p是重构样点值,b是预定义的比特移位值,bitdepth是缓冲区比特深度。

[0355] d.在一个示例中,缓冲区值根据值{p+[1<<(b-1)-1]}>>b而更新,其中p是重构样点值,b是预定义的比特移位值。

[0356] e.在一个示例中,缓冲区值根据值clip({p+[1<<(b-1)-1]}>>b,0,(1<<bitdepth)-1)而更新,其中p是重构样点值,b是预定义的比特移位值,bitdepth是缓冲区比特深度。

[0357] f.在一个示例中,缓冲区值根据值p>>b而更新。

[0358] g.在一个示例中,缓冲区值根据值clip(p>>b,0,(1<<bitdepth)-1)而更新,其中bitdepth是缓冲区比特深度。

[0359] h.在以上示例中,b可以是重构比特深度减去输入样点比特深度。39.当使用缓冲区样点来形成预测时,可以应用预处理。

[0360] a.在一个示例中,预测值是p<<b,其中p是缓冲区中的样点值,b是预定义值。

[0361] b.在一个示例中,预测值是clip(p<<b,0,1<<bitdepth),其中bitdepth是重构样点的比特深度。

[0362] c.在一个示例中,预测值是(p<<b)+(1<<(bitdepth-1)),其中p是缓冲区中的样点值,并且b是预定义值,bitdepth是重构样点的比特深度。

[0363] d.在以上示例中,b可以是重构比特深度减去输入样点比特深度。

[0364] 40.缓冲区可以以给定的顺序被更新。

[0365] a.在一个示例中,可以顺序地更新缓冲区。

[0366] b.在一个示例中,可以根据重构的块的顺序来更新缓冲区。

[0367] 41.当缓冲区已满时,可以用最新的重构样点替换缓冲区中的样点。

[0368] a.在一个示例中,样点可以以先进先出的方式被更新。

[0369] b.在一个示例中,最老的样点将被替换。

[0370] c.在一个示例中,样点可以被指定优先级并根据优先级而替换。

[0371] d.在一个示例中,样点可以被标记为“长期”,使得其它样点将首先被替换。

[0372] e.在一个示例中,可以与块一起传送标志,以指示高优先级。

[0373] f.在一个示例中,可以与块一起传送数字,以指示优先级。

[0374] g.在一个示例中,来自具有特定特性的重构块的样点将被指定更高的优先级,使得其它样点将首先被替换。

[0375] i.在一个示例中,当在IBC模式下编解码的样点的百分比大于阈值时,块的所有样点可以被指定高优先级。

[0376] ii.在一个示例中,当在调色板(Palette)模式下编解码的样点的百分比大于阈值时,块的所有样点可以被指定高优先级。

[0377] iii.在一个示例中,当在IBC或调色板模式下编解码的样点的百分比大于阈值时,块的所有样点可以被指定高优先级。

[0378] iv.在一个示例中,当在变换跳过模式下编解码的样点的百分比大于阈值时,块的所有样点可以被指定高优先级。

[0379] v.阈值可以根据块尺寸、颜色分量、CTU尺寸而不同。

[0380] vi.阈值可以在SPS/PPS/序列头/条带头/片组/片级别/区域中被信令通知。

[0381] h.在一个示例中,缓冲区已满可能意味着缓冲区中的可用样点的数量等于或大于给定阈值。

[0382] i.在一个示例中,当缓冲区中的可用样点的数量等于或大于64×64×3个亮度样点时,缓冲区可以被确定为已满。

[0383] 可选缓冲区组合

[0384] 42.代替总是使用先前编解码的三个64×64块作为参考区域,提出基于当前块(或VPDU)的位置来自适应地改变它。

[0385] a.在一个示例中,当对64×64块进行编解码/解码时,先前3个64×64块可以被用作参考。与图2相比,可以应用先前64×64块的更多种类的组合。图2示出了先前64×64块的

不同组合的示例。

[0386] 43.代替使用z扫描顺序,反而可以利用垂直扫描顺序。

[0387] a.在一个示例中,当一个块以z扫描顺序被划分为具有索引0…3的4个VPDU时,编码/解码顺序是0、2、1、3。

[0388] b.在一个示例中,当对64×64块进行编解码/解码时,先前3个64×64块可以被用作参考。与图2相比,可以应用64×64块的更多种类的编解码/解码顺序。图4示出了64×64块的不同编解码/解码顺序的示例。

[0389] c.可替代地,以上方法可以仅被应用于屏幕内容编解码。

[0390] d.可替代地,仅当对一个片/片组/图片启用CPR时,才可以应用以上方法。

[0391] e.可替代地,仅当对一个CTU或一个CTU行启用CPR时,才可以应用以上方法。

[0392] 虚拟IBC缓冲区

[0393] 下面,VPDU的宽度和高度在亮度样点中分别被表示为W_{VPDU}(例如,64)和H_{VPDU}(例如,64)。可替代地,W_{VPDU}和/或H_{VPDU}可以表示其它视频单元(例如,CTU)的宽度和/或高度。

[0394] 44.可以维护虚拟缓冲区以保持跟踪IBC参考区域状态。

[0395] a.在一个示例中,虚拟缓冲区的尺寸为$mW_{VPDU}\times nH_{VPDU}$。

[0396] i.在一个示例中,m等于3并且n等于2。

[0397] ii.在一个示例中,m和/或n可以取决于图片分辨率、CTU尺寸。

[0398] iii.在一个示例中,m和/或n可以被信令通知或预定义。

[0399] b.在一个示例中,在以上项目符号和子项目符号中描述的方法可以被应用于虚拟缓冲区。

[0400] c.在一个示例中,相对于图片/条带/片/区块的左上角的样点(x,y)可以被映射到$(x\%(mW_{VPDU}),y\%(nH_{VPDU}))$。

[0401] 45.阵列可以用于跟踪与虚拟缓冲区相关联的每个样点的可用性。

[0402] a.在一个示例中,标志可以与虚拟缓冲区中的样点相关联,以指定缓冲区中的样点是否可以被用作IBC参考。

[0403] b.在一个示例中,包含亮度样点和色度样点的每个4×4块可以共享标志,以指示与该块相关联的任何样点是否可以被用作IBC参考。

[0404] c.在一个示例中,与3×2 VPDU相对应的阵列(例如,每个4×4块可以共享相同的可用性标志)被维护,以跟踪IBC参考样点的可用性。

[0405] d.在一个示例中,与4×2 VPDU相对应的阵列(例如,每个4×4块可以共享相同的可用性标志)被维护,以跟踪IBC参考样点的可用性。

[0406] 46.在完成对VPDU或视频单元进行解码之后,与虚拟缓冲区相关联的特定样点可以被标记为不可用于IBC参考。

[0407] a.在一个示例中,哪些样点可以被标记为不可用取决于最近解码的VPDU的位置。

[0408] b.当一个样点被标记为不可用时,根据该样点的预测不被允许。

[0409] i.可替代地,可以进一步应用其它方式(例如,使用默认值)来推导预测值以替换不可用的样点。

[0410] 47.可以记录最近解码的VPDU的位置,以帮助标识与虚拟缓冲区相关联的哪些样点可以被标记为不可用。

[0411] a.在一个示例中,在开始解码VPDU时,与虚拟缓冲区相关联的特定样点可以根据最近解码的VPDU的位置而标记为不可用。

[0412] i.在一个示例中,将(xPrevVPDU,yPrevVPDU)表示为相对于最近解码的VPDU的图片/条带/片/区块/其它视频处理单元的左上角的左上角位置,如果yPrevVPDU%(nH_{VPDU})等于0,则特定位置(x,y)可以被标记为不可用。

[0413] 1.在一个示例中,x可以在一范围内,诸如[xPrevVPDU-$2W_{VPDU}$+$2mW_{VPDU}$)%mW_{VPDU},((xPrevVPDU-$2W_{VPDU}$+$2mW_{VPDU}$)%mW_{VPDU})-1+W_{VPDU}];

[0414] 2.在一个示例中,y可以在一范围内,诸如[yPrevVPDU%(nH_{VPDU}),(yPrevVPDU%(nH_{VPDU}))-1+H_{VPDU}];

[0415] 3.在一个示例中,x可以在一范围内,诸如[xPrevVPDU-$2W_{VPDU}$+$2mW_{VPDU}$)%mW_{VPDU},((xPrevVPDU-$2W_{VPDU}$+$2mW_{VPDU}$)%mW_{VPDU})-1+W_{VPDU}],并且y可以在一范围内,诸如[yPrevVPDU%(nH_{VPDU}),(yPrevVPDU%(nH_{VPDU}))-1+H_{VPDU}]。

[0416] ii.在一个示例中,将(xPrevVPDU,yPrevVPDU)表示为相对于最近解码的VPDU的图片/条带/片/区块/其它视频处理单元的左上角的左上角位置,如果yPrevVPDU%(nH_{VPDU})不等于0,则特定位置(x,y)可以被标记为不可用。

[0417] 1.在一个示例中,x可以在一范围内,诸如[xPrevVPDU-W_{VPDU}+$2mW_{VPDU}$)%mW_{VPDU},((xPrevVPDU-W_{VPDU}+$2mW_{VPDU}$)%mW_{VPDU})-1+W_{VPDU}];

[0418] 2.在一个示例中,y可以在一范围内,诸如[yPrevVPDU%(nH_{VPDU}),(yPrevVPDU%(nH_{VPDU}))-1+H_{VPDU}];

[0419] 3.在一个示例中,x可以在一范围内,诸如[xPrevVPDU-W_{VPDU}+$2mW_{VPDU}$)%mW_{VPDU},((xPrevVPDU-W_{VPDU}+$2mW_{VPDU}$)%mW_{VPDU})-1+W_{VPDU}],并且y可以在一范围内,诸如[yPrevVPDU%(nH_{VPDU}),(yPrevVPDU%(nH_{VPDU}))-1+H_{VPDU}]。

[0420] 48.当CU包含多个VPDU时,代替根据VPDU来应用IBC参考可用性标记过程,IBC参考可用性标记过程可以根据CU。

[0421] a.在一个示例中,在开始解码包含多个VPDU的CU时,在CU内的VPDU被解码之前,IBC参考可用性标记过程可以被应用于每个VPDU。

[0422] b.在这种情况下,128×64和64×128 IBC块可能不被允许。

[0423] i.在一个示例中,128×64和64×128 CU的pred_mode_ibc_flag可能不被传送,并且可以被推断为等于0。

[0424] 49.对于参考块或子块,可能不需要检查右上角的参考可用性状态来判断与参考块相关联的块矢量是否有效。

[0425] a.在一个示例中,将仅检查块/子块的左上角、左下角和右下角,以判断块矢量是否有效。

[0426] 50.IBC缓冲区尺寸可以取决于VPDU尺寸(其中,宽度/高度由vSize表示)和/或CTB/CTU尺寸(其中,宽度/高度由ctbSize表示)。

[0427] a.在一个示例中,缓冲区的高度可以等于ctbSize。

[0428] b.在一个示例中,缓冲区的宽度可以取决于min(ctbSize,64)。

[0429] i.在一个示例中,缓冲区的宽度可以为(128*128/vSize,min(ctbSize,64))。

[0430] 51.IBC缓冲区可以包含像素范围之外的值,这指示该位置可能不可用于IBC参考,

例如,不用于预测其它样点。

[0431] a.样点值可以被设置为指示样点不可用的值。

[0432] b.在一个示例中,该值可以是-1。

[0433] c.在一个示例中,该值可以是[0,1<<(internal_bit_depth)-1]之外的任何值,其中internal_bit_depth是正整数值。例如,internal_bit_depth是用于对颜色分量的样点进行编码/解码的内部比特深度。

[0434] d.在一个示例中,该值可以是[0,1<<(input_bit_depth)-1]之外的任何值,其中input_bit_depth是正整数值。例如,input_bit_depth是用于对颜色分量的样点进行编码/解码的输入比特深度。

[0435] 52.用于IBC缓冲区中的样点的可用性标记可以取决于当前块的位置、当前块的尺寸、CTU/CTB尺寸和VPDU尺寸。在一个示例中,假定(xCb,yCb)表示块相对于图片的左上角的位置;ctbSize是CTU/CTB的尺寸(即,宽度和/或高度);vSize=min(ctbSize,64);wIbcBuf和hIbcBuf是IBC缓冲区宽度和高度。

[0436] a.在一个示例中,如果(xCb%vSize)等于0并且(yCb%vSize)等于0,则IBC缓冲区中的特定的位置集合可以被标记为不可用。

[0437] b.在一个示例中,当当前块尺寸小于VPDU尺寸,即min(ctbSize,64)时,标记为不可用的区域可以根据VPDU尺寸。

[0438] c.在一个示例中,当当前块尺寸大于VPDU尺寸,即min(ctbSize,64)时,标记为不可用的区域可以根据CU尺寸。

[0439] 53.在开始解码相对于图片的左上角位置的视频单元(例如,VPDU(xV,yV))时,IBC缓冲区中的对应位置可以被设置为像素范围之外的值。

[0440] a.在一个示例中,缓冲区中位置为(x%wIbcBuf,y%hIbcBuf)的缓冲区样点将被设置为值-1,其中x=xV,…,xV+ctbSize-1并且y=yV,…,yV+ctbSize-1。其中,wIbcBuf和hIbcBuf是IBC缓冲区宽度和高度,ctbSize是CTU/CTB的宽度。

[0441] i.在一个示例中,hIbcBuf可以等于ctbSize。

[0442] 54.比特流一致性约束可以根据IBC缓冲区中的样点的值。

[0443] a.在一个示例中,如果与IBC缓冲区中的块矢量相关联的参考块包含像素范围之外的值,则比特流可能是不合法的。

[0444] 55.可以根据IBC缓冲区中的可用性指示来设置比特流一致性约束。

[0445] a.在一个示例中,如果映射在IBC缓冲区中的任何参考样点被标记为不可用于对块进行编码/解码,则比特流可能是不合法的。

[0446] b.在一个示例中,当使用单树(singletree)时,如果映射在IBC缓冲区中用于对块进行编码/解码的任何亮度参考样点被标记为不可用,则比特流可能是不合法的。

[0447] c.一致性比特流可以满足对于IBC编解码块,相关联的块矢量可以指向映射在IBC缓冲区中的参考块,并且位于IBC缓冲区中用于对块进行编码/解码的每个亮度参考样点应当被标记为可用(例如,样点的值在范围[K0,K1]内,其中例如,K0被设置为0,并且K1被设置为(1<<BitDepth-1),其中BitDepth是内部比特深度或输入比特深度)。

[0448] 56.比特流一致性约束可以取决于分割树类型和当前CU的编解码treeType(树类型)。

[0449] a.在一个示例中,如果在高级别(例如,条带/图片/区块/片)中允许双树(dualtree)并且当前视频块(例如,CU/PU/CB/PB)用单树进行编解码,则比特流约束可能需要检查映射在IBC缓冲区中的所有分量的位置是否被标记为不可用。

[0450] b.在一个示例中,如果在高级别(例如,条带/图片/区块/片)中允许双树并且当前亮度视频块(例如,CU/PU/CB/PB)用双树进行编解码,则比特流约束可以忽略映射在IBC缓冲区中的色度分量的位置是否被标记为不可用。

[0451] i.可替代地,在这种情况下,比特流约束仍然可以检查映射在IBC缓冲区中的所有分量的位置是否被标记为不可用。

[0452] c.在一个示例中,如果使用单树,则比特流约束可以忽略映射在IBC缓冲区中的色度分量的位置是否被标记为不可用。

[0453] 对当前VTM设计的改进

[0454] 57.用于IBC的预测可能具有比重构更低的精确度。

[0455] a.在一个示例中,预测值根据值clip{{p+[1<<(b-1)]}>>b,0,(1<<bitdepth)-1}<<b,其中p是重构样点值,b是预定义的比特移位值,bitdepth是预测样点比特比特深度。

[0456] b.在一个示例中,预测值根据值clip{{p+[1<<(b-1)-1]}>>b,0,(1<<bitdepth)-1}<<b,其中p是重构样点值,b是预定义的比特移位值。

[0457] c.在一个示例中,预测值根据值((p>>b)+(1<<(bitdepth-1)))<<b,其中bitdepth是预测样点比特比特深度。

[0458] d.在一个示例中,预测值根据值(clip((p>>b),0,(1<<(bitdepth-b)))+(1<<(bitdepth-1)))<<b,其中bitdepth是预测样点比特比特深度。

[0459] e.在一个示例中,取决于ILR是否被应用,预测值以不同的方式被裁剪。

[0460] f.在以上示例中,b可以是重构比特深度减去输入样点比特深度。

[0461] g.在一个示例中,比特深度或与输出比特深度/输入比特深度/内部比特深度相比的比特深度差可以在SPS/PPS/序列头/图片头/条带头/片组头/片头或其它种类的视频数据单元中被信令通知。

[0462] 58.IBC的预测的一部分可能具有更低的精确度,并且另一部分具有与重构相同的精确度。

[0463] a.在一个示例中,允许的参考区域可以包含具有不同精确度(例如,比特深度)的样点。

[0464] b.在一个示例中,来自除正在被解码的当前64×64块之外的其它64×64块的参考具有低精确度,并且来自当前64×64块的参考具有与重构相同的精确度。

[0465] c.在一个示例中,来自除正在被解码的当前CTU之外的其它CTU的参考具有低精确度,并且来自当前CTU的参考具有与重构相同的精确度。

[0466] d.在一个示例中,来自特定的颜色分量集合的参考具有低精确度,并且来自其它颜色分量的参考具有与重构相同的精确度。

[0467] 59.当CTU尺寸为M×M并且参考区域尺寸为nM×nM时,参考区域是CTU行中最近的可用的n×n CTU。

[0468] a.在一个示例中,当参考区域尺寸为128×128并且CTU尺寸为64×64时,CTU行中最近的可用的4个CTU可以用于IBC参考。

[0469] b.在一个示例中,当参考区域尺寸为128×128并且CTU尺寸为32×32时,CTU行中最近的可用的16个CTU可以用于IBC参考。

[0470] 60.当CTU尺寸为M并且参考区域尺寸为nM时,参考区域是CTU行/

[0471] 片中最近的可用的n-1个CTU。

[0472] a.在一个示例中,当参考区域尺寸为128×128或256×64并且CTU尺寸为64×64时,CTU行中最近的可用的3个CTU可以用于IBC参考。

[0473] b.在一个示例中,当参考区域尺寸为128×128或512×32并且CTU尺寸为32×32时,CTU行中最近的可用的15个CTU可以用于IBC参考。

[0474] 61.当CTU尺寸为M,VPDU尺寸为kM,并且参考区域尺寸为nM时,参考区域是CTU行/片中最近的可用的n-k个CTU。

[0475] a.在一个示例中,CTU尺寸为64×64,VPDU尺寸也为64×64,参考区域尺寸为128×128,CTU行中最近的3个CTU可以用于IBC参考值。

[0476] b.在一个示例中,CTU尺寸为32×32,VPDU尺寸也为64×64,参考区域尺寸为128×128,CTU行中最近的(16-4)=12个CTU可以用于IBC参考。

[0477] 62.对于使用IBC的左上角为(x,y)的w×h块,存在使参考块远离特定区域以用于内存重用的约束,其中w和h是当前块的宽度和高度。

[0478] a.在一个示例中,当CTU尺寸为128×128并且(x,y)=(m×64,n×64)时,参考块不能与从((m-2)×64,n×64)开始的64×64区域重叠。

[0479] b.在一个示例中,当CTU尺寸为128×128时,参考块不能与左上角为(x-128,y)的w×h块重叠。

[0480] c.在一个示例中,当CTU尺寸为128×128时,(x+BVx,y+BVy)不能在左上角为(x-128,y)的w*h块内,其中BVx和BVy表示当前块的块矢量。

[0481] d.在一个示例中,当CTU尺寸为M×M并且IBC缓冲区尺寸为k×M×M时,参考块不能与左上角为(x-k×M,y)的w×h块重叠,其中BVx和BVy表示当前块的块矢量。

[0482] e.在一个示例中,当CTU尺寸为M×M并且IBC缓冲区尺寸为k×M×M时,(x+BVx,y+BVy)不能在左上角为(x-k×M,y)的w×h块内,其中BVx和BVy表示当前块的块矢量。

[0483] 63.当CTU尺寸不为M×M并且参考区域尺寸为nM×nM时,参考区域是CTU行中最近的可用的n×n-1 CTU。

[0484] a.在一个示例中,当参考区域尺寸为128×128并且CTU尺寸为64×64时,CTU行中最近的可用的3个CTU可以用于IBC参考。

[0485] b.在一个示例中,当参考区域尺寸为128×128并且CTU尺寸为32×32时,CTU行中最近的可用的15个CTU可以用于IBC参考。

[0486] 64.对于从(2m*64,2n*64)开始的64×64块(即,128×128 CTU中的左上角64×64块)内的CU,其IBC预测可以根据从((2m-2)*64,2n*64)开始的64×64块、从((2m-1)*64,2n*64)开始的64×64块、从((2m-1)*64,(2n+1)*64)开始的64×64块和当前64×64块中的重构样点。

[0487] 65.对于从((2m+1)*64,(2n+1)*64)开始的64×64块(即,128×128CTU中的右下角64×64块)内的CU,其IBC预测可以根据当前128×128 CTU。

[0488] 66.对于从((2m+1)*64,2n*64)开始的64×64块(即,128×128 CTU中的右上角64

×64块)内的CU,其IBC预测可以根据从((2m-1)*64,2n*64)开始的64×64块、从((2m-1)*64,(2n+1)*64)开始的64×64块、从(2m*64,2n*64)开始的64×64块和当前64×64块中的重构样点。

[0489] a.可替代地,如果从(2m*64,(2n+1)*64)开始的64×64块已经被重构,则IBC预测可以根据从((2m-1)*64,2n*64)开始的64×64块、从(2m*64,2n*64)开始的64×64块、从(2m*64,(2n+1)*64)开始的64×64块和当前64×64块中的重构样点。

[0490] 67.对于从(2m*64,(2n+1)*64)开始的64×64块(即,128×128 CTU中的左下角64×64块)内的CU,其IBC预测可以根据从((2m-1)*64,(2n+1)*64)开始的64×64块、从(2m*64,2n*64)开始的64×64块、从((2m+1)*64,2n*64)开始的64×64块和当前64×64块中的重构样点。

[0491] a.可替代地,如果从((2m+1)*64,2n*64)开始的64×64块还未被重构,则IBC预测可以根据从((2m-1)*64,2n*64)开始的64×64块、从((2m-1)*64,(2n+1)*64)开始的64×64块、从(2m*64,2n*64)开始的64×64块和当前64×64块中的重构样点。

[0492] 68.提出基于当前CU属于哪些64×64块来调整参考区域。

[0493] a.在一个示例中,对于从(x,y)开始的CU,当(y>>6)&1＝＝0时,从((x>>6<<6)-128,y>>6<<6)和((x>>6<<6)-64,y>>6<<6)开始的两个或多达两个先前的64×64块可以由IBC模式参考。

[0494] b.在一个示例中,对于从(x,y)开始的CU,当(y>>6)&1＝＝1时,从((x>>6<<6)-64,y>>6<<6)开始的一个先前的64×64块可以由IBC模式参考。

[0495] 69.对于从(x,y)开始并具有块矢量(BVx,BVy)的块,如果isRec(((x+BVx)>>6<<6)+128-(((y+BVy)>>6)&1)*64+(x%64),((y+BVy)>>6<<6)+(y%64))为真,则块矢量无效。

[0496] a.在一个示例中,该块是亮度块。

[0497] b.在一个示例中,该块是4:4:4格式的色度块。

[0498] c.在一个示例中,该块包含亮度分量和色度分量两者。

[0499] 70.对于从(x,y)开始并具有块矢量(BVx,BVy)的4:2:0格式的色度块,如果isRec(((x+BVx)>>5<<5)+64-(((y+BVy)>>5)&1)*32+(x%32),((y+BVy)>>5<<5)+(y%32))为真,则块矢量无效。

[0500] 71.BV对于分量c的块是否无效的确定可以依赖于分量X的样点的可用性,而不是仅检查亮度样点。

[0501] a.对于分量c从(x,y)开始并具有块矢量(BVx,BVy)的块,如果isRec(c,((x+BVx)>>6<<6)+128-(((y+BVy)>>6)&1)*64+(x%64),((y+BVy)>>6<<6)+(y%64))为真,则块矢量可以被视为无效。

[0502] i.在一个示例中,该块是亮度块(例如,c是亮度分量,或对于RGB编解码是G分量)。

[0503] ii.在一个示例中,该块是4:4:4格式的色度块(例如,c是cb或cr分量,或对于RGB编解码是B/R分量)。

[0504] iii.在一个示例中,可以检查亮度分量和色度分量两者的样点的可用性,例如,该块包含亮度分量和色度分量两者。

[0505] b.对于从分量c的(x,y)开始并具有块矢量(BVx,BVy)的4:2:0格式的色度块,如果isRec(c,((x+BVx)>>5<<5)+64-(((y+BVy)>>5)&1)*32+(x%32),((y+BVy)>>5<<5)+(y%

32))为真,则块矢量可以被视为无效。

[0506] c.对于从分量c的(x,y)开始并具有块矢量(BVx,BVy)的色度块或子块,如果色度分量的isRec(c,x+BVx+Chroma_CTU_size,y)为真,则块矢量可以被视为无效,其中Chroma_CTU_size是色度分量的CTU尺寸。

[0507] i.在一个示例中,对于4:2:0格式,Chroma_CTU_size可以为64。

[0508] ii.在一个示例中,色度子块可以是4:2:0格式的2×2块。

[0509] iii.在一个示例中,色度子块可以是4:4:4格式的4×4块。

[0510] iv.在一个示例中,色度子块可以对应于亮度分量中的最小CU尺寸。

[0511] 1.可替代地,色度子块可以对应于色度分量的最小CU尺寸。

[0512] 72.对于以上提到的所有项目符号,假设参考缓冲区包含多个M×M块(M=64)。然而,它可以扩展到其它情况,诸如参考缓冲区包含多个N×M块(例如,N=128,M=64)。

[0513] 73.对于以上提到的所有项目符号,可以应用进一步的限制,即参考缓冲区应该在与当前块相同的区块/片/片组/条带内。

[0514] a.在一个示例中,如果参考缓冲区的一部分在当前区块/片/片组/条带之外,则可以禁用对IBC的使用。可以跳过对IBC相关的语法元素的信令通知。

[0515] b.可替代地,如果参考缓冲区的一部分在当前区块/片/片组/条带之外,则可以仍然对一个块启用IBC,然而,与一个块相关联的块矢量可以仅指向剩余的参考缓冲区。

[0516] 74.提出将CTU/CTB行的第一个VPDU行中的K1个最近编解码的VPDU(如果可用)和CTU/CTB行的第二个VPDU行中的K2个最近编解码的VPDU(如果可用)作为用于IBC的参考区域,不包括当前VPDU。

[0517] a.在一个示例中,K1等于2并且K2等于1。

[0518] b.在一个示例中,当CTU/CTB尺寸为128×128并且VPDU尺寸为64×64时,可以应用以上方法。

[0519] c.在一个示例中,当CTU/CTB尺寸为64×64并且VPDU尺寸为64×64和/或32×32时,可以应用以上方法。

[0520] d.在一个示例中,当CTU/CTB尺寸为32×32并且VPDU尺寸为32×32或更小时,可以应用以上方法。

[0521] 75.以上方法可以被应用于不同阶段。

[0522] a.在一个示例中,块矢量(BV)的模运算(例如,a mod b)可以在BV的可用性检查过程中被调用,以决定BV是否有效。

[0523] b.在一个示例中,块矢量(BV)的模运算(例如,a mod b)可以被调用,以(例如,根据当前样点的位置和BV的求模结果)标识IBC虚拟缓冲区或重构图片缓冲区中参考样点的位置(例如,在环路滤波过程之前)。

[0524] 5.实施例

[0525] 5.1实施例#1

[0526] 用于IBC的缓冲区的实施方式描述如下:

[0527] 缓冲区尺寸为128×128。CTU尺寸也为128×128。对于CTU行中的第一个CTU的编解码,缓冲区用128(对于8比特视频信令)进行初始化。对于CTU行中的第k个CTU的编解码,缓冲区用对第(k-1)个CTU的环路滤波之前的重构进行初始化。

[0528] 图3示出了对从(x,y)开始的块的编解码的示例。

[0529] 当对从与当前CTU相关的(x,y)开始的块进行编解码时,块矢量(BVx,BVy)＝(x-x0,y-y0)被传送到解码器,以指示参考块来自IBC缓冲区中的(x0,y0)。假设块的宽度和高度分别是w和h。当完成对块的编解码时,从IBC缓冲区中的(x,y)开始的w×h区域将用块在环路滤波之前的重构进行更新。

[0530] 5.2实施例#2

[0531] 图4示出了选择先前编解码的64×64块的可能的替代方式的示例。

[0532] 5.3实施例#3

[0533] 图5示出了改变64×64块的编解码/解码顺序的可能的替代方式的示例。

[0534] 5.4实施例#4

[0535] 图8示出了在64×64块的解码顺序是从顶部到底部、从左到右时选择先前编解码的64×64块的另一种可能的替代方式。

[0536] 5.5实施例#5

[0537] 图9示出了选择先前编解码的64×64块的另一种可能的替代方式。

[0538] 5.6实施例#6

[0539] 图11示出了在64×64块的解码顺序是从左到右、从顶部到底部时选择先前编解码的64×64块的另一种可能的替代方式。

[0540] 5.7实施例#7

[0541] 假设CTU尺寸为W×W,解码器处的尺寸为mM×W并且比特深度为B的IBC缓冲区的实施方式如下。

[0542] 在开始解码CTU行时,用值(1 <<(B-1))初始化缓冲区,并且将要更新的起始点(xb,yb)设置为(0,0)。

[0543] 当从与CTU左上角相关的(x,y)开始并且尺寸为w×h的CU被解码时,从(xb+x,yb+y)开始并且尺寸为w×h的区域将用CU在比特深度与B比特对齐之后的重构像素值进行更新。

[0544] 在CTU被解码之后,要更新的起点(xb,yb)将被设置为((xb+W)modmW,0)。

[0545] 当解码具有块矢量(BVx,BVy)的IBC CU时,对于与CTU左上角相关的任何像素(x,y),在与预测信号的比特深度进行比特深度对齐之后,在位置((x+BVx)mod mW,(y+BVy)mod W)处从缓冲区提取其预测。

[0546] 在一个示例中,B被设置为7或8,而块的输出/输入比特深度可以等于10。

[0547] 5.8 实施例#8

[0548] 对于从与图片的左上角相关的(x,y)开始的亮度CU或联合亮度/色度CU 以及块矢量 (BVx, BVy), 当 isRec(((x+BVx)>>6<<6)+128-(((y+BVy)>>6)&1)*64+(x％64), ((y+BVy)>>6<<6) +(y％64))为真时,块矢量无效。

[0549] 对于从与图片的左上角相关的(x,y)开始的色度CU以及块矢量(BVx,BVy) , 当isRec(((x+BVx)>>5<<5)+64-(((y+BVy)>>5)&1)*32+(x％32),((y+BVy)>>5<<5) +(y％32))为真时,块矢量无效。

