(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 111064965 B
(45)授权公告日 2022.05.24

(21)申请号 201911252271.5

(22)申请日 2015.03.09

(65)同一申请的已公布的文献号
申请公布号 CN 111064965 A

(43)申请公布日 2020.04.24

(30)优先权数据
61/953,922 2014.03.16 US
62/103,916 2015.01.15 US

(62)分案原申请数据
201580014850.X 2015.03.09

(73)专利权人 VID拓展公司
地址 美国特拉华州

(72)发明人 叶琰 修晓宇 贺玉文

(74)专利代理机构 北京润平知识产权代理有限公司 11283
专利代理师 陈潇潇 刘国平

(51)Int.Cl.
H04N 19/70(2014.01)
H04N 19/82(2014.01)
H04N 19/96(2014.01)
H04N 19/61(2014.01)

(56)对比文件
EP 2063417 A1,2009.05.27
CN 103596008 A,2014.02.19
US 2013077696 A1,2013.03.28
Tammy Lee, Sunil Lee.AHG7: Residual quadtree for HEVC lossless 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 12th Meeting: Geneva, CH, 14-23 Jan. 2013 JCTVC-L0118》.2013,
Yih Han Tan, et al.Non-RCE3: Unified lossless residual 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 17th Meeting: Geneva , CH, 23 Oct.- 1 Nov. 2013 JCTVC-O0087》.2013,

审查员 高宇腾

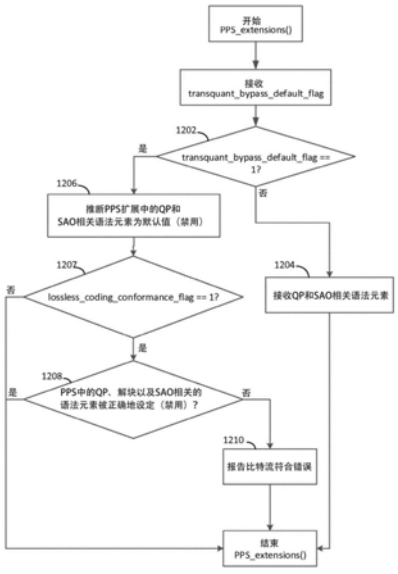
权利要求书3页 说明书34页 附图12页

(54)发明名称

用于无损视频译码的信令的方法、设备和计算机可读介质

(57)摘要

提供了一种用于无损视频译码的信令的方法和系统,所述方法包括:确定要用无损译码对多个译码单元中的至少第一译码单元进行编码;响应于要用无损译码对所述第一译码单元进行编码的所述确定,而至少部分基于从由所述第一译码单元的块大小和译码模式组成的群组中选择的参数来确定变换四叉树分离标志的默认值;以及在所述比特流中用信号发送用于所述第一译码单元的所述变换四叉树分离标志的所述默认值。



CN 111064965 B

1.一种视频编码方法,所述方法包括:

确定要用无损译码对视频中的至少第一译码单元进行编码;

响应于要用无损译码对所述第一译码单元进行编码的所述确定,而至少部分基于从由所述第一译码单元的块大小和译码模式组成的群组中选择的参数来确定变换四叉树分离标志的默认值;以及

在比特流中用信号发送用于所述第一译码单元的所述变换四叉树分离标志的所述默认值。

2.根据权利要求1所述的方法,进一步包括:仅针对所述变换四叉树分离标志的所述默认值,对所述第一译码单元执行率失真测试。

3.根据权利要求1所述的方法,其中所述确定所述变换四叉树分离标志的所述默认值是至少部分基于所述第一译码单元是否被内译码来进行的。

4.根据权利要求1所述的方法,其中所述确定所述变换四叉树分离标志的所述默认值是至少部分基于所述第一译码单元的大小来进行的。

5.根据权利要求1所述的方法,其中所述确定所述变换四叉树分离标志的所述默认值是至少部分基于所述第一译码单元中的预测单元的数量来进行的。

6.根据权利要求1所述的方法,其中确定所述变换四叉树分离标志的默认值包括:

确定所述第一译码单元是否被内译码;

确定所述第一译码单元的大小是否大于大小阈值;以及

确定所述第一译码单元是否精确地包含一个预测单元;

其中,响应于确定所述第一译码单元没有被内译码、所述第一译码单元大于所述大小阈值、并且所述第一译码单元精确地包含一个预测单元,所述变换四叉树分离标志的所述默认值指示无变换四叉树划分。

7.根据权利要求1所述的方法,其中确定所述变换四叉树分离标志的默认值包括:

确定所述第一译码单元是否被内译码;

确定所述第一译码单元的大小是否大于大小阈值;以及

确定所述第一译码单元是否包含至少两个预测单元;

其中,响应于确定所述第一译码单元没有被内译码、所述第一译码单元大于所述大小阈值、并且所述第一译码单元包含至少两个预测单元,所述变换四叉树分离标志的所述默认值指示一次变换四叉树划分。

8.根据权利要求1所述的方法,其中确定所述变换四叉树分离标志的默认值包括:

确定所述第一译码单元是否被内译码;以及

确定所述第一译码单元是否精确地包含一个预测单元;

其中,响应于确定所述第一译码单元被内译码、并且所述第一译码单元精确地包含一个预测单元,所述变换四叉树分离标志的所述默认值指示无变换四叉树划分。

9.根据权利要求1所述的方法,其中确定所述变换四叉树分离标志的默认值包括:

确定所述第一译码单元是否被内译码;以及

确定所述第一译码单元是否包含至少两个预测单元;

其中,响应于确定所述第一译码单元没有被内译码、并且所述第一译码单元包含至少两个预测单元,所述变换四叉树分离标志的所述默认值指示一次变换四叉树划分。

10.一种视频解码设备,所述设备包括处理器,所述处理器被配置成执行至少:

确定所述视频中的译码单元是否被用无损译码进行译码;

响应于确定所述译码单元被用无损译码进行译码,确定用于所述译码单元的变换四叉树分离标志是否在比特流中被用信号发送,其中,如果所述译码单元是被内译码的,则用于所述译码单元的变换四叉树分离标志被确定为没有在所述比特流中被用信号发送;以及

如果所述四叉树分离标志没有在所述比特流中被用信号发送,使用所述变换四叉树分离标志的默认值解码所述译码单元,以及如果所述四叉树分离标志在所述比特流中被用信号发送,则使用用信号发送的所述变换四叉树分离标志解码所述译码单元。

11.根据权利要求10所述的设备,其中如果所述译码单元的块大小大于阈值大小,则所述变换四叉树分离标志进一步被确定为没有在所述比特流中被用信号发送。

12.根据权利要求10所述的设备,其中如果所述译码单元的块大小大于16×16,则所述变换四叉树分离标志进一步被确定为没有在所述比特流中被用信号发送。

13.一种视频编码设备,所述设备包括一个或多个处理器,所述一个或多个处理器被配置成执行至少:

确定要用无损译码对视频中的至少第一译码单元进行编码;

响应于要用无损译码对所述第一译码单元进行编码的所述确定,而至少部分基于从由所述第一译码单元的块大小和译码模式组成的群组中选择的参数来确定变换四叉树分离标志的默认值;以及

在比特流中用信号发送用于所述第一译码单元的所述变换四叉树分离标志的所述默认值。

14.根据权利要求13所述的设备,进一步包括:仅针对所述变换四叉树分离标志的所述默认值,对所述第一译码单元执行率失真测试。

15.根据权利要求13所述的设备,其中所述确定所述变换四叉树分离标志的所述默认值是至少部分基于所述第一译码单元是否被内译码来进行的。

16.根据权利要求13所述的设备,其中所述确定所述变换四叉树分离标志的所述默认值是至少部分基于所述第一译码单元的大小来进行的。

17.根据权利要求13所述的设备,其中所述确定所述变换四叉树分离标志的所述默认值是至少部分基于所述第一译码单元中的预测单元的数量来进行的。

18.一种视频解码方法,包括:

确定所述视频中的译码单元被用无损译码进行译码;

响应于确定所述译码单元被用无损译码进行译码,确定用于所述译码单元的变换四叉树分离标志是否在比特流中被用信号发送,其中,如果所述译码单元是被内译码的,则用于所述译码单元的变换四叉树分离标志被确定为没有在所述比特流中被用信号发送;以及

如果所述四叉树分离标志没有在所述比特流中被用信号发送,使用所述变换四叉树分离标志的默认值解码所述译码单元,以及如果所述四叉树分离标志在所述比特流中被用信号发送,则使用信号发送的所述变换四叉树分离标志解码所述译码单元。

19.根据权利要求18所述的方法,其中如果所述译码单元的块大小大于阈值大小,则所述变换四叉树分离标志进一步被确定为没有在所述比特流中被用信号发送。

20.根据权利要求18所述的方法,其中如果所述译码单元的块大小大于16×16,则所述

变换四叉树分离标志进一步被确定为没有在所述比特流中被用信号发送。

21.一种计算机可读介质，包括用于引起一个或多个处理器执行根据权利要求1-9或18-20所述的方法的指令。

用于无损视频译码的信令的方法、设备和计算机可读介质

[0001] 本申请是申请日为2015年03月09日、申请号为201580014850.X、发明名称为“用于无损视频译码的信令的方法和设备”的中国专利申请的分案申请。

[0002] 相关申请的交叉引用

[0003] 本申请为非临时递交并要求享有根据35 U.S.C.§1 19(e)于2014年3月 16日提交的序列号为61/953,922的美国临时专利申请、以及2015年1月15 日提交的序列号为62/103,916的美国临时专利申请的权益。这些申请的内容通过引用他们的全部的方式结合于此。

背景技术

[0004] 在过去的二十年中,各种数字视频压缩标准已经被开发和标准化以使得高效数字视频通信、发布以及消费能够进行。大多商业上广泛部署的标准由 ISO/IEC和ITU-T开发,例如H.261、MPEG-1、MPEG-2H.263、MPEG-4(部分-2)、以及H.264/AVC(MPEG-4部分10高级视频译码)。由于新的高级视频压缩技术的出现和成熟,由ITU-T视频译码专家组(VCEG)和ISO/IEC MPEG HEVC(ITU-T H.265/ISO/IEC 23008-2)联合开发的高效视频译码 (HEVC)早在2013年被接受为国际标准,并且能够基本实现比当前最先进的技术H.264/AVC更高的译码效率。

[0005] 与传统的数字视频服务(例如通过卫星、电缆和地面传输信道发送TV 信号)相比,在异构环境中部署了越来越多的新视频应用,例如IPTV、视频聊天、移动视频以及流视频。这种异构性存在于客户端以及网络中。在客户端侧,N-屏场景(即在具有不同的屏幕大小和显示器性能的设备上欣赏(consuming)视频内容,该设备包括智能手机、平板电脑、PC以及TV),已经主导着市场并被期望继续主导市场。在网络侧,视频在因特网、WiFi 网络、移动(3G和4G)网络、和/或其组合上传送。

发明内容

[0006] 在这里描述了与无损译码的高级信令相关的系统和方法。在一些实施方式中,所述方法包括生成包含指示使用了无损译码的高级语法元素的视频数据比特流。高级信令语法可以是图片参数集(PPS)、序列参数集(SPS)、视频参数集(VPS)或片分段报头中的一者。无损译码语法元素可以用作用于生成与量化、变换、变换跳过、变换跳过循环以及环路滤波过程相关的一个或多个SPS语法元素的限制(condition)。

[0007] 在一些实施方式中,一种方法包括生成包括 transquant_bypass_default_flag的图片参数集(PPS)。transquant_bypass_default_flag设定为1,指示参考PPS的片中的所有译码单元的cu_transquant_bypass_flag的默认值。PPS也可以具有等于0的transquant_bypass_enabled_flag。

[0008] 在解码器侧,当应用无损译码时,可以绕过(bypass)包括逆量化、逆变换、解块滤波、样本自适应偏移(SAO)等的多个处理块。因此,如果解码器接收到高级无损译码指示,则解码器确定较大群组的译码单元(CU) 将不需要这些处理块。在解码之前关闭这些处理块

可以有利于功率节约、处理循环减少、更好地提供负载。

[0009] 因此在一些实施方式中，一种方法包括在解码器处接收高级无损译码指示，并且作为响应，关闭多个处理块。多个处理块可以包括以下硬件块中的任意一者或多者：逆量化、逆变换、解块滤波、和/或SAO。此外，处理块可以在视频解码之前被关闭，使得处理块硬件组件的至少一部分的功率消耗减少。

附图说明

[0010] 可从以下描述中获取更详细的理解，这些描述是结合附图通过举例给出的，其中：

[0011] 图1A是显示基于块的视频编码器的一个示例的框图。

[0012] 图1B是显示基于块的无损视频编码器的一个示例的框图。

[0013] 图2A是显示基于块的视频解码器的一个示例的框图。

[0014] 图2B是显示基于块的无损视频解码器的一个示例的框图。

[0015] 图3是八个方向预测模式的一个示例的示图。

[0016] 图4是显示33个方向预测模式和两个非方向预测模式的的一个示例的示图。

[0017] 图5是水平预测的一个示例的示图。

[0018] 图6是平面模式的一个示例的示图。

[0019] 图7是显示运动预测的一个示例的示图。

[0020] 图8是显示图片内的块级运动的一个示例的示图。

[0021] 图9是显示一个示例性通信系统的示图。

[0022] 图10是显示一个示例性无线发射/接收单元(WTRU)的示图。

[0023] 图11是显示译码比特流结构的一个示例的示图。

[0024] 图12是使用标志transquant_bypass_default_flag对修改的PPS扩展语法进行解码的方法的流程图。

具体实施方式

[0025] 现在将参考不同附图来提供示例性实施方式的详细描述。虽然该描述提供了对可能的实施的详细示例，但是应当注意所提供的细节是示例性的而并不是要限制申请范围。

[0026] 视频编码及解码

[0027] 图1A是显示基于块的视频编码器(例如混合视频译码系统)的一个示例的框图。视频编码器100可以接收输入视频信号102。输入视频信号102 可以被逐块处理。视频块可以是任意大小的块。例如，视频块单元可以包括 16x16个像素。16x16个像素的视频块单元可以称为宏块(MB)。在高效视频译码(HEVC)中，扩展的块大小(例如其可以称为译码树单元(CTU) 或译码单元(CU)，这两个术语对于本发明来说是等效的)可以用于有效地压缩高分辨率(例如1080p以及更高)视频信号。在HEVC中，CU可以达到64x64像素。CU可以划分成预测单元(PU)，独立的预测方法可以应用于所述预测单元。

[0028] 对于输入视频块(例如MB或CU)，可以执行空间预测160和/或时间预测162。空间预测(例如内预测)可以使用来自相同视频图片/片中的已经译码的邻近块来预测当前视频块。空间预测可以减少视频信号内部的空间冗余。时间预测(例如“间预测”或“运动补偿预测”)可以使用来自已经译码的视频图片(例如其可以称为“参考图片”)的像素来预测当前

视频块。时间预测可以减少视频信号固有的时间冗余。用于视频块的时间预测信号可以由一个或多个运动矢量来用信号发送，其可以指示当前块与其在参考图片中的预测块之间的运动的量和/或方向。如果支持多个参考图片(例如这是对于H.264/AVC和/或HEVC的情况)，则对于视频块，其参考图片可以被发送。参考图片索引可以用于识别时间预测信号是来自参考图片存储器164中的哪个参考图片的。

[0029] 编码器中的模式决定块180例如在空间和/或时间预测之后可以选择预测模式。在116处，预测块可以从当前视频块减去。预测残差可以被变换104 和/或量化106。量化块106可以有效地减去对预测残差进行译码所需的比特数。量化参数(QP)可以用于控制量化的程度。随着QP值的增加，可以应用更严重的量化；因此，译码后的比特数会减小，并且同时解码后的视频质量会降级。由于量化产生的公知的视觉假象包括块效应、模糊、涂抹、振铃、闪烁等等。图1A和图2A中显示的视频译码系统的其他处理块也会引起信息丢失，尤其是当这些处理块应用于需要处理管道中的即时数据的位深的上限的固定点操作时。例如，在变换块104中，水平方向中的变换可以首先应用，随后应用垂直方向中的变换。因为变换会增大数据位深(由于多应用)，在水平变换之后，可以对水平变换的结果应用右移，以便减小垂直变换的输入数据位深。虽然这种移位操作可以帮助减少实施成本(通过减小数据位深)，但是它们也会引起处理管道中的信息丢失。另外，为了使得固定点操作能够进行，最近的视频标准(例如H.264/AVC和HEVC)中的变换是整数值变换。这些整数变换中的一些变换可以接近正交但不完全正交。如果变换 (和逆变换)矩阵不完全正交，则它们不能保证完美的重构。换句话说，即使不使用任何正交，在非正交变换和逆变换被应用到输入数据块之后，输出数据块(缩放因数可以应用于输出)也可能不在数值上维持与输入数据块相同。

[0030] 量化的残差系数可以被逆量化110和/或逆变换112以便形成重构的残差，该重构的残差可以被加回到预测块126以便形成重构的视频块。

[0031] 可以应用环路滤波166(例如解块滤波、样本自适应偏移、自适应环路滤波等)到重构的视频块(在该视频块被放入参考图片存储器164之前和/ 或在用于对将来的视频块进行译码之前)。视频编码器100可以输出输出的视频流120。为了形成输出的视频比特流120，译码模式(例如间预测模式或内预测模式)、预测模式信息、运动信息、和/或量化的残差系数可以被发送给熵译码单元108以便被压缩和/或封装来形成比特流。参考图片存储器164可以称为解码图片缓存器(DPB)。

[0032] 图2A是显示基于块的视频解码器的一个示例的框图。视频解码器200 可以接收视频比特流202。视频比特流202可以在熵解码单元208处被解封装和/或熵解码。用于对视频比特流进行编码的译码模式和/或预测信息可以被发送到空间预测单元260(例如如果被内译码(intra coded))和/或时间预测单元262(例如如果被间译码(inter coded))以形成预测块。如果被间译码，则预测信息可以包括预测块大小、一个或多个运动矢量(例如其可以指示运动的方向和量)、和/或一个或多个参考索引(例如其可以指示预测信号将从哪个参考图片获得)。

[0033] 运动补偿预测可以由时间预测单元262应用以形成时间预测块。残差变换系数可以被发送给逆量化单元210和逆变换单元212以便重构残差块。预测块和残差块可以在226处被添加到一起。重构的块可以通过由环路滤波器 266进行的环路滤波(在其被存储在参考图片存储器264中之前)。参考图片存储器264中的重构的视频可以用于驱动显示设备和/

或用于预测将来的视频块。视频解码器200可以输出重构的视频信号220。参考图片存储器264 也可以称为解码图片缓存器(DPB)。

[0034] 视频编码器和/或解码器(例如视频编码器100或视频解码器200)可以执行空间预测(例如其可以称为内预测)。空间预测可以通过从多个预测方向中的一个预测方向之后的已经译码的邻近像素进行的预测来执行(其可以称为方向内预测)。

[0035] 图3是八个方向预测模式的一个示例的示图。可以在H.264/AVC中支持图3的八个方向预测模式。九个模式(包括DC模式2)是:

