



(12) 发明专利

(10) 授权公告号 CN 114846795 B

(45) 授权公告日 2024. 04. 12

(21) 申请号 202180007546.8

(22) 申请日 2021.01.13

(65) 同一申请的已公布的文献号
申请公布号 CN 114846795 A

(43) 申请公布日 2022.08.02

(30) 优先权数据
62/961,159 2020.01.14 US

(85) PCT国际申请进入国家阶段日
2022.06.23

(86) PCT国际申请的申请数据
PCT/RU2021/050007 2021.01.13

(87) PCT国际申请的公布数据
W02021/045659 EN 2021.03.11

(73) 专利权人 华为技术有限公司
地址 518129 广东省深圳市龙岗区坂田华为总部办公楼

(72) 发明人 阿列克谢·康斯坦丁诺维奇·菲利波夫
瓦西里·亚历斯维奇·拉夫特斯基
伊蕾娜·亚历山德罗夫娜·阿尔希娜

(51) Int.Cl.
H04N 19/52 (2006.01)

(56) 对比文件
CN 104811734 A, 2015.07.29
US 2014286428 A1, 2014.09.25
US 2018070100 A1, 2018.03.08
WO 2019246535 A1, 2019.12.26
Yao-Jen Chang.AHG9: Cleanup for the maximum number of triangle merging candidates.JVET-Q0336.2020, 第2-3节.
Ling Li.CE4 related: cleanup for signaling maximum number of triangle merge candidates.JVET-Q0422.2020, 第3节第1-2段.
ZHOU MINHUA.JVET-Q0160, CE4-related: CE4-1 spec text with suggested fixes, CE4_1_WD(on_top_of_CE4_CommonBaseWD)_w_fixes.JVET-Q0160.2019, 摘要、7.3.2.3节、7.3.2.6节、7.4.3.3节、7.4.3.6节. (续)

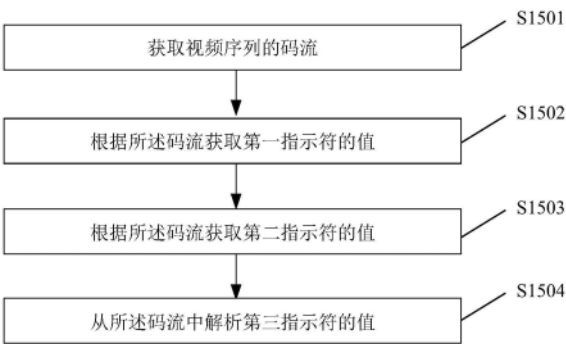
审查员 杨少魁

权利要求书2页 说明书142页 附图12页

(54) 发明名称
指示融合模式候选数量的方法和装置

(57) 摘要
本发明公开了一种获取用于视频解码的几何划分融合模式候选的最大数量的方法和一种视频解码装置。所述方法包括：获取视频序列的码流；根据所述码流获取第一指示符的值，其中，所述第一指示符表示融合运动矢量预测(motion vector prediction, MVP)候选的最大数量；根据所述码流获取第二指示符的值，其中，所述第二指示符表示是否对所述视频序列启用基于几何划分的运动补偿；当所述第一指示符的值大于阈值且所述第二指示符的值等于预设值时，从所述码流中解析第三指示符的值，其中，所述第三指

示符表示从所述第一指示符的值中减去几何划分融合模式候选的最大数量。



CN 114846795 B

[接上页]

(56) 对比文件

ZHOU MINHUA.JVET-Q0160, CE4-related: CE4-1 spec text with suggested fixes, CE4_1_WD(on_top_of_CE4_CommonBaseWD)_w_fixes.JVET-Q0160.2019,摘要、7.3.2.3节、7.3.2.6节、7.4.3.3节、7.4.3.6节.

Gary Sullivan.JVET-Q2000,Meeting Report of the 17th Meeting of the Joint Video Experts Team (JVET), Brussels, BE,

7-17 Jan. 2020.JVET-Q2000.2020,第11.2节-Track A-Open Question.

Wade Wan.JVET-Q0482,AHG9: Signalling of constant parameter values.JVET-Q0482.2020,全文.

袁媛;郑萧桢;何芸.视频编码中一种新的用于几何块划分的变换结构.上海大学学报(自然科学版).2013,(03),全文.

1. 一种获取用于视频解码的几何划分融合模式候选的最大数量的方法,其特征在于,所述方法包括:

获取视频序列的码流;

根据所述码流获取第一指示符的值,其中,所述第一指示符表示融合运动矢量预测MVP候选的最大数量;

根据所述码流获取第二指示符的值,其中,所述第二指示符表示是否对所述视频序列启用基于几何划分的运动补偿;

当所述第一指示符的值大于阈值且所述第二指示符的值等于预设值时,从所述码流中解析第三指示符的值,其中,所述第三指示符表示从所述第一指示符的值中减去几何划分融合模式候选的最大数量;所述第三指示符的值是从所述码流的序列参数集SPS中获取的;根据所述第一指示符的值和所述第三指示符的值得到所述几何划分融合模式候选的最大数量;

当所述第一指示符的值等于所述阈值且所述第二指示符的值等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为2。

2. 根据权利要求1所述的方法,其特征在于,所述阈值为2。

3. 根据权利要求1至2中任一项所述的方法,其特征在于,所述方法还包括:

当所述第一指示符的值小于所述阈值或所述第二指示符的值不等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为0。

4. 根据权利要求1至3中任一项所述的方法,其特征在于,所述预设值为1。

5. 根据权利要求1至4中任一项所述的方法,其特征在于,所述获取第二指示符的值的步骤是在所述获取第一指示符的值的步骤之后执行的。

6. 根据权利要求5所述的方法,其特征在于,当所述第一指示符的值大于或等于所述阈值时,所述第二指示符的值是从所述码流的序列参数集SPS中解析出的。

7. 根据权利要求1至6中任一项所述的方法,其特征在于,所述第二指示符的值是从所述码流的序列参数集SPS中获取的。

8. 一种视频解码装置,其特征在于,所述视频解码装置包括:

接收模块,用于获取视频序列的码流;

获取模块,用于根据所述码流获取第一指示符的值,其中,所述第一指示符表示融合运动矢量预测MVP候选的最大数量;

所述获取模块用于根据所述码流获取第二指示符的值,其中,所述第二指示符表示是否对所述视频序列启用基于几何划分的运动补偿;

解析模块,用于:当所述第一指示符的值大于阈值且所述第二指示符的值等于预设值时,从所述码流中解析第三指示符的值,其中,所述第三指示符表示从所述第一指示符的值中减去几何划分融合模式候选的最大数量;所述第三指示符的值是从所述码流的序列参数集SPS中获取的;根据所述第一指示符的值和所述第三指示符的值得到所述几何划分融合模式候选的最大数量;

当所述第一指示符的值等于所述阈值且所述第二指示符的值等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为2。

9. 根据权利要求8所述的视频解码装置,其特征在于,所述获取模块用于:当所述第一

指示符的值小于所述阈值或所述第二指示符的值不等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为0。

10.根据权利要求8至9中任一项所述的视频解码装置,其特征在于,所述阈值为2。

11.根据权利要求8至10中任一项所述的视频解码装置,其特征在于,所述预设值为1。

12.根据权利要求8至11中任一项所述的视频解码装置,其特征在于,所述获取第二指示符的值的步骤是在所述获取第一指示符的值的步骤之后执行的。

13.根据权利要求12所述的视频解码装置,其特征在于,当所述第一指示符的值大于或等于所述阈值时,所述第二指示符的值是从所述码流的序列参数集SPS中解析出的。

14.根据权利要求8至13中任一项所述的视频解码装置,其特征在于,所述第二指示符的值是从所述码流的序列参数集SPS中获取的。

15.一种解码器,其特征在于,所述解码器包括:

一个或多个处理器;

非瞬时性计算机可读存储介质,耦合到所述一个或多个处理器并存储由所述一个或多个处理器执行的程序,其中,当所述程序由所述一个或多个处理器执行时,所述解码器用于执行根据权利要求1至7中任一项所述的方法。

16.一种包括程序代码的非瞬时性计算机可读存储介质,其特征在于,当所述程序代码由计算机设备执行时,使得所述计算机设备执行根据权利要求1至7中任一项所述的方法。

指示融合模式候选数量的方法和装置

[0001] 相关申请交叉引用

[0002] 本专利申请要求于2020年1月14日提交的US62/961,159的优先权。上述专利申请的全部公开内容通过引用结合在本申请中。

技术领域

[0003] 本申请实施例大体上涉及动态图像编码领域,更具体地涉及指示融合模式候选数量。

背景技术

[0004] 视频译码(视频编码和解码)广泛用于数字视频应用,例如广播数字电视、基于互联网和移动网络的视频传输、视频聊天和视频会议等实时会话应用、DVD和蓝光光盘、视频内容采集和编辑系统以及安全应用的可便携式摄像机。

[0005] 即使视频相对较短,也需要大量的视频数据来描述,当数据要在带宽容量有限的通信网络中流式传输或以其它方式传输时,这样可能会造成困难。因此,视频数据通常要先压缩,然后通过现代电信网络进行传输。由于内存资源可能有限,当在存储设备中存储视频时,该视频的大小也可能是一个问题。视频压缩设备通常在信源侧使用软件和/或硬件对视频数据进行编码,然后传输或存储视频数据,从而减少表示数字视频图像所需的数据量。然后,对视频数据进行解码的视频解压缩设备在目的地侧接收压缩数据。在网络资源有限以及对更高视频质量的需求不断增长的情况下,需要改进压缩和解压缩技术,这些改进的技术在几乎不影响图像质量的情况下能够提高压缩比。

发明内容

[0006] 本申请实施例提供了独立权利要求所述的用于编码和解码的装置和方法。

[0007] 上述和其它目的通过独立权利要求请求保护的主体实现。其它实现方式在从属权利要求、说明书和附图中是显而易见的。

[0008] 特定实施例在所附独立权利要求中概述,其它实施例在从属权利要求中概述。

[0009] 本发明的第一方面提供了一种获取视频解码的几何划分融合模式候选的最大数量的方法。所述方法包括:

[0010] 获取视频序列的码流;根据所述码流获取第一指示符的值,其中,所述第一指示符表示融合运动矢量预测(motion vector prediction,MVP)候选的最大数量;根据所述码流获取第二指示符的值,其中,所述第二指示符表示是否对所述视频序列启用基于几何划分的运动补偿;当所述第一指示符的值大于阈值且所述第二指示符的值等于预设值时,从所述码流中解析第三指示符的值,其中,所述第三指示符表示从所述第一指示符的值中减去几何划分融合模式候选的最大数量。

[0011] 根据本发明的实施例,公开了一种指示融合模式候选数量的指示符的方案。几何划分融合模式候选的最大数量在一定条件下指示。因此,码流利用率和解码效率得到了提

高。

[0012] 在一种实现方式中,所述方法还包括:当所述第一指示符的值等于所述阈值且所述第二指示符的值等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为2。

[0013] 在一种实现方式中,所述方法还包括:当所述第一指示符的值小于所述阈值或所述第二指示符的值不等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为0。

[0014] 在一种实现方式中,所述阈值为2。

[0015] 在一种实现方式中,所述阈值为1。

[0016] 在一种实现方式中,所述获取第二指示符的值的步骤在所述获取第一指示符的值的步骤之后执行。

[0017] 在一种实现方式中,所述第一指示符是根据译码在所述码流中的语法元素获取的。

[0018] 在一种实现方式中,当所述第一指示符的值大于或等于所述阈值时,所述第二指示符的值是从所述码流的序列参数集(sequence parameter set,SPS)中解析出的。例如,解析所述码流的序列参数集(sequence parameter set,SPS)中的语法元素,以获得所述第二指示符的值。

[0019] 在一种实现方式中,所述第二指示符的值是从所述码流的序列参数集(sequence parameter set,SPS)中获取的。例如,解析所述码流的序列参数集(sequence parameter set,SPS)中的语法元素,以获得所述第二指示符的值。

[0020] 在一种实现方式中,所述第三指示符的值是从所述码流的序列参数集(sequence parameter set,SPS)中获取的。例如,解析所述码流的序列参数集(sequence parameter set,SPS)中的语法元素,以获得所述第二指示符的值。

[0021] 本发明的第二方面提供了一种视频解码装置。所述视频解码装置包括:接收模块,用于获取视频序列的码流;获取模块,用于根据所述码流获取第一指示符的值,其中,所述第一指示符表示融合运动矢量预测(motion vector prediction,MVP)候选的最大数量;获取模块用于根据所述码流获取第二指示符的值,其中,所述第二指示符表示是否对所述视频序列启用基于几何划分的运动补偿;解析模块,用于:当所述第一指示符的值大于阈值且所述第二指示符的值等于预设值时,从所述码流中解析第三指示符的值,其中,所述第三指示符表示从所述第一指示符的值中减去所述几何划分融合模式候选的最大数量。

[0022] 本发明所述第一方面提供的方法可以由本发明所述第二方面提供的装置执行。本发明所述第一方面提供的方法的其它特征和实现方式对应于本发明所述第二方面提供的装置的特征和实现方式。

[0023] 在一种实现方式中,所述获取模块用于:当所述第一指标的值等于所述阈值且所述第二指标的值等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为2。

[0024] 在一种实现方式中,所述获取模块用于:当所述第一指示符的值小于所述阈值或所述第二指示符的值不等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为0。

[0025] 在一种实现方式中,所述阈值为2。

[0026] 在一种实现方式中,所述阈值为1。

[0027] 在一种实现方式中,所述获取第二指示符的值的步骤在所述获取第一指示符的值的步骤之后执行。

[0028] 在一种实现方式中,当所述第一指示符的值大于或等于所述阈值时,所述第二指示符的值是从所述码流的序列参数集(sequence parameter set,SPS)中解析出的。

[0029] 在一种实现方式中,所述第二指示符的值是从所述码流的序列参数集(sequence parameter set,SPS)中获取的。

[0030] 在一种实现方式中,所述第三指示符的值是从所述码流的序列参数集(sequence parameterset,SPS)中获取的。

[0031] 在一种实现方式中,公开了一种获取用于视频解码的几何划分融合模式候选的最大数量的方法。所述方法包括:

[0032] 获取视频序列的码流;根据所述码流获取第一指示符的值,其中,所述第一指示符表示融合运动矢量预测(motion vector prediction,MVP)候选的最大数量;只要为所述第一指示符获取的值等于或大于阈值,根据所述码流获取第二指示符的值,其中,所述第二指示符表示是否对所述视频序列启用基于几何划分的运动补偿;只有当所述第一指示符的值大于所述阈值且所述第二指示符的值等于预设值时,从所述码流中解析第三指示符的值,其中,所述第三指示符表示从所述第一指示符的值中减去几何划分融合模式候选的最大数量。

[0033] 本发明第三方面提供了一种对几何划分融合模式候选的最大数量进行编码的方法。所述方法包括:

[0034] 确定第一指示符的值,其中,所述第一指示符表示融合运动矢量预测(motion vector prediction,MVP)候选的最大数量;确定第二指示符的值,其中,所述第二指示符表示是否对视频序列启用基于几何划分的运动补偿;当所述第一指示符的值大于阈值且所述第二指示符的值等于预设值时,将第三指示符的值编码到码流中,其中,所述第三指示符表示从所述第一指示符的值中减去几何划分融合模式候选的最大数量。

[0035] 根据本发明的实施例,公开了一种指示融合模式候选数量的指示符的方案。几何划分融合模式候选的最大数量在一定条件下指示。因此,码流利用率和解码效率得到了提高。

[0036] 在一种实现方式中,所述方法还包括:当所述第一指示符的值等于所述阈值且所述第二指示符的值等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为2。

[0037] 在一种实现方式中,所述方法还包括:当所述第一指示符的值小于所述阈值或所述第二指示符的值不等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为0。

[0038] 在一种实现方式中,所述阈值为2。

[0039] 在一种实现方式中,所述阈值为1。

[0040] 在一种实现方式中,所述确定第二指示符的值的步骤在所述确定第一指示符的值的步骤之后执行。

[0041] 在一种实现方式中,当所述第一指示符的值大于或等于所述阈值时,所述第二指示符的值被编码到所述码流的序列参数集(sequence parameter set,SPS)中。

[0042] 在一种实现方式中,所述第二指示符的值被编码在所述码流的序列参数集(sequence parameter set,SPS)中。

[0043] 在一种实现方式中,所述第三指示符的值被编码在所述码流的序列参数集(sequence parameter set,SPS)中。

[0044] 本发明的第四方面提供了一种视频编码装置。所述视频编码装置包括:确定模块,用于确定第一指示符的值,其中,所述第一指示符表示融合运动矢量预测(motion vector prediction,MVP)候选的最大数量;所述确定模块用于确定第二指示符的值,其中,所述第二指示符表示是否对视频序列启用基于几何划分的运动补偿;编码模块,用于:当所述第一指示符的值大于阈值且所述第二指示符的值等于预设值时,将第三指示符的值编码到码流中,其中,所述第三指示符表示从所述第一指示符的值中减去几何划分融合模式候选的最大数量。

[0045] 本发明所述第三方面提供的方法可以由本发明所述第四方面提供的装置执行。本发明所述第三方面提供的方法的其它特征和实现方式对应于本发明所述第四方面提供的装置的特征和实现方式。

[0046] 在一种实现方式中,所述确定模块用于:当所述第一指示符的值等于所述阈值且所述第二指示符的值等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为2。

[0047] 在一种实现方式中,所述确定模块用于:当所述第一指示符的值小于所述阈值或所述第二指示符的值不等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为0。

[0048] 在一种实现方式中,所述阈值为2。

[0049] 在一种实现方式中,所述阈值为1。

[0050] 在一种实现方式中,所述确定第二指示符的值的步骤在所述确定第一指示符的值的步骤之后执行。

[0051] 在一种实现方式中,当所述第一指示符的值大于或等于所述阈值时,所述第二指示符的值被编码到所述码流的序列参数集(sequence parameter set,SPS)中。

[0052] 在一种实现方式中,所述第二指示符的值被编码在所述码流的序列参数集(sequence parameter set,SPS)中。

[0053] 在一种实现方式中,所述第三指示符的值被编码在所述码流的序列参数集(sequence parameter set,SPS)中。

[0054] 本发明的第五方面提供了一种解码器。所述解码器包括用于执行所述第一方面和所述第一方面的任一种实现方式提供的方法的处理电路。

[0055] 本发明的第六方面提供了一种编码器。所述编码器包括用于执行所述第三方面和所述第三方面的任一种实现方式提供的方法的处理电路。

[0056] 本发明的第七方面提供了一种计算机程序。所述计算机程序包括程序代码。所述程序代码在计算机或处理器上执行时,用于执行所述第一方面、所述第三方面和所述第一方面、所述第三方面中的任一种实现方式提供的方法。

[0057] 本发明的第八方面提供了一种解码器。所述解码器包括：一个或多个处理器；非瞬时性计算机可读存储介质，耦合到所述一个或多个处理器并存储由所述一个或多个处理器执行的程序，其中，当所述程序由所述一个或多个处理器执行时，所述解码器用于执行所述第一方面、所述第三方面和所述第一方面、所述第三方面中的任一种实现方式中的任一个提供的方法。

[0058] 本发明的第九方面提供了一种包括程序代码的非瞬时性计算机可读存储介质。当所述程序代码由计算机设备执行时，使得所述计算机设备执行所述第一方面、所述第三方面和所述第一方面、所述第三方面中的任一种实现方式中的任一个提供的方法。

[0059] 本发明的第十方面提供了一种编码器。所述编码器包括用于执行所述第三方面和所述第三方面的任一种实现方式提供的方法的处理电路。

[0060] 本发明的第十一方面提供了一种编码器。所述编码器包括：一个或多个处理器；非瞬时性计算机可读存储介质，耦合到所述一个或多个处理器并存储由所述一个或多个处理器执行的程序，其中，当所述程序由所述一个或多个处理器执行时，所述解码器用于执行所述第三方面和所述第三方面的任一种实现方式中的任一个提供的方法。

[0061] 本发明的第十二方面提供了一种非瞬时性存储介质，包括通过上述任一实施例的方法进行编码/解码的码流。

[0062] 本发明的第十三方面提供了一种视频信号的经编码码流，包括多个语法元素，其中，所述多个语法元素包括第二指示符（例如sps_geo_enabled_flag），第三指示符sps_max_num_merge_cand_minus_max_num_geo_cand在一定条件下至少根据sps_geo_enabled_flag的值指示。

[0063] 本发明的第十四方面提供了一种非瞬时性存储介质，包括由图像解码设备解码的经编码码流，其中，所述码流是通过将视频信号或图像信号中的帧分成多个块生成的，所述码流包括多个语法元素，所述多个语法元素包括第三指示符（例如sps_max_num_merge_cand_minus_max_num_geo_cand）。

[0064] 本发明的第十五方面提供了一种用于视频解码的方法。所述方法包括：

[0065] 获取视频序列的码流；根据所述码流获取第一指示符的值，其中，所述第一指示符表示融合运动矢量预测（motion vector prediction, MVP）候选的最大数量；根据所述码流获取第二指示符的值，其中，所述第二指示符表示是否对所述视频序列启用基于几何划分的运动补偿；当所述第一指示符的值大于阈值且所述第二指示符的值等于预设值时，从所述码流中解析第三指示符的值，其中，所述第三指示符表示从所述第一指示符的值中减去几何划分融合模式候选的最大数量；

[0066] 根据当前编码块的相邻块的运动矢量，构建所述当前编码块的融合候选列表；

[0067] 根据所述第三指示符的值，获取融合索引；

[0068] 根据所述融合索引和所述融合候选列表，获取所述当前编码块的运动矢量；

[0069] 根据所述当前编码块的所述运动矢量，重建所述当前编码块。

[0070] 本发明的第十六方面提供了一种视频解码装置。所述视频解码装置包括：接收模块，用于获取视频序列的码流；获取模块，用于根据所述码流获取第一指示符的值，其中，所述第一指示符表示融合运动矢量预测（motion vector prediction, MVP）候选的最大数量；获取模块用于根据所述码流获取第二指示符的值，其中，所述第二指示符表示是否对所述

视频序列启用基于几何划分的运动补偿;解析模块,用于:当所述第一指示符的值大于阈值且所述第二指示符的值等于预设值时,从所述码流中解析第三指示符的值,其中,所述第三指示符表示从所述第一指示符的值中减去所述几何划分融合模式候选的最大数量;

[0071] 融合候选列表构建模块,用于根据当前编码块的相邻块的运动矢量,构建所述当前编码块的融合候选列表;

[0072] 所述获取模块用于根据所述第三指示符的值获取融合索引;

[0073] 运动矢量获取模块,用于根据所述融合索引和所述融合候选列表,获取所述当前编码块的运动矢量;

[0074] 像素重建模块,用于根据所述当前编码块的所述运动矢量重建所述当前编码块。

[0075] 关于本发明所述十五方面和本发明所述十六方面的详细内容或示例可以参考上述在本发明所述第一方面至所述十四方面中公开的示例。

[0076] 上述和其它目的通过独立权利要求请求保护的主体实现。其它实现方式在从属权利要求、说明书和附图中是显而易见的。

[0077] 以下附图和说明书详细阐述了一个或多个实施例。其它特征、目的和优点在说明书、附图和权利要求书中是显而易见的。

附图说明

[0078] 下面参考所附附图和示意图更详细地描述本发明实施例。

[0079] 图1A为用于实现本发明实施例的视频译码系统的一个示例的框图。

[0080] 图1B为用于实现本发明实施例的视频译码系统的另一个示例的框图。

[0081] 图2为用于实现本发明实施例的视频编码器的一个示例的框图。

[0082] 图3为用于实现本发明实施例的视频解码器的一种示例性结构的框图。

[0083] 图4为编码装置或解码装置的一个示例的框图。

[0084] 图5为编码装置或解码装置的另一个示例的框图。

[0085] 图6为加权预测编码器决策和参数估计的流程图。

[0086] 图7示出了三角预测模式的一个示例。

[0087] 图8示出了几何预测模式的一个示例。

[0088] 图9示出了几何预测模式的另一个示例。

[0089] 图10为实现内容分发服务的内容供应系统3100的一种示例性结构的框图。

[0090] 图11为终端设备的一种示例性结构的框图。

[0091] 图12为本申请提供的帧间预测方法的一个示例的框图。

[0092] 图13为本申请提供的用于帧间预测的装置的一个示例的框图。

[0093] 图14为本申请提供的用于帧间预测的装置的另一个示例的框图。

[0094] 图15为本发明提供的方法实施例的流程图。

[0095] 图16为本发明提供的装置实施例的框图。

[0096] 在下文,如果没有另外明确说明,相同的附图标记指代相同特征或至少在功能上等效的特征。

具体实施方式

[0097] 以下描述中,参考组成本发明一部分并以说明的方式示出本发明实施例的具体方面或可以使用本发明实施例的具体方面的附图。应当理解的是,本发明实施例可以在其它方面中使用,并且可以包括附图中未描绘的结构变化或逻辑变化。因此,以下详细描述不应以限制性的意义来理解,且本发明的范围由所附权利要求书界定。

[0098] 例如,应当理解的是,与描述方法有关的公开内容可以对用于执行所述方法的对应设备或系统也同样适用,反之亦然。例如,如果描述一个或多个具体方法步骤,则对应的设备可以包括功能单元等一个或多个单元,来执行所描述的一个或多个方法步骤(例如,一个单元执行一个或多个步骤,或多个单元分别执行多个步骤中的一个或多个),即使附图中未明确描述或说明这种一个或多个单元。此外,如果根据功能单元等一个或多个单元描述具体装置,则对应的方法可以包括一个步骤来执行一个或多个单元的功能(例如,一个步骤执行一个或多个单元的功能,或多个步骤分别执行多个单元中一个或多个单元的功能),即使附图中未明确描述或说明这种一个或多个单元。进一步地,可以理解的是,除非另外明确提出,本文中所描述的各示例性实施例和/或方面的特征可以相互组合。

[0099] 视频译码通常是指处理形成视频或视频序列的图像序列。在视频编码领域,术语“帧(frame)”与“图像(picture/image)”可以用作同义词。视频译码(或通常称为译码)包括视频编码和视频解码两部分。视频编码在源侧执行,通常包括处理(例如通过压缩)原始视频图像,以减少表示视频图像所需的数据量(从而更高效地存储和/或传输)。视频解码在目的地侧执行,通常包括相对于编码器作逆处理,以重建视频图像。实施例涉及的视频图像(或通常称为图像)的“译码”应当理解为涉及视频图像或相应视频序列的“编码”或“解码”。编码部分和解码部分也合称为编解码(CODEC)(编码和解码)。

[0100] 在无损视频译码情况下,可以重建原始视频图像,即经重建视频图像具有与原始视频图像相同的质量(假设存储或传输期间没有传输损耗或其它数据丢失)。在有损视频译码情况下,通过量化等执行进一步压缩,来减少表示视频图像所需的数据量,而解码器侧无法完全重建视频图像,即经重建视频图像的质量相比原始视频图像的质量较低或较差。

[0101] 几个视频编码标准属于“有损混合型视频编解码器”组(即,将样本域中的空间预测和时间预测与变换域中用于应用量化的二维变换编码相结合)。视频序列中的每个图像通常划分成不重叠的块集合,通常在块层级进行译码。换句话说,编码器侧通常在块(视频块)层级处理亦即编码视频,例如,通过空间(帧内)预测和/或时间(帧间)预测来生成预测块,从当前块(当前处理/待处理的块)中减去预测块以获取残差块,在变换域变换残差块并量化残差块,以减少待传输(压缩)的数据量,而解码器侧将相对于编码器的逆处理部分应用于经编码或经压缩块,以重建用于表示的当前块。此外,编码器复制解码器处理循环,使得编码器和解码器生成相同的预测(例如帧内预测和帧间预测)和/或重建,用于处理亦即译码后续块。

[0102] 在以下视频译码系统10的实施例中,视频编码器20和视频解码器30根据图1A、图1B、图2和图3进行描述。

[0103] 图1A为示例译码系统10的示意性框图,例如可以利用本申请中技术的视频译码系统10(或简称为译码系统10)。视频译码系统10中的视频编码器20(或简称为编码器20)和视频解码器30(或简称为解码器30)为两个示例,即可以用于根据本申请中描述的各种示例执

行各种技术的设备。

[0104] 如图1A所示,译码系统10包括源设备12,源设备12用于将经编码图像数据21提供给目的地设备14等,以对经编码图像数据21进行解码。

[0105] 源设备12包括编码器20,并且可以另外(即可选地)包括图像源16、预处理器(或预处理单元)18(例如图像预处理器18)和通信接口或通信单元22。

[0106] 图像源16可以包括或可以是任何类型的用于捕获真实世界图像等的图像捕获设备;和/或任何类型的图像生成设备(例如用于生成计算机动画图像的计算机图形处理器);或者任何类型的用于获取和/或提供真实世界图像、计算机动画图像(例如屏幕内容、虚拟现实(virtual reality,VR)图像)和/或其任何组合(例如增强现实(augmented reality,AR)图像)的设备。图像源可以为任何类型的存储任一上述图像的存储器(memory/storage)。

[0107] 为了区分预处理器18和预处理单元18执行的处理,图像或图像数据17也可以称为原始图像或原始图像数据17。

[0108] 预处理器18用于接收(原始)图像数据17并对图像数据17执行预处理,得到预处理图像19或预处理图像数据19。预处理器18执行的预处理可以包括修剪(trimming)、颜色格式转换(例如从RGB转换为YCbCr)、调色或去噪等。可以理解的是,预处理单元18可以为可选组件。

[0109] 视频编码器20用于接收预处理图像数据19并提供经编码图像数据21(结合图2等描述更多细节)。

[0110] 源设备12中的通信接口22可以用于接收经编码图像数据21,并通过通信信道13将经编码图像数据21(或对经编码图像数据21进一步处理后得到的数据)发送给另一设备(例如目的地设备14)或任何其它设备,以便进行存储或直接重建。

[0111] 目的地设备14包括解码器30(例如视频解码器30),并且可以另外(即可选地)包括通信接口或通信单元28、后处理器32(或后处理单元32)和显示设备34。

[0112] 目的地设备14中的通信接口28用于直接从源设备12或从存储设备(例如经编码图像数据存储设备)等任何其它源,接收经编码图像数据21(或对经编码图像数据21进一步处理后得到的数据),并将经编码图像数据21提供给解码器30。

[0113] 通信接口22和通信接口28可以用于经由源设备12与目的地设备14之间的直接通信链路(例如,直接有线或无线连接)或者经由任何类型的网络(例如,有线网络、无线网络或其任何组合,或者任何类型的私网和公网或其任何类型的组合)发送或接收经编码图像数据21或经编码数据21。

[0114] 例如,通信接口22可以用于将经编码图像数据21封装成合适的格式(例如数据包),和/或通过任何类型的传输编码或处理方式来处理经编码图像数据,以便通过通信链路或通信网络进行传输。

[0115] 例如,与通信接口22对应的通信接口28可以用于接收传输数据,并通过任何类型的对应传输解码或处理和/或解封装方式来处理传输数据,得到经编码图像数据21。

[0116] 通信接口22和通信接口28都可以配置为图1A中从源设备12指向目的地设备14的通信信道13的箭头所指示的单向通信接口,或者配置为双向通信接口,并且可以用于发送和接收消息等,以建立连接、确认并交换与通信链路和/或数据传输(例如经编码图像数据

传输)相关的任何其它信息,等等。

[0117] 解码器30用于接收经编码图像数据21并提供经解码图像数据31或经解码图像31(下文结合图3或图5等描述更多细节)。

[0118] 目的地设备14中的后处理器32用于对经解码图像数据31(也称为经重建图像数据)(例如经解码图像31)进行后处理,得到后处理图像数据33(例如后处理图像33)。后处理单元32执行的后处理可以包括颜色格式转换(例如从YCbCr转换为RGB)、调色、修剪(trimming)或重采样,或者任何其它处理,以便提供经解码图像数据31由显示设备34等显示,等等。

[0119] 目的地设备14中的显示设备34用于接收后处理图像数据33,以便向用户或观看者等显示图像。显示设备34可以为或可以包括任何类型的用于表示经重建图像的显示器,例如集成或外部显示器或显示屏。例如,显示器可以包括液晶显示器(liquid crystal display,LCD)、有机发光二极管(organic light emitting diode,OLED)显示器、等离子显示器、投影仪、微型LED显示器、硅基液晶(liquid crystal on silicon,LCoS)显示器、数字光处理器(digital light processor,DLP)或任何类型的其它显示器。

[0120] 虽然图1A中将源设备12和目的地设备14绘示为单独的设备,但是设备实施例也可以同时包括这两种设备或同时包括这两种设备的功能,即源设备12或对应的功能以及目的地设备14或对应的功能。在这些实施例中,可以使用相同硬件和/或软件,或使用单独的硬件和/或软件,或其任何组合来实施源设备12或对应的功能以及目的地设备14或对应的功能。

[0121] 本领域技术人员基于描述明显可知,不同单元的功能或图1A所示的源设备12和/或目的地设备14的功能的存在和(准确)分割可能根据实际设备 and 应用有所不同。

[0122] 编码器20(例如视频编码器20)或解码器30(例如视频解码器30)或者编码器20和解码器30可以通过图1B所示的处理电路来实现,例如一个或多个微处理器、一个或多个数字信号处理器(digital signal processor,DSP)、一个或多个专用集成电路(application-specific integrated circuit,ASIC)、一个或多个现场可编程门阵列(field-programmable gate array,FPGA)、一个或多个离散逻辑、一个或多个硬件、一个或多个视频译码专用处理器或其任意组合。编码器20可以通过处理电路46实现,以包含参照图2中的编码器20描述的各种模块和/或本文描述的任何其它编码器系统或子系统。解码器30可以通过处理电路46实现,以包含参照图3中的解码器30描述的各种模块和/或本文描述的任何其它解码器系统或子系统。处理电路可以用于执行下文描述的各种操作。如图5所示,如果上述技术部分在软件中实现,则一种设备可以将该软件的指令存储在合适的非瞬时性计算机可读介质中,并且可以使用一个或多个处理器在硬件中执行这些指令,以执行本发明中的技术。视频编码器20或视频解码器30可以作为组合编解码器(CODEC)的一部分集成在单个设备中,如图1B所示。

[0123] 源设备12和目的地设备14可以包括多种设备中的任一种,包括任何类型的手持或固定设备,例如笔记本或膝上型电脑、手机、智能手机、平板或平板电脑、摄像机、台式电脑、机顶盒、电视机、显示设备、数字媒体播放器、视频游戏机、视频流设备(例如内容服务服务器或内容分发服务器)、广播接收器设备、广播发射器等,并且可以不使用或使用任何类型的操作系统。在一些情况下,源设备12和目的地设备14可以用于无线通信。因此,源设

备12和目的地设备14可以是无线通信设备。

[0124] 在一些情况下,图1A所示的视频编码及解码系统10仅为示例,本申请中的技术可以适用于不必包含编码和解码设备之间的任何数据通信的视频编码设置(例如,视频编码或视频解码)。在其它示例中,数据可从本地存储器检索、在网络上流式传输等。视频编码设备可以对数据进行编码并且将数据存储到存储器,和/或视频解码设备可以从存储器检索数据并且对数据进行解码。在一些示例中,由并不彼此通信而是仅编码数据到存储器和/或从存储器检索数据且解码数据的设备执行编码和解码。

[0125] 为便于描述,本文(例如)参考由ITU-T视频编码专家组(Video Coding Experts Group,VCEG)和ISO/IEC运动图像专家组(Motion Picture Experts Group,MPEG)的视频编码联合协作团队(Joint Collaboration Team on Video Coding,JCT-VC)开发的高效视频编码(High-Efficiency Video Coding,HEVC)或下一代视频编码标准通用视频编码(Versatile Video Coding,VVC)参考软件来描述本发明实施例。本领域普通技术人员理解本发明实施例不限于HEVC或VVC。

[0126] 编码器和编码方法

[0127] 图2为用于实现本申请中技术的示例性视频编码器20的示意性框图。在图2的示例中,视频编码器20包括输入端201(或输入接口201)、残差计算单元204、变换处理单元206、量化单元208、反量化单元210、逆变换处理单元212、重建单元214、环路滤波器220、解码图像缓冲区(decoded picture buffer,DPB)230、模式选择单元260、熵编码单元270和输出端272(或输出接口272)。模式选择单元260可以包括帧间预测单元244、帧内预测单元254和划分单元262。帧间预测单元244可以包括运动估计单元和运动补偿单元(未示出)。图2所示的视频编码器20也可以称为混合视频编码器或基于混合型视频编解码器的视频编码器。

[0128] 残差计算单元204、变换处理单元206、量化单元208和模式选择单元260可以组成编码器20的前向信号路径,而反量化单元210、逆变换处理单元212、重建单元214、缓冲区216、环路滤波器220、解码图像缓冲区(decoded picture buffer,DPB)230、帧间预测单元244和帧内预测单元254可以组成视频编码器20的后向信号路径,其中,视频编码器20的后向信号路径对应于解码器(参见图3中的视频解码器30)的信号路径。反量化单元210、逆变换处理单元212、重建单元214、环路滤波器220、解码图像缓冲区(decoded picture buffer,DPB)230、帧间预测单元244和帧内预测单元254还组成视频编码器20的“内置解码器”。

[0129] 图像和图像划分(图像和块)

[0130] 编码器20可以用于通过输入端201等接收图像17(或图像数据17)。图像17可以是组成视频或视频序列的一系列图像中的图像。接收到的图像或图像数据也可以是预处理图像19(或预处理图像数据19)。为简单起见,以下描述使用图像17。图像17还可以称为当前图像或待译码图像(尤其是在视频译码中为了将当前图像与同一视频序列(也就是同样包括当前图像的视频序列)中的其它图像(例如先前的经编码和/或解码图像)区分开)。

[0131] (数字)图像为或可以视为由具有强度值的样本(sample)组成的二维阵列或矩阵。阵列中的样本也可以称为像素(pixel/pel)(图像元素的简称)。阵列或图像的水平方向和垂直方向(或轴线)上的样本数量限定了图像的大小和/或分辨率。为了表示颜色,通常采用3个颜色分量,即图像可以表示为或可以包括3个样本阵列。在RGB格式或颜色空间中,一个

图像包括对应的红色、绿色和蓝色样本阵列。但是,在视频译码中,每个像素通常以亮度和色度格式或颜色空间表示,例如YCbCr,包括Y表示的亮度分量(有时也用L表示)和Cb和Cr表示的2个色度分量。亮度(luminance,简写为luma)分量Y表示亮度或灰度级强度(例如在灰度等级图像中两者相同),而2个色度(chrominance,简写为chroma)分量Cb和Cr表示色度或颜色信息分量。因此,YCbCr格式的图像包括由亮度样本值(sample value)(Y)组成的亮度样本阵列和2个由色度值(Cb和Cr)组成的色度样本阵列。RGB格式的图像可以转换或变换为YCbCr格式,反之亦然。该过程也称为颜色变换或转换。如果图像是黑白的,则该图像可以仅包括亮度样本阵列。相应地,例如,图像可以为黑白格式的亮度样本阵列或4:2:0、4:2:2和4:4:4颜色格式的亮度样本阵列和2个对应的色度样本阵列。

[0132] 视频编码器20的实施例可以包括图像划分单元(图2中未示出),用于将图像17划分成多个(通常不重叠的)图像块203。这些块也可以称为根块、宏块(H.264/AVC),或(H.265/HEVC和VVC中的)编码树块(coding tree block,CTB)或编码树单元(coding tree unit,CTU)。图像划分单元可以用于对视频序列中的所有图像使用相同的块大小和使用限定块大小的对应网格,或者在图像或图像子集或图像组之间改变块大小,并将每个图像划分成多个对应块。

[0133] 在其它实施例中,视频编码器可以用于直接接收图像17中的块203,例如组成图像17的一个、几个或所有块。图像块203也可以称为当前图像块或待译码图像块。

[0134] 与图像17类似,图像块203同样是或可以视为具有强度值(样本值)的像素点组成的二维阵列或矩阵,但是图像块203的尺寸比图像17的尺寸小。换句话说,根据所应用的颜色格式,块203可以包括(例如)一个样本阵列(例如黑白图像17情况下的亮度阵列或彩色图像情况下的亮度阵列或色度阵列)或3个样本阵列(例如彩色图像17情况下的1个亮度阵列和2个色度阵列)或任何其它数量和/或类型的阵列。块203的水平方向和垂直方向(或轴线)上的样本数量限定了块203的大小。相应地,一个块可以为 $M \times N$ (M列 $\times$ N行)的样本阵列,或 $M \times N$ 的变换系数阵列等。

[0135] 图2所示的视频编码器20的实施例可以用于逐块对图像17进行编码,例如对每个块203执行编码和预测。

[0136] 图2所示的视频编码器20的实施例还可以用于使用条带(slice)(也称为视频条带)对图像进行划分和/或编码。一个图像可以划分成一个或多个(通常不重叠的)条带或使用一个或多个(通常不重叠的)条带进行编码,每个条带可以包括一个或多个块(例如CTU)。

[0137] 图2所示的视频编码器20的实施例还可以用于使用分块(tile)组(也称为视频分块组)和/或分块(也称为视频分块)对图像进行划分和/或编码。一个图像可以划分成一个或多个(通常不重叠的)分块组或使用一个或多个(通常不重叠的)分块组进行编码;每个分块组可以包括一个或多个块(例如CTU)或一个或多个分块等;每个分块可以为矩形等并可以包括一个或多个完整或部分块(例如CTU)等。

[0138] 残差计算

[0139] 残差计算单元204可以用于通过以下方式根据图像块203和预测块265(后续详细介绍了预测块265)来计算残差块205(也称为残差205)以得到样本域中的残差块205:例如,逐个样本(逐个像素)从图像块203的样本值中减去预测块265的样本值。

[0140] 变换

[0141] 变换处理单元206可以用于在残差块205的样本值上应用离散余弦变换(discrete cosine transform,DCT)或离散正弦变换(discrete sine transform,DST)等变换,以在变换域中获取变换系数207。变换系数207也可以称为变换残差系数,表示变换域中的残差块205。