[0550] 5.9 实施例9

[0551] 对于从与图片的左上角相关的4:2:0格式的(x,y)开始的色度块或子块以及块矢

量(BVx,BVy),当isRec(c,(x+BVx+64,y+BVy)为真时,块矢量无效,其中c是色度分量。

[0552] 对于从与图片的左上角相关的4:4:4格式的(x,y)开始的色度块或子块以及块矢量(BVx,BVy),当isRec(c,(x+BVx+128,y+BVy)为真时,块矢量无效,其中c是色度分量。

[0553] 5.10 实施例#10

[0554] 对于从与图片的左上角相关的(x,y)开始的亮度CU或联合亮度/色度CU以及块矢量(BVx,BVy),当isRec(((x+BVx)>>6<<6)+128-(((y+BVy)>>6)&1)*64+(x%64),((y+BVy)>>6<<6)+(y%64))为真时,块矢量无效。

[0555] 对于从与图片的左上角相关的4:2:0格式的(x,y)开始的色度块或子块以及块矢量(BVx,BVy),当isRec(c,((x+BVx)>>5<<5)+64-(((y+BVy)>>5)&1)*32+(x%32),((y+BVy)>>5<<5)+(y%32))为真时,块矢量无效,其中c是色度分量。

[0556] 5.11实施例#11

[0557] 该实施例突出了保持CTU/CTB行的第一个VPDU行中的两个最近编解码的VPDU和第二个VPDU行中的一个最近编解码的VPDU的实施方式,不包括当前VPDU。

[0558] 当VPDU编解码顺序是从顶部到底部和从左到右时,参考区域如图13所示。

[0559] 当VPDU编解码顺序是从左到右和从顶部到底部,并且当前VPDU不在图片边界的右侧时,参考区域如图14所示。

[0560] 当VPDU编解码顺序是从左到右和从顶部到底部,并且当前VPDU在图片边界的右侧时,参考区域可以如图15所示。

[0561] 给定尺寸为w×h的亮度块(x,y),块矢量(BVx,BVy)是否有效可以通过检查以下条件进行判断:

[0562] isRec(((x+BVx+128)>>6<<6)-(refy&0x40)+(x%64),((y+BVy)>>6<<6)+(refy>>6==y>>6)?(y%64):0),其中refy=(y&0x40)?(y+BVy):(y+BVy+w-1)。

[0563] 如果以上函数返回真,则块矢量(BVx,BVy)无效,否则块矢量可能有效。

[0564] 5.12实施例#12

[0565] 如果CTU尺寸为192×128,则维护尺寸为192×128的虚拟缓冲区以跟踪用于IBC的参考样点。

[0566] 相对于图片的左上角的样点(x,y)与相对于缓冲区的左上角的位置(x%192,y%128)相关联。以下步骤示出了如何标记与用于IBC参考的虚拟缓冲区相关联的样点的可用性。

[0567] 相对于图片的左上角的位置(xPrevVPDU,yPrevVPDU)被记录,以代表最近解码的VPDU的左上角样点。

[0568] 1)在开始解码VPDU行时,缓冲区的所有位置都被标记为不可用。(xPrevVPDU,yPrevVPDU)被设置为(0,0)。

[0569] 2)在开始解码VPDU的第一个CU时,位置(x,y)(其中,x=(xPrevVPDU-2WVPDU+2mWVPDU)%(mWVPDU),..,((xPrevVPDU-2WVPDU+2mWVPDU)%(mWVPDU))-1+WVPDU;并且y=yPrevVPDU%(nHVPDU),..,(yPrevVPDU%(nHVPDU))-1+HVPDU)可以被标记为不可用。然后(xPrevVPDU,yPrevVPDU)被设置为(xCU,yCU),即CU相对于图片的左上角位置。

[0570] 3)在对CU进行解码之后,位置(x,y)(其中,x=xCU%(mWVPDU),...,(xCU+CU_width-1)%(mWVPDU)并且y=yCU%(nHVPDU),…,(yCU+CU_height-1)%(nHVPDU))被标记

为可用。

[0571] 4)对于具有块矢量(xBV,yBV)的IBC CU,如果任何位置(x,y)(其中,x＝(xCU+xBV)％(mWVPDU),...,(xCU+xBV+CU_width-1)％(mWVPDU)并且y＝(yCU+yBV)％(nHVPDU),…,(yCU+yBV+CU_height-1)％(nHVPDU))被标记为不可用,则块矢量被认为是无效的。

[0572] 图16示出了缓冲区状态以及图片中的VPDU解码状态。

[0573] 5.13实施例#13

[0574] 如果CTU尺寸为128×128或者CTU尺寸大于VPDU尺寸(例如,在当前设计中为64×64)或者CTU尺寸大于VPDU尺寸(例如,在当前设计中为64×64),则维护尺寸为192×128的虚拟缓冲区以跟踪用于IBC的参考样点。下面,当a<0时,(a％b)被定义为floor(a/b)*b,其中floor©返回不大于c的最大整数。

[0575] 相对于图片的左上角的样点(x,y)与相对于缓冲区的左上角的位置(x％192,y％128)相关联。以下步骤示出了如何标记与用于IBC参考的虚拟缓冲区相关联的样点的可用性。

[0576] 相对于图片的左上角的位置(xPrevVPDU,yPrevVPDU)被记录,以代表最近解码的VPDU的左上角样点。

[0577] 1)在开始解码VPDU行时,缓冲区的所有位置都被标记为不可用。(xPrevVPDU,yPrevVPDU)被设置为(0,0)。

[0578] 2)在开始解码VPDU的第一个CU时,

[0579] a.如果yPrevVPDU％64等于0,位置(x,y)(其中,x＝(xPrevVPDU-128)％192,..,((xPrevVPDU-128)％192)+63;并且y＝yPrevVPDU％128,..,(yPrevVPDU％128)+63)被标记为不可用。然后(xPrevVPDU,yPrevVPDU)被设置为(xCU,yCU),即CU相对于图片的左上角位置。

[0580] b.否则,位置(x,y)(其中,x＝(xPrevVPDU-64)％192,..,((xPrevVPDU-64)％192)+63;并且y＝yPrevVPDU％128,..,(yPrevVPDU％128)+63)被标记为不可用。然后(xPrevVPDU,yPrevVPDU)被设置为(xCU,yCU),即CU相对于图片的左上角位置。

[0581] 3)在对CU进行解码之后,位置(x,y)(其中,x＝xCU％192,...,(xCU+CU_width-1)％192并且y＝yCU％128,…,(yCU+CU_height-1)％128)被标记为可用。

[0582] 4)对于具有块矢量(xBV,yBV)的IBC CU,如果任何位置(x,y)(其中,x＝(xCU+xBV)％192,...,(xCU+xBV+CU_width-1)％192并且y＝(yCU+yBV)％128,…,(yCU+yBV+CU_height-1)％128)被标记为不可用,块矢量被认为是无效的。

[0583] 如果CTU尺寸为S×S,S不等于128,则假定Wbuf等于128*128/S。维护尺寸为Wbuf×S的虚拟缓冲区以跟踪用于IBC的参考样点。在这种情况下,VPDU尺寸等于CTU尺寸。

[0584] 相对于图片的左上角的位置(xPrevVPDU,yPrevVPDU)被记录,以代表最近解码的VPDU的左上角样点。

[0585] 1)在开始解码VPDU行时,缓冲区的所有位置都被标记为不可用。(xPrevVPDU,yPrevVPDU)被设置为(0,0)。

[0586] 2)在开始解码VPDU的第一个CU时,位置(x,y)(其中,x＝(xPrevVPDU-W_{buf}*S)％S,..,((xPrevVPDU-W_{buf}*S)％S)+S-1;并且y＝yPrevVPDU％S,..,(yPrevVPDU％S)+S-1)被

标记为不可用。然后(xPrevVPDU,yPrevVPDU)被设置为(xCU,yCU),即CU相对于图片的左上角位置。

[0587] 3)在对CU进行解码之后,位置(x,y)(其中,x＝xCU％(W_{buf}),...,(xCU+CU_width-1)％(W_{buf})并且y＝yCU％S,…,(yCU+CU_height-1)％S)被标记为可用。

[0588] 4)对于具有块矢量(xBV,yBV)的IBC CU,如果任何位置(x,y)(其中,x＝(xCU+xBV)％(Wbuf),...,(xCU+xBV+CU_width-1)％(Wbuf)并且y＝(yCU+yBV)％S,…,(yCU+yBV+CU_height-1)％S)被标记为不可用,则块矢量被认为是无效的。

[0589] 5.14实施例#14

[0590] 如果CTU尺寸为128×128或者CTU尺寸大于VPDU尺寸(例如,在当前设计中为64×64)或者CTU尺寸大于VPDU尺寸(例如,在当前设计中为64×64),则维护尺寸为256×128的虚拟缓冲区以跟踪用于IBC的参考样点。下面,当a<0时,(a％b)被定义为floor(a/b)*b,其中floor©返回不大于c的最大整数。

[0591] 相对于图片的左上角的样点(x,y)与相对于缓冲区的左上角的位置(x％256,y％128)相关联。以下步骤示出了如何标记与用于IBC参考的虚拟缓冲区相关联的样点的可用性。

[0592] 相对于图片的左上角的位置(xPrevVPDU,yPrevVPDU)被记录,以代表最近解码的VPDU的左上角样点。

[0593] 1)在开始解码VPDU行时,缓冲区的所有位置都被标记为不可用。(xPrevVPDU,yPrevVPDU)被设置为(0,0)。

[0594] 2)在开始解码VPDU的第一个CU时,

[0595] a.如果yPrevVPDU％64等于0,则位置(x,y)(其中,x＝(xPrevVPDU-128)％256,..,((xPrevVPDU-128)％256)+63;并且y＝yPrevVPDU％128,..,(yPrevVPDU％128)+63)被标记为不可用。然后(xPrevVPDU,yPrevVPDU)被设置为(xCU,yCU),即CU相对于图片的左上角位置。

[0596] b.否则,位置(x,y)(其中,x＝(xPrevVPDU-64)％256,..,((xPrevVPDU-64)％256)+63;并且y＝yPrevVPDU％128,..,(yPrevVPDU％128)+63)被标记为不可用。然后(xPrevVPDU,yPrevVPDU)被设置为(xCU,yCU),即CU相对于图片的左上角位置。

[0597] 3)在对CU进行解码之后,位置(x,y)(其中,x＝xCU％256,...,(xCU+CU_width-1)％256并且y＝yCU％128,…,(yCU+CU_height-1)％128)被标记为可用。

[0598] 4)对于具有块矢量(xBV,yBV)的IBC CU,如果任何位置(x,y)(其中,x＝(xCU+xBV)％256,...,(xCU+xBV+CU_width-1)％256并且y＝(yCU+yBV)％128,…,(yCU+yBV+CU_height-1)％128)被标记为不可用,则块矢量被认为是无效的。

[0599] 当CTU尺寸不为128×128或小于64×64或小于64×64时,与先前实施例(即,实施例#14)中相同的过程适用。

[0600] 5.15实施例#15

[0601] IBC参考可用性标记过程描述如下。在本文档中,改变以粗体、下划线和斜体本文进行指示。

[0602] 7.3.7.1通用条带数据语法

[0603]

slice_data( ) {	描述符
for(i = 0; i < NumBricksInCurrSlice; i++) {	
CtbAddrInBs = FirstCtbAddrBs[SliceBrickIdx[i]]	
for(j = 0; j < NumCtusInBrick[SliceBrickIdx[i]]; j++, CtbAddrInBs++) {	
if((j % BrickWidth[SliceBrickIdx[i]]) == 0) {	
NumHmvpCand = 0	
NumHmvpIbcCand = 0	
xPrevVPDU = 0	
yPrevVPDU = 0	
if(CtbSizeY == 128)	
reset_ibc_isDecoded(0, 0, 256, CtbSizeY, BufWidth, BufHeight)	
else	
reset_ibc_isDecoded(0, 0, 128*128/CtbSizeY, CtbSizeY, BufWidth, BufHeight)	
}	
CtbAddrInRs = CtbAddrBsToRs[CtbAddrInBs]	
……	

reset_ibc_isDecoded(x0, y0, w, h, BufWidth, BufHeight) {	***描述符***
if(x0 >= 0)	
for (x = x0 % BufWidth; x < x0 + w; x+=4)	
for (y = y0 % BufHeight; y < y0 + h; y+=4)	
isDecoded[x >> 2][y >> 2] = 0	
}	

[0604] ***BufWidth 等于(CtbSizeY==128)?256:(128*128/CtbSizeY)并且 BufHeight 等于 CtbSizeY。***

[0605] 7.3.7.5编解码单元语法

[0606]

coding_unit(x0, y0, cbWidth, cbHeight, treeType) {	描述符
if(treeType != DUAL_TREE_CHROMA && (CtbSizeY = = 128) && (x0 % 64) = = 0 && (y0 % 64) = = 0) {	
for(x = x0; x < x0 + cbWidth; x += 64)	
for(y = y0; y < y0 + cbHeight; y += 64)	
if((yPrevVPDU % 64) = = 0)	
reset_ibc_isDecoded(xPrevVPDU – 128, yPrevVPDU, 64, 64, BufWidth, BufHeight)	
else	
reset_ibc_isDecoded(xPrevVPDU – 64, yPrevVPDU, 64, 64, BufWidth, BufHeight)	
xPrevVPDU = x0	
yPrevVPDU = y0	
}	
if(treeType != DUAL_TREE_CHROMA && (CtbSizeY < 128) && (x0 % CtbSizeY) = = 0 && (y0 % CtbSizeY) = = 0) {	
reset_ibc_isDecoded(xPrevVPDU – (128*128/CtbSizeY – CtbSizeY), yPrevVPDU, 64, 64, BufWidth, BufHeight)	
xPrevVPDU = x0	
yPrevVPDU = y0	
}	
if(slice_type != I || sps_ibc_enabled_flag) {	
if(treeType != DUAL_TREE_CHROMA && !(cbWidth = = 4 && cbHeight = = 4 && !sps_ibc_enabled_flag))	
cu_skip_flag[x0][y0]	ae(v)
if(cu_skip_flag[x0][y0] = = 0 && slice_type != I && !(cbWidth = = 4 && cbHeight = = 4))	
pred_mode_flag	ae(v)
if(((slice_type = = I && cu_skip_flag[x0][y0] = =0) || (slice_type != I && (CuPredMode[x0][y0] != MODE_INTRA || (cbWidth = = 4 && cbHeight = = 4 && cu_skip_flag[x0][y0] = = 0)))) && sps_ibc_enabled_flag && (cbWidth != 128 && cbHeight != 128))	
pred_mode_ibc_flag	ae(v)

[0607]

}	
…	

[0608] 8.6.2 IBC块的运动矢量分量的推导过程

[0609] 8.6.2.1总体

[0610] …

[0611] ***比特流一致性的要求是，当用块矢量 mvL 调用条款 8.6.3.2 中的块矢量有效性检查过程时，isBVvalid 应当为真。***

[0612] …

[0613] 8.6.3 ibc块的解码过程

[0614] 8.6.3.1总体

[0615] 当对在ibc预测模式下编解码的编解码单元进行解码时,该过程被调用。

[0616] 该过程的输入是：

[0617] -亮度位置(xCb,yCb),指定当前编解码块相对于当前图片的左上角亮度样点的左上角样点，

[0618] -变量cbWidth,指定亮度样点中的当前编解码块的宽度，

[0619] -变量cbHeight,指定亮度样点中的当前编解码块的高度，

[0620] -变量numSbX和numSbY,指定水平和垂直方向上亮度编解码子块的数量，

[0621] -运动矢量mv[xSbIdx][ySbIdx],其中,xSbIdx＝0..numSbX-1,并且ySbIdx＝0..numSbY-1，

[0622] -变量cIdx,指定当前块的颜色分量索引。

[0623] - ***(nIbcBufW)×(ctbSize)阵列 ibcBuf***

[0624] …

[0625] 对于子块索引(xSbIdx,ySbIdx)(其中,xSbIdx＝0..numSbX-1,并且ySbIdx＝0..numSbY-1)处的每个编解码子块,以下适用：

[0626] -指定当前编解码子块相对于当前图片的左上角亮度样点的左上角样点的亮度位置(xSb,ySb)被推导如下：

[0627] (xSb,ySb)＝(xCb+xSbIdx*sbWidth,yCb+ySbIdx*sbHeight)(8-913)

[0628] ***如果 cIdx 等于 0，则 nIbcBufW 被设置为 ibcBufferWidth，否则 nIbcBufW 被设置为(ibcBufferWidth / SubWidthC)。以下适用:***

predSamples[xSb + x][ySb + y] = ibcBuf [(xSb + x + (mv[xSb][ySb][0] >> 4)) % nIbcRefW][ySb + y + (mv[xSb][ySb][1] >> 4)]

其中 x = 0..sbWidth – 1 并且 y = 0..sbHeight – 1。

…

8.6.3.2 块矢量有效性检查过程

该过程的输入是:

- *亮度位置（xCb，yCb ），指定当前编解码块相对于当前图片的左上角亮度样点的左上角样点，*
- *变量cbWidth，指定亮度样点中的当前编解码块的宽度，*
- *变量cbHeight，指定亮度样点中的当前编解码块的高度，*
- *变量numSbX 和numSbY，指定水平和垂直方向上亮度编解码子块的数量，*
- *块矢量 mv[xSbIdx][ySbIdx]，其中，xSbIdx = 0 .. numSbX − 1，并且 ySbIdx = 0 .. numSbY − 1，*
- *变量cIdx，指定当前块的颜色分量索引。*

[0629] - *(nIbcBufW)×(ctbSize)阵列 ibcBuf*

该过程的输出是用以指示块矢量是否有效的标志 isBVvalid。

以下适用

1. *isBVvalid 被设置为等于真。*
2. *如果((yCb & (ctbSize – 1)) + mv[0][0][1] + cbHeight) > ctbSize，则 isBVvalid 被设置为等于假。*
3. *否则，对于每个子块索引 xSbIdx 、ySbIdx （其中，xSbIdx = 0 .. numSbX – 1，并且 ySbIdx = 0 .. numSbY – 1），其相对于 ibcBuf 的左上角亮度样点的位置被推导为:*

   *xTL = (xCb + xSbIdx * sbWidth + mv[xSbIdx][ySbIdx][0]) & (nIbcBufW – 1)*

   *yTL = (yCb & (ctbSize – 1)) + ySbIdx * sbHeight + mv[xSbIdx][ySbIdx][1]*

   *xBR = (xCb + xSbIdx * sbWidth + sbWidth – 1 + mv[xSbIdx][ySbIdx][0]) & (nIbcBufW – 1)*

   *yBR = (yCb & (ctbSize – 1)) + ySbIdx * sbHeight + sbHeight – 1 + mv[xSbIdx][ySbIdx][1]*

[0630] *如果(isDecoded[xTL>>2][yTL>>2] == 0)或(isDecoded[xBR>>2][yTL>>2] == 0)或(isDecoded[xBR>>2][yBR>>2] == 0)，则 isBVvalid 被设置为等于假。*

[0631] 8.7.5图片重构过程

[0632] 8.7.5.1总体

[0633] 该过程的输入是：

[0634] -位置(xCurr,yCurr)，指定当前块相对于当前图片分量的左上角样点的左上角样点，

[0635] -变量nCurrSw和nCurrSh，分别指定当前块的宽度和高度，

[0636] -变量cIdx，指定当前块的颜色分量，

[0637] -(nCurrSw)x(nCurrSh)阵列predSamples，指定当前块的预测样点，

[0638] -(nCurrSw)x(nCurrSh)阵列resSamples，指定当前块的残差样点。

[0639] 该过程的输出是

[0640] -重构图片样点阵列recSamples。

- ***IBC 参考阵列 ibcBuf。***

…

[0641] ***用 nIbcBufW 表示 ibcBuf 的宽度，以下适用：***

ibcBuf[(xCurr + i) & (nIbcBufW – 1)][(yCurr + j) & (ctbSize – 1)] = recSamples[xCurr + i][yCurr + j]

其中 i = 0..nCurrSw – 1， j = 0..nCurrSh – 1。

[0642] 5.16实施例#16

[0643] 除了以下改变以外，与先前的实施例相同。

[0644]

slice_data( ) {	描述符
for(i = 0; i < NumBricksInCurrSlice; i++) {	
CtbAddrInBs = FirstCtbAddrBs[SliceBrickIdx[i]]	
for(j = 0; j < NumCtusInBrick[SliceBrickIdx[i]]; j++, CtbAddrInBs++) {	
if((j % BrickWidth[SliceBrickIdx[i]]) == 0) {	
NumHmvpCand = 0	
NumHmvpIbcCand = 0	
xPrevVPDU = 0	
yPrevVPDU = 0	
if(CtbSizeY == 128)	

reset_ibc_isDecoded(0, 0, 192, CtbSizeY, BufWidth, BufHeight)	
else	
reset_ibc_isDecoded(0, 0, 128*128/CtbSizeY, CtbSizeY, BufWidth, BufHeight)	
}	
CtbAddrInRs = CtbAddrBsToRs[CtbAddrInBs]	
……	

[0645]

reset_ibc_isDecoded(x0, y0, w, h, BufWidth, BufHeight) {	***描述符***
if(x0 >= 0)	
for (x = x0 % BufWidth; x < x0 + w; x+=4)	
for (y = y0 % BufHeight; y < y0 + h; y+=4)	
isDecoded[x >> 2][y >> 2] = 0	
}	

[0646] ***BufWidth 等于(CtbSizeY==128)?192:(128*128/CtbSizeY)并且 BufHeight 等于 CtbSizeY。***

[0647] 5.17实施例#17

[0648] 在本文档中，一些示例中的改变以粗体、下划线文本进行指示。

[0649] 7.3.7条带数据语法

[0650] 7.3.7.1通用条带数据语法

[0651]

slice_data( ) {	描述符
for(i = 0; i < NumBricksInCurrSlice; i++) {	
CtbAddrInBs = FirstCtbAddrBs[SliceBrickIdx[i]]	
for(j = 0; j < NumCtusInBrick[SliceBrickIdx[i]]; j++, CtbAddrInBs++) {	
if((j % BrickWidth[SliceBrickIdx[i]]) == 0) {	
NumHmvpCand = 0	
NumHmvpIbcCand = 0	
resetIbcBuf = 1	
}	
CtbAddrInRs = CtbAddrBsToRs[CtbAddrInBs]	
coding_tree_unit( )	
if(entropy_coding_sync_enabled_flag && ((j + 1) % BrickWidth[SliceBrickIdx[i]] == 0)) {	
end_of_subset_one_bit /* equal to 1 */	ae(v)
if(j < NumCtusInBrick[SliceBrickIdx[i]] − 1)	

[0652]

byte_alignment( )	
}	
}	
if(!entropy_coding_sync_enabled_flag) {	
end_of_brick_one_bit /* equal to 1 */	ae(v)
if(i < NumBricksInCurrSlice − 1)	
byte_alignment( )	
}	
}	
}	

[0653] 7.4.8.5编解码单元语义

[0654] 当所有以下条件都为真时，通过将NumHmvpSmrIbcCand设置为等于NumHmvpIbcCand，并且将HmvpSmrIbcCandList[i]设置为等于HmvpIbcCandList[i](对于i＝0..NumHmvpIbcCand-1)来更新共享Merge候选列表区域的基于历史的运动矢量预测值列表：

[0655] -IsInSmr[x0][y0]等于TRUE(真)。

[0656] -SmrX[x0][y0]等于x0。

[0657] -SmrY[x0][y0]等于y0。

[0658] 对于x＝x0..x0+cbWidth-1和y＝y0..y0+cbHeight-1,进行以下指定：

[0659] CbPosX[x][y]＝x0 (7-135)

[0660] CbPosY[x][y]＝y0 (7-136)

[0661] CbWidth[x][y]＝cbWidth (7-137)

[0662] CbHeight[x][y]＝cbHeight (7-138)

[0663] ***将vSize设置为min(ctbSize,64)，并且将wIbcBuf设置为(128*128/ctbSize)。***

ibcBuf的宽度和高度相应地为wIbcBuf和ctbSize。

如果refreshIbcBuf等于1，则以下适用：

- ***对于x＝x0..x0＋wIbcBuf－1和y＝y0..y0＋ctbSize－1，ibcBuf[x % wIbcBuf][y % ctbSize]＝－1***
- ***resetIbcBuf＝0***

当对于x＝x0..x0＋vSize－1和y＝y0..y0＋vSize－1，（x0 % vSize）等于0并且（y0 % vSize）等于0时，以下适用：

ibcBuf[x % wIbcBuf][y % ctbSize]＝－1

[0664] 8.6.2 IBC块的运动矢量分量的推导过程

[0665] 8.6.2.1总体

[0666] 该过程的输入是：

[0667] -当前亮度编解码块相对于当前图片的左上角亮度样点的左上角样点的亮度位置(xCb,yCb)，

[0668] -变量cbWidth,指定亮度样点中的当前编解码块的宽度，

[0669] -变量cbHeight,指定亮度样点中的当前编解码块的高度。

[0670] 该过程的输出是:

[0671] -1/16分数样点精度的亮度运动矢量mvL。

[0672] 亮度运动矢量mvL被推导如下:

[0673] -以亮度位置(xCb,yCb)、变量cbWidth和cbHeight作为输入来调用如在条款8.6.2.2中指定的IBC亮度运动矢量预测的推导过程,并且输出是亮度运动矢量mvL。

[0674] -当general_merge_flag[xCb][yCb]等于0时,以下适用:

[0675] 1.变量mvd被推导如下:

[0676] mvd[0]＝MvdL0[xCb][yCb][0] (8-883)

[0677] mvd[1]＝MvdL0[xCb][yCb][1] (8-884)

[0678] 2.以设置为等于mvL的mvX、设置为等于MvShift+2的rightShift和设置为等于MvShift+2的leftShift作为输入并且以取整后的mvL作为输出来调用如在条款8.5.2.14中指定的运动矢量的取整过程。

[0679] 3.亮度运动矢量mvL被修改如下:

[0680] $u[0]＝(mvL[0]+mvd[0]+2^{18})\%2^{18}$ (8-885)

[0681] $mvL[0]＝(u[0]>=2^{17})?(u[0]-2^{18}):u[0]$ (8-886)

[0682] $u[1]＝(mvL[1]+mvd[1]+2^{18})\%2^{18}$ (8-887)

[0683] $mvL[1]＝(u[1]>=2^{17})?(u[1]-2^{18}):u[1]$ (8-888)

[0684] 注1-如上指定的mvL[0]和mvL[1]的结果值总是在范围-2^{17}至$2^{17}-1$(包括-2^{17}和$2^{17}-1$)中。

[0685] 用亮度运动矢量mvL调用如在条款8.6.2.6中指定的基于历史的运动矢量预测值列表的更新过程。

[0686] ***比特流一致性的要求是,亮度块矢量mvL应当遵守以下约束:***

[0687]
- ***((yCb+(mvL[1]>>4))%wIbcBuf)+cbHeight小于或等于ctbSize***
- ***对于x=xCb..xCb+cbWidth-1和y=yCb..yCb+cbHeight-1,ibcBuf[(x+(mvL[0]>>4))%wIbcBuf][(y+(mvL[1]>>4))%ctbSize]不应等于-1。***

[0688] 8.7.5图片重构过程

[0689] 8.7.5.1总体

[0690] 该过程的输入是:

[0691] -位置(xCurr,yCurr),指定当前块相对于当前图片分量的左上角样点的左上角样点,

[0692] -变量nCurrSw和nCurrSh,分别指定当前块的宽度和高度,

[0693] -变量cIdx,指定当前块的颜色分量,

[0694] -(nCurrSw)x(nCurrSh)阵列predSamples,指定当前块的预测样点,

[0695] -(nCurrSw)x(nCurrSh)阵列resSamples,指定当前块的残差样点。

[0696] 该过程的输出是重构图片样点阵列resSamples和IBC缓冲区阵列ibcBuf。

[0697] 取决于颜色分量cIdx的值,进行以下指定:

[0698] -如果cIdx等于0,则recSamples对应于重构图片样点阵列S_L,并且函数clipCidx1对应于$Clip1_Y$。

[0699] -否则,如果cIdx等于1,则tuCbfChroma被设置为等于tu_cbf_cb[xCurr][yCurr],recSamples对应于重构色度样点阵列S_{Cb},并且函数clipCidx1对应于$Clip1_C$。

[0700] -否则(cIdx等于2),tuCbfChroma被设置为等于tu_cbf_cr[xCurr][yCurr],recSamples对应于重构色度样点阵列S_{Cr},并且函数clipCidx1对应于$Clip1_C$。

[0701] 取决于slice_lmcs_enabled_flag的值,以下适用:

[0702] -如果slice_lmcs_enabled_flag等于0,则对于i=0..nCurrSw-1,j=0..nCurrSh-1,位置(xCurr,yCurr)处的重构样点recSamples的(nCurrSw)x(nCurrSh)块被推导如下:

[0703] recSamples[xCurr+i][yCurr+j]=clipCidx1(predSamples[i][j]+resSamples[i][j]) (8-992)

[0704] -否则(slice_lmcs_enabled_flag等于1),以下适用:

[0705] -如果cIdx等于0,则以下适用:

[0706] -以亮度位置(xCurr,yCurr)、块宽度nCurrSw和高度nCurrSh、预测亮度样点阵列predSamples以及残差亮度样点阵列resSamples作为输入来调用如在条款8.7.5.2中指定的亮度样点的图片重构与映射过程,并且输出是重构亮度样点阵列recSamples。

[0707] -否则(cIdx大于0),以色度位置(xCurr,yCurr)、变换块宽度nCurrSw和高度nCurrSh、当前色度变换块的编解码块标志tuCbfChroma、预测色度样点阵列predSamples以及残差色度样点阵列resSamples作为输入来调用如在条款8.7.5.3中指定的色度样点的图片重构与依赖于亮度的色度残差缩放过程,并且输出是重构色度样点阵列recSamples。

[0708] ***在对当前编解码单元进行解码之后,以下适用:***

对于i=0..nCurrSw-1,j=0..nCurrSh-1

ibcBuf[(xCurr+i)%wIbcBuf][(yCurr+j)%ctbSize]=recSamples[xCurr+i][yCurr+j]。

[0709] 5.18实施例#18

[0710] 在本文档中,一些示例中的改变以粗体、下划线和斜体文本进行指示。

[0711] 7.3.7条带数据语法

[0712] 7.3.7.1通用条带数据语法

[0713]

slice_data( ) {	描述符
for(i = 0; i < NumBricksInCurrSlice; i++) {	
CtbAddrInBs = FirstCtbAddrBs[SliceBrickIdx[i]]	
for(j = 0; j < NumCtusInBrick[SliceBrickIdx[i]]; j++, CtbAddrInBs++) {	
if((j % BrickWidth[SliceBrickIdx[i]]) == 0) {	
NumHmvpCand = 0	
NumHmvpIbcCand = 0	
resetIbcBuf = 1	
}	
CtbAddrInRs = CtbAddrBsToRs[CtbAddrInBs]	
coding_tree_unit( )	
if(entropy_coding_sync_enabled_flag && ((j + 1) % BrickWidth[SliceBrickIdx[i]] == 0)) {	
end_of_subset_one_bit /* equal to 1 */	ae(v)
if(j < NumCtusInBrick[SliceBrickIdx[i]] − 1)	
byte_alignment( )	
}	
}	
if(!entropy_coding_sync_enabled_flag) {	
end_of_brick_one_bit /* equal to 1 */	ae(v)
if(i < NumBricksInCurrSlice − 1)	
byte_alignment( )	
}	
}	
}	