[0036] • 模式0:垂直预测

[0037] • 模式1:水平预测

[0038] • 模式2:DC预测

[0039] • 模式3:对角下-左预测

[0040] • 模式4:对角下-右预测

[0041] • 模式5:垂直-右预测

[0042] • 模式6:水平-下预测

[0043] • 模式7:垂直-左预测

[0044] • 模式8:水平-上预测

[0045] 空间预测可以对各种大小和/或形状的视频块执行。可以对例如4x4、8x8 以及16x16像素的块大小(例如在H.264/AVC中)执行视频信号的亮度分量的空间预测。可以对例如8x8像素的块大小(例如在H.264/AVC中)执行视频信号的色度分量的空间预测。对于4x4或8x8大小的亮度块,可以支持总数为九个预测模式,例如八个方向预测模式和DC模式(例如在H.264/AVC 中)。对于例如16x16大小的亮度块,可以支持四个预测模式:例水平、垂直、DC以及平面预测。

[0046] 可以支持方向内预测模式和非方向预测模式。图4是显示了33个方向预测模式和两个非方向预测模式的一个示例的示图。HEVC可以支持图4的 33个方向预测模式和两个非方向预测模式。可以支持使用更大块大小的空间预测。例如,可以对任意大小的块执行空间预测,例如大小为4x4、8x8、16x16、 32x32或64x64的方块。方向内预测(例如在HEVC中)可以使用1/32像素精度执行。

[0047] 例如,除了方向内预测之外,还可以支持非方向内预测模式(例如在 H.264/AVC、HEVC等中)。非方向内预测模式可以包括DC模式和/或平面模式。对于DC模式,方向值可以通过对邻近像素求平均值来获得,并且方向值可以一致地应用到整个块。对于平面模式,线性内插可以用于预测具有慢过渡的平滑区域。H.264/AVC可以允许为16x16亮度块和色度块使用平面模式。

[0048] 编码器(例如编码器100)可以执行模式决定(例如在图1中的块180 处)来为视频块确定最优译码模式。当编码器确定应用内预测(例如不是间预测)时,编码器可以从可用的模式集合确定最佳内预测模式。所选择的方向内预测模式可以提供与输入视频块中的任何纹理、边缘和/或结构的方向相关的强烈暗示。图5是水平方向(例如对于4x4块)的一个示例的示图。已经重构的像素P0、P1、P2以及P3(例如阴影块)可以用于预测当前4x4视频块中的像素。在水平方向中,重构像素例如像素P0、P1、P2和/或P3可以沿着相对应的行的方向水平传播以便预测4x4块。例如,可以根据下面的等式(1)来执行预测,其中L(x,y)可以是在

(x,y)处预测的像素,x,y=0…3。

[0049]
$$\begin{aligned} L(x,0) &= P0 \\ L(x,1) &= P1 \\ L(x,2) &= P2 \\ L(x,3) &= P3 \end{aligned} \qquad (1)$$

[0050] 图6是平面模式的一个示例的示图。平面模式可以相应地执行。顶行中的最右像素(例如标记为T)可以被复制来预测最右列中的像素。左列中的底像素(例如标记为L)可以被复制来预测底行中的像素。可以执行水平方向中的双显性内插(例如如左块所示)来产生中心像素的第一预测H(x,y)。可以执行垂直方向中的双显性内插(例如如右块所示)来产生中心像素的第二预测V(x,y)。使用L(x,y)=((H(x,y)+V(x,y))>>1,可以执行在水平预测与垂直预测之间求平均值以获得最后的预测L(x,y)。

[0051] 图7和图8是显示视频块的运动预测(例如使用图1的运动预测单元 162)的一个示例的示图。图8是显示一个示例性解码图片缓存器的示图,其包括例如参考图片“Ref pic 0”、“Ref pic 1”以及“Ref pic 2”。当前图片中的块B0、B1以及B2可以分别从参考图片“Ref pic 0”、“Ref pic 1”以及“Ref pic 2”中的块预测。运动预测可以使用来自邻近视频帧的视频块来预测当前视频块。运动预测可以使用视频信号中固有的时间相关性和/或移除视频信号中固有的时间冗余。例如在H.264/AVC和HEVC中,可以对各种大小的视频块执行时间预测(例如对于亮度分量,时间预测块大小可以在 H.264/AVC中从16x16变化到4x4,在HEVC中从64x64变化到4x4)。在(mvx, mvy)的运动矢量中,可以执行由等式(1)提供的时间预测:

[0052] P(x,y)=ref(x-mvx,y-mvy) (1)

[0053] 其中ref(x,y)可以是参考图片中的位置(x,y)处的像素值,并且P(x,y)可以是被预测的块。视频译码系统可以支持具有分数像素精度的间预测。当运动矢量 (mvx,mvy)具有分数像素值时,可以应用一个或多个内插滤波来获得分数像素位置处的像素值。基于块的视频译码系统可以使用多假设预测来改进时间预测,例如,其中预测信号可以通过合并来自不同的参考图片的多个预测信号来形成。例如,H.264/AVC和/或HEVC可以使用可合并两个方向信号的双向预测。双向可以合并两个预测信号(每个预测信号来自一个参考图片)来形成预测,例如下面的等式(2):

[0054]
$$P(x,y)=\frac{P_0(x,y)+P_1(x,y)}{2}=\frac{ref_0(x-mvx_0,y-mvy_0)+ref_1(x-mvx_1,y-mvy_1)}{2} \qquad (2)$$

[0055] 其中P_0(x,y)和P_1(x,y)可以分别为第一和第二预测块。如等式(2)中所示,两个方向块可以通过执行分别从两个参考图片ref_0(x,y)和ref_x(x,y)用运动矢量(mvx_0,mvy_0)和(mvx_1,mvy_1)执行运动补偿预测来获得。可以从源视频块减去预测块(例如在加法器116处)来形成预测残差块。预测残差块可以被变换 (例如在变换单元104处)和/或量化(例如在量化单元106处)。量化残差变换系数块可以被发送给熵译码单元(例如熵译码单元108)以便被熵译码以减小比特率。熵译码残差系数可以被封装以便形成部分输出的视频比特流 (例如比特流120)。

[0056] 图11是显示译码的比特流结构的一个示例的示图。译码后的比特流 1000由多个NAL(网络应用层)单元1001组成。NAL单元可以包含诸如译码后的片1006之类的译码后的样本数据、或诸如参数集数据之类的高级语法元数据、片报头数据1005或补充增强信息数据

1007(其可以称为SEI 消息)。参数集是包含必要的语法元素的高级语法结构,该必要的语法元素可应用于多个比特流层(例如视频参数集1002(VPS))、或可应用于一个层内的译码后的视频序列(例如序列参数集1003(SPS))、或可应用于一个译码后的视频序列内的多个译码后的图片(例如图片参数集1004(PPS))。参数集可以与视频比特流的译码后的图片一起被发送,或者可以通过其他方式发送(包括使用可靠的信号、硬译码等的带外传输)。片报头1005也是高级语法结构,其可以包含相对较小或者仅针对特定片或图片类型相关的一些图片相关信息。SEI消息1007携带解码过程可能不需要但是能用于各种其它目的(例如图片输出定时或显示和/或损失检测和隐藏)的信息。

[0057] 图9是显示通信系统的一个示例的示图。通信系统1300可以包括编码器1302、通信网络1304、以及解码器1306。编码器1302可以经由连接1308 与通信网络1304进行通信。连接1308可以是有线连接或无线连接。编码器 1302可以类似于图1的基于块的视频编码器。编码器1302可以包括单层编解码器(例如图1)或多层编解码器。

[0058] 解码器1306可以经由连接1310与通信网络1306进行通信。连接1310 可以是有线连接或无线连接。解码器1306可以类似于图2A的基于块的视频解码器。解码器1306可以包括单层编解码器(例如图2A)或多层编解码器。例如,解码器1306可以是使用图片级ILP支持的多层(例如两层)可缩放解码系统。

[0059] 编码器1302和/或解码器1306可以合并到各种有线通信设备和/或无线发射/接收单元(WTRU),例如但不限于数字电视、无线广播系统、网络元件/终端、服务器、例如内容或网页服务器(例如超文本传输协议(HTTP) 服务器)、个人数字助理(PDA)、笔记本电脑或台式电脑、平板电脑、数码相机、数码录音设备、视频游戏设备、视频游戏机、蜂窝或卫星无线电话、数字媒体播放器等。

[0060] 通信网络1304可以是任意合适的类型的通信网络。例如,通信网络1304 可以是用于提供诸如语音、数据、视频、消息、广播等内容给多个无线用户的多址系统。通信网络1304能够使得多个无线用户通过共享系统资源(包括无线带宽)来接入这些内容。例如,通信网络1304可以使用一种或多种信道接入方法,例如码分多址(CDMA)、时分多址(TDMA)、频分多址 (FDMA)、正交FDMA(OFDMA)、单载波FDMA(SC-FDMA)等。通信网络1304可以包括多个连接的通信网络。通信网络1304可以包括因特网和 /或一个或多个私人商业网络,例如蜂窝网、WiFi热点、因特网服务供应商网络(ISP的网络)等。

[0061] 图10是示例WTRU的系统图。WTRU 902可以包括处理器918、收发信机920、发射/接收元件922、扬声器/麦克风924、小键盘或大键盘926、显示器/触控板928、不可移除存储器930、可移除存储器932、电源934、全球定位系统(GPS)芯片组936、以及其它外围设备938。应认识到WTRU 902 在保持与实施方式一致的同时,可以包括前述元件的任何子组合。另外,可合并编码器(例如编码器802)和/或解码器(例如解码器806)的终端可以包括图10画出的以及这里参考图10的WTRU 902描述的一些元件和所有元件。

[0062] 处理器918可以是通用处理器、专用处理器、常规处理器、数字信号处理器(DSP)、地图处理单元(GPU)、多个微处理器、与DSP核相关联的一个或多个微处理器、控制器、微控制器、专用集成电路(ASIC)、现场可编程门阵列(FPGA)电路、任何其它类型的集成电路(IC)、状态机等等。处理器918可以执行信号译码、数据处理、功率控制、输入/输出处理、和/或使得WTRU 902能够在有线和/或无线环境中操作的任何其它功能。处理器918可以耦合到

收发信机920,收发信机920可以耦合到发射/接收元件922。虽然图10将处理器918和收发信机920画为单独的元件,但应认识到处理器918和收发信机920可以被一起集成在电子组件和/或芯片中。

[0063] 发射/接收元件922可以被配置为通过空中接口915向另一终端发射信号和/或从另一终端接收信号。例如,在一个或多个实施方式中,发射/接收元件922可以是被配置为发射和/或接收RF信号的天线。在一个或多个实施方式中,发射/接收元件922可以是被配置为发射和/或接收例如IR、UV、或可见光信号的发射器/检测器。在一个或多个实施方式中,发射/接收元件922 可以被配置为发射和/或接收RF和光信号两者。应认识到发射/接收元件922 可以被配置为发射和/或接收无线信号的任何组合。

[0064] 另外,虽然发射/接收元件922在图10中被画为单个元件,但WTRU 902 可以包括任何数目的发射/接收元件922。更具体而言,WTRU 902可以采用 MIMO技术。因此,在一些实施方式中,WTRU 902可以包括用于通过空中接口915来发射和接收无线信号的两个或更多个发射/接收元件922(例如多个天线)。

[0065] 收发信机920可以被配置为调制将由发射/接收元件922发射的信号和/ 或对由发射/接收元件922接收到的信号进行解调。如上所述,WTRU 902 可以具有多模式能力。因此,例如,收发信机920可以包括多个收发信机,该多个收发信机用于使得WTRU 902能够包括经由诸如UTRA和IEEE 802.11之类的多种RAT进行通信。

[0066] WTRU 902的处理器918可以耦合到扬声器/麦克风924、键盘926、和/ 或显示器/触控板928(例如液晶显示器(LCD)显示单元或有机发光二极管 (OLED)显示单元),并且可以从这些组件接收用户输入数据。处理器918 还可以向扬声器/麦克风924、键盘926、和/或显示器/触控板928输出用户数据。另外,处理器918可以访问来自任意类型的合适的存储器(例如不可移除存储器930和可移除存储器932)的信息,或者将数据存储在该任意类型的合适的存储器中。不可移除存储器930可以包括随机存取存储器 (RAM)、只读存储器(ROM)、硬盘、或任何其它类型的存储器存储设备。可移除存储器932可以包括用户标识模块(SIM)卡、记忆棒、安全数字(SD) 存储卡等。在一个或多个实施方式中,处理器918可以访问来自在物理上不位于WTRU 902上(诸如在服务器或家用计算机(未示出))的存储器的信息并将数据存储在该存储器中。

[0067] 处理器918可以从电源934接收电力,并且可以被配置为分配和/或控制到WTRU 902中的其它元件的电力。电源934可以是用于为WTRU 902 供电的任何适当设备。例如,电源934可以包括一个或多个干电池(例如镍镉(NiCd)、镍锌(NiZn)、镍金属氢化物(NiMH)、锂离子(Li)等等)、太阳能电池、燃料电池等等。

[0068] 处理器918还可以耦合到GPS芯片组936,GPS芯片组936可以被配置为提供关于WTRU 902的当前位置的位置信息(例如,经度和纬度)。除来自GPS芯片组936的信息之外或作为其替代,WTRU 902可以通过空中接口 915从终端(例如,基站)接收位置信息和/或基于从两个或更多个附近的基站接收到信号的时序来确定其位置。应认识到WTRU 902可以在保持与实施方式一致的同时,通过任何适当的位置确定方法来获取位置信息。

[0069] 处理器918还可以耦合到其它外围设备938,外围设备938可以包括提供附加特征、功能和/或有线或无线连接的一个或多个软件和/或硬件模块。例如,外围设备938可以包括加速计、方向传感器、移动传感器、接近传感器、电子指南针、卫星收发信机、数码相机和/或

录像机(例如用于拍照和/ 或视频)、通用串行总线(USB)端口、振动设备、电视收发信机、免提耳机、蓝牙®模块、调频(FM)无线电单元、以及软件模块例如数字音乐播放器、媒体播放器、视频游戏机模块、因特网浏览器等等。

[0070] 举例来说,WTRU 902可以被配置为传送和/或接收无线信号,并且可以包括用户设备(UE)、移动站、固定或移动用户单元、寻呼机、蜂窝电话、个人数字助理(PDA)、智能手机、笔记本电脑、上网本、平板电脑、个人电脑、无线传感器、消费电子设备、或能够接收和处理压缩视频通信的任意其他终端。

[0071] WTRU 902和/或通信网络(例如通信网络804)可以实施无线电技术,例如通用移动通信系统(UMTS)陆地无线接入(UTRA),其可以使用宽带 CDMA(WCDMA)来建立空中接口915。WCDMA可以包括诸如高速分组接入(HSPA)和/或演进型HSPA(HSPA+)之类的通信协议。HSPA可以包括高速下行链路分组接入(HSDPA)和/或高速上行链路分组接入(HSUPA)。WTRU 902和/或通信网络(例如通信网络804)可以实施诸如演进型UMTS 陆地无线接入(E-UTRAN)之类的无线电技术,其可以使用长期演进(LTE) 和/或高级LTE(LTE-A)来建立空中接口915。

[0072] WTRU 902和/或通信网络(例如通信网络804)可以实施诸如IEEE 802.16((例如全球互通微波接入(WiMAX))、CDMA2000、CDMA2000 1X、CDMA2000 EV-DO、暂行标准2000(IS-2000)、暂行标准95(IS-95)、暂行标准856(IS-856)、全球移动通信系统(GSM)、增强型GSM演进数据率 (EDGE)、GSM EDGE(GERAN)等之类的无线电技术。WTRU 902和/或通信网络(例如通信网络804)可以实施诸如IEEE 802.11、IEEE 802.15等之类的无线电技术。

[0073] 无损译码

[0074] 对于一些视频应用,例如医学视频应用和高端专业视频应用,可能希望没有任何损失的保留原始视频信号中的所有信息。对于这种视频应用,可以使用无损译码。在无损译码中,可引起信息丢失(例如变换和量化)的视频编解码器中的处理块可以被修改和/或绕过。图1B和图2B中的编码器和解码器配置可以分别用于实现无损译码。在无损译码中,不应用用于变换、量化、逆变换以及逆量化的处理块。另外,由于重构的视频块作为图1B的加法器126和图2B的加法器226的结果在数值上与原始视频块相同,因此环路滤波可能不是必须的,并且其确实在实际上引入了不希望的失真。因此,在一些实施方式中也不应用用于环路滤波的处理块。

[0075] 由于诸如无线显示之类的视频应用和云计算的快速增长,近年来屏幕内容译码(SCC)已经收到了来自学术和工业上的许多关注。虽然与先前的视频译码标准相比,HEVC已经在译码效率上实现了显著改进,但是其主要针对由摄像机捕获的自然视频而进行了设计。然而,屏幕内容视频(其典型地由计算机生成的内容(例如文本和图像)组成)显示了与自然内容非常不同的属性。鉴于这种情况,期望针对屏幕内容译码来扩展HEVC。内块复制(IBC)是HEVC屏幕内容译码扩展中已经采用的一种译码方法,如R.Joshi、J.Xu在2014年10月(Joshi 2014)所著的文献编号为No.JCTVC-S1005的 HEVC Screen Content Coding Draft Text 2中所述。IBC被设计为通过从同一图片的已重构区域的像素预测当前PU的像素,来利用一个图片内固有的图片内冗余(尤其是在图片包含文本和图像丰富的大量屏幕内容的情况下)。类似于间模式,对于使用IBC模式译码的CU,一个预测的PU与其参考块之间的位移由块矢量(BV)表示。BV与比特流中的相对应残差一起译码。

[0076] 在HEVC与其扩展中,称为transquant_bypass_enabled_flag的语法元素在图片参

数集(PPS)中被用信号发送以便指示变换和量化是否可以在逐块的基础上被绕过。如D.Flynn,M.Naccari,C.Rosewarne、J.Sole,G.Sullivan、T.Suzuk在2014年1月所著的文献编号为JCTVC-P1005的“High Efficiency Video Coding(HEVC)Range Extensions text specification:Draft 6”中所述，HEVC中的PPS语法表格被显示在下面的表1中,其中transquant_bypass_enabled_flag被显示在第22行：