[0142] 变换处理单元206可以用于应用DCT/DST(例如为H.265/HEVC指定的变换)的整数近似值。与正交DCT变换相比,这种整数近似值通常由某一因子按比例缩放。为了维持经正变换和逆变换处理的残差块的范数,应用其它比例缩放因子作为变换过程的一部分。比例缩放因子通常是某些约束条件选择的,例如,比例缩放因子是用于移位运算的2的幂、变换系数的位深度、精度与实现成本之间的权衡等。例如,通过逆变换处理单元212等为逆变换(以及在视频解码器30侧,通过逆变换处理单元312等为对应的逆变换)指定具体比例缩放因子;相应地,可以在编码器20侧通过变换处理单元206等为正变换指定对应比例缩放因子。

[0143] 视频编码器20(相应地,变换处理单元206)的实施例可以用于直接输出或通过熵编码单元270进行编码或压缩后输出变换参数(例如,一种或多种变换的类型),使得视频解码器30可以接收并使用变换参数进行解码,等等。

[0144] 量化

[0145] 量化单元208可以用于通过应用标量量化或矢量量化等来量化变换系数207,得到经量化系数209。经量化系数209也可以称为经量化变换系数209或经量化残差系数209。

[0146] 量化过程可以减小与部分或全部变换系数207相关的位深度。例如,可以在量化期间将n位变换系数向下舍入到m位变换系数,其中,n大于m。可以通过调整量化参数(quantization parameter,QP)修改量化程度。例如,对于标量量化,可以应用不同的缩放来实现较细或较粗的量化。较小量化步长对应较细量化,而较大量化步长对应较粗量化。可以通过量化参数(quantization parameter,QP)指示合适的量化步长。例如,量化参数可以为合适的量化步长的预定义集合的索引。例如,较小的量化参数可以对应精细量化(较小量化步长),较大量化参数可以对应粗糙量化(较大量化步长),反之亦然。量化可以包含除以量化步长以及例如通过逆量化210执行的对应反量化或解量化,或者可以包含乘以量化步长。根据HEVC等一些标准的实施例可以使用量化参数来确定量化步长。一般而言,可以基于量化参数使用包括除法的等式的定点近似来计算量化步长。可以引入额外比例缩放因子来进行量化和反量化,以恢复可能由于在用于量化步长和量化参数的等式的定点近似中使用的缩放而修改的残差块的范数。在一种示例性实现方式中,可以合并逆变换和解量化的缩放。可选地,可以使用自定义量化表并在码流等中将其从编码器通过信号发送到解码器。量化是有损操作,其中,量化步长越大,损耗越大。

[0147] 视频编码器20(相应地,量化单元208)的实施例可以用于直接输出或通过熵编码单元270进行编码后输出量化参数(quantization parameter,QP),使得视频解码器30可以接收并使用量化参数进行解码,等等。

[0148] 反量化

[0149] 反量化单元210用于在经量化系数上应用量化单元208的反量化,得到经解量化系数211,例如根据或使用与量化单元208相同的量化步长,应用量化单元208应用的量化方案的反量化方案。经解量化系数211也可以称为经反量化残差系数211,对应于变换系数207,

但是由于量化造成的损耗,通常与变换系数不相同。

[0150] 逆变换

[0151] 逆变换处理单元212用于应用变换处理单元206应用的变换的逆变换,例如,逆离散余弦变换(discrete cosine transform,DCT)或逆离散正弦变换(discrete sine transform,DST)或其它逆变换,以在样本域中获取经重建残差块213(或对应的经解量化系数213)。经重建残差块213也可以称为变换块213。

[0152] 重建

[0153] 重建单元214(例如加法器或求和器214)用于通过以下方式将变换块213(即经重建残差块213)添加到预测块265,以在样本域中获取经重建块215:例如,逐个样本将经重建残差块213的样本值和预测块265的样本值相加。

[0154] 滤波

[0155] 环路滤波器单元220(或简称“环路滤波器”220)用于对经重建块215进行滤波,获得经滤波块221,或通常用于对经重建样本进行滤波,得到经滤波样本值。例如,环路滤波器单元用于顺利进行像素转变或以其它方式提高视频质量。环路滤波器单元220可以包括一个或多个环路滤波器,例如去块效应滤波器、样本自适应偏移(sample-adaptive offset, SAO)滤波器或一个或多个其它滤波器,例如双边滤波器、自适应环路滤波器(adaptive loop filter, ALF)、锐化或平滑滤波器、协同滤波器或其任意组合。虽然环路滤波器单元220在图2中示出为环内滤波器,但在其它配置中,环路滤波器单元220可以实施为环后滤波器。经滤波块221也可以称为经滤波的经重建块221。

[0156] 视频编码器20(相应地,环路滤波器单元220)的实施例可以用于输出环路滤波器参数(例如样本自适应偏移信息),例如直接输出或由熵编码单元270进行编码后输出,使得解码器30可以接收并应用相同的环路滤波器参数或相应的环路滤波器用于解码。

[0157] 解码图像缓冲区

[0158] 解码图像缓冲区(decoded picture buffer, DPB)230可以是存储参考图像或通常存储参考图像数据供视频编码器20在对视频数据进行编码时使用的存储器。DPB 230可以由多种存储器设备中的任一种形成,例如动态随机存取存储器(dynamic random access memory, DRAM),包括同步DRAM(synchronous DRAM, SDRAM)、磁阻RAM(magnetoresistive RAM, MRAM)、电阻RAM(resistive RAM, RRAM)或其它类型的存储器设备。解码图像缓冲区(decoded picture buffer, DPB)230可以用于存储一个或多个经滤波块221。解码图像缓冲区230还可以用于存储同一当前图像或不同图像(例如先前的经重建图像)中的其它先前经滤波块(例如先前经滤波的经重建块221),并可以提供先前完整的经重建(即经解码)图像(和对应的参考块和样本)和/或部分重建的当前图像(和对应的参考块和样本),以进行帧间预测等。解码图像缓冲区(decoded picture buffer, DPB)230还可以用于,如果经重建块215未由环路滤波器单元220进行滤波,存储一个或多个未经滤波的经重建块215,或通常存储未经滤波的经重建样本;用于存储未进行任何其它进一步处理的经重建块或经重建样本。

[0159] 模式选择(划分和预测)

[0160] 模式选择单元260包括划分单元262、帧间预测单元244和帧内预测单元254,并且用于从解码图像缓冲区230或其它缓冲区(例如行缓冲区,图中未显示)接收或获取原始块

203(当前图像17的当前块203)等原始图像数据以及经重建图像数据(例如同一个(当前)图像和/或一个或多个先前的经解码图像的经滤波和/或未经滤波的经重建样本或经重建块)。经重建图像数据用作帧间预测或帧内预测等预测所需的参考图像数据,得到预测块265或预测值265。

[0161] 模式选择单元260可以用于为当前块预测模式(包括不划分)确定或选择一种划分方式以及确定或选择一种预测模式(例如帧内预测模式或帧间预测模式),生成对应的预测块265,以对残差块205进行计算和对经重建块215进行重建。

[0162] 模式选择单元260的实施例可以用于选择划分方式和预测模式(例如,从模式选择单元260支持或可用的那些模式中选择)。所述预测模式提供最佳匹配或者说最小残差(最小残差意味着传输或存储中更好的压缩),或提供最小信令开销(最小信令开销意味着传输或存储中更好的压缩),或同时考虑或平衡以上两者。模式选择单元260可以用于根据率失真优化(rate distortion optimization,RDO)确定划分方式和预测模式,即选择提供最小率失真的预测模式。本文中的“最佳”、“最小”、“最优”等术语不一定指总体上“最佳”、“最小”、“最优”等,但也可以指满足终止或选择标准的情况,例如,超过或低于阈值的值或其它约束条件可能导致“次优选择”,但会降低复杂度且减少处理时间。

[0163] 换句话说,划分单元262可以用于通过以下方式将块203划分成更小的块分区(partition)或子块(再次形成块):例如,通过迭代使用四叉树(quad-tree,QT)划分、二叉树(binary-tree,BT)划分或三叉树(triple-tree,TT)划分或其任意组合,并且用于对块分区或子块中的每一个执行预测等,其中,模式选择包括选择划分块203的树结构,并且对块分区或子块中的每一个使用预测模式。

[0164] 下文将详细地描述由示例性视频编码器20执行的划分(例如由划分单元260执行)和预测处理(由帧间预测单元244和帧内预测单元254执行)。

[0165] 划分

[0166] 划分单元262可以将当前块203划分(或分割)成更小的分区,例如正方形或矩形大小的较小块。可以将这些较小块(也可以称为子块)进一步划分成甚至更小的分区。这也称为树划分或分层树划分。在根树级别0(层次级别0、深度0)等的根块可以递归地划分成两个或更多下一个较低树级别的块,例如树级别1(层次级别1、深度1)的节点。这些块可以又划分成两个或更多下一个较低级别的块,例如树级别2(层次级别2、深度2)等,直到划分结束(因为满足结束标准,例如达到最大树深度或最小块尺寸)。未进一步划分的块也称为树的叶块或叶节点。划分成两个分区的树称为二叉树(binary-tree,BT),划分成3个分区的树称为三叉树(ternary-tree,TT),划分成4个分区的树称为四叉树(quad-tree,QT)。

[0167] 如上所述,本文使用的术语“块”可以是图像的一部分,特别是正方形或矩形部分。参考HEVC和VVC等,块可以为或可以对应于编码树单元(coding tree unit,CTU)、编码单元(coding unit,CU)、预测单元(prediction unit,PU)和变换单元(transform unit,TU),和/或对应于多个对应块,例如编码树块(coding tree block,CTB)、编码块(coding block,CB)、变换块(transform block,TB)或预测块(prediction block,PB)。

[0168] 例如,编码树单元(coding tree unit,CTU)可以为或可以包括具有3个样本阵列的图像中的亮度样本组成的一个CTB以及色度样本组成的两个对应CTB,或者可以为或可以包括黑白图像或使用3个单独颜色平面和语法结构进行译码的图像中的样本组成的一个

CTB。这些语法结构用于对样本进行译码。相应地,编码树块(coding tree block,CTB)可以为 $N \times N$ 个样本块,其中, N 可以设为某个值,使得一个分量分成多个CTB,这就是一种划分方式。编码单元(coding unit,CU)可以为或可以包括具有3个样本阵列的图像中的亮度样本组成的一个编码块以及色度样本组成的两个对应编码块,或者黑白图像或使用3个单独颜色平面和语法结构进行译码的图像中的样本组成的一个编码块。这些语法结构用于对样本进行译码。相应地,编码块(coding block,CB)可以为 $M \times N$ 个样本块,其中, M 和 N 可以设为某个值,使得一个CTB分成多个编码块,这就是一种划分方式。

[0169] 在实施例中,例如根据HEVC,可以使用表示为编码树的二叉树结构将编码树单元(coding tree unit,CTU)分割成多个CU。在CU层级决定是否使用帧间(时间)预测或帧内(空间)预测对图像区域进行译码。每个CU可以根据PU分割类型进一步分割成1个、2个或4个PU。一个PU内进行相同的预测过程,并以PU为单位向解码器发送相关信息。在根据PU分割类型进行预测过程得到残差块之后,可以根据类似于用于CU的编码树的其它二叉树结构将CU划分成变换单元(transform unit,TU)。

[0170] 在实施例中,例如根据当前开发的称为通用视频编码(Versatile Video Coding,VVC)的最新视频编码标准,使用二叉树结合二叉树(quad-tree and binary-tree,QTBT)划分等来划分编码块。在QTBT块结构中,一个CU可以为正方形或矩形。例如,编码树单元(coding tree unit,CTU)首先通过二叉树结构进行划分。二叉树叶节点进一步通过二叉树或三叉(ternary/triple)树结构进行划分。划分树叶节点称为编码单元(coding unit,CU),这种分割(segmentation)用于预测和变换处理,无需任何进一步划分。这表示在QTBT编码块结构中,CU、PU和TU的块大小相同。与此同时,三叉树划分等多重划分可以与QTBT块结构一起使用。

[0171] 在一个示例中,视频编码器20中的模式选择单元260可以用于执行本文描述的划分技术的任意组合。

[0172] 如上所述,视频编码器20用于从(预定的)预测模式集合中确定或选择最佳或最优的预测模式。预测模式集合可以包括帧内预测模式和/或帧间预测模式等。

[0173] 帧内预测

[0174] 帧内预测模式集合可以包括35种不同的帧内预测模式,例如像DC(或均值)模式和平面模式的非方向性模式或如HEVC中定义的方向性模式,或者可以包括67种不同的帧内预测模式,例如像DC(或均值)模式和平面模式的非方向性模式或如VVC中定义的方向性模式。

[0175] 帧内预测单元254用于根据帧内预测模式集合中的帧内预测模式,使用同一当前图像的相邻块的经重建样本来生成帧内预测块265。

[0176] 帧内预测单元254(或总称为模式选择单元260)还用于将帧内预测参数(或总称为表示块的选定帧内预测模式的信息)以语法元素266的形式输出到熵编码单元270,以包含在经编码图像数据21中,使得视频解码器30可以接收并使用预测参数进行解码,等等。

[0177] 帧间预测

[0178] (可能的)帧间预测模式集合取决于可用参考图像(即(例如)上述存储在DPB 230中的至少部分经解码图像)和其它帧间预测参数,例如取决于是否使用整个参考图像或只使用参考图像的一部分(例如当前块的区域周围的搜索窗口区域)来搜索最佳匹配参考块,和/或例如取决于是否进行像素插值(例如二分之一/半像素插值和/或四分之一像素插

值)。

[0179] 除上述预测模式之外,还可以使用跳过模式和/或直接模式。

[0180] 帧间预测单元244可以包括运动估计(motion estimation,ME)单元和运动补偿(motion compensation,MC)单元(两者在图2中未示出)。运动估计单元可以用于接收或获取图像块203(当前图像17的当前图像块203)和经解码图像231,或者至少一个或多个先前的经重建块(例如一个或多个其它/不同的先前经解码图像231的经重建块),以进行运动估计。例如,视频序列可以包括当前图像和先前经解码图像231,或换句话说,当前图像和先前经解码图像231可以为一系列图像的一部分或组成一系列图像,这一系列图像组成视频序列。

[0181] 例如,编码器20可以用于从多个其它图像中的同一或不同图像的多个参考块中选择一个参考块,并将参考图像(或参考图像索引)和/或参考块的位置(x坐标、y坐标)与当前块的位置之间的偏移(空间偏移)作为帧间预测参数提供给运动估计单元。这种偏移也称为运动矢量(motion vector,MV)。

[0182] 运动补偿单元用于获取(例如接收)帧间预测参数,并根据或使用帧间预测参数进行帧间预测,得到帧间预测块265。由运动补偿单元执行的运动补偿可以包括根据通过运动估计确定的运动/块矢量来提取或生成预测块,还可以包括执行插值以获得子像素精度。插值滤波可以根据已知像素样本生成其它像素样本,从而可能增加可以用于对图像块进行译码的候选预测块的数量。一旦接收到当前图像块的PU对应的运动矢量,运动补偿单元可以在其中一个参考图像列表中定位运动矢量指向的预测块。

[0183] 运动补偿单元还可以生成与块和视频条带相关的语法元素,以供视频解码器30在解码视频条带的图像块时使用。除条带和相应语法元素之外或作为条带和相应语法元素的替代,还可以生成或使用分块组和/或分块以及相应语法元素。

[0184] 熵编码

[0185] 熵编码单元270用于将熵编码算法或方案(例如可变长度编码(variable length coding,VLC)方案、上下文自适应VLC(context adaptive VLC,CAVLC)方案、算术编码方案、二值化,上下文自适应二进制算术编码(context adaptive binary arithmetic coding,CABAC)、基于语法的上下文自适应二进制算术编码(syntax-based context-adaptive binary arithmetic coding,SBAC)、概率区间划分熵(probability interval partitioning entropy,PIPE)编码或其它熵编码方法或技术)等应用于或不应用于(无压缩)经量化系数209、帧间预测参数、帧内预测参数、环路滤波器参数和/或其它语法元素,得到可以通过输出端272以经编码码流21等形式输出的经编码图像数据21,使得视频解码器30可以接收并使用这些参数进行解码,等等。可以将经编码码流21发送给视频解码器30,或者将其存储在存储器中稍后由视频解码器30发送或检索。

[0186] 视频编码器20的其它结构变型可以用于对视频流进行编码。例如,基于非变换的编码器20可以在没有变换处理单元206的情况下为某些块或帧直接量化残差信号。在另一种实现方式中,编码器20可以包括组合成单个单元的量化单元208和反量化单元210。

[0187] 解码器和解码方法

[0188] 图3为用于实现本申请中技术的视频解码器30的一个示例。视频解码器30用于接收(例如)由编码器20编码的经编码图像数据21(例如经编码码流21),得到经解码图像331。

经编码图像数据或码流包括用于解码该经编码图像数据的信息,例如表示经编码视频条带(和/或分块组或分块)的图像块的数据和相关语法元素。

[0189] 在图3的示例中,解码器30包括熵解码单元304、反量化单元310、逆变换处理单元312、重建单元314(例如求和器314)、环路滤波器320、解码图像缓冲区(decoded picture buffer,DPB)330、模式应用单元360、帧间预测单元344和帧内预测单元354。帧间预测单元344可以为或可以包括运动补偿单元。在一些示例中,视频解码器30可以执行大体上与参照图2中的视频编码器100描述的编码过程互逆的解码过程。

[0190] 如参照编码器20所述,反量化单元210、逆变换处理单元212、重建单元214、环路滤波器220、解码图像缓冲区(decoded picture buffer,DPB)230、帧间预测单元344和帧内预测单元354还组成视频编码器20的“内置解码器”。相应地,反量化单元310在功能上可以与反量化单元110相同,逆变换处理单元312在功能上可以与逆变换处理单元212相同,重建单元314在功能上可以与重建单元214相同,环路滤波器320在功能上可以与环路滤波器220相同,解码图像缓冲区330在功能上可以与解码图像缓冲区230相同。因此,视频编码器20的相应单元和功能的解释相应地适用于视频解码器30的相应单元和功能。

[0191] 熵解码

[0192] 熵解码单元304用于解析码流21(或通常称为经编码图像数据21)并对经编码图像数据21执行熵解码等,得到经量化系数309和/或经解码编码参数(图3中未示出)等,例如帧间预测参数(例如参考图像索引和运动矢量)、帧内预测参数(例如帧内预测模式或索引)、变换参数、量化参数、环路滤波器参数和/或其它语法元素中的任一个或全部。熵解码单元304可以用于应用与参照编码器20中的熵编码单元270描述的编码方案对应的解码算法或方案。熵解码单元304还可以用于将帧间预测参数、帧内预测参数和/或其它语法元素提供给模式应用单元360,并将其它参数提供给解码器30中的其它单元。视频解码器30可以接收视频条带级和/或视频块级的语法元素。除条带和相应的语法元素之外或作为条带和相应的语法元素的替代,还可以接收和/或使用分块组和/或分块以及相应语法元素。

[0193] 反量化

[0194] 反量化单元310可以用于从经编码图像数据21(例如通过熵解码单元304等解析和/或解码)接收量化参数(quantization parameter,QP)(或通常称为与反量化相关的信息)和经量化系数,并根据这些量化参数对经解码量化系数309进行反量化,得到经解量化系数311。经解量化系数311也可以称为变换系数311。反量化过程可以包括使用视频编码器20为视频条带(或分块或分块组)中的每个视频块确定的量化参数来确定量化程度,同样也确定需要应用的反量化的程度。

[0195] 逆变换

[0196] 逆变换处理单元312可以用于接收经解量化系数311(也称为变换系数311),并对经解量化系数311进行变换,得到样本域中的经重建残差块213。经重建残差块213也可以称为变换块313。变换可以为逆变换,例如逆DCT、逆DST、逆整数变换或概念上类似的逆变换过程。逆变换处理单元312还可以用于(例如通过熵解码单元304等解析和/或解码)从经编码图像数据21接收变换参数或对应的信息,以确定要对经解量化系数311进行的变换。

[0197] 重建

[0198] 重建单元314(例如加法器或求和器314)可以用于通过以下方式将经重建残差块

313添加到预测块365以得到样本域中的经重建块315:例如,将经重建残差块313的样本值和预测块365的样本值相加。

[0199] 滤波

[0200] 环路滤波器单元320(在译码环路中或之后)用于对经重建块315进行滤波,得到经滤波块321,从而顺利进行像素转变或以其它方式提高视频质量等。环路滤波器单元320可以包括一个或多个环路滤波器,例如去块效应滤波器、样本自适应偏移(sample-adaptive offset,SAO)滤波器或一个或多个其它滤波器,例如双边滤波器、自适应环路滤波器(adaptive loop filter,ALF)、锐化或平滑滤波器、协同滤波器或其任意组合。虽然环路滤波器单元320在图3中示出为环内滤波器,但在其它配置中,环路滤波器单元320可以实施为环后滤波器。

[0201] 解码图像缓冲区

[0202] 随后将一个图像的经解码视频块321存储在解码图像缓冲区330中,解码图像缓冲区330存储经解码图像331作为参考图像,以便后续对其它图像进行运动补偿和/或输出或显示。

[0203] 解码器30用于通过输出端312等输出经解码图像311,向用户显示或供用户观看。

[0204] 预测

[0205] 帧间预测单元344在功能上可以与帧间预测单元244(特别是与运动补偿单元)相同,帧内预测单元354在功能上可以与帧内预测单元254相同,并根据从经编码图像数据21(例如通过熵解码单元304等解析和/或解码)接收的划分方式和/或预测参数或相应的信息来执行分割或划分决策和执行预测。模式应用单元360可以用于根据经重建图像、块或相应的样本(经滤波或未经滤波)对每个块执行预测(帧内预测或帧间预测),得到预测块365。

[0206] 当视频条带被译码为经帧内译码(I)条带时,模式应用单元360中的帧内预测单元354用于根据指示(signal)的帧内预测模式和来自当前图像的先前经解码块的数据生成当前视频条带的图像块的预测块365。当视频图像被译码为经帧间译码(即B或P)条带时,模式应用单元360中的帧间预测单元344(例如运动补偿单元)用于根据运动矢量和从熵解码单元304接收的其它语法元素为当前视频条带的视频块生成预测块365。对于帧间预测,可以根据其中一个参考图像列表内的其中一个参考图像产生这些预测块。视频解码器30可以根据存储在DPB 330中的参考图像,使用默认构建技术来构建参考帧列表0和列表1。除条带(例如视频条带)之外或作为条带的替代,相同或类似的过程可以应用于使用分块组(例如视频分块组)和/或分块(例如视频分块)的实施例或由这些实施例应用,例如,视频可以使用I、P或B分块组和/或分块进行译码。

[0207] 模式应用单元360用于通过解析运动矢量或相关信息以及其它语法元素为当前视频条带的视频块确定预测信息,并使用该预测信息为正在解码的当前视频块生成预测块。例如,模式应用单元360使用接收到的一些语法元素来确定用于对视频条带的视频块进行译码的预测模式(例如帧内或帧间预测)、帧间预测条带类型(例如B条带、P条带或GPB条带)、用于条带的一个或多个参考图像列表的构建信息、用于条带的每个经帧间编码视频块的运动矢量、用于条带的每个经帧间译码视频块的帧间预测状态以及其它信息,以对当前视频条带中的视频块进行解码。除条带(例如视频条带)之外或作为条带的替代,相同或类似的过程可以应用于使用分块组(例如视频分块组)和/或分块(例如视频分块)的实施例或

由这些实施例应用,例如,视频可以使用I、P或B分块组和/或分块进行译码。

[0208] 图3中所示的视频解码器30的实施例可以用于使用条带(也称为视频条带)对图像进行划分和/或解码。一个图像可以划分成一个或多个(通常不重叠的)条带或使用一个或多个(通常不重叠的)条带进行解码,每个条带可以包括一个或多个块(例如CTU)。

[0209] 在实施例中,图3所示的视频解码器30可以用于使用分块组(也称为视频分块组)和/或分块(也称为视频分块)对图像进行划分和/或解码。一个图像可以划分成一个或多个(通常不重叠的)分块组或使用一个或多个(通常不重叠的)分块组进行解码;每个分块组可以包括一个或多个块(例如CTU) 或一个或多个分块等;每个分块可以为矩形等并可以包括一个或多个完整或部分块(例如CTU)等。

[0210] 视频解码器30的其它变型可以用于对经编码图像数据21进行解码。例如,解码器30可以在没有环路滤波器单元320的情况下生成输出视频流。例如,基于非变换的解码器30可以在没有逆变换处理单元312的情况下针对某些块或帧直接反量化残差信号。在另一种实现方式中,视频解码器30可以包括组合成单个单元的反量化单元310和逆变换处理单元312。

[0211] 应当理解的是,在编码器20和解码器30中,可以对当前步骤的处理结果做进一步处理,然后输出到下一步骤。例如,在插值滤波、运动矢量推导或环路滤波之后,可以对插值滤波、运动矢量推导或环路滤波的处理结果进行进一步运算,例如限幅(Clip)或移位(shift)运算。

[0212] 需要说明的是,可以对当前块的推导运动矢量(包括但不限于仿射模式的控制点运动矢量,仿射模式、平面模式、ATMVP模式的子块运动矢量,时间运动矢量等)进行进一步运算。例如,根据运动矢量的表示位将运动矢量的值限制在预定义范围。如果运动矢量的表示位为bitDepth,则范围为 $-2^{(\text{bitDepth}-1)} \sim 2^{(\text{bitDepth}-1)}-1$,其中,“ $\sim$ ”表示幂次方。例如,如果bitDepth设置为16,则范围为-32768~32767;如果bitDepth设置为18,则范围为-131072~131071。例如,对推导出的运动矢量(例如,一个 8×8 块中的4个 4×4 子块的MV)的值进行限制,使得这4个 4×4 子块MV的整数部分之间的最大差值不超过N个像素,例如不超过1个像素。这里提供了两种根据bitDepth来限制运动矢量的方法。

[0213] 方法1:通过平滑运算来去除溢出的最高有效位(most significant bit,MSB)

$$[0214] \quad ux = (mvx + 2^{\text{bitDepth}}) \% 2^{\text{bitDepth}} \quad (1)$$

$$[0215] \quad mvx = (ux \geq 2^{\text{bitDepth}-1}) ? (ux - 2^{\text{bitDepth}}) : ux \quad (2)$$

$$[0216] \quad uy = (mvy + 2^{\text{bitDepth}}) \% 2^{\text{bitDepth}} \quad (3)$$

$$[0217] \quad mvy = (uy \geq 2^{\text{bitDepth}-1}) ? (uy - 2^{\text{bitDepth}}) : uy \quad (4)$$

[0218] 其中,mvx为一个图像块或子块的运动矢量的水平分量;mvy为一个图像块或子块的运动矢量的垂直分量;ux和uy表示中间值。

[0219] 例如,如果mvx的值为-32769,则使用公式(1)和(2)之后得到的值为32767。在计算机系统中,以二进制补码的形式存储十进数。-32769的二进制补码为1,0111,1111,1111,1111(17位),这时丢弃MSB,那么得到的二进制补码为0111,1111,1111,1111(十进数为32767),这与使用公式(1)和(2)之后得到的输出结果相同。

$$[0220] \quad ux = (mvpx + mvd_x + 2^{\text{bitDepth}}) \% 2^{\text{bitDepth}} \quad (5)$$

$$[0221] \quad mvx = (ux \geq 2^{\text{bitDepth}-1}) ? (ux - 2^{\text{bitDepth}}) : ux \quad (6)$$

[0222] $uy = (mvpy + mvdy + 2^{\text{bitDepth}}) \% 2^{\text{bitDepth}}$ (7)

[0223] $mv_y = (uy \geq 2^{\text{bitDepth}-1}) ? (uy - 2^{\text{bitDepth}}) : uy$ (8)

[0224] 这些运算可以在对mvp和mvd求和的过程中执行,如公式(5)至(8)所示。

[0225] 方法2:对值进行限幅来去除溢出的MSB

[0226] $vx = \text{Clip3}(-2^{\text{bitDepth}-1}, 2^{\text{bitDepth}-1}-1, vx)$

[0227] $vy = \text{Clip3}(-2^{\text{bitDepth}-1}, 2^{\text{bitDepth}-1}-1, vy)$

[0228] 其中,vx为一个图像块或子块的运动矢量的水平分量,vy为一个图像块或子块的运动矢量的垂直分量;x、y和z分别与MV限幅过程的3个输入值对应,函数Clip3的定义如下:

[0229]
$$\text{Clip3}(x, y, z) = \begin{cases} x & ; z < x \\ y & ; z > y \\ z & ; \text{其它} \end{cases}$$

[0230] 图4为本发明一个实施例提供的视频译码设备400的示意图。视频译码设备400适用于实现本文描述的公开实施例。在一个实施例中,视频译码设备400可以是解码器(例如图1A中的视频解码器30)或编码器(例如图1A中的视频编码器20)。

[0231] 视频译码设备400包括:用于接收数据的入端口410(或输入端口410)和接收单元(Rx)420;用于处理所述数据的处理器、逻辑单元或中央处理器(central processing unit,CPU)430;用于发送所述数据的发送单元(Tx)440和出端口450(或输出端口450);用于存储所述数据的存储器460。视频译码设备400还可以包括包括与入端口410、接收单元420、发送单元440和出端口450耦合的光电(optical-to-electrical,OE)组件和电光(electrical-to-optical,E0)组件,用作光信号或电信号的出口或入口。

[0232] 处理器430通过硬件和软件来实现。处理器430可以实现为一个或多个CPU芯片、一个或多个核(例如多核处理器)、一个或多个FPGA、一个或多个ASIC和一个或多个DSP。处理器430与入端口410、接收单元420、发送单元440、出端口450和存储器460通信。处理器430包括译码模块470。译码模块470实现上文描述的公开实施例。例如,译码模块470执行、处理、准备或提供各种译码操作。因此,将译码模块470包含在内为视频译码设备400的功能提供了实质性的改进,并且影响了视频译码设备400到不同状态的转换。可选地,以存储在存储器460中并由处理器430执行的指令来实现译码模块470。

[0233] 存储器460可以包括一个或多个磁盘、一个或多个磁带机以及一个或多个固态硬盘,并且可以用作溢出数据存储设备,以在选择程序来执行时存储这些程序以及存储在执行程序过程中读取的指令和数据。例如,存储器460可以是易失性和/或非易失性的,并且可以是只读存储器(read-only memory,ROM)、随机存取存储器(random access memory,RAM)、三态内容寻址存储器(ternary content-addressable memory,TCAM)和/或静态随机存取存储器(static random-access memory,SRAM)。

[0234] 图5为一个示例性实施例提供的装置500的简化框图。装置500可以用作图1A的源设备12和目的地设备14中的任一个或两个。

[0235] 装置500中的处理器502可以是中央处理单元。可选地,处理器502可以是现有的或今后将开发出的能够操作或处理信息的任何其它类型的设备或多个设备。虽然所公开的实现方式可以使用如图所示的处理器502等单个处理器来实施,但使用多个处理器可以提高速度和效率。

[0236] 在一种实现方式中,装置500中的存储器504可以是只读存储器(read only memory,ROM)设备或随机存取存储器(random access memory,RAM)设备。任何其它合适类型的存储设备都可以用作存储器504。存储器504可以包括处理器502通过总线512访问的代码和数据506。存储器504还可包括操作系统508和应用程序510,应用程序510包括至少一个程序,这个程序使得处理器502执行本文所述方法。例如,应用程序510可以包括应用1至N,还可以包括执行本文所述方法的视频译码应用。

[0237] 装置500还可以包括一个或多个输出设备,例如显示器518。在一个示例中,显示器518可以是将显示器与触敏元件组合的触敏显示器,该触敏元件能够用于感测触摸输入。显示器518可以通过总线512耦合到处理器502。

[0238] 虽然装置500中的总线512在本文中描述为单个总线,但是总线512可以包括多个总线。此外,辅助存储器514可以直接耦合到装置500中的其它组件或可以通过网络被访问,并且可以包括单个集成单元(例如一个存储卡)或多个单元(例如多个存储卡)。因此,装置500可以通过多种配置实现。

[0239] 三角划分模式(triangular partitioning mode,TPM)和几何运动划分(GEO)还分别称为三角融合模式和几何融合模式,它们是实现预测分区之间的非水平边界和非垂直边界的划分技术,其中,预测单元PU1和预测单元PU1使用它们与不同颜色分量相关的样本子集的加权平均过程在区域中组合。TPM能够使预测分区之间的边界沿着矩形块对角线,而根据GEO的边界可以位于任意位置上。在应用加权平均过程的区域中,正方形内的整数数字表示应用于预测单元PU1的亮度分量的权重 W_{PU1} 。在一个示例中,应用于预测单元PU2的亮度分量的权重 W_{PU2} 计算如下:

[0240] $W_{PU2} = 8 - W_{PU1}$ 。

[0241] 应用于对应预测单元的色度分量的权重可能不同于应用于对应预测单元的亮度分量的权重。

[0242] 关于TPM语法的详细信息如表1所示,其中,4个语法元素用于指示关于TPM的信息:

[0243] MergeTriangleFlag是确定是否选择TPM的标志(“0”表示不选择TPM;否则,选择TPM);

[0244] merge_triangle_split_dir是TPM的分割方向标志(“0”表示分割方向是从左上角到右下角;否则,分割方向是从右上角到左下角);

[0245] merge_triangle_idx0和merge_triangle_idx1是用于TPM的融合候选0和1的索引。

[0246] 表1:包括TPM语法的融合数据语法

[0247]

merge_data(x0, y0, cbWidth, cbHeight) {	描述符
if (CuPredMode[x0][y0] == MODE_IBC) {	
if(MaxNumMergeCand > 1)	
merge_idx [x0][y0]	ae(v)
} else {	
if(sps_mmvd_enabled_flag cbWidth * cbHeight != 32)	
regular_merge_flag [x0][y0]	ae(v)
if (regular_merge_flag[x0][y0] == 1){	
if(MaxNumMergeCand > 1)	
merge_idx [x0][y0]	ae(v)
} else {	
if(sps_mmvd_enabled_flag && cbWidth * cbHeight != 32)	
mmvd_merge_flag [x0][y0]	ae(v)
if(mmvd_merge_flag[x0][y0] == 1) {	
if(MaxNumMergeCand > 1)	

[0248]

mmvd_cand_flag[x0][y0]	ae(v)
mmvd_distance_idx[x0][y0]	ae(v)
mmvd_direction_idx[x0][y0]	ae(v)
} else {	
if(MaxNumSubblockMergeCand > 0 && cbWidth >= 8 && cbHeight >= 8)	
merge_subblock_flag[x0][y0]	ae(v)
if(merge_subblock_flag[x0][y0] == 1) {	
if(MaxNumSubblockMergeCand > 1)	
merge_subblock_idx[x0][y0]	ae(v)
} else {	
if(sps_ciip_enabled_flag && cu_skip_flag[x0][y0] == 0 && (cbWidth * cbHeight) >= 64 && cbWidth < 128 && cbHeight < 128) {	
ciip_flag[x0][y0]	ae(v)
if(ciip_flag[x0][y0] && MaxNumMergeCand > 1)	
merge_idx[x0][y0]	ae(v)
}	
if(MergeTriangleFlag[x0][y0]) {	
merge_triangle_split_dir[x0][y0]	ae(v)
merge_triangle_idx0[x0][y0]	ae(v)
merge_triangle_idx1[x0][y0]	ae(v)
}	
}	
}	
}	
}	
}	

[0249] 在一个示例中,TPM在以下提案中详述:R-L.Liao和C.S.Lim发表在2018年10月中国澳门的第12次JVET会议的文献JVET-L0124中的“CE10.3.1.b:Triangular prediction unit mode (CE10.3.1.b:三角预测单元模式)”。GEO在以下论文中解释说明:S.Esenlik、H.Gao、A.Filippov、V.Rufitskiy、A.M.Kotra、B.Wang、E.Alshina、M. Bläser和J.Sauer发表在2019年7月瑞典哥德堡的第15次JVET会议的文献JVET-00489中的“Non-CE4:Geometrical partitioning for inter blocks(非CE4:帧间块的几何划分)”。

[0250] 所公开的使TPM和/或GEO与WP协调的方法是在应用WP时禁用TPM和/或GEO。第一实施方式如表2所示,检查编码单元(coding unit)的变量weightedPredFlag的值是否等于0。

[0251] 变量weightedPredFlag的推导如下:

[0252] -如果slice_type为P,则weightedPredFlag被设置为pps_weighted_pred_flag。

[0253] -否则(slice_type为B),weightedPredFlag被设置为pps_weighted_bipred_flag。

[0254] 加权预测过程可以使用语法元素pps_weighted_pred_flag和sps_weighted_pred_flag在图像级和条带级之间切换。

[0255] 如上所述,变量weightedPredFlag表示条带级加权预测是否可以在获取条带的帧间预测样本时使用。

[0256] 表2:使TPM与WP协调的融合数据语法

[0257]	merge_data(x0, y0, cbWidth, cbHeight, chType) {	描述符
	if (CuPredMode[chType][x0][y0] == MODE_IBC) {	
	if(MaxNumIbcMergeCand > 1)	
	merge_idx [x0][y0]	ac(v)
	} else {	
	if(MaxNumSubblockMergeCand > 0 && cbWidth >= 8 && cbHeight >= 8)	
	merge_subblock_flag [x0][y0]	ac(v)
	if(merge_subblock_flag[x0][y0] == 1) {	
	if(MaxNumSubblockMergeCand > 1)	
	merge_subblock_idx [x0][y0]	ac(v)
	} else {	
	if((cbWidth * cbHeight) >= 64 && ((sps_ciip_enabled_flag && cu_skip_flag[x0][y0] == 0 && cbWidth < 128 && cbHeight < 128) (sps_triangle_enabled_flag && MaxNumTriangleMergeCand > 1 && slice_type == B)))	
	regular_merge_flag [x0][y0]	ac(v)
	if (regular_merge_flag[x0][y0] == 1) {	
	if(sps_mmvd_enabled_flag)	
	mmvd_merge_flag [x0][y0]	ac(v)
	if(mmvd_merge_flag[x0][y0] == 1) {	
	if(MaxNumMergeCand > 1)	
	mmvd_cand_flag [x0][y0]	ac(v)
	mmvd_distance_idx [x0][y0]	ac(v)
	mmvd_direction_idx [x0][y0]	ac(v)
	} else {	
	if(MaxNumMergeCand > 1)	

	merge_idx[x0][y0]	ae(v)
	}	
	} else {	
	if(sps_ciip_enabled_flag && sps_triangle_enabled_flag && MaxNumTriangleMergeCand > 1 && weightedPredFlag == 0 && slice_type == B && cu_skip_flag[x0][y0] == 0 && (cbWidth * cbHeight) >= 64 && cbWidth < 128 && cbHeight < 128) {	
	ciip_flag[x0][y0]	ae(v)
	if(ciip_flag[x0][y0] && MaxNumMergeCand > 1)	
[0258]	merge_idx[x0][y0]	ae(v)
	if(!ciip_flag[x0][y0] && MaxNumTriangleMergeCand > 1) {	
	merge_triangle_split_dir[x0][y0]	ae(v)
	merge_triangle_idx0[x0][y0]	ae(v)
	merge_triangle_idx1[x0][y0]	ae(v)
	}	
	}	
	}	
	}	
	}	

[0259] ciip_flag[x0][y0]表示组合的帧间融合和帧内预测是否应用于当前编码单元(coding unit)。阵列索引x0和y0表示当前编码块的左上亮度样本相对于图像的左上亮度样本的位置(x0,y0)。

[0260] 当ciip_flag[x0][y0]不存在时,ciip_flag[x0][y0]的推断如下:

[0261] -如果以下所有条件都为真(true),则ciip_flag[x0][y0]被推断为1:

[0262] -sps_ciip_enabled_flag等于1,

[0263] -general_merge_flag[x0][y0]等于1,

[0264] -merge_subblock_flag[x0][y0]等于0,

[0265] -regular_merge_flag[x0][y0]等于0,

[0266] -cbWidth小于128,

[0267] -cbHeight小于128,

[0268] -cbWidth*cbHeight大于或等于64;

[0269] -否则,ciip_flag[x0][y0]被推断为0。

[0270] 当ciip_flag[x0][y0]等于1时,变量IntraPredModeY[x][y] (其中,x=x0..x0+cbWidth-1,y=y0..y0+cbHeight-1)被设置为INTRA_PLANAR。

[0271] 变量MergeTriangleFlag[x0][y0]表示在解码B条带时,基于三角形的运动补偿是否用于生成当前编码单元的预测样本。该变量的推导如下:

[0272] -如果以下所有条件都为真,则MergeTriangleFlag[x0][y0]被设置为1:

[0273] -sps_triangle_enabled_flag等于1,

- [0274] -slice_type为B,
- [0275] -general_merge_flag[x0][y0]等于1,
- [0276] -MaxNumTriangleMergeCand大于或等于2,
- [0277] -cbWidth*cbHeight大于或等于64,
- [0278] -regular_merge_flag[x0][y0]等于0,
- [0279] -merge_subblock_flag[x0][y0]等于0,
- [0280] -ciip_flag[x0][y0]等于0,
- [0281] -weightedPredFlag等于0;
- [0282] -否则,MergeTriangleFlag[x0][y0]被设置为0。
- [0283] 第二实现方式如表3所示。如果weightedPredFlag等于1,则语法元素max_num_merge_cand_minus_max_num_triangle_cand不存在,并被推断具有一个值,使得MaxNumTriangleMergeCand小于2。

[0284] 表3:使TPM与WP协调的通用条带头语法

[0285]	slice_header() {	描述符
	slice_pic_parameter_set_id	ue(v)
	if(rect_slice_flag NumBricksInPic > 1)	
	slice_address	u(v)
	if(!rect_slice_flag && !single_brick_per_slice_flag)	
	num_bricks_in_slice_minus1	ue(v)
	non_reference_picture_flag	u(1)
	slice_type	ue(v)
	if(separate_colour_plane_flag == 1)	
	colour_plane_id	u(2)
	slice_pic_order_cnt_lsb	u(v)
	if(nal_unit_type == GDR_NUT)	
	recovery_poc_cnt	ue(v)
	if(nal_unit_type == IDR_W_RADL nal_unit_type == IDR_N_LP nal_unit_type == CRA_NUT NalUnitType == GDR_NUT)	
	no_output_of_prior_pics_flag	u(1)
	if(output_flag_present_flag)	