[0714] 7.4.8.5编解码单元语义

[0715] 当所有以下条件都为真时，通过将NumHmvpSmrIbcCand设置为等于NumHmvpIbcCand，并且将HmvpSmrIbcCandList[i]设置为等于HmvpIbcCandList[i](对于i＝0..NumHmvpIbcCand-1)来更新共享Merge候选列表区域的基于历史的运动矢量预测值列表：

[0716] -IsInSmr[x0][y0]等于TRUE(真)。

[0717] -SmrX[x0][y0]等于x0。

[0718] -SmrY[x0][y0]等于y0。

[0719] 对于x＝x0..x0+cbWidth-1和y＝y0..y0+cbHeight-1，进行以下指定：

[0720] CbPosX[x][y]＝x0 (7-135)

[0721] CbPosY[x][y]＝y0 (7-136)

[0722] CbWidth[x][y]＝cbWidth (7-137)

[0723] CbHeight[x][y]＝cbHeight (7-138)

*将vSize设置为min(ctbSize,64),并且将wIbcBufY设置为(128*128/CtbSizeY)。*

$ibcBuf_L$是宽度为wIbcBufY并且高度为CtbSizeY的阵列。

$ibcBuf_{Cb}$和$ibcBuf_{Cr}$是宽度为wIbcBufC =(wIbcBufY/SubWidthC)并且高度为(CtbSizeY/SubHeightC)（即，CtbSizeC）的阵列。

如果resetIbcBuf等于1，则以下适用

- *对于x=x0..x0+wIbcBufY−1和y=y0..y0+CtbSizeY−1，$ibcBuf_L$[x % wIbcBufY][y % CtbSizeY]=−1*
- *对于x=x0..x0+wIbcBufC−1和y=y0..y0+CtbSizeC−1,$ibcBuf_{Cb}$[x % wIbcBufC][y % CtbSizeC]=−1*
- *对于x=x0..x0+wIbcBufC−1和y=y0..y0+CtbSizeC−1,$ibcBuf_{Cr}$[x % wIbcBufC][y % CtbSizeC]=−1*

[0724]

- *resetIbcBuf=0*

当（x0 % vSizeY）等于0并且（y0 % vSizeY）等于0时，以下适用

- *对于 x=x0..x0+vSize −1 和 y=y0..y0+vSize −1，$ibcBuf_L$[x % wIbcBufY][y % CtbSizeY]=−1*
- *对于 x=x0/SubWidthC..x0/ SubWidthC+vSize/ SubWidthC −1 和 y=y0/SubHeightC..y0/SubHeightC+vSize/SubHeightC −1，$ibcBuf_{Cb}$[x % wIbcBufC][y % CtbSizeC]=−1*
- *对于 x=x0/SubWidthC..x0/ SubWidthC+vSize/ SubWidthC −1 和 y=y0/SubHeightC..y0/SubHeightC+vSize/SubHeightC −1，$ibcBuf_{Cr}$[x % wIbcBufC][y % CtbSizeC]=−1*

[0725] 8.6.2 IBC块的运动矢量分量的推导过程

[0726] 8.6.2.1总体

[0727] 该过程的输入是：

[0728] -当前亮度编解码块相对于当前图片的左上角亮度样点的左上角样点的亮度位置(xCb,yCb)，

[0729] -变量cbWidth,指定亮度样点中的当前编解码块的宽度，

[0730] -变量cbHeight,指定亮度样点中的当前编解码块的高度。

[0731] 该过程的输出是：

[0732] -1/16分数样点精度的亮度运动矢量mvL。

[0733] 亮度运动矢量mvL被推导如下：

[0734] -以亮度位置(xCb,yCb)、变量cbWidth和cbHeight作为输入来调用如在条款8.6.2.2中指定的IBC亮度运动矢量预测的推导过程,并且输出是亮度运动矢量mvL。

[0735] -当general_merge_flag[xCb][yCb]等于0时,以下适用：

[0736] 4.变量mvd被推导如下：

[0737] mvd[0]＝MvdL0[xCb][yCb][0] (8-883)

[0738] mvd[1]＝MvdL0[xCb][yCb][1] (8-884)

[0739] 5.以设置为等于mvL的mvX、设置为等于MvShift+2的rightShift和设置为等于MvShift+2的leftShift作为输入并且以取整后的mvL作为输出来调用如在条款8.5.2.14中指定的运动矢量的取整过程。

[0740] 6.亮度运动矢量mvL被修改如下：

[0741] $u[0]＝(mvL[0]+mvd[0]+2^{18})\%2^{18}$ (8-885)

[0742] $mvL[0]＝(u[0]>=2^{17})?(u[0]-2^{18}):u[0]$ (8-886)

[0743] $u[1]＝(mvL[1]+mvd[1]+2^{18})\%2^{18}$ (8-887)

[0744] $mvL[1]＝(u[1]>=2^{17})?(u[1]-2^{18}):u[1]$ (8-888)

[0745] 注1-如上指定的mvL[0]和mvL[1]的结果值总是在范围-2^{17}至$2^{17}-1$(包括-2^{17}和$2^{17}-1$)中。

[0746] 用亮度运动矢量mvL调用如在条款8.6.2.6中指定的基于历史的运动矢量预测值列表的更新过程。

以 mvL 作为输入并且以 mvC 作为输出来调用条款 8.6.2.5。

比特流一致性的要求是，亮度块矢量 mvL 应当遵守以下约束：

- *((yCb + (mvL[1] >> 4)) % CtbSizeY) + cbHeight 小于或等于 CtbSizeY*

[0747] - *对于 $x = xCb..xCb + cbWidth - 1$ 和 $y = yCb..yCb + cbHeight - 1$，$ibcBuf_L[(x + (mvL[0]>>4)) \% wIbcBufY][(y +(mvL[1]>>4)) \% CtbSizeY]$ 不应等于−1。*

- *如果 treeType 等于 SINGLE_TREE，则对于 $x = xCb..xCb + cbWidth - 1$ 和*

[0748] *$y = yCb..yCb + cbHeight - 1$，$ibcBuf_{Cb}[(x + (mvC[0]>>5)) \% wIbcBufC][(y +(mvC[1]>>5)) \% CtbSizeC]$ 不应等于−1。*

[0749] 8.6.3 ibc块的解码过程

[0750] 8.6.3.1总体

[0751] 当对在ibc预测模式下编解码的编解码单元进行解码时，该过程被调用。

[0752] 该过程的输入是：

[0753] -亮度位置(xCb,yCb)，指定当前编解码块相对于当前图片的左上角亮度样点的左上角样点，

[0754] -变量cbWidth，指定亮度样点中的当前编解码块的宽度，

[0755] -变量cbHeight，指定亮度样点中的当前编解码块的高度，

[0756] -当前块的颜色分量索引。

- *运动矢量 mv，*

[0757] - *(wIbcBufY)×(CtbSizeY) 阵列 $ibcBuf_L$，(wIbcBufC)×(CtbSizeC) 阵列 $ibcBuf_{Cb}$，(wIbcBufC)×(CtbSizeC) 阵列 $ibcBuf_{Cr}$。*

[0758] 该过程的输出是：

[0759] -预测样点的阵列predSamples。

对于 x = xCb.. xCb+ Width – 1 和 y = yCb..yCb + Height – 1，以下适用

如果 cIdx 等于 0

predSamples[x][y] = ibcBuf$_L$[(x + mv[0] >> 4)) % wIbcBufY][(y + (mv[1] >> 4)) % CtbSizeY]

如果 cIdx 等于 1

[0760]

predSamples[x][y] = ibcBuf$_{Cb}$[(x + mv[0] >> 5)) % wIbcBufC][(y + (mv[1] >> 5)) % CtbSizeC]

如果 cIdx 等于 2

predSamples[x][y] = ibcBuf$_{Cr}$[(x + mv[0] >> 5)) % wIbcBufC][(y + (mv[1] >> 5)) % CtbSizeC]

[0761] 8.7.5图片重构过程

[0762] 8.7.5.1总体

[0763] 该过程的输入是：

[0764] -位置(xCurr,yCurr)，指定当前块相对于当前图片分量的左上角样点的左上角样点，

[0765] -变量nCurrSw和nCurrSh，分别指定当前块的宽度和高度，

[0766] -变量cIdx，指定当前块的颜色分量，

[0767] -(nCurrSw)x(nCurrSh)阵列predSamples，指定当前块的预测样点，

[0768] -(nCurrSw)x(nCurrSh)阵列resSamples，指定当前块的残差样点。该过程的输出***是***重构图片样点阵列recSamples***和IBC缓冲区阵列ibcBuf$_L$、ibcBuf$_{Cb}$、 ibcBuf$_{Cr}$。***

[0769] 取决于颜色分量cIdx的值，进行以下指定：

[0770] -如果cIdx等于0，则recSamples对应于重构图片样点阵列S_L，并且函数clipCidx1对应于$Clip1_Y$。

[0771] -否则，如果cIdx等于1，则tuCbfChroma被设置为等于tu_cbf_cb[xCurr][yCurr]，recSamples对应于重构色度样点阵列S_{Cb}，并且函数clipCidx1对应于$Clip1_C$。

[0772] -否则(cIdx等于2)，tuCbfChroma被设置为等于tu_cbf_cr[xCurr][yCurr]，recSamples对应于重构色度样点阵列S_{Cr}，并且函数clipCidx1对应于$Clip1_C$。

[0773] 取决于slice_lmcs_enabled_flag的值，以下适用：

[0774] -如果slice_lmcs_enabled_flag等于0，则对于i＝0..nCurrSw-1，

[0775] j＝0..nCurrSh-1，位置(xCurr,yCurr)处的重构样点recSamples的(nCurrSw)x(nCurrSh)块被推导如下：

[0776] recSamples[xCurr+i][yCurr+j]＝clipCidx1(predSamples[i][j]+resSamples[i][j]) (8-992)

[0777] -否则(slice_lmcs_enabled_flag等于1)，以下适用：

[0778] -如果cIdx等于0，则以下适用：

[0779] -以亮度位置(xCurr,yCurr)、块宽度nCurrSw和高度nCurrSh、预测亮度样点阵列

predSamples以及残差亮度样点阵列resSamples作为输入来调用如在条款8.7.5.2中指定的亮度样点的图片重构与映射过程,并且输出是重构亮度样点阵列recSamples。

[0780] -否则(cIdx大于0),以色度位置(xCurr,yCurr)、变换块宽度nCurrSw和高度nCurrSh、当前色度变换块的编解码块标志tuCbfChroma、预测色度样点阵列predSamples以及残差色度样点阵列resSamples作为输入来调用如在条款8.7.5.3中指定的色度样点的图片重构与依赖于亮度的色度残差缩放过程,并且输出是重构色度样点阵列recSamples。

[0781] ***在对当前编解码单元进行解码之后，以下可以适用:***

如果 cIdx 等于 0，并且如果 treeType 等于 SINGLE_TREE 或 DUAL_TREE_LUMA，则以下适用

对于 i = 0..nCurrSw – 1，j = 0..nCurrSh – 1

$$ibcBuf_L[\ (\ xCurr + i\ )\ \%\ wIbcBufY\ ][\ (\ yCurr + j\ )\ \%\ CtbSizeY\ ] = recSamples[\ xCurr + i\ ][\ yCurr + j\ ]。$$

如果 cIdx 等于 1，并且如果 treeType 等于 SINGLE_TREE 或 DUAL_TREE_CHROMA，则以下适用

对于 i = 0..nCurrSw – 1，j = 0..nCurrSh – 1

$$ibcBuf_{Cb}[\ (\ xCurr + i\ )\ \%\ wIbcBufC\ ][\ (\ yCurr + j\ )\ \%\ CtbSizeC\ ] = recSamples[\ xCurr + i\ ][\ yCurr + j\ ]。$$

如果 cIdx 等于 2，并且如果 treeType 等于 SINGLE_TREE 或 DUAL_TREE_CHROMA，则以下适用

对于 i = 0..nCurrSw – 1，j = 0..nCurrSh – 1

$$ibcBuf_{Cr}[\ (\ xCurr + i\ )\ \%\ wIbcBufC\ ][\ (\ yCurr + j\ )\ \%\ CtbSizeC\ ] = recSamples[\ xCurr + i\ ][\ yCurr + j\ ]。$$

[0782] 5.19实施例#19

[0783] 在本文档中,一些示例中的改变以粗体、下划线文本进行指示。

[0784] 7.3.7条带数据语法

[0785] 7.3.7.1通用条带数据语法

[0786]

slice_data( ) {	描述符
for(i = 0; i < NumBricksInCurrSlice; i++) {	
CtbAddrInBs = FirstCtbAddrBs[SliceBrickIdx[i]]	
for(j = 0; j < NumCtusInBrick[SliceBrickIdx[i]]; j++, CtbAddrInBs++) {	
if((j % BrickWidth[SliceBrickIdx[i]]) == 0) {	
NumHmvpCand = 0	
NumHmvpIbcCand = 0	
resetIbcBuf = 1	

[0787]

}	
CtbAddrInRs = CtbAddrBsToRs[CtbAddrInBs]	
coding_tree_unit( )	
if(entropy_coding_sync_enabled_flag && ((j + 1) % BrickWidth[SliceBrickIdx[i]] == 0)) {	
end_of_subset_one_bit /* equal to 1 */	ae(v)
if(j < NumCtusInBrick[SliceBrickIdx[i]] − 1)	
byte_alignment( )	
}	
}	
if(!entropy_coding_sync_enabled_flag) {	
end_of_brick_one_bit /* equal to 1 */	ae(v)
if(i < NumBricksInCurrSlice − 1)	
byte_alignment( )	
}	
}	
}	

[0788] 7.4.8.5编解码单元语义

[0789] 当所有以下条件都为真时，通过将NumHmvpSmrIbcCand设置为等于NumHmvpIbcCand，并且将HmvpSmrIbcCandList[i]设置为等于HmvpIbcCandList[i](对于i＝0..NumHmvpIbcCand-1)来更新共享Merge候选列表区域的基于历史的运动矢量预测值列表：

[0790] -IsInSmr[x0][y0]等于TRUE(真)。

[0791] -SmrX[x0][y0]等于x0。

[0792] -SmrY[x0][y0]等于y0。

[0793] 对于x＝x0..x0+cbWidth-1和y＝y0..y0+cbHeight-1，进行以下指定：

[0794] CbPosX[x][y]＝x0 (7-135)

[0795] CbPosY[x][y]＝y0 (7-136)

[0796] CbWidth[x][y]＝cbWidth (7-137)

[0797] CbHeight[x][y]＝cbHeight (7-138)

[0798] ***将vSize设置为min(ctbSize,64)，并且将wIbcBufY设置为(128*128/CtbSizeY)。***

$ibcBuf_L$是宽度为wIbcBufY并且高度为CtbSizeY的阵列。

$ibcBuf_{Cb}$和$ibcBuf_{Cr}$是宽度为wIbcBufC =(wIbcBufY/SubWidthC)并且高度为(CtbSizeY/SubHeightC)（即，CtbSizeC）的阵列。

如果 resetIbcBuf 等于 1，则以下适用

- *对于 x = x0..x0 + wIbcBufY − 1 和 y = y0..y0 + CtbSizeY − 1，ibcBuf$_L$[x % wIbcBufY][y % CtbSizeY] = − 1*
- *对于 x = x0..x0 + wIbcBufC − 1 和 y = y0..y0 + CtbSizeC − 1，ibcBuf$_{Cb}$[x % wIbcBufC][y % CtbSizeC] = − 1*
- *对于 x = x0..x0 + wIbcBufC − 1 和 y = y0..y0 + CtbSizeC − 1，ibcBuf$_{Cr}$[x % wIbcBufC][y % CtbSizeC] = − 1*
- *resetIbcBuf = 0*

当（x0 % vSizeY）等于 0 并且（y0 % vSizeY）等于 0 时，以下适用

[0799]
- *对于 x = x0..x0 + min(vSize, cbWidth) − 1 和 y = y0..y0 + min(vSize, cbHeight)− 1，ibcBuf$_L$[x % wIbcBufY][y % CtbSizeY] = − 1*
- *对于 x = x0/SubWidthC..x0/ SubWidthC + min(vSize/ SubWidthC, cbWidth) − 1 和 y = y0/SubHeightC..y0/SubHeightC + min(vSize/SubHeightC, cbHeight) − 1，ibcBuf$_{Cb}$[x % wIbcBufC][y % CtbSizeC] = − 1*
- *对于 x = x0/SubWidthC..x0/ SubWidthC + min(vSize/ SubWidthC, cbWidth) − 1 和 y = y0/SubHeightC..y0/SubHeightC + min(vSize/SubHeightC, cbHeight) − 1，ibcBuf$_{Cr}$[x % wIbcBufC][y % CtbSizeC] = − 1*

[0800] 8.6.2 IBC块的运动矢量分量的推导过程

[0801] 8.6.2.1总体

[0802] 该过程的输入是：

[0803] -当前亮度编解码块相对于当前图片的左上角亮度样点的左上角样点的亮度位置(xCb,yCb)，

[0804] -变量cbWidth,指定亮度样点中的当前编解码块的宽度，

[0805] -变量cbHeight,指定亮度样点中的当前编解码块的高度。

[0806] 该过程的输出是：

[0807] -1/16分数样点精度的亮度运动矢量mvL。

[0808] 亮度运动矢量mvL被推导如下：

[0809] -以亮度位置(xCb,yCb)、变量cbWidth和cbHeight作为输入来调用如在条款8.6.2.2中指定的IBC亮度运动矢量预测的推导过程,并且输出是亮度运动矢量mvL。

[0810] -当general_merge_flag[xCb][yCb]等于0时,以下适用：

[0811] 7.变量mvd被推导如下：

[0812] mvd[0]＝MvdL0[xCb][yCb][0] (8-883)

[0813] mvd[1]＝MvdL0[xCb][yCb][1] (8-884)

[0814] 8.以设置为等于mvL的mvX、设置为等于MvShift+2的rightShift和设置为等于

MvShift+2的leftShift作为输入并且以取整后的mvL作为输出来调用如在条款8.5.2.14中指定的运动矢量的取整过程。

[0815] 9.亮度运动矢量mvL被修改如下：

[0816] $u[0]=(mvL[0]+mvd[0]+2^{18})\%2^{18}$ (8-885)

[0817] $mvL[0]=(u[0]>=2^{17})?(u[0]-2^{18}):u[0]$ (8-886)

[0818] $u[1]=(mvL[1]+mvd[1]+2^{18})\%2^{18}$ (8-887)

[0819] $mvL[1]=(u[1]>=2^{17})?(u[1]-2^{18}):u[1]$ (8-888)

[0820] 注1-如上指定的mvL[0]和mvL[1]的结果值总是在范围-2^{17}至$2^{17}-1$(包括-2^{17}和$2^{17}-1$)中。

[0821] 用亮度运动矢量mvL调用如在条款8.6.2.6中指定的基于历史的运动矢量预测值列表的更新过程。

[0822] ***以 mvL 作为输入并且以 mvC 作为输出来调用条款 8.6.2.5。***

比特流一致性的要求是，亮度块矢量 mvL 应当遵守以下约束：

- ***((yCb + (mvL[1] >> 4)) % CtbSizeY) + cbHeight 小于或等于 CtbSizeY***
- ***对于 x = xCb..xCb + cbWidth – 1 和 y = yCb..yCb + cbHeight – 1，$ibcBuf_L$[(x + (mvL[0]>>4)) % wIbcBufY][(y +(mvL[1]>>4)) % CtbSizeY]不应等于–1。***
- ***如果 treeType 等于 SINGLE_TREE，则 对于 x = xCb..xCb + cbWidth – 1 和 y = yCb..yCb + cbHeight – 1，$ibcBuf_{Cb}$[(x + (mvC[0]>>5)) % wIbcBufC][(y +(mvC[1]>>5)) % CtbSizeC]不应等于–1。***

[0823] 8.6.3 ibc块的解码过程

[0824] 8.6.3.1总体

[0825] 当对在ibc预测模式下编解码的编解码单元进行解码时,该过程被调用。

[0826] 该过程的输入是：

[0827] -亮度位置(xCb,yCb),指定当前编解码块相对于当前图片的左上角亮度样点的左上角样点，

[0828] -变量cbWidth,指定亮度样点中的当前编解码块的宽度，

[0829] -变量cbHeight,指定亮度样点中的当前编解码块的高度，

- ***变量cIdx，指定当前块的颜色分量索引。***
- ***运动矢量mv，***
- ***(wIbcBufY)×(CtbSizeY)阵列 $ibcBuf_L$，(wIbcBufC)×(CtbSizeC)阵列 $ibcBuf_{Cb}$，(wIbcBufC)×(CtbSizeC)阵列 $ibcBuf_{Cr}$。***

该过程的输出是：

- ***预测样点的阵列predSamples。***

对于x = xCb.. xCb+ Width – 1 和y = yCb..yCb + Height – 1，以下适用

如果cIdx 等于0

[0830] ***predSamples[x][y] = $ibcBuf_L$[(x + mv[0] >> 4)) % wIbcBufY][(y + (mv[1] >> 4)) % CtbSizeY]***

如果cIdx 等于1

predSamples[x][y] = $ibcBuf_{Cb}$[(x + mv[0] >> 5)) % wIbcBufC][(y + (mv[1] >> 5)) % CtbSizeC]

如果cIdx 等于2

predSamples[x][y] = $ibcBuf_{Cr}$[(x + mv[0] >> 5)) % wIbcBufC][(y + (mv[1] >> 5)) % CtbSizeC]

[0831] 8.7.5图片重构过程

[0832] 8.7.5.1总体

[0833] 该过程的输入是：

[0834] -位置(xCurr,yCurr)，指定当前块相对于当前图片分量的左上角样点的左上角样点，

[0835] -变量nCurrSw和nCurrSh，分别指定当前块的宽度和高度，

[0836] -变量cIdx，指定当前块的颜色分量，

[0837] -(nCurrSw)x(nCurrSh)阵列predSamples，指定当前块的预测样点，

[0838] -(nCurrSw)x(nCurrSh)阵列resSamples，指定当前块的残差样点。

[0839] 该过程的输出是重构图片样点阵列recSamples和IBC缓冲区阵列$ibcBuf_L$、$ibcBuf_{Cb}$、$ibcBuf_{Cr}$。

[0840] 取决于颜色分量cIdx的值，进行以下指定：

[0841] -如果cIdx等于0，则recSamples对应于重构图片样点阵列S_L，并且函数clipCidx1对应于$Clip1_Y$。

[0842] -否则，如果cIdx等于1，则tuCbfChroma被设置为等于tu_cbf_cb[xCurr][yCurr]，recSamples对应于重构色度样点阵列S_{Cb}，并且函数clipCidx1对应于$Clip1_C$。

[0843] -否则(cIdx等于2)，tuCbfChroma被设置为等于tu_cbf_cr[xCurr][yCurr]，recSamples对应于重构色度样点阵列S_{Cr}，并且函数clipCidx1对应于$Clip1_C$。

[0844] 取决于slice_lmcs_enabled_flag的值，以下适用：

[0845] -如果slice_lmcs_enabled_flag等于0，则对于i＝0..nCurrSw-1，j＝

0..nCurrSh-1,位置(xCurr,yCurr)处的重构样点recSamples的(nCurrSw)x(nCurrSh)块被推导如下:

[0846] recSamples[xCurr+i][yCurr+j]＝clipCidx1(predSamples[i][j]+resSamples[i][j]) (8-992)

[0847] -否则(slice_lmcs_enabled_flag等于1),以下适用:

[0848] -如果cIdx等于0,则以下适用:

[0849] -以亮度位置(xCurr,yCurr)、块宽度nCurrSw和高度nCurrSh、预测亮度样点阵列predSamples以及残差亮度样点阵列resSamples作为输入来调用如在条款8.7.5.2中指定的亮度样点的图片重构与映射过程,并且输出是重构亮度样点阵列recSamples。

[0850] -否则(cIdx大于0),以色度位置(xCurr,yCurr)、变换块宽度nCurrSw和高度nCurrSh、当前色度变换块的编解码块标志tuCbfChroma、预测色度样点阵列predSamples以及残差色度样点阵列resSamples作为输入来调用如在条款8.7.5.3中指定的色度样点的图片重构与依赖于亮度的色度残差缩放过程,并且输出是重构色度样点阵列recSamples。

[0851] *在对当前编解码单元进行解码之后，以下可以适用:*

如果 cIdx 等于 0，并且如果 treeType 等于 SINGLE_TREE 或 DUAL_TREE_LUMA，则以下适用

对于 i = 0..nCurrSw – 1，j = 0..nCurrSh – 1

$ibcBuf_L[\ (\ xCurr\ +\ i\ )\ \%\ wIbcBufY\ ][(\ yCurr + j\ )\ \%\ CtbSizeY\ ] =$
$recSamples[\ xCurr + i\ ][\ yCurr + j\ ]$。

如果 cIdx 等于 1，并且如果 treeType 等于 SINGLE_TREE 或 DUAL_TREE_CHROMA，则以下适用

对于 i = 0..nCurrSw – 1，j = 0..nCurrSh – 1

[0852] $ibcBuf_{Cb}[\ (\ xCurr\ +\ i\ )\ \%\ wIbcBufC\ ][(\ yCurr + j\ )\ \%\ CtbSizeC\ ] =$
$recSamples[\ xCurr + i\ ][\ yCurr + j\ ]$。

如果 cIdx 等于 2，并且如果 treeType 等于 SINGLE_TREE 或 DUAL_TREE_CHROMA，则以下适用

对于 i = 0..nCurrSw – 1，j = 0..nCurrSh – 1

$ibcBuf_{Cr}[\ (\ xCurr\ +\ i\ )\ \%\ wIbcBufC\ ][(\ yCurr + j\ )\ \%\ CtbSizeC\ ] =$
$recSamples[\ xCurr + i\ ][\ yCurr + j\ ]$。

[0853] 5.20实施例#20

[0854] 在本文档中,一些示例中的改变以粗体、下划线和斜体文本进行指示。

[0855] 7.3.7条带数据语法

[0856] 7.3.7.1通用条带数据语法

[0857]

slice_data( ) {	描述符
for(i = 0; i < NumBricksInCurrSlice; i++) {	
CtbAddrInBs = FirstCtbAddrBs[SliceBrickIdx[i]]	
for(j = 0; j < NumCtusInBrick[SliceBrickIdx[i]]; j++, CtbAddrInBs++) {	
if((j % BrickWidth[SliceBrickIdx[i]]) == 0) {	
NumHmvpCand = 0	
NumHmvpIbcCand = 0	
resetIbcBuf = 1	
}	
CtbAddrInRs = CtbAddrBsToRs[CtbAddrInBs]	
coding_tree_unit( )	
if(entropy_coding_sync_enabled_flag && ((j + 1) % BrickWidth[SliceBrickIdx[i]] == 0)) {	
end_of_subset_one_bit /* equal to 1 */	ae(v)
if(j < NumCtusInBrick[SliceBrickIdx[i]] − 1)	
byte_alignment( )	

[0858]

}	
}	
if(!entropy_coding_sync_enabled_flag) {	
end_of_brick_one_bit /* equal to 1 */	ae(v)
if(i < NumBricksInCurrSlice − 1)	
byte_alignment( )	
}	
}	
}	

[0859] 7.4.8.5编解码单元语义

[0860] 当所有以下条件都为真时，通过将NumHmvpSmrIbcCand设置为等于NumHmvpIbcCand，并且将HmvpSmrIbcCandList[i]设置为等于HmvpIbcCandList[i](对于i＝0..NumHmvpIbcCand-1)来更新共享Merge候选列表区域的基于历史的运动矢量预测值列表：

[0861] -IsInSmr[x0][y0]等于TRUE(真)。

[0862] -SmrX[x0][y0]等于x0。

[0863] -SmrY[x0][y0]等于y0。

[0864] 对于x＝x0..x0+cbWidth-1和y＝y0..y0+cbHeight-1，进行以下指定：

[0865] CbPosX[x][y]＝x0 (7-135)

[0866] CbPosY[x][y]＝y0 (7-136)

[0867] CbWidth[x][y]＝cbWidth (7-137)

[0868] CbHeight[x][y]＝cbHeight (7-138)

*将vSize设置为min(ctbSize,64)，并且将wIbcBufY设置为(128*128/CtbSizeY)。*

$ibcBuf_L$是宽度为wIbcBufY并且高度为CtbSizeY的阵列。

$ibcBuf_{Cb}$和$ibcBuf_{Cr}$是宽度为wIbcBufC =(wIbcBufY/SubWidthC)并且高度为(CtbSizeY/SubHeightC)（即，CtbSizeC）的阵列。

如果resetIbcBuf等于1，则以下适用

[0869]

- *对于x = x0..x0 + wIbcBufY – 1和y = y0..y0 + CtbSizeY – 1，$ibcBuf_L$[x % wIbcBufY][y % CtbSizeY] = – 1*
- *对于x = x0..x0 + wIbcBufC – 1和y = y0..y0 + CtbSizeC – 1，$ibcBuf_{Cb}$[x % wIbcBufC][y % CtbSizeC] = – 1*
- *对于x = x0..x0 + wIbcBufC – 1和y = y0..y0 + CtbSizeC – 1，$ibcBuf_{Cr}$[x % wIbcBufC][y % CtbSizeC] = – 1*
- *resetIbcBuf = 0*

当（x0 % vSizeY）等于0并且（y0 % vSizeY）等于0时，以下适用

- *对于 x = x0..x0 + max(vSize, cbWidth) – 1 和 y = y0..y0 + max(vSize, cbHeight)– 1，$ibcBuf_L$[x % wIbcBufY][y % CtbSizeY] = – 1*
- *对 于 x = x0/SubWidthC..x0/ SubWidthC + max(vSize/ SubWidthC, cbWidth) – 1 和 y = y0/SubHeightC..y0/SubHeightC + max(vSize/SubHeightC, cbHeight) – 1，$ibcBuf_{Cb}$[x % wIbcBufC][y % CtbSizeC] = – 1*

[0870]

- *对 于 x = x0/SubWidthC..x0/ SubWidthC + max(vSize/ SubWidthC, cbWidth) – 1 和 y = y0/SubHeightC..y0/SubHeightC + max(vSize/SubHeightC, cbHeight) – 1，$ibcBuf_{Cr}$[x % wIbcBufC][y % CtbSizeC] = – 1*

[0871] 8.6.2 IBC块的运动矢量分量的推导过程

[0872] 8.6.2.1总体

[0873] 该过程的输入是：

[0874] -当前亮度编解码块相对于当前图片的左上角亮度样点的左上角样点的亮度位置(xCb,yCb)，

[0875] -变量cbWidth,指定亮度样点中的当前编解码块的宽度，

[0876] -变量cbHeight,指定亮度样点中的当前编解码块的高度。

[0877] 该过程的输出是：

[0878] -1/16分数样点精度的亮度运动矢量mvL。

[0879] 亮度运动矢量mvL被推导如下：

[0880] -以亮度位置(xCb,yCb)、变量cbWidth和cbHeight作为输入来调用如在条款8.6.2.2中指定的IBC亮度运动矢量预测的推导过程,并且输出是亮度运动矢量mvL。

[0881] -当general_merge_flag[xCb][yCb]等于0时,以下适用：

[0882] 10.变量mvd被推导如下：

[0883] mvd[0]＝MvdL0[xCb][yCb][0] (8-883)

[0884] mvd[1]＝MvdL0[xCb][yCb][1] (8-884)