[0077] 表1:HEVC范围扩展草案6中的PPS语法表格

1	pic_parameter_set_rbsp( ) {	描述符
2	**pps_pic_parameter_set_id**	ue(v)
3	**pps_seq_parameter_set_id**	ue(v)
4	**dependent_slice_segments_enabled_flag**	u(1)
5	**output_flag_present_flag**	u(1)
6	**num_extra_slice_header_bits**	u(3)
7	**sign_data_hiding_enabled_flag**	u(1)
8	**cabac_init_present_flag**	u(1)
9	**num_ref_idx_l0_default_active_minus1**	ue(v)
10	**num_ref_idx_l1_default_active_minus1**	ue(v)
11*	**init_qp_minus26**	se(v)
12	**constrained_intra_pred_flag**	u(1)
13*	**transform_skip_enabled_flag**	u(1)
14*	**cu_qp_delta_enabled_flag**	u(1)
15	if(cu_qp_delta_enabled_flag)	
16*	**diff_cu_qp_delta_depth**	ue(v)
17*	**pps_cb_qp_offset**	se(v)
18*	**pps_cr_qp_offset**	se(v)
19*	**pps_slice_chroma_qp_offsets_present_flag**	u(1)
[0078] 20	**weighted_pred_flag**	u(1)
21	**weighted_bipred_flag**	u(1)
22	**transquant_bypass_enabled_flag**	u(1)
23	**tiles_enabled_flag**	u(1)
24	**entropy_coding_sync_enabled_flag**	u(1)
25	if(tiles_enabled_flag) {	
26	**num_tile_columns_minus1**	ue(v)
27	**num_tile_rows_minus1**	ue(v)
28	**uniform_spacing_flag**	u(1)
29	if(!uniform_spacing_flag) {	
30	for(i = 0; i < num_tile_columns_minus1; i++)	
31	**column_width_minus1**[i]	ue(v)
32	for(i = 0; i < num_tile_rows_minus1; i++)	
33	**row_height_minus1**[i]	ue(v)
34	}	
35*	**loop_filter_across_tiles_enabled_flag**	u(1)
36	}	
37*	**pps_loop_filter_across_slices_enabled_flag**	u(1)
38*	**deblocking_filter_control_present_flag**	u(1)
39	if(deblocking_filter_control_present_flag) {	

[0079]

40*	**deblocking_filter_override_enabled_flag**	u(1)
41*	**pps_deblocking_filter_disabled_flag**	u(1)
42	if(!pps_deblocking_filter_disabled_flag) {	
43*	**pps_beta_offset_div2**	se(v)
44*	**pps_tc_offset_div2**	se(v)
45	}	
46	}	
47*	**pps_scaling_list_data_present_flag**	u(1)
48	if(pps_scaling_list_data_present_flag)	
49	scaling_list_data( )	
50	**lists_modification_present_flag**	u(1)
51	**log2_parallel_merge_level_minus2**	ue(v)
52	**slice_segment_header_extension_present_flag**	u(1)
53	**pps_extension_present_flag**	u(1)
54	if(pps_extension_present_flag) {	
55	for(i = 0; i < 1; i++)	
56	**pps_extension_flag**[i]	u(1)
57	**pps_extension_7bits**	u(7)
58	}	
59	if(pps_extension_flag[0]) {	
60	if(transform_skip_enabled_flag)	
61*	**log2_max_transform_skip_block_size_minus2**	ue(v)
62	**cross_component_prediction_enabled_flag**	u(1)
63*	**chroma_qp_adjustment_enabled_flag**	u(1)
64	if(chroma_qp_adjustment_enabled_flag) {	
65*	**diff_cu_chroma_qp_adjustment_depth**	ue(v)
66*	**chroma_qp_adjustment_table_size_minus1**	ue(v)
67	for(i = 0; i <= chroma_qp_adjustment_table_size_minus1; i++) {	
68*	**cb_qp_adjustment**[i]	se(v)
69*	**cr_qp_adjustment**[i]	se(v)
70	}	
71	}	
72*	**log2_sao_offset_scale_luma**	ue(v)
73*	**log2_sao_offset_scale_chroma**	ue(v)
74	}	
75	if(pps_extension_7bits)	
76	while(more_rbsp_data( ))	
77	**pps_extension_data_flag**	u(1)
78	rbsp_trailing_bits( )	
79	}	

[0080] 如果当前片参考其中transquant_bypass_enabled_flag(表1的第22行) 被设定为1(这通过将片报头中的slice_pic_parameter_set_id(表2中的第5 行)设定为适当的值以识别适当的PPS来执行)的PPS,则在译码单元或 CU级别,称为cu_transquant_bypass_flag的另外的标志针对当前片中的所有 CU而被发送。表3中显示了coding_unit元素表。

[0081] 表2:HEVC范围扩展草案6中的片报头语法表格

[0082]

1	slice_segment_header( ) {	描述符
2	**first_slice_segment_in_pic_flag**	u(1)
3	if(nal_unit_type >= BLA_W_LP && nal_unit_type <= RSV_IRAP_VCL23)	
4	**no_output_of_prior_pics_flag**	u(1)
5	**slice_pic_parameter_set_id**	ue(v)
6	if(!first_slice_segment_in_pic_flag) {	
7	if(dependent_slice_segments_enabled_flag)	
8	**dependent_slice_segment_flag**	u(1)
9	**slice_segment_address**	u(v)
10	}	
11	if(!dependent_slice_segment_flag) {	
12	for(i = 0; i < num_extra_slice_header_bits; i++)	
13	**slice_reserved_flag**[i]	u(1)
14	**slice_type**	ue(v)
15	if(output_flag_present_flag)	
16	**pic_output_flag**	u(1)
17	if(separate_colour_plane_flag = = 1)	
18	**colour_plane_id**	u(2)
19	if(nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) {	
20	**slice_pic_order_cnt_lsb**	u(v)
21	**short_term_ref_pic_set_sps_flag**	u(1)
22	if(!short_term_ref_pic_set_sps_flag)	
23	short_term_ref_pic_set(num_short_term_ref_pic_sets)	
24	else if(num_short_term_ref_pic_sets > 1)	
25	**short_term_ref_pic_set_idx**	u(v)
26	if(long_term_ref_pics_present_flag) {	
27	if(num_long_term_ref_pics_sps > 0)	
28	**num_long_term_sps**	ue(v)
29	**num_long_term_pics**	ue(v)
30	for(i = 0; i < num_long_term_sps + num_long_term_pics; i++) {	
31	if(i < num_long_term_sps) {	
32	if(num_long_term_ref_pics_sps > 1)	
33	**lt_idx_sps**[i]	u(v)
34	} else {	
35	**poc_lsb_lt**[i]	u(v)
36	**used_by_curr_pic_lt_flag**[i]	u(1)
37	}	
38	**delta_poc_msb_present_flag**[i]	u(1)
39	if(delta_poc_msb_present_flag[i])	
40	**delta_poc_msb_cycle_lt**[i]	ue(v)
41	}	
42	}	
43	if(sps_temporal_mvp_enabled_flag)	

[0083]

44	**slice_temporal_mvp_enabled_flag**	u(1)
45	}	
46	if(sample_adaptive_offset_enabled_flag) {	
47*	**slice_sao_luma_flag**	u(1)
48	if(ChromaArrayType != 0)	
49*	**slice_sao_chroma_flag**	u(1)
50	}	
51	if(slice_type = = P | | slice_type = = B) {	
52	**num_ref_idx_active_override_flag**	u(1)
53	if(num_ref_idx_active_override_flag) {	
54	**num_ref_idx_l0_active_minus1**	ue(v)
55	if(slice_type = = B)	
56	**num_ref_idx_l1_active_minus1**	ue(v)
57	}	
58	if(lists_modification_present_flag && NumPocTotalCurr > 1)	
59	ref_pic_lists_modification( )	
60	if(slice_type = = B)	
61	**mvd_l1_zero_flag**	u(1)
62	if(cabac_init_present_flag)	
63	**cabac_init_flag**	u(1)
64	if(slice_temporal_mvp_enabled_flag) {	
65	if(slice_type = = B)	
66	**collocated_from_l0_flag**	u(1)
67	if((collocated_from_l0_flag && num_ref_idx_l0_active_minus1 > 0) | | (!collocated_from_l0_flag && num_ref_idx_l1_active_minus1 > 0))	
68	**collocated_ref_idx**	ue(v)
69	}	
70	if((weighted_pred_flag && slice_type = = P) | | (weighted_bipred_flag && slice_type = = B))	
71	pred_weight_table( )	
72	**five_minus_max_num_merge_cand**	ue(v)
73	}	
74*	**slice_qp_delta**	se(v)
75	if(pps_slice_chroma_qp_offsets_present_flag) {	
76*	**slice_cb_qp_offset**	se(v)
77*	**slice_cr_qp_offset**	se(v)
78	}	
79	if(chroma_qp_adjustment_enabled_flag)	
80*	**slice_chroma_qp_adjustment_enabled_flag**	u(1)
81	if(deblocking_filter_override_enabled_flag)	
82*	**deblocking_filter_override_flag**	u(1)
83	if(deblocking_filter_override_flag) {	
84*	**slice_deblocking_filter_disabled_flag**	u(1)
85	if(!slice_deblocking_filter_disabled_flag) {	
86*	**slice_beta_offset_div2**	se(v)
87*	**slice_tc_offset_div2**	se(v)

[0084]

88	}	
89	}	
90	if(pps_loop_filter_across_slices_enabled_flag && (slice_sao_luma_flag || slice_sao_chroma_flag || !slice_deblocking_filter_disabled_flag))	
91*	**slice_loop_filter_across_slices_enabled_flag**	u(1)
92	}	
93	if(tiles_enabled_flag || entropy_coding_sync_enabled_flag) {	
94	**num_entry_point_offsets**	ue(v)
95	if(num_entry_point_offsets > 0) {	
96	**offset_len_minus1**	ue(v)
97	for(i = 0; i < num_entry_point_offsets; i++)	
98	**entry_point_offset_minus1**[i]	u(v)
99	}	
100	}	
101	if(slice_segment_header_extension_present_flag) {	
102	**slice_segment_header_extension_length**	ue(v)
103	for(i = 0; i < slice_segment_header_extension_length; i++)	
104	**slice_segment_header_extension_data_byte**[i]	u(8)
105	}	
106	byte_alignment( )	
107	}	

[0085] 表3:HEVC范围扩展草案6中的译码单元(CU)语法表格

[0086]

1	coding_unit(x0, y0, log2CbSize) {	描述符
2	if(transquant_bypass_enabled_flag)	
3	**cu_transquant_bypass_flag**	ae(v)
4	if(slice_type != I)	
5	**cu_skip_flag**[x0][y0]	ae(v)
6	....	
7	}	

[0087] cu_transquant_bypass_flag(表3的第3行)的值指示变换和量化针对当前CU被绕过(在cu_transquant_bypass_flag被设定为1的情况下)还是变换和量化被应用于当前CU(在cu_transquant_bypass_flag被设定为0的情况下)。当PPS中的transquant_bypass_enabled_flag标记被设定为0时,另外的 CU级标志cu_transquant_bypass_flag不被用信号发送,并且被推断为0(也就是说,变换和量化被应用于当前CU)。

[0088] 当标志cu_transquant_bypass_flag针对当前CU被设定为1时,变换和量化不被应用于预测残差。相反,预测残差被直接熵译码并且与其预测模式信息、运动信息等一起封装到视频比特流中。另外,解块和样本自适应偏移 (SAO)针对当前CU而被绕过。这样,实现了块级别的无损译码,也就是说,重构的CU在数值上与原始CU相同。

[0089] 对于需要无损译码的视频应用来说,期望将无损译码应用于序列级别 (即整个序列被无损译码)或图片/片级别(即整个图片/片被无损译码)。第一版HEVC标准以及当前JCT-VC发展下的HEVC范围扩展不包括或者以其他方式提供用于指示序列/图片/片级别的无损译码的高级信令。相反,为了使用现有信令方案来实现序列/图片/片级别的无损译码可以执行以下操作:(i)创建transquant_bypass_enabled_flag被设定为1的PPS;(ii)在视频序列/图片/片的片分段报头中,参考transquant_bypass_enabled_flag被设定为1 的

PPS;(iii)对于序列/图片/片中的所有CU,将标志cu_transquant_bypass_flag 设定为1。

[0090] 仅依赖于无损模式的块级别信令具有多种缺点。特别是,其需要为序列 /图片/片中的所有CU发送cu_transquant_bypass_flag,可能效率不高。虽然上下文自适应二进制算术译码(CABAC)用于对该CU级别标志 cu_transquant_bypass_flag进行译码,其可以有效减少信令开销,但这仍然需要解码器为序列/图片/片中的所有CU解析另外的语法元素,其可能是冗余操作。

[0091] 此外,如果针对无损模式仅块级别的信令可用,则解码器可能不能为整个无损译码进行正确地准备,这是因为仍然可能诸如逆量化、逆变换、解块滤波器、SAO等之类的处理块可能对于一些将来的CU来说仍然有必要(也就是说,一些将来的CU的cu_transquant_bypass_flag的值可以设定为0)。这限制了以其他方式可从关闭不需要的处理块得到的可用的功率节约。

[0092] 类似地,当前视频比特流包含无损译码不需要的不同级别处的多种语法元素。这些元素涉及高级语法结构中的逆量化、逆变换、解块以及SAO。例如,在PPS中,一些语法元素(例如表1中第11、14、17、18行中分别显示的init_qp_minus26、cu_qp_delta_enabled_flag、pps_cb/cr_qp_offset)与逆量化过程相关;一些语法元素(例如表1中第40、43、44行中分别显示的 deblocking_filter_override_enabled_flag、pps_beta_offset_div2、pps_tc_offset_div2)与解块过程相关;并且一些语法元素(例如表1中第72、 73行中分别显示的log2_sao_offset_scale_luma、 log2_sao_offset_scale_chroma)与SAO过程先关。类似地,片分段报头中的一些语法元素涉及逆量化(表2中第74、76、77行中分别显示的slice_qp_delta、slice_cb_qp_offset、slice_cr_qp_offset)、解块(第86、87行中显示的slice_beta_offset_div2、slice_tc_offset_div2)、以及SAO(表2中第 47和49行中显示的slice_sao_luma_flag、slice_sao_chroma_flag)。这些和其他语法元素在表1-3中用星号标记。当无损译码应用于序列/图片/片级别时,可能必须用信号发送这些语法元素,其减少了信令开销。但是,在没有无损译码模式的高级指示的情况下,这些高级语法元素必须被编码到比特流中并传送给单独的解码器。

[0093] 更进一步,视频流可以包含块级别(例如CU级别)的变换相关的信令。例如,在transform_tree()语法结构(表7中显示的简化版本的transform_tree()) 中,标志被发送以指示是否执行变换单元的四叉树分离(transform_split_flag) 和/或在用于亮度和色度的变换块中是否存在任意非零系数(cbf_luma、 cbf_cb、以及cbf_cr)。在无损译码模式中,transform_tree()语法可以通过显性地绕过这些标志的信令并且相反将这些标志设置为适当的默认值而被简化。

[0094] 信令无损译码模式

[0095] 在这里描述无损译码模式中使用的信令方法的各种实施方式,其可以克服上述缺点中的一个或多个。在一个这样的实施方式中,无损译码模式的信令可以通过修改PPS语法结构来执行。表4显示了根据一些实施方式的修改后的PSS语法表,在该实施方式中,添加了另外的标志 transquant_bypass_default_flag来指示参考该PPS的片中的所有译码单元的标志cu_transquant_bypass_flag的默认值。在该实施方式中,比特流不需要在每个单独的译码单元中用信号发送cu_transquant_bypass_flag。通过将该新的标志设定为1,并且将标志transquant_bypass_enabled_flag设定为0,编码器可以向解码器指示在序列、和/

或图片、和/或片级别处应用了无损译码。也就是说，无需任意CU级别信令，绕过了参考当前PPS的所有CU的变换、变换跳过、量化、以及环路滤波过程。

[0096] 当cu_transquant_bypass_flag不存在时，可以考虑新的 transquant_bypass_default_flag来规定cu_transquant_bypass_flag的默认值。

[0097] CU级别标志cu_transquant_bypass_flag在0的现有语义可以修改如下：cu_transquant_bypass_flag等于1规定了8.6小节中规定的缩放和变换过程和 8.7小节中规定的环路滤波过程被绕过。当不存在cu_transquant_bypass_flag 时，其被推断为等于transquant_bypass_default_flag。如表4所示，可以使用新的标志transquant_bypass_default_flag(表4第11行)来限制(condition) 与逆量化、逆变换以及环路滤波过程相关的PPS中的多个语法元素的存在性 (如表4第12、15、17、21、28、42、45、57、74以及84所示)，其在表4 中用星号标记。这些限制被设定为使得这些语法元素仅在 transquant_bypass_default_flag等于0时(即在应用有损译码时)被发送。当 transquant_bypass_default_flag等于1时(即在应用无损译码时)，这些语法元素不被发送，相反，它们的值被推断为0。例如，通过当transquant_bypass_default_flag被设定为1时推断cu_qp_delta_enabled_flag的值为0，其指示变量QP相关的语法元素在CU级别不被发送，由此节约了比特并且简化了语法解析。

[0098] 此外，新的标志transquant_bypass_default_flag用于限制 transform_skip_enabled_flag的存在性。transform_skip_enabled_flag用于绕过仅变换过程(而不是量化过程)。因此，它是transquant_bypass_enabled_flag 的子集。此外，新的标志transquant_bypass_default_flag用于限制 transquant_bypass_enabled_flag的存在性。这样，当transquant_bypass_default_flag设定为1时(即无损译码模式)， transquant_bypass_enabled_flag被推断为0，并且CU级别处的 cu_transquant_bypass_flag的信号发送被跳过。

[0099] 表4:具有无损译码模式信令的PPS语法表

[0100]

1	pic_parameter_set_rbsp( ) {	描述符
2	**pps_pic_parameter_set_id**	ue(v)
3	**pps_seq_parameter_set_id**	ue(v)
4	**dependent_slice_segments_enabled_flag**	u(1)
5	**output_flag_present_flag**	u(1)
6	**num_extra_slice_header_bits**	u(3)
7	**sign_data_hiding_enabled_flag**	u(1)
8	**cabac_init_present_flag**	u(1)
9	**num_ref_idx_l0_default_active_minus1**	ue(v)
10	**num_ref_idx_l1_default_active_minus1**	ue(v)
11†	**transquant_bypass_default_flag**	u(1)
12†	if(!transquant_bypass_default_flag)	
13*	**init_qp_minus26**	se(v)
14	**constrained_intra_pred_flag**	u(1)

[0101]

15†	if(!transquant_bypass_default_flag)	
16*	**transform_skip_enabled_flag**	u(1)
17†	if(!transquant_bypass_default_flag)	
18*	**cu_qp_delta_enabled_flag**	u(1)
19	if(cu_qp_delta_enabled_flag)	
20*	**diff_cu_qp_delta_depth**	ue(v)
21†	if(!transquant_bypass_default_flag) {	
22*	**pps_cb_qp_offset**	se(v)
23*	**pps_cr_qp_offset**	se(v)
24*	**pps_slice_chroma_qp_offsets_present_flag**	u(1)
25†	}	
26	**weighted_pred_flag**	u(1)
27	**weighted_bipred_flag**	u(1)
28†	if(!transquant_bypass_default_flag)	
29	**transquant_bypass_enabled_flag**	u(1)
30	**tiles_enabled_flag**	u(1)
31	**entropy_coding_sync_enabled_flag**	u(1)
32	if(tiles_enabled_flag) {	
33	**num_tile_columns_minus1**	ue(v)
34	**num_tile_rows_minus1**	ue(v)
35	**uniform_spacing_flag**	u(1)
36	if(!uniform_spacing_flag) {	
37	for(i = 0; i < num_tile_columns_minus1; i++)	
38	**column_width_minus1**[i]	ue(v)
39	for(i = 0; i < num_tile_rows_minus1; i++)	
40	**row_height_minus1**[i]	ue(v)
41	}	
42†	if(!transquant_bypass_default_flag)	
43*	**loop_filter_across_tiles_enabled_flag**	u(1)
44	}	
45†	if(!transquant_bypass_default_flag) {	
46*	**pps_loop_filter_across_slices_enabled_flag**	u(1)
47*	**deblocking_filter_control_present_flag**	u(1)
48†	}	
49	if(deblocking_filter_control_present_flag) {	
50*	**deblocking_filter_override_enabled_flag**	u(1)
51*	**pps_deblocking_filter_disabled_flag**	u(1)
52	if(!pps_deblocking_filter_disabled_flag) {	
53*	**pps_beta_offset_div2**	se(v)
54*	**pps_tc_offset_div2**	se(v)
55	}	
56	}	
57†	if(!transquant_bypass_default_flag)	
58*	**pps_scaling_list_data_present_flag**	u(1)
59	if(pps_scaling_list_data_present_flag)	
60	scaling_list_data( )	
61	**lists_modification_present_flag**	u(1)