[0286]

pic_output_flag	u(1)
if((nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) sps_idr_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
ref_pic_list_sps_flag[i]	u(1)
if(ref_pic_list_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
ref_pic_list_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
slice_poc_lsb_lf[i][j]	u(v)
delta_poc_msb_present_flag[i][j]	u(1)
if(delta_poc_msb_present_flag[i][j])	
delta_poc_msb_cycle_lf[i][j]	ue(v)
}	
}	
if((slice_type != I && num_ref_entries[0][RplIdx[0]] > 1) (slice_type == B && num_ref_entries[1][RplIdx[1]] > 1)) {	
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag)	
for(i = 0; i < (slice_type == B ? 2 : 1); i++)	
if(num_ref_entries[i][RplIdx[i]] > 1)	
num_ref_idx_active_minus1[i]	ue(v)
}	

[0287]

}	
if(partition_constraints_override_enabled_flag) {	
partition_constraints_override_flag	ue(v)
if(partition_constraints_override_flag) {	
slice_log2_diff_min_qt_min_cb_luma	ue(v)
slice_max_mtt_hierarchy_depth_luma	ue(v)
if(slice_max_mtt_hierarchy_depth_luma != 0)	
slice_log2_diff_max_bt_min_qt_luma	ue(v)
slice_log2_diff_max_tt_min_qt_luma	ue(v)
}	
if(slice_type == I && qtbtt_dual_tree_intra_flag) {	
slice_log2_diff_min_qt_min_cb_chroma	ue(v)
slice_max_mtt_hierarchy_depth_chroma	ue(v)
if(slice_max_mtt_hierarchy_depth_chroma != 0)	
slice_log2_diff_max_bt_min_qt_chroma	ue(v)
slice_log2_diff_max_tt_min_qt_chroma	ue(v)
}	
}	
}	
}	
if(slice_type != I) {	
if(sps_temporal_mvp_enabled_flag && !pps_temporal_mvp_enabled_idc)	
slice_temporal_mvp_enabled_flag	u(1)
if(slice_type == B && !pps_mv_d_11_zero_idc)	
mv_d_11_zero_flag	u(1)
if(cabac_init_present_flag)	
cabac_init_flag	u(1)
if(slice_temporal_mvp_enabled_flag) {	

[0288]

if(slice_type == B && !pps_collocated_from_l0_idc)	
collocated_from_l0_flag	u(1)
if((collocated_from_l0_flag && NumRefIdxActive[0] > 1) (!collocated_from_l0_flag && NumRefIdxActive[1] > 1))	
collocated_ref_idx	ue(v)
}	
if((pps_weighted_pred_flag && slice_type == P) (pps_weighted_bipred_flag && slice_type == B))	
pred_weight_table()	
if(!pps_six_minus_max_num_merge_cand_plus1)	
six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag && !pps_five_minus_max_num_subblock_merge_cand_plus1)	
five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
slice_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_dmvr_slice_present_flag)	
slice_disable_bdof_dmvr_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && slice_type == B && !weightedPredFlag && !pps_max_num_merge_cand_minus_max_num_triangle_cand_minus1) {	
max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
}	
}	
if(sps_ibc_enabled_flag)	
slice_six_minus_max_num_ibc_merge_cand	ue(v)
if(sps_joint_cbr_enabled_flag)	
slice_joint_cbr_sign_flag	u(1)
slice_qp_delta	se(v)

[0289]

if(pps_slice_chroma_qp_offsets_present_flag) {	
slice_cb_qp_offset	se(v)
slice_cr_qp_offset	se(v)
if(sps_joint_cbr_enabled_flag)	
slice_joint_cbr_qp_offset	se(v)
}	
if(sps_sao_enabled_flag) {	
slice_sao_luma_flag	u(1)
if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
if(sps_alf_enabled_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
}	
}	
if(!pps_dep_quant_enabled_flag)	
dep_quant_enabled_flag	u(1)
if(!dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag)	

[0290]

deblocking_filter_override_flag	u(1)
if(deblocking_filter_override_flag) {	
slice_deblocking_filter_disabled_flag	u(1)
if(!slice_deblocking_filter_disabled_flag) {	
slice_beta_offset_div2	se(v)
slice_tc_offset_div2	se(v)
}	
}	
if(sps_lmcs_enabled_flag) {	
slice_lmcs_enabled_flag	u(1)
if(slice_lmcs_enabled_flag) {	
slice_lmcs_aps_id	u(2)
if(ChromaArrayType != 0)	
slice_chroma_residual_scale_flag	u(1)
}	
}	
if(sps_scaling_list_enabled_flag) {	
slice_scaling_list_present_flag	u(1)
if(slice_scaling_list_present_flag)	
slice_scaling_list_aps_id	u(3)
}	
if(entry_point_offsets_present_flag && NumEntryPoints > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumEntryPoints; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)

[0291]	for(i = 0; i < slice_header_extension_length; i++)	
	slice_header_extension_data_byte[i]	u(8)
	}	
	byte_alignment()	
	}	

[0292] 具体地,以下语义可以用于第二实现方式:

[0293] max_num_merge_cand_minus_max_num_triangle_cand表示从MaxNumMergeCand中减去条带中支持的三角融合模式候选的最大数量。

[0294] 当max_num_merge_cand_minus_max_num_triangle_cand不存在,sps_triangle_enabled_flag等于1,slice_type为B,weightedPredFlag等于0,且MaxNumMergeCand大于或等于2时,max_num_merge_cand_minus_max_num_triangle_cand被推断为pps_max_num_merge_cand_minus_max_num_triangle_cand_minus1+1。

[0295] 当max_num_merge_cand_minus_max_num_triangle_cand不存在,sps_triangle_enabled_flag等于1,slice_type为B,weightedPredFlag等于1,且MaxNumMergeCand大于或等于2时,max_num_merge_cand_minus_max_num_triangle_cand被推断为MaxNumMergeCand或xNumMergeCand-1。

[0296] 三角融合模式候选的最大数量MaxNumTriangleMergeCand推导如下:

[0297] MaxNumTriangleMergeCand=MaxNumMergeCand-

[0298] max_num_merge_cand_minus_max_num_triangle_cand

[0299] 当max_num_merge_cand_minus_max_num_triangle_cand存在时,MaxNumTriangleMergeCand的取值范围应为2至MaxNumMergeCand(包括端值)。

[0300] 当max_num_merge_cand_minus_max_num_triangle_cand不存在(sps_triangle_enabled_flag等于0或MaxNumMergeCand小于2)时,MaxNumTriangleMergeCand被设置为0。

[0301] 当MaxNumTriangleMergeCand等于0时,不允许三角融合模式用于当前条带。

[0302] 本发明公开的机制不仅适用于TPM和GE0,而且还适用于其它非矩形预测和划分模式,例如使用三角分区的组合的帧内帧间预测。

[0303] 由于TPM和GE0只应用于B条带,因此上述实施例中的变量weightedPredFlag可以直接由变量pps_weighted_bipred_flag代替。

[0304] 第三实现方式如表6所示,检查编码单元(coding unit)的变量weightedPredFlag的值是否等于0。

[0305] 变量weightedPredFlag的推导如下:

[0306] -如果以下所有条件都为真,则weightedPredFlag被设置为0:

[0307] luma_weight_10_flag[i]等于0,其中,i的范围为0至NumRefIdxActive[0],

[0308] luma_weight_11_flag[i]等于0,其中,i的范围为0至NumRefIdxActive[1],

[0309] chroma_weight_10_flag[i]等于0,其中,i的范围为0至NumRefIdxActive[0],

[0310] chroma_weight_11_flag[i]等于0,其中,i的范围为0至NumRefIdxActive[1];

[0311] -否则,weightedPredFlag被设置为1。

[0312] weightedPredFlag的推导过程表示:如果亮度分量和色度分量的所有加权标志以及当前条带的所有参考索引的所有加权标志都为0,则加权预测在当前条带中禁用;否则,加权预测可以用于当前条带。

[0313] 如上所述,变量weightedPredFlag表示条带级加权预测是否可以在获取条带的帧间预测样本时使用。

[0314] 第四实现方式如表2所示,其中,weightedPredFlag由slice_weighted_pred_flag代替,slice_weighted_pred_flags是在条带头中指示的,如表4所示。

[0315] 如上所述,语法slice_weighted_pred_flag表示条带级加权预测是否可以在获取条带的帧间预测样本时使用。

[0316] 表4:指示条带级加权预测标志的通用条带头语法

[0317]	slice_header() {	描述符
	slice_pic_parameter_set_id	ue(v)
	if(rect_slice_flag NumBricksInPic > 1)	
	slice_address	u(v)
	if(!rect_slice_flag && !single_brick_per_slice_flag)	
	num_bricks_in_slice_minus1	ue(v)
	non_reference_picture_flag	u(1)
	slice_type	ue(v)
	if(separate_colour_plane_flag == 1)	
	colour_plane_id	u(2)
	slice_pic_order_cnt_lsb	u(v)
	if(nal_unit_type == GDR_NUT)	
	recovery_poc_cnt	ue(v)
	if(nal_unit_type == IDR_W_RADL nal_unit_type == IDR_N_LP nal_unit_type == CRA_NUT NalUnitType == GDR_NUT)	
	no_output_of_prior_pics_flag	u(1)
	if(output_flag_present_flag)	
	pic_output_flag	u(1)
	if((nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) sps_idr_rpl_present_flag) {	
	for(i = 0; i < 2; i++) {	

[0318]

if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
ref_pic_list_sps_flag[i]	u(1)
if(ref_pic_list_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
ref_pic_list_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
slice_poc_lsb_lt[i][j]	u(v)
delta_poc_msb_present_flag[i][j]	u(1)
if(delta_poc_msb_present_flag[i][j])	
delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
if((slice_type != I && num_ref_entries[0][RplIdx[0]] > 1) (slice_type == B && num_ref_entries[1][RplIdx[1]] > 1)) {	
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag)	
for(i = 0; i < (slice_type == B ? 2 : 1); i++)	
if(num_ref_entries[i][RplIdx[i]] > 1)	
num_ref_idx_active_minus1[i]	ue(v)
}	
}	
if(partition_constraints_override_enabled_flag) {	
partition_constraints_override_flag	ue(v)
if(partition_constraints_override_flag) {	

[0319]

slice_log2_diff_min_qt_min_cb_luma	ue(v)
slice_max_mtt_hierarchy_depth_luma	ue(v)
if(slice_max_mtt_hierarchy_depth_luma != 0)	
slice_log2_diff_max_bt_min_qt_luma	ue(v)
slice_log2_diff_max_tt_min_qt_luma	ue(v)
}	
if(slice_type == I && qtbtt_dual_tree_intra_flag) {	
slice_log2_diff_min_qt_min_cb_chroma	ue(v)
slice_max_mtt_hierarchy_depth_chroma	ue(v)
if(slice_max_mtt_hierarchy_depth_chroma != 0)	
slice_log2_diff_max_bt_min_qt_chroma	ue(v)
slice_log2_diff_max_tt_min_qt_chroma	ue(v)
}	
}	
}	
}	
if (slice_type != I) {	
if(sps_temporal_mvp_enabled_flag && !pps_temporal_mvp_enabled_idc)	
slice_temporal_mvp_enabled_flag	u(1)
if(slice_type == B && !pps_mv_d_11_zero_idc)	
mv_d_11_zero_flag	u(1)
if(cabac_init_present_flag)	
cabac_init_flag	u(1)
if(slice_temporal_mvp_enabled_flag) {	
if(slice_type == B && !pps_collocated_from_l0_idc)	
collocated_from_l0_flag	u(1)
if((collocated_from_l0_flag && NumRefIdxActive[0] > 1) (!collocated_from_l0_flag && NumRefIdxActive[1] > 1))	

[0320]

collocated_ref_idx	ue(v)
}	
if((pps_weighted_pred_flag && slice_type == P) (pps_weighted_bipred_flag && slice_type == B))	
slice_weighted_pred_flag	u(1)
if(slice_weighted_pred_flag)	
pred_weight_table()	
if(!pps_six_minus_max_num_merge_cand_plus1)	
six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag && !pps_five_minus_max_num_subblock_merge_cand_plus1)	
five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
slice_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_dmvr_slice_present_flag)	
slice_disable_bdof_dmvr_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_minus1) {	
max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
}	
}	
if(sps_ibc_enabled_flag)	
slice_six_minus_max_num_ibc_merge_cand	ue(v)
if(sps_joint_cbr_enabled_flag)	
slice_joint_cbr_sign_flag	u(1)
slice_qp_delta	se(v)
if(pps_slice_chroma_qp_offsets_present_flag) {	
slice_cb_qp_offset	se(v)
slice_cr_qp_offset	se(v)

[0321]

if(sps_joint_cbr_enabled_flag)	
slice_joint_cbr_qp_offset	se(v)
}	
if(sps_sao_enabled_flag) {	
slice_sao_luma_flag	u(1)
if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
if(sps_alf_enabled_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
}	
}	
if(!pps_dep_quant_enabled_flag)	
dep_quant_enabled_flag	u(1)
if(!dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag)	
deblocking_filter_override_flag	u(1)
if(deblocking_filter_override_flag) {	
slice_deblocking_filter_disabled_flag	u(1)

[0322]

if(!slice_deblocking_filter_disabled_flag) {	
slice_beta_offset_div2	se(v)
slice_tc_offset_div2	se(v)
}	
}	
if(sps_lmcs_enabled_flag) {	
slice_lmcs_enabled_flag	u(1)
if(slice_lmcs_enabled_flag) {	
slice_lmcs_aps_id	u(2)
if(ChromaArrayType != 0)	
slice_chroma_residual_scale_flag	u(1)
}	
}	
if(sps_scaling_list_enabled_flag) {	
slice_scaling_list_present_flag	u(1)
if(slice_scaling_list_present_flag)	
slice_scaling_list_aps_id	u(3)
}	
if(entry_point_offsets_present_flag && NumEntryPoints > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumEntryPoints; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)
for(i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte[i]	u(8)
}	

[0323]

byte_alignment()	
}	

[0324] 具体地,以下语义可以用于第四实现方式:

[0325] slice_weighted_pred_flag等于0表示加权预测不应用于当前条带。slice_weighted_pred_flag等于1表示加权预测应用于当前条带。当slice_weighted_pred_flag不存在时,slice_weighted_pred_flag的值被推断为0。

[0326] 第五实现方式是通过一致性约束禁用块级TPM。在TPM编码块(coded block)的情况下,帧间预测值 P_0 710和 P_1 720(如图7所示)对应的参考图像的亮度分量和色度分量的加权因子可以不存在。

[0327] 关于更多细节,refIdxA和predListFlagA表示帧间预测值 P_0 的参考索引和参考图像列表;refIdxB和predListFlagB表示帧间预测值 P_1 的参考索引和参考图像列表。

[0328] 变量lumaWeightedFlag和chromaWeightedFlag的推导如下:

[0329] lumaWeightedFlagA=predListFlagA?luma_weight_l1_flag[refIdxA]:luma_weight_l0_flag[refIdxA]

[0330] lumaWeightedFlagB=predListFlagB?luma_weight_l1_flag[refIdxB]:luma_weight_l0_flag[refIdxB]

[0331] chromaWeightedFlagA=predListFlagA?chroma_weight_l1_flag[refIdxA]:chroma_weight_l0_flag[refIdxA]

[0332] chromaWeightedFlagB=predListFlagB?chroma_weight_l1_flag[refIdxB]:chroma_weight_l0_flag[refIdxB]

[0333] lumaWeightedFlag=lumaWeightedFlagA||lumaWeightedFlagB

[0334] chromaWeightedFlag=chromaWeightedFlagA||chromaWeightedFlagB

[0335] 码流一致性的要求为:lumaWeightedFlag和chromaWeightedFlag可以等于0。

[0336] 第六实现方式是在使用显式加权预测时,对TPM编码块(coded block)禁用混合加权样本预测过程。

[0337] 图7和图8分别说明了TPM和GE0的示例。需要说明的是,TPM的实施例也可以用于GE0模式。

[0338] 在TPM编码块的情况下,如果帧间预测值 P_0 710或 P_1 720对应的参考图像的亮度分量或色度分量的加权因子存在,则根据WP参数(P_0 和 P_1 的WP参数730 $\{w_0, 0_0\}$ 和WP参数740 $\{w_1, 0_1\}$)的加权过程用于生成帧间预测值块;否则,根据混合加权参数的加权过程用于生成块750的帧间预测值。如图9所示,帧间预测值901需要两个预测块 P_0 911和 P_1 912,这两个预测块具有重叠区域921,其中,非零权重应用于块911和912,以部分混合预测值 P_0 911和 P_1 912。在图9中,与块901相邻的块表示为931、932、933、934、935和936。图8示出了TPM融合模式和GE0融合模式之间的一些区别。在GE0融合模式的情况下,预测值851和852之间的重叠区域不仅仅可以位于帧间预测块850的对角线上。预测值 P_0 851和 P_1 852的接收可以通过将权重和偏移 $\{w_0, 0_0\}$ 830和 $\{w_1, 0_1\}$ 840分别应用于或不应用于块810和820,从其它图像中复制块810和820来实现。

[0339] 在一个示例中,refIdxA和predListFlagA表示帧间预测值 P_0 的参考索引和参考图像列表;refIdxB和predListFlagB表示帧间预测值 P_1 的参考索引和参考图像列表。

[0340] 变量lumaWeightedFlag和chromaWeightedFlag的推导如下:

[0341] lumaWeightedFlagA=predListFlagA?luma_weight_l1_flag[refIdxA]:luma_weight_l0_flag[refIdxA]

[0342] $\text{lumaWeightedFlagB} = \text{predListFlagB} ? \text{luma_weight_l1_flag}[\text{refIdxB}] : \text{luma_weight_l0_flag}[\text{refIdxB}]$

[0343] $\text{chromaWeightedFlagA} = \text{predListFlagA} ? \text{chroma_weight_l1_flag}[\text{refIdxA}] : \text{chroma_weight_l0_flag}[\text{refIdxA}]$

[0344] $\text{chromaWeightedFlagB} = \text{predListFlagB} ? \text{chroma_weight_l1_flag}[\text{refIdxB}] : \text{chroma_weight_l0_flag}[\text{refIdxB}]$

[0345] $\text{lumaWeightedFlag} = \text{lumaWeightedFlagA} \mid \text{lumaWeightedFlagB}$

[0346] $\text{chromaWeightedFlag} = \text{chromaWeightedFlagA} \mid \text{chromaWeightedFlagB}$

[0347] 然后, 如果 lumaWeightedFlag 为真, 则显式加权过程被调用; 如果 lumaWeightedFlag 为假 (false), 则混合加权过程被调用。此外, 色度分量也由 $\text{chromaWeightedFlag}$ 确定。

[0348] 在一个可选实现方式中, 统一使用所有分量的加权标志。如果 lumaWeightedFlag 或 $\text{chromaWeightedFlag}$ 中的一个为真, 则显式加权过程被调用; 如果 lumaWeightedFlag 和 $\text{chromaWeightedFlag}$ 都为假, 则混合加权过程被调用。

[0349] 使用双向预测机制预测的矩形块的显式加权过程如下所述执行。

[0350] 该过程的输入包括:

[0351] - 两个变量 $nCbW$ 和 $nCbH$, 表示当前编码块 (coding block) 的宽度和高度;

[0352] - 两个 $(nCbW) \times (nCbH)$ 阵列 predSamplesA 和 predSamplesB ;

[0353] - 预测列表标志 predListFlagA 和 predListFlagB ;

[0354] - 参考索引 refIdxA 和 refIdxB ;

[0355] - 变量 $cIdx$, 表示颜色分量索引;

[0356] - 样本位深度 bitDepth 。

[0357] 该过程的输出为预测样本值组成的 $(nCbW) \times (nCbH)$ 阵列 pbSamples 。

[0358] 变量 shift1 被设置为 $\text{Max}(2, 14 - \text{bitDepth})$ 。

[0359] 变量 $\log2Wd$ 、 $o0$ 、 $o1$ 、 $w0$ 和 $w1$ 的推导如下:

[0360] - 如果亮度样本的 $cIdx$ 等于 0, 则以下内容适用:

[0361] $\log2Wd = \text{luma_log2_weight_denom} + \text{shift1}$

[0362] $w0 = \text{predListFlagA} ? \text{LumaWeightL1}[\text{refIdxA}] : \text{LumaWeightL0}[\text{refIdxA}]$

[0363] $w1 = \text{predListFlagB} ? \text{LumaWeightL1}[\text{refIdxB}] : \text{LumaWeightL0}[\text{refIdxB}]$

[0364] $o0 = (\text{predListFlagA} ? \text{luma_offset_l1}[\text{refIdxA}] : \text{luma_offset_l0}[\text{refIdxA}]) \ll (\text{BitDepth}_y - 8)$

[0365] $o1 = (\text{predListFlagB} ? \text{luma_offset_l1}[\text{refIdxB}] : \text{luma_offset_l0}[\text{refIdxB}]) \ll (\text{BitDepth}_y - 8)$

[0366] - 否则 (色度样本的 $cIdx$ 不等于 0), 以下内容适用:

[0367] $\log2Wd = \text{ChromaLog2WeightDenom} + \text{shift1}$

[0368] $w0 = \text{predListFlagA} ? \text{ChromaWeightL1}[\text{refIdxA}][cIdx - 1] :$

[0369] $\text{ChromaWeightL0}[\text{refIdxA}][cIdx - 1]$

[0370] $w1 = \text{predListFlagA} ? \text{ChromaWeightL1}[\text{refIdxB}][cIdx - 1] : \text{ChromaWeightL0}[\text{refIdxB}][cIdx - 1]$

[0371] $o0 = (\text{predListFlagA} ? \text{ChromaOffsetL1}[\text{refIdxA}][\text{cIdx}-1] : \text{ChromaOffsetL0}[\text{refIdxA}][\text{cIdx}-1]) \ll (\text{BitDepth}_c - 8)$

[0372] $o1 = (\text{predListFlagB} ? \text{ChromaOffsetL1}[\text{refIdxB}][\text{cIdx}-1] : \text{ChromaOffsetL0}[\text{refIdxB}][\text{cIdx}-1]) \ll (\text{BitDepth}_c - 8)$

[0373] 预测样本 $\text{pbSamples}[x][y]$ (其中, $x=0..nCbW-1, y=0..nCbH-1$) 的推导如下:

[0374] $\text{pbSamples}[x][y] = \text{Clip3}(0, (1 \ll \text{bitDepth}) - 1, (\text{predSamplesA}[x][y] * w0 + \text{predSamplesB}[x][y] * w1 + ((o0 + o1 + 1) \ll \log2Wd)) \gg (\log2Wd + 1))$

[0375] 条带级加权预测的参数可以表示为一组变量, 这些变量被分配给参考图像列表中的每个元素。元素的索引进一步表示为“i”。这些参数可以包括:

[0376] - $\text{LumaWeightL0}[i]$

[0377] - $\text{luma_offset_10}[i]$ 是应用于使用 $\text{RefPicList}[0][i]$ 对列表0进行预测所使用的亮度预测值的附加偏移。 $\text{luma_offset_10}[i]$ 的取值范围应为-128至127 (包括端值)。当 $\text{luma_weight_10_flag}[i]$ 等于0时, $\text{luma_offset_10}[i]$ 被推断为0。

[0378] 变量 $\text{LumaWeightL0}[i]$ 被推导为 $(1 \ll \text{luma_log2_weight_denom}) + \text{delta_luma_weight_10}[i]$ 。当 $\text{luma_weight_10_flag}[i]$ 等于1时, $\text{delta_luma_weight_10}[i]$ 的取值范围应为-128至127 (包括端值)。当 $\text{luma_weight_10_flag}[i]$ 等于0时, $\text{LumaWeightL0}[i]$ 被推断为 $2^{\text{luma_log2_weight_denom}}$ 。

[0379] 使用双向预测机制预测的矩形块的混合加权过程如下所述执行。

[0380] 该过程的输入包括:

[0381] - 两个变量 $nCbW$ 和 $nCbH$, 表示当前编码块 (coding block) 的宽度和高度;

[0382] - 两个 $(nCbW) \times (nCbH)$ 阵列 predSamplesLA 和 predSamplesLB ;

[0383] - 变量 triangleDir , 表示划分方向;

[0384] - 变量 $cIdx$, 表示颜色分量索引。

[0385] 该过程的输出为预测样本值组成的 $(nCbW) \times (nCbH)$ 阵列 pbSamples 。

[0386] 变量 $nCbR$ 的推导如下:

[0387] $nCbR = (nCbW > nCbH) ? (nCbW / nCbH) : (nCbH / nCbW)$

[0388] 变量 bitDepth 的推导如下:

[0389] - 如果 $cIdx$ 等于0, 则 bitDepth 被设置为 BitDepth_y 。

[0390] - 否则, bitDepth 被设置为 BitDepth_c 。

[0391] 变量 shift1 和 offset1 的推导如下:

[0392] - 变量 shift1 被设置为 $\text{Max}(5, 17 - \text{bitDepth})$ 。

[0393] - 变量 offset1 被设置为 $1 \ll (\text{shift1} - 1)$ 。

[0394] 根据 triangleDir 、 wS 和 $cIdx$ 的值, 预测样本 $\text{pbSamples}[x][y]$ (其中, $x=0..nCbW-1, y=0..nCbH-1$) 的推导如下:

[0395] - 变量 $wIdx$ 的推导如下:

[0396] - 如果 $cIdx$ 等于0且 triangleDir 等于0, 则以下内容适用:

[0397] $wIdx = (nCbW > nCbH) ? (\text{Clip3}(0, 8, (x / nCbR - y) + 4)) : (\text{Clip3}(0, 8, (x - y / nCbR) + 4))$

[0398] - 如果 $cIdx$ 等于0且 triangleDir 等于1, 则以下内容适用:

[0399] $wIdx = (nCbW > nCbH) ? (\text{Clip3}(0, 8, (nCbH - 1 - x / nCbR - y) + 4)) : (\text{Clip3}(0, 8, (nCbW - 1 -$

$x-y/nCbR)+4))$

[0400] -如果cIdx大于0且triangleDir等于0,则以下内容适用:

[0401] $wIdx = (nCbW > nCbH) ? (Clip3(0, 4, (x/nCbR - y) + 2)) : (Clip3(0, 4, (x - y/nCbR) + 2))$

[0402] -否则(如果cIdx大于0且triangleDir等于1),以下内容适用:

[0403] $wIdx = (nCbW > nCbH) ? (Clip3(0, 4, (nCbH - 1 - x/nCbR - y) + 2)) (Clip3(0, 4, (nCbW - 1 - x - y/nCbR) + 2))$

[0404] -表示预测样本权重的变量wValue使用wIdx和cIdx推导如下:

[0405] $wValue = (cIdx == 0) ? Clip3(0, 8, wIdx) : Clip3(0, 8, wIdx * 2)$

[0406] -预测样本值的推导如下:

[0407] $pbSamples[x][y] = Clip3(0, (1 \ll bitDepth) - 1, (predSamplesLA[x][y] * wValue + predSamplesLB[x][y] * (8 - wValue) + offset1) \gg shift1)$

[0408] 对于几何模式,使用双向预测机制预测的矩形块的混合加权过程如下所述执行。

[0409] 该过程的输入包括:

[0410] -两个变量nCbW和nCbH,表示当前编码块(coding block)的宽度和高度;

[0411] -两个 $(nCbW) \times (nCbH)$ 阵列predSamplesLA和predSamplesLB;

[0412] -变量angleIdx,表示几何分区的角度索引;

[0413] -变量distanceIdx,表示几何分区的距离索引;

[0414] -变量cIdx,表示颜色分量索引。

[0415] 该过程的输出包括预测样本值组成的 $(nCbW) \times (nCbH)$ 阵列pbSamples和变量partIdx。

[0416] 变量bitDepth的推导如下:

[0417] -如果cIdx等于0,则bitDepth被设置为BitDepth_y。

[0418] -否则,bitDepth被设置为BitDepth_c。

[0419] 变量shift1和offset1的推导如下:

[0420] -变量shift1被设置为Max(5, 17-bitDepth)。

[0421] -变量offset1被设置为 $1 \ll (shift1 - 1)$ 。

[0422] 亮度的权重阵列sampleWeight_L[x][y]和色度的权重阵列sampleWeight_C[x][y](其中, $x=0..nCbW-1, y=0..nCbH-1$)的推导如下:

[0423] 以下变量的值设置如下:

[0424] -hwRatio被设置为nCbH/nCbW

[0425] -displacementX被设置为angleIdx

[0426] -displacementY被设置为 $(displacementX + 8) \% 32$

[0427] -partIdx被设置为 $angleIdx \geq 13 \&\& angleIdx \leq 27 ? 1 : 0$

[0428] -rho通过表示为Dis的查找表(如表8-12所述)被设置为以下值:

[0429] $\rho = (Dis[displacementX] \ll 8) + (Dis[displacementY] \ll 8)$

[0430] 如果以下条件之一为真,则变量shiftHor被设置为0:

[0431] angleIdx%16等于8,

[0432] angleIdx%16不等于0且hwRatio ≥ 1 ;

[0433] 否则,shiftHor被设置为1。

[0434] 如果shiftHor等于0,则offsetX和offsetY的推导如下:

[0435] $\text{offsetX} = (256 - \text{nCbW}) \gg 1$,

[0436] $\text{offsetY} = (256 - \text{nCbH}) \gg 1 + \text{angleIdx} < 16 ? (\text{distanceIdx} * \text{nCbH}) \gg 3 : -((\text{distanceIdx} * \text{nCbH}) \gg 3)$

[0437] 如果shiftHor等于1,则offsetX和offsetY的推导如下:

[0438] $\text{offsetX} = (256 - \text{nCbW}) \gg 1 + \text{angleIdx} < 16 ? (\text{distanceIdx} * \text{nCbW}) \gg 3 : -((\text{distanceIdx} * \text{nCbW}) \gg 3)$

[0439] $\text{offsetY} = (256 - \text{nCbH}) \gg 1$

[0440] 变量weightIdx和weightIdxAbs通过查找表(表9)(其中, $x=0..nCbW-1, y=0..nCbH-1$)计算如下:

[0441] $\text{weightIdx} = (((x + \text{offsetX}) \ll 1) + 1) * \text{Dis}[\text{displacementX}] + (((y + \text{offsetY}) \ll 1) + 1) * \text{Dis}[\text{displacementY}] - \text{rho}$ 。

[0442] $\text{weightIdxAbs} = \text{Clip3}(0, 26, \text{abs}(\text{weightIdx}))$ 。

[0443] $\text{sampleWeight}_L[x][y]$ (其中, $x=0..nCbW-1, y=0..nCbH-1$) 的值根据表示为GeoFilter的表10设置如下:

[0444] $\text{sampleWeight}_L[x][y] = \text{weightIdx} \leq 0 ? \text{GeoFilter}[\text{weightIdxAbs}] : 8 - \text{GeoFilter}[\text{weightIdxAbs}]$

[0445] $\text{sampleWeight}_C[x][y]$ (其中, $x=0..nCbW-1, y=0..nCbH-1$) 的值设置如下:

[0446] $\text{sampleWeight}_C[x][y] = \text{sampleWeight}_L[(x \ll (\text{SubWidthC} - 1))] [(y \ll (\text{SubHeightC} - 1))]$

[0447] 注: $\text{sampleWeight}_L[x][y]$ 的值也可以从 $\text{sampleWeight}_L[x - \text{shiftX}][y - \text{shiftY}]$ 中推导出。如果angleIdx大于4且小于12,或者angleIdx大于20且小于24,则shiftX为分割角度的正切,而shiftY为1;否则,shiftX为1,而shiftY为分割角度的余切。如果正切(相对于余切)值是无穷大,则shiftX为1(相对于0)或shiftY为0(相对于1)。

[0448] 预测样本值的推导如下,其中,X表示为L或C,cIdx等于0或不等于0:

[0449] $\text{pbSamples}[x][y] = \text{partIdx} ? \text{Clip3}(0, (1 \ll \text{bitDepth}) - 1, (\text{predSamplesLA}[x][y] * (8 - \text{sampleWeight}_X[x][y]) + \text{predSamplesLB}[x][y] * \text{sampleWeight}_X[x][y] + \text{offset1}) \gg \text{shift1}) : \text{Clip3}(0, (1 \ll \text{bitDepth}) - 1, (\text{predSamplesLA}[x][y] * \text{sampleWeight}_X[x][y] + \text{predSamplesLB}[x][y] * (8 - \text{sampleWeight}_X[x][y]) + \text{offset1}) \gg \text{shift1})$

[0450] 表5:推导几何划分距离的查找表Dis

[0451]

idx	0	1	2	4	6	7	8	9	10	12	14	15
Dis[idx]	8	8	8	8	4	2	0	-2	-4	-8	-8	-8
idx	16	17	18	20	22	23	24	25	26	28	30	31
Dis[idx]	-8	-8	-8	-8	-4	-2	0	2	4	8	8	8

[0452] 表6:推导几何划分滤波器权重的滤波器权重查找表GeoFilter

[0453]

idx	0	1	2	3	4	5	6	7	8	9	10	11	12	13
GeoFilter[idx]	4	4	4	4	5	5	5	5	5	5	5	6	6	6
idx	14	15	16	17	18	19	20	21	22	23	24	25	26	
GeoFilter[idx]	6	6	6	6	7	7	7	7	7	7	7	7	8	

[0454] 在VVC规范草案7(文件JVET-P2001-vE:B.Bross、J.Chen、S.Liu、Y.-K.Wang发表的“Versatile Video Coding(Draft 7)(通用视频编码(草案7))”,即瑞士日内瓦的第16次JVET会议的输出文件JVET-P2001;该文件包含在文件JVET-P2001-v14:http://phenix.it-sudparis.eu/jvet/doc_end_user/documents/16_Geneva/wg11/JVET-P2001-v14.zip中)中,图像头(picture header,PH)的概念介绍如下:将一部分语法元素从条带头(slice header,SH)中移到PH中,以减少由于将相同或类似值分配给与PH关联的每个SH中的相同语法元素带来的信令开销。如表7所示,用于控制TPM融合模式的融合候选的最大数量的语法元素是在PH中指示的,而如表8和表10所示,加权预测参数仍然是在SH中指示的。表8和表9中使用的语法元素的语义描述如下。

[0455] 表7:图像头RBSP语法

[0456]	picture_header_rbsp() {	描述符
	non_reference_picture_flag	u(1)
	gdr_pic_flag	u(1)
	no_output_of_prior_pics_flag	u(1)
	if(gdr_pic_flag)	
	recovery_poc_cnt	ue(v)
	ph_pic_parameter_set_id	ue(v)
	if(sps_poc_msb_flag) {	
	ph_poc_msb_present_flag	u(1)
	if(ph_poc_msb_present_flag)	
	poc_msb_val	u(v)
	}	
	...	
	if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1)	
[0457]	pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
	...	
	rbsp_trailing_bits()	
	}	

[0458] 图像头RBSP语义

[0459] PH包括与PH关联的经编码图像(coded picture)的所有条带共用的信息。

[0460] non_reference_picture_flag等于1表示与PH关联的图像从不用作参考图像。non_reference_picture_flag等于0表示与PH关联的图像可以用作参考图像,也可以不用作参考图像。

[0461] gdr_pic_flag等于1表示与PH关联的图像是逐步解码刷新(gradual decoding refresh,GDR)图像。gdr_pic_flag等于0表示与PH关联的图像不是GDR图像。

[0462] 在解码不是码流中的第一个图像的编码层视频序列起始(coded layer video sequence start,CLVSS)图像之后,no_output_of_prior_pics_flag会影响解码图像缓冲区(decoded picture buffer,DPB)中先前经解码图像的输出。

[0463] recovery_poc_cnt表示输出顺序中经解码图像的恢复点。如果当前图像是与PH关联的GDR图像,以及编码层视频序列(coded layer video sequence,CLVS)中存在解码顺序在当前GDR图像之后的图像picA且图像picA的PicOrderCntVal等于当前GDR图像的PicOrderCntVal加上recovery_poc_cnt的值,则图像picA称为恢复点图像。否则,输出顺序中的PicOrderCntVal大于当前图像的PicOrderCntVal加上recovery_poc_cnt的值的第一个图像称为恢复点图像。恢复点图像在解码顺序中不应在当前GDR图像之前。recovery_poc_cnt的取值范围应为0至MaxPicOrderCntLsb-1(包括端值)。

[0464] 注1:当gdr_enabled_flag等于1且当前图像的PicOrderCntVal大于或等于关联GDR图像的RpPicOrderCntVal时,输出顺序中的当前经解码图像和后续经解码图像与从前一帧内随机接入点(intra random access point,IRAP)图像(存在时,解码顺序在关联GDR图像之前)开始解码过程而产生的对应图像精确匹配。

[0465] ph_pic_parameter_set_id表示当前使用的PPS的pps_pic_parameter_set_id的值。ph_pic_parameter_set_id的取值范围应为0至63(包括端值)。

[0466] 码流一致性的要求为:PH的TemporalId的值应大于或等于pps_pic_parameter_set_id等于ph_pic_parameter_set_id的图像参数集(Picture Parameter Set,PPS)的TemporalId的值。

[0467] sps_poc_msb_flag等于1表示语法元素ph_poc_msb_cycle_present_flag存在于参考序列参数集(Sequence Parameter Set,SPS)的PH中。sps_poc_msb_flag等于0表示语法元素ph_poc_msb_cycle_present_flag不存在于参考SPS的PH中。

[0468] ph_poc_msb_present_flag等于1表示语法元素poc_msb_val存在于PH中。ph_poc_msb_present_flag等于0表示语法元素poc_msb_val不存在于PH中。当vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]]等于0且当前层的参考层中的当前接入单元(Access Unit,AU)中存在图像时,ph_poc_msb_present_flag的值应等于0。

[0469] poc_msb_val表示当前图像的图像顺序编号(picture order count,POC)最高有效位(mostsignificant bit,MSB)值。语法元素poc_msb_val的长度为(poc_msb_len_minus1+1)个比特。

[0470] sps_triangle_enabled_flag表示基于三角形的运动补偿是否可以用于帧间预测。sps_triangle_enabled_flag等于0表示语法应被约束为使得基于三角形的运动补偿不用于编码层视频序列(coded layer video sequence,CLVS)中且merge_triangle_split_dir、merge_triangle_idx0和merge_triangle_idx1不存在于CLVS的编码单元(coding unit)语法中。sps_triangle_enabled_flag等于1表示基于三角形的运动补偿可以用于CLVS中。

[0471] pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1等于0表示pic_max_num_merge_cand_minus_max_num_triangle_cand存在于参考图像参数集(Picture

Parameter Set,PPS)的PH中。`pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1`大于0表示`pic_max_num_merge_cand_minus_max_num_triangle_cand`不存在于参考PPS的PH中。`pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1`的取值范围应为0至`MaxNumMergeCand-1`。

[0472] `pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1`等于0表示`pic_max_num_merge_cand_minus_max_num_triangle_cand`存在于参考PPS的条带的PH中。`pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1`大于0表示`pic_max_num_merge_cand_minus_max_num_triangle_cand`不存在于参考PPS的PH中。`pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1`的取值范围应为0至`MaxNumMergeCand-1`。

[0473] `pic_six_minus_max_num_merge_cand`表示从6中减去与PH关联的条带中支持的融合运动矢量预测(motion vector prediction,MVP)候选的最大数量。融合MVP候选的最大数量`MaxNumMergeCand`推导如下:

[0474]
$$\text{MaxNumMergeCand} = 6 - \text{picsix_minus_max_num_merge_cand}$$

[0475] `MaxNumMergeCand`的取值范围应为1至6(包括端值)。当`pic_six_minus_max_num_merge_cand`不存在时,`pic_six_minus_max_num_merge_cand`的值被推断为`pps_six_minus_max_num_merge_cand_plus1-1`。

[0476] 表8:通用条带头语法

[0477]	<code>slice_header() {</code>	描述符
	<code>slice_pic_order_cnt_lsb</code>	<code>u(v)</code>
	<code>if(subpics_present_flag)</code>	
	<code>slice_subpic_id</code>	<code>u(v)</code>
	<code>if(rect_slice_flag NumTilesInPic > 1)</code>	
	<code>slice_address</code>	<code>u(v)</code>
	<code>if(!rect_slice_flag && NumTilesInPic > 1)</code>	
	<code>num_tiles_in_slice_minus1</code>	<code>ue(v)</code>

[0478]

slice_type	ue(v)
if(!pic_rpl_present_flag &&((nal_unit_type != IDR_W_RADL && nal_unit_type !=	
IDR_N_LP) sps_idr_rpl_present_flag)) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
slice_rpl_sps_flag[i]	u(1)
if(slice_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
slice_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
slice_poc_lsb_lt[i][j]	u(v)
slice_delta_poc_msb_present_flag[i][j]	u(1)
if(slice_delta_poc_msb_present_flag[i][j])	
slice_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
if(pic_rpl_present_flag ((nal_unit_type != IDR_W_RADL && nal_unit_type !=	
IDR_N_LP) sps_idr_rpl_present_flag)) {	
if((slice_type != I && num_ref_entries[0][RplIdx[0]] > 1) (slice_type == B && num_ref_entries[1][RplIdx[1]] > 1)) {	
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag)	

[0479]	for(i = 0; i < (slice_type == B ? 2: 1); i++)	
	if(num_ref_entries[i][RplIdx[i]] > 1)	
	num_ref_idx_active_minus1[i]	ue(v)
	}	
	}	
	if(slice_type != I) {	
	...	
	if((pps_weighted_pred_flag && slice_type == P) (pps_weighted_bipred_flag && slice_type == B))	
	pred_weight_table()	
	}	
	...	
	byte_alignment()	
	}	

[0480] 通用条带头语义

[0481] 当条带头语法元素slice_pic_order_cnt_lsb存在时,该语法元素的值对于经编码图像(coded picture)的所有条带头应是相同的。

[0482] 变量CuQpDeltaVal表示包括cu_qp_delta_abs的编码单元(coding unit)的亮度量化参数与其预测参数之间的差值,该变量被设置为0。变量CuQpOffset_{Cb}、CuQpOffset_{Cr}和CuQpOffset_{CbCr}表示在确定包括cu_chroma_qp_offset_flag的编码单元的量化参数Qp'_{Cb}、Qp'_{Cr}和Qp'_{CbCr}的相应值时要使用的值,这些变量全部都被设置为0。