[0885] 11.以设置为等于mvL的mvX、设置为等于MvShift+2的rightShift和设置为等于MvShift+2的leftShift作为输入并且以取整后的mvL作为输出来调用如在条款8.5.2.14中指定的运动矢量的取整过程。

[0886] 12.亮度运动矢量mvL被修改如下：

[0887] $u[0]＝(mvL[0]+mvd[0]+2^{18})\%2^{18}$ (8-885)

[0888] $mvL[0]＝(u[0]>=2^{17})?(u[0]-2^{18}):u[0]$ (8-886)

[0889] $u[1]＝(mvL[1]+mvd[1]+2^{18})\%2^{18}$ (8-887)

[0890] $mvL[1]＝(u[1]>=2^{17})?(u[1]-2^{18}):u[1]$ (8-888)

[0891] 注1-如上指定的mvL[0]和mvL[1]的结果值总是在范围-2^{17}至$2^{17}-1$(包括-2^{17}和$2^{17}-1$)中。

[0892] 用亮度运动矢量mvL调用如在条款8.6.2.6中指定的基于历史的运动矢量预测值列表的更新过程。

以 mvL 作为输入并且以 mvC 作为输出来调用条款 8.6.2.5。

比特流一致性的要求是，亮度块矢量 mvL 应当遵守以下约束:

[0893]
- *((yCb + (mvL[1] >> 4)) % CtbSizeY) + cbHeight 小于或等于 CtbSizeY*
- *对于 x = xCb..xCb + cbWidth – 1 和 y = yCb..yCb + cbHeight – 1，$ibcBuf_L$[(x + (mvL[0]>>4)) % wIbcBufY][(y +(mvL[1]>>4)) % CtbSizeY]不应等于–1。*

[0894] 8.6.3 ibc块的解码过程

[0895] 8.6.3.1总体

[0896] 当对在ibc预测模式下编解码的编解码单元进行解码时，该过程被调用。

[0897] 该过程的输入是：

[0898] -亮度位置(xCb,yCb)，指定当前编解码块相对于当前图片的左上角亮度样点的左上角样点，

[0899] -变量cbWidth，指定亮度样点中的当前编解码块的宽度，

[0900] -变量cbHeight，指定亮度样点中的当前编解码块的高度，

[0901] -变量cIdx，指定当前块的颜色分量索引。

[0902]
- *运动矢量 mv，*
- *(wIbcBufY)×(CtbSizeY) 阵列 $ibcBuf_L$， (wIbcBufC)×(CtbSizeC) 阵列 $ibcBuf_{Cb}$， (wIbcBufC)×(CtbSizeC)阵列 $ibcBuf_{Cr}$。*

该过程的输出是:

– ***预测样点的阵列predSamples。***

对于x = xCb.. xCb+ Width – 1 和y = yCb..yCb + Height – 1，以下适用

如果cIdx 等于0

predSamples[x][y] = ibcBuf$_L$[(x + mv[0] >> 4)) % wIbcBufY][(y + (mv[1] >> 4)) % CtbSizeY]

[0903] ***如果cIdx 等于1***

predSamples[x][y] = ibcBuf$_{Cb}$[(x + mv[0] >> 5)) % wIbcBufC][(y + (mv[1] >> 5)) % CtbSizeC]

如果cIdx 等于2

predSamples[x][y] = ibcBuf$_{Cr}$[(x + mv[0] >> 5)) % wIbcBufC][(y + (mv[1] >> 5)) % CtbSizeC]

[0904] 8.7.5图片重构过程

[0905] 8.7.5.1总体

[0906] 该过程的输入是：

[0907] -位置(xCurr,yCurr)，指定当前块相对于当前图片分量的左上角样点的左上角样点，

[0908] -变量nCurrSw和nCurrSh，分别指定当前块的宽度和高度，

[0909] -变量cIdx，指定当前块的颜色分量，

[0910] -(nCurrSw)x(nCurrSh)阵列predSamples，指定当前块的预测样点，

[0911] -(nCurrSw)x(nCurrSh)阵列resSamples，指定当前块的残差样点。

[0912] 该过程的输出是重构图片样点阵列recSamples和IBC缓冲区阵列ibcBuf$_L$、ibcBuf$_{Cb}$、ibcBuf$_{Cr}$。

[0913] 取决于颜色分量cIdx的值，进行以下指定：

[0914] -如果cIdx等于0，则recSamples对应于重构图片样点阵列S_L，并且函数clipCidx1对应于Clip1$_Y$。

[0915] -否则，如果cIdx等于1，则tuCbfChroma被设置为等于tu_cbf_cb[xCurr][yCurr]，recSamples对应于重构色度样点阵列S_{Cb}，并且函数clipCidx1对应于Clip1$_C$。

[0916] -否则(cIdx等于2)，tuCbfChroma被设置为等于tu_cbf_cr[xCurr][yCurr]，recSamples对应于重构色度样点阵列S_{Cr}，并且函数clipCidx1对应于Clip1$_C$。

[0917] 取决于slice_lmcs_enabled_flag的值，以下适用：

[0918] -如果slice_lmcs_enabled_flag等于0，则对于i＝0..nCurrSw-1，j＝0..nCurrSh-1，位置(xCurr,yCurr)处的重构样点recSamples的(nCurrSw)x(nCurrSh)块被推导如下：

[0919] recSamples[xCurr+i][yCurr+j]＝clipCidx1(predSamples[i][j]+resSamples[i][j]) (8-992)

[0920] -否则(slice_lmcs_enabled_flag等于1)，以下适用：

[0921] -如果cIdx等于0，则以下适用：

[0922] -以亮度位置(xCurr,yCurr)、块宽度nCurrSw和高度nCurrSh、预测亮度样点阵列predSamples以及残差亮度样点阵列resSamples作为输入来调用如在条款8.7.5.2中指定的亮度样点的图片重构与映射过程,并且输出是重构亮度样点阵列recSamples。

[0923] -否则(cIdx大于0),以色度位置(xCurr,yCurr)、变换块宽度nCurrSw和高度nCurrSh、当前色度变换块的编解码块标志tuCbfChroma、预测色度样点阵列predSamples以及残差色度样点阵列resSamples作为输入来调用如在条款8.7.5.3中指定的色度样点的图片重构与依赖于亮度的色度残差缩放过程,并且输出是重构色度样点阵列recSamples。

[0924] ***在对当前编解码单元进行解码之后，以下可以适用:***

如果 cIdx 等于 0，并且如果 treeType 等于 SINGLE_TREE 或 DUAL_TREE_LUMA，则以下适用

对于 i = 0..nCurrSw – 1，j = 0..nCurrSh – 1

$$ibcBuf_L[\ (\ xCurr\ +\ i\ )\ \%\ wIbcBufY\ ][\ (\ yCurr + j\ )\ \%\ CtbSizeY\ ] = recSamples[\ xCurr + i\ ][\ yCurr + j\ ]。$$

如果 cIdx 等于 1，并且如果 treeType 等于 SINGLE_TREE 或 DUAL_TREE_CHROMA，则以下适用

对于 i = 0..nCurrSw – 1，j = 0..nCurrSh – 1

$$ibcBuf_{Cb}[\ (\ xCurr\ +\ i\ )\ \%\ wIbcBufC\ ][\ (\ yCurr + j\ )\ \%\ CtbSizeC\ ] = recSamples[\ xCurr + i\ ][\ yCurr + j\ ]。$$

[0925] ***如果 cIdx 等于 2，并且如果 treeType 等于 SINGLE_TREE 或 DUAL_TREE_CHROMA，则以下适用***

对于 i = 0..nCurrSw – 1，j = 0..nCurrSh – 1

$$ibcBuf_{Cr}[\ (\ xCurr\ +\ i\ )\ \%\ wIbcBufC\ ][\ (\ yCurr + j\ )\ \%\ CtbSizeC\ ] = recSamples[\ xCurr + i\ ][\ yCurr + j\ ]。$$

[0926] 图6是可视媒体(视频或图像)处理的示例方法600的流程图。方法600包括:对于当前视频块和当前视频块的比特流表示之间的转换,确定(602)缓冲区的尺寸以存储使用帧内块复制编解码模式的当前视频块的参考样点;以及使用存储在缓冲区中的参考样点来执行(604)该转换。

[0927] 以下条款描述了由方法600和其它方法的实施例实施的一些示例优选特征。在本文档的第4章节中提供了附加示例。

[0928] 1．一种视频处理的方法，包括:对于当前视频块和当前视频块的比特流表示之间的转换,确定缓冲区的尺寸以存储使用帧内块复制编解码模式的当前视频块的参考样点；以及使用存储在缓冲区中的参考样点来执行该转换。

[0929] 2．根据条款1所述的方法,其中,缓冲区的尺寸为预定常数。

[0930] 3．根据条款1-2中任一项所述的方法,其中,尺寸为M×N,其中M和N为整数。

[0931] 4．根据条款3所述的方法,其中,M×N等于64×64或128×128或64×128。

[0932] 5．根据条款1所述的方法,其中,缓冲区的尺寸等于当前视频块的编解码树单元

的尺寸。

[0933] 6．根据条款1所述的方法，其中，缓冲区的尺寸等于在该转换期间使用的虚拟管道数据单元的尺寸。

[0934] 7．根据条款1所述的方法，其中，缓冲区的尺寸对应于比特流表示中的字段。

[0935] 8．根据条款7所述的方法，其中，该字段被包括在视频参数集或序列参数集或图片参数集或图片头或条带头或片组头级别的比特流表示中。

[0936] 9．根据条款1-8中任一项所述的方法，其中，对于亮度分量的参考样点和色度分量的参考样点，缓冲区的尺寸不同。

[0937] 10．根据条款1-8中任一项所述的方法，其中，缓冲区的尺寸取决于当前视频块的色度子采样格式。

[0938] 11．根据条款1-8中任一项所述的方法，其中，参考样点以RGB格式被存储。

[0939] 12．根据条款1-11中任一项所述的方法，其中，缓冲区用于存储环路滤波之前的重构样点和环路滤波之后的重构样点。

[0940] 13．根据条款12所述的方法，其中，环路滤波包括去方块滤波或自适应环路滤波(ALF)或样点自适应偏移(SAO)滤波。

[0941] 14．一种视频处理的方法，包括：对于当前视频块和当前视频块的比特流表示之间的转换，使用参考样点的初始值来初始化缓冲区，以存储使用帧内块复制编解码模式的当前视频块的参考样点；以及使用存储在缓冲区中的参考样点来执行该转换。

[0942] 15．根据条款14所述的方法，其中，初始值对应于常数。

[0943] 16．根据条款14-15中任一项所述的方法，其中，初始值是当前视频块的比特深度的函数。

[0944] 17．根据条款15所述的方法，其中，常数对应于中间灰度值。

[0945] 18．根据条款14所述的方法，其中，初始值对应于先前解码的视频块的像素值。

[0946] 19．根据条款18所述的方法，其中，先前解码的视频块对应于环路滤波之前的解码块。

[0947] 20．根据条款14-19中任一项所述的方法，其中，缓冲区的尺寸如条款1-13中的一项所述。

[0948] 21．根据条款1-20中任一项所述的方法，其中，缓冲区内的像素位置使用x数字和y数字进行寻址。

[0949] 22．根据条款1-20中任一项所述的方法，其中，缓冲区内的像素位置使用从0扩展到M*N-1的单个数字进行寻址，其中M和N是缓冲区的像素宽度和像素高度。

[0950] 23．根据条款1-20中任一项所述的方法，其中，当前比特流表示包括用于该转换的块矢量，其中，表示为(BVx,BVy)的块矢量等于(x-x0,y-y0)，其中(x0,y0)对应于当前视频块的编解码树单元的左上角位置。

[0951] 24．根据条款1-20中任一项所述的方法，其中，当前比特流表示包括用于该转换的块矢量，其中，表示为(BVx,BVy)的块矢量等于(x-x0+Tx,y-y0+Ty)，其中(x0,y0)对应于当前视频块的编解码树单元的左上角位置，并且其中，Tx和Ty是偏移值。

[0952] 25．根据条款24所述的方法，其中，Tx和Ty是预定义的偏移值。

[0953] 26．根据条款1-20中任一项所述的方法，其中，在该转换期间，对于在位置(x0，

y0)处并具有块矢量(BVx,BVy)的像素,缓冲区中的对应参考在参考位置(x0+BVx,y0+BVy)处被找到。

[0954] 27. 根据条款26所述的方法,其中,在参考位置在缓冲区之外的情况下,缓冲区中的参考通过在缓冲区的边界处进行裁剪而确定。

[0955] 28. 根据条款26所述的方法,其中,在参考位置在缓冲区之外的情况下,缓冲区中的参考被确定为具有预定值。

[0956] 29. 根据条款1-20中任一项所述的方法,其中,在该转换期间,对于在位置(x0,y0)处并具有块矢量(BVx,BVy)的像素,缓冲区中的对应参考在参考位置((x0+BVx)mod M,(y0+BVy)mod N)处被找到,其中"mod"是模运算,并且M和N是表示缓冲区的x维度和y维度的整数。

[0957] 30. 一种视频处理的方法,包括:在视频和当前视频块的比特流表示之间的转换期间,重置存储用于视频边界处的帧内块复制编解码的参考样点的缓冲区;以及使用存储在缓冲区中的参考样点来执行该转换。

[0958] 31. 根据条款30所述的方法,其中,视频边界对应于新图片或新片。

[0959] 32. 根据条款30所述的方法,其中,该转换通过在重置之后用虚拟管道数据单元(VPDU)的重构值更新缓冲区而执行。

[0960] 33. 根据条款30所述的方法,其中,该转换通过在重置之后用编解码树单元的重构值更新缓冲区而执行。

[0961] 34. 根据条款30所述的方法,其中,重置在每个编解码树单元行的开始处被执行。

[0962] 35. 根据条款1所述的方法,其中,缓冲区的尺寸对应于L个64×64的先前解码的块,其中L为整数。

[0963] 36. 根据条款1-35中任一项所述的方法,其中,垂直扫描顺序用于在该转换期间读取缓冲区中的样点或将样点存储在缓冲区中。

[0964] 37. 一种视频处理的方法,包括:对于当前视频块和当前视频块的比特流表示之间的转换,使用缓冲区来存储使用帧内块复制编解码模式的当前视频块的参考样点,其中,缓冲区的第一比特深度不同于编解码数据的第二比特深度;以及使用存储在缓冲区中的参考样点来执行该转换。

[0965] 38. 根据条款37所述的方法,其中,第一比特深度大于第二比特深度。

[0966] 39. 根据条款37-38中任一项所述的方法,其中,第一比特深度与在该转换期间使用的重构缓冲区的比特深度相同。

[0967] 40. 根据条款37-39中任一项所述的方法,其中,第一比特深度作为值或差值在比特流表示中被信令通知。

[0968] 41. 根据条款37-40中任一项所述的方法,其中,对于色度分量和亮度分量,该转换使用不同的比特深度。

[0969] 在第4章节的第7项中描述了条款37至41的附加实施例和示例。

[0970] 42. 一种视频处理的方法,包括:执行使用帧内块复制模式的当前视频块和当前视频块的比特流表示之间的转换,其中在该帧内块复制模式下,在该转换期间用于预测计算的第一精确度低于在该转换期间用于重构计算的第二精确度。

[0971] 43. 根据条款43所述的方法,其中,预测计算包括使用clip{{p+[1<<(b-1)]}>>b,

0，(1<<bitdepth)-1}<<b从重构样点值确定预测样点值，其中p是重构样点值，b是预定义的比特移位值，并且bitdepth是预测样点精确度。

[0972] 在第4章节的第28至31项和第34项中描述了条款42至43的附加实施例和示例。

[0973] 44．一种视频处理的方法，包括：执行使用帧内块复制模式的当前视频块和当前视频块的比特流表示之间的转换，其中在该帧内块复制模式下，尺寸为nM×nM的参考区域用于尺寸为M×M的编解码树单元，其中n和M为整数，并且其中，当前视频块位于编解码树单元中，并且其中，参考区域是对应于当前视频块的编解码树单元行中的最近的可用的n×n编解码树单元。

[0974] 在第4章节的第35项中描述了条款4的附加实施例和示例。

[0975] 45．一种视频处理的方法，包括：执行使用帧内块复制模式的当前视频块和当前视频块的比特流表示之间的转换，其中在该帧内块复制模式下，尺寸为nM×nM的参考区域用于尺寸为除M×M之外的编解码树单元，其中n和M为整数，并且其中，当前视频块位于编解码树单元中，并且其中，参考区域是对应于当前视频块的编解码树单元行中的最近的可用的n×n-1编解码树单元。

[0976] 在第4章节的第36项中描述了条款4的附加实施例和示例。图8和图9示出了附加的示例实施例。

[0977] 46．根据条款3所述的方法，其中，M＝mW并且N＝H，其中W和H是当前视频块的编解码树单元(CTU)的宽度和高度，并且m是正整数。

[0978] 47．根据条款3所述的方法，其中，M＝W并且N＝nH，其中W和H是编解码树单元(CTU)的宽度和高度，并且n是正整数。

[0979] 48．根据条款3所述的方法，其中，M＝mW并且N＝nH，其中W和H是编解码树单元(CTU)的宽度和高度，m和n是正整数。

[0980] 49．根据条款46-48中任一项所述的方法，其中，n和m取决于CTU的尺寸。

[0981] 50．一种视频处理的方法，包括：对于视频的当前视频块和当前视频块的比特流表示之间的转换，使用视频的分量X来确定与视频的分量c的当前视频块相对应的块矢量的有效性，其中，分量X不同于视频的亮度分量；以及当确定块矢量对于当前视频块有效时，使用块矢量来执行该转换。这里，表示为(BVx,BVy)的块矢量等于(x-x0,y-y0)，其中(x0,y0)对应于当前视频块的编解码树单元的左上角位置。

[0982] 51．根据条款50所述的方法，其中，分量c对应于视频的亮度分量。

[0983] 52．根据条款50所述的方法，其中，当前视频块是色度块，并且视频为4:4:4格式。

[0984] 53．根据条款50所述的方法，其中，视频为4:2:0格式，并且其中，当前视频块是在位置(x,y)处开始的色度块，并且其中，该确定包括将块矢量确定为对于其中isRec(c, ((x+BVx)>>5<<5)+64-(((y+BVy)>>5)&1)*32+(x%32),((y+BVy)>>5<<5) +(y%32))为真的情况无效。

[0985] 54．根据条款50所述的方法，其中，视频为4:2:0格式，并且其中，当前视频块是在位置(x,y)处开始的色度块，并且其中，该确定包括将块矢量确定为对于如果isRec(c, x+BVx+Chroma_CTU_size, y)为真的情况无效。

[0986] 55．一种视频处理的方法，包括：对于视频区域的当前虚拟管道数据单元(VPDU)的当前视频块和当前视频块的比特流表示之间的转换，选择性地确定使用来自视频区域的

第一行的K1个先前处理的VPDU和来自视频区域的第二行的K2个先前处理的VPDU;以及执行该转换,其中,该转换不包括使用当前VPDU的剩余部分。

[0987] 56. 根据条款55所述的方法,其中,K1 = 1并且K2 = 2。

[0988] 57. 根据条款55-56中任一项所述的方法,其中,当前视频块基于视频区域的大小或当前VPDU的大小而选择性地处理。

[0989] 58. 一种视频处理的方法,包括:为当前视频块和当前视频块的比特流表示之间的转换执行块矢量的有效性检查,其中,块矢量用于帧内块复制模式;以及在该转换期间使用有效性检查的结果来选择性地使用块矢量。

[0990] 59. 根据条款58所述的方法,其中,帧内块复制(IBC)缓冲区在该转换期间被使用,其中,IBC缓冲区的宽度和高度为Wbuf和Hbuf,当前视频块的大小为W×H,并且其中,块矢量被表示为(BVx,BVy),并且其中,当前视频块在具有大小Wpic和Hpic的当前图片以及具有Wctu和Hctu作为宽度和高度的编解码树单元中,并且其中,有效性检查使用预定规则。

[0991] 60. 根据条款58-59中任一项所述的方法,其中,当前视频块是从像素坐标(X,Y)开始的亮度块、色度块、编解码单元CU、变换单元TU、4×4块、2×2块或父块的子块。

[0992] 61. 根据条款58-60中任一项所述的方法,其中,有效性检查将落在当前图片的边界之外的块矢量视为有效。

[0993] 62. 根据条款58-60中任一项所述的方法,其中,有效性检查将落在编解码树单元的边界之外的块矢量视为有效。

[0994] 先前章节中的第23-30项提供了以上条款58-62的附加示例和变型。

[0995] 63. 根据条款1-62中任一项所述的方法,其中,该转换包括从当前视频块生成比特流表示。

[0996] 64. 根据条款1-62中任一项所述的方法,其中,该转换包括从比特流表示生成当前视频块的像素值。

[0997] 65. 一种视频编码器装置,包括被配置为实施根据条款1-62中任一项或多项所述的方法的处理器。

[0998] 66. 一种视频解码器装置,包括被配置为实施根据条款1-62中任一项或多项所述的方法的处理器。

[0999] 67. 一种其上存储有代码的计算机可读介质,该代码体现用于实施根据条款1-62中任一项或多项所述的方法的处理器可执行指令。

[1000] 图7是视频/图像处理装置700的硬件平台的框图。装置700可以用于实施本文描述的方法中的一种或多种。装置700可以体现在智能电话、平板电脑、计算机、物联网(Internet of Things,IoT)接收器等中。装置700可以包括一个或多个处理器702、一个或多个存储器704、以及视频处理硬件706。(多个)处理器702可以被配置为实施本文档中描述的一种或多种方法(包括但不限于方法600)。存储器(多个存储器)704可以用于存储用于实施本文描述的方法和技术的数据和代码。视频处理硬件706可以用于以硬件电路实施本文档中描述的一些技术。

[1001] 对应于当前视频块的比特流表示不需要是连续的比特集合,并且可以跨头、参数集和网络抽象层(Network Abstraction Layer,NAL)分组而分布。

[1002] 章节A:另一附加示例实施例

[1003] 在章节A中,我们呈现了另一示例实施例,其中可以修改VVC标准的当前版本,以实施在本文档中描述的技术中的一些。

[1004] 本章节分析当前IBC参考缓冲区设计中的几个问题,并呈现不同的设计来解决问题。代替与解码内存混合,提出了一种独立的IBC参考缓冲区来。与当前锚(anchor)相比,所提出的方案示出了类F的-0.99%/-0.71%/-0.79%的AI/RA/LD-B亮度BD率和4:2:0 TGM的-2.57%/-1.81%/-1.36%的AI/RA/LD-B亮度BD率,其中6.7%的内存减少;或类F的-1.31%/-1.01%/-0.81%的AI/RA/LD-B亮度BD率和4:2:0 TGM的-3.23%/-2.33%/-1.71%的AI/RA/LD-B亮度BD率,其中6.7%的内存增加。

[1005] A1.介绍

[1006] 采用帧内块复制,即IBC(或当前图片参考,即先前的CPR)编解码模式。已经意识到,IBC参考样点应该被存储在片上内存中,因此定义了一个CTU的有限参考区域。为了限制用于缓冲区的额外片上内存,当前设计重用64×64内存以用于对当前VPDU进行解码,使得仅需要3个附加的64×64块的内存来支持IBC。当CTU尺寸为128×128时,参考区域当前在图2中被示出。

[1007] 在当前草案(VVC草案4)中,区域被定义为:

[1008]

– 以下条件应当为真:

$$(yCb + (mvL[1] >> 4)) >> CtbLog2SizeY = yCb >> CtbLog2SizeY \quad (8\text{-}972)$$

$$(yCb + (mvL[1] >> 4) + cbHeight - 1) >> CtbLog2SizeY = yCb >> CtbLog2SizeY \quad (8\text{-}973)$$

$$(xCb + (mvL[0] >> 4)) >> CtbLog2SizeY >= (xCb >> CtbLog2SizeY) - 1 \quad (8\text{-}974)$$

$$(xCb + (mvL[0] >> 4) + cbWidth - 1) >> CtbLog2SizeY <= (xCb >> CtbLog2SizeY) \quad (8\text{-}975)$$

[Ed. (SL): 条件(8-218)和(8-216)可能已经通过6.4.X进行检查。]

[1009]

– 当(xCb + (mvL[0] >> 4)) >> CtbLog2SizeY 等于(xCb >> CtbLog2SizeY) – 1 时,以设置为(xCb, yCb)的当前亮度位置(xCurr, yCurr)以及邻近亮度位置(((xCb + (mvL[0] >> 4) + CtbSizeY) >> (CtbLog2SizeY – 1)) << (CtbLog2SizeY – 1), ((yCb + (mvL[1] >> 4)) >> (CtbLog2SizeY – 1)) << (CtbLog2SizeY – 1))作为输入来调用如在条款 6.4.X 中指定的块可用性的推导过程[Ed. (BB): 邻近块可用性检查过程tbd],并且输出应当等于FALSE(假)。

[1010] 因此,总的参考尺寸为CTU。

[1011] A2.当前设计的潜在问题

[1012] 当前设计假设重用64×64内存以用于对当前VPDU进行解码,并且相应地将IBC参考与VPDU内存重用对齐。这样的设计将VPDU解码内存与IBC缓冲区捆绑在一起。可能存在几个问题:

[1013] 1.处理更小的CTU可能是个问题。假设CTU尺寸为32×32,不清楚用于对当前VPDU

进行解码的当前64×64内存是否可以在不同的架构中有效地支持32×32级别内存重用。

[1014] 2.参考区域显著变化。因此,引入了过多的比特流一致性约束。它给编码器有效地利用参考区域并避免生成合法的比特流增加了额外的负担。这也增加了在不同模块(例如,Merge列表)中具有无效BV的可能性。为了处理那些无效BV,可能会引入额外的逻辑或额外的一致性约束。它不仅会给编码器或解码器引入负担,还可能会造成BV编解码和MV编解码之间的差异。

[1015] 3.该设计不能很好地扩展。由于VPDU解码与IBC缓冲区混合,相对于当前的一个128×128 CTU的设计增加或减少参考区域并不容易。这可能会限制在以后的开发(例如,更低或更高的简表)中利用更好的编解码效率对片上内存的折衷的灵活性。

[1016] 4.IBC参考缓冲区的比特深度与解码缓冲区相关联。即使屏幕内容通常具有比内部解码比特深度更低的比特深度,缓冲区仍然需要花费内存来存储主要表示取整或量化噪声的比特。当考虑更高的解码比特深度配置时,这个问题变得更加严重。

[1017] A3.清晰的IBC缓冲区设计

[1018] 为了解决以上小节中列出的问题,我们提出利用专用IBC缓冲区,它不与解码内存混合。

[1019] 对于128×128 CTU,缓冲区被定义为具有8比特样点的128×128,当尺寸为w×h的CU(x,y)已经被解码时,其在环路滤波之前的重构被转换为8比特,并被写入从位置(x%128,y%128)开始的w×h块区域。这里,模运算符%总是返回正数,即对于x<0,$x\%L \triangleq -(-x\%L)$,例如-3%128=125。

[1020] 假设用BV=(BVx,BVy)在IBC模式下对像素(x,y)进行编解码,它在IBC参考缓冲区中的预测样点位于((x+BVx)%128,(y+BVy)%128)处,并且像素值将在预测之前被转换为10比特。

[1021] 当缓冲区被认为是(W,H)时,在对从(x,y)开始的CTU或CU进行解码之后,环路滤波之前的重构像素将被存储在从(x%W,y%H)开始的缓冲区中。因此,在对CTU进行解码之后,对应的IBC参考缓冲区将被相应地更新。当CTU尺寸不为128×128时,这样的设置可能会发生。例如,对于64×64 CTU,在当前缓冲区尺寸的情况下,它可以被认为是256×64缓冲区。对于64×64 CTU,图2示出了缓冲区状态。

[1022] 图12是IBC参考缓冲区状态的图示,其中块表示64×64 CTU。

[1023] 在这样的设计中,因为IBC缓冲区不同于VPDU解码内存,所以所有的IBC参考缓冲区都可以被用作参考。

[1024] 当IBC缓冲区的比特深度为8比特时,与需要3个附加的10比特64×64缓冲区的当前设计相比,片上内存增加为(8*4)/(10*3)-100%=6.7%。

[1025] 如果我们进一步降低比特深度。可以进一步降低内存需求。例如,对于7比特缓冲区,片上内存节省为100%-(7*4)/(10*3)= 6.7%。

[1026] 在该设计的情况下,唯一的比特流一致性约束是参考块应当在当前片的当前CTU行中的重构区域内。

[1027] 当在每个CTU行的开始处允许初始化到512时,可以移除所有比特流一致性约束。

[1028] A4. 实验结果

[1029] 在一些实施例中,所公开的方法可以使用VTM-4.0软件而实施。

[1030] 对于10比特缓冲区实施方式和CTC,解码器与当前的VTM4.0编码器完全兼容,这意味着所提出的解码器可以正确地解码VTM-4.0 CTC比特流。

[1031] 对于7比特缓冲区实施方式,结果如表I所示

[1032] 对于8比特缓冲区实施方式,结果如表II所示。

[1033] 表I. 7比特缓冲区的性能。锚是VTM-4.0,其中IBC对所有序列开启。

	所有帧内				
	基于 VTM-4.0，其中 IBC 开启				
	Y	U	V	EncT	DecT
类 A1	-0.01%	-0.09%	-0.10%	132%	101%
类 A2	0.05%	0.00%	0.06%	135%	100%
类 B	0.00%	-0.02%	0.01%	135%	100%
类 C	-0.02%	0.01%	0.03%	130%	98%
类 E	-0.13%	-0.16%	-0.04%	135%	99%
总体	-0.02%	-0.05%	0.00%	133%	100%
类 D	0.04%	0.04%	0.12%	127%	107%
类 F	-0.99%	-1.14%	-1.18%	115%	99%
4:2:0 TGM	-2.57%	-2.73%	-2.67%	104%	102%

[1034]

	随机访问				
	基于 VTM-4.0，其中 IBC 开启				
	Y	U	V	EncT	DecT
类 A1	0.02%	-0.01%	0.01%	109%	100%
类 A2	0.00%	-0.04%	0.03%	111%	100%
类 B	-0.01%	-0.10%	-0.22%	113%	101%
类 C	-0.01%	0.17%	0.12%	115%	100%
类 E					
总体	0.00%	0.00%	-0.04%	112%	100%
类 D	0.05%	0.16%	0.20%	117%	101%
类 F	-0.71%	-0.77%	-0.77%	109%	99%
4:2:0 TGM	-1.81%	-1.65%	-1.64%	107%	101%

	低延迟 B				
	基于 VTM-4.0，其中 IBC 开启				
	Y	U	V	EncT	DecT
类 A1					
类 A2					

[1035]

类 B	0.01%	0.36%	0.30%	114%	95%
类 C	-0.01%	-0.12%	-0.10%	120%	98%
类 E	0.10%	0.20%	0.18%	107%	99%
总体	0.03%	0.16%	0.13%	114%	97%
类 D	-0.01%	1.07%	0.18%	123%	104%
类 F	-0.79%	-0.89%	-1.01%	110%	100%
4:2:0 TGM	-1.36%	-1.30%	-1.26%	109%	102%

[1036] 表II.8比特缓冲区的性能。锚是VTM-4.0，其中IBC对所有序列开启。

[1037]