[0102]

62	**log2_parallel_merge_level_minus2**	ue(v)
63	**slice_segment_header_extension_present_flag**	u(1)
64	**pps_extension_present_flag**	u(1)
65	if(pps_extension_present_flag) {	
66	for(i = 0; i < 1; i++)	
67	**pps_extension_flag**[i]	u(1)
68	**pps_extension_7bits**	u(7)
69	}	
70	if(pps_extension_flag[0]) {	
71	if(transform_skip_enabled_flag)	
72*	**log2_max_transform_skip_block_size_minus2**	ue(v)
73	**cross_component_prediction_enabled_flag**	u(1)
74†	if(!transquant_bypass_default_flag)	
75*	**chroma_qp_adjustment_enabled_flag**	u(1)
76	if(chroma_qp_adjustment_enabled_flag) {	
77*	**diff_cu_chroma_qp_adjustment_depth**	ue(v)
78*	**chroma_qp_adjustment_table_size_minus1**	ue(v)
79	for(i = 0; i <= chroma_qp_adjustment_table_size_minus1; i++) {	
80*	**cb_qp_adjustment**[i]	se(v)
81*	**cr_qp_adjustment**[i]	se(v)
82	}	
83	}	
84†	if(!transquant_bypass_default_flag) {	
85*	**log2_sao_offset_scale_luma**	ue(v)
86*	**log2_sao_offset_scale_chroma**	ue(v)
87†	}	
88	}	
89	if(pps_extension_7bits)	
90	while(more_rbsp_data( ))	
91	**pps_extension_data_flag**	u(1)
92	rbsp_trailing_bits( )	
93	}	

[0103] 在另一实施方式中，为了通过HEVC范围扩展来实施提议的另外的语法元素，transquant_bypass_default_flag的位置可以进一步向下移动作为PPS扩展的一部分(即，在pps_extension_flag[0]的“if”条件内)。该布置可以保证HEVC范围扩展的PPS语法最大程度地与HEVC标准的第一版本(B. Bross、W.-J.Han,G.J.Sullivan、J.-R.Ohm、Y.K.Wang, T.Wiegand.于2013 年1月所著的文献编号为JCTVC-L1003的High Efficiency Video Coding (HEVC)text specification draft 10.Document)后向兼容。表5显示了这种布置的一个示例。在这种布置中，新的标志transquant_bypass_default_flag可以根据transquant_bypass_enabled_flag的值来限制。也就是说，transquant_bypass_default_flag仅在transquant_bypass_enabled_flag等于0时被用信号发送。当transquant_bypass_enabled_flag等于1时，transquant_bypass_default_flag不被用信号发送，并且参考该PPS的每个译码单元中的cu_transquant_bypass_flag的值在译码单元级别从比特

流被显性地接收到。但是，在该实施方式的布置中，因为该新的标志不是主PPS语法的一部分，它不用于限制如上面描述的表4中的量化、变换、变换跳过和环路滤波相关的那些现有PPS语法元素(用星号标记的语法元素)的存在性。如果transquant_bypass_default_flag为1，则用信号发送另一标志 lossless_coding_conformance_flag。如果标志lossless_coding_conformance_flag 为1，则可以应用比特流符合(conformance)要求来保证无损译码模式中未用的那些语法元素的被用信号发送的值具有适当的值。例如，在符合比特流中，如果新的标志transquant_bypass_default_flag被设定为1，则包括 cu_qp_delta_enabled_flag、pps_loop_filter_across_slices_enabled_flag、deblocking_filter_control_present_flag、loop_filter_across_tiles_enabled_flag、pps_scaling_list_data_present_flag等的语法元素的值可能需要被设定为0。这种比特流符合要求可以帮助最小化无损译码中未用的语法元素相关的信令开销。

[0104] 如上所述，在一些实施方式中，PPS用于携带新的标志 transquant_bypass_default_flag来用于无损译码指示。但是，在其他实施方式中，其他高级语法结构例如序列参数集(SPS)或视频参数集(VPS)也可以用于携带所提议的标志。可替换地，如果仅期望片级别的无损译码指示，则片分段报头可以用于携带所提议的新的标志。

[0105] 注意与量化、变换、以及环路滤波过程相关的一些语法元素可以作为 SPS的一部分而被用信号发送。这种SPS语法元素的示例包括 log2_min_transform_block_size_minus2、log2_diffmax_min_transform_block_size、max_transform_hierarchy_depth_inter、max_transform_hierarchy_depth_intra、scaling_list_enabled_flag、sample_adaptive_offset_enabled_flag、pcm_loop_filter_disabled_flag、transform_skip_rotation_enabled_flag、transform_skip_context_enabled_flag等。如果无损译码模式通过使用标志 transquant_bypass_default_flag在SPS级别或VPS级别被指示，则所提议的标志可以用于限制与量化、变换、变换跳过、变换跳过循环、以及环路滤波过程相关的SPS那些语法元素的存在性。可替换地，也可以应用类似的比特流符合要求来保证用信号为这些语法元素发送适当的值，例如在设定了符合标志的情况下，保证环路滤波被禁用。

[0106] 表5:PPS扩展中的信令transquant_bypass_default_flag

[0107]

1	pic_parameter_set_rbsp( ) {	描述符
2	**pps_pic_parameter_set_id**	ue(v)
3	**pps_seq_parameter_set_id**	ue(v)
4	…	
5	if(pps_extension_flag[0]) {	
6†	if(!transquant_bypass_enabled_flag)	
7†	**transquant_bypass_default_flag**	u(1)
8†	if (transquant_bypass_default_flag)	
9†	**lossless_coding_conformance_flag**	u(1)
10†	if(transform_skip_enabled_flag && !transquant_bypass_default_flag)	
11*	**log2_max_transform_skip_block_size_minus2**	ue(v)
12	**cross_component_prediction_enabled_flag**	u(1)
13†	if(!transquant_bypass_default_flag)	
14*	**chroma_qp_adjustment_enabled_flag**	u(1)
15	if(chroma_qp_adjustment_enabled_flag) {	
16*	**diff_cu_chroma_qp_adjustment_depth**	ue(v)
17*	**chroma_qp_adjustment_table_size_minus1**	ue(v)
18	for(i = 0; i <= chroma_qp_adjustment_table_size_minus1; i++) {	
19*	**cb_qp_adjustment**[i]	se(v)
20*	**cr_qp_adjustment**[i]	se(v)
21	}	
22	}	
23†	if(!transquant_bypass_default_flag) {	
24*	**log2_sao_offset_scale_luma**	ue(v)
25*	**log2_sao_offset_scale_chroma**	ue(v)
26†	}	
27	}	
28	…	
29	}	

[0108] 片报头信令

[0109] 在再一实施方式中,可以使用片报头信令。类似于PPS,表2中的片分段报头还包含用于变换、量化以及环路滤波处理块的多个语法元素(用星号标记的语法元素)。这些语法元素可以根据新标志 transquant_bypass_default_flag的值而被限制,并且可能在通过将 transquant_bypass_default_flag设定为1来指示无损译码时不需要被用信号发送。表6显示了一个这样的实例。

[0110] 如表2所示,片分段报头包含与量化、变换以及环路滤波过程相关的几个语法元素。这种片分段报头语法元素(其行数用星号标记)包括 slice_sao_luma_flag、slice_sao_chroma_flag、slice_qp_delta、slice_cb_qp_offset、 slice_cr_qp_offset、slice_chroma_qp_adjustment_enabled_flag、 deblocking_filter_override_flag、slice_deblocking_filter_disabled_flag、 slice_beta_offset_div2、slice_tc_offset_div2以及 slice_loop_filter_across_slices_enabled_flag。如果片级别的无损译码通过用信号发送片分段报头处的所提议的标志transquant_bypass_default_flag而被启用,则所提议的标志可以用于限制片分段报头中的与量化、变换以及环路滤波过程相关的这些语法元素的存在性。因此,在一个实施方式中,标志 transquant_bypass_default_flag放置在

片分段报头中的与量化、变换以及环路滤波相关的那些语法元素前面。在另一实施方式中，标志 transquant_bypass_default_flag放置在片分段报头中的可替换位置，例如在与量化、变换以及环路滤波相关的那些语法元素之后。在这种情况下，如果 transquant_bypass_default_flag设定为1，则比特流符合要求应当被应用以保证那些语法元素的值被适当地设定。transquant_bypass_default_flag设定为1 指示了当前片中的所有译码单元在无损模式中被译码，而无需用信号发送比特流中的每个单独的译码单元的cu_transquant_bypass_flag。另外，可以应用以下比特流符合要求：当所提议的transquant_bypass_default_flag设定为1 时，当前片参考的PPS的标志transquant_bypass_enabled_flag的值等于1。

[0111] 表6显示了当所提议的标志transquant_bypass_default_flag在片分段报头中的与量化、变换以及环路滤波相关的语法元素之前被用信号发送时，修改后的片分段报头的一个示例。注意虽然表6中的示例显示了所提议的 transquant_bypass_default_flag在片分段报头中被发送，但是作为代替，该标志可以在PPS(表5)中被用信号发送，并且可以用于限制用星号指示的片分段报头语法元素的存在性。

[0112] 表6：使用transquant_bypass_default_flag的修改后的片分段报头语法

[0113]

1	slice_segment_header( ) {	描述符
2	…	
3	if(!dependent_slice_segment_flag) {	
4	for(i = 0; i < num_extra_slice_header_bits; i++)	
5	**slice_reserved_flag**[i]	u(1)
6	**slice_type**	ue(v)
7	if(output_flag_present_flag)	
8	**pic_output_flag**	u(1)
9	if(separate_colour_plane_flag = = 1)	
10	**colour_plane_id**	u(2)
11†	**transquant_bypass_default_flag**	u(1)
12	…	
13†	if(sample_adaptive_offset_enabled_flag && !transquant_bypass_default_flag) {	
14*	**slice_sao_luma_flag**	u(1)
15	if(ChromaArrayType != 0)	
16*	**slice_sao_chroma_flag**	u(1)
17	}	
18	…	
19†	if(!transquant_bypass_default_flag)	
20*	**slice_qp_delta**	se(v)
21†	if(pps_slice_chroma_qp_offsets_present_flag && !transquant_bypass_default_flag) {	
22*	**slice_cb_qp_offset**	se(v)
23*	**slice_cr_qp_offset**	se(v)
24	}	
25†	if(chroma_qp_adjustment_enabled_flag && !transquant_bypass_default_flag)	
26*	**slice_chroma_qp_adjustment_enabled_flag**	u(1)
27†	if(deblocking_filter_override_enabled_flag && !transquant_bypass_default_flag)	
28*	**deblocking_filter_override_flag**	u(1)
29†	if(deblocking_filter_override_flag && !transquant_bypass_default_flag) {	
30*	**slice_deblocking_filter_disabled_flag**	u(1)
31	if(!slice_deblocking_filter_disabled_flag) {	
32*	**slice_beta_offset_div2**	se(v)
33*	**slice_tc_offset_div2**	se(v)
34	}	
35	}	
36†	if(pps_loop_filter_across_slices_enabled_flag && !transquant_bypass_default_flag && (slice_sao_luma_flag || slice_sao_chroma_flag || !slice_deblocking_filter_disabled_flag))	
37*	**slice_loop_filter_across_slices_enabled_flag**	u(1)
38	}	
39	…	
40	}	

[0114] 用表5作为例子,其中transquant_bypass_default_flag作为PPS扩展的一部分而被用信号发送,图12显示了一种用于在解码器侧解析修改后的PPS 扩展语法的算法的一个实施方式。在修改后的PPS扩展中,所提议的标志 transquant_bypass_default_flag被解析,并且其值在步骤1202中被检查。如果transquant_bypass_default_flag等于0,则高效视频译码(HEVC)范围扩展文本规范(草案6)中的现有PPS扩展语法元素被解析和处理(步骤1204)。如果transquant_bypass_default_flag等于1,则语法元素 chroma_qp_adjustment_enabled_flag、log2_sao_offset_scale_luma以及 log2_sao_offset_scale_chroma不被解析,并且作为代替,其值被推断为0(步骤1206)。如果标志lossless_coding_conformance_flag为1(步骤1207),则应用比特流符合要求。符合要求通过以下过程来应用:检查与量化、变换以及环路滤波相关的现有PPS语法元素(例如init_qp_minus26、cu_

qp_delta_enabled_flag、pps_cb/cr_qp_offset、cu_qp_delta_enabled_flag、pps_loop_filter_across_slices_enabled_flag、deblocking_filter_control_present_flag、loop_filter_across_tiles_enabled_flag、pps_scaling_list_data_present_flag等)的被用信号发送的值,来保证这些语法元素被正确地禁用(步骤1208)。如果一个或多个语法元素被禁用,则解码器将报告比特流符合违背(步骤1210);否则,PPS扩展的解析被正常地完成。

[0115] 变换树语法信令

[0116] 用于无损译码的变换四叉树分离

[0117] HEVC和HEVC范围扩展使用变换树分离语法来用信号发送变换单元 (TU)的大小。在一个实施方式中,高效视频译码(HEVC)范围扩展文本规范(草案6)的7.3.8.8节中规定的变换树语法不需要基于所提议的新标志 transquant_bypass_default_flag而改变。在另一实施方式中,变换树语法可以被简化以便当transquant_bypass_default_flag等于1时绕过四叉树分离标志 (split_transform_flag)和/或用信号发送用于亮度和色度分量(cbf_luma、cbf_cb、cbf_cr)的译码块标志。表7中显示了简化的transform_tree()语法。transquant_bypass_default_flag用作split_transform_flag的存在性的另外的限制。当不存在时,split_transform_flag的值对于大多数情况被推断为0(即不应用变换四叉树分离),并且对于当变换四叉树分离被强制执行时(例如当在内译码中使用NxN划分时,或者当当前CU大小大于32X32的最大变换大小时等等)的一些现有特定情况split_transform_flag的值被推断为0(即应用变换四叉树分离)。另外,如表7所示,当transquant_bypass_default_flag 等于1时,transform_tree()中的所有cbf标志(对于亮度分量和色度分量)的信号发送可以跳过;代替的是,其值可以推断为1,这是因为由于在无损译码中没有量化过程,cbf标志最可能具有非零值。

[0118] cbf_luma、cbf_cb以及cbf_cr的语义也可以修改。cbf_luma[x0][y0][trafoDepth]的值等于1可以用于规定亮度变换块包含不等于0的一个或多个变换系数级别。阵列索引x0、y0规定了考虑的相对于图片的左上角亮度样本的变换块的左上角亮度样本的位置(x0,y0)。阵列索引trafoDepth规定了译码块到块的当前细分级别,以便用于变换译码。对于对应于译码块的块来说,trafoDepth等于0。当cbf_luma[x0][y0][trafoDepth]不存在时,trafoDepth被推断为等于1。

[0119] cbf_cb[x0][y0][trafoDepth]的值等于1可以用于指示Cb变换块包含不等于0的一个或多个变换系数级别。阵列索引x0、y0规定了考虑的变换单元的左上角位置(x0,y0)。阵列索引trafoDepth规定了译码块到块的当前细分级别,以便用于变换译码。对于对应于译码块的块来说,trafoDepth等于 0。当cbf_cb[x0][y0][trafoDepth]不存在时,cbf_cb[x0][y0][trafoDepth]的值被推断如下。

[0120] • 如果transquant_bypass_default_flag等于1,则cbf_cb[x0][y0][trafoDepth] 被推断为等于1,

[0121] • 否则,如果trafoDepth大于0并且log2TrafoSize等于2,则 cbf_cb[x0][y0][trafoDepth]被推断为等于cbf_cb[xBase][yBase][trafoDepth-1],

[0122] • 否则,cbf_cb[x0][y0][trafoDepth]被推断为等于0。

[0123] cbf_cr[x0][y0][trafoDepth]的值等于1规定了Cr变换块包含不等于0的一个或

多个变换系数级别。阵列索引x0,y0规定了考虑的变换单元的左上角位置(x0,y0)。阵列索引trafoDepth规定了译码块到块的当前细分级别,以便用于变换译码。对于对应于译码块的块来说,trafoDepth等于0。

[0124] 当cbf_cr[x0][y0][trafoDepth]不存在时,cbf_cr[x0][y0][trafoDepth]的值被推断如下:

[0125] • 如果transquant_bypass_default_flag等于1,则cbf_cr[x0][y0][trafoDepth] 被推断为等于1,

[0126] • 否则,如果trafoDepth大于0并且log2TrafoSize等于2,则 cbf_cr[x0][y0][trafoDepth]被推断为等于cbf_cr[xBase][yBase][trafoDepth-1],

[0127] • 否则,cbf_cr[x0][y0][trafoDepth]被推断为等于0。

[0128] 应当注意,虽然表7显示了使用所提议的高级别标志 transquant_bypass_default_flag来简化transform_tree()语法的示例,但是现有的块级别标志cu_tranquant_bypass_flag可以作为代替用于通过遵循相同的逻辑来限制split_transform_flag、cbf_luma、cbf_cb以及cbf_cr的存在性。

[0129] 表7:简化的变换树语法

[0130]

1	transform_tree(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx) {	**描述符**
2†	if(!transquant_bypass_default_flag && log2TrafoSize <= Log2MaxTrafoSize && log2TrafoSize > Log2MinTrafoSize && trafoDepth < MaxTrafoDepth && !(IntraSplitFlag && (trafoDepth = = 0)))	
3	**split_transform_flag**[x0][y0][trafoDepth]	ae(v)
4†	if(!transquant_bypass_default_flag && log2TrafoSize > 2 && ChromaArrayType != 0) {	
5	if(trafoDepth = = 0 || cbf_cb[xBase][yBase][trafoDepth − 1]) {	
6	**cbf_cb**[x0][y0][trafoDepth]	ae(v)
7	if(ChromaArrayType = = 2 && ! split_transform_flag[x0][y0][trafoDepth])	
8	**cbf_cb**[x0][y0 + (1 << (log2TrafoSize − 1))][trafoDepth]	
9	}	
10	if(trafoDepth = = 0 || cbf_cr[xBase][yBase][trafoDepth − 1]) {	
11	**cbf_cr**[x0][y0][trafoDepth]	ae(v)
12	if(ChromaArrayType = = 2 && ! split_transform_flag[x0][y0][trafoDepth])	
13	**cbf_cr**[x0][y0 + (1 << (log2TrafoSize − 1))][trafoDepth]	
14	}	
15	}	
16	if(split_transform_flag[x0][y0][trafoDepth]) {	
17	x1 = x0 + (1 << (log2TrafoSize − 1))	
18	y1 = y0 + (1 << (log2TrafoSize − 1))	
19	transform_tree(x0, y0, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 0)	
20	transform_tree(x1, y0, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 1)	
21	transform_tree(x0, y1, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 2)	
22	transform_tree(x1, y1, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 3)	
23	} else {	
24†	if(!transquant_bypass_default_flag && ((CuPredMode[x0][y0] = = MODE_INTRA && intra_bc_flag[x0][y0] != 1) || trafoDepth != 0 || cbf_cb[x0][y0][trafoDepth] || cbf_cr[x0][y0][trafoDepth] || (ChromaArrayType = = 2 && (cbf_cb[x0][y0 + (1 << (log2TrafoSize − 1))][trafoDepth] || cbf_cr[x0][y0 + (1 << (log2TrafoSize − 1))][trafoDepth])))	
25	**cbf_luma**[x0][y0][trafoDepth]	ae(v)
26	transform_unit(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx)	
27	}	
28	}	