[0483] slice_pic_order_cnt_lsb表示当前图像的图像顺序编号对MaxPicOrderCntLsb取模。语法元素slice_pic_order_cnt_lsb的长度为(log2_max_pic_order_cnt_lsb_minus4+4)个比特。slice_pic_order_cnt_lsb的取值范围应为0至MaxPicOrderCntLsb-1(包括端值)。

[0484] 在当前图像是GDR图像时,变量RpPicOrderCntVal的推导如下:

[0485] RpPicOrderCntVal = PicOrderCntVal + recovery_poc_cnt。

[0486] slice_subpic_id表示包括条带的子图像的子图像标识符。如果slice_subpic_id存在,则变量SubPicIdx的值被推导出,使得SubPicIdxList[SubPicIdx]等于slice_subpic_id。否则(slice_subpic_id不存在),变量SubPicIdx被推导为0。slice_subpic_id的长度(以比特为单位)推导如下:

[0487] -如果sps_subpic_id_signalling_present_flag等于1,则slice_subpic_id的长度等于sps_subpic_id_len_minus1+1。

[0488] -否则,如果ph_subpic_id_signalling_present_flag等于1,则slice_subpic_id的长度等于ph_subpic_id_len_minus1+1。

[0489] -否则,如果pps_subpic_id_signalling_present_flag等于1,则slice_subpic_id

id的长度等于pps_subpic_id_len_minus1+1。

[0490] -否则,slice_subpic_id的长度等于Ceil(Log2(sps_num_subpics_minus1+1))。

[0491] slice_address表示条带的条带地址。当slice_address不存在时,slice_address的值被推断为0。

[0492] 如果rect_slice_flag等于0,则以下内容适用:

[0493] -条带地址是光栅扫描分块索引。

[0494] -slice_address的长度为Ceil(Log2(NumTilesInPic))个比特。

[0495] -slice_address的取值范围应为0至NumTilesInPic-1(包括端值)。

[0496] 否则(rect_slice_flag is等于1),以下内容适用:

[0497] -条带地址是第SubPicIdx个子图像内的条带的条带索引。

[0498] -条带地址的长度为Ceil(Log2(NumSlicesInSubpic[SubPicIdx]))个比特。

[0499] -slice_address的取值范围应为0至NumSlicesInSubpic[SubPicIdx]-1(包括端值)。

[0500] 码流一致性的要求为以下约束条件适用:

[0501] -如果rect_slice_flag等于0或subpics_present_flag等于0,则slice_address的值应不等于同一经编码图像的任何其它经编码条带(coded slice)网络抽象层(Network Abstraction Layer,NAL)单元的slice_address的值。

[0502] -否则,slice_subpic_id和slice_address的一对值应不等于同一经编码图像的任何其它经编码条带NAL单元的slice_subpic_id和slice_address的一对值。

[0503] -当rect_slice_flag等于0时,图像中的条带应按照条带的slice_address值升序排列。

[0504] -图像中的条带的形状应使得每个编码树单元(Coding Tree Unit,CTU)在解码时应使其整个左边界和整个上边界由图像边界组成或由先前一个或多个经解码CTU的边界组成。

[0505] num_tiles_in_slice_minus1+1(存在时)表示条带中的分块的数量。num_tiles_in_slice_minus1的取值范围应为0至NumTilesInPic-1(包括端值)。

[0506] 变量NumCtuInCurrSlice表示当前条带中的CTU的数量,列表CtbAddrInCurrSlice[i](i的范围为0至NumCtuInCurrSlice-1(包括端值))表示条带内的第i个编码树块(Coding Tree Block,CTB)的图像光栅扫描地址,这两者的推导如下:

```

        if( rect_slice_flag ) {
            picLevelSliceIdx = SliceSubpicToPicIdx[ SubPicIdx ][ slice_address ]
            NumCtuInCurrSlice = NumCtuInSlice[ picLevelSliceIdx ]
            for( i = 0; i < NumCtuInCurrSlice; i++ )
                CtbAddrInCurrSlice[ i ] = CtbAddrInSlice[ picLevelSliceIdx ][ i ]
        } else {
[0507]     NumCtuInCurrSlice = 0
            for( tileIdx = slice_address; tileIdx <= slice_address + num_tiles_in_slice_minus1[ i ];
            tileIdx++ ) {
                tileX = tileIdx % NumTileColumns
                tileY = tileIdx/NumTileColumns
                for( ctbY = tileRowBd[ tileY ]; ctbY < tileRowBd[ tileY + 1 ]; ctbY++ ) {
                    for( ctbX = tileColBd[ tileX ]; ctbX < tileColBd[ tileX + 1 ]; ctbX++ ) {
                        CtbAddrInCurrSlice[ NumCtuInCurrSlice ] = ctbY * PicWidthInCtb + ctbX
                        NumCtuInCurrSlice++
[0508]         }
                    }
                }
            }
        }
    }
}

```

[0509] 变量SubPicLeftBoundaryPos、SubPicTopBoundaryPos、SubPicRightBoundaryPos和SubPicBotBoundaryPos的推导如下：

```

        if( subpic_treated_as_pic_flag[ SubPicIdx ] ) {
            SubPicLeftBoundaryPos = subpic_ctu_top_left_x[ SubPicIdx ] * CtbSizeY
            SubPicRightBoundaryPos = Min( pic_width_max_in_luma_samples - 1,
            ( subpic_ctu_top_left_x[ SubPicIdx ] + subpic_width_minus1[ SubPicIdx ] + 1 ) *
[0510] CtbSizeY - 1 )
            SubPicTopBoundaryPos = subpic_ctu_top_left_y[ SubPicIdx ] * CtbSizeY
            SubPicBotBoundaryPos = Min( pic_height_max_in_luma_samples - 1,
            ( subpic_ctu_top_left_y[ SubPicIdx ] + subpic_height_minus1[ SubPicIdx ] + 1 ) * CtbSizeY -
            1 )
        }
    }
}

```

[0511] slice_type表示条带的编码类型,如表9所示。

[0512] 表9:名称与slice_type的关联关系

[0513]	slice_type	slice_type的名称
	0	B(B条带)
	1	P(P条带)
	2	I(I条带)

[0514] slice_rpl_sps_flag[i]等于1表示当前条带的参考图像列表i是根据SPS中的listIdx等于i的其中一个语法结构ref_pic_list_struct(listIdx,rplsIdx)推导出的。slice_rpl_sps_flag[i]等于0表示当前条带的参考图像列表i是根据直接包括在当前图像

的条带头中的listIdx等于i的语法结构ref_pic_list_struct(listIdx,rplsIdx)推导出的。

[0515] 当slice_rpl_sps_flag[i]不存在时,以下内容适用:

[0516] -如果pic_rpl_present_flag等于1,则slice_rpl_sps_flag[i]的值被推断为pic_rpl_sps_flag[i];

[0517] -否则,如果num_ref_pic_lists_in_sps[i]等于0,则ref_pic_list_sps_flag[i]的值被推断为0;

[0518] -否则,如果num_ref_pic_lists_in_sps[i]大于0且如果rpl1_idx_present_flag等于0,则slice_rpl_sps_flag[1]的值被推断为slice_rpl_sps_flag[0]。

[0519] slice_rpl_idx[i]表示用于推导当前图像的参考图像列表i的listIdx等于i的语法结构ref_pic_list_struct(listIdx,rplsIdx)在SPS中包括的listIdx等于i的语法结构ref_pic_list_struct(listIdx,rplsIdx)列表中的索引。语法元素slice_rpl_idx[i]由Ceil(Log2(num_ref_pic_lists_in_sps[i]))个比特表示。当slice_rpl_idx[i]不存在时,slice_rpl_idx[i]的值被推断为0。slice_rpl_idx[i]的取值范围应为0到num_ref_pic_lists_in_sps[i]-1(包括端值)。当slice_rpl_sps_flag[i]等于1且num_ref_pic_lists_in_sps[i]等于1时,slice_rpl_idx[i]的值被推断为0。当slice_rpl_sps_flag[i]等于1且rpl1_idx_present_flag等于0时,slice_rpl_idx[1]的值被推断为slice_rpl_idx[0]。

[0520] 变量RplsIdx[i]的推导如下:

if(pic_rpl_present_flag)

RplsIdx[i] = PicRplsIdx[i]

[0521] else

RplsIdx[i] = slice_rpl_sps_flag[i] ? slice_rpl_idx[i] : num_ref_pic_lists_in_sps[i]

[0522] slice_poc_lsb_lt[i][j]表示第i个参考图像列表中的第j个LTRP条目的图像顺序编号的值对MaxPicOrderCntLsb取模。语法元素slice_poc_lsb_lt[i][j]的长度为(log2_max_pic_order_cnt_lsb_minus4+4)个比特。

[0523] 变量PocLsbLt[i][j]的推导如下:

if(pic_rpl_present_flag)

PocLsbLt[i][j] = PicPocLsbLt[i][j]

[0524] else

PocLsbLt[i][j] = ltrp_in_slice_header_flag[i][RplsIdx[i]] ?

slice_poc_lsb_lt[i][j] : rpls_poc_lsb_lt[listIdx][RplsIdx[i]][j]

[0525] slice_delta_poc_msb_present_flag[i][j]等于1表示slice_delta_poc_msb_cycle_lt[i][j]存在。slice_delta_poc_msb_present_flag[i][j]等于0表示slice_delta_poc_msb_cycle_lt[i][j]不存在。

[0526] 假设prevTid0Pic为解码顺序中的先前图像,先前图像与当前图像具有相同的nuh_layer_id,先前图像的TemporalId等于0,先前图像不是跳过随机接入前置(Random Access Skipped Leading,RASL)图像或可解码随机接入前置(Random Access Decodable Leading,RADL)图像。假设setOfPrevPocVals为以下内容组成的集合:

[0527] -prevTid0Pic的PicOrderCntVal,

[0528] -由prevTid0Pic的RefPicList[0]或RefPicList[1]中的条目参考且与当前图像具有相同nuh_layer_id的每个图像的PicOrderCntVal,

[0529] -解码顺序在prevTid0Pic之后、与当前图像具有相同nuh_layer_id且解码顺序在当前图像之前的每个图像的PicOrderCntVal。

[0530] 当pic_rpl_present_flag等于0且setOfPrevPocVals中存在多个值(其值对MaxPicOrderCntLsb取模等于PocLsbLt[i][j])时,slice_delta_poc_msb_present_flag[i][j]的值应等于1。

[0531] slice_delta_poc_msb_cycle_lt[i][j]表示变量FullPocLt[i][j]的值,如下所示:

```
[0532]   if( pic_rpl_present_flag )
        FullPocLt[ i ][ j ] = PicFullPocLt[ i ][ j ]
    else {
        if( j == 0 )
            DeltaPocMsbCycleLt[ i ][ j ] = delta_poc_msb_cycle_lt[ i ][ j ]
        else
[0533]         DeltaPocMsbCycleLt[ i ][ j ] = delta_poc_msb_cycle_lt[ i ][ j ] +
            DeltaPocMsbCycleLt[ i ][ j - 1 ]
            FullPocLt[ i ][ j ] = PicOrderCntVal - DeltaPocMsbCycleLt[ i ][ j ] * MaxPicOrderCntLsb -
            ( PicOrderCntVal & ( MaxPicOrderCntLsb - 1 ) ) + PocLsbLt[ i ][ j ]
    }
```

[0534] slice_delta_poc_msb_cycle_lt[i][j]的取值范围应为0至 $2^{(32 - \log_2 \max_pic_order_cnt_lsb_minus4 - 4)}$ (包括端值)。当slice_delta_poc_msb_cycle_lt[i][j]不存在时,slice_delta_poc_msb_cycle_lt[i][j]的值被推断为0。

[0535] num_ref_idx_active_override_flag等于1表示P条带和B条带存在语法元素num_ref_idx_active_minus1[0]且B条带存在语法元素num_ref_idx_active_minus1[1]。num_ref_idx_active_override_flag等于0表示语法元素num_ref_idx_active_minus1[0]和num_ref_idx_active_minus1[1]不存在。当num_ref_idx_active_override_flag不存在时,num_ref_idx_active_override_flag的值被推断为1。

[0536] num_ref_idx_active_minus1[i]用于推导等式145所示的变量NumRefIdxActive[i]。num_ref_idx_active_minus1[i]的取值范围应为0至14(包括端值)。

[0537] 当i等于0或1时,在当前条带为B条带,num_ref_idx_active_override_flag等于1,且num_ref_idx_active_minus1[i]不存在时,num_ref_idx_active_minus1[i]被推断为0。

[0538] 在当前条带为P条带,num_ref_idx_active_override_flag等于1,且num_ref_idx_active_minus1[0]不存在时,num_ref_idx_active_minus1[0]被推断为0。

[0539] 变量NumRefIdxActive[i]的推导如下:

```

    for( i = 0; i < 2; i++ ) {
        if( slice_type == B || ( slice_type == P && i == 0 ) ) {
            if( num_ref_idx_active_override_flag )
                NumRefIdxActive[ i ] = num_ref_idx_active_minus1[ i ] + 1
            else {
                if( num_ref_entries[ i ][ RplIdx[ i ] ] >= num_ref_idx_default_active_minus1[ i ] +
[0540] 1 )
                    NumRefIdxActive[ i ] = num_ref_idx_default_active_minus1[ i ] + 1
                else
                    NumRefIdxActive[ i ] = num_ref_entries[ i ][ RplIdx[ i ] ]
            }
        } else /* slice_type == I || ( slice_type == P && i == 1 ) */
            NumRefIdxActive[ i ] = 0
    }

```

[0541] NumRefIdxActive[i]-1的值表示参考图像列表i中可以用于解码条带的最大参考索引。当NumRefIdxActive[i]的值等于0时,参考图像列表i中的参考索引不可以用于解码条带。在当前条带为P条带时,NumRefIdxActive[0]的值应大于0。

[0542] 在当前条带为B条带,NumRefIdxActive[0]和NumRefIdxActive[1]都应大于0。

[0543] 加权预测参数语法

[0544]

pred_weight_table() {	描述符
luma_log2_weight_denom	uc(v)
if(ChromaArrayType != 0)	
delta_chroma_log2_weight_denom	se(v)
for(i = 0; i < NumRefIdxActive[0]; i++)	
luma_weight_l0_flag[i]	u(1)
if(ChromaArrayType != 0)	
for(i = 0; i < NumRefIdxActive[0]; i++)	
chroma_weight_l0_flag[i]	u(1)
for(i = 0; i < NumRefIdxActive[0]; i++) {	
if(luma_weight_l0_flag[i]) {	
delta_luma_weight_l0[i]	se(v)
luma_offset_l0[i]	se(v)
}	
if(chroma_weight_l0_flag[i])	
for(j = 0; j < 2; j++) {	
delta_chroma_weight_l0[i][j]	se(v)
delta_chroma_offset_l0[i][j]	se(v)
}	
}	
if(slice_type == B) {	
for(i = 0; i < NumRefIdxActive[1]; i++)	
luma_weight_l1_flag[i]	u(1)
if(ChromaArrayType != 0)	
for(i = 0; i < NumRefIdxActive[1]; i++)	
chroma_weight_l1_flag[i]	u(1)

[0545]	for(i = 0; i < NumRefIdxActive[1]; i++) {	
	if(luma_weight_l1_flag[i]) {	
	delta_luma_weight_l1[i]	se(v)
	luma_offset_l1[i]	se(v)
	}	
	if(chroma_weight_l1_flag[i])	
	for(j = 0; j < 2; j++) {	
	delta_chroma_weight_l1[i][j]	se(v)
	delta_chroma_offset_l1[i][j]	se(v)
	}	
	}	
	}	
	}	

[0546] 加权预测参数语义

[0547] luma_log2_weight_denom是所有亮度加权因子的分母以2为底的对数。luma_log2_weight_denom的取值范围应为0至7(包括端值)。

[0548] delta_chroma_log2_weight_denom是所有色度加权因子的分母以2为底的对数的差值。当delta_chroma_log2_weight_denom不存在时,delta_chroma_log2_weight_denom被推断为0。

[0549] 变量ChromaLog2WeightDenom被推导为luma_log2_weight_denom+delta_chroma_log2_weight_denom,该变量的取值范围应为0至7(包括端值)。

[0550] luma_weight_l0_flag[i]等于1表示使用RefPicList[0][i]对列表0进行预测所使用的亮度分量的加权因子存在。luma_weight_l0_flag[i]等于0表示这些加权因子不存在。

[0551] chroma_weight_l0_flag[i]等于1表示使用RefPicList[0][i]对列表0进行预测所使用的色度预测值的加权因子存在。chroma_weight_l0_flag[i]等于0表示这些加权因子不存在。当chroma_weight_l0_flag[i]不存在时,chroma_weight_l0_flag[i]被推断为0。

[0552] delta_luma_weight_l0[i]是应用于使用RefPicList[0][i]对列表0进行预测所使用的亮度预测值的加权因子的差值。

[0553] 变量LumaWeightL0[i]被推导为 $(1 \ll \text{luma_log2_weight_denom}) + \text{delta_luma_weight_l0}[i]$ 。当luma_weight_l0_flag[i]等于1时,delta_luma_weight_l0[i]的取值范围应为-128至127(包括端值)。当luma_weight_l0_flag[i]等于0时,LumaWeightL0[i]被推断为 $2 \times \text{luma_log2_weight_denom}$ 。

[0554] luma_offset_l0[i]是应用于使用RefPicList[0][i]对列表0进行预测所使用的亮度预测值的附加偏移。luma_offset_l0[i]的取值范围应为-128至127(包括端值)。当

luma_weight_10_flag[i]等于0时,luma_offset_10[i]被推断为0。

[0555] delta_chroma_weight_10[i][j]是应用于使用RefPicList[0][i]对列表0进行预测所使用的色度预测值的加权因子的差值,其中,对于Cb,j等于0,对于Cr,j等于1。

[0556] 变量ChromaWeightL0[i][j]被推导为 $(1 \ll \text{ChromaLog2WeightDenom}) + \text{delta_chroma_weight_10}[i][j]$ 。当chroma_weight_10_flag[i]等于1时,delta_chroma_weight_10[i][j]的取值范围应为-128至127(包括端值)。当chroma_weight_10_flag[i]等于0时,ChromaWeightL0[i][j]被推断为2ChromaLog2WeightDenom。

[0557] delta_chroma_offset_10[i][j]是应用于使用RefPicList[0][i]对列表0进行预测所使用的色度预测值的附加偏移的差值,其中,对于Cb,j等于0,对于Cr,j等于1。

[0558] 变量ChromaOffsetL0[i][j]的推导如下:

[0559] $\text{ChromaOffsetL0}[i][j] = \text{Clip3}(-128, 127, (128 + \text{delta_chroma_offset_10}[i][j] - ((128 * \text{ChromaWeightL0}[i][j]) \gg \text{ChromaLog2WeightDenom})))$

[0560] ChromaWeightL0[i][j]>>ChromaLog2WeightDenom)))

[0561] delta_chroma_offset_10[i][j]的取值范围应为-4*128至4*127(包括端值)。当chroma_weight_10_flag[i]等于0时,ChromaOffsetL0[i][j]被推断为0。

[0562] luma_weight_11_flag[i]、chroma_weight_11_flag[i]、delta_luma_weight_11[i]、luma_offset_11[i]、delta_chroma_weight_11[i][j]和delta_chroma_offset_11[i][j]的语义分别与luma_weight_10_flag[i]、chroma_weight_10_flag[i]、delta_luma_weight_10[i]、luma_offset_10[i]、delta_chroma_weight_10[i][j]和delta_chroma_offset_10[i][j]的语义相同,其中,10、L0、列表0和List0分别由11、L1、列表1和List1代替。

[0563] 变量sumWeightL0Flags被推导为luma_weight_10_flag[i]+2*chroma_weight_10_flag[i]之和,其中, $i=0.. \text{NumRefIdxActive}[0]-1$ 。

[0564] 当slice_type为B时,变量sumWeightL1Flags被推导为luma_weight_11_flag[i]+2*chroma_weight_11_flag[i]之和,其中, $i=0.. \text{NumRefIdxActive}[1]-1$ 。

[0565] 码流一致性的要求为:当slice_type为P时,sumWeightL0Flags应小于或等于24;当slice_type为B时,sumWeightL0Flags与sumWeightL1Flags之和应小于或等于24。

[0566] 参考图像列表结构语义

[0567] 语法结构ref_pic_list_struct(listIdx,rplsIdx)可以存在于SPS中,也可以存在于条带头中。根据语法结构是包括在条带头中还是包括在SPS中,以下内容适用:

[0568] -如果语法结构ref_pic_list_struct(listIdx,rplsIdx)存在于条带头中,则该语法结构表示当前图像(包括条带的图像)的参考图像列表listIdx。

[0569] -否则(存在于SPS中),语法结构ref_pic_list_struct(listIdx,rplsIdx)表示参考图像列表listIdx的候选。本节其余内容规定的语义中的术语“当前图像”是指:(1)包括一个或多个条带的每个图像,其中,一个或多个条带包括的ref_pic_list_idx[listIdx]等于SPS中包括的语法结构ref_pic_list_struct(listIdx,rplsIdx)列表中的索引,(2)参考SPS的编码视频序列(Coded Video Sequence,CVS)中的每个图像。

[0570] num_ref_entries[listIdx][rplsIdx]表示语法结构ref_pic_list_struct(listIdx,rplsIdx)中的条目的数量.num_ref_entries[listIdx][rplsIdx]的取值范围应

为0至MaxDecPicBuffMinus1+14(包括端值)。

[0571] ltrp_in_slice_header_flag[listIdx][rplsIdx]等于0表示语法结构ref_pic_list_struct(listIdx,rplsIdx)中的LTRP条目的POC LSB存在于语法结构ref_pic_list_struct(listIdx,rplsIdx)中。ltrp_in_slice_header_flag[listIdx][rplsIdx]等于1表示语法结构ref_pic_list_struct(listIdx,rplsIdx)中的长期参考图像(Long-Term Reference Picture,LTRP)条目的POC LSB不存在于语法结构ref_pic_list_struct(listIdx,rplsIdx)中。

[0572] inter_layer_ref_pic_flag[listIdx][rplsIdx][i]等于1表示语法结构ref_pic_list_struct(listIdx,rplsIdx)中的第i个条目是层间参考图像(Inter-Layer Reference Picture,ILRP)条目。inter_layer_ref_pic_flag[listIdx][rplsIdx][i]等于0表示语法结构ref_pic_list_struct(listIdx,rplsIdx)中的第i个条目不是ILRP条目。当inter_layer_ref_pic_flag[listIdx][rplsIdx][i]不存在时,inter_layer_ref_pic_flag[listIdx][rplsIdx][i]的值被推断为0。

[0573] st_ref_pic_flag[listIdx][rplsIdx][i]等于1表示语法结构ref_pic_list_struct(listIdx,rplsIdx)中的第i个条目是STRP条目。st_ref_pic_flag[listIdx][rplsIdx][i]等于0表示语法结构ref_pic_list_struct(listIdx,rplsIdx)中的第i个条目是LTRP条目。当inter_layer_ref_pic_flag[listIdx][rplsIdx][i]等于0且st_ref_pic_flag[listIdx][rplsIdx][i]不存在时,st_ref_pic_flag[listIdx][rplsIdx][i]的值被推断为1。

[0574] 变量NumLtrpEntries[listIdx][rplsIdx]的推导如下:

```
for( i = 0, NumLtrpEntries[ listIdx ][ rplsIdx ] = 0; i < num_ref_entries[ listIdx ][ rplsIdx ]; i++ )
```

```
    if( !inter_layer_ref_pic_flag[ listIdx ][ rplsIdx ][ i ]
        && !st_ref_pic_flag[ listIdx ][ rplsIdx ][ i ] )
        NumLtrpEntries[ listIdx ][ rplsIdx ]++
```

[0576] abs_delta_poc_st[listIdx][rplsIdx][i]表示变量AbsDeltaPocSt[listIdx][rplsIdx][i]的值,如下所示:

```
if( sps_weighted_pred_flag || sps_weighted_bipred_flag )
    AbsDeltaPocSt[ listIdx ][ rplsIdx ][ i ] = abs_delta_poc_st[ listIdx ][ rplsIdx ][ i ]
```

[0577] else
AbsDeltaPocSt[listIdx][rplsIdx][i] = abs_delta_poc_st[listIdx][rplsIdx][i] + 1
abs_delta_poc_st[listIdx][rplsIdx][i]的取值范围应为 0 至 $2^{15} - 1$ (包括端值)。

[0578] strp_entry_sign_flag[listIdx][rplsIdx][i]等于1表示语法结构ref_pic_list_struct(listIdx,rplsIdx)中的第i个条目的值大于或等于0。strp_entry_sign_flag[listIdx][rplsIdx][i]等于0表示语法结构ref_pic_list_struct(listIdx,rplsIdx)中的第i个条目的值小于0。当strp_entry_sign_flag[listIdx][rplsIdx][i]不存在时, strp_entry_sign_flag[listIdx][rplsIdx][i]的值被推断为1。

[0579] 列表DeltaPocValSt[listIdx][rplsIdx]的推导如下:

```

        for( i = 0; i < num_ref_entries[ listIdx ][ rplsIdx ]; i++ )
            if( !inter_layer_ref_pic_flag[ listIdx ][ rplsIdx ][ i ] &&
[0580]    st_ref_pic_flag[ listIdx ][ rplsIdx ][ i ] )
                DeltaPocValSt[ listIdx ][ rplsIdx ][ i ] = ( strp_entry_sign_flag[ listIdx ][ rplsIdx ][ i ] ) ?
                AbsDeltaPocSt[ listIdx ][ rplsIdx ][ i ] : 0 - AbsDeltaPocSt[ listIdx ][ rplsIdx ][ i ]
[0581]    rpls_poc_lsb_lt[ listIdx ][ rplsIdx ][ i ]表示由语法结构ref_pic_list_struct
    (listIdx, rplsIdx)中的第i个条目参考的图像的图像顺序编号的值对MaxPicOrderCntLsb
    取模。语法元素rpls_poc_lsb_lt[listIdx][rplsIdx][i]的长度为(log2_max_pic_order_
    cnt_lsb_minus4+4)个比特。
[0582]    ilrp_idx[listIdx][rplsIdx][i]表示语法结构ref_pic_list_struct(listIdx,
    rplsIdx)中的第i个条目的ILRP在直接参考层列表中的索引。ilrp_idx[listIdx]
    [rplsIdx][i]的取值范围应为0至NumDirectRefLayers[GeneralLayerIdx[nuh_layer_
    id]]-1(包括端值)。
[0583]    因此,不同的机制可以用于控制GEO/TPM融合模式,取决于WP是否应用于通过以下
    方式从中获取参考块P0和P1的参考图像:
[0584]    -将表14中列出的WP参数从SH移动到PH;
[0585]    -将GEO/TPM参数从PH移回SH;
[0586]    -改变MaxNumTriangleMergeCand的语义,即,当可以使用经过WP的参考图像(例
    如,至少一个标志lumaWeightedFlag为真)时,将这些条带的MaxNumTriangleMergeCand设
    置为0或1。
[0587]    对于TPM融合模式,示例性参考块P0和P1在图7中分别表示为710和720。对于GEO融
    合模式,示例性参考块P0和P1在图8中分别表示为810和820。
[0588]    因此,不同的机制可以用于控制GEO/TPM融合模式,取决于WP是否应用于通过以下
    方式从中获取参考块P0和P1的参考图像:
[0589]    -将表14中列出的WP参数从SH移动到PH;
[0590]    -将GEO/TPM参数从PH移回SH;
[0591]    -改变MaxNumTriangleMergeCand的语义,即,当可以使用经过WP的参考图像(例
    如,至少一个标志lumaWeightedFlag为真)时,将这些条带的MaxNumTriangleMergeCand设
    置为0或1。
[0592]    对于TPM融合模式,示例性参考块P0和P1在图7中分别表示为710和720。对于GEO融
    合模式,示例性参考块P0和P1在图8中分别表示为810和820。
[0593]    在一个实施例中,当非矩形模式(例如,GEO和TPM)的启用和WP参数在图像头中指
    示时,可以使用以下语法,如下表所示:
[0594]    表:图像头RBSP语法

```

[0595]

picture_header_rbsp() {	描述符
non_reference_picture_flag	u(1)
gdr_pic_flag	u(1)
no_output_of_prior_pics_flag	u(1)
if(gdr_pic_flag)	
recovery_poc_cnt	ue(v)
ph_pic_parameter_set_id	ue(v)
if(sps_poc_msb_flag) {	
ph_poc_msb_present_flag	u(1)

[0596]

if(ph_poc_msb_present_flag)	
poc_msb_val	u(v)
}	
...	
pic_rpl_present_flag	u(1)
if(pic_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
...	
if((pps_weighted_pred_flag && slice_type == P) (pps_weighted_bipred_flag && slice_type == B))	
pred_weight_table()	

[0597]	...	
	...	
	if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1 && WPDisabled)	
	pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
	...	
	rbsp_trailing_bits()	
	}	

[0598]

[0599] 当luma_weight_10_flag[i]、chroma_weight_10_flag[i]、luma_weight_11_flag[j]和chroma_weight_11_flag[j]的所有值都被设置为0时,变量WPDisabled被设置为1,其中,i=0..NumRefIdxActive[0],j=0..NumRefIdxActive[1];否则,WPDisabled的值被设置为0。

[0600] 当变量WPDisabled被设置为0时,pic_max_num_merge_cand_minus_max_num_triangle_cand的值被设置为MaxNumMergeCand。

[0601] 在一个示例中,WP参数的指示和非矩形模式(例如,GE0和TPM)的启用在条带头中执行。下表给出了示例性语法。

[0602]	slice_header() {	描述符
	slice_pic_parameter_set_id	ue(v)
	if(rect_slice_flag NumBricksInPic > 1)	
	slice_address	u(v)
	if(!rect_slice_flag && !single_brick_per_slice_flag)	
	num_bricks_in_slice_minus1	ue(v)
	non_reference_picture_flag	u(1)
	slice_type	ue(v)
	if(separate_colour_plane_flag == 1)	
	colour_plane_id	u(2)
	slice_pic_order_cnt_lsb	u(v)
	if(nal_unit_type == GDR_NUT)	
	recovery_poc_cnt	ue(v)

[0603]

if(nal_unit_type == IDR_W_RADL nal_unit_type == IDR_N_LP nal_unit_type == CRA_NUT NalUnitType == GDR_NUT)	
no_output_of_prior_pics_flag	u(1)
if(output_flag_present_flag)	
pic_output_flag	u(1)
if((nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) sps_idr_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
ref_pic_list_sps_flag[i]	u(1)
if(ref_pic_list_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
ref_pic_list_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
slice_poc_lsb_lt[i][j]	u(v)
delta_poc_msb_present_flag[i][j]	u(1)
if(delta_poc_msb_present_flag[i][j])	
delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
if((slice_type != I && num_ref_entries[0][RplIdx[0]] > 1) (slice_type == B && num_ref_entries[1][RplIdx[1]] > 1)) {	
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag)	

[0604]

for(i = 0; i < (slice_type == B ? 2: 1); i++)	
if(num_ref_entries[i][RplIdx[i]] > 1)	
num_ref_idx_active_minus1[i]	ue(v)
}	
}	
if(partition_constraints_override_enabled_flag) {	
partition_constraints_override_flag	ue(v)
if(partition_constraints_override_flag) {	
slice_log2_diff_min_qt_min_cb_luma	ue(v)
slice_max_mtt_hierarchy_depth_luma	ue(v)
if(slice_max_mtt_hierarchy_depth_luma != 0)	
slice_log2_diff_max_bt_min_qt_luma	ue(v)
slice_log2_diff_max_tt_min_qt_luma	ue(v)
}	
if(slice_type == I && qtbt_dual_tree_intra_flag) {	
slice_log2_diff_min_qt_min_cb_chroma	ue(v)
slice_max_mtt_hierarchy_depth_chroma	ue(v)
if(slice_max_mtt_hierarchy_depth_chroma != 0)	
slice_log2_diff_max_bt_min_qt_chroma	ue(v)
slice_log2_diff_max_tt_min_qt_chroma	ue(v)
}	
}	
}	
}	
if (slice_type != I) {	
if(sps_temporal_mvp_enabled_flag && !pps_temporal_mvp_enabled_idc)	
slice_temporal_mvp_enabled_flag	u(1)
if(slice_type == B && !pps_mv_d_11_zero_idc)	

[0605]

mvd_l1_zero_flag	u(1)
if(cabac_init_present_flag)	
cabac_init_flag	u(1)
if(slice_temporal_mvp_enabled_flag) {	
if(slice_type == B && !pps_collocated_from_l0_idc)	
collocated_from_l0_flag	u(1)
if((collocated_from_l0_flag && NumRefIdxActive[0] > 1) (!collocated_from_l0_flag && NumRefIdxActive[1] > 1))	
collocated_ref_idx	ue(v)
}	
if((pps_weighted_pred_flag && slice_type == P) (pps_weighted_bipred_flag && slice_type == B))	
slice_weighted_pred_flag	u(1)
if(slice_weighted_pred_flag)	
pred_weight_table()	
if(!pps_six_minus_max_num_merge_cand_plus1)	
six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag && !pps_five_minus_max_num_subblock_merge_cand_plus1)	
five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
slice_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_dmvr_slice_present_flag)	
slice_disable_bdof_dmvr_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_minus1 && WPDIsabled) {	
max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
}	

[0606]

}	
if(sps_ibc_enabled_flag)	
slice_six_minus_max_num_ibc_merge_cand	ue(v)
if(sps_joint_cbr_enabled_flag)	
slice_joint_cbr_sign_flag	u(1)
slice_qp_delta	se(v)
if(pps_slice_chroma_qp_offsets_present_flag) {	
slice_cb_qp_offset	se(v)
slice_cr_qp_offset	se(v)
if(sps_joint_cbr_enabled_flag)	
slice_joint_cbr_qp_offset	se(v)
}	
if(sps_sao_enabled_flag) {	
slice_sao_luma_flag	u(1)
if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
if(sps_alf_enabled_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
}	

[0607]

}	
if(!pps_dep_quant_enabled_flag)	
dep_quant_enabled_flag	u(1)
if(!dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag)	
deblocking_filter_override_flag	u(1)
if(deblocking_filter_override_flag) {	
slice_deblocking_filter_disabled_flag	u(1)
if(!slice_deblocking_filter_disabled_flag) {	
slice_beta_offset_div2	se(v)
slice_tc_offset_div2	se(v)
}	
}	
if(sps_lmcs_enabled_flag) {	
slice_lmcs_enabled_flag	u(1)
if(slice_lmcs_enabled_flag) {	
slice_lmcs_aps_id	u(2)
if(ChromaArrayType != 0)	
slice_chroma_residual_scale_flag	u(1)
}	
}	
if(sps_scaling_list_enabled_flag) {	
slice_scaling_list_present_flag	u(1)
if(slice_scaling_list_present_flag)	
slice_scaling_list_aps_id	u(3)
}	
if(entry_point_offsets_present_flag && NumEntryPoints > 0) {	

	offset_len_minus1	ue(v)
	for(i = 0; i < NumEntryPoints; i++)	
	entry_point_offset_minus1[i]	u(v)
	}	
	if(slice_header_extension_present_flag) {	
[0608]	slice_header_extension_length	ue(v)
	for(i = 0; i < slice_header_extension_length; i++)	
	slice_header_extension_data_byte[i]	u(8)
	}	
	byte_alignment()	
	}	

[0609]

[0610] 当luma_weight_l0_flag[i]、chroma_weight_l0_flag[i]、luma_weight_l1_flag[j]和chroma_weight_l1_flag[j]的所有值都被设置为0时,变量WPDisabled被设置为1,其中,i=0..NumRefIdxActive[0],j=0..NumRefIdxActive[1];否则,WPDisabled的值被设置为0。

[0611] 当变量WPDisabled被设置为0时,max_num_merge_cand_minus_max_num_triangle_cand的值被设置为MaxNumMergeCand。

[0612] 在上述公开的实施例中,加权预测参数可以在图像头或条带头中指示。

[0613] 在一个示例中,在考虑块可以用来进行非矩形加权预测的参考图像列表时,确定是否启用TPM或GE0。当块的融合列表只包括一个参考图像列表k中的元素时,变量WPDisabled[k]的值确定该融合模式是否被启用。

[0614] 在一个示例中,非矩形帧间预测模式的融合列表被构造为使得该融合列表只包括未启用加权预测的元素。

[0615] 说明书的以下部分举例说明了该示例:

[0616] 该过程的输入包括:

[0617] -当前亮度编码块(coding block)的左上样本相对于当前图像的左上亮度样本的亮度位置(xCb,yCb);

[0618] -变量cbWidth,表示当前编码块的以亮度样本为单位的宽度;

[0619] -变量cbHeight,表示当前编码块的以亮度样本为单位的高度。

[0620] 该过程的输出如下,其中,X为0或1:

[0621] -相邻编码单元(coding unit)的可用性标志availableFlagA₀、availableFlagA₁、availableFlagB₀、availableFlagB₁和availableFlagB₂;

[0622] -相邻编码单元的参考索引refIdxLXA₀、refIdxLXA₁、refIdxLXB₀、refIdxLXB₁和refIdxLXB₂;

[0623] -相邻编码单元的预测列表使用标志predFlagLXA₀、predFlagLXA₁、predFlagLXB₀、

predFlagLXB₁和predFlagLXB₂;

[0624] -相邻编码单元的1/16分数样本精度的运动矢量mvLXA₀、mvLXA₁、mvLXB₀、mvLXB₁和mvLXB₂;

[0625] -半样本插值滤波器索引hpelIfIdxA₀、hpelIfIdxA₁、hpelIfIdxB₀、hpelIfIdxB₁和hpelIfIdxB₂;

[0626] -双向预测权重索引bcwIdxA₀、bcwIdxA₁、bcwIdxB₀、bcwIdxB₁和bcwIdxB₂。

[0627] 为了推导availableFlagB₁、refIdxLXB₁、predFlagLXB₁、mvLXB₁、hpelIfIdxB₁和bcwIdxB₁,以下内容适用:

[0628] -相邻亮度编码块内的亮度位置(xNbB₁,yNbB₁)被设置为(xCb+cbWidth-1,yCb-1)。

[0629] -第6.4.4节详述的相邻块可用性推导过程被调用,其中,输入包括设置为(xCb,yCb)的当前亮度位置(xCurr,yCurr)、相邻亮度位置(xNbB₁,yNbB₁)、设置为真的checkPredModeY和设置为0的cIdx,输出被赋值给块可用性标志availableB₁。

[0630] -变量availableFlagB₁、refIdxLXB₁、predFlagLXB₁、mvLXB₁、hpelIfIdxB₁和bcwIdxB₁的推导如下:

[0631] -如果availableB₁为假,则availableFlagB₁被设置为0,mvLXB₁的两个分量都被设置为0,refIdxLXB₁被设置为-1且predFlagLXB₁被设置为0(其中,X为0或1),hpelIfIdxB₁被设置为0,bcwIdxB₁被设置为0。

[0632] -否则,availableFlagB₁被设置为1,并进行以下赋值:

[0633] $mvLXB_1 = MvLX[xNbB_1][yNbB_1]$ (501)

[0634] $refIdxLXB_1 = RefIdxLX[xNbB_1][yNbB_1]$ (502)

[0635] $predFlagLXB_1 = PredFlagLX[xNbB_1][yNbB_1]$ (503)

[0636] $hpelIfIdxB_1 = HpelIfIdx[xNbB_1][yNbB_1]$ (504)

[0637] $bcwIdxB_1 = BcwIdx[xNbB_1][yNbB_1]$ (505)

[0638] 为了推导availableFlagA₁、refIdxLXA₁、predFlagLXA₁、mvLXA₁、hpelIfIdxA₁和bcwIdxA₁,以下内容适用:

[0639] -相邻亮度编码块内的亮度位置(xNbA₁,yNbA₁)被设置为(xCb-1,yCb+cbHeight-1)。

[0640] -第6.4.4节详述的相邻块可用性推导过程被调用,其中,输入包括设置为(xCb,yCb)的当前亮度位置(xCurr,yCurr)、相邻亮度位置(xNbA₁,yNbA₁)、设置为真的checkPredModeY和设置为0的cIdx,输出被赋值给块可用性标志availableA₁。

[0641] -变量availableFlagA₁、refIdxLXA₁、predFlagLXA₁、mvLXA₁、hpelIfIdxA₁和bcwIdxA₁的推导如下:

[0642] -如果以下一个或多个条件为真,则availableFlagA₁被设置为0,mvLXA₁的两个分量都被设置为0,refIdxLXA₁被设置为-1且predFlagLXA₁被设置为0(其中,X为0或1),hpelIfIdxA₁被设置为0,bcwIdxA₁被设置为0:

[0643] -availableA₁为假,

[0644] -availableB₁为真,亮度位置(xNbA₁,yNbA₁)和(xNbB₁,yNbB₁)具有相同的运动矢量和相同的参考索引,

[0645] -WpDisabledX[RefIdxLX[xNbA₁][yNbA₁]]被设置为0,融合模式为非矩形融合模式

(例如,块的三角标志在当前亮度位置(xCurr,yCurr)上被设置为1),

[0646] -WPDisabledX[RefIdxLX[xNbB₁][yNbB₁]]被设置为0,融合模式为非矩形融合模式(例如,块的三角标志在当前亮度位置(xCurr,yCurr)上被设置为1);

[0647] -否则,availableFlagA₁被设置为1,并进行以下赋值:

[0648] $mvLXA_1 = MvLX[xNbA_1][yNbA_1]$ (506)

[0649] $refIdxLXA_1 = RefIdxLX[xNbA_1][yNbA_1]$ (507)

[0650] $predFlagLXA_1 = PredFlagLX[xNbA_1][yNbA_1]$ (508)

[0651] $hpelIfIdxA_1 = HpelIfIdx[xNbA_1][yNbA_1]$ (509)

[0652] $bcwIdxA_1 = BcwIdx[xNbA_1][yNbA_1]$ (510)

[0653] 为了推导availableFlagB₀、refIdxLXB₀、predFlagLXB₀、mvLXB₀、hpelIfIdxB₀和bcwIdxB₀,以下情况适用:

[0654] -相邻亮度编码块内的亮度位置(xNbB₀,yNbB₀)被设置为(xCb+cbWidth,yCb-1)。

[0655] -第6.4.4节详述的相邻块可用性推导过程被调用,其中,输入包括设置为(xCb,yCb)的当前亮度位置(xCurr,yCurr)、相邻亮度位置(xNbB₀,yNbB₀)、设置为真的checkPredModeY和设置为0的cIdx,输出被赋值给块可用性标志availableB₀。

[0656] -变量availableFlagB₀、refIdxLXB₀、predFlagLXB₀、mvLXB₀、hpelIfIdxB₀和bcwIdxB₀的推导如下:

[0657] -如果以下一个或多个条件为真,则availableFlagB₀被设置为0,mvLXB₀的两个分量都被设置为0,refIdxLXB₀被设置为-1且predFlagLXB₀被设置为0(其中,X为0或1),hpelIfIdxB₀被设置为0,bcwIdxB₀被设置为0:

[0658] -availableB₀为假,

[0659] -availableB₁为真,亮度位置(xNbB₁,yNbB₁)和(xNbB₀,yNbB₀)具有相同的运动矢量和相同的参考索引,

[0660] -WPDisabledX[RefIdxLX[xNbB₀][yNbB₀]]被设置为0,融合模式为非矩形融合模式(例如,块的三角标志在当前亮度位置(xCurr,yCurr)上被设置为1),

[0661] -WPDisabledX[RefIdxLX[xNbB₁][yNbB₁]]被设置为0,融合模式为非矩形融合模式(例如,块的三角标志在当前亮度位置(xCurr,yCurr)上被设置为1);

[0662] -否则,availableFlagB₀被设置为1,并进行以下赋值:

[0663] $mvLXB_0 = MvLX[xNbB_0][yNbB_0]$ (511)

[0664] $refIdxLXB_0 = RefIdxLX[xNbB_0][yNbB_0]$ (512)

[0665] $predFlagLXB_0 = PredFlagLX[xNbB_0][yNbB_0]$ (513)

[0666] $hpelIfIdxB_0 = HpelIfIdx[xNbB_0][yNbB_0]$ (514)

[0667] $bcwIdxB_0 = BcwIdx[xNbB_0][yNbB_0]$ (515)

[0668] 为了推导availableFlagA₀、refIdxLXA₀、predFlagLXA₀、mvLXA₀、hpelIfIdxA₀和bcwIdxA₀,以下内容适用:

[0669] -相邻亮度编码块内的亮度位置(xNbA₀,yNbA₀)被设置为(xCb-1,yCb+cbWidth)。

[0670] -第6.4.4节详述的相邻块可用性推导过程被调用,其中,输入包括设置为(xCb,yCb)的当前亮度位置(xCurr,yCurr)、相邻亮度位置(xNbA₀,yNbA₀)、设置为真的checkPredModeY和设置为0的cIdx,输出被赋值给块可用性标志availableA₀。

[0671] -变量availableFlagA₀、refIdxLXA₀、predFlagLXA₀、mvLXA₀、hpelIfIdxA₀和bcwIdxA₀的推导如下:

[0672] -如果以下一个或多个条件为真,则availableFlagA₀被设置为0,mvLXA₀的两个分量都被设置为0,refIdxLXA₀被设置为-1且predFlagLXA₀被设置为0(其中,

[0673] X为0或1),hpelIfIdxA₀被设置为0,bcwIdxA₀被设置为0:

[0674] -availableA₀为假,

[0675] -availableA₁为真,亮度位置(xNbA₁,yNbA₁)和(xNbA₀,yNbA₀)具有相同的运动矢量和相同的参考索引,

[0676] -WpDisabledX[RefIdxLX[xNbA₀][yNbA₀]]被设置为0,融合模式为非矩形融合模式(例如,块的三角标志在当前亮度位置(xCurr,yCurr)上被设置为1),

[0677] -WpDisabledX[RefIdxLX[xNbA₁][yNbA₁]]被设置为0,融合模式为非矩形融合模式(例如,块的三角标志在当前亮度位置(xCurr,yCurr)上被设置为1);

[0678] -否则,availableFlagA₀被设置为1,并进行以下赋值:

[0679] $mvLXA_0 = MvLX[xNbA_0][yNbA_0]$ (516)

[0680] $refIdxLXA_0 = RefIdxLX[xNbA_0][yNbA_0]$ (517)

[0681] $predFlagLXA_0 = PredFlagLX[xNbA_0][yNbA_0]$ (518)

[0682] $hpelIfIdxA_0 = HpelIfIdx[xNbA_0][yNbA_0]$ (519)

[0683] $bcwIdxA_0 = BcwIdx[xNbA_0][yNbA_0]$ (520)

[0684] 为了推导availableFlagB₂、refIdxLXB₂、predFlagLXB₂、mvLXB₂、hpelIfIdxB₂和bcwIdxB₂,以下情况适用:

[0685] -相邻亮度编码块内的亮度位置(xNbB₂,yNbB₂)被设置为(xCb-1,yCb-1)。

[0686] -第6.4.4节详述的相邻块可用性推导过程被调用,其中,输入包括设置为(xCb,yCb)的当前亮度位置(xCurr,yCurr)、相邻亮度位置(xNbB₂,yNbB₂)、设置为真的checkPredModeY和设置为0的cIdx,输出被赋值给块可用性标志availableB₂。

[0687] -变量availableFlagB₂、refIdxLXB₂、predFlagLXB₂、mvLXB₂、hpelIfIdxB₂和bcwIdxB₂的推导如下:

[0688] -如果以下一个或多个条件为真,则availableFlagB₂被设置为0,mvLXB₂的两个分量都被设置为0,refIdxLXB₂被设置为-1且predFlagLXB₂被设置为0(其中,X为0或1),hpelIfIdxB₂被设置为0,bcwIdxB₂被设置为0:

[0689] -availableB₂为假,

[0690] -availableA₁为真,亮度位置(xNbA₁,yNbA₁)和(xNbB₂,yNbB₂)具有相同的运动矢量和相同的参考索引,

[0691] -availableB₁为真,亮度位置(xNbB₁,yNbB₁)和(xNbB₂,yNbB₂)具有相同的运动矢量和相同的参考索引,

[0692] -availableFlagA₀+availableFlagA₁+availableFlagB₀+availableFlagB₁等于4。

[0693] -WpDisabledX[RefIdxLX[xNbB₁][yNbB₁]]被设置为0,融合模式为非矩形融合模式(例如,块的三角标志在当前亮度位置(xCurr,yCurr)上被设置为1),

[0694] -WpDisabledX[RefIdxLX[xNbB₂][yNbB₂]]被设置为0,融合模式为非矩形融合模式(例如,块的三角标志在当前亮度位置(xCurr,yCurr)上被设置为1);

[0695] -否则,availableFlagB₂被设置为1,并进行以下赋值:

[0696] $mvLXB_2 = MvLX[xNbB_2][yNbB_2]$ (521)

[0697] $refIdxLXB_2 = RefIdxLX[xNbB_2][yNbB_2]$ (522)

[0698] $predFlagLXB_2 = PredFlagLX[xNbB_2][yNbB_2]$ (523)

[0699] $hpelIfIdxB_2 = HpelIfIdx[xNbB_2][yNbB_2]$ (524)

[0700] $bcwIdxB_2 = BcwIdx[xNbB_2][yNbB_2]$ (525)

[0701] 在上面公开的示例中,使用以下变量定义:

[0702] 当luma_weight_l0_flag[i]和chroma_weight_l0_flag[i]的所有值都被设置为0时,变量WpDisabled0[i]被设置为1,其中,i=0..NumRefIdxActive[0]。否则,WpDisabled0[i]的值被设置为0。

[0703] 当luma_weight_l1_flag[i]和chroma_weight_l1_flag[i]的所有值都被设置为0时,变量WpDisabled1[i]被设置为1,其中,i=0..NumRefIdxActive[1]。否则,WpDisabled1[1]的值被设置为0。

[0704] 在另一个示例中,根据以下条件之一,在条带头处定义变量SliceMaxNumTriangleMergeCand:

[0705] -SliceMaxNumTriangleMergeCand=(lumaWeightedFlag||chromaWeightedFlag)?0:MaxNumTriangleMergeCand;

[0706] -SliceMaxNumTriangleMergeCand=(lumaWeightedFlag||chromaWeightedFlag)?1:MaxNumTriangleMergeCand;

[0707] -SliceMaxNumTriangleMergeCand=slice_weighted_pred_flag?0:MaxNumTriangleMergeCand;

[0708] 或者

[0709] -SliceMaxNumTriangleMergeCand=slice_weighted_pred_flag?1:MaxNumTriangleMergeCand。

[0710] SliceMaxNumTriangleMergeCand的值进一步用于在块级解析融合信息。下表给出了示例性语法。

[0711]	merge_data(x0, y0, cbWidth, cbHeight, chType) {	描述符
	if (CuPredMode[chType][x0][y0] == MODE_IBC) {	
	if(MaxNumIbcMergeCand > 1)	
	merge_idx [x0][y0]	ac(v)
	} else {	
	if(MaxNumSubblockMergeCand > 0 && cbWidth >= 8 && cbHeight >= 8)	
	merge_subblock_flag [x0][y0]	ac(v)
	if(merge_subblock_flag[x0][y0] == 1) {	
	if(MaxNumSubblockMergeCand > 1)	
	merge_subblock_idx [x0][y0]	ac(v)
	} else {	
	if((cbWidth * cbHeight) >= 64 && (sps_ciip_enabled_flag &&	

[0712]

cu_skip_flag[x0][y0] == 0 && cbWidth < 128 && cbHeight < 128) (sps_triangle_enabled_flag && SliceMaxNumTriangleMergeCand > 1 && slice_type == B))	
regular_merge_flag [x0][y0]	ae(v)
if (regular_merge_flag[x0][y0] == 1) {	
if(sps_mmvd_enabled_flag)	
mmvd_merge_flag [x0][y0]	ae(v)
if(mmvd_merge_flag[x0][y0] == 1) {	
if(MaxNumMergeCand > 1)	
mmvd_cand_flag [x0][y0]	ae(v)
mmvd_distance_idx [x0][y0]	ae(v)
mmvd_direction_idx [x0][y0]	ae(v)
} else {	
if(MaxNumMergeCand > 1)	
merge_idx [x0][y0]	ae(v)
}	
} else {	
if(sps_ciip_enabled_flag && sps_triangle_enabled_flag && SliceMaxNumTriangleMergeCand > 1 && weightedPredFlag == 0 && slice_type == B && cu_skip_flag[x0][y0] == 0 && (cbWidth * cbHeight) >= 64 && cbWidth < 128 && cbHeight < 128) {	
ciip_flag [x0][y0]	ae(v)
if(ciip_flag[x0][y0] && MaxNumMergeCand > 1)	
merge_idx [x0][y0]	ae(v)
if(!ciip_flag[x0][y0] && SliceMaxNumTriangleMergeCand > 1) {	
merge_triangle_split_dir [x0][y0]	ae(v)
merge_triangle_idx0 [x0][y0]	ae(v)
merge_triangle_idx1 [x0][y0]	ae(v)
}	
}	
}	
}	
}	

[0713] 当非矩形帧间预测模式为GEO模式时,进一步描述以下示例。

[0714] 不同的机制可以用于控制GEO/TPM融合模式,取决于WP是否应用于通过以下方式从中获取参考块P0和P1的参考图像:

[0715] -将表14中列出的WP参数从SH移动到PH;

[0716] -将GEO参数从PH移回SH;

[0717] -改变MaxNumGeoMergeCand的语义,例如,当可以使用经过WP的参考图像(例如,至少一个标志lumaWeightedFlag为真)时,将这些条带的MaxNumGeoMergeCand设置为0或1。

[0718] 对于GE0融合模式,示例性参考块P0和P1在图8中分别表示为810和820。

[0719] 在一个示例中,当非矩形模式(例如,GE0和TPM)的启用和WP参数在图像头中指示时,可以使用以下语法,如下表所示:

[0720] 表:图像头RBSP语法

[0721]

picture_header_rbsp() {	描述符
non_reference_picture_flag	u(1)
gdr_pic_flag	u(1)
no_output_of_prior_pics_flag	u(1)
if(gdr_pic_flag)	
recovery_poc_cnt	ue(v)
ph_pic_parameter_set_id	ue(v)
if(sps_poc_msb_flag) {	
ph_poc_msb_present_flag	u(1)
if(ph_poc_msb_present_flag)	
poc_msb_val	u(v)
}	
...	
pic_rpl_present_flag	u(1)
if(pic_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	

[0722]

if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
...	
if((pps_weighted_pred_flag && slice_type == P) (pps_weighted_bipred_flag && slice_type == B))	
pred_weight_table()	
...	
...	
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1 && WPDisabled)	
pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
...	
rbsp_trailing_bits()	
}	

[0723]

[0724] 当luma_weight_10_flag[i]、chroma_weight_10_flag[i]、luma_weight_11_flag[j]和chroma_weight_11_flag[j]的所有值都被设置为0时,变量WPDisabled被设置为1,其中,i=0..NumRefIdxActive[0],j=0..NumRefIdxActive[1];否则,WPDisabled的值被设置为0。

[0725] 当变量WPDisabled被设置为0时,pic_max_num_merge_cand_minus_max_num_geo_cand的值被设置为MaxNumMergeCand。

[0726] 在另一个示例中, `pic_max_num_merge_cand_minus_max_num_geo_cand` 被设置为 `MaxNumMergeCand-1`。

[0727] 在一个示例中, WP 参数的指示和非矩形模式 (例如, GEO 和 TPM) 的启用在条带头中执行。下表给出了示例性语法。

[0728]	<code>slice_header() {</code>	描述符
	<code> slice_pic_order_cnt_lsb</code>	<code>u(v)</code>
	<code> if(subpics_present_flag)</code>	
	<code> slice_subpic_id</code>	<code>u(v)</code>
	<code> if(rect_slice_flag NumTilesInPic > 1)</code>	
	<code> slice_address</code>	<code>u(v)</code>
	<code> if(!rect_slice_flag && NumTilesInPic > 1)</code>	
	<code> num_tiles_in_slice_minus1</code>	<code>ue(v)</code>
	<code> slice_type</code>	<code>ue(v)</code>
	<code> if(!pic_rpl_present_flag && (nal_unit_type != IDR_W_RADL && nal_unit_type !=</code>	
	<code> IDR_N_LP) sps_idr_rpl_present_flag) {</code>	
	<code> for(i = 0; i < 2; i++) {</code>	
	<code> if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] &&</code>	
	<code> (i == 0 (i == 1 && rpl1_idx_present_flag)))</code>	
	<code> slice_rpl_sps_flag[i]</code>	<code>u(1)</code>
	<code> if(slice_rpl_sps_flag[i]) {</code>	
	<code> if(num_ref_pic_lists_in_sps[i] > 1 &&</code>	
	<code> (i == 0 (i == 1 && rpl1_idx_present_flag)))</code>	
	<code> slice_rpl_idx[i]</code>	<code>u(v)</code>
	<code> } else</code>	
	<code> ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])</code>	
	<code> for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {</code>	

[0729]

if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
slice_poc_lsb_lf [i][j]	u(v)
slice_delta_poc_msb_present_flag [i][j]	u(1)
if(slice_delta_poc_msb_present_flag[i][j])	
slice_delta_poc_msb_cycle_lf [i][j]	ue(v)
}	
}	
}	
if(pic_rpl_present_flag ((nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) sps_idr_rpl_present_flag)) {	
if((slice_type != I && num_ref_entries[0][RplIdx[0]] > 1) (slice_type == B && num_ref_entries[1][RplIdx[1]] > 1)) {	
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag)	
for(i = 0; i < (slice_type == B ? 2 : 1); i++)	
if(num_ref_entries[i][RplIdx[i]] > 1)	
num_ref_idx_active_minus1 [i]	ue(v)
}	
}	
if(slice_type != I) {	
if(cabac_init_present_flag)	
cabac_init_flag	u(1)
if(pic_temporal_mvp_enabled_flag) {	
if(slice_type == B && !pps_collocated_from_l0_idc)	
collocated_from_l0_flag	u(1)
if((collocated_from_l0_flag && NumRefIdxActive[0] > 1) (!collocated_from_l0_flag && NumRefIdxActive[1] > 1))	
collocated_ref_idx	ue(v)

[0730]

}	
if (slice_weighted_pred_flag)	
pred_weight_table()	
if(!pps_six_minus_max_num_merge_cand_plus1)	
six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag && !pps_five_minus_max_num_subblock_merge_cand_plus1)	
five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
slice_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_dmvr_slice_present_flag)	
slice_disable_bdof_dmvr_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_minus1 && WPDIsabled) {	
max_num_merge_cand_minus_max_num_geo_cand	ue(v)
}	
...	
}	
slice_qp_delta	se(v)
if(pps_slice_chroma_qp_offsets_present_flag) {	
slice_cb_qp_offset	se(v)
slice_cr_qp_offset	se(v)
if(sps_joint_cbr_enabled_flag)	
slice_joint_cbr_qp_offset	se(v)
}	
if(pps_cu_chroma_qp_offset_list_enabled_flag)	
cu_chroma_qp_offset_enabled_flag	u(1)
if(sps_sao_enabled_flag && !pic_sao_enabled_present_flag) {	

[0731]

slice_sao_luma_flag	u(1)
if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
if(sps_alf_enabled_flag && !pic_alf_enabled_present_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
}	
}	
if(deblocking_filter_override_enabled_flag && !pic_deblocking_filter_override_present_flag)	
slice_deblocking_filter_override_flag	u(1)
if(slice_deblocking_filter_override_flag) {	
slice_deblocking_filter_disabled_flag	u(1)
if(!slice_deblocking_filter_disabled_flag) {	
slice_beta_offset_div2	se(v)
slice_tc_offset_div2	se(v)
}	
}	
if(entry_point_offsets_present_flag && NumEntryPoints > 0) {	
offset_len_minus1	ue(v)

[0732]

for(i = 0; i < NumEntryPoints; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)
for(i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte[i]	u(8)
}	
byte_alignment()	
}	

[0733] 当luma_weight_10_flag[i]、chroma_weight_10_flag[i]、luma_weight_11_flag[j]和chroma_weight_11_flag[j]的所有值都被设置为0时,变量WPDisabled被设置为1,其中,i=0..NumRefIdxActive[0],j=0..NumRefIdxActive[1];否则,WPDisabled的值被设置为0。

[0734] 当变量WPDisabled被设置为0时,max_num_merge_cand_minus_max_num_geo_cand的值被设置为MaxNumMergeCand。

[0735] 在另一个实施例中,当变量WPDisabled被设置为0时,max_num_merge_cand_minus_max_num_geo_cand的值被设置为MaxNumMergeCand-1。

[0736] 在上述公开的示例中,加权预测参数可以在图像头或条带头中指示。

[0737] 在其它实施例中,根据以下条件之一,在条带头处定义变量SliceMaxNumGeoMergeCand:

[0738] -SliceMaxNumGeoMergeCand=(lumaWeightedFlag||chromaWeightedFlag)?0:MaxNumGeoMergeCand;

[0739] -SliceMaxNumGeoMergeCand=(lumaWeightedFlag||chromaWeightedFlag)?1:MaxNumGeoMergeCand;

[0740] -SliceMaxNumGeoMergeCand=slice_weighted_pred_flag?0:MaxNumGeoMergeCand;或者SliceMaxNumGeoMergeCand=slice_weighted_pred_flag?1:MaxNumGeoMergeCand。

[0741] 不同的实施例使用上面列出的不同情况。

[0742] 变量SliceMaxNumGeoMergeCand的值进一步用于在块级解析融合信息。下表给出了示例性语法。

[0743] 7.3.9.7融合数据语法

[0744]

merge_data(x0, y0, cbWidth, cbHeight, chType) {	描述符
if(CuPredMode[chType][x0][y0] == MODE_IBC) {	
if(MaxNumIbcMergeCand > 1)	
merge_idx [x0][y0]	ac(v)
} else {	

[0745]

if(MaxNumSubblockMergeCand > 0 && cbWidth >= 8 && cbHeight >= 8)	
merge_subblock_flag [x0][y0]	ae(v)
if(merge_subblock_flag[x0][y0] == 1) {	
if(MaxNumSubblockMergeCand > 1)	
merge_subblock_idx [x0][y0]	ae(v)
} else {	
if(sps_ciip_enabled_flag && cu_skip_flag[x0][y0] == 0 && (cbWidth * cbHeight) >= 64 && cbWidth < 128 && cbHeight < 128) (sps_geo_enabled_flag && SliceMaxNumGeoMergeCand > 1 && cbWidth >= 8 && cbHeight >= 8 && slice_type == B))	
regular_merge_flag [x0][y0]	ae(v)
if(regular_merge_flag[x0][y0] == 1) {	
if(sps_mmvd_enabled_flag)	
mmvd_merge_flag [x0][y0]	ae(v)
if(mmvd_merge_flag[x0][y0] == 1) {	
if(MaxNumMergeCand > 1)	
mmvd_cand_flag [x0][y0]	ae(v)
mmvd_distance_idx [x0][y0]	ae(v)
mmvd_direction_idx [x0][y0]	ae(v)
} else if(MaxNumMergeCand > 1)	
merge_idx [x0][y0]	ae(v)
} else {	
if(sps_ciip_enabled_flag && sps_geo_enabled_flag && SliceMaxNumGeoMergeCand > 1 && slice_type == B && cu_skip_flag[x0][y0] == 0 && cbWidth >= 8 && cbHeight >= 8 && cbWidth < 128 && cbHeight < 128)	
ciip_flag [x0][y0]	ae(v)
if(ciip_flag[x0][y0] && MaxNumMergeCand > 1)	
merge_idx [x0][y0]	ae(v)
if(!ciip_flag[x0][y0] && SliceMaxNumGeoMergeCand > 1) {	
merge_geo_partition_idx [x0][y0]	ae(v)
merge_geo_idx0 [x0][y0]	ae(v)
if(SliceMaxNumGeoMergeCand > 2)	
merge_geo_idx1 [x0][y0]	ae(v)

[0746]	}	
	}	
	}	
	}	
	}	

[0747] 相关图像头语义如下所述：

[0748] `pic_max_num_merge_cand_minus_max_num_geo_cand`表示从`MaxNumMergeCand`中减去与图像头关联的条带中支持的geo融合模式候选的最大数量。

[0749] 当`pic_max_num_merge_cand_minus_max_num_geo_cand`不存在, `sps_geo_enabled_flag`等于1, 且`MaxNumMergeCand`大于或等于2时, `pic_max_num_merge_cand_minus_max_num_geo_cand`被推断为`pps_max_num_merge_cand_minus_max_num_geo_cand_plus1-1`。

[0750] geo融合模式候选的最大数量`MaxNumGeoMergeCand`推导如下：

[0751]
$$\text{MaxNumGeoMergeCand} = \text{MaxNumMergeCand} - \text{pic_max_num_merge_cand_minus_max_num_geo_cand}$$

[0752] 当`pic_max_num_merge_cand_minus_max_num_geo_cand`存在时, `MaxNumGeoMergeCand`的取值范围应为2至`MaxNumMergeCand` (包括端值)。

[0753] 当`pic_max_num_merge_cand_minus_max_num_geo_cand`不存在 (`sps_geo_enabled_flag`等于0或`MaxNumMergeCand`小于2) 时, `MaxNumGeoMergeCand`被设置为0。

[0754] 当`MaxNumGeoMergeCand`等于0时, 不允许geo融合模式用于与PH关联的条带。

[0755] 在以下示例中, 考虑了几个与指示相关的方面。这些方面如下所述：

[0756] -与融合模式候选的数量相关的语法元素()在序列参数集(sequence parameter set, SPS)中指示, 这使得特定实现方式可以推导SPS级的非矩形模式融合候选的数量(`MaxNumGeoMergeCand`)；

[0757] -当图像只包括一个条带时, PH可以在SH中指示；

[0758] -定义一种PH/SH参数覆盖机制, 包括以下内容：

[0759] PPS标志, 用于表示相关编码工具的语法元素是存在于PH中还是存在于SH中 (但能同时存在于两者中)。

[0760] 具体地, 参考图像列表和加权预测表可以使用该机制。

[0761] -预测权重表, 即可以在PH或SH中指示的第五类数据 (如ALF、去块效应滤波、RPL和SAO)；

[0762] -当对图像启动加权预测时, 该图像中的所有条带都需要具有相同的参考图像列表；

[0763] -如果在与PH关联的图像中只使用一些条带类型, 则在一定条件下指示帧间和帧内相关语法元素。

[0764] 具体地, 引入了两个标志, 即`pic_inter_slice_present_flag`和`pic_intra_slice_present_flag`。

[0765] 在一个示例中, 与融合模式候选的数量相关的语法元素()在序列参数集

(sequence parameter set, SPS) 中指示, 这使得特定实现方式可以推导SPS级的非矩形模式融合候选的数量 (MaxNumGeoMergeCand)。该方面可以根据以下语法通过编码过程或解码过程来实现。

[0766] 7.3.2.3 序列参数集RBSP语法

[0767]

seq_parameter_set_rbsp() {	描述符
sps_decoding_parameter_set_id	u(4)
sps_video_parameter_set_id	u(4)
sps_max_sublayers_minus1	u(3)
sps_reserved_zero_4bits	u(4)
sps_ptl_dpb_hrd_params_present_flag	u(1)
if(sps_ptl_dpb_hrd_params_present_flag)	
profile_tier_level(1, sps_max_sublayers_minus1)	
gdr_enabled_flag	u(1)
sps_seq_parameter_set_id	u(4)
...	
sps_sbt_enabled_flag	u(1)
sps_affine_enabled_flag	u(1)
if(sps_affine_enabled_flag) {	
sps_affine_type_flag	u(1)
sps_affine_amvr_enabled_flag	u(1)
sps_affine_prof_enabled_flag	u(1)
if(sps_affine_prof_enabled_flag)	
sps_prof_pic_present_flag	u(1)
}	
if(chroma_format_idc == 3) {	
sps_palette_enabled_flag	u(1)
sps_act_enabled_flag	u(1)
}	
sps_bcw_enabled_flag	u(1)
sps_ibc_enabled_flag	u(1)
sps_ciip_enabled_flag	u(1)
if(sps_mmvd_enabled_flag)	

[0768]	sps_fpel_mmvd_enabled_flag	u(1)
	sps_geo_enabled_flag	u(1)
	sps_six_minus_max_num_merge_cand_plus1	ue(v)
	if (sps_geo_enabled_flag)	
	sps_max_num_merge_cand_minus_max_num_geo_cand_plus1	ue(v)
	sps_lmcs_enabled_flag	u(1)
	sps_lfnst_enabled_flag	u(1)
	sps_ladf_enabled_flag	u(1)
	...	

[0769] 上文描述的语法具有以下语义。

[0770] sps_six_minus_max_num_merge_cand_plus1等于0表示pic_six_minus_max_num_merge_cand存在于参考PPS的PH中。sps_six_minus_max_num_merge_cand_plus1大于0表示pic_six_minus_max_num_merge_cand不存在于参考PPS的PH中。sps_six_minus_max_num_merge_cand_plus1的取值范围应为0至6(包括端值)。

[0771] sps_max_num_merge_cand_minus_max_num_geo_cand_plus1等于0表示pic_max_num_merge_cand_minus_max_num_geo_cand存在于参考PPS的条带的PH中。sps_max_num_merge_cand_minus_max_num_geo_cand_plus1大于0表示pic_max_num_merge_cand_minus_max_num_geo_cand不存在于参考PPS的PH中。sps_max_num_merge_cand_minus_max_num_geo_cand_plus1的取值范围应为0至MaxNumMergeCand-1。

[0772] PH的对应元素的语义如下：

[0773] pic_six_minus_max_num_merge_cand表示从6中减去与PH关联的条带中支持的融合运动矢量预测(motion vector prediction, MVP) 候选的最大数量。融合MVP候选的最大数量MaxNumMergeCand推导如下：

[0774] $\text{MaxNumMergeCand} = 6 - \text{pic_six_minus_max_num_merge_cand}$

[0775] MaxNumMergeCand的取值范围应为1至6(包括端值)。当pic_six_minus_max_num_merge_cand不存在时, pic_six_minus_max_num_merge_cand的值被推断为sps_six_minus_max_num_merge_cand_plus1-1。

[0776] pic_max_num_merge_cand_minus_max_num_geo_cand表示从MaxNumMergeCand中减去与图像头关联的条带中支持的geo融合模式候选的最大数量。

[0777] 当sps_max_num_merge_cand_minus_max_num_geo_cand不存在, sps_geo_enabled_flag等于1, 且MaxNumMergeCand大于或等于2时, pic_max_num_merge_cand_minus_max_num_geo_cand被推断为sps_max_num_merge_cand_minus_max_num_geo_cand_plus1-1。

[0778] geo融合模式候选的最大数量MaxNumGeoMergeCand推导如下：

[0779] $\text{MaxNumGeoMergeCand} = \text{MaxNumMergeCand} - \text{pic_max_num_merge_cand_minus_max_num_geo_cand}$

[0780] 当pic_max_num_merge_cand_minus_max_num_geo_cand存在时, MaxNumGeoMergeCand的取值范围应为2至MaxNumMergeCand(包括端值)。

[0781] 当pic_max_num_merge_cand_minus_max_num_geo_cand不存在(sps_geo_enabled_flag等于0或MaxNumMergeCand小于2)时, MaxNumGeoMergeCand被设置为0。

[0782] 当MaxNumGeoMergeCand等于0时, 不允许geo融合模式用于与PH关联的条带。

[0783] 可选地, max_num_merge_cand_minus_max_num_geo_cand表示从MaxNumMergeCand中减去SPS中支持的GEO融合模式候选的最大数量。

[0784] 当sps_geo_enabled_flag等于1且MaxNumMergeCand大于或等于3时, GEO融合模式候选的最大数量MaxNumGeoMergeCand推导如下:

[0785] $\text{MaxNumGeoMergeCand} = \text{MaxNumMergeCand} - \text{max_num_merge_cand_minus_max_num_geo_cand}$

[0786] 如果sps_geo_enabled_flag的值等于1, MaxNumGeoMergeCand的取值范围应为2至MaxNumMergeCand(包括端值)。

[0787] 如果sps_geo_enabled_flag等于1且MaxNumMergeCand等于2, 则MaxNumGeoMergeCand被设置为2。

[0788] 否则, MaxNumGeoMergeCand被设置为0。

[0789] 该示例的可选语法和语义如下:

[0790]	...	
	sps_geo_enabled_flag	u(1)
	sps_six_minus_max_num_merge_cand	ue(v)
	if (sps_geo_enabled_flag)	
	sps_max_num_merge_cand_minus_max_num_geo_cand	ue(v)
	...	

[0791] sps_six_minus_max_num_merge_cand表示从6中减去与PH关联的条带中支持的融合运动矢量预测(motion vector prediction, MVP) 候选的最大数量。融合MVP候选的最大数量MaxNumMergeCand推导如下:

[0792] $\text{MaxNumMergeCand} = 6 - \text{sps_six_minus_max_num_merge_cand}$

[0793] MaxNumMergeCand的取值范围应为1至6(包括端值)。

[0794] sps_six_minus_max_num_merge_cand表示从MaxNumMergeCand中减去与图像头关联的条带中支持的geo融合模式候选的最大数量。

[0795] geo融合模式候选的最大数量MaxNumGeoMergeCand推导如下:

[0796] $\text{MaxNumGeoMergeCand} = \text{MaxNumMergeCand} - \text{sps_max_num_merge_cand_minus_max_num_geo_cand}$

[0797] 当sps_max_num_merge_cand_minus_max_num_geo_cand存在时, MaxNumGeoMergeCand的取值范围应为2至MaxNumMergeCand(包括端值)。

[0798] 当sps_max_num_merge_cand_minus_max_num_geo_cand不存在(sps_geo_enabled_flag等于0或MaxNumMergeCand小于2)时, MaxNumGeoMergeCand被设置为0。

[0799] 当MaxNumGeoMergeCand等于0时,不使用geo融合模式。

[0800] 对于上述示例和两种可选语法定义,检查是否启用加权预测。这种检查会影响变量MaxNumGeoMergeCand的推导,并且在以下其中一种情况下,MaxNumGeoMergeCand的值被设置为0:

[0801] -当 $i=0..NumRefIdxActive[0]$, $j=0..NumRefIdxActive[1]$ 时,luma_weight_10_flag[i]、chroma_weight_10_flag[i]、luma_weight_11_flag[j]和chroma_weight_11_flag[j]被设置为0或不存在;

[0802] -SPS或PPS中的标志指示存在双向加权预测(pps_weighted_bipred_flag);

[0803] -在图像头(picture header,PH)或条带头(slice header,SH)中指示存在双向加权预测。

[0804] 指示存在加权预测参数的SPS级标志可以如下所示:

[0805]	sps_bcw_enabled_flag	u(1)
	sps_ibc_enabled_flag	u(1)
	sps_ciip_enabled_flag	u(1)
	if(sps_mmvd_enabled_flag)	
	sps_fpel_mmvd_enabled_flag	u(1)
	sps_wp_enabled_flag	
	if (!sps_wp_enabled_flag)	
	sps_geo_enabled_flag	u(1)
	sps_six_minus_max_num_merge_cand_plus1	ue(v)
	if (sps_geo_enabled_flag)	
	sps_max_num_merge_cand_minus_max_num_geo_cand_plus1	ue(v)
	sps_lmcs_enabled_flag	u(1)
	sps_lfnst_enabled_flag	u(1)
	sps_ladf_enabled_flag	u(1)
	...	

[0806] 语法元素“sps_wp_enabled_flag”确定是否可以在低级别(PPS、PH或SH)上启用加权预测。下文给出了示例性实现方式。

[0807]	if(pps_cu_chroma_qp_offset_list_enabled_flag) {	
	chroma_qp_offset_list_len_minus1	ue(v)

[0808]	for(i = 0; i <= chroma_qp_offset_list_len_minus1; i++) {	
	cb_qp_offset_list[i]	se(v)
	cr_qp_offset_list[i]	se(v)
	if(pps_joint_cbr_qp_offset_present_flag)	
	joint_cbr_qp_offset_list[i]	se(v)
	}	
	}	
	if (sps_wp_enabled_flag) {	
	pps_weighted_pred_flag	u(1)
	pps_weighted_bipred_flag	u(1)
	}	

[0809] 在上表中,pps_weighted_pred_flag和pps_weighted_bipred_flag是码流中的标志,表示是否对单向预测块和双向预测块启用加权预测。

[0810] 在一个示例中,当加权预测标志在图像头中指定为pic_weighted_pred_flag和pic_weighted_bipred_flag等时,以下对sps_wp_enabled_flag的依赖性可以在码流语法中指定:

[0811]	...	
	if (sps_wp_enabled_flag) {	
	pic_weighted_pred_flag	
	pic_weighted_bipred_flag	
	}	
	...	

[0812] 在一个示例中,如果加权预测标志在条带头中指定为weighted_pred_flag和weighted_bipred_flag等,则以下对sps_wp_enabled_flag的依赖性可以在码流语法中指定:

[0813]	...	
	if (sps_wp_enabled_flag) {	
	weighted_pred_flag	
	weighted_bipred_flag	
	}	
	...	

[0814] 在一个示例中,参考图像列表可以在PPS或PH或SH(但不能同时在两者中)指示。在

一些示例中,参考图像列表的指示依赖于指示存在加权预测的语法元素(例如,pps_weighted_pred_flag和pps_weighted_bipred_flag)。因此,根据参考图像列表是否在PPS、PH或SH中指示,加权预测参数在将PPS、PH或SH中相应地指示在参考图像列表中之前指示。

[0815] 针对本实施例,可以指定以下语法:

[0816] 图像参数集语法

	pic_parameter_set_rbsp() {	描述符
	...	
	rpl_present_in_ph_flag	u(1)
	sao_present_in_ph_flag	u(1)
	alf_present_in_ph_flag	u(1)
	...	
	pps_weighted_pred_flag	u(1)
	pps_weighted_bipred_flag	u(1)
	if(pps_weighted_pred_flag pps_weighted_bipred_flag rpl_present_in_ph_flag)	
	weighted_pred_table_present_in_ph_flag	u(1)
	deblocking_filter_control_present_flag	u(1)
[0817]	if(deblocking_filter_control_present_flag) {	
	deblocking_filter_override_enabled_flag	u(1)
	if(deblocking_filter_override_enabled_flag)	
	deblocking_filter_override_present_in_ph_flag	u(1)
	pps_deblocking_filter_disabled_flag	u(1)
	if(!pps_deblocking_filter_disabled_flag) {	
	pps_beta_offset_div2	se(v)
	pps_tc_offset_div2	se(v)
	}	
	}	
	constant_slice_header_params_enabled_flag	u(1)
	...	
[0818]	}	

[0819]

[0820] rpl_present_in_ph_flag等于1表示参考图像列表指示不存在于参考PPS的条带头中,但可能存在于参考PPS的PH中。rpl_present_in_ph_flag等于0表示参考图像列表指

示不存在于参考PPS的PH中,但可能存在于参考PPS的条带头中。

[0821] sao_present_in_ph_flag等于1表示用于启用SAO使用的语法元素不存在于参考PPS的条带头中,但可能存在于参考PPS的PH中。sao_present_in_ph_flag等于0表示用于启用SAO使用的语法元素不存在于参考PPS的PH中,但可能存在于参考PPS的条带头中。

[0822] alf_present_in_ph_flag等于1表示用于启用ALF使用的语法元素不存在于参考PPS的条带头中,但可能存在于参考PPS的PH中。alf_present_in_ph_flag等于0表示用于启用ALF使用的语法元素不存在于参考PPS的PH中,但可能存在于参考PPS的条带头中。

[0823]

[0824] weighted_pred_table_present_in_ph_flag等于1表示加权预测表不存在于参考PPS的条带头中,但可能存在于参考PPS的PH中。weighted_pred_table_present_in_ph_flag等于0表示加权预测表不存在于参考PPS的PH中,但可能存在于参考PPS的条带头中。当weighted_pred_table_present_in_ph_flag不存在时,weighted_pred_table_present_in_ph_flag的值被推断为0。

[0825]

[0826] deblocking_filter_override_enabled_flag等于1表示去块效应滤波器覆盖可能存在于参考PPS的PH或条带头中。deblocking_filter_override_enabled_flag等于0表示去块效应滤波器覆盖不存在于参考PPS的PH或条带中。当deblocking_filter_override_enabled_flag不存在时,deblocking_filter_override_enabled_flag的值被推断为0。

[0827] deblocking_filter_override_present_in_ph_flag等于1表示去块效应滤波器覆盖不存在于参考PPS的条带头中,但可能存在于参考PPS的PH中。deblocking_filter_override_present_in_ph_flag等于0表示去块滤波器覆盖不存在于参考PPS的PH中,但可能存在于参考PPS的条带头中。

[0828]

[0829]

picture_header_rbsp() {	描述符
...	
...	
if((pps_weighted_pred_flag pps_weighted_bipred_flag) && weighted_pred_table_present_in_ph_flag)	
pred_weight_table()	
...	
if(rpl_present_in_ph_flag) {	
for(i = 0; i < 2; i++) {	

[0830]

if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
if(sps_sao_enabled_flag && sao_present_in_ph_flag) {	
pic_sao_luma_enabled_flag	u(1)
if(ChromaArrayType != 0)	
pic_sao_chroma_enabled_flag	u(1)
——}	
}	
if(sps_alf_enabled_flag && alf_present_in_ph_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	

[0831]

pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
...	
if(deblocking_filter_override_enabled_flag && deblocking_filter_override_present_in_ph_flag)	
pic_deblocking_filter_override_flag	u(1)
if(pic_deblocking_filter_override_flag) {	
pic_deblocking_filter_disabled_flag	u(1)
if(!pic_deblocking_filter_disabled_flag) {	
pic_beta_offset_div2	se(v)
pic_tc_offset_div2	se(v)
}	
}	
...	
}	

[0832]

slice_header() {	描述符
...	
if(!rpl_present_in_ph_flag &&((nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) sps_idr_rpl_present_flag)) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	

[0833]

slice_rpl_sps_flag[i]	u(1)
if(slice_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 &&	
(i == 0 (i == 1 && rpl1_idx_present_flag)))	
slice_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
slice_poc_lsb_lt[i][j]	u(v)
slice_delta_poc_msb_present_flag[i][j]	u(1)
if(slice_delta_poc_msb_present_flag[i][j])	
slice_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
if(rpl_present_in_ph_flag ((nal_unit_type != IDR_W_RADL && nal_unit_type !=	
IDR_N_LP) sps_idr_rpl_present_flag)) {	
if((slice_type != I && num_ref_entries[0][RplIdx[0]] > 1)	
(slice_type == B && num_ref_entries[1][RplIdx[1]] > 1)) {	
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag)	
for(i = 0; i < (slice_type == B ? 2 : 1); i++)	
if(num_ref_entries[i][RplIdx[i]] > 1)	
num_ref_idx_active_minus1[i]	ue(v)
}	
}	
}	
...	