	所有帧内				
	基于 VTM-4.0，其中 IBC 开启				
	Y	U	V	EncT	DecT
类 A1	-0.01%	0.02%	-0.10%	129%	102%
类 A2	0.02%	-0.06%	-0.02%	134%	102%
类 B	-0.04%	-0.02%	-0.07%	135%	101%
类 C	-0.03%	0.04%	0.00%	130%	98%
类 E	-0.16%	-0.14%	-0.08%	134%	100%
总体	-0.04%	-0.03%	-0.05%	133%	100%
类 D	0.00%	0.04%	0.02%	126%	101%
类 F	-1.31%	-1.27%	-1.29%	114%	98%
4:2:0 TGM	-3.23%	-3.27%	-3.24%	101%	100%

	随机访问				
	基于 VTM-4.0，其中 IBC 开启				
	Y	U	V	EncT	DecT
类 A1	-0.01%	-0.08%	0.04%	107%	99%
类 A2	-0.03%	-0.16%	0.06%	110%	99%
类 B	-0.01%	-0.14%	-0.22%	111%	99%
类 C	-0.01%	0.15%	0.09%	115%	100%
类 E					
总体	-0.01%	-0.05%	-0.03%	111%	99%

类 D	0.01%	0.19%	0.22%	116%	101%
类 F	-1.01%	-0.99%	-1.01%	108%	99%
4:2:0 TGM	-2.33%	-2.14%	-2.19%	105%	100%

[1038]

	低延迟 B				
	基于 VTM-4.0，其中 IBC 开启				
	Y	U	V	EncT	DecT
类 A1					
类 A2					
类 B	0.00%	0.04%	-0.14%	113%	#NUM!
类 C	-0.05%	-0.28%	-0.15%	119%	98%
类 E	0.04%	-0.16%	0.43%	107%	#NUM!
总体	0.00%	-0.11%	0.00%	113%	#NUM!
类 D	-0.07%	1.14%	0.13%	122%	99%
类 F	-0.81%	-0.92%	-0.96%	111%	99%
4:2:0 TGM	-1.71%	-1.67%	-1.71%	106%	95%

[1039] 图17是示出可以在其中实施本文公开的各种技术的示例视频处理系统1700的框图。各种实施方式可以包括系统1700的一些或所有组件。系统1700可以包括用于接收视频内容的输入1702。视频内容可以以例如8或10比特多分量像素值的原始或未压缩格式而接收，或者可以是压缩或编码格式。输入1702可以表示网络接口、外围总线接口或存储接口。网络接口的示例包括诸如以太网、无源光网络(Passive Optical Network,PON)等的有线接口和诸如Wi-Fi或蜂窝接口的无线接口。

[1040] 系统1700可以包括可以实施本文档中描述的各种编解码或编码方法的编解码组件1704。编解码组件1704可以将来自输入1702的视频的平均比特率减小到编解码组件1704的输出，以产生视频的编解码表示。编解码技术因此有时被称为视频压缩或视频转码技术。编解码组件1704的输出可以被存储，或者经由如由组件1706表示的通信连接而发送。在输入1702处接收的视频的存储或通信传送的比特流(或编解码)表示可以由组件1708用于生成像素值或传送到显示接口1710的可显示视频。从比特流表示生成用户可视视频的过程有时被称为视频解压缩。此外，虽然某些视频处理操作被称为“编解码”操作或工具，但是将理解，编解码工具或操作在编码器处被使用，并且反转编解码结果的对应的解码工具或操作将由解码器执行。

[1041] 外围总线接口或显示接口的示例可以包括通用串行总线(Universal SerialBus,USB)、或高清晰度多媒体接口(High Definition Multimedia Interface,HDMI)、或显示端口(Displayport)等。存储接口的示例包括SATA(SerialAdvanced Technology

Attachment,串行高级技术附件)、PCI、IDE接口等。本文档中描述的技术可以体现在各种电子设备中,诸如移动电话、膝上型电脑、智能电话、或能够执行数字数据处理和/或视频显示的其他设备。

[1042] 图18是可视数据处理的示例方法的流程图。该流程图的步骤结合在本文档的第4章节中讨论的示例18进行讨论。在步骤1802处,该过程对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换,确定存储用于帧内块复制模式下的预测的参考样点的缓冲区,其中,该转换在帧内块复制模式下被执行,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息。在步骤1804处,对于在空域上位于当前视频块相对于包括当前视频块的编解码树单元的左上角位置的位置(x0,y0)处并具有块矢量(BVx,BVy)的样点,该过程计算缓冲区中在参考位置(P,Q)处的对应参考,其中,参考位置(P,Q)是使用块矢量(BVx,BVy)和位置(x0,y0)而确定的。当确定参考位置(P,Q)位于缓冲区之外时,在步骤1805处,该过程至少部分基于当前视频块相对于包括当前视频块的编解码树单元的位置来重新计算参考位置。

[1043] 图19是可视数据处理的示例方法的流程图。该流程图的步骤结合在本文档的第4章节中讨论的示例29进行讨论。在步骤1902处,该过程对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换,确定存储用于帧内块复制模式下的预测的参考样点的缓冲区,其中,该转换在帧内块复制模式下被执行,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息。在步骤1904处,对于在空域上位于当前视频块相对于包括当前视频块的图片的左上角位置的位置(x,y)处并具有块矢量(BVx,BVy)的样点,该过程至少部分基于满足与以下中的至少一个相关联的一个或多个条件来将块矢量(BVx,BVy)指定为有效:当前视频块的位置(x,y)、当前视频块的大小、图片的大小、包括当前视频块的编解码树单元的大小、或缓冲区的大小。在步骤1906处,该过程执行检查以确定块矢量(BVx,BVy)有效。在步骤1908处,当识别出块矢量(BVx,BVy)有效时,该过程计算缓冲区中在参考位置(P,Q)处的对应参考,其中,参考位置(P,Q)是使用块矢量(BVx,BVy)、位置(x,y)以及缓冲区的大小而确定的。

[1044] 图20是可视数据处理的示例方法的流程图。该流程图的步骤结合在本文档的第4章节中讨论的示例19进行讨论。在步骤2002处,该过程对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换,确定当前视频块的块矢量(BVx,BVy)或块矢量差(BVDx,BVDy),其中,该转换在帧内块复制模式下被执行,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息。在步骤2004处,该过程将块矢量(BVx,BVy)的至少一个分量或块矢量差(BVDx,BVDy)的至少一个分量归一化为位于一范围内。

[1045] 图21是可视数据处理的示例方法的流程图。该流程图的步骤结合在本文档的第4章节中讨论的示例40进行讨论。在步骤2102处,该过程对于当前视频块和当前视频块的比特流表示之间的转换,确定用于存储用于帧内块复制模式下的预测的重构样点的缓冲区,其中,该转换在帧内块复制模式下被执行,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息。在步骤2104处,该过程根据顺序来更新存储在缓冲区中的重构样点。

[1046] 图22是可视数据处理的示例方法的流程图。该流程图的步骤结合在本文档的第4

章节中讨论的示例57进行讨论。在步骤2202处，该过程执行当前视频块和当前视频块的比特流表示之间的转换，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，用于预测计算的第一精确度低于用于重构计算的第二精确度。

[1047] 图23是可视数据处理的示例方法的流程图。该流程图的步骤结合在本文档的第4章节中讨论的示例59进行讨论。在步骤2302处，该过程使用帧内块复制模式来执行当前视频块和当前视频块的比特流表示之间的转换，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，尺寸为nM×nM的参考区域被使用，其中n和M为整数，并且其中，当前视频块位于编解码树单元中，并且其中，参考区域包括来自对应于当前视频块的编解码树单元行中的n×n个最近的可用编解码树单元的样点。

[1048] 图24是可视数据处理的示例方法的流程图。该流程图的步骤结合在本文档的第4章节中讨论的示例60进行讨论。在步骤2402处，该过程使用帧内块复制模式来执行当前视频块和当前视频块的比特流表示之间的转换，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，尺寸为nM×pM的参考区域被使用，其中n、p和M为整数，并且其中，当前视频块位于编解码树单元中，并且其中，参考区域包括来自对应于当前视频块的编解码树单元行中的n×p-1个最近的可用编解码树单元的样点。

[1049] 图25是可视数据处理的示例方法的流程图。该流程图的步骤结合在本文档的第4章节中讨论的示例61进行讨论。在步骤2502处，该过程使用帧内块复制模式来执行视频区域的虚拟管道数据单元(VPDU)的当前视频块和当前视频块的比特流表示之间的转换，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，尺寸为nM×nM的参考区域被使用，VPDU的尺寸为kM×kM，其中k、n和M为整数，并且其中，当前视频块位于编解码树单元中，并且其中，参考区域包括来自对应于当前视频块的编解码树单元行中的n×n-k个最近的可用编解码树单元的样点。

[1050] 图26是可视数据处理的示例方法的流程图。该流程图的步骤结合在本文档的第4章节中讨论的示例62-66进行讨论。在步骤2602处，该过程对于可视媒体数据的尺寸为w×h的当前视频块和当前视频块的比特流表示之间的转换，确定存储用于帧内块复制模式下的预测的参考样点的缓冲区，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息。在步骤2604处，对于在空域上位于当前视频块相对于包括当前视频块的尺寸为M×M的编解码树单元(CTU)的左上角位置的位置(x0,y0)处并具有块矢量(BVx,BVy)的样点，该过程计算在缓冲区中的参考位置(P,Q)处开始的对应参考区域，其中，参考位置(P,Q)是使用块矢量(BVx,BVy)和/或位置(x0,y0)而确定的。在步骤2606处，该过程将一个或多个基于规则的约束应用于参考区域和/或参考位置(P,Q)，以限制参考区域与视频区域的重叠。

[1051] 图27是可视数据处理的示例方法的流程图。该流程图的步骤结合在本文档的第4章节中讨论的示例68进行讨论。在步骤2702处，该过程对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换，确定存储用于帧内块复制模式下的预测的参考样点的缓冲区，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前

视频块位于相同视频区域中的重构块相关的运动信息。在步骤2704处，对于在空域上位于当前视频块相对于包括当前视频块的编解码单元(CU)的位置(x0,y0)处的样点，该过程计算在缓冲区中的参考位置处开始的对应参考区域。在步骤2706处，该过程调整参考区域和参考位置以确定先前处理的块中的哪些被用于预测。

[1052] 图28是可视数据处理的示例方法的流程图。该流程图的步骤结合在本文档的第4章节中讨论的示例69-75进行讨论。在步骤2802处，该过程对于视频的当前视频块和当前视频块的比特流表示之间的转换，使用视频的分量X来确定与视频的分量c的当前视频块相对应的块矢量的有效性，其中，分量X不同于视频的亮度分量。在步骤2804处，该过程当确定块矢量对于当前视频块有效时，使用块矢量来执行该转换，其中，该转换在帧内块复制(IBC)模式下被执行，其中该帧内块复制(IBC)模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息。

[1053] 本文档的一些实施例现在以基于条款的格式呈现。

[1054] A1. 一种可视媒体处理的方法，包括：

[1055] 对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换，确定存储用于帧内块复制模式下的预测的参考样点的缓冲区，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息；

[1056] 对于在空域上位于当前视频块相对于包括当前视频块的编解码树单元的左上角位置的位置(x0,y0)处并具有块矢量(BVx,BVy)的样点，计算缓冲区中在参考位置(P,Q)处的对应参考，其中，参考位置(P,Q)是使用块矢量(BVx,BVy)和位置(x0,y0)而确定的；以及

[1057] 当确定参考位置(P,Q)位于缓冲区之外时，至少部分基于当前视频块相对于包括当前视频块的编解码树单元的位置来重新计算参考位置。

[1058] A2. 根据条款A1所述的方法，其中，参考位置(P,Q)被确定为P＝x0+BVx并且Q＝y0+BVy。

[1059] A3. 根据条款1所述的方法，其中，重新计算包括：

[1060] 至少部分基于当前视频块是位于相对于编解码树单元的水平方向还是位于相对于编解码树单元的垂直方向来重新计算参考位置(P,Q)。

[1061] A4. 根据条款A1-A3中任一项或多项所述的方法，其中，当前视频块位于相对于编解码树单元的水平方向。

[1062] A5. 根据条款A1-A3中任一项或多项所述的方法，其中，当前视频块位于相对于编解码树单元的垂直方向。

[1063] A6. 根据条款1所述的方法，其中，重新计算包括：

[1064] 至少部分基于当前视频块是否位于距可视媒体数据的边界的预定义距离内来重新计算参考位置(P,Q)。

[1065] A7. 根据条款A1-A2和A6中任一项或多项所述的方法，其中，当前视频块位于距可视媒体数据的边界的预定义距离内。

[1066] A8. 根据条款A1-A2中任一项或多项所述的方法，其中，响应于确定(y0+BVy)位于范围[0,…,N-1]之外，参考位置(P,Q)被指定预定义值，其中N是表示缓冲区的y维度的整数。

[1067] A9. 根据条款A1-A2中任一项或多项所述的方法,其中,响应于确定(x0+BVx)位于范围[0,…,M-1]之外,参考位置(P,Q)被指定预定义值,其中M是表示缓冲区的x维度的整数。

[1068] A10. 根据条款A1-A2中任一项或多项所述的方法,其中,参考位置(P,Q)被指定为((x0+BVx)mod M,y0+BVy),其中“mod”是定义为x mod y= x - y * floor(x/y)的模运算,其中floor(a)是不大于a的最大整数,并且M是表示缓冲区的x维度的整数。

[1069] A11. 根据条款A1-A2中任一项或多项所述的方法,其中,参考位置(P,Q)被指定为(x0+BVx,(y0+BVy)mod N)),其中“mod”是定义为x mody = x - y * floor(x/y)的模运算,其中floor(a)是不大于a的最大整数,并且N是表示缓冲区的y维度的整数。

[1070] A12. 根据条款A10所述的方法,其中,响应于确定((x0+BVx)mod M,y0+BVy)位于缓冲区之外,附加处理被执行。

[1071] A13. 根据条款A12所述的方法,其中,响应于确定(x0+BVx,(y0+BVy)mod N))位于缓冲区之外,附加处理被执行。

[1072] B1. 一种可视媒体处理的方法,包括:

[1073] 对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换,确定存储用于帧内块复制模式下的预测的参考样点的缓冲区,其中,该转换在帧内块复制模式下被执行,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息;

[1074] 对于在空域上位于当前视频块相对于包括当前视频块的图片的左上角位置的位置(x,y)处并具有块矢量(BVx,BVy)的样点,至少部分基于满足与以下中的至少一个相关联的一个或多个条件来将块矢量(BVx,BVy)指定为有效:当前视频块的位置(x,y)、当前视频块的大小、图片的大小、包括当前视频块的编解码树单元的大小、或缓冲区的大小;

[1075] 执行检查以确定块矢量(BVx,BVy)有效;以及

[1076] 当识别出块矢量(BVx,BVy)有效时,计算缓冲区中在参考位置(P,Q)处的对应参考,其中,参考位置(P,Q)是使用块矢量(BVx,BVy)、位置(x,y)以及缓冲区的大小而确定的。

[1077] B2. 根据条款B1所述的方法,其中,参考位置(P,Q)被确定为((x+BVx)% Wbuf,(x+BVy)% Hbuf),其中,Wbuf×Hbuf表示缓冲区的大小,其中,“%”表示模运算,并且“x%y”在x<0时被定义为x - y * floor(x/y),其中floor(a)是不大于a的最大整数。

[1078] C1. 一种可视媒体处理的方法,包括:

[1079] 对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换,确定当前视频块的块矢量(BVx,BVy)或块矢量差(BVDx,BVDy),其中,该转换在帧内块复制模式下被执行,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息;以及

[1080] 将块矢量(BVx,BVy)的至少一个分量或块矢量差(BVDx,BVDy)的至少一个分量归一化为位于一范围内。

[1081] C2. 根据条款C1所述的方法,其中,归一化包括:

[1082] 基于缓冲区的大小来将块矢量(BVx,BVy)的至少一个分量或块矢量差(BVDx,BVDy)的至少一个分量归一化为位于该范围内,其中,缓冲区存储用于帧内块复制模式下的预测的参考样点。

[1083] C3. 根据条款C2所述的方法,其中,分量BVx被归一化为(BVx modM),其中M是表示缓冲区的x维度的整数,并且“mod”是定义为x mod y＝x-y*floor(x/y)的模运算,其中floor(a)是不大于a的最大整数。

[1084] C4.根据条款C2所述的方法,其中,分量BVx被归一化为((BVx+x0)mod M)-V,其中M是表示缓冲区的x维度的整数,“mod”是定义为x mody＝x-y*floor(x/y)的模运算,其中floor(a)是不大于a的最大整数,并且V是预定义值。

[1085] C5.根据条款C4所述的方法,其中,V是64。

[1086] C6.根据条款C4所述的方法,其中,V是M/2。

[1087] C7.根据条款C4所述的方法,其中,V是x0。

[1088] C8.根据条款C2所述的方法,其中,分量BVy被归一化为(BVy modN),其中N是表示缓冲区的y维度的整数,并且“mod”是定义为x mod y＝x-y*floor(x/y)的模运算,其中floor(a)是不大于a的最大整数。

[1089] C9.根据条款C2所述的方法,其中,分量BVy被归一化为((BVy+y0)mod N)-V,其中N是表示缓冲区的y维度的整数,“mod”是定义为x mody＝x-y*floor(x/y)的模运算,其中floor(a)是不大于a的最大整数,并且V是预定义值。

[1090] C10.根据条款C9所述的方法,其中,V是64。

[1091] C11.根据条款C9所述的方法,其中,V是N/2。

[1092] C12.根据条款C9所述的方法,其中,V是y0。

[1093] C13.根据条款C1所述的方法,其中,分量BVx和BVy被归一化为位于不同的范围内。

[1094] C14.根据条款C2所述的方法,其中,分量BVDx被归一化为(BVDxmod M),其中M是表示缓冲区的x维度的整数,并且“mod”是定义为x mody＝x-y*floor(x/y)的模运算,其中floor(a)是不大于a的最大整数。

[1095] C15.根据条款C2所述的方法,其中,分量BVDx被归一化为((BVDx+x0)mod M)-V,其中M是表示缓冲区的x维度的整数,“mod”是定义为x mod y＝x-y*floor(x/y)的模运算,其中floor(a)是不大于a的最大整数,并且V是预定义值。

[1096] C16.根据条款C15所述的方法,其中,V是64。

[1097] C17.根据条款C15所述的方法,其中,V是M/2。

[1098] C18.根据条款C15所述的方法,其中,V是x0。

[1099] C19.根据条款C2所述的方法,其中,分量BVDy被归一化为(BVDymod N),其中N是表示缓冲区的y维度的整数,并且“mod”是模运算。

[1100] C20.根据条款C2所述的方法,其中,分量BVDy被归一化为((BVDy+y0)mod N)-V,其中N是表示缓冲区的y维度的整数,“mod”是模运算,并且V是预定义值。

[1101] C21.根据条款C20所述的方法,其中,V是64。

[1102] C22.根据条款C20所述的方法,其中,V是N/2。

[1103] C23.根据条款C20所述的方法,其中,V是y0。

[1104] C24.根据条款C1所述的方法,其中,分量BVDx和BVDy被归一化为位于不同的范围内。

[1105] D1.一种可视媒体处理的方法,包括:

[1106] 对于当前视频块和当前视频块的比特流表示之间的转换,确定用于存储用于帧内

块复制模式下的预测的重构样点的缓冲区，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息；以及

[1107] 根据顺序来更新存储在缓冲区中的重构样点。

[1108] D2.根据条款D1所述的方法，其中，缓冲区根据先后顺序而更新。

[1109] D3.根据条款D1所述的方法，其中，缓冲区根据重构块的顺序而更新。

[1110] D4.根据条款D1所述的方法，还包括：

[1111] 当确定缓冲区已满时，用最近重构的样点替换存储在缓冲区中的重构样点。

[1112] D5.根据条款D1所述的方法，其中，存储在缓冲区中的重构样点以先进先出的顺序被替换。

[1113] D6.根据条款D1所述的方法，其中，存储在缓冲区中的重构样点的最老集合被替换。

[1114] D7.根据条款D1所述的方法，其中，存储在缓冲区中的重构样点被指定优先级值，并且其中，存储在缓冲区中的重构样点根据优先级值被替换。

[1115] D8.根据条款D1所述的方法，其中，存储在缓冲区中的重构样点的子集被标记以用于未来的替换，并且其中，不包括在该子集中的样点最初被替换。

[1116] D9.根据条款D7所述的方法，其中，被包括在比特流表示中的标志指示满足一个或多个条件的优先级值。

[1117] D10.根据条款D7所述的方法，其中，优先级值是基于当前视频块的特性而指定的。

[1118] D11. 根据条款D9所述的方法，其中，一个或多个条件与使用调色板模式和/或帧内块编解码(IBC)模式和/或变换跳过模式编解码的重构样点的百分比相关。

[1119] D12. 根据条款D11所述的方法，其中，响应于确定使用调色板模式和/或帧内块编解码(IBC)模式和/或变换跳过模式编解码的重构样点的百分比超过阈值，将当前视频块中的所有样点指定为高优先级。

[1120] D13. 根据条款D12所述的方法，其中，阈值基于当前视频块的尺寸和/或当前视频块的颜色分量和/或包括当前视频块的编解码树单元(CTU)的尺寸。

[1121] D14. 根据条款D13所述的方法，其中，阈值作为字段被包括在比特流表示中。

[1122] D15. 根据条款D14所述的方法，其中，字段被包括在序列参数集(SPS)、图片参数集(PPS)、序列头、条带头、片组、片级别或视频区域中。

[1123] D16. 根据条款D4所述的方法，其中，确定缓冲区中的可用样点的数量等于或大于阈值指示缓冲区已满。

[1124] D17. 根据条款D16所述的方法，其中，阈值是64×64×3个亮度样点。

[1125] E1. 一种可视媒体处理的方法，包括：

[1126] 执行当前视频块和当前视频块的比特流表示之间的转换，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，用于预测计算的第一精确度低于用于重构计算的第二精确度。

[1127] E2. 根据条款E1所述的方法，其中，预测计算包括使用clip{{p+[1<<(b-1)]}>>b, 0, (1<<bitdepth)-1}<<b从重构样点值确定预测样点值，其中p是重构样点值，b是预定义

的比特移位值,bitdepth是预测样点精确度,并且“clip”是定义为clip(x, y, z)＝(x<y)?y:(x>z?z:x)的裁剪操作。

[1128] E3. 根据条款E1所述的方法,其中,预测计算包括使用clip{{p+[1<<(b-1)-1]}>>b, 0, (1<<bitdepth)-1}<<b从重构样点值确定预测样点值,其中p是重构样点值,b是预定义的比特移位值,bitdepth是预测样点精确度,并且“clip”是定义为clip(x, y, z)＝(x<y)?y:(x>z?z:x)的裁剪操作。

[1129] E4. 根据条款E1 所述的方法, 其中, 预测计算包括使用((p>>b)+(1<<(bitdepth-1)))<<b从重构样点值确定预测样点值,其中p是重构样点值,b是预定义的比特移位值,并且bitdepth是预测样点精确度。

[1130] E5. 根据条款E1 所述的方法, 其中, 预测计算包括使用(clip((p>>b),0,(1<<(bitdepth-b)))+(1<<(bitdepth-1)))<<b从重构样点值确定预测样点值,其中p是重构样点值,b是预定义的比特移位值,bitdepth是预测样点精确度,并且“clip”是定义为clip(x, y, z)＝(x<y)?y:(x>z?z:x)的裁剪操作。

[1131] E6. 根据条款E1所述的方法,其中,预测计算包括基于是否应用了环路整形(ILR)步骤,使用裁剪操作从重构样点值确定预测样点值。

[1132] E7. 根据条款E1-E5中任一项或多项所述的方法,其中,b是重构样点和当前视频块的比特深度的差。

[1133] E8. 根据条款E1所述的方法,其中,第一精确度和/或第二精确度和/或第一精确度和第二精确度之间的差作为字段在比特流表示中被信令通知。

[1134] E9. 根据条款E1所述的方法,其中,预测计算包括从重构样点值确定预测样点值,其中,预测样点值的第一部分具有第一精确度,并且预测样点值的第二部分具有第二精确度。

[1135] E10. 根据条款E9所述的方法,其中,当前视频块位于包括具有不同精确度的样点的编解码树单元之内。

[1136] E11. 根据条款E10所述的方法,其中,具有不同精确度的样点被包括在针对当前视频块允许的参考区域中。

[1137] E12. 根据条款E9所述的方法,其中,与包括另一视频块的编解码树单元相关联的第一参考区域使用第一精确度,并且与包括当前视频块的编解码树单元相关联的第二参考区域使用第二精确度。

[1138] E13. 根据条款E12所述的方法,其中,第一参考区域对应于第一颜色分量,并且第二参考区域对应于第二颜色分量。

[1139] F1. 一种可视媒体处理的方法,包括:

[1140] 使用帧内块复制模式来执行当前视频块和当前视频块的比特流表示之间的转换,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息,其中,在该转换期间,尺寸为nM×nM的参考区域被使用,其中n和M为整数,并且其中,当前视频块位于编解码树单元中,并且其中,参考区域包括来自对应于当前视频块的编解码树单元行中的n×n个最近的可用编解码树单元的样点。

[1141] F2. 根据条款F1所述的方法,其中,参考区域的尺寸为128×128个样点,编解码树单元尺寸为64×64,并且缓冲区包括在包括当前视频块的相同编解码树单元行中的四个最

近的可用编解码树单元。

[1142] F3. 根据条款F1所述的方法,其中,参考区域的尺寸为128×128个样点,编解码树单元尺寸为32×32,并且缓冲区包括在包括当前视频块的相同编解码树单元行中的十六个最近的可用编解码树单元。

[1143] G1. 一种可视媒体处理的方法,包括:

[1144] 使用帧内块复制模式来执行当前视频块和当前视频块的比特流表示之间的转换,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息,其中,在该转换期间,尺寸为nM×pM的参考区域被使用,其中n、p和M为整数,并且其中,当前视频块位于编解码树单元中,并且其中,参考区域包括来自对应于当前视频块的编解码树单元行中的n×p-1个最近的可用编解码树单元的样点。

[1145] G2. 根据条款G1所述的方法,其中,参考区域的尺寸为128×128个样点或256×64个样点,编解码树单元尺寸为64×64,并且缓冲区包括在包括当前视频块的相同编解码树单元行中的三个最近的可用编解码树单元。

[1146] G3. 根据条款G1所述的方法,其中,参考区域的尺寸为128×128个样点或512×32个样点,编解码树单元尺寸为32×32,并且缓冲区包括在包括当前视频块的相同编解码树单元行中的十五个最近的可用编解码树单元。

[1147] G4. 根据条款G1所述的方法,其中,编解码树单元的尺寸为M×M。

[1148] G5. 根据条款G1至G4中任一项所述的方法,其中,缓冲区之外的样点在该转换期间不被允许使用。

[1149] H1. 一种可视媒体处理的方法,包括:

[1150] 使用帧内块复制模式来执行视频区域的虚拟管道数据单元(VPDU)的当前视频块和当前视频块的比特流表示之间的转换,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息,其中,在该转换期间,尺寸为nM×nM的参考区域被使用,VPDU的尺寸为kM×kM,其中k、n和M为整数,并且其中,当前视频块位于编解码树单元中,并且其中,参考区域包括来自对应于当前视频块的编解码树单元行中的n×n-k个最近的可用编解码树单元的样点。

[1151] H2. 根据条款H1所述的方法,其中,参考区域的尺寸为128×128个样点,编解码树单元尺寸为64×64,VPDU的尺寸为64×64,并且缓冲区包括在包括当前视频块的相同编解码树单元行中的三个最近的可用编解码树单元。

[1152] H3. 根据条款H1所述的方法,其中,参考区域的尺寸为128×128个样点,编解码树单元尺寸为32×32,VPDU的尺寸为64×64,并且缓冲区包括在包括当前视频块的相同编解码树单元行中的十二个最近的可用编解码树单元。

[1153] I1. 一种可视媒体处理的方法,包括:

[1154] 对于可视媒体数据的尺寸为w×h的当前视频块和当前视频块的比特流表示之间的转换,确定存储用于帧内块复制模式下的预测的参考样点的缓冲区,其中,该转换在帧内块复制模式下被执行,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息;

[1155] 对于在空域上位于当前视频块相对于包括当前视频块的尺寸为M×M的编解码树单元(CTU)的左上角位置的位置(x0,y0)处并具有块矢量(BVx,BVy)的样点,计算在缓冲区

中的参考位置(P,Q)处开始的对应参考区域,其中,参考位置(P,Q)是使用块矢量(BVx,BVy)和/或位置(x0,y0)而确定的;以及

[1156] 将一个或多个基于规则的约束应用于参考区域和/或参考位置(P,Q),以限制参考区域与视频区域的重叠。

[1157] I2.根据条款I1所述的方法,其中,当CTU的尺寸为128×128、(x0,y0)被表示为(m×64,n×64)时,参考区域被限制与左上角被表示为((m-2)× 64,n×64)的尺寸为64×64的视频区域重叠,其中m和n为整数。

[1158] I3.根据条款I1所述的方法,其中,当CTU的尺寸为128×128、(x0,y0)被表示为(m×64,n×64)时,参考区域被限制与左上角被表示为(x0-128,y0)的尺寸为w×h的视频区域重叠,其中m和n为整数。

[1159] I4.根据条款I1所述的方法,其中,当缓冲区的尺寸为kM×M时,参考区域被限制与左上角被表示为(x0-kM y0)的尺寸为w×h的视频区域重叠。

[1160] I5.根据条款I1所述的方法,其中,当当前视频块的左上角位置被表示为(2m*64,2n*64)时,重构块的左上角位置被表示为((2m-2)*64,2n*64),其中m和n为整数,并且当前视频块的尺寸为64×64。

[1161] I6.根据条款I1所述的方法,其中,当当前视频块的左上角位置被表示为(2m*64,2n*64)时,重构块的左上角位置被表示为((2m-1)*64,2n*64),其中m和n为整数,并且当前视频块的尺寸为64×64。

[1162] I7.根据条款I1所述的方法,其中, 当当前视频块的左上角位置被表示为(2m*64,2n*64)时,重构块的左上角位置被表示为((2m-1)*64,(2n+1)*64),其中m和n为整数,并且当前视频块的尺寸为64×64。

[1163] I8.根据条款I1所述的方法,其中, 当当前视频块的左上角位置被表示为(2m*64,2n*64)时,重构块的左上角位置位于当前视频块的左上角位置处,其中m和n为整数,并且当前视频块的尺寸为64×64。

[1164] I9.根据条款I1所述的方法,其中, 当当前视频块的左上角位置被表示为((2m+1)*64,(2n+1)*64)时,重构块的左上角位置位于当前视频块的左上角位置处,其中m和n为整数,并且当前视频块的尺寸为64×64。

[1165] I10.根据条款I1所述的方法,其中,当当前视频块的左上角位置被表示为((2m+1)*64,2n*64)时,重构块的左上角位置被表示为((2m-1)*64,2n*64),其中m和n为整数,并且当前视频块的尺寸为64×64。

[1166] I11.根据条款I1所述的方法,其中,当当前视频块的左上角位置被表示为((2m+1)*64,2n*64)时,重构块的左上角位置被表示为((2m-1)*64,(2n+1)*64),其中m和n为整数,并且当前视频块的尺寸为64×64。