[0131] 在表7显示的一实施方式中，对于具有不同大小并且用不同译码模式译码的CU来说，当标志transquant_bypass_default_flag等于1时，四叉树分离标志split_transform_flag和译码块标志cbf_luma、cbf_cb以及cbf_cr被绕过并且被推断为相对应的默认值。但是，根据输入视频的特性，具有不同块大小和译码模式的CU的残差可以呈现出不同的统计特性。在这种情况下，它可能不能有利于禁用一个图片或序列中的所有SU的变换四叉树分离。代替的是，为了改进译码性能，作为本发明的一个实施方式，所提议的 transquant_bypass_default_flag用于根据要译码的CU的块大小、译码模式(即间、内或IBC)、或块大小与译码模式的组合来有条件地绕过变换四叉树分离标志和/或译码块标志的信令。例如，根据复杂性与性能的折衷，它可以有利于在块通过使用非内模式而被译码(即通过使用间或IBC模式而被译码) 并且块大小为8x8或16x16的情况下，仅允许变换四叉树分离。

[0132] 在一个实施方式中，以下过程适用于在标志 transquant_bypass_default_flag等于1时。对于所有内译码的CU，以及块大小从64x64到32x32的所有间译码和IBC译码的CU，split_transform_flag 和/或译码块标志(cbf_luma、cbf_cb以及cbf_cr)不被用信号发送；它们被推断为如上文所探讨的相对应的默认值。对于用间模式或IBC模式译码的 8x8和16x16CU，可以允许进一步的分离。split_transform_flag和/或译码块标志(cbf_luma、cbf_cb以及cbf_cr)仍然被用信号发送以便分别指示当前块是否被进一步划分为四象限(quadrant)和/或一个TU中的系数是否为全0。

[0133] 在一些实施方式中，可以添加两个语法元素 log2_intra_max_no_transform_split_coding_block_size_minus3和 log2_inter_max_no_transform_split_coding_block_size_minus3到SPS或PPS 来规定最大CU大小，针对该最大CU大小，变换四叉树分离分别被应用于内和间/IBC译码的CU。例如通过使用上面的限制1)和2)， log2_intra_max_no_transform_split_coding_block_size_minus3和 log2_inter_max_no_transform_split_coding_block_size_minus3分别设定为 log2(64)-3＝3和log2(16)-3＝1。表8显示了具有两个提议的语法元素的修改后的SPS屏幕内容译码扩展语法表。

[0134] 虽然表8显示了两个另外的语法元素，但是在本发明的另一实施方式中，可以不用信号发送 log2_intra_max_no_transform_split_coding_block_size_minus3的值。代替的是，该值可以被推断为一直与能通过增大最小CU大小(如SPS中由语法元素log2_min_luma_coding_block_size_minus3所规定的)获得的允许的最大 CU大小相同，差值在最大与最小CU大小(如SPS中由语法元素 log2_diff_max_min_luma_coding_block_size所规定的)之间。在该实施方式中，当CU在内模式中被译码并且无损译码被应用时，不允许变换四叉树分离。

[0135] 表8:序列参数集屏幕内容译码语法

[0136]

1	sps_scc_extensions( ) {	描述符
2	**intra_block_copy_enabled_flag**	u(1)
3	**palette_mode_enabled_flag**	u(1)
4	if(palette_mode_enabled_flag) {	
5	**palette_max_size**	ue(v)
6	**palette_max_predictor_size**	ue(v)
7	}	
8	**adaptive_mv_resolution_enabled_flag**	u(1)
9	**intra_boundary_filtering_disabled_flag**	u(1)
10†	**log2_intra_max_no_transform_partition_coding_block_size_minus3**	ue(v)
11†	**log2_inter_max_no_transform_partition_coding_block_size_minus3**	ue(v)
12	}	

[0137] log2_intra_max_no_transform_partition_coding_block_size_minus3的值加上3规定了当译码单元被内译码并且当cu_transquant_bypass_flag等于1时应用变换四叉树分离的译码单元的最大块大小。

[0138] log2_intra_max_no_transform_partition_coding_block_size_minus3的值加上3规定了当译码单元被间译码并且cu_transquant_bypass_flag等于1时应用于变换四叉树分离的译码单元的最大块大小。

[0139] 表9显示了修改的transform_tree()语法表格,其具有根据当前CU的块大小和译码模式限制的split_transform_flag、cbf_luma、cbf_cb以及cbf_cr 的所提议的信令约束。注意虽然该示例性实施方式使用CU译码模式和CU 大小来限制是否允许split_transform_flag和cbf信令,但是可以使用其他修改限制。例如,可以使用译码模式和块大小中的任一者(但不是二者)。另外,独立(并且不同)的限制可以应用于split_transform_flag信令或cbf信令。

[0140] 表9:修改后的变换树语法

[0141]

1	transform_tree(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx) {	**描述符**
2†	if(!cu_transquant_bypass_flag || ((CuPredMode[x0][y0] == MODE_INTRA && log2TrafoSize <= log2_intra_max_no_transform_partition_coding_unit_size_minus3+3) || (CuPredMode[x0][y0] != MODE_INTRA && log2TrafoSize <= log2_inter_max_no_transform_partition_coding_unit_size_minus3+3)) && log2TrafoSize <= Log2MaxTrafoSize && log2TrafoSize > Log2MinTrafoSize && trafoDepth < MaxTrafoDepth && !(IntraSplitFlag && (trafoDepth = = 0)))	
3	**split_transform_flag**[x0][y0][trafoDepth]	ae(v)
4†	if((!cu_transquant_bypass_flag || ((CuPredMode[x0][y0] == MODE_INTRA && log2TrafoSize <= log2_intra_max_no_transform_partition_coding_unit_size_minus3+3) || (CuPredMode[x0][y0] != MODE_INTRA && log2TrafoSize <= log2_inter_max_no_transform_partition_coding_unit_size_minus3+3)) && log2TrafoSize > 2 && ChromaArrayType != 0) {	
5	if(trafoDepth = = 0 || cbf_cb[xBase][yBase][trafoDepth − 1]) {	
6	**cbf_cb**[x0][y0][trafoDepth]	ae(v)
7	if(ChromaArrayType = = 2 && ! split_transform_flag[x0][y0][trafoDepth])	
8	**cbf_cb**[x0][y0 + (1 << (log2TrafoSize − 1))][trafoDepth]	
9	}	
10	if(trafoDepth = = 0 || cbf_cr[xBase][yBase][trafoDepth − 1]) {	
11	**cbf_cr**[x0][y0][trafoDepth]	ae(v)
12	if(ChromaArrayType = = 2 && ! split_transform_flag[x0][y0][trafoDepth])	
13	**cbf_cr**[x0][y0 + (1 << (log2TrafoSize − 1))][trafoDepth]	
14	}	
15	}	
16	if(split_transform_flag[x0][y0][trafoDepth]) {	
17	x1 = x0 + (1 << (log2TrafoSize − 1))	
18	y1 = y0 + (1 << (log2TrafoSize − 1))	
19	transform_tree(x0, y0, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 0)	
20	transform_tree(x1, y0, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 1)	
21	transform_tree(x0, y1, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 2)	
22	transform_tree(x1, y1, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 3)	
23	} else {	
24†	if((!cu_transquant_bypass_flag || ((CuPredMode[x0][y0] == MODE_INTRA && log2TrafoSize <= log2_intra_max_no_transform_partition_coding_unit_size_minus3+3) || (CuPredMode[x0][y0] != MODE_INTRA && log2TrafoSize <= log2_inter_max_no_transform_partition_coding_unit_size_minus3+3)) && ((CuPredMode[x0][y0] = = MODE_INTRA && intra_bc_flag[x0][y0] != 1) || trafoDepth != 0 || cbf_cb[x0][y0][trafoDepth] || cbf_cr[x0][y0][trafoDepth] || (ChromaArrayType = = 2 && (cbf_cb[x0][y0 + (1 << (log2TrafoSize − 1))][trafoDepth] || cbf_cr[x0][y0 + (1 << (log2TrafoSize − 1))][trafoDepth]))))	
25	**cbf_luma**[x0][y0][trafoDepth]	ae(v)
26	transform_unit(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx)	

[0142]

27	}	
28	}	

[0143] 在另一实施方式中,为了跳过无损情况下的split_transform_flag的信令，SPS中的语法元素max_transform_hierarchy_depth_inter和 max_transform_hierarchy_depth_intra可以设定为0。该方法在序列级别无损译码的情况下不需要对当前的transform_tree()语法进行低级别改变并且绕过了transform_skip_flag的信令。在这种实施方式中,split_transform_flag不被用信号发送,并且代替的是,被推断为默认值,其在大多数情况下为0,除非TU四叉树分离被强制(例如,NxN划分被用于内译码,或者当CU大小大于最大TU大小时,等等)。因此,当应用序列级别无损译码时,通过施加要求max_transform_hierarchy_depth_inter和 max_transform_hierarchy_depth_intra被适当地设定为0的比特流约束， transform_split_flag的信令可以被绕过,而无需任何块级别的

改变。类似地，通过施加最大变换大小和最小变换大小在序列级别无损译码的情况下必须相同的比特流约束，transform_split_flag的信令可以被绕过，而无需任何块级别的改变。该约束在序列级别无损译码的情况下能够通过要求SPS语法 log2_diff_max_min_transform_block_size设定为0来实现。由于 max_transform_hierarchy_depth_inter、max_transform_hierarchy_depth_intra以及log2_diff_max_min_transform_block_size位于SPS中，因此更优选的可以是在这种情况中也将transquant_bypass_default_flag放置在SPS中。

[0144] 在本发明的另一实施方式中，提议了仅编码器方法来用于进行无损译码，而无需添加标志transquant_bypass_default_flag。在这种实施方式中，使用transquant_bypass_default_flag的限制术语可以从用剑号(†)标记的表7 和表9中的语法元素省略。该实施方式不需要修改SPS中的语法元素 max_transform_hierarchy_depth_inter和max_transform_hierarchy_depth_intra 或transform_tree()中的语法元素的值。为了减小编码复杂性，对于仅编码器方法仍用信号发送标志split_tranform_flag，但是仅由如表7或表9中所述的 split_tranform_flag的默认值指示的变换四叉树分离针对每个CU而被测试。

[0145] 在一些实施方式中，当对于当前CU来说标志cu_transquant_bypass_flag 等于1时，编码器将会针对大多数情况仅测试无变换四叉树划分的率失真 (R-D)性能，并且对于一些特别情况(例如当NxN个PU划分被应用于内译码CU或者当前CU大小大于阈值时)仅测试一次变换四叉树划分的R-D 性能。这样，该仅编码器方法与应用于用剑号(†)标记的语法元素中的表7 中的标志split_tranform_flag的限制一致。

[0146] 在另一实施方式中，当标志cu_transquant_bypass_flag等于1时，当CU 精确地包含一个预测单元时，编码器将会对于块大小从64x64到32x32的所有内译码CU和所有间/IBC译码的CU仅测试无变换四叉树划分的R-D性能；当CU包含至少两个预测单元时，编码器将会对于块大小从64x64到 32x32的所有内译码CU和所有间/IBC译码的CU仅测试一次变换四叉树划分的R-D性能；对于块大小16x16和8x8的间/IBC译码的CU，编码器测试无变换四叉树分离和进一步的变换四叉树分离的R-D性能。因此这种仅编码器方法与应用于表9中用剑号(†)标记的标志split_tranform_flag的限制一致。另外，虽然对于上述仅编码器方法，标志split_tranform_flag仍在比特流中被用信号发送，但是可以应用一个比特流符合约束以便要求语法元素 split_tranform_flag的值根据块大小和块译码模式而被设定为其默认值。

[0147] 在一些实施方式中，变换四叉树分离标志的默认值的确定是至少部分基于相关译码单元中的预测单元的数量的。在一个这样的实施方式中，默认值的确定包括确定译码单元是否被内译码、译码单元的大小是否大于大小阈值、以及译码单元是否精确地包含一个预测单元。响应于确定译码单元不被内译码、译码单元大于大小阈值、以及译码单元精确地包含一个预测单元，变换四叉树分离标志的默认值被设定为指示无变换四叉树划分。

[0148] 在另一这样的实施方式中，变换四叉树分离标志的默认值的确定包括确定相关译码单元是否被内译码、译码单元的大小是否大于大小阈值、以及第一译码单元是否包含至少两个预测单元。响应于确定译码单元不被内译码、译码单元大于大小阈值、以及译码单元

包含至少两个预测单元,变换四叉树分离标志的默认值被设定为指示一次变换四叉树划分。

[0149] 在进一步的实施方式中,变换四叉树分离标志的默认值的确定包括确定相关译码单元是否被内译码、以及第一译码单元是否精确地包含一个预测单元。响应于确定译码单元被内译码以及精确地包含一个预测单元,变换四叉树分离标志的默认值被设定为指示无变换四叉树划分。

[0150] 在另一实施方式中,确定变换四叉树分离标志的默认值包括确定相关译码单元是否被内译码以及译码单元是否包含至少两个预测单元。响应于确定译码单元不被内译码以及译码单元包含至少两个预测单元,变换四叉树分离标志的默认值被设定为指示一次变换四叉树划分。

[0151] 译码单元是精确地包括一个预测单元还是包括至少两个预测单元的确定可以通过确定译码单元的划分模式来执行。例如,使用2Nx2N划分模式的译码单元精确地包括一个预测单元,而使用2NxN、Nx2N、2NxnU、2NxnD、 nLx2N、nRx2N或NxN划分模式的译码单元举例来说包含至少两个预测单元。

[0152] 用于内块复制模式的变换树分离

[0153] 变换树分离的最大深度与编码和解码复杂度密切相关。为了提供译码效率与计算复杂度之间的灵活的折衷,HEVC与其扩展使用SPS中的语法元素来规定TU大小和TU分离深度。值 log2_min_luma_transform_block_size_minus2和log2_diff_max_min_luma_transform_block_size指示用于对视频序列进行译码的TU大小的设置,并且max_transform_hierarchy_depth_inter和 max_transform_hierarchy_depth_intra分别为内和间译码CU指示最大分离深度。在某些限制下,不可以应用变换四叉树分离。例如,如果max_transform_hierarchy_depth_intra/inter设定为0,则变换四叉树不被应用于当前内/间译码的CU。

[0154] 在HEVC、其范围扩展、以及SCC草案中,当禁用变换四叉树分离时,一个隐性的TU划分方法应用于有损和无损译码,从而使得 split_transform_flag的值对于被划分为多个预测单元(PU)且在间模式中被译码的CU来说一直被推断为1(即变换四叉树分离被应用)。这是因为CU 内的PU的不同运动矢量可能引起虚假的高频信息并且导致跨越邻近PU之间的边界的不一致残差。在这种情况下,将CU分离成较小的TU可以比使用与CU大小一样大的TU大小提供更好的译码效率。

[0155] 在HEVC屏幕内容译码扩展[0057]的工作草案中,上述隐性TU划分不应用于IBC模式中译码的CU。更特别地,当变换四叉树分离被禁用时, split_transform_flag的值对于对于所有IBC译码的CU来说一直被推断为0 (即,变换单元大小被被设置为与CU的大小相同)。假定IBC模式与间模式之间的固有相似性,IBC译码的CU的残差可以呈现与间译码的CU相似的特征。因此,它可能也有利于将隐性TU划分(即推断split_transform_flag 的值为1)应用于IBC译码的CU,以便进一步改进变换译码的效率。在这里描述的实施方式中,当变换四叉树划分被禁用时,应用于间模式的相同隐性TU划分方法也用于有损译码和无损译码中的IBC译码的CU。换句话说,当多于一个PU划分(例如2NxN、Nx2N以及NxN)存在于用IBC模式译码的当前CU中时,split_transform_flag的值被推断为1(即,变换四叉树被分离)。

[0156] 在[0057]的7.4.9.9节中规定了split_transform_flag的值的获取过程。在这里

公开的示例性实施方式中,通过对于IBC模式启用隐性的TU划分, split_transform_flag的语义操作如下。

[0157] 阵列split_transform_flag[x0][y0][trafoDepth]规定了块是否被分离成半水平且半垂直大小的四个块,以便进行变换译码。阵列索引x0,y0规定了所考虑的块的左上角亮度样本相对于图片的左上角亮度样本的位置(x0,y0)。阵列索引trafoDepth规定了译码块到块的当前细分级别,以便进行变换译码。对于对应于译码块的块,trafoDepth的值等于0。

[0158] 变量interSplitFlag的获取如下。在以下限制中的一个或多个限制适用的情况下,interSplitFlag被设定为等于1:max_transform_hierarchy_depth_inter 等于0并且CuPredMode[x0][y0]等于MODE_INTER;或者 intra_bc_flag[x0][y0]等于1,并且PartMode不等于PART_2Nx2N且trafoDepth 等于0。否则,interSplitFlag被设定等于0。

[0159] 当split_transform_flag[x0][y0][trafoDepth]不存在时,其值推断如下。如果以下限制中的一个或多个限制为真,则split_transform_flag[x0][y0][trafoDepth]的值被推断为等于1:log2TrafoSize 大于MaxTbLog2SizeY;IntraSplitFlag等于1并且trafoDepth等于0;或者 interSplitFlag等于1。否则,split_transform_flag[x0][y0][trafoDepth]的值被推断为等于0。

[0160] 实施方式

[0161] 在一个示例性的实施方式中,提供了一种用于对包括片分段报头和多个译码单元的视频片进行译码的方法。该方法包括在片分段报头中生成绕过标志,该绕过标志指示是否片中的所有译码单元都不被用无损译码进行译码。

[0162] 在一些这样的实施方式中,绕过标志是transquant_bypass_default_flag。所述方法可以包括生成图片参数集(PPS),该PPS包括 transquant_bypass_enabled_flag,其中片参考图片参数集,并且其中当 transquant_bypass_default_flag设定为一时,transquant_bypass_enabled_flag 被设定为零。

[0163] 在一些实施方式中,绕过标志指示不是片中的所有译码单元都被用无损译码进行译码,并且所述方法还包括在片分段报头中生成与无损译码相关的语法元素。绕过标志可以放置在与无损译码相关的语法元素的前面。