[0834]

if(slice_type != I) {	
...	
if(((pps_weighted_pred_flag && slice_type == P) (pps_weighted_bipred_flag && slice_type == B)) && !weighted_pred_table_present_in_ph_flag)	
pred_weight_table()	
}	
if(sps_sao_enabled_flag && !sao_present_in_ph_flag) {	
slice_sao_luma_flag	u(1)
if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
if(sps_alf_enabled_flag && !alf_present_in_ph_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
}	
}	
if(deblocking_filter_override_enabled_flag && !deblocking_filter_override_present_in_ph_flag)	
slice_deblocking_filter_override_flag	u(1)
if(slice_deblocking_filter_override_flag) {	
slice_deblocking_filter_disabled_flag	u(1)

[0835]

if(!slice_deblocking_filter_disabled_flag) {	
slice_beta_offset_div2	se(v)
slice_tc_offset_div2	se(v)
}	
}	
...	
}	

[0836] 图像头的一种可选语法如下：

[0837]

picture_header_rbsp() {	描述符
...	
if(rpl_present_in_ph_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	

[0838]

}	
...	
if((pps_weighted_pred_flag pps_weighted_bipred_flag) && weighted_pred_table_present_in_ph_flag)	
pred_weight_table()	
...	
if(sps_sao_enabled_flag && sao_present_in_ph_flag) {	
pic_sao_luma_enabled_flag	u(1)
if(ChromaArrayType != 0)	
pic_sao_chroma_enabled_flag	u(1)
——}	
}	
if(sps_alf_enabled_flag && alf_present_in_ph_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
}	
...	
if(deblocking_filter_override_enabled_flag && deblocking_filter_override_present_in_ph_flag)	
pic_deblocking_filter_override_flag	u(1)
if(pic_deblocking_filter_override_flag) {	

[0839]	pic_deblocking_filter_disabled_flag	u(1)
	if(!pic_deblocking_filter_disabled_flag) {	
	pic_beta_offset_div2	se(v)
	pic_tc_offset_div2	se(v)
	}	
	}	
	...	
	}	

[0840] 在另一个示例中,图像头和条带头元素的指示可以在单个过程中组合。

[0841] 该示例引入了一个标志 (picture_header_in_slice_header_flag), 该标志表示图像头和条带头是否组合。根据该示例,码流的语法如下:

[0842] 图像头RBSP语法

[0843]	picture_header_rbsp() {	描述符
	picture_header_structure()	
	}	

[0844] 图像头结构语法

[0845]	picture_header_structure() {	描述符
	non_reference_picture_flag	u(1)
	gdr_pic_flag	u(1)
	no_output_of_prior_pics_flag	u(1)
	if(gdr_pic_flag)	
	recovery_poc_cnt	ue(v)
	ph_pic_parameter_set_id	ue(v)
	...	
	}	

[0846] 通用条带头语法

[0847]	slice_header() {	描述符
	picture_header_in_slice_header_flag	u(1)
	if(picture_header_in_slice_header_flag)	

[0848]

picture_header_structure()	
if(subpics_present_flag)	
slice_subpic_id	u(v)
if(rect_slice_flag NumTilesInPic > 1)	
slice_address	u(v)
if(!rect_slice_flag && NumTilesInPic > 1)	
num_tiles_in_slice_minus1	ue(v)
slice_type	ue(v)
if(!pic_rpl_present_flag && (nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) sps_idr_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
slice_rpl_sps_flag[i]	u(1)
...	
}	

[0849] picture_header_in_slice_header_flag的语义和相关码流约束条件如下：

[0850] picture_header_in_slice_header_flag等于1表示图像头语法结构存在于条带头中。picture_header_in_slice_header_flag等于0表示图像头语法结构不存在于条带头中。

[0851] 码流一致性的要求为：picture_header_in_slice_header_flag的值对于CLVS中的所有条带是相同的。

[0852] 当picture_header_in_slice_header_flag等于1时，码流一致性的要求为：CLVS中不存在NAL单元类型为PH_NUT的NAL单元。

[0853] 当picture_header_in_slice_header_flag等于0时，码流一致性的要求为：PU中存在NAL单元类型为PH_NUT的NAL单元，该NAL单元在PU的第一个VCL NAL单元之前。

[0854] 这些例子的各个方面组合如下。

[0855] 当picture_header_in_slice_header_flag等于0时，这些标志表示相关编码工具的语法元素是否存在于PH或SH中(但不能同时存在于两者中)；

[0856] 否则(当picture_header_in_slice_header_flag等于1时)，这些标志被推断为0，表示条带级的工具参数指示。

[0857] 一种可选组合如下所述：

[0858] 当picture_header_in_slice_header_flag等于0时，这些标志表示相关编码工具的语法元素是否存在于PH或SH中(但不能同时存在于两者中)；

[0859] 否则(当picture_header_in_slice_header_flag等于1时)，这些标志被推断为0，

表示图像头级的工具参数指示。

[0860] 这种组合具有以下语法：

[0861] 图像参数集语法

[0862]

pic_parameter_set_rbsp() {	描述符
picture_header_in_slice_header_flag	u(1)
...	
if(picture_header_in_slice_header_flag)	
{	
rpl_present_in_ph_flag	u(1)
sao_present_in_ph_flag	u(1)
alf_present_in_ph_flag	u(1)
}	
...	
pps_weighted_pred_flag	u(1)
pps_weighted_bipred_flag	u(1)
if(pps_weighted_pred_flag pps_weighted_bipred_flag rpl_present_in_ph_flag)	
weighted_pred_table_present_in_ph_flag	u(1)
deblocking_filter_control_present_flag	u(1)
if(deblocking_filter_control_present_flag) {	
deblocking_filter_override_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag)	
deblocking_filter_override_present_in_ph_flag	u(1)
pps_deblocking_filter_disabled_flag	u(1)
if(!pps_deblocking_filter_disabled_flag) {	
pps_beta_offset_div2	se(v)
pps_tc_offset_div2	se(v)
}	

[0863]

}	
constant_slice_header_params_enabled_flag	u(1)
...	
}	

[0864] 在该示例中,通过指示参考图像列表中通过加权预测引用的条目的数量,检查是否启用加权预测。

[0865] 该示例中的语法和语义定义如下:

[0866]

pred_weight_table() {	描述符
luma_log2_weight_denom	uc(v)
if(ChromaArrayType != 0)	
delta_chroma_log2_weight_denom	se(v)
num_l0_weighted_ref_pics	uc(v)
for(i = 0; i < num_l0_weighted_ref_pics; i++)	
luma_weight_l0_flag[i]	u(1)
if(ChromaArrayType != 0)	
for(i = 0; i < NumRefIdxActive[0]; i++)	
chroma_weight_l0_flag[i]	u(1)
for(i = 0; i < NumRefIdxActive[0]; i++) {	
if(luma_weight_l0_flag[i]) {	
delta_luma_weight_l0[i]	se(v)
luma_offset_l0[i]	se(v)
}	
if(chroma_weight_l0_flag[i])	
for(j = 0; j < 2; j++) {	
delta_chroma_weight_l0[i][j]	se(v)
delta_chroma_offset_l0[i][j]	se(v)
}	
}	
num_l1_weighted_ref_pics	uc(v)

[0867]	for(i = 0; i < num_l1_weighted_ref_pics; i++)	
	luma_weight_l1_flag[i]	u(1)
	if(ChromaArrayType != 0)	
	for(i = 0; i < NumRefIdxActive[1]; i++)	
	chroma_weight_l1_flag[i]	u(1)
	for(i = 0; i < NumRefIdxActive[1]; i++) {	
	if(luma_weight_l1_flag[i]) {	
	delta_luma_weight_l1[i]	se(v)
	luma_offset_l1[i]	se(v)
	}	
	if(chroma_weight_l1_flag[i])	
	for(j = 0; j < 2; j++) {	
	delta_chroma_weight_l1[i][j]	se(v)
	delta_chroma_offset_l1[i][j]	se(v)
	}	
	}	
	}	

[0868]

[0869] num_l0_weighted_ref_pics表示参考图像列表0中进行加权的参考图像的数量。num_l0_weighted_ref_pics的取值范围应为0至MaxDecPicBuffMinus1+14(包括端值)。

[0870] 码流一致性的要求为,当num_l0_weighted_ref_pics存在时,num_l0_weighted_ref_pics的值不应小于与图像头关联的图像中的任何条带的L0的活动参考图像的数量。

[0871] num_l1_weighted_ref_pics表示参考图像列表1中进行加权的参考图像的数量。num_l1_weighted_ref_pics的取值范围应为0至MaxDecPicBuffMinus1+14(包括端值)。

[0872] 码流一致性的要求为,当num_l1_weighted_ref_pics存在时,num_l1_weighted_ref_pics的值不应小于与图像头关联的图像中的任何条带的L1的活动参考图像的数量。

[0873]

[0874] 当num_l0_weighted_ref_pics或num_l1_weighted_ref_pics为非0时,MaxNumGeoMergeCand被设置为0。以下语法是如何利用这种依赖性的示例:

[0875]	if(sps_prof_pic_present_flag)	
	pic_disable_prof_flag	u(1)

[0876]	if(sps_geo_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_geo_cand_plus1 && num_l0_weighted_ref_pics==0 && num_l1_weighted_ref_pics==0)	
	pic_max_num_merge_cand_minus_max_num_geo_cand	ue(v)
	if(sps_ibc_enabled_flag)	
	pic_six_minus_max_num_ibc_merge_cand	ue(v)
	if(sps_joint_cbr_enabled_flag)	

[0877] 本实施例中的pic_max_num_merge_cand_minus_max_num_geo_cand的语义与上述实施例相同。

[0878] 在一个示例中,如果在与PH关联的图像中仅使用某些条带类型,则在一定条件下指示帧间和帧内相关语法元素。

[0879] 下文给出了该示例的语法。

[0880]	picture_header_rbsp() {	描述符
	pic_inter_slice_present_flag	u(1)
	if(pic_inter_slice_present_flag)	
	pic_intra_slice_present_flag	u(1)
	non_reference_picture_flag	u(1)
	gdr_pic_flag	u(1)
	no_output_of_prior_pics_flag	u(1)
	if(gdr_pic_flag)	
	recovery_poc_cnt	ue(v)
	ph_pic_parameter_set_id	ue(v)
	if(sps_poc_msb_flag) {	
	ph_poc_msb_present_flag	u(1)
	if(ph_poc_msb_present_flag)	
	poc_msb_val	u(v)
	}	
	if(sps_subpic_id_present_flag && !sps_subpic_id_signalling_flag) {	
	ph_subpic_id_signalling_present_flag	u(1)
	if(ph_subpics_id_signalling_present_flag) {	

[0881]

ph_subpic_id_len_minus1	ue(v)
for(i = 0; i <= sps_num_subpics_minus1; i++)	
ph_subpic_id[i]	u(v)
}	
}	
if(!sps_virtual_boundaries_present_flag) {	
ph_virtual_boundaries_present_flag	u(1)
if(ph_virtual_boundaries_present_flag) {	
ph_num_ver_virtual_boundaries	u(2)
for(i = 0; i < ph_num_ver_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_x[i]	u(13)
ph_num_hor_virtual_boundaries	u(2)
for(i = 0; i < ph_num_hor_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_y[i]	u(13)
}	
}	
if(separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
if(output_flag_present_flag)	
pic_output_flag	u(1)
pic_rpl_present_flag	u(1)
if(pic_rpl_present_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	

[0882]

pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)
pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
if(partition_constraints_override_enabled_flag) {	
partition_constraints_override_flag	u(1)
if(pic_intra_slice_present_flag) {	
if(partition_constraints_override_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
if(qtbtt_dual_tree_intra_flag) {	
pic_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
pic_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(pic_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
pic_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
pic_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)

[0883]

}	
}	
}	
→}	
if(cu_qp_delta_enabled_flag) {	
pic_cu_qp_delta_subdiv_intra_slice	ue(v)
if(pps_cu_chroma_qp_offset_list_enabled_flag)	
pic_cu_chroma_qp_offset_subdiv_intra_slice	ue(v)
}	
if(pic_inter_slice_present_flag) {	
if(partition_constraints_override_flag) {	
pic_log2_diff_min_qt_min_cb_inter_slice	ue(v)
pic_max_mtt_hierarchy_depth_inter_slice	ue(v)
if(pic_max_mtt_hierarchy_depth_inter_slice != 0) {	
pic_log2_diff_max_bt_min_qt_inter_slice	ue(v)
pic_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
}	
if(cu_qp_delta_enabled_flag)	
pic_cu_qp_delta_subdiv_inter_slice	ue(v)
if(pps_cu_chroma_qp_offset_list_enabled_flag) {	
pic_cu_chroma_qp_offset_subdiv_inter_slice	ue(v)
if(sps_temporal_mvp_enabled_flag)	
pic_temporal_mvp_enabled_flag	u(1)
if(!pps_mvd_l1_zero_idc)	
mvd_l1_zero_flag	u(1)
if(!pps_six_minus_max_num_merge_cand_plus1)	
pic_six_minus_max_num_merge_cand	ue(v)

[0884]

if(sps_affine_enabled_flag)	
pic_five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
pic_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_pic_present_flag)	
pic_disable_bdof_flag	u(1)
if(sps_dmv_r_pic_present_flag)	
pic_disable_dmv_r_flag	u(1)
if(sps_prof_pic_present_flag)	
pic_disable_prof_flag	u(1)
if(sps_triangle_enabled_flag && MaxNumMergeCand >= 2 && !pps_max_num_merge_cand_minus_max_num_triangle_cand_plus1)	
pic_max_num_merge_cand_minus_max_num_triangle_cand	ue(v)
}	
if(sps_ibc_enabled_flag)	
pic_six_minus_max_num_ibc_merge_cand	ue(v)
...	

[0885] 7.3.7.1通用条带头语法

[0886]

slice_header() {	描述符
slice_pic_order_cnt_lsb	u(v)
if(subpics_present_flag)	
slice_subpic_id	u(v)
if(rect_slice_flag NumTilesInPic > 1)	
slice_address	u(v)
if(!rect_slice_flag && NumTilesInPic > 1)	
num_tiles_in_slice_minus1	ue(v)
if(pic_inter_slice_present_flag)	
slice_type	ue(v)
...	

[0887] 7.4.3.6图像头RBSP语义

[0888] pic_inter_slice_present_flag等于1表示slice_type等于0(B)或1(P)的一个或多个条带可以存在于与PH关联的图像中。pic_inter_slice_present_flag等于0表示

slice_type等于0(B)或1(P)的条带可以不存在于与PH关联的图像中。

[0889] pic_intra_slice_present_flag等于1表示slice_type等于2(I)的一个或多个条带可以存在于与PH关联的图像中。pic_intra_slice_present_flag等于0表示slice_type等于2(I)的条带可以不存在于与PH关联的图像中。当pic_intra_slice_only_flag不存在时,pic_intra_slice_only_flag的值被推断为1。

[0890] 注:在与包括一个或多个子图像的图像关联的图像头中,pic_inter_slice_present_flag和pic_intra_slice_present_flag的值都被设置为1,其中,一个或多个子图像包括一个或多个帧内译码条带,并可以与包括一个或多个帧间译码条带的一个或多个子图像融合。

[0891] 7.4.8.1通用条带头语法

[0892] slice_type指定条带的编码类型,如表7-5所示。

[0893] 表7-5:名称与slice_type的关联关系

[0894]

slice_type	slice_type的名称
0	B(B条带)
1	P(P条带)
2	I(I条带)

[0895] 当nal_unit_type的取值范围为IDR_W_RADL至CRA_NUT(包括端值),且当前图像是接入单元中的第一个图像时,slice_type应等于2。

[0896] 当slice_type不存在时,slice_type的值被推断为2。

[0897] 当pic_intra_slice_present_flag等于0时,slice_type的取值范围应为0至1(包括端值)。

[0898] 该示例可以与pred_weight_table()在图像头中的指示组合。上述示例公开了pred_weight_table()在图像头中指示。

[0899] 一种示例性语法如下所述:

[0900]

picture_header_rbsp() {	描述符
...	
...	
if((pps_weighted_pred_flag pps_weighted_bipred_flag) && weighted_pred_table_present_in_ph_flag)	
pred_weight_table()	
...	

[0901] 当指示pred_weight_table()存在于图像头中时,可以使用以下语法。

[0902]

picture_header_rbsp() {	描述符
-------------------------	-----

[0903]

...	
pic_inter_slice_present_flag	u(1)
if(pic_inter_slice_present_flag)	
pic_intra_slice_present_flag	u(1)
...	
if((pps_weighted_pred_flag pps_weighted_bipred_flag) && pic_inter_slice_present_flag)	
pred_weight_table()	
...	

[0904] 可选示例可以使用以下语法：

[0905]

picture_header_rbsp() {	描述符
...	
pic_inter_slice_present_flag	u(1)
if(pic_inter_slice_present_flag)	
pic_intra_slice_present_flag	u(1)
...	
if((pps_weighted_pred_flag pps_weighted_bipred_flag) && pic_inter_slice_present_flag && weighted_pred_table_present_in_ph_flag)	
pred_weight_table()	
...	

[0906] 可选示例可以使用以下语法：

[0907]

picture_header_rbsp() {	描述符
...	
pic_inter_bipred_slice_present_flag	u(1)
if(!pic_inter_bipred_slice_present_flag)	
pic_inter_slice_present_flag	u(1)
if(pic_inter_slice_present_flag)	
pic_intra_slice_present_flag	u(1)
...	

[0908]

if((pps_weighted_pred_flag pps_weighted_bipred_flag) && pic_inter_slice_present_flag && weighted_pred_table_present_in_ph_flag)	
pred_weight_table()	
...	

[0909] 在上文的语法中,pic_inter_bipred_slice_present_flag表示存在参考图像头的所有条带类型:I条带、B条带和P条带。

[0910] 当pic_inter_bipred_slice_present_flag为0时,图像只包括I类型条带或B类型条带。

[0911] 在这种情况下,不使用非矩形模式。

[0912] 在一个示例中,公开了上述示例的组合。一种示例性语法如下所述:

[0913]

picture_header_rbsp() {	描述符
...	
pic_inter_slice_present_flag	u(1)
if(pic_inter_slice_present_flag)	
pic_intra_slice_present_flag	u(1)
~~pic_rpl_present_flag~~	~~u(1)~~
~~if(pic_rpl_present_flag) {~~	
if(rpl_present_in_ph_flag) {	
for(i = 0; i < 2; i++) {	
if(num_ref_pic_lists_in_sps[i] > 0 && !pps_ref_pic_list_sps_idc[i] && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_sps_flag[i]	u(1)
if(pic_rpl_sps_flag[i]) {	
if(num_ref_pic_lists_in_sps[i] > 1 && (i == 0 (i == 1 && rpl1_idx_present_flag)))	
pic_rpl_idx[i]	u(v)
} else	
ref_pic_list_struct(i, num_ref_pic_lists_in_sps[i])	
for(j = 0; j < NumLtrpEntries[i][RplIdx[i]]; j++) {	
if(ltrp_in_slice_header_flag[i][RplIdx[i]])	
pic_poc_lsb_lt[i][j]	u(v)

[0914]

pic_delta_poc_msb_present_flag[i][j]	u(1)
if(pic_delta_poc_msb_present_flag[i][j])	
pic_delta_poc_msb_cycle_lt[i][j]	ue(v)
}	
}	
}	
...	
if((pps_weighted_pred_flag pps_weighted_bipred_flag) && weighted_pred_table_present_in_ph_flag && pic_inter_slice_present_flag)	
pred_weight_table()	
...	
if(sps_sao_enabled_flag && sao_present_in_ph_flag) {	
—— pic_sao_enabled_present_flag	u(1)
—— if(pic_sao_enabled_present_flag) {	
pic_sao_luma_enabled_flag	u(1)
if(ChromaArrayType != 0)	
pic_sao_chroma_enabled_flag	u(1)
—— }	
}	
if(sps_alf_enabled_flag && alf_present_in_ph_flag) {	
—— pic_alf_enabled_present_flag	u(1)
—— if(pic_alf_enabled_present_flag) {	
pic_alf_enabled_flag	u(1)
if(pic_alf_enabled_flag) {	
pic_num_alf_aps_ids_luma	u(3)
for(i = 0; i < pic_num_alf_aps_ids_luma; i++)	
pic_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	

[0915]

pic_alf_chroma_idc	u(2)
if(pic_alf_chroma_idc)	
pic_alf_aps_id_chroma	u(3)
}	
—}	
}	
...	
if(deblocking_filter_override_enabled_flag && deblocking_filter_override_present_in_ph_flag) {	
— pic_deblocking_filter_override_present_flag	u(1)
—if(pic_deblocking_filter_override_present_flag) {	
pic_deblocking_filter_override_flag	u(1)
if(pic_deblocking_filter_override_flag) {	
pic_deblocking_filter_disabled_flag	u(1)
if(!pic_deblocking_filter_disabled_flag) {	
pic_beta_offset_div2	se(v)
pic_tc_offset_div2	se(v)
}	
}	
—}	
—}	
...	
}	

[0916] 在一个示例中,在不使用加权预测因子的情况下选择参考图像的非矩形(例如GEO)模式。

[0917] 在该示例中,语义的定义如下:

[0918] 7.4.10.7融合数据语义

[0919]

[0920] 变量MergeGeoFlag[x0][y0]表示在解码B条带时,基于三角形的运动补偿是否用于生成当前编码单元(coding unit)的预测样本。该变量的推导如下:

[0921] -如果以下所有条件都为真,则MergeGeoFlag[x0][y0]被设置为1:

[0922] -sps_geo_enabled_flag等于1,

[0923] -slice_type为B,

[0924] -general_merge_flag[x0][y0]等于1,

- [0925] -MaxNumGeoMergeCand大于或等于2,
- [0926] -cbWidth大于或等于8,
- [0927] -cbHeight大于或等于8,
- [0928] -cbWidth小于 $8 \times \text{cbHeight}$,
- [0929] -cbHeight小于 $8 \times \text{cbWidth}$,
- [0930] -regular_merge_flag[x0][y0]等于0,
- [0931] -merge_subblock_flag[x0][y0]等于0,
- [0932] -ciip_flag[x0][y0]等于0;
- [0933] -否则,MergeGeoFlag[x0][y0]被设置为0。
- [0934] 码流流一致性的要求为:如果CU的亮度显式加权标志或色度显式加权标志中的一个标志为真,则MergeGeoFlag[x0][y0]应等于0。
- [0935] 在一个示例中,VVC规范的一部分解释如下:
- [0936] 8.5.7 GEO帧间块的解码过程
- [0937] 8.5.7.1概述
- [0938] 当对MergeGeoFlag[xCb][yCb]等于1的编码单元(coding unit)进行解码时,调用该过程。
- [0939] 该过程的输入包括:
- [0940] -亮度位置(xCb,yCb),表示当前编码块(coding block)的相对于当前图像的左上亮度样本的左上样本;
- [0941] -变量cbWidth,表示当前编码块的以亮度样本为单位的宽度;
- [0942] -变量cbHeight,表示当前编码块的以亮度样本为单位的高度;
- [0943] -1/16分数样本精度的亮度运动矢量mvA和mvB;
- [0944] -色度运动矢量mvCA和mvCB;
- [0945] -参考索引refIdxA和refIdxB;
- [0946] -预测列表标志predListFlagA和predListFlagB。
- [0947]
- [0948] 假设predSamplesLA_L和predSamplesLB_L为预测亮度样本值组成的 $(\text{cbWidth}) \times (\text{cbHeight})$ 阵列,predSamplesLA_{cb}、predSamplesLB_{cb}、predSamplesLA_{cr}和predSamplesLB_{cr}为预测色度样本值组成的 $(\text{cbWidth}/\text{SubWidthC}) \times (\text{cbHeight}/\text{SubHeightC})$ 阵列。
- [0949] predSamples_L、predSamples_{cb}和predSamples_{cr}通过以下按顺序执行的步骤推导:
- [0950] 1.当N是A和B中的每一个时,以下内容适用:
- [0951]
- [0952] 2.融合geo模式的划分角度和距离变量angleIdx和distanceIdx的根据表36中详述的merge_geo_partition_idx[xCb][yCb]的值设置。
- [0953] 3.变量explicitWeightedFlag的推导如下:
- [0954] lumaWeightedFlagA=predListFlagA?luma_weight_11_flag[refIdxA]:luma_weight_10_flag[refIdxA]
- [0955] lumaWeightedFlagB=predListFlagB?luma_weight_11_flag[refIdxB]:luma_weight_10_flag[refIdxB]

[0956] $\text{chromaWeightedFlagA} = \text{predListFlagA} ? \text{chroma_weight_l1_flag}[\text{refIdxA}] : \text{chroma_weight_l0_flag}[\text{refIdxA}]$

[0957] $\text{chromaWeightedFlagB} = \text{predListFlagB} ? \text{chroma_weight_l1_flag}[\text{refIdxB}] : \text{chroma_weight_l0_flag}[\text{refIdxB}]$

[0958] $\text{weightedFlag} = \text{lumaWeightedFlagA} \mid \mid \text{lumaWeightedFlagB} \mid \mid \text{chromaWeightedFlagA} \mid \mid \text{chromaWeightedFlagB}$

[0959] 4. 当前亮度编码块内的预测样本 $\text{predSamples}_L[x_L][y_L]$ (其中, $x_L = 0 \dots \text{cbWidth} - 1$, $y_L = 0 \dots \text{cbHeight} - 1$) 是通过以下方式推导的: 如果 weightedFlag 等于 0, 则调用第 8.5.7.2 中详述的 geo 融合模式的加权样本预测过程; 如果 weightedFlag 等于 1, 则调用第 8.5.6.6.3 节详述的显式加权样本预测过程, 其中, 输入包括设置为 cbWidth 的编码块宽度 nCbW 、设置为 cbHeight 的编码块高度 nCbH 、样本阵列 predSamplesLA_L 和 predSamplesLB_L 、变量 angleIdx 和 distanceIdx 和设置为 0 的 cIdx 。

[0960] 5. 当前色度分量 Cb 编码块内的预测样本 $\text{predSamples}_{cb}[x_c][y_c]$ (其中, $x_c = 0 \dots \text{cbWidth}/\text{SubWidthC} - 1$, $y_c = 0 \dots \text{cbHeight}/\text{SubHeightC} - 1$) 通过以下方式推导: 如果 weightedFlag 等于 0, 则调用第 8.5.7.2 中详述的 geo 融合模式的加权样本预测过程; 如果 weightedFlag 等于 1, 则调用第 8.5.6.6.3 节详述的显式加权样本预测过程, 其中, 输入包括设置为 $\text{cbWidth}/\text{SubWidthC}$ 的编码块宽度 nCbW 、设置为 $\text{cbHeight}/\text{SubHeightC}$ 的编码块高度 nCbH 、样本阵列 $\text{predSamplesLA}_{cb}$ 和 $\text{predSamplesLB}_{cb}$ 、变量 angleIdx 和 distanceIdx 和设置为 1 的 cIdx 。

[0961] 6. 当前色度分量 Cr 编码块内的预测样本 $\text{predSamples}_{cr}[x_c][y_c]$ (其中, $x_c = 0 \dots \text{cbWidth}/\text{SubWidthC} - 1$, $y_c = 0 \dots \text{cbHeight}/\text{SubHeightC} - 1$) 通过以下方式推导: 如果 weightedFlag 等于 0, 则调用第 8.5.7.2 中详述的 geo 融合模式的加权样本预测过程; 如果 weightedFlag 等于 1, 则调用第 8.5.6.6.3 节详述的显式加权样本预测过程, 其中, 输入包括设置为 $\text{cbWidth}/\text{SubWidthC}$ 的编码块宽度 nCbW 、设置为 $\text{cbHeight}/\text{SubHeightC}$ 的编码块高度 nCbH 、样本阵列 $\text{predSamplesLA}_{cr}$ 和 $\text{predSamplesLB}_{cr}$ 、变量 angleIdx 和 distanceIdx 和设置为 2 的 cIdx 。

[0962] 7. 调用第 8.5.7.3 节详述的融合 geo 模式的运动矢量存储过程, 其中, 输入包括亮度编码块位置 (x_{Cb}, y_{Cb})、亮度编码块宽度 cbWidth 、亮度编码块高度 cbHeight 、划分方向 angleIdx 和 distanceIdx 、亮度运动矢量 mvA 和 mvB 、参考索引 refIdxA 和 refIdxB 以及预测列表标志 predListFlagA 和 predListFlagB 。

[0963] 表 36: 基于 $\text{merge_geo_partition_idx}$ 值的 angleIdx 和 distanceIdx 值的详述

[0964]	$\text{merge_geo_partition_idx}$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	angleIdx	0	0	1	1	1	1	2	2	2	2	3	3	3	3	4	4	4
	distanceIdx	1	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2
	$\text{merge_geo_partition_idx}$	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33

[0965]

idx																	
angleIdx	4	6	6	8	8	8	8	9	9	9	9	10	10	10	10	11	11
distanceIdx	3	1	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1
merge_geo_partition_idx	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
angleIdx	11	11	12	12	13	13	13	14	14	14	15	15	15	16	16	16	18
distanceIdx	2	3	1	3	1	2	3	1	2	3	1	2	3	1	2	3	1
merge_geo_partition_idx	51	52	53	54	55	56	57	58	59	60	61	62	63				
angleIdx	18	20	20	20	21	21	21	22	22	22	23	23	23				
distanceIdx	3	1	2	3	1	2	3	1	2	3	1	2	3				

[0966] 8.5.6.6.3显式加权样本预测过程

[0967] 该过程的输入包括：

[0968] -两个变量nCbW和nCbH,表示当前编码块(coding block)的宽度和高度；

[0969] -两个(nCbW) × (nCbH)阵列predSamplesL0和predSamplesL1；

[0970] -预测列表使用标志predFlagL0和predFlagL1；

[0971] -参考索引refIdxL0和refIdxL1；

[0972] -变量cIdx,表示颜色分量索引；

[0973] -样本位深度bitDepth。

[0974] 该过程的输出为预测样本值组成的(nCbW) × (nCbH)阵列pbSamples。

[0975] 变量shift1被设置为Max(2, 14-bitDepth)。

[0976] 变量log2Wd、o0、o1、w0和w1的推导如下：

[0977] -如果亮度样本的cIdx等于0,则以下内容适用：

[0978] $\log2Wd = \text{luma_log2_weight_denom} + \text{shift1}$ (1010)[0979] $w0 = \text{LumaWeightL0}[\text{refIdxL0}]$ (1011)[0980] $w1 = \text{LumaWeightL1}[\text{refIdxL1}]$ (1012)[0981] $o0 = \text{luma_offset_l0}[\text{refIdxL0}] \ll (\text{bitDepth} - 8)$ (1013)[0982] $o1 = \text{luma_offset_l1}[\text{refIdxL1}] \ll (\text{bitDepth} - 8)$ (1014)

[0983] -否则(色度样本的cIdx不等于0),以下内容适用：

[0984] $\log2Wd = \text{ChromaLog2WeightDenom} + \text{shift1}$ (1015)[0985] $w0 = \text{ChromaWeightL0}[\text{refIdxL0}][\text{cIdx} - 1]$ (1016)[0986] $w1 = \text{ChromaWeightL1}[\text{refIdxL1}][\text{cIdx} - 1]$ (1017)[0987] $o0 = \text{ChromaOffsetL0}[\text{refIdxL0}][\text{cIdx} - 1] \ll (\text{bitDepth} - 8)$ (1018)[0988] $o1 = \text{ChromaOffsetL1}[\text{refIdxL1}][\text{cIdx} - 1] \ll (\text{bitDepth} - 8)$ (1019)

[0989] 预测样本pbSamples[x][y](其中,x=0..nCbW-1,y=0..nCbH-1)的推导如下：

[0990] -如果predFlagL0等于1,predFlagL1等于0,则预测样本值的推导如下：

if(log2Wd >= 1)
 pbSamples[x][y] = Clip3(0, (1 << bitDepth) - 1,
 ((predSamplesL0[x][y] * w0 + 2^{log2Wd-1}) >> log2Wd) + o0) (1020)
 [0991] else
 pbSamples[x][y] = Clip3(0, (1 << bitDepth) - 1, predSamplesL0[x][y] * w0
 + o0)

[0992] -如果predFlagL0等于0,predFlagL1等于1,则预测样本值的推导如下:

if(log2Wd >= 1)
 pbSamples[x][y] = Clip3(0, (1 << bitDepth) - 1,
 ((predSamplesL1[x][y] * w1 + 2^{log2Wd-1}) >> log2Wd) + o1) (1021)
 [0993] else
 pbSamples[x][y] = Clip3(0, (1 << bitDepth) - 1, predSamplesL1[x][y] * w1
 + o1)

[0994] -否则(predFlagL0等于1,predFlagL1等于1),预测样本值的推导如下:

[0995] pbSamples[x][y] = Clip3(0, (1 << bitDepth) - 1, (predSamplesL0[x][y] * w0 +
 predSamplesL1[x][y] * w1 + ((o0 + o1 + 1) << log2Wd) >> (log2Wd + 1)) (1022)

[0996] 在该示例中,公开了融合数据参数的语法,该语法包括检查变量表示存在非矩形融合模式(例如GEO模式)。下文给出了语法示例。

	merge_data(x0, y0, cbWidth, cbHeight, chType) {	描述符
	if(CuPredMode[chType][x0][y0] == MODE_IBC) {	
	if(MaxNumIbcMergeCand > 1)	
	merge_idx [x0][y0]	ae(v)
	} else {	
	if(MaxNumSubblockMergeCand > 0 && cbWidth >= 8 && cbHeight >= 8)	
	merge_subblock_flag [x0][y0]	ae(v)
	if(merge_subblock_flag[x0][y0] == 1) {	
[0997]	if(MaxNumSubblockMergeCand > 1)	
	merge_subblock_idx [x0][y0]	ae(v)
	} else {	
	if(cbWidth < 128 && cbHeight < 128 && ((sps_ciip_enabled_flag && cu_skip_flag[x0][y0] == 0 && (cbWidth * cbHeight) >= 64) (sps_geo_enabled_flag && MaxNumGeoMergeCand > 1 && cbWidth >= 8 && cbHeight >= 8 && cbWidth < 8 * cbHeight && cbHeight < 8 * cbWidth && slice_type == B)))	
	regular_merge_flag [x0][y0]	ae(v)
	if(regular_merge_flag[x0][y0] == 1) {	

[0998]

if(sps_mmvd_enabled_flag)	
mmvd_merge_flag [x0][y0]	ac(v)
if(mmvd_merge_flag[x0][y0] == 1) {	
if(MaxNumMergeCand > 1)	
mmvd_cand_flag [x0][y0]	ac(v)
mmvd_distance_idx [x0][y0]	ac(v)
mmvd_direction_idx [x0][y0]	ac(v)
} else if(MaxNumMergeCand > 1)	
merge_idx [x0][y0]	ac(v)
} else {	
if(sps_ciip_enabled_flag && sps_geo_enabled_flag && MaxNumGeoMergeCand > 1 && slice_type == B && cu_skip_flag[x0][y0] == 0 && cbWidth >= 8 && cbHeight >= 8 && cbWidth < 8 * cbHeight && cbHeight < 8 * cbWidth && cbWidth < 128 && cbHeight < 128)	
ciip_flag [x0][y0]	ac(v)
if(ciip_flag[x0][y0] && MaxNumMergeCand > 1)	
merge_idx [x0][y0]	ac(v)
if(!ciip_flag[x0][y0] && MaxNumGeoMergeCand > 1) {	
merge_geo_partition_idx [x0][y0]	ac(v)
merge_geo_idx0 [x0][y0]	ac(v)
if(MaxNumGeoMergeCand > 2)	
merge_geo_idx1 [x0][y0]	ac(v)
}	
}	
}	
}	
}	

[0999] 变量MaxNumGeoMergeCand是根据任一上述示例推导的。

[1000] 可以使用根据变量MaxNumGeoMergeCand推导出的可选变量SliceMaxNumGeoMergeCand。MaxNumGeoMergeCand的值是在较高的指示级(例如如,PH、PPS或SPS)获得的。

[1001] 在一个示例中,SliceMaxNumGeoMergeCand是根据MaxNumGeoMergeCand的值和对条带执行的其它检查推导的。

[1002] 例如,SliceMaxNumGeoMergeCand=(num_l0_weighted_ref_pics>0||num_l1_weighted_ref_pics>0)?0:MaxNumGeoMergeCand。

[1003] 在另一个示例中,以下表达式用于确定MaxNumGeoMergeCand值:

[1004] SliceMaxNumGeoMergeCand=(!pic_inter_slice_present_flag)?0:

MaxNumGeoMergeCand。

[1005] 在一个示例中，

[1006] 定义了以下语法表：

[1007]	picture_header_rbsp() {	描述符
	...	
	pic_inter_bipred_slice_present_flag	
	if (!pic_inter_bipred_slice_present_flag)	
	pic_inter_slice_present_flag	
	if(pic_inter_slice_present_flag)	
	pic_intra_slice_present_flag	u(1)
	...	
	if((pps_weighted_pred_flag pps_weighted_bipred_flag) && pic_inter_slice_present_flag && weighted_pred_table_present_in_ph_flag)	
	pred_weight_table()	
	...	

[1008] 变量MaxNumGeoMergeCand的推导如下：

[1009] SliceMaxNumGeoMergeCand= (!pic_inter_bipred_slice_present_flag)?0:MaxNumGeoMergeCand。

[1010] 本发明公开了一种指示矩形模式和非矩形模式的融合候选数量的方法。矩形模式和非矩形模式的融合候选数量是相关的，如果指示矩形模式的融合候选数量小于阈值时，可能不需要指示非矩形模式的融合候选数量。

[1011] 尤其是对于TPM融合模式或Geo融合模式，由于使用任一非矩形融合模式预测的块需要具有不同指定的MV的两个帧间预测值，因此融合模式应该存在至少两个候选。在一个实施例中，当融合模式候选数量在序列参数集(sequence parameter set, SPS)中指示时，可以使用以下语法：

[1012] 7.3.2.3序列参数集RBSP语法

[1013]	seq_parameter_set_rbsp() {	描述符
	sps_decoding_parameter_set_id	u(4)
	sps_video_parameter_set_id	u(4)
	sps_max_sublayers_minus1	u(3)
	sps_reserved_zero_4bits	u(4)

[1014]

sps_ptl_dpb_hrd_params_present_flag	u(1)
if(sps_ptl_dpb_hrd_params_present_flag)	
profile_tier_level(1, sps_max_sublayers_minus1)	
gdr_enabled_flag	u(1)
sps_seq_parameter_set_id	u(4)
chroma_format_idc	u(2)
if(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
ref_pic_resampling_enabled_flag	u(1)
pic_width_max_in_luma_samples	ue(v)
pic_height_max_in_luma_samples	ue(v)
sps_log2_ctu_size_minus5	u(2)
subpics_present_flag	u(1)
if(subpics_present_flag) {	
sps_num_subpics_minus1	u(8)
for(i = 0; i <= sps_num_subpics_minus1; i++) {	
subpic_ctu_top_left_x[i]	u(v)
subpic_ctu_top_left_y[i]	u(v)
subpic_width_minus1[i]	u(v)
subpic_height_minus1[i]	u(v)
subpic_treated_as_pic_flag[i]	u(1)
loop_filter_across_subpic_enabled_flag[i]	u(1)
}	
}	
sps_subpic_id_present_flag	u(1)
if(sps_subpic_id_present_flag) {	
sps_subpic_id_signalling_present_flag	u(1)
if(sps_subpic_id_signalling_present_flag) {	

[1015]

sps_subpic_id_len_minus1	ue(v)
for(i = 0; i <= sps_num_subpics_minus1; i++)	
sps_subpic_id[i]	u(v)
}	
}	
bit_depth_minus8	ue(v)
min_qp_prime_ts_minus4	ue(v)
sps_weighted_pred_flag	u(1)
sps_weighted_bipred_flag	u(1)
log2_max_pic_order_cnt_lsb_minus4	u(4)
sps_poc_msb_flag	u(1)
if(sps_poc_msb_flag)	
poc_msb_len_minus1	ue(v)
if(sps_max_sublayers_minus1 > 0)	
sps_sublayer_dpb_params_flag	u(1)
if(sps_ptl_dpb_hrd_params_present_flag)	
dpb_parameters(0, sps_max_sublayers_minus1, sps_sublayer_dpb_params_flag)	
long_term_ref_pics_flag	u(1)
inter_layer_ref_pics_present_flag	u(1)
sps_idr_rpl_present_flag	u(1)
rpl1_same_as_rpl0_flag	u(1)
for(i = 0; i < !rpl1_same_as_rpl0_flag ? 2 : 1; i++) {	
num_ref_pic_lists_in_sps[i]	ue(v)
for(j = 0; j < num_ref_pic_lists_in_sps[i]; j++)	
ref_pic_list_struct(i, j)	
}	
if(ChromaArrayType != 0)	
qtbtt_dual_tree_intra_flag	u(1)

[1016]

log2_min_luma_coding_block_size_minus2	ue(v)
partition_constraints_override_enabled_flag	u(1)
sps_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
sps_log2_diff_min_qt_min_cb_inter_slice	ue(v)
sps_max_mtt_hierarchy_depth_inter_slice	ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(sps_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
sps_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)
}	
if(sps_max_mtt_hierarchy_depth_inter_slice != 0) {	
sps_log2_diff_max_bt_min_qt_inter_slice	ue(v)
sps_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
if(qtbtt_dual_tree_intra_flag) {	
sps_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(sps_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
sps_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
sps_max_luma_transform_size_64_flag	u(1)
sps_joint_cbr_enabled_flag	u(1)
if(ChromaArrayType != 0) {	
same_qp_table_for_chroma	u(1)
numQpTables = same_qp_table_for_chroma ? 1 : (sps_joint_cbr_enabled_flag ? 3 : 2)	
for(i = 0; i < numQpTables; i++) {	

[1017]

qp_table_start_minus26[i]	se(v)
num_points_in_qp_table_minus1[i]	ue(v)
for(j = 0; j <= num_points_in_qp_table_minus1[i]; j++) {	
delta_qp_in_val_minus1[i][j]	ue(v)
delta_qp_diff_val[i][j]	ue(v)
}	
}	
}	
sps_sao_enabled_flag	u(1)
sps_alf_enabled_flag	u(1)
sps_transform_skip_enabled_flag	u(1)
if(sps_transform_skip_enabled_flag)	
sps_bdpcm_enabled_flag	u(1)
if(sps_bdpcm_enabled_flag && chroma_format_idc == 3)	
sps_bdpcm_chroma_enabled_flag	u(1)
sps_ref_wraparound_enabled_flag	u(1)
if(sps_ref_wraparound_enabled_flag)	
sps_ref_wraparound_offset_minus1	ue(v)
sps_temporal_mvp_enabled_flag	u(1)
if(sps_temporal_mvp_enabled_flag)	
sps_sbtmvp_enabled_flag	u(1)
sps_amvr_enabled_flag	u(1)
sps_bdof_enabled_flag	u(1)
if(sps_bdof_enabled_flag)	
sps_bdof_pic_present_flag	u(1)
sps_smvd_enabled_flag	u(1)
sps_dmvr_enabled_flag	u(1)
if(sps_dmvr_enabled_flag)	

[1018]

sps_dmv_r_pic_present_flag	u(1)
sps_mmvd_enabled_flag	u(1)
sps_isp_enabled_flag	u(1)
sps_mrl_enabled_flag	u(1)
sps_mip_enabled_flag	u(1)
if(ChromaArrayType != 0)	
sps_cclm_enabled_flag	u(1)
if(chroma_format_idc == 1) {	
sps_chroma_horizontal_collocated_flag	u(1)
sps_chroma_vertical_collocated_flag	u(1)
}	
sps_mts_enabled_flag	u(1)
if(sps_mts_enabled_flag) {	
sps_explicit_mts_intra_enabled_flag	u(1)
sps_explicit_mts_inter_enabled_flag	u(1)
}	
sps_six_minus_max_num_merge_cand	ue(v)
sps_sbt_enabled_flag	u(1)
sps_affine_enabled_flag	u(1)
if(sps_affine_enabled_flag) {	
sps_five_minus_max_num_subblock_merge_cand	ue(v)
sps_affine_type_flag	u(1)
sps_affine_amvr_enabled_flag	u(1)
sps_affine_prof_enabled_flag	u(1)
if(sps_affine_prof_enabled_flag)	
sps_prof_pic_present_flag	u(1)
}	
if(chroma_format_idc == 3) {	

[1019]

sps_act_enabled_flag	u(1)
sps_palette_enabled_flag	u(1)
}	
sps_bcw_enabled_flag	u(1)
sps_ibc_enabled_flag	u(1)
if (sps_ibc_enabled_flag)	
sps_six_minus_max_num_ibc_merge_cand	ue(v)
sps_ciip_enabled_flag	u(1)
if(sps_mmvd_enabled_flag)	
sps_fpel_mmvd_enabled_flag	u(1)
sps_geo_enabled_flag	u(1)
if (sps_geo_enabled_flag && MaxNumMergeCand >= 3)	
sps_max_num_merge_cand_minus_max_num_geo_cand	ue(v)
sps_lmcs_enabled_flag	u(1)
sps_lfnst_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_scaling_list_enabled_flag	u(1)
sps_virtual_boundaries_present_flag	u(1)
if(sps_virtual_boundaries_present_flag) {	
sps_num_ver_virtual_boundaries	u(2)

[1020]	for(i = 0; i < sps_num_ver_virtual_boundaries; i++)	
	sps_virtual_boundaries_pos_x[i]	u(13)
	sps_num_hor_virtual_boundaries	u(2)
	for(i = 0; i < sps_num_hor_virtual_boundaries; i++)	
	sps_virtual_boundaries_pos_y[i]	u(13)
	}	
	if(sps_ptl_dpb_hrd_params_present_flag) {	
	sps_general_hrd_params_present_flag	u(1)
	if(sps_general_hrd_params_present_flag) {	
	general_hrd_parameters()	
	if(sps_max_sublayers_minus1 > 0)	
	sps_sublayer_cpb_params_present_flag	u(1)
	firstSubLayer = sps_sublayer_cpb_params_present_flag ? 0 :	
	sps_max_sublayers_minus1	
	ols_hrd_parameters(firstSubLayer, sps_max_sublayers_minus1)	
	}	
	}	
	field_seq_flag	u(1)
	vui_parameters_present_flag	u(1)
	if(vui_parameters_present_flag)	
	vui_parameters() /* Specified in ITU-T H.SEI ISO/IEC 23002-7 */	
	sps_extension_flag	u(1)
	if(sps_extension_flag)	
	while(more_rbsp_data())	
	sps_extension_data_flag	u(1)
	rbp_trailing_bits()	
	}	

[1021] 根据本发明的一个实施例,执行以下步骤以在SPS中指示融合模式候选数量:

[1022] -指示常规模式的融合模式候选数量 (MaxNumMergeCand) ;

[1023] -指示非矩形模式否是通过非矩形融合启用标志 (sps_geo_enabled_flg) 启用;

[1024] -当非矩形融合启用标志值为非0且常规融合模式的融合模式候选数量大于第一阈值时,指示非矩形模式数量 (sps_max_num_merge_cand_minus_max_num_geo_cand) 。

[1025] 当常规模式的融合模式候选数量大于第二阈值 (例如1) 时,指示非矩形融合启用

标志。

[1026] 在实施例1中,步骤的顺序如VVC规范的SPS语法的以下部分所示:

[1027]	...	
	if (MaxNumMergeCand > 1)	
	sps_geo_enabled_flag	u(1)
	if (sps_geo_enabled_flag && MaxNumMergeCand >= 3)	
	sps_max_num_merge_cand_minus_max_num_geo_cand	ue(v)
	...	