[1167] I12.根据条款I1所述的方法,其中,当当前视频块的左上角位置被表示为((2m+1)*64,2n*64)时,重构块的左上角位置被表示为(2m*64,2n*64),其中m和n为整数,并且当前视频块的尺寸为64×64。

[1168] I13.根据条款I1所述的方法,其中,当当前视频块的左上角位置被表示为((2m+1)*64,2n*64)时,重构块的左上角位置位于当前视频块的左上角位置处,其中m和n为整数,并且当前视频块的尺寸为64×64。

[1169] I14. 根据条款I1所述的方法,其中,当当前视频块的左上角位置被表示为(2m*64,(2n+1)*64)时,重构块的左上角位置被表示为((2m-1)*64,(2n+1)*64),其中m和n为整数,并且当前视频块的尺寸为64×64。

[1170] I15. 根据条款I1所述的方法,其中,当当前视频块的左上角位置被表示为(2m*64,(2n+1)*64)时,重构块的左上角位置被表示为(2m*64,2n*64),其中m和n为整数,并且当前视频块的尺寸为64×64。

[1171] I16. 根据条款I1所述的方法,其中,当当前视频块的左上角位置被表示为(2m*64,(2n+1)*64)时,重构块的左上角位置被表示为((2m+1)*64,2n*64),其中m和n为整数,并且当前视频块的尺寸为64×64。

[1172] I17. 根据条款I1所述的方法,其中,当当前视频块的左上角位置被表示为(2m*64,(2n+1)*64)时,重构块的左上角位置位于当前视频块的左上角位置处,其中m和n为整数,并且当前视频块的尺寸为64×64。

[1173] J1. 一种可视媒体处理的方法,包括:

[1174] 对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换,确定存储用于帧内块复制模式下的预测的参考样点的缓冲区,其中,该转换在帧内块复制模式下被执行,其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息;

[1175] 对于在空域上位于当前视频块相对于包括当前视频块的编解码单元(CU)的位置(x0,y0)处的样点,计算在缓冲区中的参考位置处开始的对应参考区域;以及

[1176] 调整参考区域和参考位置以确定先前处理的块中的哪些被用于预测。

[1177] J2. 根据条款J1所述的方法,其中,当(y0>>6)&1 == 0时,多达两个先前处理的块用于预测,其中,两个先前处理的块的左上角位置被表示为((x>>6<<6)- 128,y>>6<<6)和((x>>6<<6)- 64,y>>6<<6)。

[1178] J3. 根据条款J1所述的方法,其中,当(y0>>6)&1 == 1时,左上角位置被表示为((x>>6<<6)- 64,y>>6<<6)的一个先前处理的块用于预测。

[1179] K1. 一种视频处理的方法,包括:

[1180] 对于视频的当前视频块和当前视频块的比特流表示之间的转换,使用视频的分量X来确定与视频的分量c的当前视频块相对应的块矢量的有效性,其中,分量X不同于视频的亮度分量;以及

[1181] 当确定块矢量对于当前视频块有效时,使用块矢量来执行该转换,其中,该转换在帧内块复制(IBC)模式下被执行,其中该帧内块复制(IBC)模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息。

[1182] K2. 根据条款K1所述的方法,其中,当前视频块在位置(x,y)处开始,并且其中,该确定包括确定块矢量对于其中isRec(((x+BVx)>>6<<6)+128-(((y+BVy)>>6)&1)*64+(x%64), ((y+BVy)>>6<<6) +(y%64))为真的情况无效,其中如果样点(x,y)通过IBC模式进行重构,则isRec(x, y)为真,并且块矢量被表示为(BVx,BVy))。

[1183] K3. 根据条款K1所述的方法,其中,分量c对应于视频的亮度分量。

[1184] K4. 根据条款K1所述的方法,其中,当前视频块是色度块,并且视频为4:4:4格式。

[1185] K5.根据条款K1所述的方法,其中,当前视频块包括亮度分量和色度分量。

[1186] K6.根据条款K1所述的方法,其中,视频为4:2:0格式,并且其中,当前视频块是在位置(x,y)处开始的色度块,并且其中,该确定包括确定块矢量对于其中isRec(c,((x+BVx)>>5<<5)+64-(((y+BVy)>>5)&1)*32+(x%32),((y+BVy)>>5<<5)+(y%32))为真的情况无效,其中如果样点(x,y)通过IBC模式进行重构,则isRec(x,y)为真。

[1187] K7.根据条款K1所述的方法,其中,该确定至少部分基于视频的分量X的样点的可用性。

[1188] K8.根据条款K7所述的方法,其中,当前视频块在位置(x,y)处开始,并且其中,该确定包括确定块矢量对于其中isRec(c,((x+BVx)>>6<<6)+128-(((y+BVy)>>6)&1)*64+(x%64),((y+BVy)>>6<<6)+(y%64))为真的情况无效,其中如果分量c的样点(x,y)可用并通过IBC模式进行重构,则isRec(c,x,y)为真,并且块矢量被表示为(BVx,BVy)。

[1189] K9.根据条款K8所述的方法,其中,当前视频块是亮度块。

[1190] K10.根据条款K8所述的方法,其中,当前视频块是色度块,并且视频为4:4:4格式。

[1191] K11.根据条款K7所述的方法,其中,该确定包括确定视频的分量X的样点的可用性。

[1192] K12.根据条款K7所述的方法,其中,当前视频块在位置(x,y)处开始,并且其中,该确定包括确定块矢量对于其中isRec(c,x+BVx+Chroma_CTU_size,y)的情况无效,其中如果分量c的样点(x,y)可用并通过IBC模式进行重构,则isRec(c,x,y)为真,并且块矢量被表示为(BVx,BVy))并且Chroma_CTU_size表示色度分量的编解码树单元的尺寸。

[1193] K13.根据条款K12所述的方法,其中,色度分量的编解码树单元的尺寸为64。

[1194] K14.根据条款K1所述的方法,还包括:

[1195] 确定存储用于帧内块复制模式下的预测的参考样点的缓冲区;以及

[1196] 对于在空域上位于当前视频块相对于包括当前视频块的编解码树单元(CTU)的左上角位置的位置(x0,y0)处的样点,计算在缓冲区中的参考位置(P,Q)处开始的对应参考区域。

[1197] K15. 根据条款K14所述的方法,其中,缓冲区存储尺寸为M×M的块。

[1198] K16. 根据条款K14所述的方法,其中,缓冲区存储尺寸为N×M的块,其中M和N不相等。

[1199] K17. 根据条款K14所述的方法,其中,当满足一个或多个条件时,缓冲区被限制使用。

[1200] K18. 根据条款K14所述的方法,其中,当缓冲区与当前视频块在相同的区块/片/片组/条带内时,缓冲区被限制使用。

[1201] L1. 根据条款A1-K18中任一项所述的方法,其中,该转换包括从当前视频块生成比特流表示。

[1202] L2. 根据条款A1-K18中任一项所述的方法,其中,该转换包括从比特流表示生成当前视频块的像素值。

[1203] L3. 一种视频编码器装置,包括被配置为实施根据条款A1-K18中任一项或多项所述的方法的处理器。

[1204] L4. 一种视频解码器装置,包括被配置为实施根据条款A1-K18中任一项或多项所述的方法的处理器。

[1205] L5．一种其上存储有代码的计算机可读介质，该代码体现用于实施根据条款A1-K18中任一项或多项所述的方法的处理器可执行指令。

[1206] 在本文档中，术语“视频处理”可以指视频编码、视频解码、视频压缩或视频解压缩。例如，视频压缩算法可以在从视频的像素表示到对应的比特流表示的转换期间被应用，反之亦然。当前视频块的比特流表示可以例如对应于比特流内并置的或分散在不同地方的比特，如通过语法定义的。例如，宏块可以根据变换和编解码误差残差值而编码，并且也可以使用比特流中的头和其它字段中的比特进行编码。

[1207] 从前面可以理解，为了说明的目的，本文已经描述了当前公开的技术的特定实施例，但是在不脱离本发明的范围的情况下，可以进行各种修改。因此，当前公开的技术不受除了所附权利要求之外的限制。

[1208] 本专利文档中描述的主题和功能操作的实施方式可以在各种系统、数字电子电路中被实施，或者在计算机软件、固件或硬件(包括本说明书中公开的结构及其结构等同物)中被实施，或者在它们中的一个或多个的组合中被实施。本说明书中描述的主题的实施方式可以被实施为一个或多个计算机程序产品，即编码在有形和非暂时性计算机可读介质上的计算机程序指令的一个或多个模块，该计算机程序指令用于由数据处理装置执行或控制数据处理装置的操作。计算机可读介质可以是机器可读存储设备、机器可读存储基板、存储器设备、影响机器可读传播信号的物质的组合、或它们中的一个或多个的组合。术语“数据处理单元”或“数据处理装置”包含用于处理数据的所有装置、设备和机器，包括例如可编程处理器、计算机、或多个处理器或计算机。除了硬件之外，装置还可以包括为所讨论的计算机程序创建执行环境的代码，例如，构成处理器固件、协议栈、数据库管理系统、操作系统、或它们中的一个或多个的组合的代码。

[1209] 计算机程序(也已知为程序、软件、软件应用、脚本或代码)可以以任何形式的编程语言(包括编译或解释语言)编写，并且其可以以任何形式部署，包括作为独立程序或作为适合在计算环境中使用的模块、组件、子例程或其他单元。计算机程序不一定对应于文件系统中的文件。程序可以存储在保存其他程序或数据(例如，存储在标记语言文档中的一个或多个脚本)的文件的一部分中，存储在专用于所讨论的程序的单个文件中，或存储在多个协调文件中(例如，存储一个或多个模块、子程序或代码部分的文件)。计算机程序可以被部署以在一个计算机上或在位于一个站点上或跨多个站点分布并通过通信网络互连的多个计算机上执行。

[1210] 本说明书中描述的过程和逻辑流程可以由执行一个或多个计算机程序的一个或多个可编程处理器执行，以通过对输入数据进行操作并生成输出来执行功能。过程和逻辑流程也可以由专用逻辑电路执行，并且装置也可以被实施为专用逻辑电路，例如，FPGA(Field Programmable Gate Array，现场可编程门阵列)或ASIC(Application Specific Integrated Circuit，专用集成电路)。

[1211] 适合于执行计算机程序的处理器包括例如通用和专用微处理器、以及任何类型的数字计算机的任何一个或多个处理器。通常，处理器将从只读存储器或随机存取存储器或两者接收指令和数据。计算机的基本元件是用于执行指令的处理器和用于存储指令和数据的一个或多个存储器设备。通常，计算机还将包括用于存储数据的一个或多个大容量存储设备(例如，磁盘、磁光盘或光盘)，或可操作地耦合以从该一个或多个大容量存储设备接收

数据或向该一个或多个大容量存储设备传递数据、或者从其接收数据并向其传递数据。然而，计算机不需要这样的设备。适用于存储计算机程序指令和数据的计算机可读介质包括所有形式的非易失性存储器、介质和存储器设备，包括例如半导体存储器设备，例如EPROM、EEPROM和闪存设备。处理器和存储器可以由专用逻辑电路补充或并入专用逻辑电路中。

[1212] 本说明书以及附图旨在被认为仅是示例性的，其中示例性意味着示例。如本文所使用的，“或”的使用旨在包括“和/或”，除非上下文另外清楚地指示。

[1213] 虽然本专利文档包含许多细节，但这些细节不应被解释为对任何发明或可能要求保护的范围的限制，而是作为特定于特定发明的特定实施例的特征的描述。在本专利文档中在单独的实施例的上下文中描述的某些特征也可以在单个实施例中组合实施。相反，在单个实施例的上下文中描述的各种特征也可以在多个实施例中分开实施或以任何合适的子组合实施。此外，尽管特征可以在上面描述为以某些组合起作用并且甚至最初如此要求保护，但是在一些情况下可以从组合排除来自所要求保护的组合的一个或多个特征，并且所要求保护的组合可以针对子组合或子组合的变化。

[1214] 类似地，虽然在附图中以特定顺序描绘了操作，但是这不应该被理解为需要以所示的特定顺序或以先后顺序执行这样的操作或者执行所有示出的操作以实现期望的结果。此外，在本专利文档中描述的实施例中的各种系统组件的分离不应被理解为在所有实施例中都需要这样的分离。

[1215] 仅描述了一些实施方式和示例，并且可以基于本专利文档中描述和示出的内容来进行其他实施方式、增强和变化。

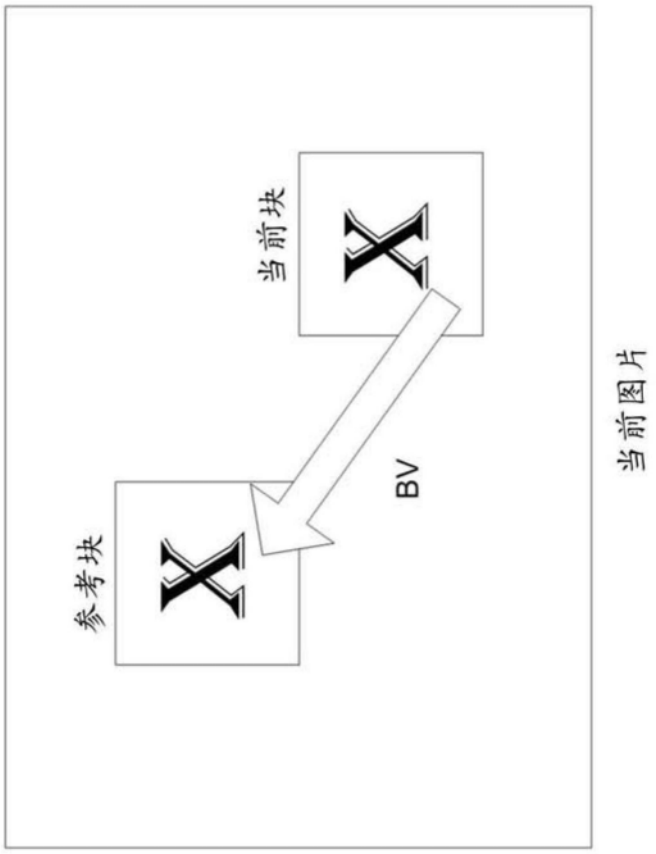


图1

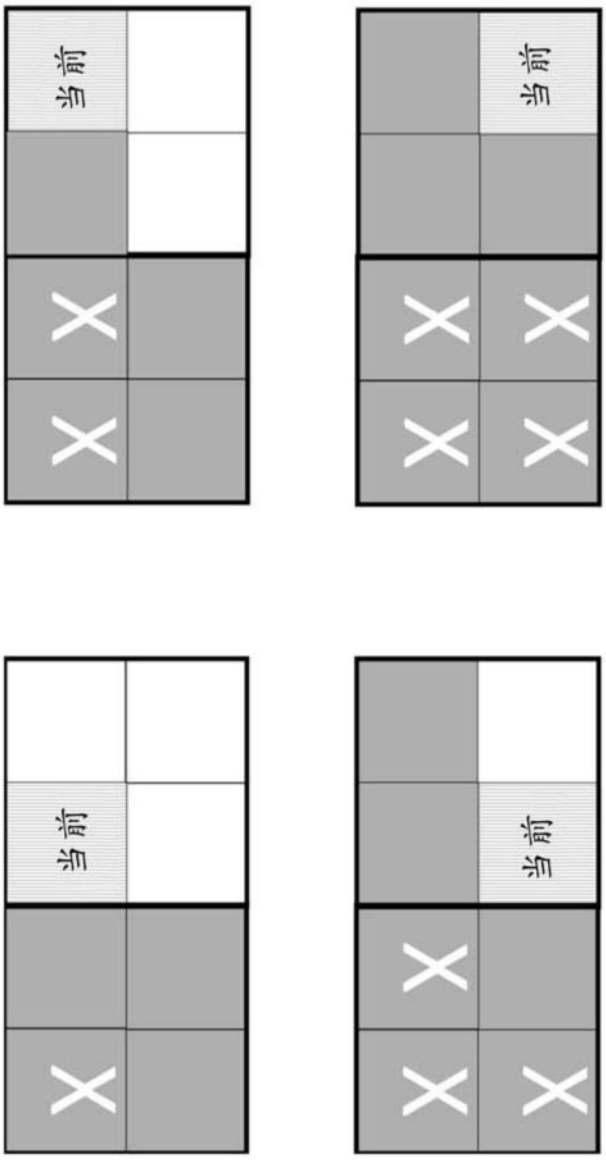


图2

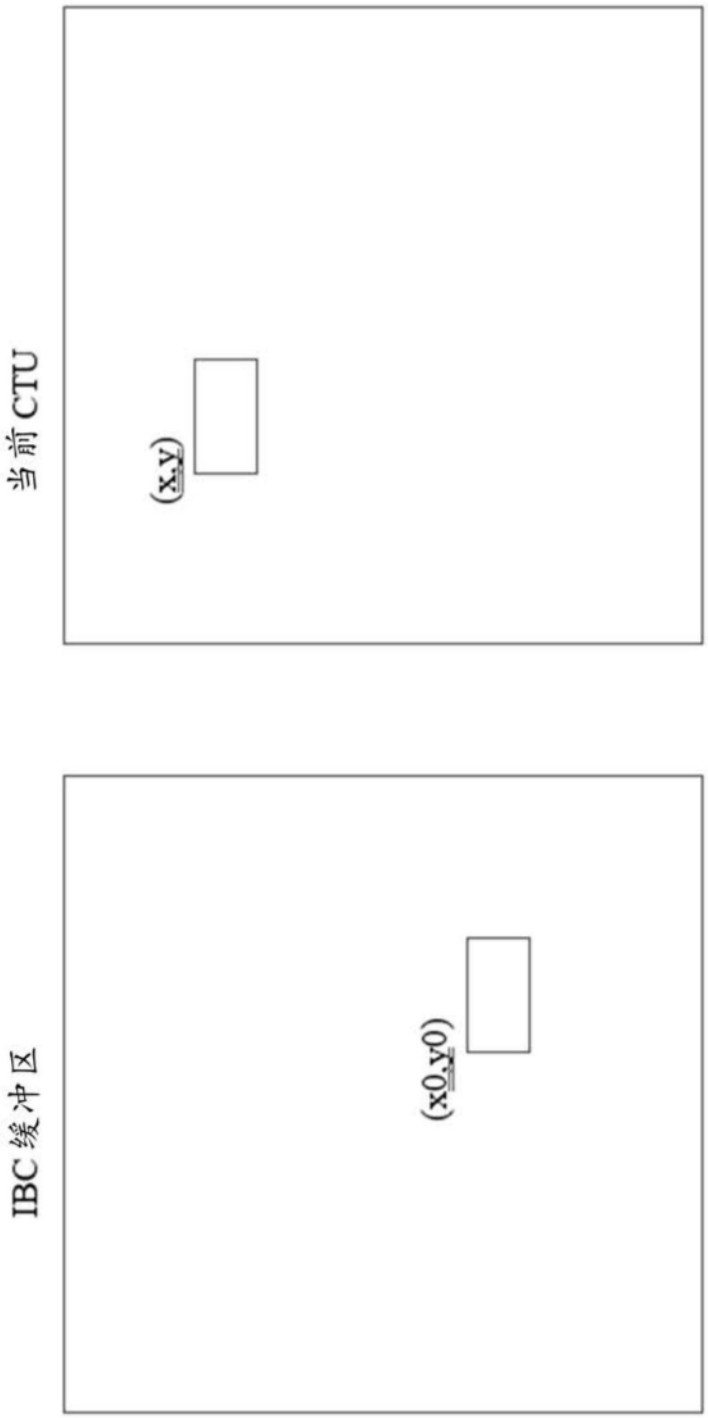


图3

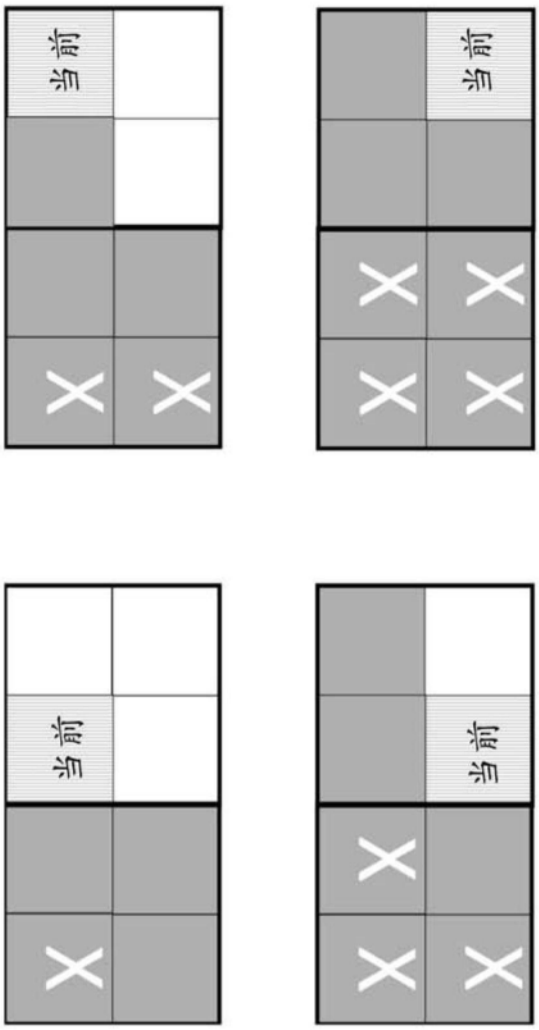


图4

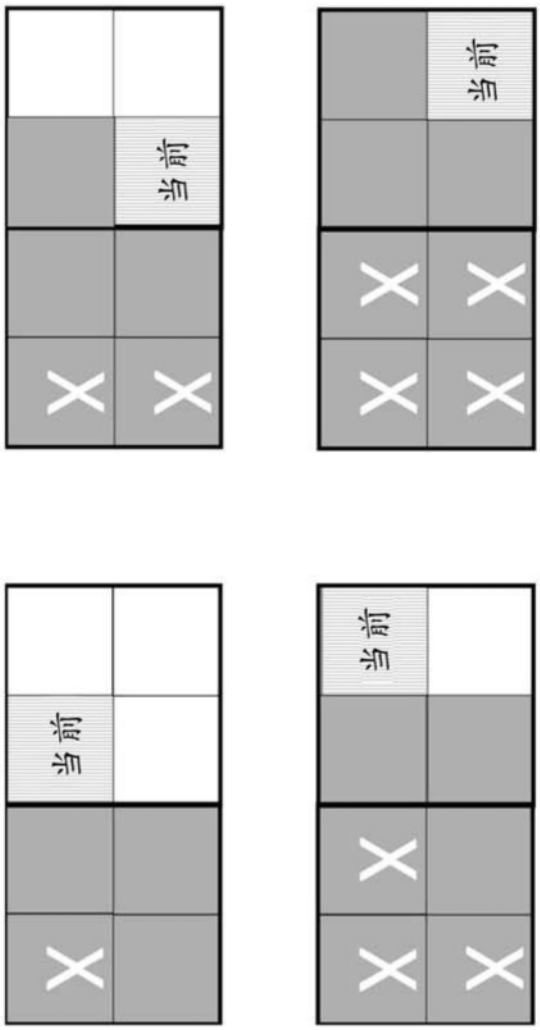


图5


600
602
对于当前视频块和当前视频块的比特流表示之间的转换，
确定缓冲区的尺寸以存储使用帧内块复制编解码模式的当前视频块的参考样点
604
使用存储在缓冲区中的参考样点来执行该转换

图6

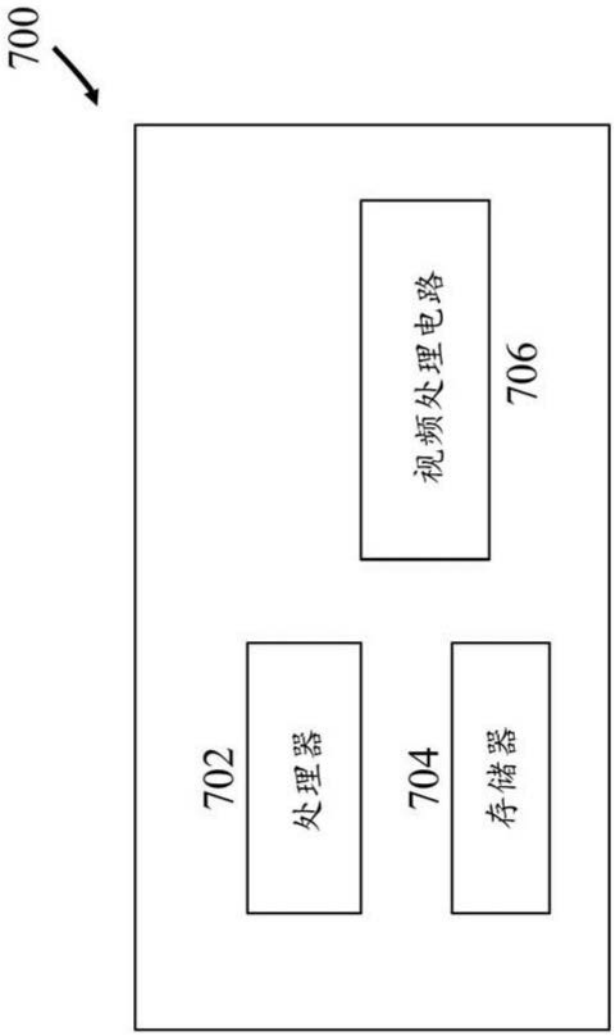


图7

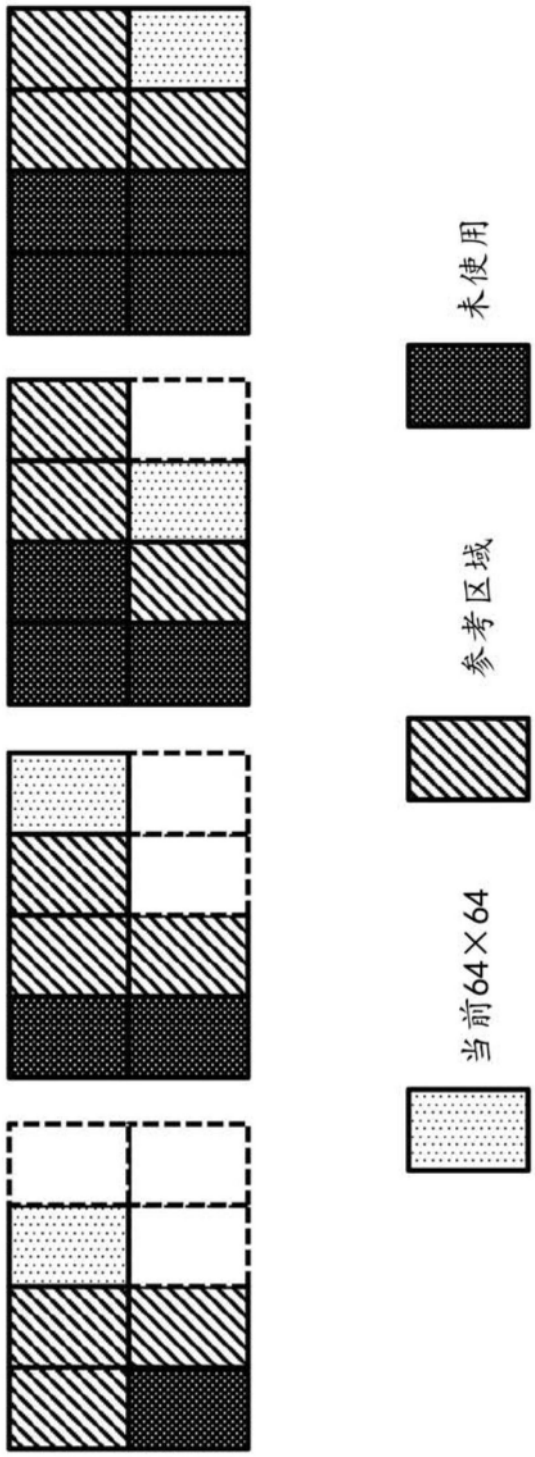


图8

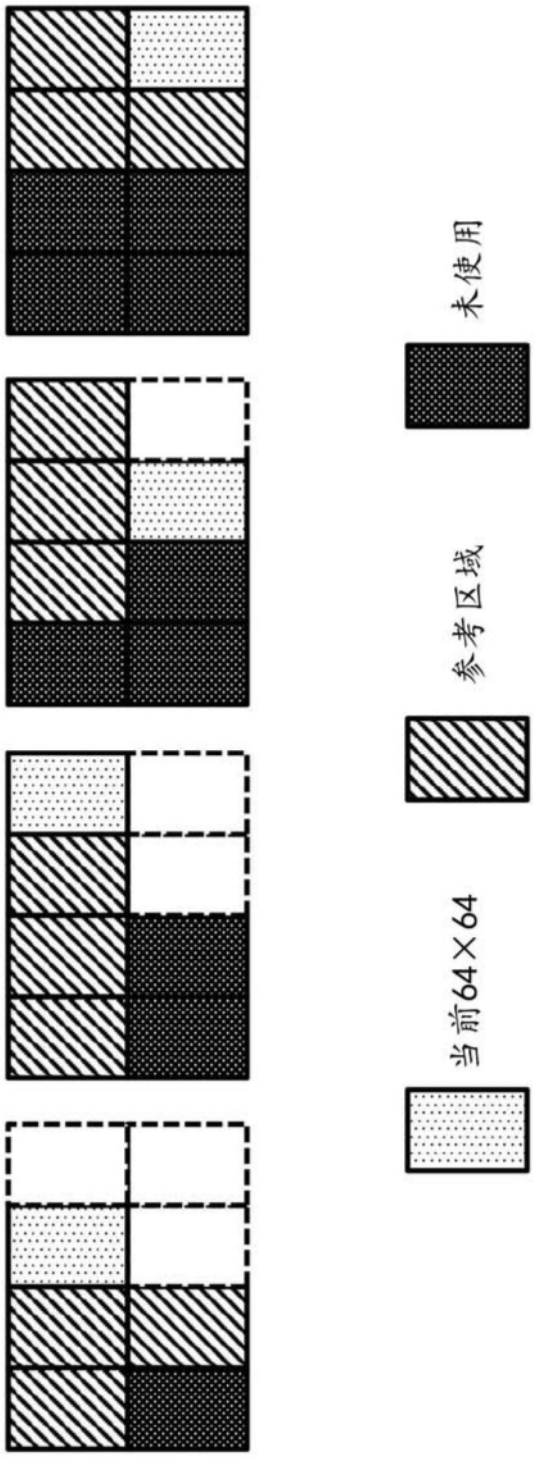


图9

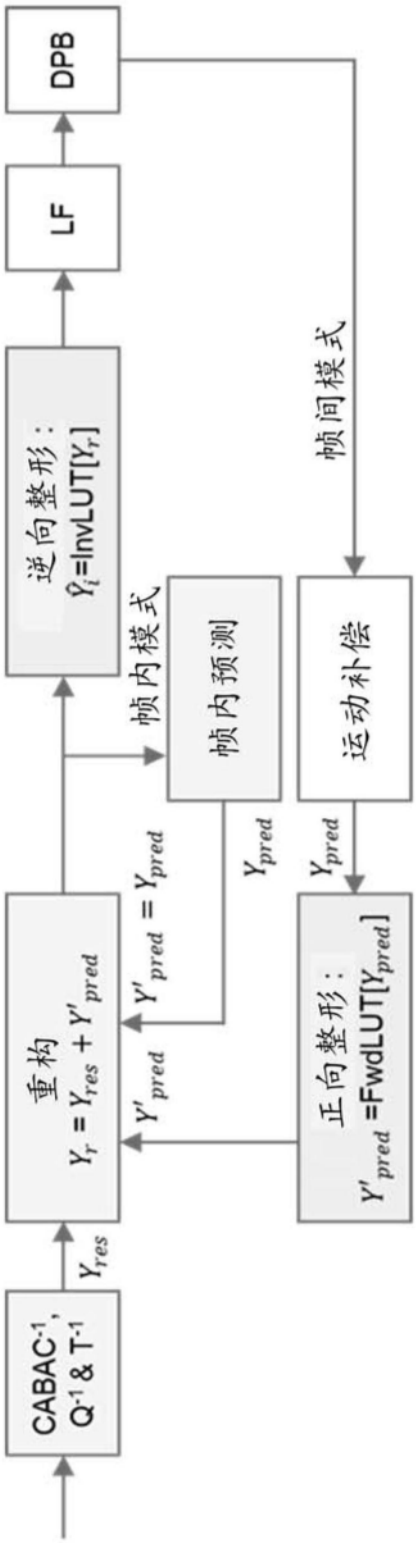

DPB
LF
逆向整形：
Y_I=InvLUT[Y_r]
帧间模式
帧内模式
帧内预测
运动补偿
重构
$Y_r = Y_{res} + Y'_{pred}$
$Y'_{pred} = Y_{pred}$
Y_{pred}
Y_{pred}
Y'_{pred}
正向整形：
Y'_{pred} =FwdLUT[Y_{pred}]
Y_{res}
CABAC^{-1},
Q^{-1} & T^{-1}