[0164] 在一些实施方式中,其中绕过标志指示不是片中的所有译码单元都被用无损译码进行译码,所述方法还包括在片分段报头中生成与量化、变换、以及环路滤波过程相关的语法元素。绕过标志可以放置在与量化、变换、以及环路滤波过程相关的语法元素的前面。

[0165] 在一些实施方式中,其中绕过标志指示不是片中的所有译码单元都被用无损译码进行译码,所述方法还包括在片分段报头中生成从由以下语法元素组成的群组中选择的一个或多个语法元素:slice_sao_luma_flag、slice_sao_chroma_flag、slice_qp_delta、slice_cb_qp_offset、slice_cr_qp_offset、slice_chroma_qp_adjustment_enabled_flag、deblocking_filter_override_flag、slice_deblocking_filter_disabled_flag、slice_beta_offset_div2、slice_tc_offset_div2以及slice_loop_filter_across_slices_enabled_flag。绕过标志可以放置在从由以下语法元素组成的群组中选择的一个或多个语法元素前面:slice_sao_luma_flag、slice_sao_chroma_flag、slice_qp_delta、slice_cb_qp_offset、slice_cr_qp_offset、slice_chroma_qp_adjustment_enabled_

flag、deblocking_filter_override_flag、slice_deblocking_filter_disabled_flag、slice_beta_offset_div2、slice_tc_offset_div2以及slice_loop_filter_across_slices_enabled_flag。

[0166] 在一些实施方式中，绕过标志指示片中的所有译码单元都被用无损译码进行译码，并且所述方法还包括从片分段报头中排除与无损译码相关的语法元素。

[0167] 在一些实施方式中，其中绕过标志指示片中的所有译码单元都被用无损译码进行译码，所述方法还包括从片分段报头中排除与量化、变换以及环路滤波过程相关的语法元素。

[0168] 在一些实施方式中，其中绕过标志指示片中的所有译码单元都被用无损译码进行译码，所述方法还包括从片分段报头中排除从由以下语法元素组成的群组中选择的一个或多个语法元素：slice_sao_luma_flag、slice_sao_chroma_flag、slice_qp_delta、slice_cb_qp_offset、slice_cr_qp_offset、slice_chroma_qp_adjustment_enabled_flag、deblocking_filter_override_flag、slice_deblocking_filter_disabled_flag、slice_beta_offset_div2、slice_tc_offset_div2以及slice_loop_filter_across_slices_enabled_flag。

[0169] 在一些实施方式中，其中绕过标志指示不是片中的所有译码单元都被用无损译码进行译码，所述方法还包括为片中的每个译码单元用信号发送 cu_transquant_bypass_flag。

[0170] 在一些实施方式中，其中绕过标志指示片中的所有译码单元都被用无损译码进行译码，所述方法还包括从片中的每个译码单元排除 cu_transquant_bypass_flag。

[0171] 在一个示例性的实施方式中，提供了一种用于对包括图片参数集和参考图片参数集(PPS)的至少一个片的视频进行译码的方法，其中片包括片分段报头和多个译码单元。在该实施方式中，所述方法包括在图片参数集中生成绕过标志，该绕过标志指示是否参考图片参数集的片中的所有译码单元都被用无损译码进行译码。

[0172] 在一些这样的实施方式中，绕过标志是transquant_bypass_default_flag。所述图片参数集(PPS)包括transquant_bypass_enabled_flag，并且当transquant_bypass_default_flag设定为一时，transquant_bypass_enabled_flag 被设定为零。

[0173] 在一些实施方式中，其中绕过标志指示不是片中的所有译码单元都被用无损译码进行译码，所述方法还包括在片分段报头中生成与有损译码相关的语法元素。

[0174] 在一些实施方式中，其中绕过标志指示不是片中的所有译码单元都被用无损译码进行译码，所述方法还包括在片分段报头中生成与量化、变换、以及环路滤波过程相关的语法元素。

[0175] 在一些实施方式中，其中绕过标志指示不是片中的所有译码单元都被用无损译码进行译码，所述方法还包括在片分段报头中生成从由以下语法元素组成的群组中选择的一个或多个语法元素：slice_sao_luma_flag、slice_sao_chroma_flag、slice_qp_delta、slice_cb_qp_offset、slice_cr_qp_offset、slice_chroma_qp_adjustment_enabled_flag、deblocking_filter_override_flag、slice_deblocking_filter_disabled_flag、slice_beta_offset_div2、slice_tc_offset_div2以及slice_loop_filter_across_slices_enabled_flag。

[0176] 在一些实施方式中,其中绕过标志指示片中的所有译码单元都被用无损译码进行译码,所述方法还包括从片分段报头中排除与有损译码相关的语法元素。

[0177] 在一些实施方式中,其中绕过标志指示片中的所有译码单元都被用无损译码进行译码,所述方法还包括从片分段报头中排除与量化、变换以及环路滤波过程相关的语法元素。

[0178] 在一些实施方式中,其中绕过标志指示片中的所有译码单元都被用无损译码进行译码,所述方法包括从片分段报头中排除从由以下语法元素组成的群组中选择的一个或多个语法元素:slice_sao_luma_flag、slice_sao_chroma_flag、slice_qp_delta、slice_cb_qp_offset、slice_cr_qp_offset、slice_chroma_qp_adjustment_enabled_flag、deblocking_filter_override_flag、slice_deblocking_filter_disabled_flag、slice_beta_offset_div2、slice_tc_offset_div2以及slice_loop_filter_across_slices_enabled_flag。

[0179] 在一些实施方式中,其中绕过标志指示不是片中的所有译码单元都被用无损译码进行译码,所述方法还包括为片中的每个译码单元用信号发送 cu_transquant_bypass_flag。

[0180] 在一些实施方式中,其中绕过标志指示片中的所有译码单元都被用无损译码进行译码,所述方法还包括从片中的每个译码单元排除 cu_transquant_bypass_flag。

[0181] 在一个示例性的实施方式中,提供了一种用于对视频进行译码的方法,其中该视频包括高级语法结构和参考所述高级语法结构的至少一个片,所述片包括多个译码单元。所述方法包括在高级语法结构中生成绕过标志,该绕过标志指示是否片中的所有译码单元都被用无损译码进行译码。对于每个单独的译码单元,确定是否生成四叉树分离标志。该确定是至少部分基于从由单独的译码单元的块大小和译码模式组成的群组中选择的参数来进行的。四叉树分离标志是仅在确定生成四叉树分离标志之后针对单独的译码单元生成的。

[0182] 在这样的实施方式中,高级语法结构可以是图片参数集、分段片报头、或split_transform_flag。在一些这样的实施方式中,四叉树分离标志仅在以下情况下生成:块是使用非内模式译码的、并且块大小是8x8或16x16。四叉树分离标志在块大小为从64x64到32x32的情况下不被生成。

[0183] 一些实施方式还涉及生成指示最大无变换四叉树分离块译码大小的高级语法元素。在一些实施方式中,指示最大无变换四叉树分离块译码大小的高级语法元素在序列参数集(SPS)中被生成。在一些实施方式中,指示最大无变换四叉树分离块译码大小的高级语法元素在图片参数集(PPS)中被生成。

[0184] 在一些实施方式中,高级语法元素用于指示最大无变换四叉树分离块译码大小,该最大无变换四叉树分离块译码大小为在非内模式中译码的块指示最大无变换四叉树分离块译码大小。

[0185] 在一些实施方式中,如果块被使用非内模式译码并且块大小不大于针对非内模式中译码的块的最大无变换四叉树分离块译码大小,则生成四叉树分离标志。

[0186] 在一些实施方式中,指示最大无变换四叉树分离块译码大小的高级语法元素为在内模式中译码的块指示最大无变换四叉树分离块译码大小。

[0187] 在一些实施方式中，如果块被使用内模式译码并且块大小不大于针对内模式中译码的块的最大无变换四叉树分离块译码大小，则生成四叉树分离标志。

[0188] 在一示例性的实施方式中，提供了一种用于对视频进行译码的方法，其中该视频包括高级语法结构和参考所述高级语法结构的至少一个片，所述片包括多个译码单元。所述方法包括在高级语法结构中生成绕过标志，该绕过标志指示是否片中的所有译码单元都被用无损译码进行译码。对于每个单独的译码单元，至少部分基于从由单独的译码单元的块大小和译码模式组成的群组中选择的参数来确定是否生成译码块标志。译码块标志是仅在确定生成译码块标志之后针对单独的译码单元生成的。

[0189] 在一这样的实施方式中，高级语法结构可以是图片参数集或分段片报头。译码块标志可以是cbf_luma标志、cbf_cb标志或cbf_cr标志中的一者或多者。在一些实施方式中，译码块标志仅在以下情况下生成：块是使用非内模式译码的并且块大小是8x8或16x16；然而译码块标志在块被使用内模式译码的情况下或者块大小为从64x64到32x32的情况下不被生成。

[0190] 在一些实施方式中，生成指示最大无变换四叉树分离块译码大小的高级语法元素。指示最大无变换四叉树分离块译码大小的高级语法元素在序列参数集(SPS)或图片参数集(PPS)中被生成。

[0191] 在一些实施方式中，高级语法元素指示最大无变换四叉树分离块译码大小，该最大无变换四叉树分离块译码大小为在非内模式中译码的块指示最大无变换四叉树分离块译码大小。如果块被使用非内模式译码并且块大小不大于针对间模式中译码的块的最大无变换四叉树分离块译码大小，则生成译码块标志。

[0192] 在一些实施方式中，指示最大无变换四叉树分离块译码大小的高级语法元素为在内模式中译码的块指示最大无变换四叉树分离块译码大小。

[0193] 在一些实施方式中，如果块被使用内模式译码并且块大小不大于针对内模式中译码的块的最大无变换四叉树分离块译码大小，则生成译码块标志。

[0194] 在一示例性的实施方式中，提供了一种用于对视频进行译码的方法，其中该视频包括高级语法结构和参考所述高级语法结构的至少一个片，所述片包括多个译码单元。所述方法包括为译码单元确定标志 cu_transquant_bypass_flag等于1。确定是基于译码单元大小和译码单元的译码模式中的至少一者来执行的，以便测试一次变换四叉树划分的率失真性能。在该确定之后，一次变换四叉树划分的率失真性能被测试。

[0195] 在一些这样的实施方式中，所述方法还包括：对于另外的译码单元，确定不测试一次变换四叉树划分的率失真性能。率失真性能的测试仅在对于另外的译码单元来说无变换四叉树划分的情况下被执行。

[0196] 在一示例性实施方式中，提供了一种用于对视频进行译码的方法，其中视频包括变换树语法并且还包括多个译码单元，所述变换树语法包括分离变换标志。所述方法包括为单独的译码单元确定译码单元是否在内块复制模式中被译码以及在译码单元中是否存在多于一个预测单元划分。响应于确定译码单元在内块复制模式中被译码并且在译码单元中存在多于一个预测单元划分，分离变换标志的值被推断为1。

[0197] 在一示例性方法中，生成包括针对译码单元标志的默认值的高级语法结构，该默认值指示变换和量化过程被绕过。

[0198] 在一些这样的实施方式中,所述默认值transquant_bypass_default_flag 设定为1,指示参考高级语法结构的片的所有译码单元的 cu_transquant_bypass_flag的默认值。高级语法结构可以是图片参数集(PPS)、序列参数集(SPS)、视频参数集(VPS)或片分段报头中的至少一者。

[0199] 在一些实施方式中,所述方法包括生成包括具有等于0的 transquant_bypass_default_flag的PPS的比特流。在一些实施方式中,所述方法包括生成比特流,在该比特流中译码单元参数不包括 cu_transquant_bypass_flag。

[0200] 在一示例性实施方式中,提供了一种用于经由特定高级语法结构中的高级语法元素向解码器发送信号以便绕过针对参考该特定高级语法结构的所有CU的变换、变换跳过、量化、以及环路滤波过程的方法。

[0201] 在另一示例性实施方式中,提供了一种操作解码器来接收和处理高级语法元素以便标识与逆量化、逆变换、以及环路滤波过程中的任意一者或多着相关的多个PPS语法元素的存在性。

[0202] 在一些这样的方法中,高级语法元素是默认标志值 (transquant_bypass_default_flag)。在一些这样的方法中,高级语法元素用于标识transform_skip_enabled_flag元素的存在性。

[0203] 在一些实施方式中,所述方法包括通过推断来确定 transquant_bypass_enabled_flag为0,并且作为响应,去除或跳过CU级别处的cu_transquant_bypass_flag的信令。在一些实施方式中,默认标志包含在 PPS扩展参数集、SPS扩展参数集中的至少一者中。

[0204] 在一些实施方式中,默认标志的存在性取决于transquant_bypass_enabled_flag的值。在一些实施方式中,默认标志是 transquant_bypass_default_flag。

[0205] 在一些实施方式中,所述方法还包括接收另外的符合标志的信令来指示无损译码模式中未用的语法元素的被用信号发送的值被适当地设定。

[0206] 在一些实施方式中,如果标志transquant_bypass_default_flag设定为1,则包括cu_qp_delta_enabled_flag、pps_loop_filter_across_slices_enabled_flag、deblocking_filter_control_present_flag、loop_filter_across_tiles_enabled_flag、pps_scaling_list_data_present_flag的语法元素设定为0。

[0207] 在一示例性实施方式中,提供了一种接收包含指示使用了无损译码的高级信令无损译码语法元素的视频数据比特流的方法。

[0208] 在一些这样的方法中,高级信令语法是图片参数集(PPS)、序列参数集 (SPS)、视频参数集(VPS)或片分段报头中的一者。在一些这样的方法中,无损译码语法元素用作用于提出与量化、变换、变换跳过、变换跳过循环以及环路滤波过程相关的一个或多个SPS语法元素的限制。

[0209] 在一示例性实施方式中,片分段报头被接收,并且默认标志用于限制用于变换、量化以及环路滤波处理块的片分段语法元素的标识。在一些这样的实施方式中,默认标志是transquant_bypass_default_flag。

[0210] 在一示例性实施方式中,在视频解码器处执行一种用于接收高级无损译码指示并且作为响应关闭多个处理块的方法。在一些这样的实施方式中,高级无损译码指示是PPS、

SPS、VPS、或片报头中的一者的参数元素。在一些实施方式中，多个处理块包括以下硬件块中的任意一者或多者：逆量化、逆变换、解块滤波器、SAO。

[0211] 使用前述技术中的任意技术来编码的视频可以通过使用任意合适的有线或无线传输媒介来传送，和/或可以记录在任意合适的非临时性数字存储媒介上。

[0212] 虽然上面的特征和元件是以特定组合来描述的，但是本领域技术人员可以理解每个特征或元件可以单独使用或者与其他特征和元件以任意组合使用。另外，这里描述的方法可以在结合在计算机可读介质中的计算机程序、软件、和/或固件中实施，以便由计算机或处理器执行。计算机可读介质的示例包括电信号(通过有线或无线连接发送)和计算机可读存储介质。计算机可读存储介质的示例包括但不限于只读存储器(ROM)、随机存取存储器(RAM)、寄存器、高速缓冲存储器、半导体存储器装置、磁介质(诸如内部硬盘和可移动磁盘)、磁光介质、和光学介质，诸如CD-ROM磁盘和数字多功能磁盘(DVD)。与软件相关联的处理器可以用于实现射频收发信机，以在WTRU、UE、终端、基站、RNC或任意主机中使用。

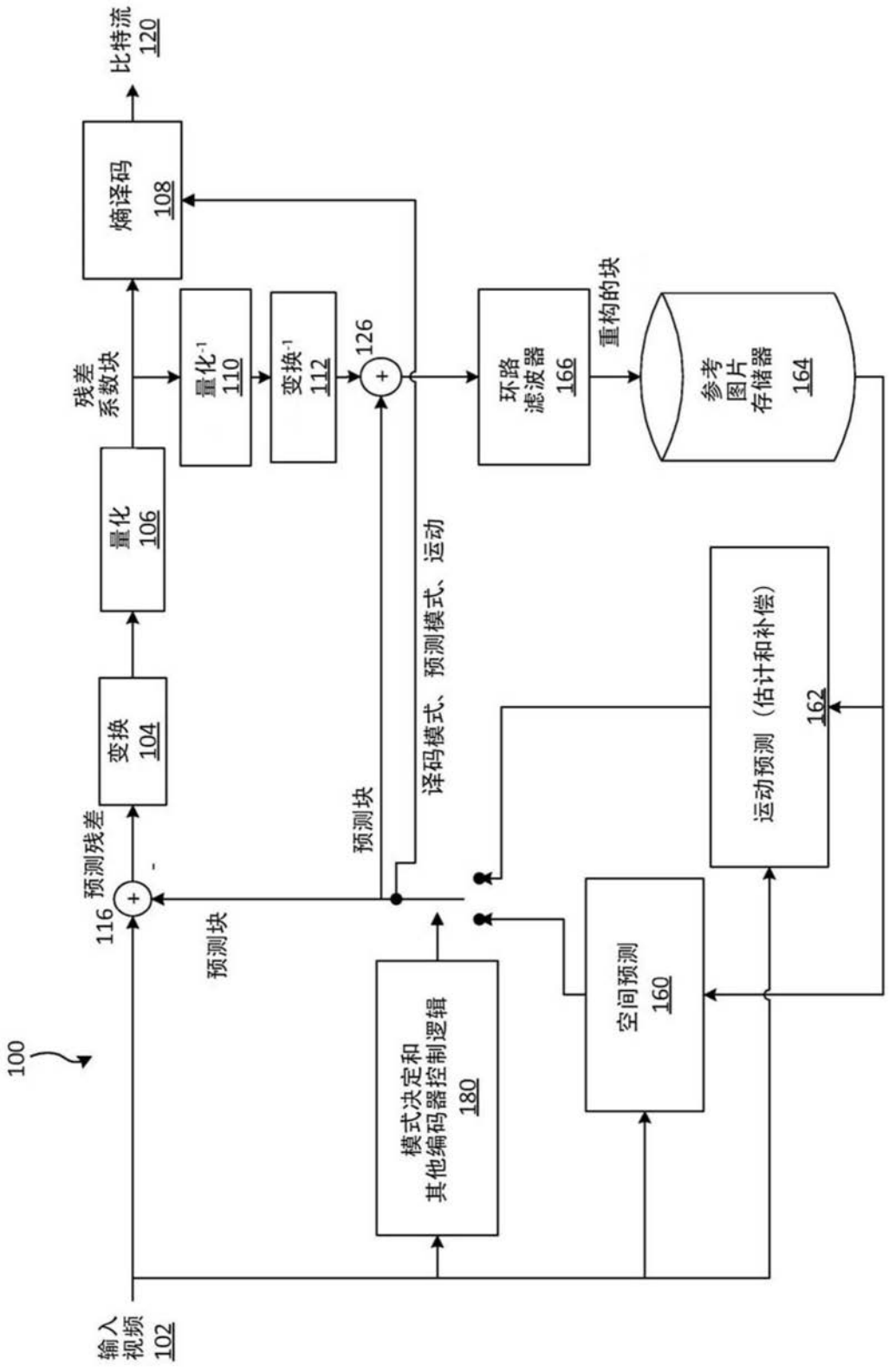

比特流
120
熵译码
108
残差
系数块
量化-1
110
变换-1
112
126
环路
滤波器
166
重构的块
参考
图片
存储器
164
量化
106
变换
104
预测残差
116
预测块
译码模式、预测模式、运动
运动预测（估计和补偿）
162
空间预测
160
模式决定和
其他编码器控制逻辑
180
100
输入
视频
102