[1028] 在本实施例中,执行两次顺序检查,第二次检查取决于标志的值,该标志根据第一次检查的结果指示或不指示。

[1029] 与实施例1描述的过程相比,实施例2以不同的方式执行第二次检查。具体地,实施例1使用条件“大于”而不是“大于或等于”。步骤的顺序如VVC规范的SPS语法的以下部分所示:

[1030]	...	
	if (MaxNumMergeCand > 1)	
	sps_geo_enabled_flag	u(1)
	if (sps_geo_enabled_flag && MaxNumMergeCand > 2)	
	sps_max_num_merge_cand_minus_max_num_geo_cand	ue(v)
	...	

[1031] 实施例3与实施例1的不同之处在于,当第一次检查的结果为假值时,不执行第二次检查,由于sps_geo_enabled_flag的值不对MaxNumMergeCand的一些值引用,因此非矩形融合启用标志值(sps_geo_enabled_flag)在根据语法元素sps_six_minus_max_num_merge_cand推导MaxNumMergeCand值的过程完成之后确定,这是一个技术优势,因此跳过解析过程的处理。根据实施例3执行的步骤的顺序如VVC规范的SPS语法的以下部分所示:

[1032]	...	
	if (MaxNumMergeCand > 1) {	
	sps_geo_enabled_flag	u(1)
	if (sps_geo_enabled_flag && MaxNumMergeCand >= 3)	
	sps_max_num_merge_cand_minus_max_num_geo_cand	ue(v)
[1033]	}	
	...	

[1034] 实施例4是实施例2和实施例3的各个方面的组合。根据实施例4执行的步骤的顺序如VVC规范的SPS语法的以下部分所示:

[1035]	...	
	if (MaxNumMergeCand > 1) {	
	sps_geo_enabled_flag	u(1)
	if (sps_geo_enabled_flag && MaxNumMergeCand > 2)	
	sps_max_num_merge_cand_minus_max_num_geo_cand	ue(v)
	}	
	...	

[1036] 实施例5至8公开了第一次检查和第二次检查的不同表述。这些实施例可以解释如下：

[1037] 实施例5

[1038]	...	
	if (MaxNumMergeCand >= 2)	
	sps_geo_enabled_flag	u(1)
	if (sps_geo_enabled_flag && MaxNumMergeCand >= 3)	
	sps_max_num_merge_cand_minus_max_num_geo_cand	ue(v)
	...	

[1039] 实施例6

[1040]	...	
	if (MaxNumMergeCand >= 2)	
	sps_geo_enabled_flag	u(1)
	if (sps_geo_enabled_flag && MaxNumMergeCand > 2)	
	sps_max_num_merge_cand_minus_max_num_geo_cand	ue(v)
	...	

[1041] 实施例7

[1042]	...	
	if (MaxNumMergeCand >= 2) {	
	sps_geo_enabled_flag	u(1)
	if (sps_geo_enabled_flag && MaxNumMergeCand >= 3)	
	sps_max_num_merge_cand_minus_max_num_geo_cand	ue(v)
	}	
	...	

[1043] 实施例8

	...	
	if (MaxNumMergeCand >= 2) {	
	sps_geo_enabled_flag	u(1)
[1044]	if (sps_geo_enabled_flag && MaxNumMergeCand > 2)	
	sps_max_num_merge_cand_minus_max_num_geo_cand	ue(v)
	}	
	...	

[1045] 在图15所示的一种实现方式中,公开了一种获取用于视频解码的几何划分融合模式候选的最大数量的方法。所述方法包括:

[1046] S1501:获取视频序列的码流。

[1047] 所述码流可以根据无线网络或有线网络获取。所述码流可以使用同轴电缆、光纤电缆、双绞线、数字用户线(digital subscriber line,DSL),或红外、无线电、微波、Wi-Fi、蓝牙、LTE或5G等无线技术从网站、服务器或其它远程源发出。

[1048] 在一个实施例中,所述码流是比特序列,所述比特序列以网络抽象层(network abstraction layer,NAL)单元流或字节流为形式,并构成一系列接入单元(access unit,AU)的表示,所述一系列接入单元构成一个或多个编码视频序列(coded video sequence,CVS)。

[1049] 在一些实施例中,对于解码过程,解码端读取码流并从码流中推导经解码图像;对于编码过程,编码端生成码流。

[1050] 通常,码流包括由语法结构构成的语法元素。语法元素:在码流中表示的数据的元素。

[1051] 语法结构:0个或多个语法元素以指定顺序一起出现在码流中。

[1052] 在一个具体示例中,码流格式表示网络抽象层(network abstraction layer,NAL)单元流与字节流之间的关系,网络抽象层单元流或字节流称为码流。

[1053] 码流可以有两种格式:NAL单元流格式和字节流格式。NAL单元流格式在概念上是更“基本的”类型。NAL单元流格式包括称为NAL单元的一系列语法结构。这个序列是按照解码顺序排序的。对NAL单元流中的NAL单元的解码顺序(和内容)使用一些约束条件。

[1054] 按照解码顺序对NAL单元进行排序,并在每个NAL单元前添加开始码前缀和0个或多个零值字节以形成字节流,由此可以根据NAL单元流格式构建字节流格式。在字节流中搜索唯一的开始码前缀模式的位置,由此可以从字节流格式中提取NAL单元流格式。

[1055] 这一节详述通过码流给出的源图像与经解码图像之间的关系。

[1056] 由码流表示的视频源是按照解码顺序排列的图像序列。

[1057] 源图像和经解码图像分别包括一个或多个样本阵列:

[1058] -只有亮度(Y)(单色)样本的阵列;

[1059] -一个亮度样本阵列和两个色度样本阵列(YCbCr或YCgCo);

[1060] -绿蓝红(GBR,也称为RGB)样本阵列;

- [1061] -表示其它未指定的单色或三刺激彩色样本的阵列(例如YZX,也称为XYZ)。
- [1062] 与这些阵列关联的变量和术语称为亮度(或L或Y)和色度,其中,两个色度阵列称为Cb和Cr,与实际使用的颜色表示方法无关。实际使用的颜色表示方法可以在语法中指示,该语法在ITU-T H.SEI|ISO/IEC 23002-7详述的VUI参数中指定。
- [1063] S1502:根据所述码流获取第一指示符的值。
- [1064] 所述第一指示符表示融合运动矢量预测(motion vector prediction,MVP)候选的最大数量。
- [1065] 在一个示例中,所述第一指示符通过变量MaxNumMergeCand表示。
- [1066] 在一个示例中,融合MVP候选的最大数量MaxNumMergeCand推导如下:
- [1067] $\text{MaxNumMergeCand} = 6 - \text{sps_six_minus_max_num_merge_cand}$ 。
- [1068] sps_six_minus_max_num_merge_cand表示从6中减去SPS中支持的融合运动矢量预测(motion vector prediction,MVP)候选的最大数量。sps_six_minus_max_num_merge_cand的取值范围应为0至5(包括端值)。
- [1069] 在一个示例中,sps_six_minus_max_num_merge_cand是从码流中的序列参数集RBSP语法结构中解析出的。
- [1070] S1503:根据所述码流获取第二指示符的值。
- [1071] 所述第二指示符表示是否对所述视频序列启用基于几何划分的运动补偿。
- [1072] 在一个示例中,所述第二指示符通过sps_geo_enabled_flag(sps_gpm_enabled_flag)表示。sps_geo_enabled_flag等于1表示对CLVS启用基于几何划分的运动补偿,且merge_gpm_partition_idx,merge_gpm_idx0和merge_gpm_idx1可以存在于CLVS的编码单元语法中。sps_geo_enabled_flag等于0表示对CLVS禁用基于几何划分的运动补偿,且merge_gpm_partition_idx,merge_gpm_idx0和merge_gpm_idx1不存在于CLVS的编码单元语法中。当sps_geo_enabled_flag不存在时,sps_geo_enabled_flag的值被推断为0。
- [1073] 在一种实现方式中,所述获取第二指示符的值的步骤在所述获取第一指示符的值的步骤之后执行。
- [1074] 在一种实现方式中,所述第二指示符的值是从所述码流的序列参数集(sequence parameter set,SPS)中获取的。
- [1075] 在一种实现方式中,当所述第一指示符的值大于或等于阈值时,所述第二指示符的值是从所述码流的序列参数集(sequence parameter set,SPS)中解析出的。所述阈值为整数,在一个示例中,所述阈值为2。
- [1076] 例如,所述第二指示符sps_gpm_enabled_flag的值是根据
- [1077] 序列参数集RBSP语法获取的。

[1078]	if(MaxNumMergeCand >= 2) {	
	sps_gpm_enabled_flag	u(1)

- [1079] S1504:从所述码流中解析第三指示符的值。
- [1080] 在一种实现方式中,当所述第一指示符的值大于阈值且所述第二指示符的值等于预设值时,从所述码流中解析第三指示符的值,其中,所述第三指示符表示从所述第一指示符的值中减去几何划分融合模式候选的最大数量。

- [1081] 所述阈值为整数值,所述预设值为整数值。在一个示例中,所述阈值为2。
- [1082] 在一个示例中,所述阈值为1。
- [1083] 在一个示例中,所述第三指示符的值是从所述码流的序列参数集(sequence parameter set,SPS)中获取的。
- [1084] 在一个示例中,所述第三指示符通过sps_max_num_merge_cand_minus_max_num_geo_cand(sps_max_num_merge_cand_minus_max_num_gpm_cand)表示。
- [1085] 例如,所述第三指示符sps_max_num_merge_cand_minus_max_num_gpm_cand的值是根据序列参数集RBSP语法获取的。

[1086]	if(MaxNumMergeCand >= 2) {	
	sps_gpm_enabled_flag	u(1)
	if(sps_gpm_enabled_flag && MaxNumMergeCand >= 3)	
	sps_max_num_merge_cand_minus_max_num_gpm_cand	ue(v)
	}	

- [1087] 在一种实现方式中,所述方法还包括:当所述第一指示符的值等于所述阈值且所述第二指示符的值等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为2。
- [1088] 在一种实现方式中,所述方法还包括:当所述第一指示符的值小于所述阈值或所述第二指示符的值不等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为0。
- [1089] 在一个示例中,sps_max_num_merge_cand_minus_max_num_gpm_cand表示从MaxNumMergeCand中减去SPS中支持的划分融合模式候选的最大数量。sps_max_num_merge_cand_minus_max_num_gpm_cand的取值范围应为0至MaxNumMergeCand-2(包括端值)。
- [1090] 几何划分融合模式候选的最大数量MaxNumGpmMergeCand (MaxNumGeoMergeCand)推导如下:

```
if( sps_gpm_enabled_flag && MaxNumMergeCand >= 3 )
    MaxNumGpmMergeCand = MaxNumMergeCand -
[1091]         sps_max_num_merge_cand_minus_max_num_gpm_cand
else if( sps_gpm_enabled_flag && MaxNumMergeCand == 2 )
    MaxNumGpmMergeCand = 2
else
[1092]     MaxNumGpmMergeCand = 0。
```

- [1093] 在图16所示的实现方式中,公开了一种视频解码装置1600。所述视频解码装置包括:接收模块1601,用于获取视频序列的码流;获取模块1602,用于根据所述码流获取第一指示符的值,其中,所述第一指示符表示融合运动矢量预测(motion vector prediction, MVP) 候选的最大数量;获取模块1602用于根据所述码流获取第二指示符的值,其中,所述第二指示符表示是否对所述视频序列启用基于几何划分的运动补偿;解析模块1603,用于:当所述第一指示符的值大于阈值且所述第二指示符的值等于预设值时,从所述码流中解析第

三指示符的值,其中,所述第三指示符表示从所述第一指示符的值中减去所述几何划分融合模式候选的最大数量。

[1094] 在一种实现方式中,获取模块1602用于:当所述第一指标的值等于所述阈值且所述第二指标的值等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为2。

[1095] 在一种实现方式中,获取模块1602用于:当所述第一指示符的值小于所述阈值或所述第二指示符的值不等于所述预设值时,将所述几何划分融合模式候选的最大数量的值设置为0。

[1096] 在一种实现方式中,所述阈值为2。

[1097] 在一种实现方式中,所述阈值为1。

[1098] 在一种实现方式中,所述获取第二指示符的值的步骤在所述获取第一指示符的值的步骤之后执行。

[1099] 在一种实现方式中,当所述第一指示符的值大于或等于所述阈值时,所述第二指示符的值是从所述码流的序列参数集(sequence parameter set,SPS)中解析出的。

[1100] 在一种实现方式中,所述第二指示符的值是从所述码流的序列参数集(sequence parameter set,SPS)中获取的。

[1101] 在一种实现方式中,所述第三指示符的值是从所述码流的序列参数集(sequence parameter set,SPS)中获取的。

[1102] 接收模块1601、获取模块1602和解析模块1603的更多细节可以参考上述方法示例和实现方式。

[1103] 示例1.所述视频编码方法包括指示融合模式候选数量,其中,所述方法包括:

[1104] -指示常规模式的融合模式候选数量(MaxNumMergeCand);

[1105] -指示非矩形模式是否通过非矩形融合启用标志(sps_geo_enabled_flag)启用;

[1106] -当非矩形融合启用标志值为非0且常规融合模式的融合模式候选数量大于第一阈值时,指示非矩形模式数量(sps_max_num_merge_cand_minus_max_num_geo_cand),其中,当常规模式的融合模式候选数量大于第二阈值(1)时,指示所述非矩形融合启用标志。

[1107] 示例2.根据示例1所述的方法,其中,所述非矩形融合启用标志值是在从语法元素sps_six_minus_max_num_merge_cand中推导MaxNumMergeCand值的过程完成之后确定的。

[1108] 示例3.根据任一上述示例所述的方法,其中,所述阈值的检查是常规融合模式的融合模式候选数量是否大于2的比较。

[1109] 示例4.根据示例1或示例2所述的方法,其中,所述第一阈值的检查是常规融合模式的融合模式候选数量是否大于或等于3的比较。

[1110] 在一个示例中,公开了一种帧间预测方法,包括:确定是否允许非矩形帧间预测模式用于块组;获取所述块组的一个或多个帧间预测模式参数和加权预测参数;根据所述一个或多个帧间预测模式参数和加权预测参数,获取当前块的预测值,其中,所述一个或多个帧间预测模式参数中的一个帧间预测模式参数表示所述当前块的参考图像信息,所述块组包括所述当前块。

[1111] 在一个示例中,所述参考图像信息包括是否对参考图像索引启用加权预测;如果加权预测被启用,则所述非矩形帧间预测模式被禁用。

[1112] 在一种可行的实现方式中,如果加权预测被禁用,则所述非矩形帧间预测模式被启用。

[1113] 在一个示例中,所述确定是否允许非矩形帧间预测模式包括:指示三角融合候选的最大数量(MaxNumTriangleMergeCand)大于1。

[1114] 在一个示例中,所述块组包括图像,所述加权预测参数和用于确定是否允许所述非矩形帧间预测模式的指示信息在所述图像的图像头中。

[1115] 在一个示例中,所述块组包括条带,所述加权预测参数和用于确定是否允许所述非矩形帧间预测模式的指示信息在所述条带的条带头中。

[1116] 在一个示例中,所述非矩形帧间预测模式为三角划分模式。

[1117] 在一个示例中,所述非矩形帧间预测模式为地理(GEO)划分模式。

[1118] 在一个示例中,所述加权预测参数用于条带级亮度补偿。

[1119] 在一个示例中,所述加权预测参数用于块级亮度补偿。

[1120] 在一个示例中,所述加权预测参数包括:标志,表示所述加权预测是否应用于预测块的亮度分量和/或色度分量;线性模型参数,指定所述预测块的值的线性变换。

[1121] 在一个示例中,公开了一种用于帧间预测的装置,包括:非瞬时性存储器,存储有处理器可执行指令;处理器,耦合到所述存储器并用于执行所述处理器可执行指令,以实现上述任一方法示例。

[1122] 在一个示例中,公开了一种用于帧间预测的码流,包括:用于确定是否允许非矩形帧间预测模式用于块组的指示信息;所述块组的一个或多个帧间预测模式参数和加权预测参数,其中,当前块的预测值是根据所述一个或多个帧间预测模式参数和加权预测参数获得的,所述一个或多个帧间预测模式参数中的一个帧间预测模式参数表示所述当前块的参考图像信息,所述块组包括所述当前块。

[1123] 在一个示例中,所述参考图像信息包括是否对参考图像索引启用加权预测;如果加权预测被启用,则所述非矩形帧间预测模式被禁用。

[1124] 在一个示例中,如果加权预测被禁用,则所述非矩形帧间预测模式被启用。

[1125] 在一个示例中,所述指示信息包括三角融合候选的最大数量(MaxNumTriangleMergeCand)大于1。

[1126] 在一个示例中,所述块组包括图像,所述加权预测参数和所述指示信息在所述图像的图像头中。

[1127] 在一个示例中,所述块组由条带组成,所述加权预测参数和所述指示信息在所述条带的条带头中。

[1128] 在一个示例中,所述非矩形帧间预测模式为三角划分模式。

[1129] 在一个示例中,所述非矩形帧间预测模式为地理(GEO)划分模式。

[1130] 在一个示例中,所述加权预测参数用于条带级亮度补偿。

[1131] 在一个示例中,所述加权预测参数用于块级亮度补偿。

[1132] 在一个示例中,所述加权预测参数包括:标志,表示所述加权预测是否应用于预测块的亮度分量和/或色度分量;线性模型参数,指定所述预测块的值的线性变换。

[1133] 在一个示例中,公开了一种帧间预测装置,包括:确定模块,确定是否允许非矩形帧间预测模式用于块组;获取模块,用于获取所述块组的一个或多个帧间预测模式参数和

加权预测参数;预测模块,用于根据所述一个或多个帧间预测模式参数和加权预测参数,获取当前块的预测值,其中,所述一个或多个帧间预测模式参数中的一个帧间预测模式参数表示所述当前块的参考图像信息,所述块组包括所述当前块。

[1134] 在一个示例中,所述参考图像信息包括是否对参考图像索引启用加权预测;如果加权预测被启用,则所述非矩形帧间预测模式被禁用。

[1135] 在一个示例中,如果加权预测被禁用,则所述非矩形帧间预测模式被启用。

[1136] 在一个示例中,所述确定模块具体用于:指示三角融合候选的最大数量(MaxNumTriangleMergeCand)大于1。

[1137] 在一个示例中,所述块组包括图像,所述加权预测参数和用于确定是否允许所述非矩形帧间预测模式的指示信息在所述图像的图像头中。

[1138] 在一个示例中,所述块组包括条带,所述加权预测参数和用于确定是否允许所述非矩形帧间预测模式的指示信息在所述条带的条带头中。

[1139] 在一个示例中,所述非矩形帧间预测模式为三角划分模式。

[1140] 在一个示例中,所述非矩形帧间预测模式为地理(GEO)划分模式。

[1141] 在一个示例中,所述加权预测参数用于条带级亮度补偿。

[1142] 在一个示例中,所述加权预测参数用于块级亮度补偿。

[1143] 在一个示例中,所述加权预测参数包括:标志,表示所述加权预测是否应用于预测块的亮度分量和/或色度分量;线性模型参数,指定所述预测块的值的线性变换。

[1144] 在本发明实施例中,可以只使用允许或启用双向帧间预测的条带(例如双向(B)预测条带(也称为B条带))的条带头中的信号相关信息进行高效编码和/或解码。

[1145] 下面对上述实施例中所示的编码方法和解码方法的应用以及使用这些应用的系统进行解释说明。

[1146] 图10为用于实现内容分发服务的内容供应系统3100的框图。内容供应系统3100包括捕获设备3102、终端设备3106,并且可选地包括显示器3126。捕获设备3102通过通信链路3104与终端设备3106通信。该通信链路可以包括上述通信信道13。通信链路3104包括但不限于Wi-Fi、以太网、电缆、无线(3G/4G/5G)、USB或者其任何种类的组合等。

[1147] 捕获设备3102生成数据,并且可以通过上述实施例中所示的编码方法对数据进行编码。可选地,捕获设备3102可以将数据分发给流媒体服务器(图中未示出),该服务器对数据进行编码并将经编码数据发送给终端设备3106。捕获设备3102包括但不限于摄像机、智能手机或平板电脑、计算机或笔记本电脑、视频会议系统、PDA、车载设备,或它们的任意组合等。例如,捕获设备3102可以包括上文所述的源设备12。当数据包括视频时,捕获设备3102中包括的视频编码器20实际上可以执行视频编码处理。当数据包括音频(即声音)时,捕获设备3102中包括的视频编码器实际上可以执行音频编码处理。对于一些实际场景,捕获设备3102通过将经编码视频数据和经编码音频数据一起复用来分发经编码视频数据和经编码音频数据。对于其它实际场景,例如在视频会议系统中,不复用经编码音频数据和经编码视频数据。捕获设备3102将经编码音频数据和经编码视频数据分别分发给终端设备3106。

[1148] 在内容供应系统3100中,终端设备310接收并再生成经编码数据。终端设备3106可以是具有数据接收和恢复能力的设备,例如智能手机或平板电脑3108、计算机或笔记本电

脑3110、网络视频录像机(network video recorder,NVR)/数字视频录像机(digital video recorder,DVR)3112、电视3114、机顶盒(set top box,STB)3116、视频会议系统3118、视频监控系统3120、个人数字助理(personal digital assistant,PDA)3122、车载设备3124,或能够对上述经编码数据进行解码的以上设备的任意组合等。例如,终端设备3106可以包括上文所述的目的地设备14。当经编码数据包括视频时,终端设备中包括的视频解码器30优先执行视频解码。当经编码数据包括音频时,终端设备中包括的音频解码器优先执行音频解码处理。

[1149] 对于具有显示器的终端设备,例如智能手机或平板电脑3108、计算机或笔记本电脑3110、网络视频录像机(network video recorder,NVR)/数字视频录像机(digital video recorder,DVR)3112、电视3114、个人数字助理(personal digital assistant,PDA)3122或车载设备3124,终端设备可以将经解码数据馈送给其显示器。对于不配备显示器的终端设备,例如STB 3116、视频会议系统3118或视频监控系统3120,在其中连接外部显示器3126以接收和显示经解码数据。

[1150] 当该系统中的每个设备执行编码或解码时,可以使用如上述实施例中所示的图像编码设备或图像解码设备。

[1151] 图11是为终端设备3106的一种示例性结构的示意图。在终端设备3106从捕获设备3102接收到流后,协议处理单元3202分析该流的传输协议。该协议包括但不限于实时流协议(Real Time Streaming Protocol,RTSP)、超文本传输协议(Hyper Text Transfer Protocol,HTTP)、HTTP直播流媒体协议(HTTP Live streaming protocol,HLS)、MPEG-DASH、实时传输协议(Real-time Transport protocol,RTP)、实时消息传输协议(Real Time Messaging Protocol,RTMP),或其任何种类的组合等。

[1152] 协议处理单元3202对流进行处理后,生成流文件。文件被输出给解复用单元3204。解复用单元3204可以将复用数据分成经编码音频数据和经编码视频数据。如上所述,对于其它实际场景,例如在视频会议系统中,不复用经编码音频数据和经编码视频数据。在这种情况下,不通过解复用单元3204,将经编码数据发送给视频解码器3206和音频解码器3208。

[1153] 通过解复用处理,生成视频基本流(elementary stream,ES)、音频ES和可选的字幕。视频解码器3206,包括上述实施例中说明的视频解码器30,通过上述实施例中所示的解码方法对视频ES进行解码以生成视频帧,并将此数据馈送给同步单元3212。音频解码器3208对音频ES进行解码以生成音频帧,并将此数据馈送给同步单元3212。可选地,在将视频帧馈送给同步单元3212之前可以将视频帧存储在缓冲区(图11中未示出)中。类似地,在将音频帧馈送给同步单元3212之前可以将音频帧存储在缓冲区(图11中未示出)中。

[1154] 同步单元3212同步视频帧和音频帧,并将视频/音频提供给视频/音频显示器3214。例如,同步单元3212同步视频和音频信息的呈现。信息可以使用与经译码音频和可视数据的呈现有关的时间戳以及与数据流本身的传送有关的时间戳而以语法进行译码。

[1155] 如果流中包括字幕,则字幕解码器3210对字幕进行解码,使字幕与视频帧和音频帧同步,并将视频/音频/字幕提供给视频/音频/字幕显示器3216。

[1156] 本发明并不限于上述系统,上述实施例中的图像编码设备或图像解码设备都可以包括在汽车系统等其它系统中。

[1157] 数学运算符

[1158] 本申请中使用的数学运算符与C编程语言中使用的数学运算符类似。但是,本申请准确定义了整除运算和算术移位运算的结果,并且还定义了其它运算,例如幂运算和实值除法。编号和计数规范通常从0开始,例如,“第一个”相当于第0个,“第二个”相当于第1个,以此类推。

[1159] 算术运算符

[1160] 算术运算符的定义如下:

[1161] + 加法运算

[1162] - 减法运算(二元运算符)或者取反(一元前缀运算符)

[1163] * 乘法运算,包括矩阵乘法运算

[1164] x^y 幂运算,表示x的y次幂。根据上下文,这种表示符用作上标,而不是作为幂运算。

[1165] / 整除运算,沿向0的取值方向截断。例如,7/4和-7/-4截断至1,-7/4和7/-4截断至-1。

[1166] ÷ 除法运算,不做截断或取整。

[1167] $\frac{x}{y}$ 除法运算,不做截断或取整。

[1168] $\sum_{i=x}^y f(i)$ i取x到y(包括y)的所有整数值时,函数f(i)的累加和。

[1169] $x \% y$ 模运算,表示x除y的余数,其中,x和y都是整数,并且 $x \geq 0$ 和 $y > 0$ 。

[1170] 逻辑运算符

[1171] 逻辑运算符的定义如下:

[1172] $x \& y$ x和y的布尔逻辑“与”运算

[1173] $x || y$ x和y的布尔逻辑“或”运算

[1174] ! 布尔逻辑“非”运算

[1175] $x ? y : z$ 如果x为真或不等于0,则求y的值,否则,求z的值。

[1176] 关系运算符

[1177] 关系运算符的定义如下:

[1178] > 大于

[1179] >= 大于或等于

[1180] < 小于

[1181] <= 小于或等于

[1182] == 等于

[1183] != 不等于

[1184] 当一个关系运算符应用于一个已被赋值“na”(不适用)的语法元素或变量时,值“na”被视为该语法元素或变量的不同值。值“na”被视为不等于任何其它值。

[1185] 按位运算符

[1186] 按位运算符的定义如下:

[1187] & 按位“与”运算。当对整数参数运算时,运算的是整数值的补码表示。当对二进制

参数运算时,如果它包含的位比另一个参数少,则通过添加更多等于0的有效位来扩展较短的参数。

[1188] | 按位“或”运算。当对整数参数运算时,运算的是整数值的补码表示。当对二进制参数运算时,如果它包含的位比另一个参数少,则通过添加更多等于0的有效位来扩展较短的参数。

[1189] ^ 按位“异或”运算。当对整数参数运算时,运算的是整数值的补码表示。当对二进制参数运算时,如果它包含的位比另一个参数少,则通过添加更多等于0的有效位来扩展较短的参数。

[1190] x>>y 将x以2的补码整数表示的形式向右算术移动y个二进制位。只有y为非负整数时才有这个函数定义。右移的结果是移进最高有效位 (most significant bit,MSB) 的比特位等于移位运算之前的x的MSB。

[1191] x<<y 将x以2的补码整数表示的形式向左算术移动y个二进制位。只有y为非负整数时才有这个函数定义。左移的结果是移进最低有效位 (least significant bit,LSB) 的比特位等于0。

[1192] 赋值运算符

[1193] 算术运算符的定义如下:

[1194] = 赋值运算符

[1195] ++ 递增,即x++相当于x=x+1;当用于数组下标时,在自加运算前先求变量值。

[1196] -- 递减,即x--相当于x=x-1;当用于数组下标时,在自减运算前先求变量值。

[1197] += 自加指定值,即x+=3相当于x=x+3,x+=(-3)相当于x=x+(-3)。

[1198] -= 自减指定值,即x-=3相当于x=x-3,x-=-3相当于x=x-(-3)。

[1199] 范围表示法

[1200] 下面的表示法用来说明值的范围:

[1201] x=y..z x取从y到z (包括y和z) 的整数值,其中,x、y和z是整数,z大于y。

[1202] 数学函数

[1203] 数学函数的定义如下:

[1204]
$$\text{Abs}(x) = \begin{cases} x & ; \quad x \geq 0 \\ -x & ; \quad x < 0 \end{cases}$$

[1205] Asin(x) 三角反正弦函数,对参数x运算,x在-1.0至1.0 (包括端值) 的范围之间,输出值在 $-\pi \div 2$ 至 $\pi \div 2$ (包括端值) 范围之间,单位为弧度。

[1206] Atan(x) 三角反正切函数,对参数x运算,输出值在 $-\pi \div 2$ 至 $\pi \div 2$ (包括端值) 范围之间,单位为弧度。

[1207]
$$\text{Atan2}(y, x) = \begin{cases} \text{Atan}\left(\frac{y}{x}\right) & ; \quad x > 0 \\ \text{Atan}\left(\frac{y}{x}\right) + \pi & ; \quad x < 0 \ \&\& \ y \geq 0 \\ \text{Atan}\left(\frac{y}{x}\right) - \pi & ; \quad x < 0 \ \&\& \ y < 0 \\ +\frac{\pi}{2} & ; \quad x = 0 \ \&\& \ y \geq 0 \\ -\frac{\pi}{2} & ; \quad \text{其它} \end{cases}$$

[1208] Ceil(x) 大于或等于x的最小整数。

[1209] $\text{Clip1}_y(x) = \text{Clip3}(0, (1 \ll \text{BitDepth}_y) - 1, x)$

[1210] $\text{Clip1}_c(x) = \text{Clip3}(0, (1 \ll \text{BitDepth}_c) - 1, x)$

[1211]
$$\text{Clip3}(x, y, z) = \begin{cases} x & ; z < x \\ y & ; z > y \\ z & ; \text{其它} \end{cases}$$

[1212] $\text{Cos}(x)$ 三角余弦函数,对参数 x 运算,单位为弧度。

[1213] $\text{Floor}(x)$ 小于或等于 x 的最大整数。

[1214]
$$\text{GetCurrMsb}(a, b, c, d) = \begin{cases} c + d & ; b - a \geq d / 2 \\ c - d & ; a - b > d / 2 \\ c & ; \text{其它} \end{cases}$$

[1215] $\text{Ln}(x)$ 返回 x 的自然对数(以 e 为底的对数,其中, e 是自然对数底数常数2.718281828……)。

[1216] $\text{Log2}(x)$ x 以2为底的对数。

[1217] $\text{Log10}(x)$ x 以10为底的对数。

[1218]
$$\text{Min}(x, y) = \begin{cases} x & ; x \leq y \\ y & ; x > y \end{cases}$$

[1219]
$$\text{Max}(x, y) = \begin{cases} x & ; x \geq y \\ y & ; x < y \end{cases}$$

[1220] $\text{Round}(x) = \text{Sign}(x) * \text{Floor}(\text{Abs}(x) + 0.5)$

[1221]
$$\text{Sign}(x) = \begin{cases} 1 & ; x > 0 \\ 0 & ; x = 0 \\ -1 & ; x < 0 \end{cases}$$

[1222] $\text{Sin}(x)$ 表示三角正弦函数,对参数 x 运算,单位为弧度。

[1223] $\text{Sqrt}(x) = \sqrt{x}$

[1224] $\text{Swap}(x, y) = (y, x)$

[1225] $\text{Tan}(x)$ 表示三角正切函数,对参数 x 运算,单位为弧度。

[1226] 运算优先级顺序

[1227] 当没有使用括号来显式表示表达式中的优先顺序时,以下规则适用:

[1228] -高优先级的运算在低优先级的任何运算之前计算。

[1229] -相同优先级的运算从左到右依次计算。

[1230] 下表从最高到最低的顺序说明运算的优先级,表中位置越高,优先级越高。

[1231] 对于C编程语言中也使用的运算符,本规范中运算符优先级顺序与C编程语言中优先级顺序相同。

[1232] 表:运算优先级从最高(表格顶部)到最低(表格底部)排序

[1233]

运算（操作数 x 、 y 和 z ）
“ $x++$ ”，“ $x--$ ”
“ $!x$ ”，“ $-x$ ”（一元前缀运算符）
x^y
“ $x * y$ ”，“ x/y ”，“ $x \div y$ ”，“ $\frac{x}{y}$ ”，“ $x \% y$ ”
“ $x + y$ ”，“ $x - y$ ”（二元运算符），“ $\sum_{i=x}^y f(i)$ ”
“ $x < y$ ”，“ $x > y$ ”
“ $x < y$ ”，“ $x \leq y$ ”，“ $x > y$ ”，“ $x \geq y$ ”
“ $x = y$ ”，“ $x \neq y$ ”
“ $x \& y$ ”
“ $x y$ ”
“ $x \&\& y$ ”
“ $x \parallel y$ ”
“ $x ? y : z$ ”
“ $x..y$ ”
“ $x = y$ ”，“ $x += y$ ”，“ $x -= y$ ”

[1234] 逻辑运算的文本说明

[1235] 在文本中,用数学形式描述如下的逻辑运算语句:

if(condition 0)

statement 0

else if(condition 1)

[1236] statement 1

...

else /* informative remark on remaining condition */

statement n

[1237] 可以用以下方式描述:

[1238] ……如下/……以下为准:

[1239] -如果条件0,则语句0

[1240] -否则,如果条件1,则语句1

[1241] -……

[1242] -否则(关于剩余条件的提示性说明),则语句n

[1243] 文本中的每个“如果……否则,如果……否则,……”语句都以“……如下”或“……以下为准”引入,后面紧跟着“如果……”。“如果……否则,如果……否则,……”的最后一个

条件总有一个“否则,……”中间有“如果……否则,如果……否则”语句可以通过使“……如下”或“……以下为准”与结尾“否则……”匹配来识别。

[1244] 在文本中,用数学形式描述如下的逻辑运算语句:

if(condition 0a && condition 0b)

statement 0

[1245] else if(condition 1a || condition 1b)

statement 1

...