图10

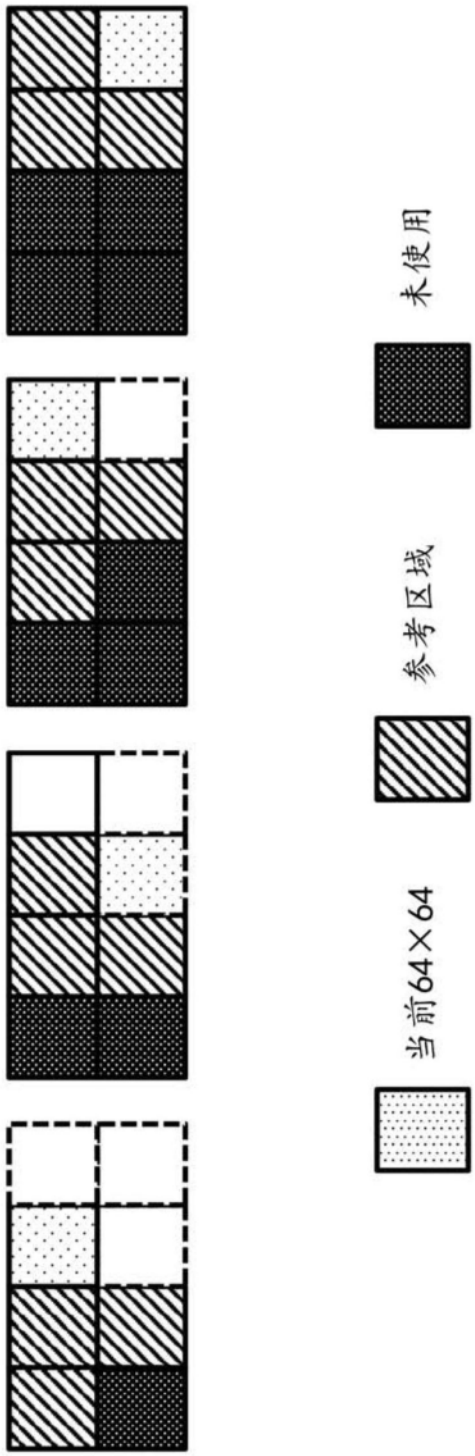


图11

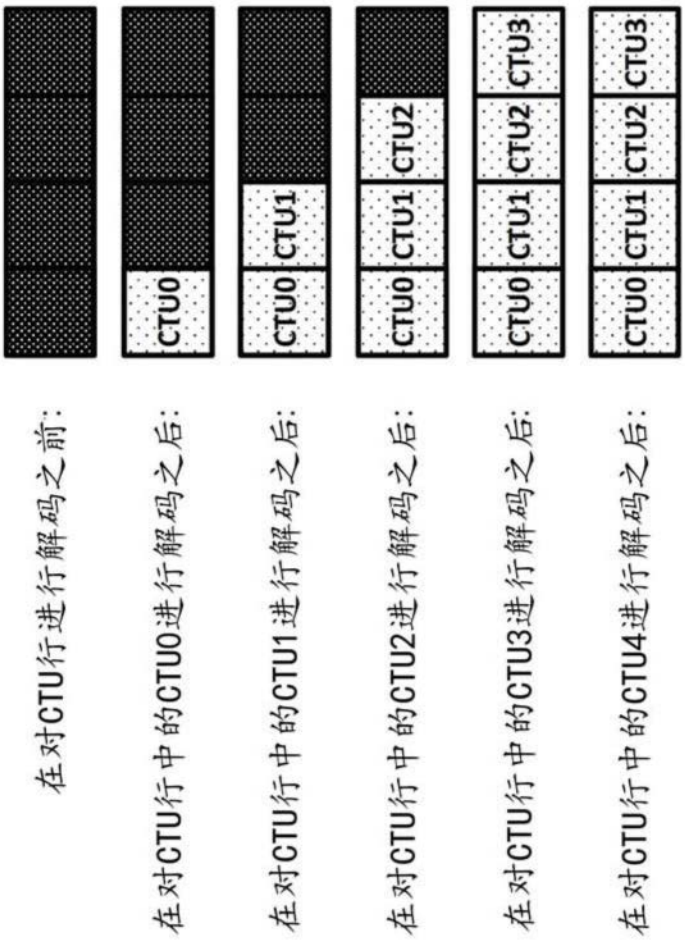

在对CTU行进行解码之前:
CTU0
在对CTU行中的CTU0进行解码之后:
CTU0
CTU1
在对CTU行中的CTU1进行解码之后:
CTU0
CTU1
CTU2
在对CTU行中的CTU2进行解码之后:
CTU0
CTU1
CTU2
CTU3
在对CTU行中的CTU3进行解码之后:
CTU0
CTU1
CTU2
CTU3
在对CTU行中的CTU4进行解码之后:

图12

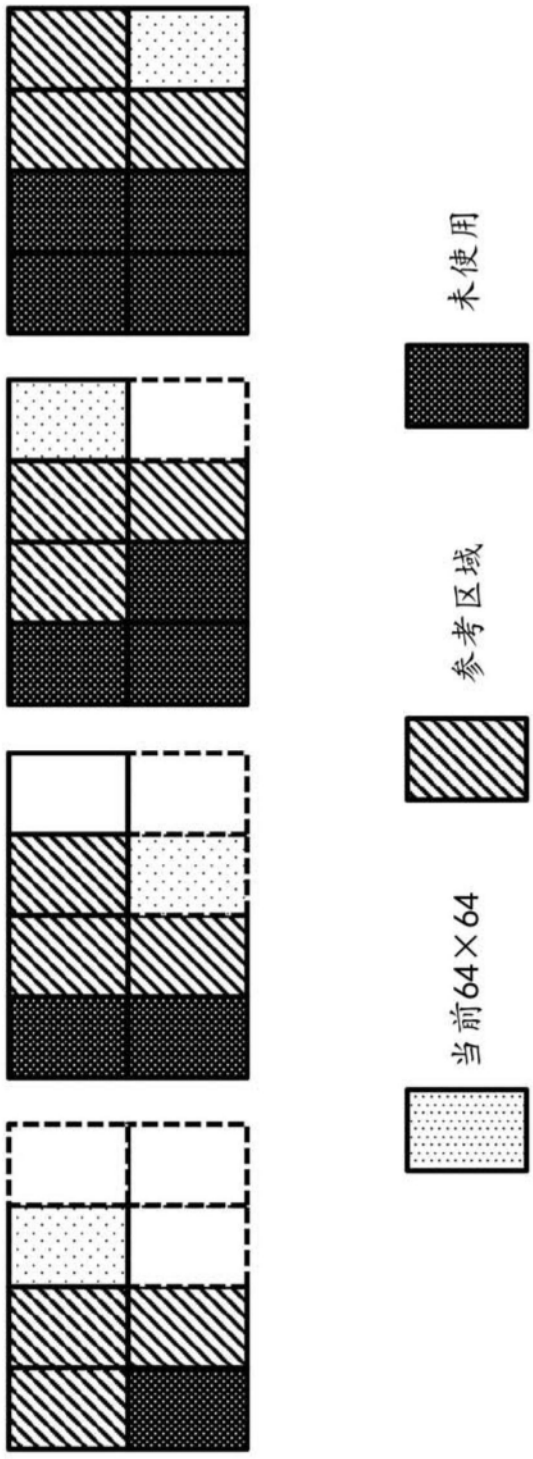


图13

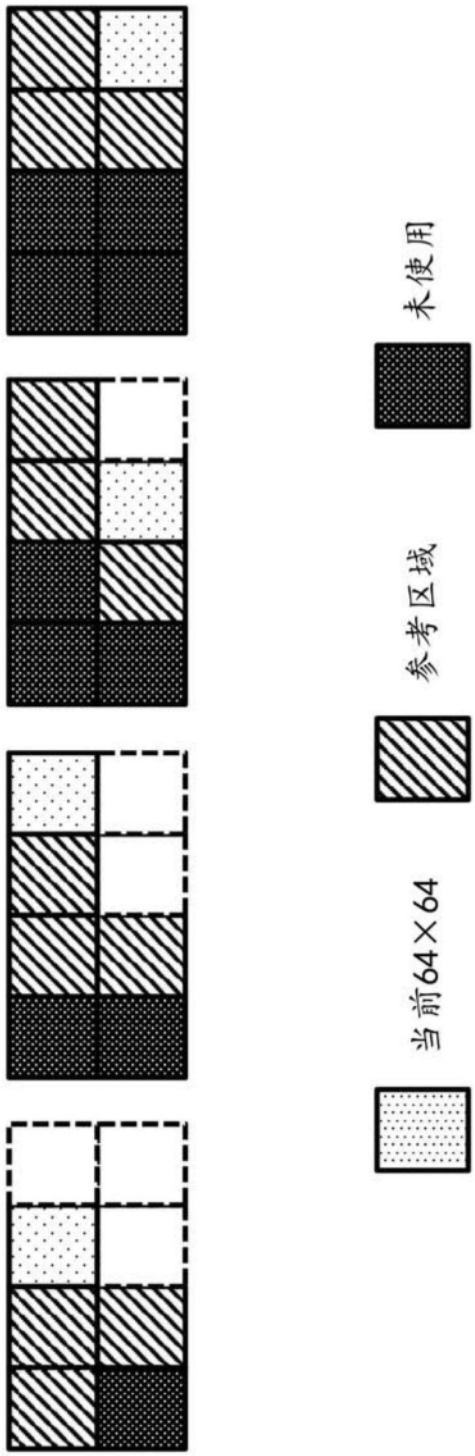


图14

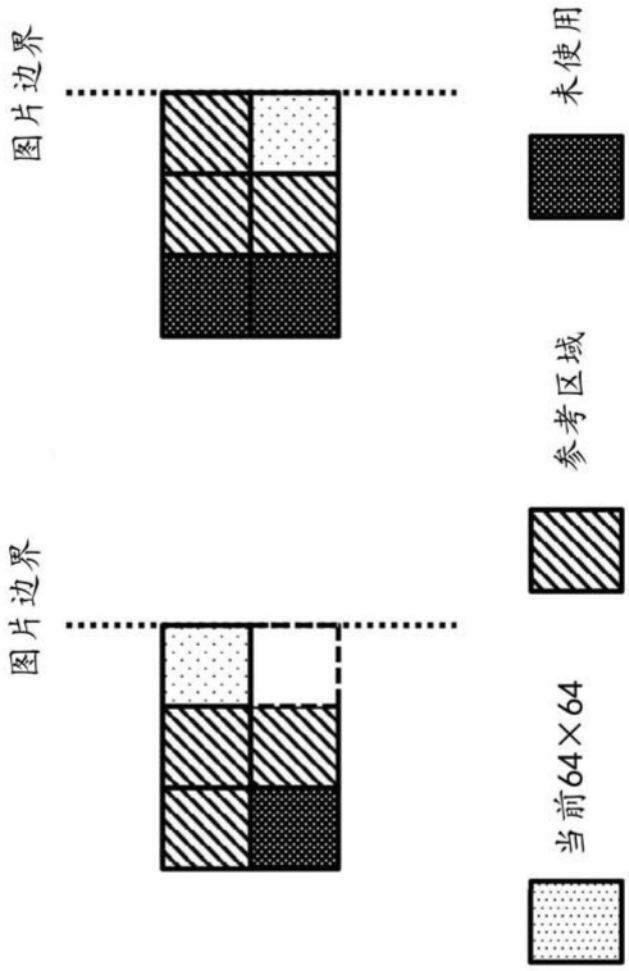


图15

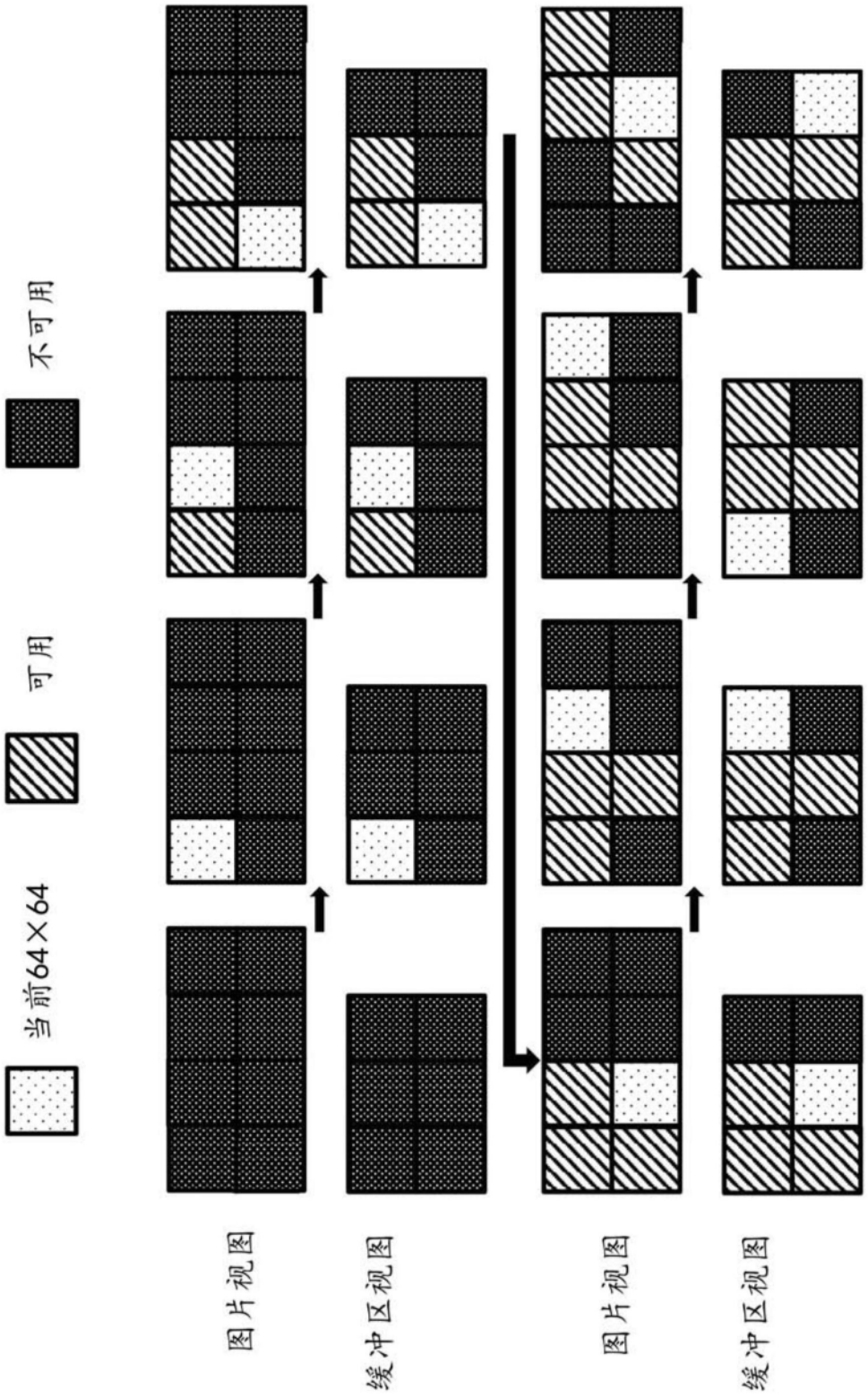


图16

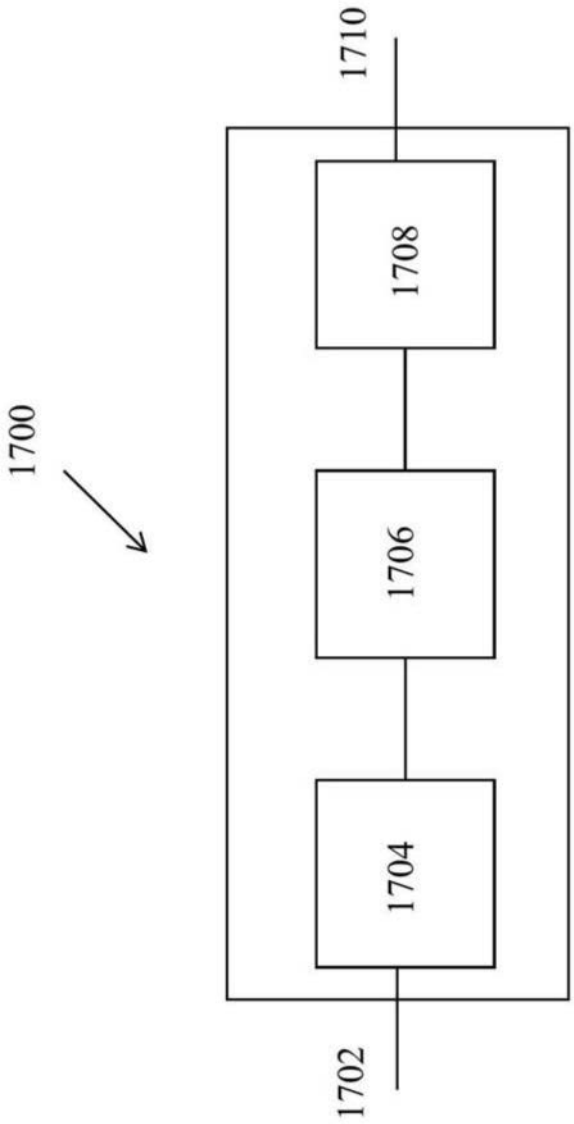


图17

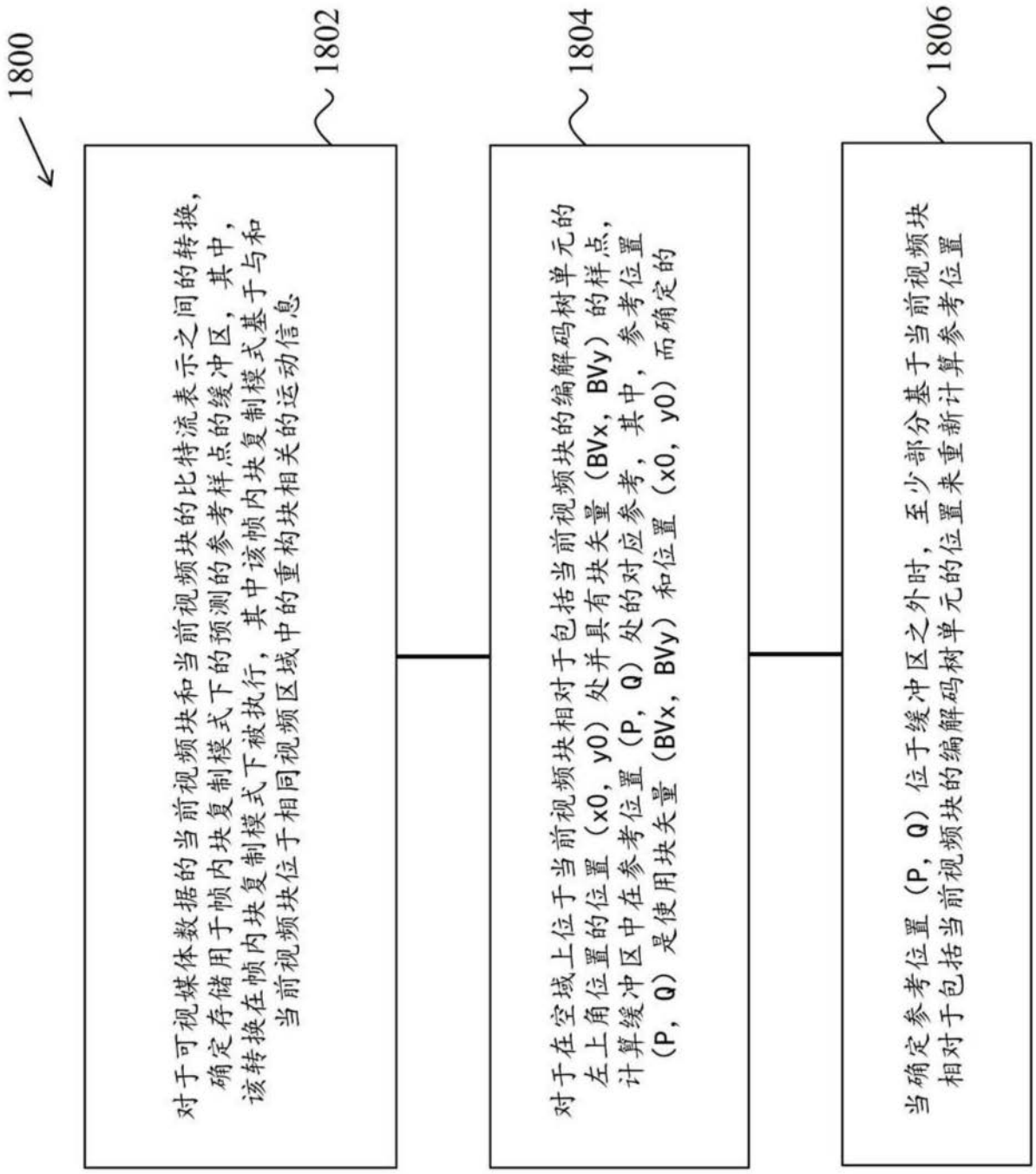

1800
1802
对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换，确定存储用于帧内块复制模式下的预测的参考样点的缓冲区，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息
1804
对于在空域上位于当前视频块相对于包括当前视频块的编解码树单元的左上角位置（x0，y0）处并具有块矢量（BVx，BVy）的样点，计算缓冲区中在参考位置（P，Q）处的对应参考，其中，参考位置（P，Q）是使用块矢量（BVx，BVy）和位置（x0，y0）而确定的
1806
当确定参考位置（P，Q）位于缓冲区之外时，至少部分基于当前视频块相对于包括当前视频块的编解码树单元的位置来重新计算参考位置

图18

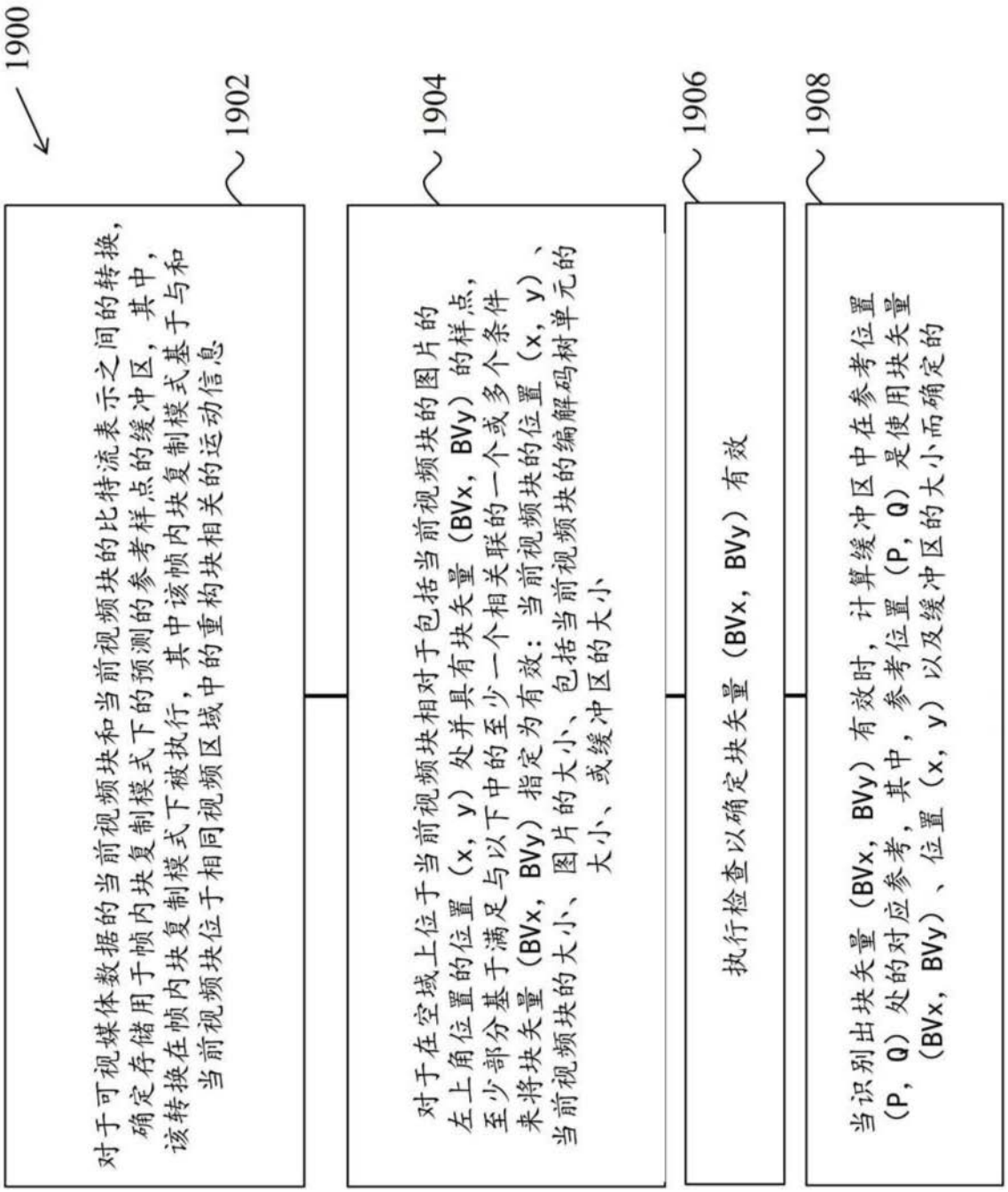

1900
对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换，确定存储用于帧内块复制模式下的预测的参考样点的缓冲区，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息
1902
对于在空域上位于当前视频块相对于包括当前视频块的图片的左上角位置的位置（x，y）处并具有块矢量（BVx，BVy）的样点，至少部分基于满足与以下中的至少一个相关联的一个或多个条件来将块矢量（BVx，BVy）指定为有效：当前视频块的位置（x，y）、当前视频块的大小、图片的大小、包括当前视频块的编解码树单元的大小、或缓冲区的大小
1904
执行检查以确定块矢量（BVx，BVy）有效
1906
当识别出块矢量（BVx，BVy）有效时，计算缓冲区中在参考位置（P，Q）处的对应参考，其中，参考位置（P，Q）是使用块矢量（BVx，BVy）、位置（x，y）以及缓冲区的大小而确定的
1908

图19

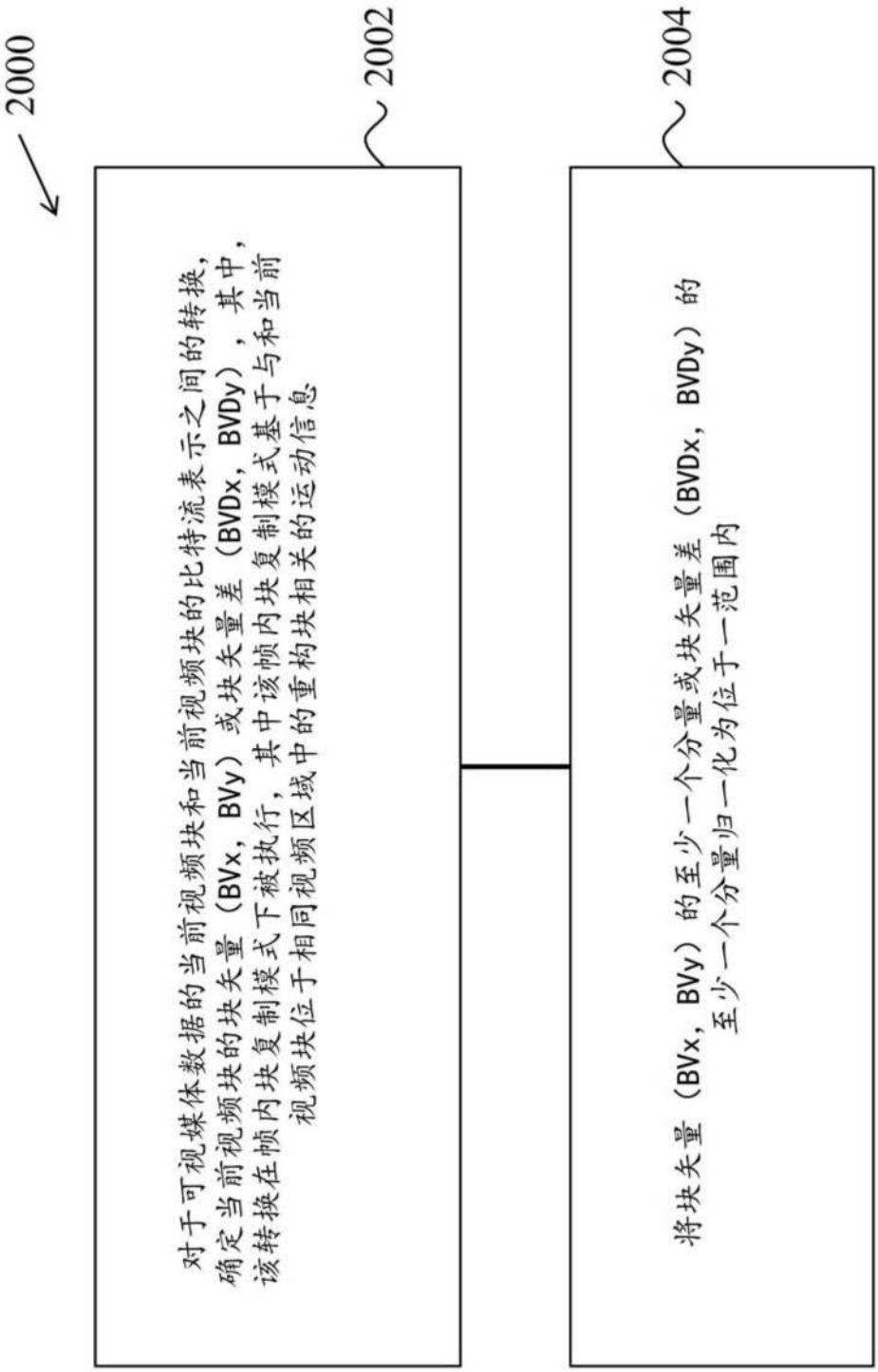

2000
2002
对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换，确定当前视频块的块矢量（BVx，BVy）或块矢量差（BVDx，BVDy），其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息
2004
将块矢量（BVx，BVy）的至少一个分量或块矢量差（BVDx，BVDy）的至少一个分量归一化为位于一范围内