图1A

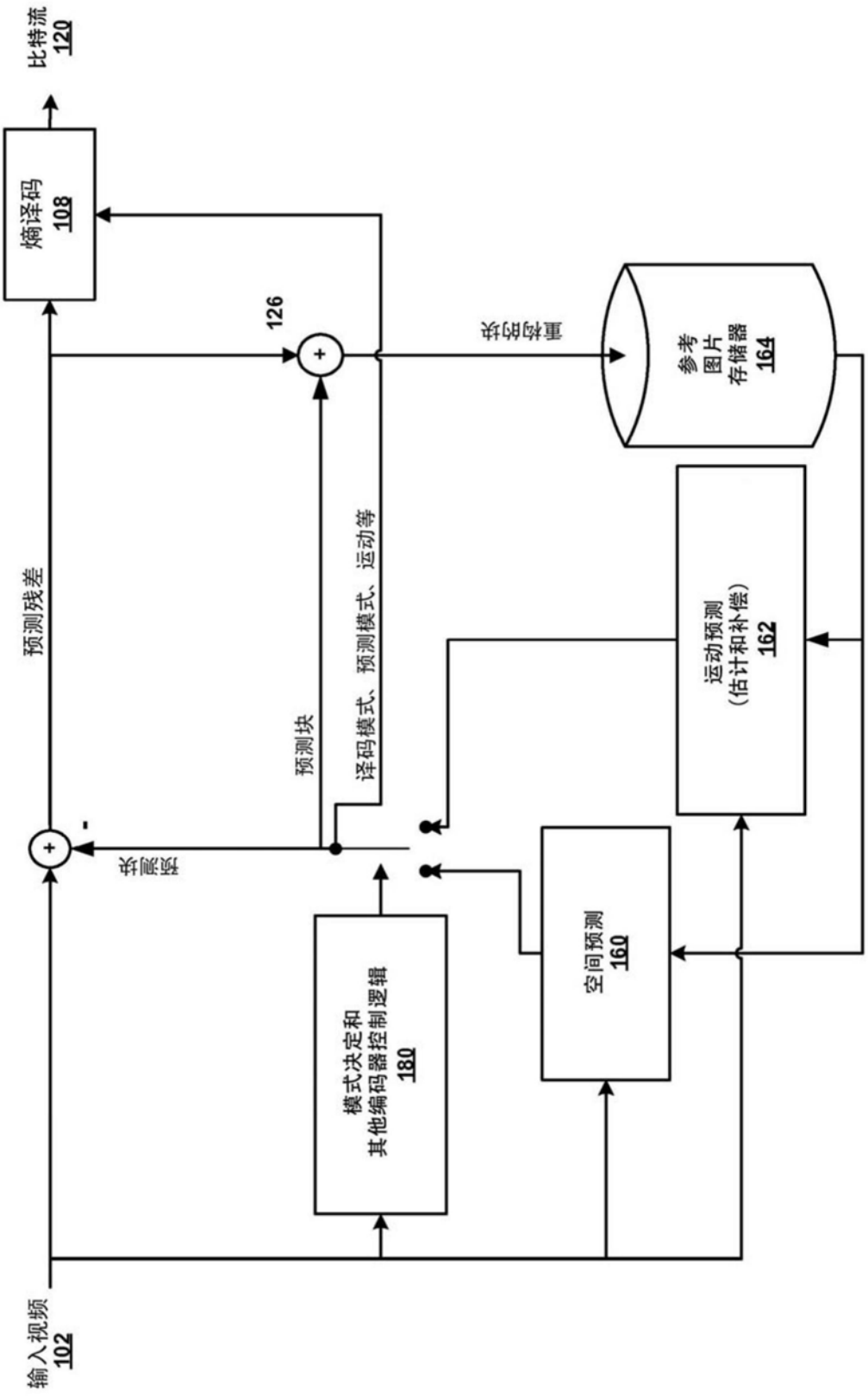

比特流
120
熵译码
108
126
重构宏块
参考
图片
存储器
164
预测残差
预测块
译码模式、预测模式、运动等
运动预测
(估计和补偿)
162
-
残差
模式决定和
其他编码器控制逻辑
180
空间预测
160
输入视频
102

图1B

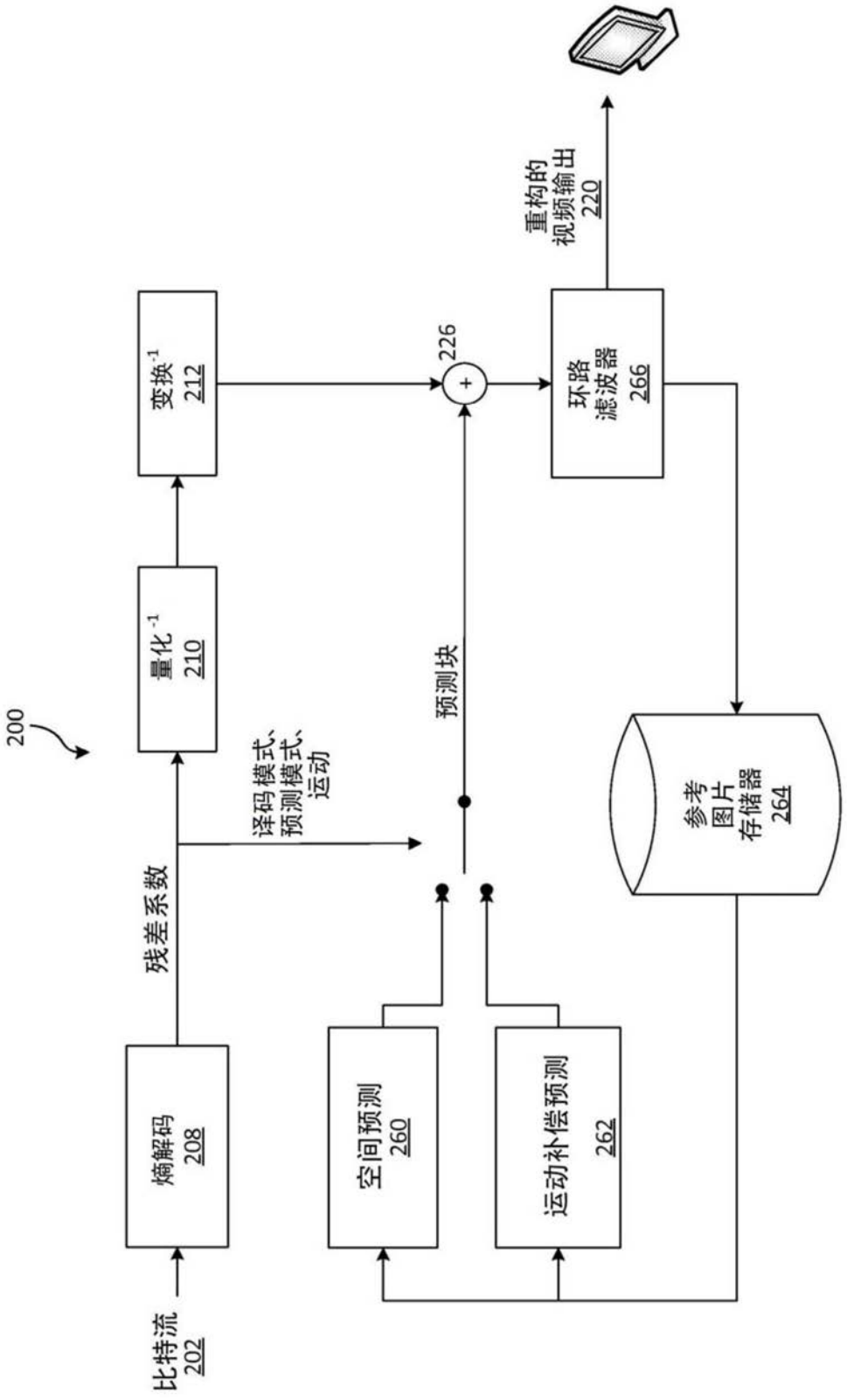

200
比特流
202
熵解码
208
残差系数
量化-1
210
变换-1
212
译码模式、
预测模式、
运动
空间预测
260
运动补偿预测
262
预测块
226
环路
滤波器
266
重构的
视频输出
220
参考
图片
存储器
264

图2A

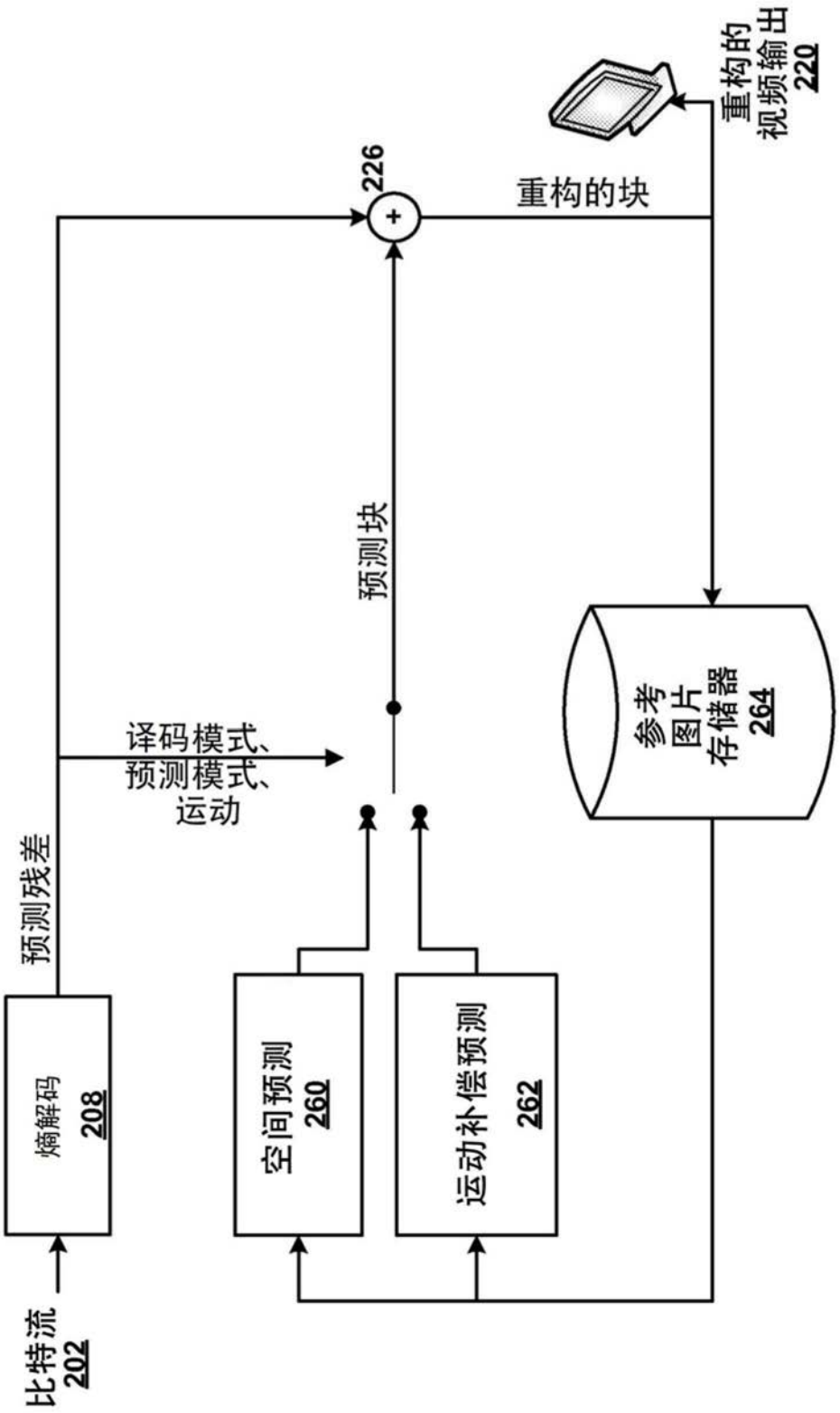

重构的视频输出
220
226
重构的块
预测块
参考图片存储器
264
译码模式、预测模式、运动
预测残差
熵解码
208
空间预测
260
运动补偿预测
262
比特流
202

图2B

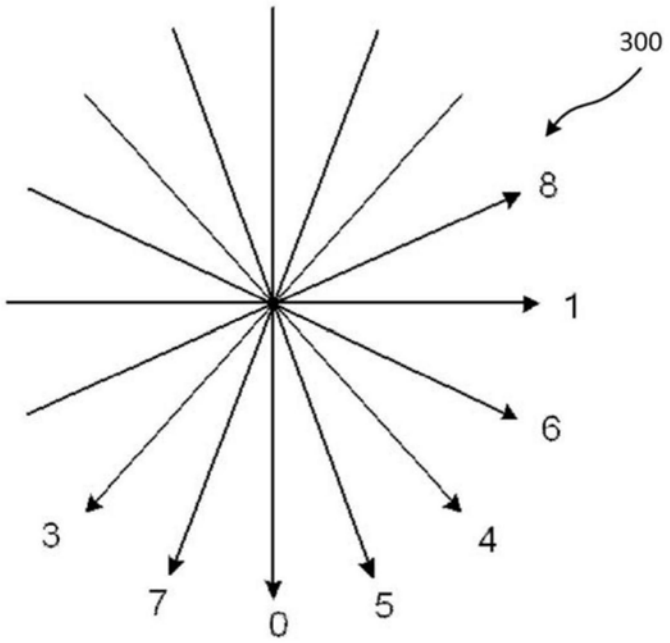

300
8
1
6
4
5
0
7
3

图3

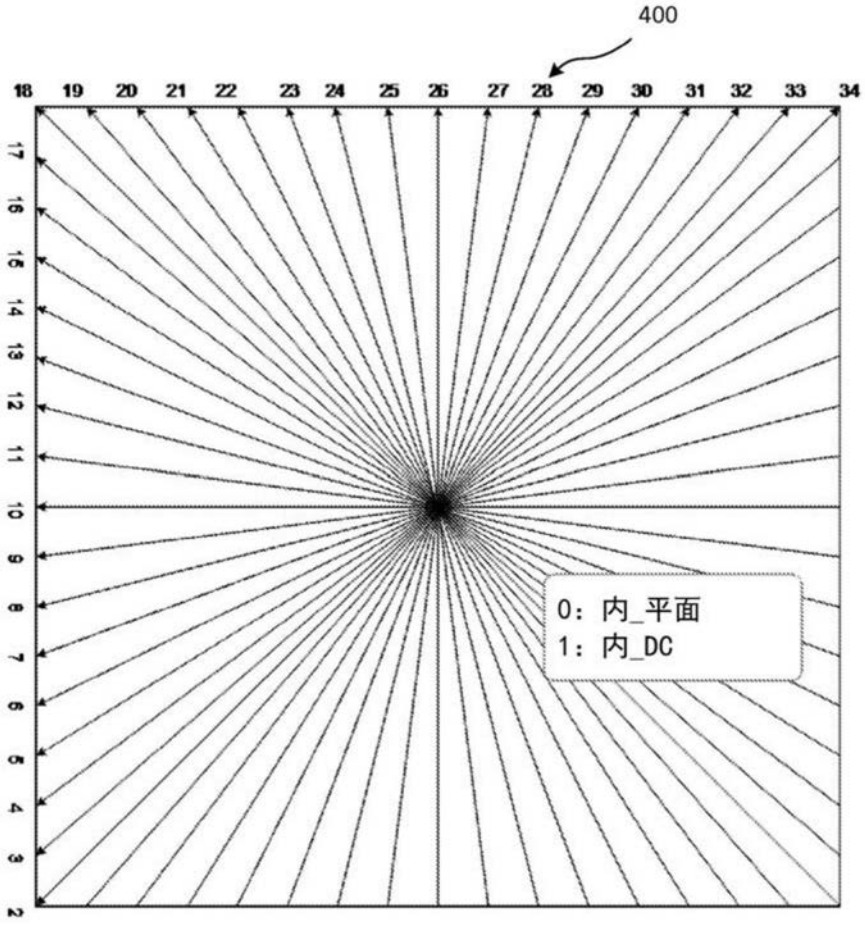

400
18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2
0：内_平面
1：内_DC

图4

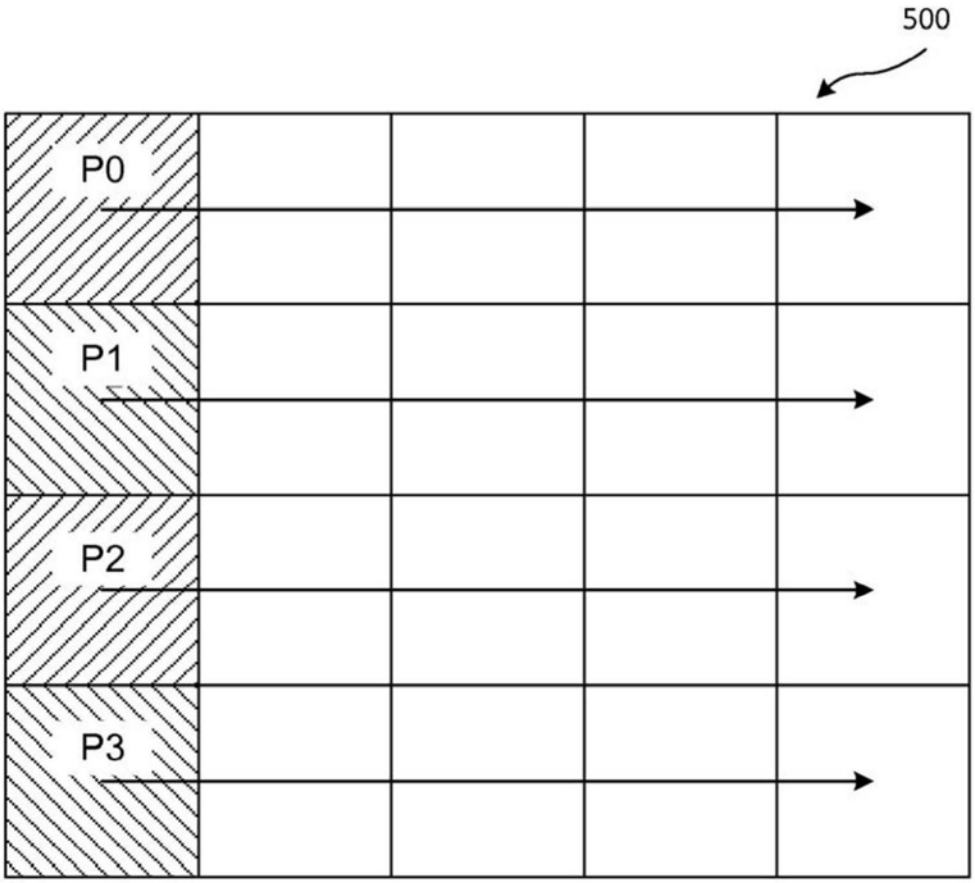


图5

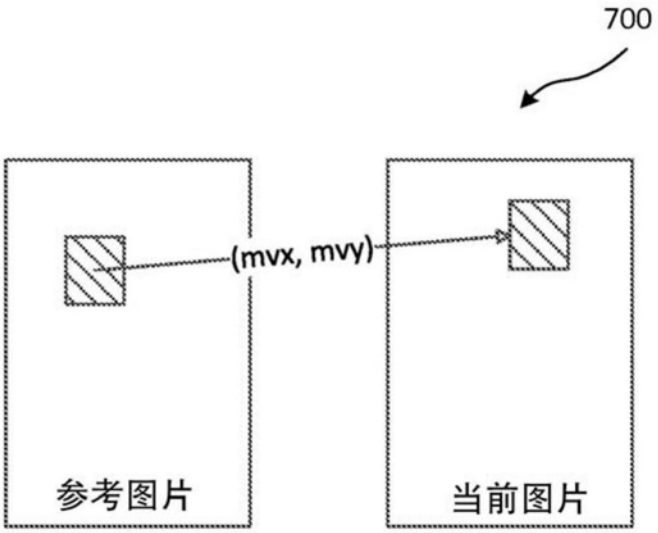


图7

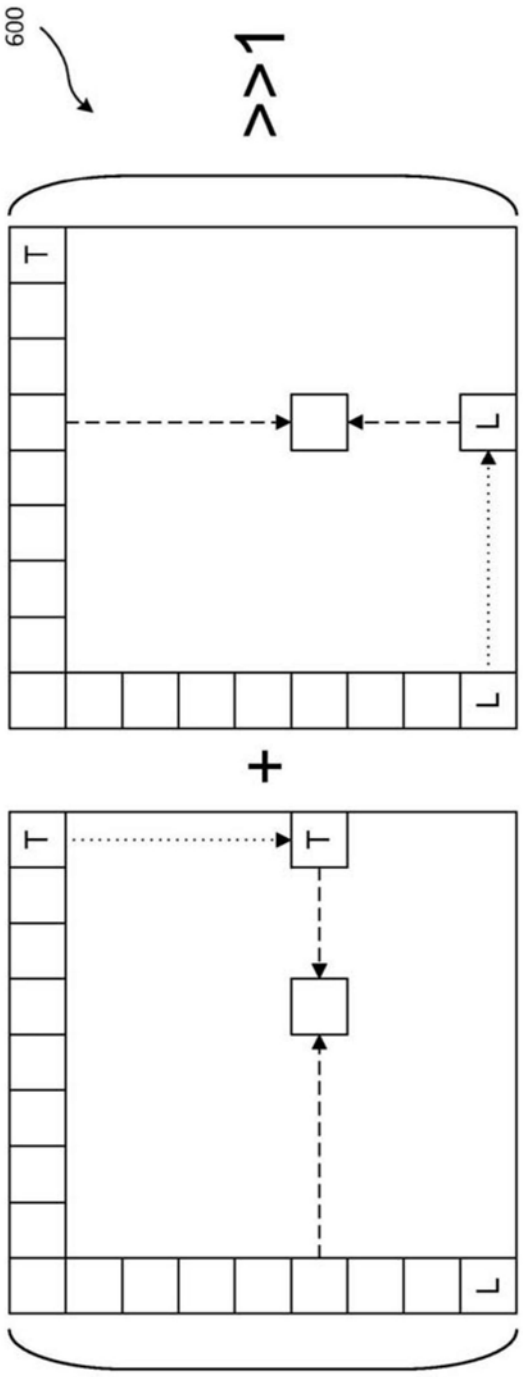


图6

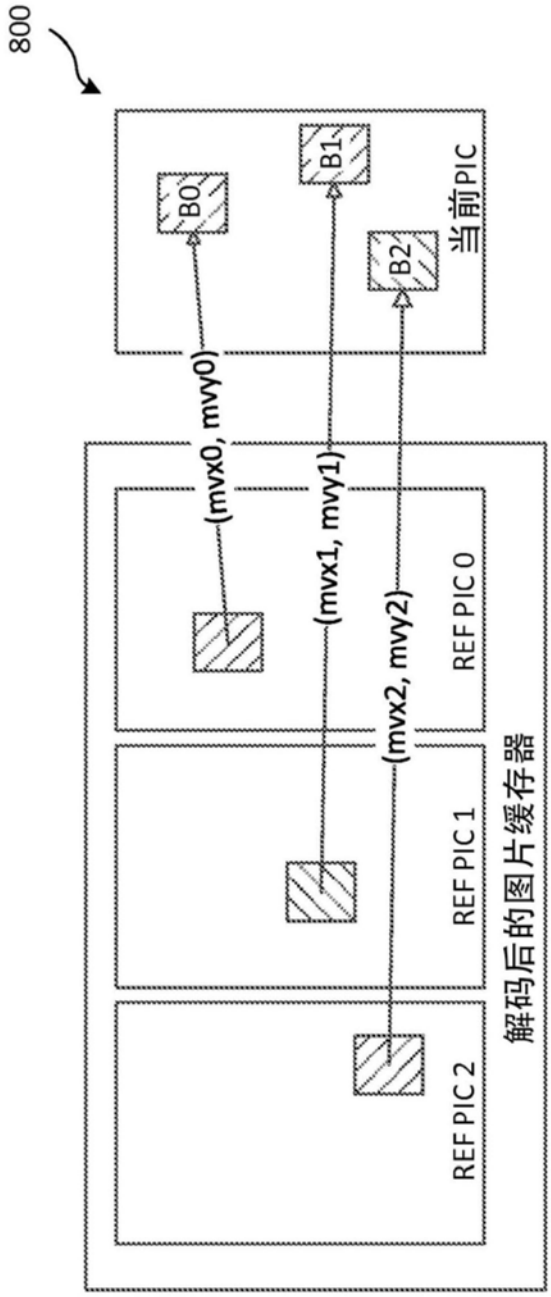

800
B0
B1
B2
当前PIC
(mvx0, mvy0)
(mvx1, mvy1)
(mvx2, mvy2)
REF PIC 0
REF PIC 1
REF PIC 2
解码后的图片缓存器

图8

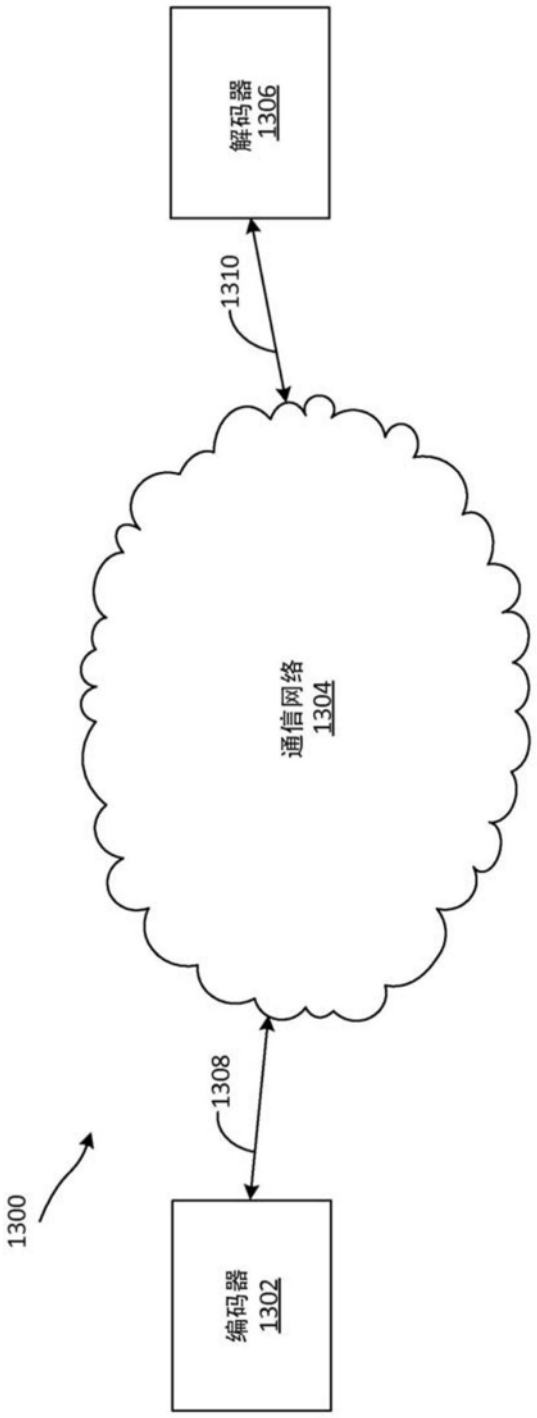


图9

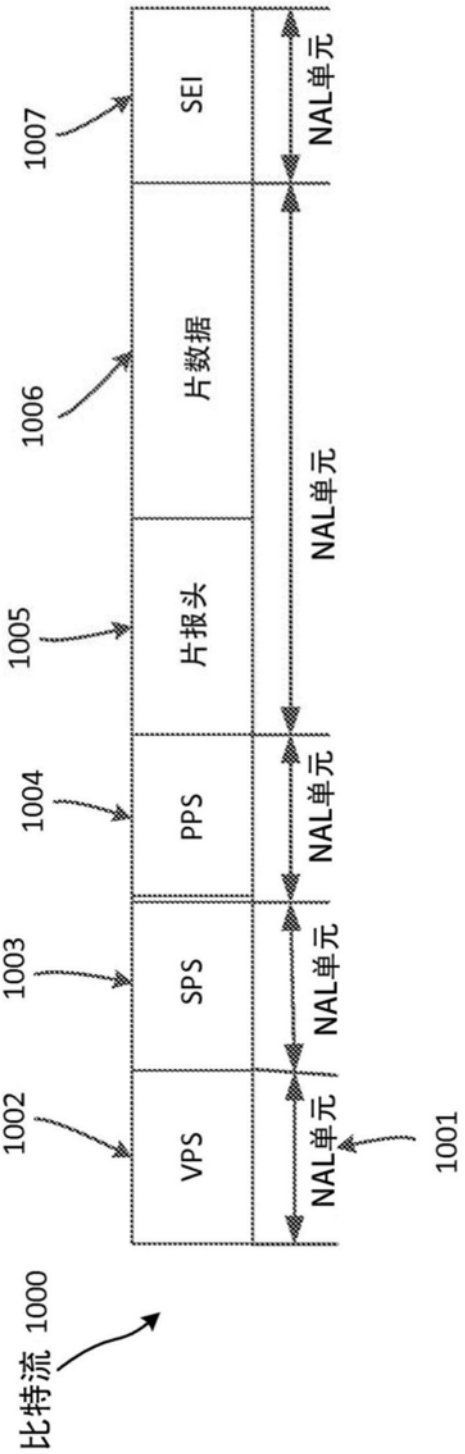

比特流 1000
1002
VPS
1003
SPS
1004
PPS
1005
片报头
1006
片数据
1007
SEI
NAL单元
NAL单元
NAL单元
NAL单元
NAL单元
1001

图11

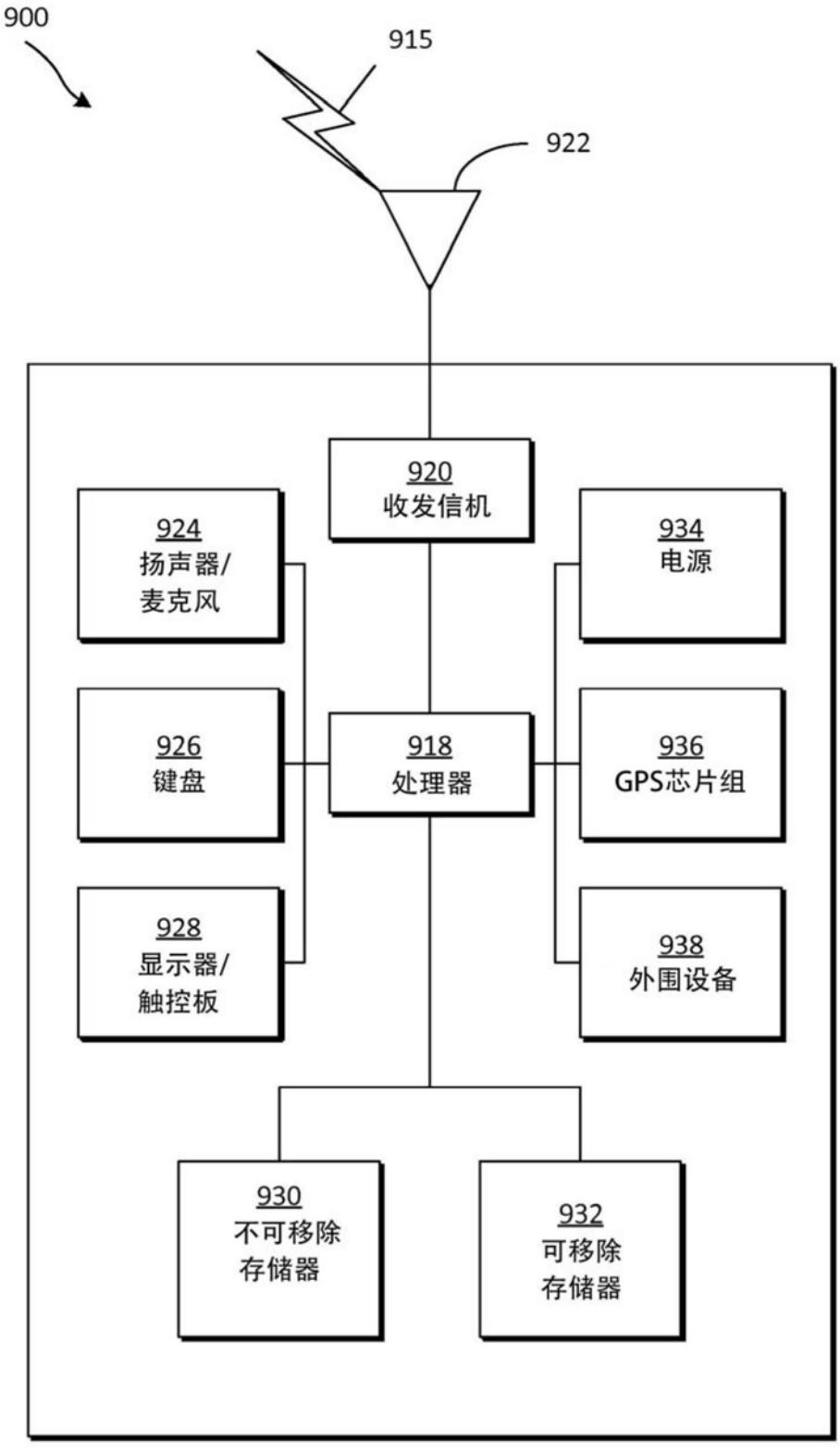

900
915
922
920
收发信机
924
扬声器/
麦克风
926
键盘
928
显示器/
触控板
918
处理器
934
电源
936
GPS芯片组
938
外围设备
930
不可移除
存储器
932
可移除
存储器

图10

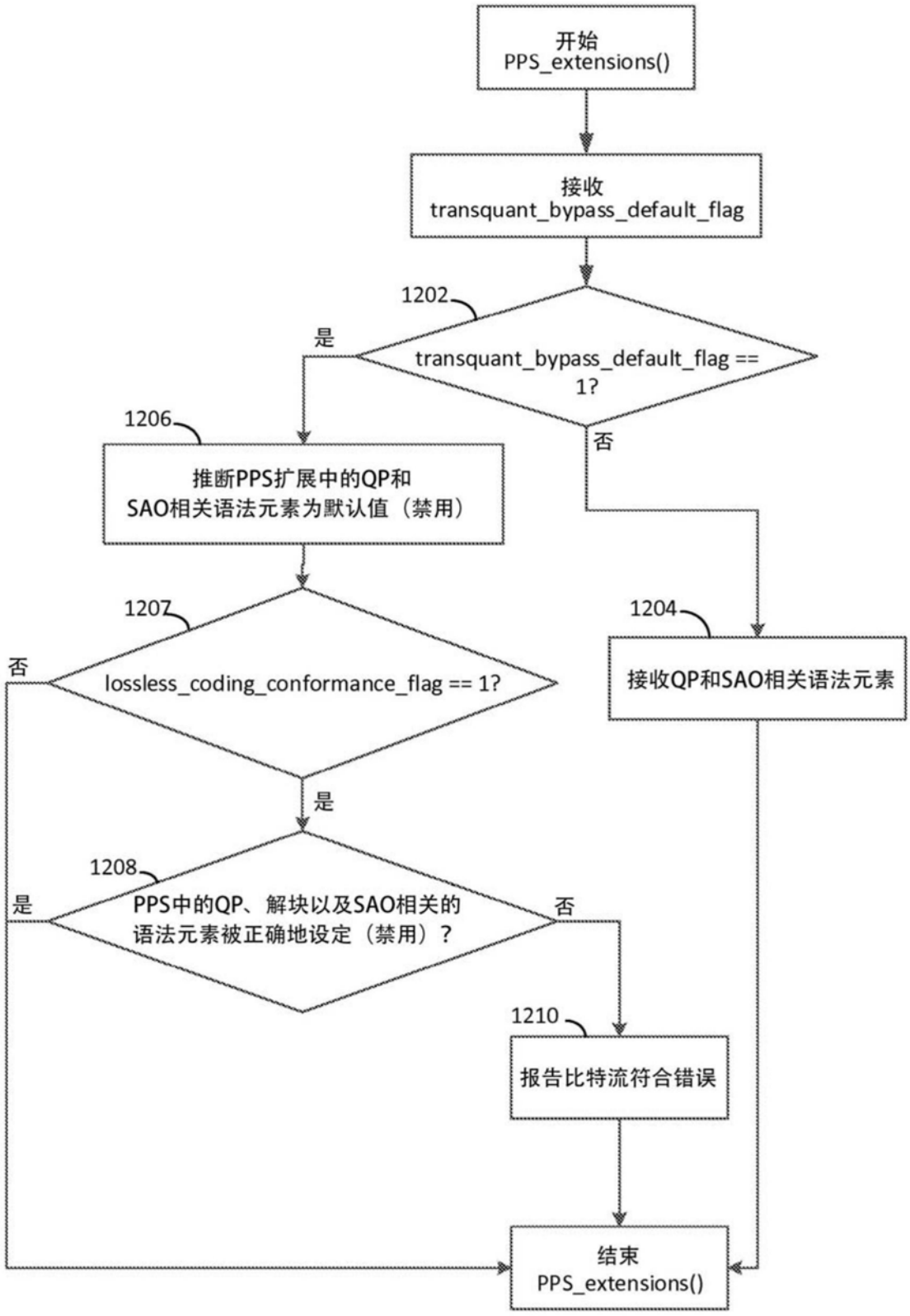

开始
PPS_extensions()
接收
transquant_bypass_default_flag
1202
是
transquant_bypass_default_flag == 1?
否
1206
推断PPS扩展中的QP和
SAO相关语法元素为默认值（禁用）
1204
接收QP和SAO相关语法元素
1207
否
lossless_coding_conformance_flag == 1?
是
1208
是
PPS中的QP、解块以及SAO相关的
语法元素被正确地设定（禁用）？
否
1210
报告比特流符合错误
结束
PPS_extensions()

图12