图20

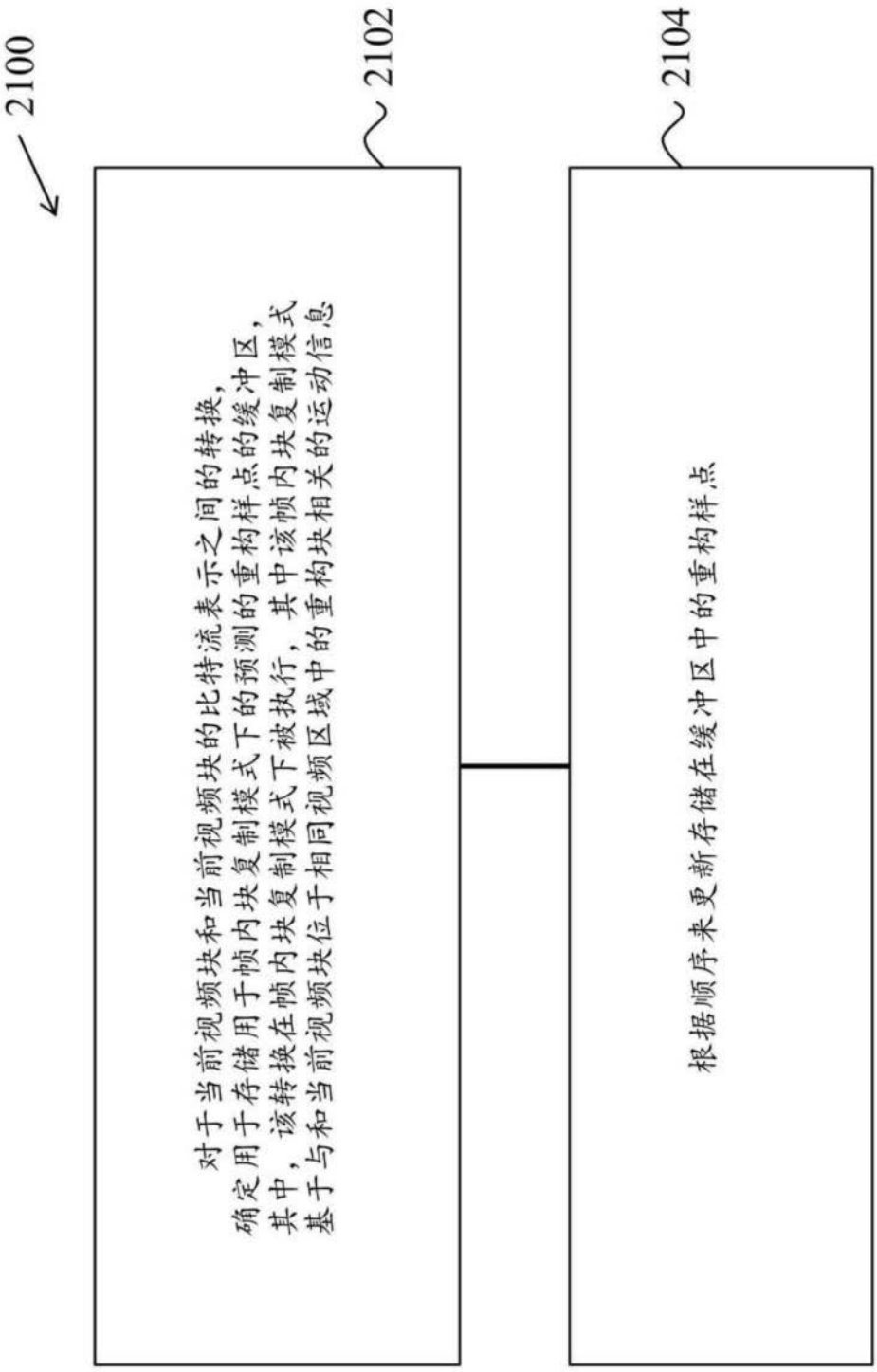

2100
2102
对于当前视频块和当前视频块的比特流表示之间的转换，确定用于存储用于帧内块复制模式下的预测的重构样点的缓冲区，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息
2104
根据顺序来更新存储在缓冲区中的重构样点

图21

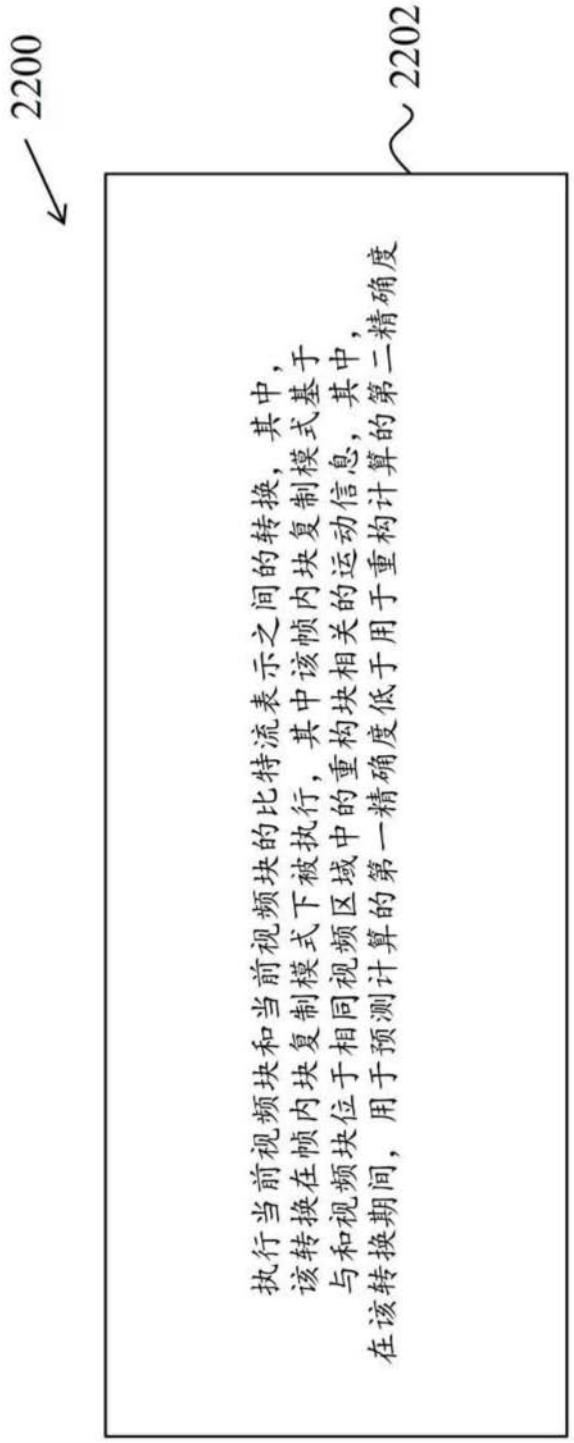

2200
2202
执行当前视频块和当前视频块的比特流表示之间的转换，其中，
该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于
与和视频块位于相同视频区域中的重构块相关的运动信息，其中，
在该转换期间，用于预测计算的第一精确度低于用于重构计算的第二精确度

图22

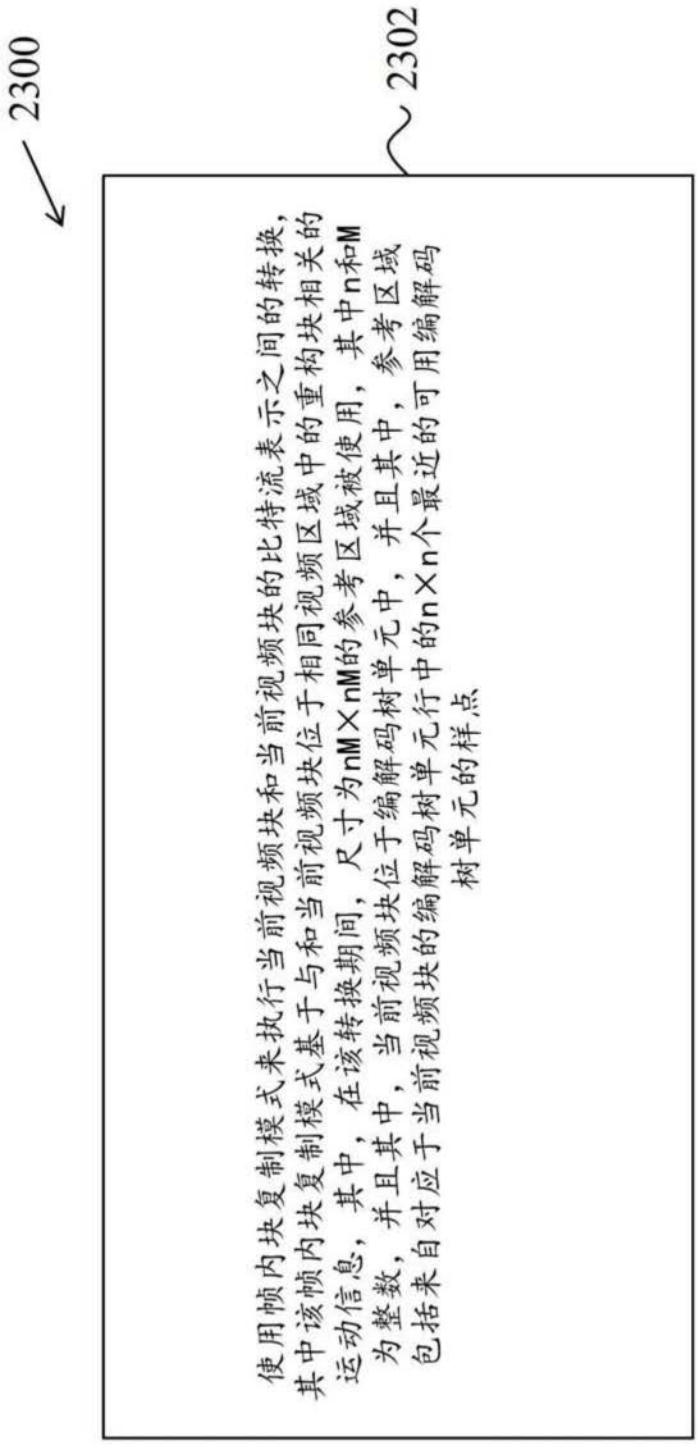

2300
2302
使用帧内块复制模式来执行当前视频块和当前视频块的比特流表示之间的转换，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，尺寸为nM×nM的参考区域被使用，其中n和M为整数，并且其中，当前视频块位于编解码树单元中，并且其中，参考区域包括来自对应于当前视频块的编解码树单元行中的n×n个最近的可用编解码树单元的样点

图23

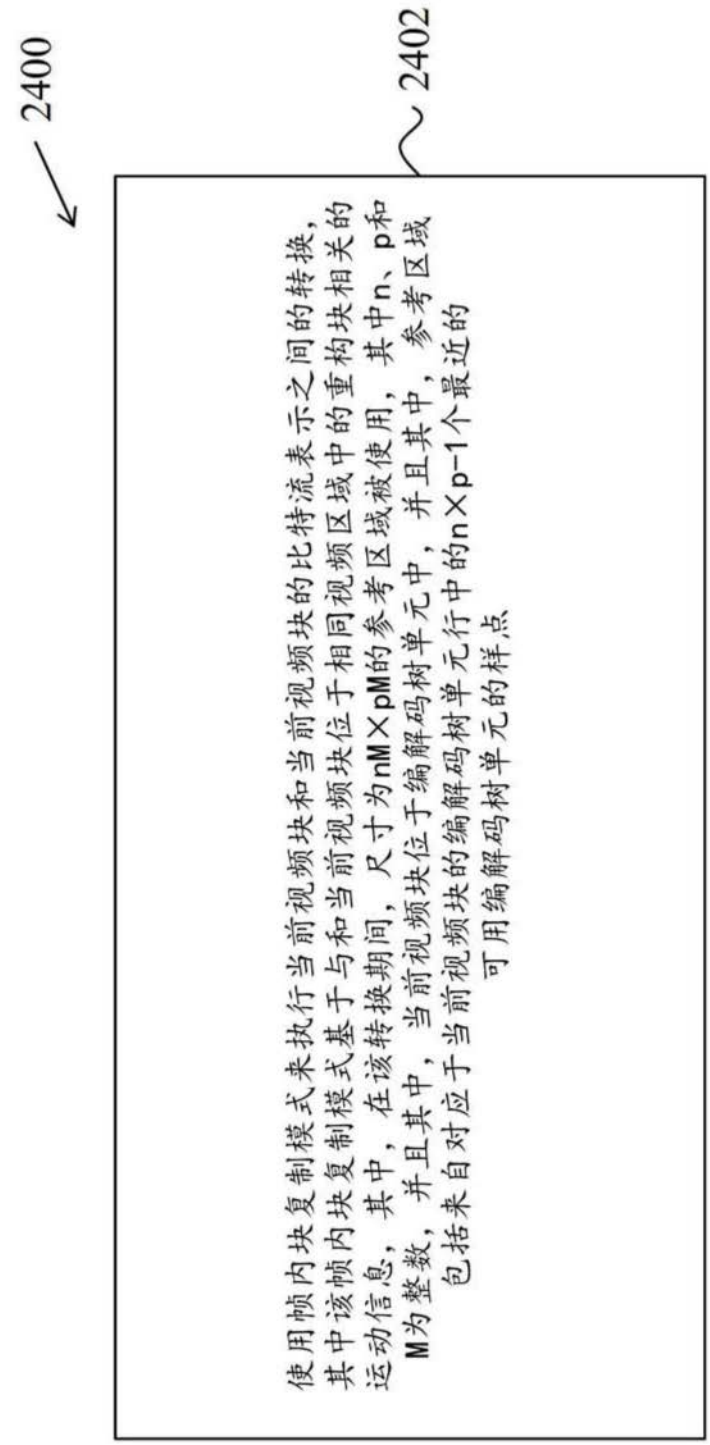

2400
2402
使用帧内块复制模式来执行当前视频块和当前视频块的比特流表示之间的转换，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，尺寸为nM×pM的参考区域被使用，其中n、p和M为整数，并且其中，当前视频块位于编解码树单元中，并且其中，参考区域包括来自对应于当前视频块的编解码树单元行中的n×p-1个最近的可用编解码树单元的样点

图24

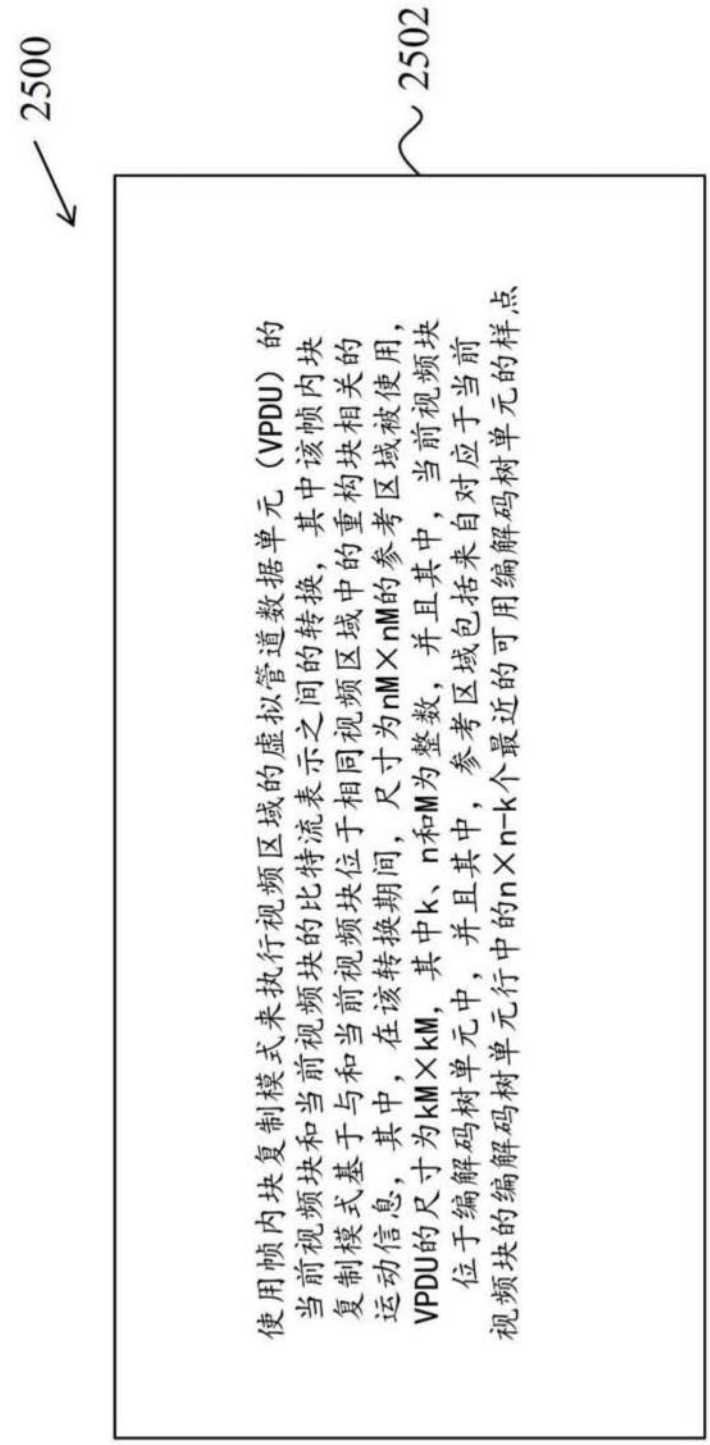

2500
2502
使用帧内块复制模式来执行视频区域的虚拟管道数据单元（VPDU）的当前视频块和当前视频块的比特流表示之间的转换，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息，其中，在该转换期间，尺寸为nM×nM的参考区域被使用，VPDU的尺寸为kM×kM，其中k、n和M为整数，并且其中，当前视频块位于编解码树单元中，并且其中，参考区域包括来自对应于当前视频块的编解码树单元行中的n×n-k个最近的可用编解码树单元的样点

图25

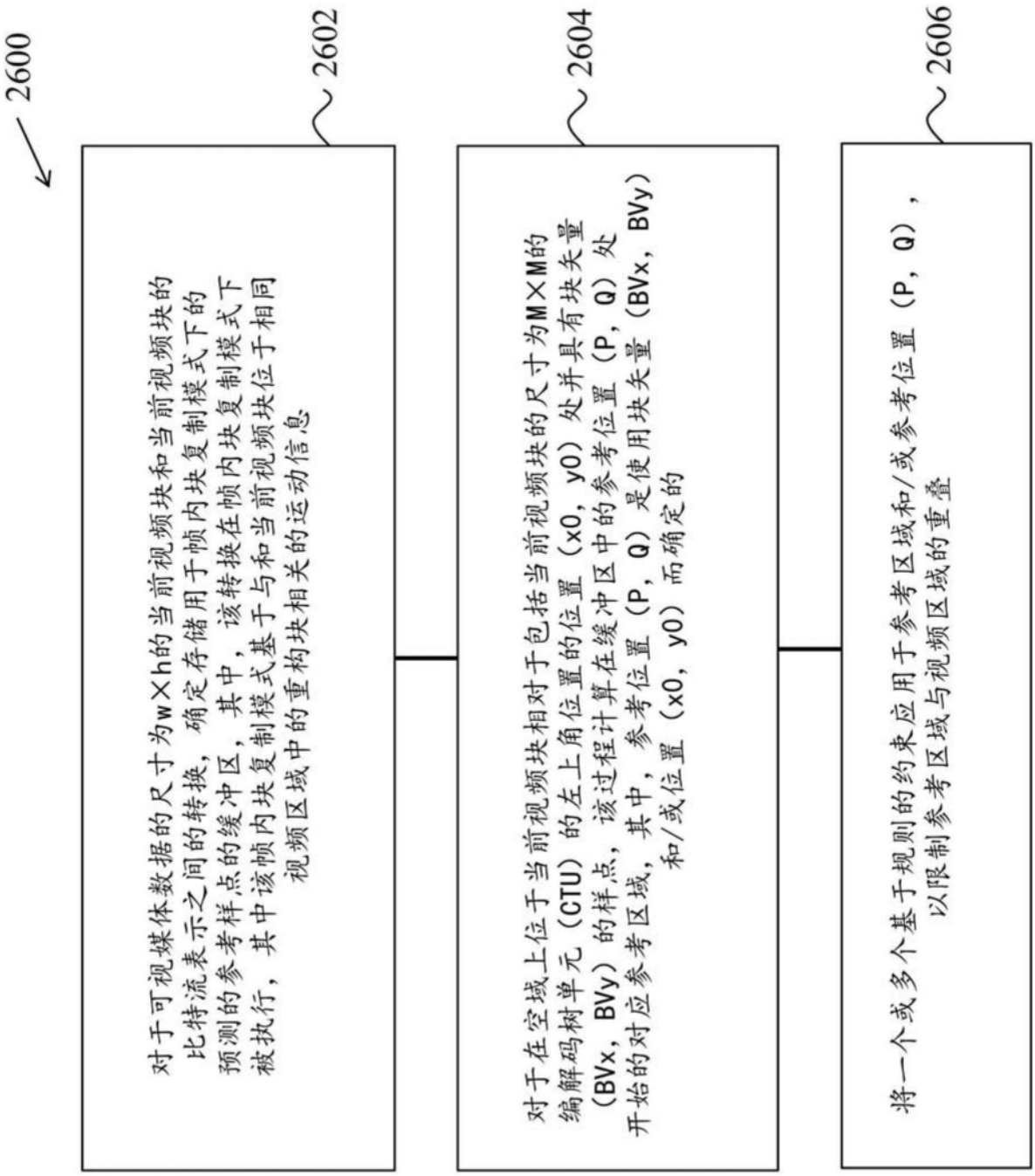

2600
2602
对于可视媒体数据的尺寸为w×h的当前视频块和当前视频块的比特流表示之间的转换，确定存储用于帧内块复制模式下的预测的参考样点的缓冲区，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息
2604
对于在空域上位于当前视频块相对于包括当前视频块的尺寸为M×M的编解码树单元（CTU）的左上角位置的位置（x0，y0）处并具有块矢量（BVx，BVy）的样点，该过程计算在缓冲区中的参考位置（P，Q）处开始的对应参考区域，其中，参考位置（P，Q）是使用块矢量（BVx，BVy）和/或位置（x0，y0）而确定的
2606
将一个或多个基于规则的约束应用于参考区域和/或参考位置（P，Q），以限制参考区域与视频区域的重叠

图26

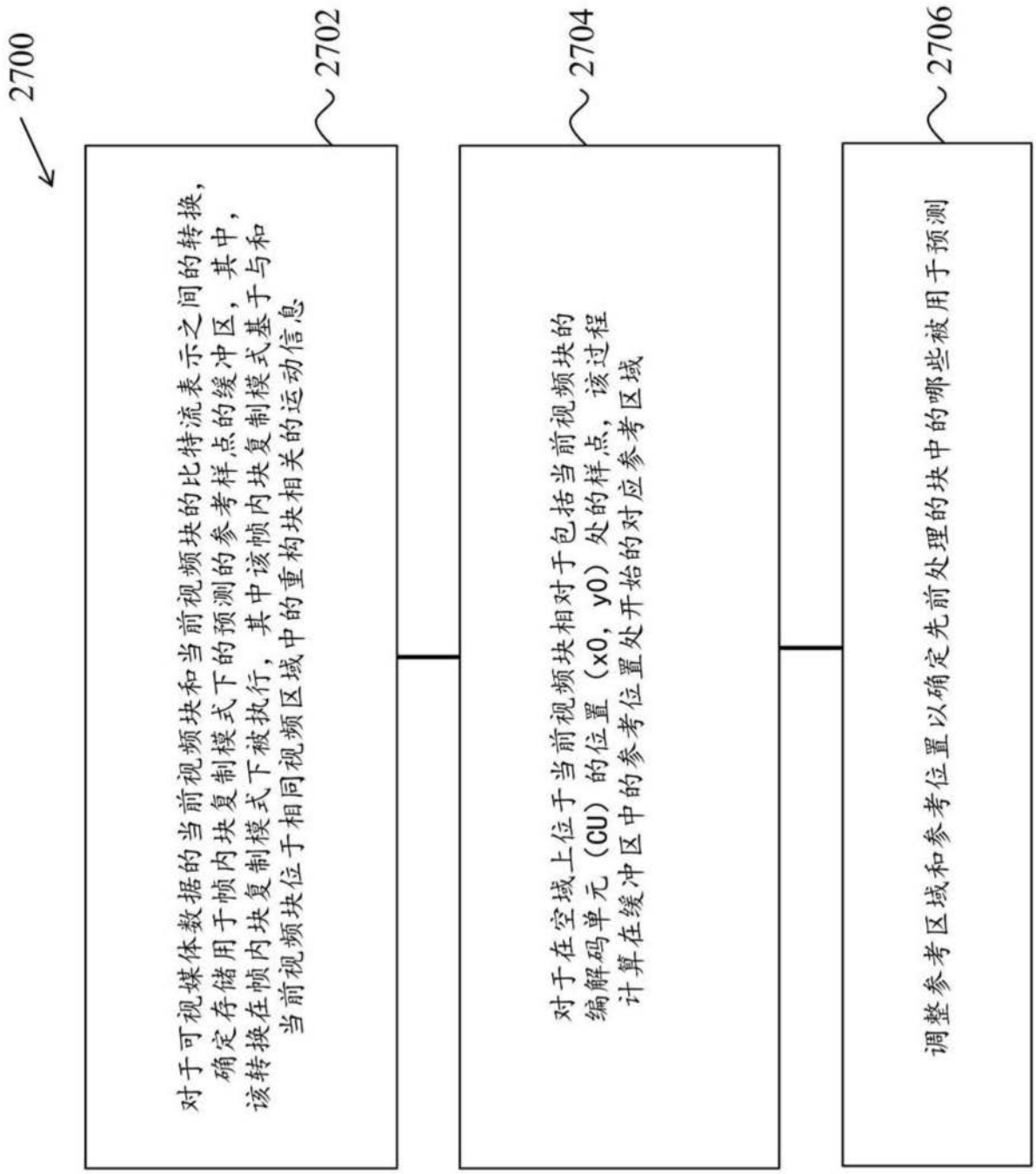

2700
2702
对于可视媒体数据的当前视频块和当前视频块的比特流表示之间的转换，确定存储用于帧内块复制模式下的预测的参考样点的缓冲区，其中，该转换在帧内块复制模式下被执行，其中该帧内块复制模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息
2704
对于在空域上位于当前视频块相对于包括当前视频块的编解码单元（CU）的位置（x0，y0）处的样点，该过程计算在缓冲区中的参考位置处开始的对应参考区域
2706
调整参考区域和参考位置以确定先前处理的块中的哪些被用于预测

图27

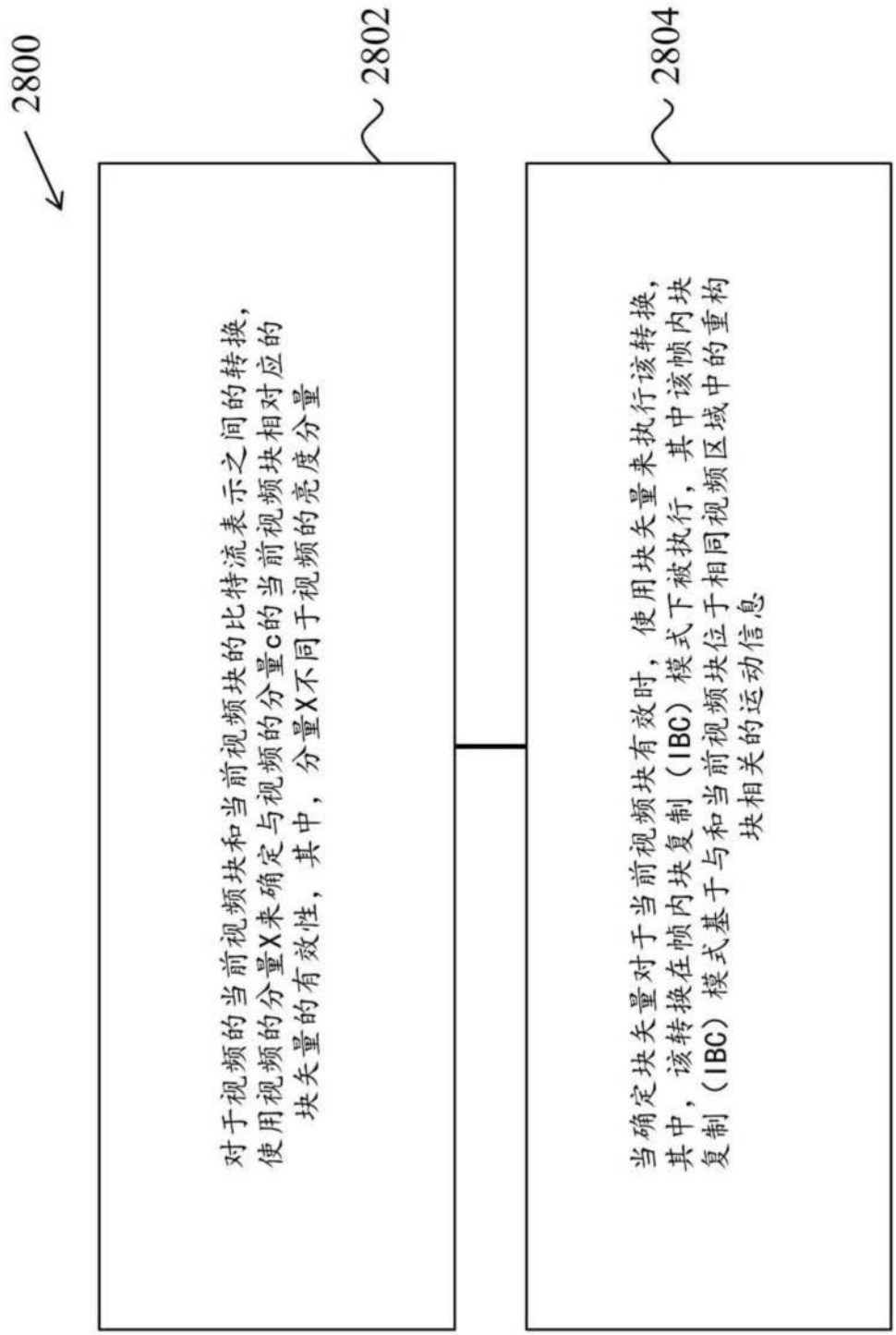

2800
2802
对于视频的当前视频块和当前视频块的比特流表示之间的转换，使用视频的分量X来确定与视频的分量c的当前视频块相对应的块矢量的有效性，其中，分量X不同于视频的亮度分量
2804
当确定块矢量对于当前视频块有效时，使用块矢量来执行该转换，其中，该转换在帧内块复制（IBC）模式下被执行，其中该帧内块复制（IBC）模式基于与和当前视频块位于相同视频区域中的重构块相关的运动信息。

图28