[1246] else

statement n

[1247] 可以用以下方式描述:

[1248] ……如下/……以下为准:

[1249] -如果以下所有条件为真,则语句0:

[1250] -条件0a

[1251] -条件0b

[1252] -否则,如果满足以下一个或多个条件,则语句1:

[1253] -条件1a

[1254] -条件1b

[1255] -……

[1256] -否则,语句n

[1257] 在文本中,用数学形式描述如下的逻辑运算语句:

if(condition 0)

statement 0

[1258] if(condition 1)

statement 1

[1259] 可以用以下方式描述:

[1260] 当条件0时,语句0

[1261] 当条件1时,语句1。

[1262] 编码器20和解码器30等的实施例以及本文参照编码器20和解码器30等描述的功能可以在硬件、软件、固件或其任意组合中实现。如果在软件中实现,则这些功能可以作为一个或多个指令或代码存储在计算机可读介质中或通过通信介质发送,且由基于硬件的处理单元执行。计算机可读介质可以包括与有形介质(例如,数据存储介质)对应的计算机可读存储介质,或包括任何便于将计算机程序从一处传送到另一处的介质(例如,根据通信协议)的通信介质。通过这种方式,计算机可读介质通常可以对应(1)非瞬时性的有形计算机可读存储介质或(2)信号或载波等通信介质。数据存储介质可以通过一个或多个计算机或一个或多个处理器访问的任何可用介质,以检索用于实施本发明所述技术的指令、代码和/或数据结构。计算机程序产品可以包括计算机可读介质。

[1263] 作为示例而非限制,这类计算机可读存储介质可以包括RAM、ROM、EEPROM、CD-ROM或其它光盘存储器、磁盘存储器或其它磁性存储设备、闪存或可以用于存储指令或数据结

构形式的所需程序代码并且可以由计算机访问的任何其它介质。此外,任何连接都可以适当地称为计算机可读介质。例如,如果使用同轴缆线、光纤缆线、双绞线、数字用户线(digital subscriber line,DSL)或红外线、无线电和微波等无线技术从网站、服务器或其它远程源传输指令,则同轴缆线、光纤缆线、双绞线、DSL或红外线、无线电和微波等无线技术包含在介质的定义中。但是,应当理解的是,计算机可读存储介质和数据存储介质并不包括连接、载波、信号或其它瞬时性介质,而是涉及非瞬时性有形存储介质。本文所使用的磁盘和光盘包括压缩光盘(compact disc,CD)、激光光盘、光学光盘、数字多功能光盘(digital versatile disc,DVD)和蓝光光盘,其中,磁盘通常以磁性方式再现数据,而光盘利用激光以光学方式再现数据。以上各项的组合也应包括在计算机可读介质的范围内。

[1264] 指令可以通过一个或多个数字信号处理器(digital signal processor,DSP)、一个或多个通用微处理器、一个或多个专用集成电路(application specific integrated circuit,ASIC)、一个或多个现场可编程逻辑阵列(field programmable logic array,FPGA)或其它同等集成或离散逻辑电路等一或多个处理器来执行。因此,本文所使用的术语“处理器”可以指上述结构中的任一种或适于实施本文所述技术的任何其它结构。另外,在一些方面中,本文描述的各种功能可以提供在用于编码和解码的专用硬件和/或软件模块内,或者并入在组合编解码器中。而且,这些技术可以在一个或多个电路或逻辑元件中完全实现。

[1265] 本发明中的技术可以在多种设备或装置中实现,这些设备或装置包括无线手机、集成电路(integrated circuit,IC)或一组IC(例如芯片组)。本发明描述了各种组件、模块或单元,以强调用于执行所公开技术的设备的功能方面,但未必需要由不同的硬件单元实现。实际上,如上所述,各种单元可以结合合适的软件和/或固件组合在编解码器硬件单元中,或者通过包括如上所述的一个或多个处理器的互操作硬件单元的集合来提供。

[1266] 下面提供了本发明的其它实施例。需要说明的是,以下部分使用的编号不一定需要与上述部分使用的编号保持一致。

[1267] 实施例1:一种用于图像块的帧间预测的方法,其中,对一组预测块执行加权预测参数的指示和非矩形帧间预测的启用,所述方法包括:获取块的帧间预测模式参数,其中,所述获取包括检查是否对所述块组启用非矩形帧间预测模式,所述块组包括所述预测块;获取与所述块关联的加权预测参数、一个块相对于为所述块指示的参考图像的帧间预测模式参数和为所述块组指定的加权预测参数。

[1268] 实施例2:根据实施例1所述的方法,其中,通过指示大于1的三角融合候选(MaxNumTriangleMergeCand)的最大数量来执行非矩形帧间预测的启用。

[1269] 实施例3:根据实施例1或2所述的方法,其中,当加权预测参数表示对至少一个参考索引启用加权预测时,非矩形帧间预测被推断为禁用。

[1270] 实施例4:根据实施例1至3中任一个所述的方法,其中,块组是图像,帧间预测非矩形模式参数的启用和加权预测参数都在图像头中指示。

[1271] 实施例5:根据实施例1至4中任一个所述的方法,其中,块组是条带,帧间预测非矩形模式参数的启用和加权预测参数都在条带头中指示。

[1272] 实施例6:根据实施例1至5中任一个所述的方法,其中,帧间预测模式参数包括用于确定所述参考图像的参考索引和用于确定所述参考块在所述参考图像中的位置的运动

矢量信息。

[1273] 实施例7:根据实施例1至6中任一个所述的方法,其中,非矩形融合模式是三角划分模式。

[1274] 实施例8:根据实施例1至7中任一个所述的方法,其中,非矩形融合模式是GEO模式。

[1275] 实施例9:根据实施例1至8中任一个所述的方法,其中,加权预测是条带级亮度补偿机制(例如全局加权预测)。

[1276] 实施例10:根据实施例1至9中任一个所述的方法,其中,加权预测是块级亮度补偿机制,例如局部照明补偿(local illumination compensation,LIC)。

[1277] 实施例11:根据实施例1至10中任一个所述的方法,其中,加权预测参数包括:一组标志,表示加权预测是否应用于所述预测块的亮度分量和色度分量;线性模型参数 α 和 β ,指定所述预测块的值的线性变换。

[1278] 在本申请的第一方面中,如图12所示,公开了一种帧间预测方法1200,包括:S1201:确定是否允许非矩形帧间预测模式用于块组;S1202:获取所述块组的一个或多个帧间预测模式参数和加权预测参数;S1203:根据所述一个或多个帧间预测模式参数和加权预测参数,获取当前块的预测值,其中,所述一个或多个帧间预测模式参数中的一个帧间预测模式参数表示所述当前块的参考图像信息,所述块组包括所述当前块。

[1279] 在一种可行的实现方式中,所述参考图像信息包括是否对参考图像索引启用加权预测;如果加权预测被启用,则所述非矩形帧间预测模式被禁用。

[1280] 在一种可行的实现方式中,如果加权预测被禁用,则所述非矩形帧间预测模式被启用。

[1281] 在一种可行的实现方式中,所述确定是否允许非矩形帧间预测模式包括:指示三角融合候选的最大数量(MaxNumTriangleMergeCand)大于1。

[1282] 在一种可行的实现方式中,所述块组包括图像,所述加权预测参数和用于确定是否允许所述非矩形帧间预测模式的指示信息在所述图像的图像头中。

[1283] 在一种可行的实现方式中,所述块组包括条带,所述加权预测参数和用于确定是否允许所述非矩形帧间预测模式的指示信息在所述条带的条带头中。

[1284] 在一种可行的实现方式中,所述非矩形帧间预测模式为三角划分模式。

[1285] 在一种可行的实现方式中,所述非矩形帧间预测模式为几何(GEO)划分模式。

[1286] 在一种可行的实现方式中,所述加权预测参数用于条带级亮度补偿。

[1287] 在一种可行的实现方式中,所述加权预测参数用于块级亮度补偿。

[1288] 在一种可行的实现方式中,所述加权预测参数包括:标志,表示所述加权预测是否应用于预测块的亮度分量和/或色度分量;线性模型参数,指定所述预测块的值的线性变换。

[1289] 在本申请的第二方面中,公开了一种用于帧间预测的装置1300,如图13所示,包括:非瞬时性存储器1301,存储有处理器可执行指令;处理器1302,耦合到存储器1301并用于执行所述处理器可执行指令,以实现本申请所述第一方面中的任一可行的实现方式。

[1290] 在本申请的第三方面中,一种用于帧间预测的码流包括:用于确定是否允许非矩形帧间预测模式用于块组的指示信息;所述块组的一个或多个帧间预测模式参数和加权预

测参数,其中,当前块的预测值是根据所述一个或多个帧间预测模式参数和加权预测参数获得的,所述一个或多个帧间预测模式参数中的一个帧间预测模式参数表示所述当前块的参考图像信息,所述块组包括所述当前块。

[1291] 在一种可行的实现方式中,所述参考图像信息包括是否对参考图像索引启用加权预测;如果加权预测被启用,则所述非矩形帧间预测模式被禁用。

[1292] 在一种可行的实现方式中,如果加权预测被禁用,则所述非矩形帧间预测模式被启用。

[1293] 在一种可行的实现方式中,所述指示信息包括三角融合候选的最大数量(MaxNumTriangleMergeCand)大于1。

[1294] 在一种可行的实现方式中,所述块组包括图像,所述加权预测参数和所述指示信息在所述图像的图像头中。

[1295] 在一种可行的实现方式中,所述块组由条带组成,所述加权预测参数和所述指示信息在所述条带的条带头中。

[1296] 在一种可行的实现方式中,所述非矩形帧间预测模式为三角划分模式。

[1297] 在一种可行的实现方式中,所述非矩形帧间预测模式为几何(GEO)划分模式。

[1298] 在一种可行的实现方式中,所述加权预测参数用于条带级亮度补偿。

[1299] 在一种可行的实现方式中,所述加权预测参数用于块级亮度补偿。

[1300] 在一种可行的实现方式中,所述加权预测参数包括:标志,表示所述加权预测是否应用于预测块的亮度分量和/或色度分量;线性模型参数,指定所述预测块的值的线性变换。

[1301] 在本申请的第四方面中,如图14所示,公开了一种帧间预测装置1400,包括:确定模块1401,用于确定是否允许非矩形帧间预测模式用于块组;获取模块1402,用于获取所述块组的一个或多个帧间预测模式参数和加权预测参数;预测模块1403,用于根据所述一个或多个帧间预测模式参数和加权预测参数,获取当前块的预测值,其中,所述一个或多个帧间预测模式参数中的一个帧间预测模式参数表示所述当前块的参考图像信息,所述块组包括所述当前块。

[1302] 在一种可行的实现方式中,所述参考图像信息包括是否对参考图像索引启用加权预测;如果加权预测被启用,则所述非矩形帧间预测模式被禁用。

[1303] 在一种可行的实现方式中,如果加权预测被禁用,则所述非矩形帧间预测模式被启用。

[1304] 在一种可行的实现方式中,确定模块1401具体用于:指示三角融合候选的最大数量(MaxNumTriangleMergeCand)大于1。

[1305] 在一种可行的实现方式中,所述块组包括图像,所述加权预测参数和用于确定是否允许所述非矩形帧间预测模式的指示信息在所述图像的图像头中。

[1306] 在一种可行的实现方式中,所述块组包括条带,所述加权预测参数和用于确定是否允许所述非矩形帧间预测模式的指示信息在所述条带的条带头中。

[1307] 在一种可行的实现方式中,所述非矩形帧间预测模式为三角划分模式。

[1308] 在一种可行的实现方式中,所述非矩形帧间预测模式为几何(GEO)划分模式。

[1309] 在一种可行的实现方式中,所述加权预测参数用于条带级亮度补偿。

[1310] 在一种可行的实现方式中,所述加权预测参数用于块级亮度补偿。

[1311] 在一种可行的实现方式中,所述加权预测参数包括:标志,表示所述加权预测是否应用于预测块的亮度分量和/或色度分量;线性模型参数,指定所述预测块的值的线性变换。

[1312] 现有技术提供的方法可以概括为以下各方面:

[1313] 方面1.一种帧间预测方法,其中,所述方法包括:

[1314] 确定是否允许非矩形帧间预测模式用于块组;

[1315] 获取所述块组的一个或多个帧间预测模式参数和加权预测参数;

[1316] 根据所述一个或多个帧间预测模式参数和加权预测参数,获取当前块的预测值,其中,所述一个或多个帧间预测模式参数中的一个帧间预测模式参数表示所述当前块的参考图像信息,所述块组包括所述当前块。

[1317] 方面2.根据方面1所述的方法,其中,所述参考图像信息包括是否对参考图像索引启用加权预测;如果加权预测被启用,则所述非矩形帧间预测模式被禁用。

[1318] 方面3.根据方面1或2所述的方法,其中,如果加权预测被禁用,则所述非矩形帧间预测模式被启用。

[1319] 方面4.根据方面1至3中任一个所述的方法,其中,所述确定是否允许非矩形帧间预测模式包括:

[1320] 指示三角融合候选的最大数量(MaxNumTriangleMergeCand)大于1。

[1321] 方面5.根据方面1至4中任一个所述的方法,其中,所述块组包括图像,所述加权预测参数和用于确定是否允许所述非矩形帧间预测模式的指示信息在所述图像的图像头中。

[1322] 方面6.根据方面1至4中任一个所述的方法,其中,所述块组包括条带,所述加权预测参数和用于确定是否允许所述非矩形帧间预测模式的指示信息在所述条带的条带头中。

[1323] 方面7.根据方面1至6中任一个所述的方法,其中,所述非矩形帧间预测模式为三角划分模式。

[1324] 方面8.根据方面1至6中任一个所述的方法,其中,所述非矩形帧间预测模式为几何(GEO)划分模式。

[1325] 方面8a.根据方面1至8中任一个所述的方法,其中,与融合模式候选的数量相关的语法元素(表示用于确定所述非矩形帧间预测的信息)在序列参数集(sequence parameter set,SPS)指示。

[1326] 方面8b.根据方面1至8a中任一个所述的方法,其中,当图像只包括一个条带时,图像头在条带头中指示。

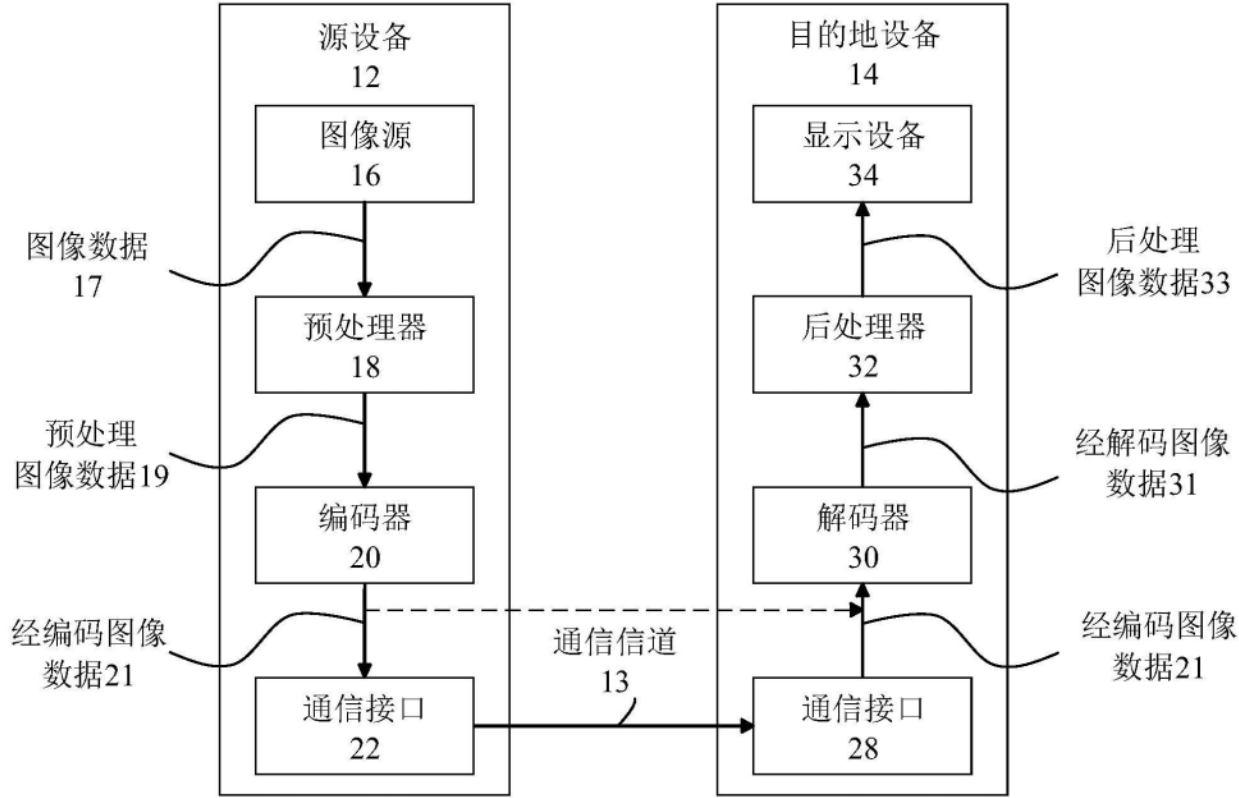
[1327] 方面8c.根据方面1至8b中任一个所述的方法,其中,当图像只包括一个条带时,图像头在条带头中指示。

[1328] 方面8d.根据方面1至8c中任一个所述的方法,其中,图像参数集包括标志,所述标志的值定义加权预测参数是存在于图像头中还是存在于条带头中。

[1329] 方面8e.根据方面1至8d中任一个所述的方法,其中,图像头中的标志表示非帧内类型的条带是否存在以及是否为所述条带指示帧间预测模式参数。

[1330] 方面9.根据方面1至8中任一个所述的方法,其中,所述加权预测参数用于条带级亮度补偿。

- [1331] 方面10.根据方面1至8中任一个所述的方法,其中,所述加权预测参数用于块级亮度补偿。
- [1332] 方面11.根据方面1至10中任一个所述的方法,其中,所述加权预测参数包括:
- [1333] 标志,表示加权预测是否应用于预测块的亮度分量和/或色度分量;
- [1334] 线性模型参数,指定所述预测块的值的线性变换。
- [1335] 方面12.一种用于帧间预测的装置,其中,所述装置包括:
- [1336] 非瞬时性存储器,存储有处理器可执行指令;
- [1337] 处理器,耦合到所述存储器并用于执行所述处理器可执行指令,以实现方面1至11中任一个所述的方法。
- [1338] 方面13.一种用于帧间预测的码流,其中,所述码流包括:
- [1339] 指示信息,用于确定是否允许非矩形帧间预测模式用于块组;
- [1340] 所述块组的一个或多个帧间预测模式参数和加权预测参数,其中,当前块的预测值是根据所述一个或多个帧间预测模式参数和加权预测参数获得的,所述一个或多个帧间预测模式参数中的一个帧间预测模式参数表示所述当前块的参考图像信息,所述块组包括所述当前块。
- [1341] 方面14.根据方面13所述的码流,其中,所述参考图像信息包括是否对参考图像索引启用加权预测;如果加权预测被启用,则所述非矩形帧间预测模式被禁用。
- [1342] 方面15.根据方面13或14所述的码流,其中,如果加权预测被禁用,则所述非矩形帧间预测模式被启用。
- [1343] 方面16.根据方面13至15中任一个所述的码流,其中,所述指示信息包括三角融合候选的最大数量(MaxNumTriangleMergeCand)大于1。
- [1344] 方面17.根据方面13至16中任一个所述的码流,其中,所述块组包括图像,所述加权预测参数和所述指示信息在所述图像的图像头中。
- [1345] 方面18.根据方面13至17中任一个所述的码流,其中,所述块组包括条带,所述加权预测参数和所述指示信息在所述条带的条带头中。
- [1346] 方面19.根据方面13至18中任一个所述的码流,其中,所述非矩形帧间预测模式为三角划分模式。
- [1347] 方面20.根据方面13至19中任一个所述的码流,其中,所述非矩形帧间预测模式为几何(GEO)划分模式。
- [1348] 方面21.根据方面13至20中任一个所述的码流,其中,所述加权预测参数用于条带级亮度补偿。
- [1349] 方面22.根据方面13至20中任一项所述的码流,其中,所述加权预测参数用于块级亮度补偿。
- [1350] 方面23.根据方面13至22中任一个所述的码流,其中,所述加权预测参数包括:
- [1351] 标志,表示加权预测是否应用于预测块的亮度分量和/或色度分量;
- [1352] 线性模型参数,指定所述预测块的值的线性变换。



10

图1A

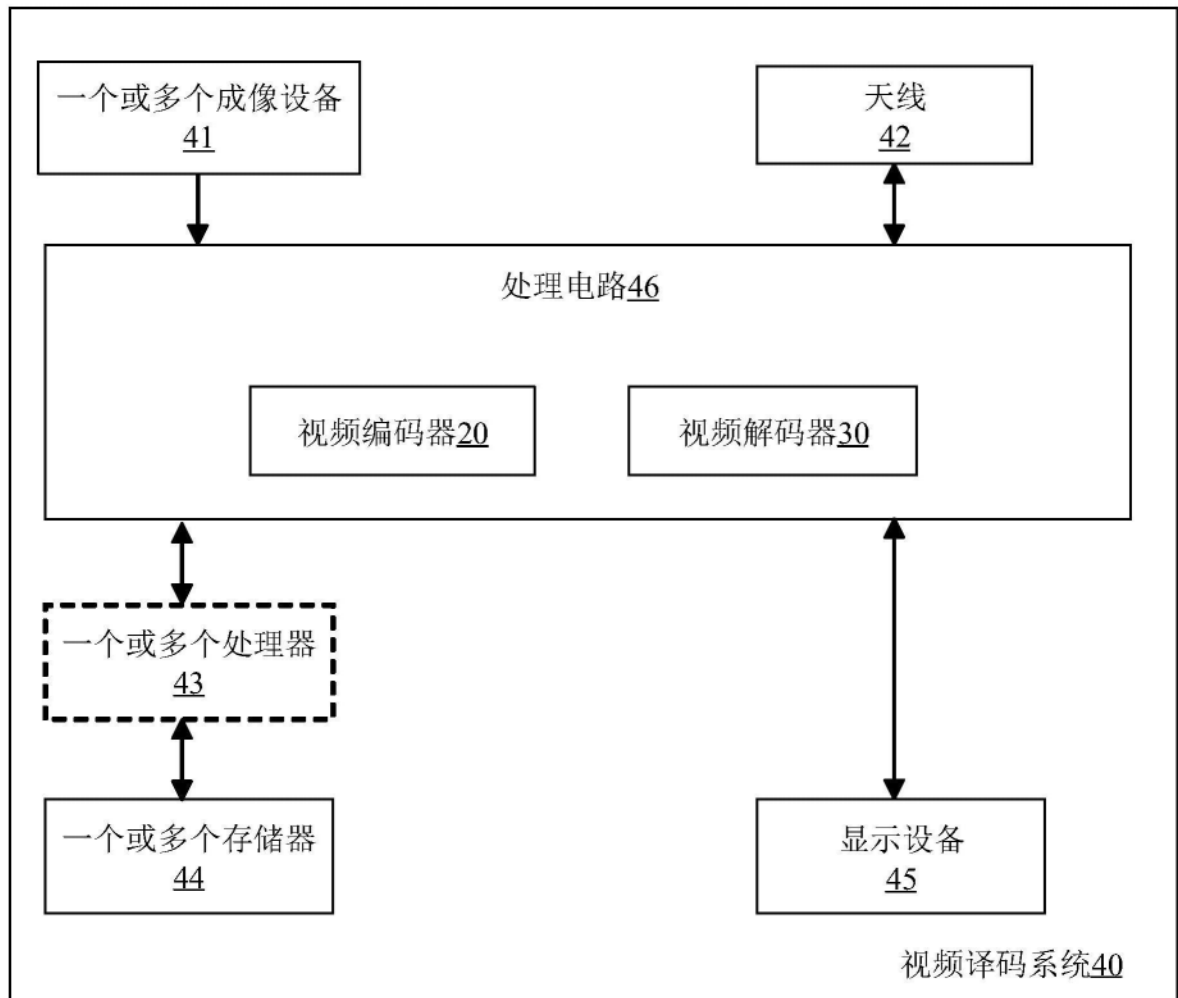


图1B

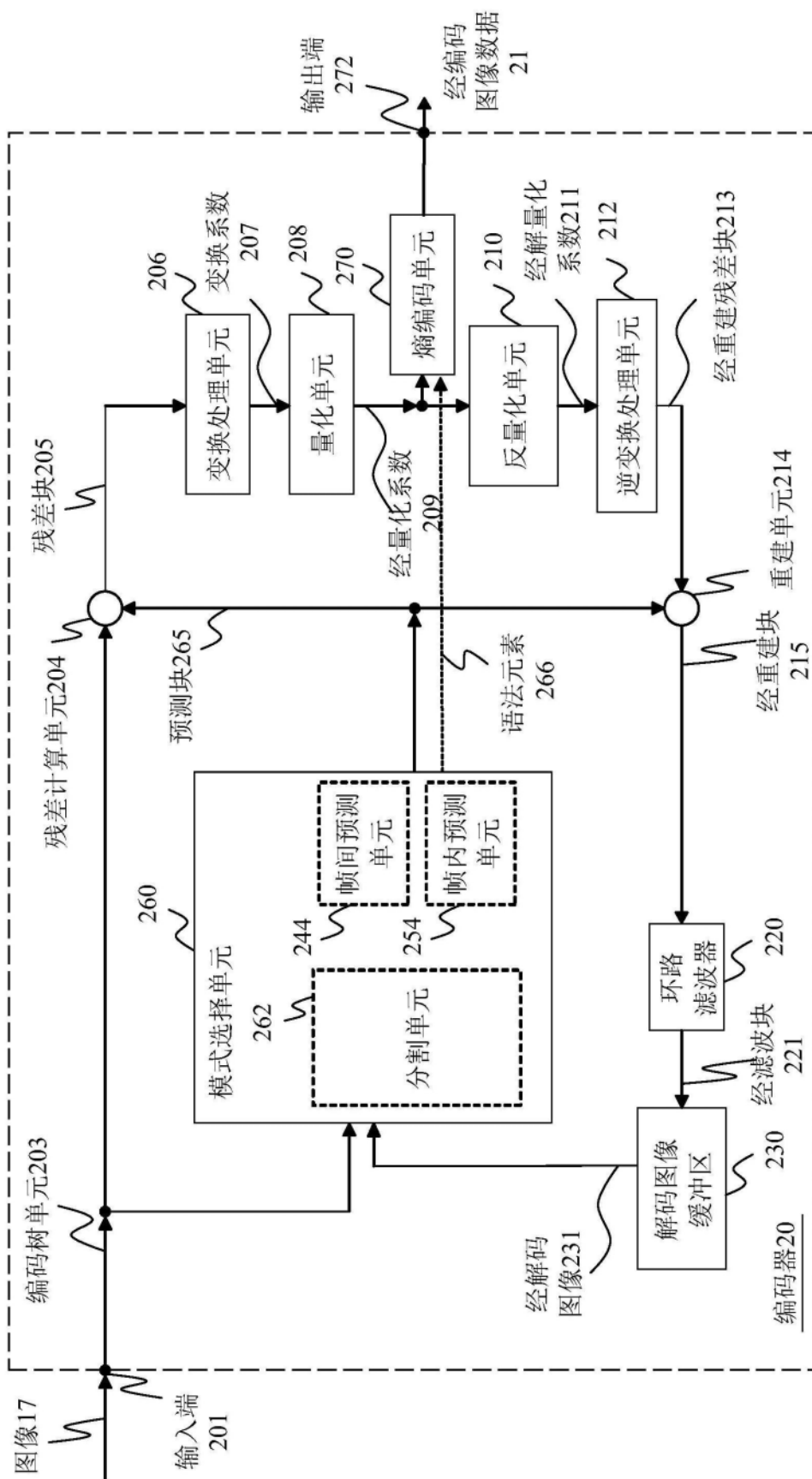


图2

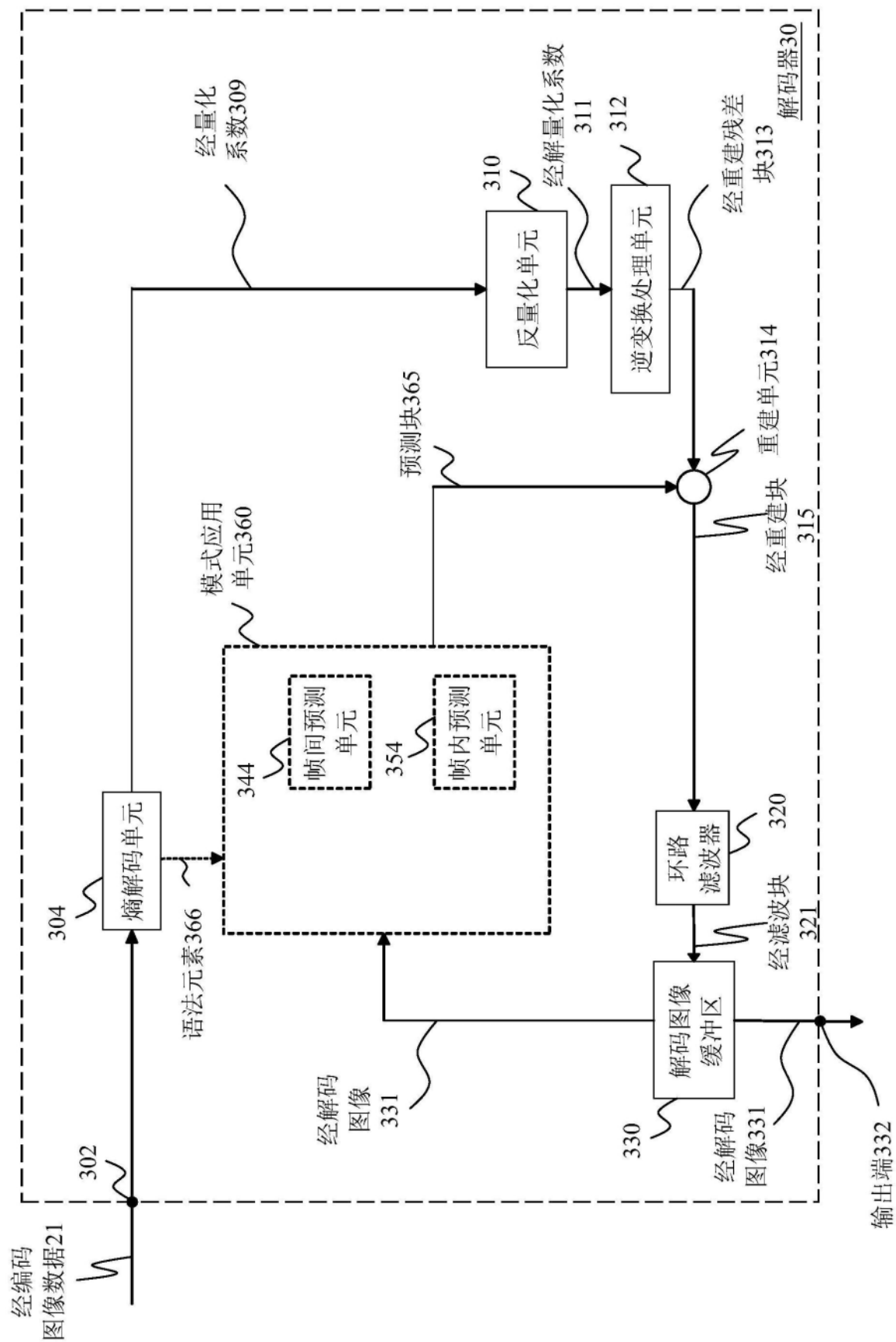


图3

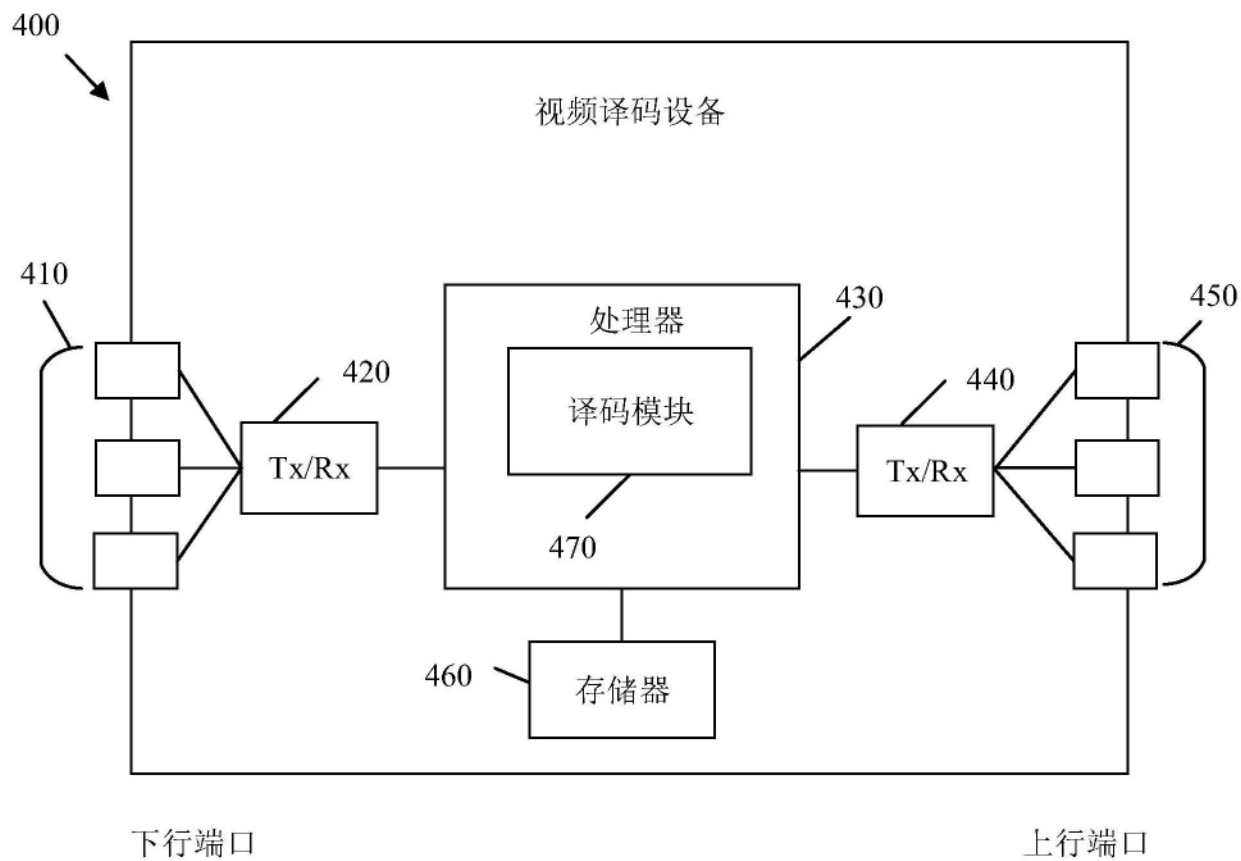


图4

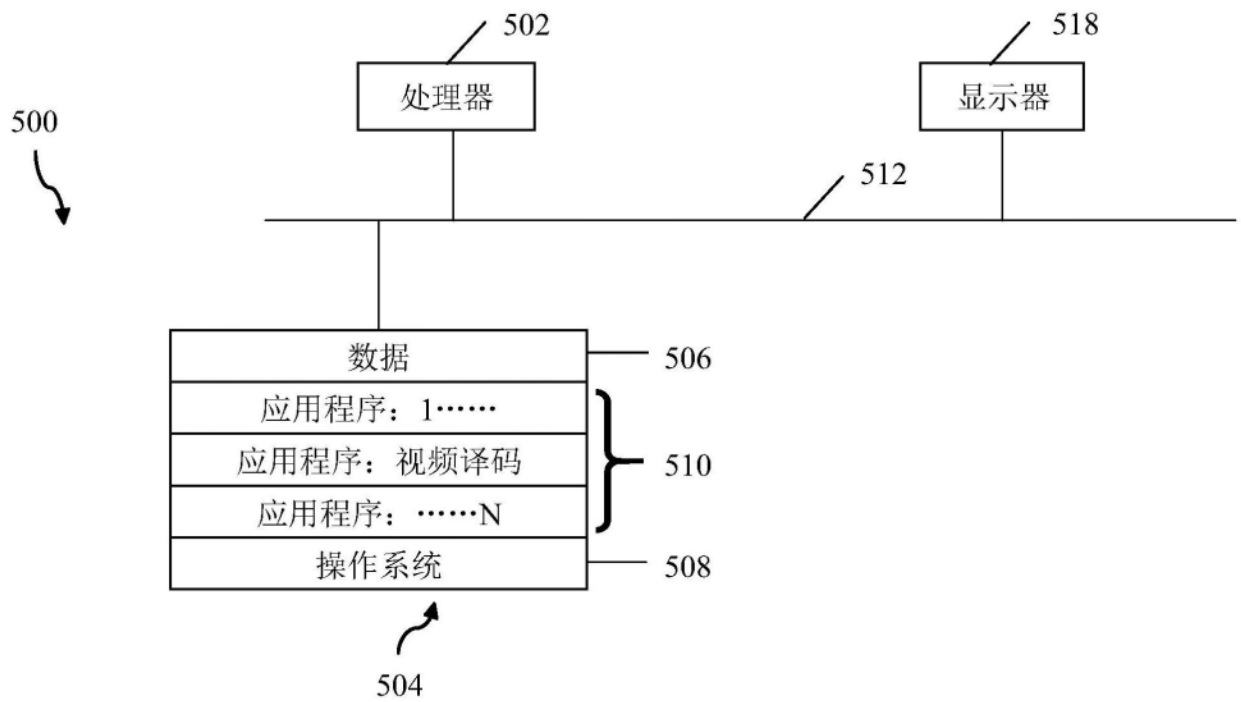


图5

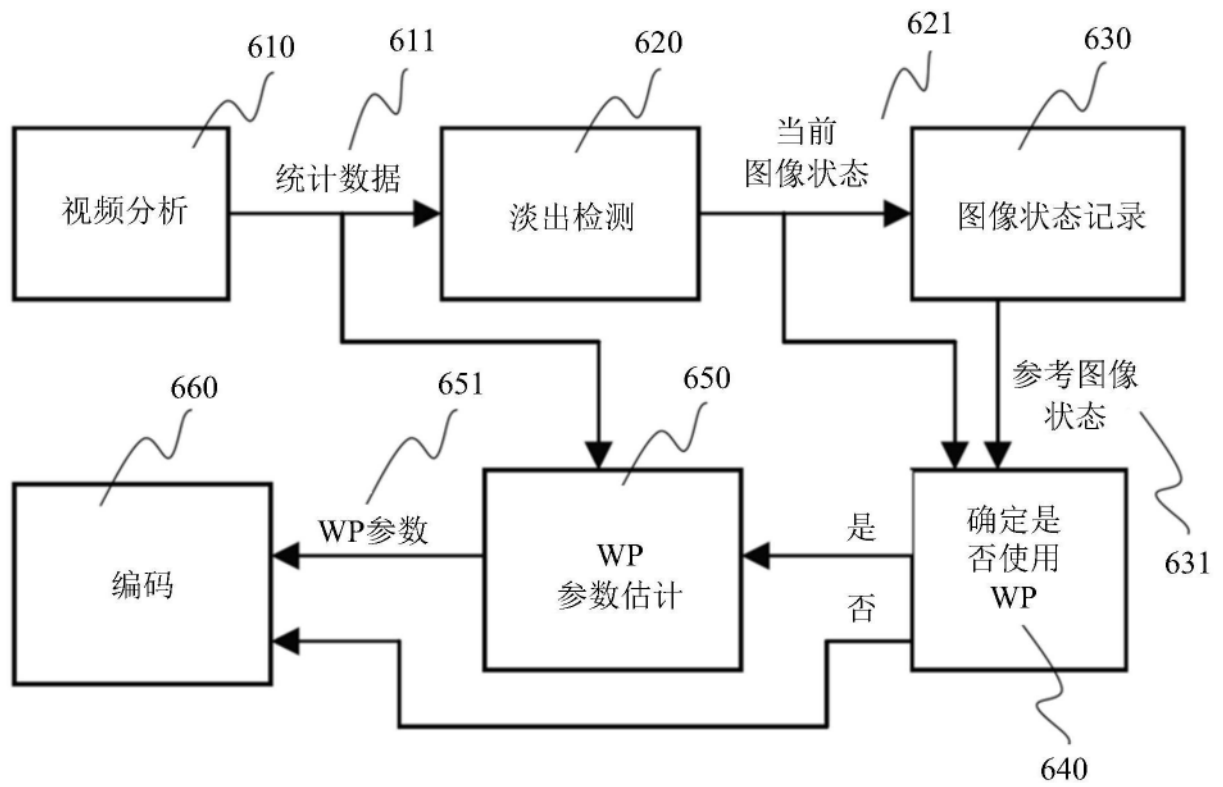


图6

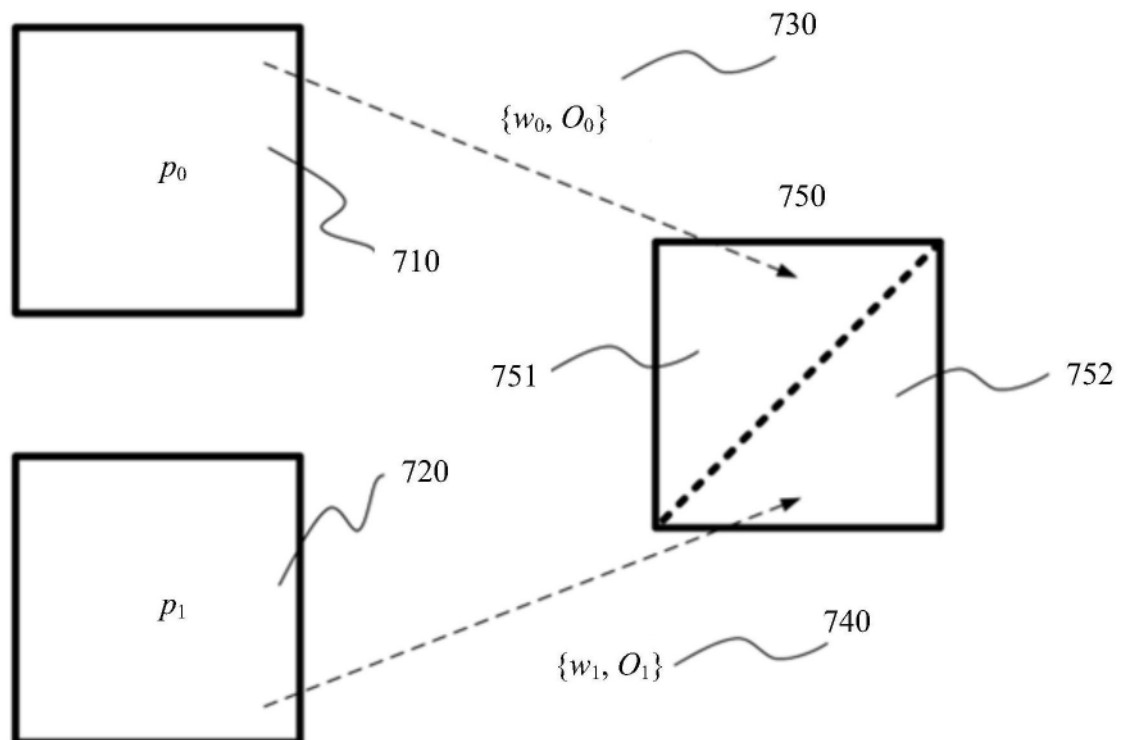


图7

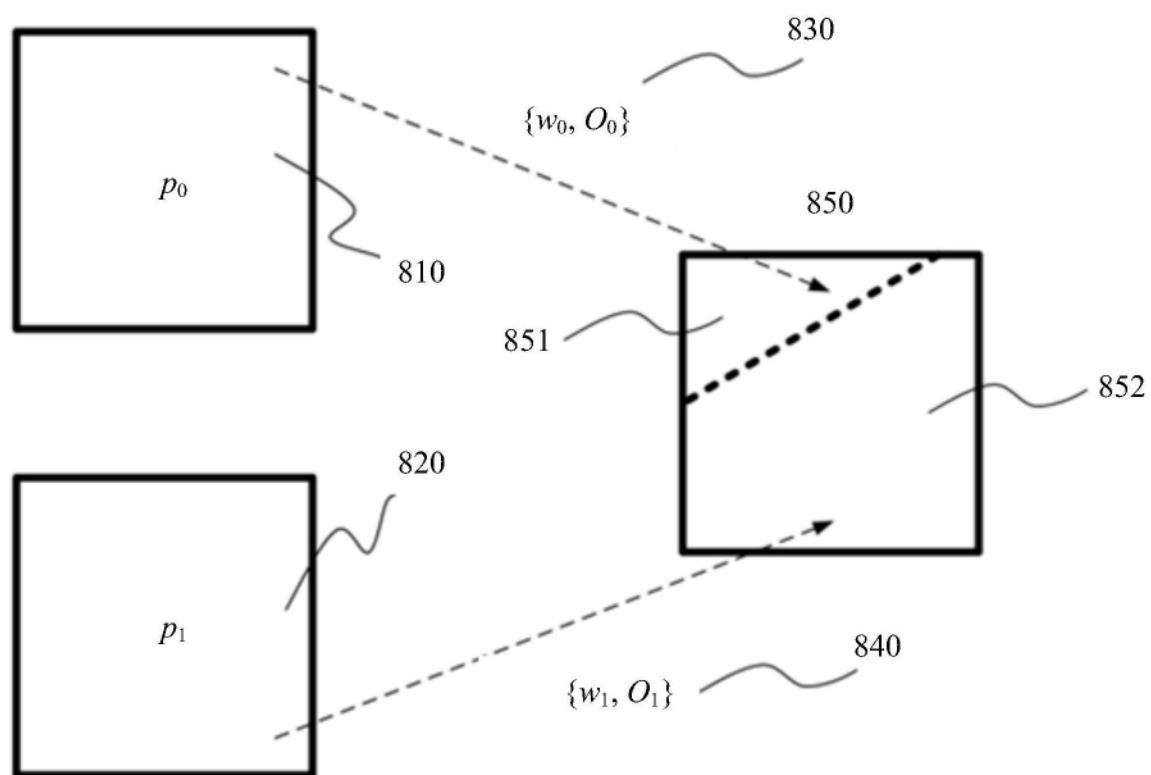


图8

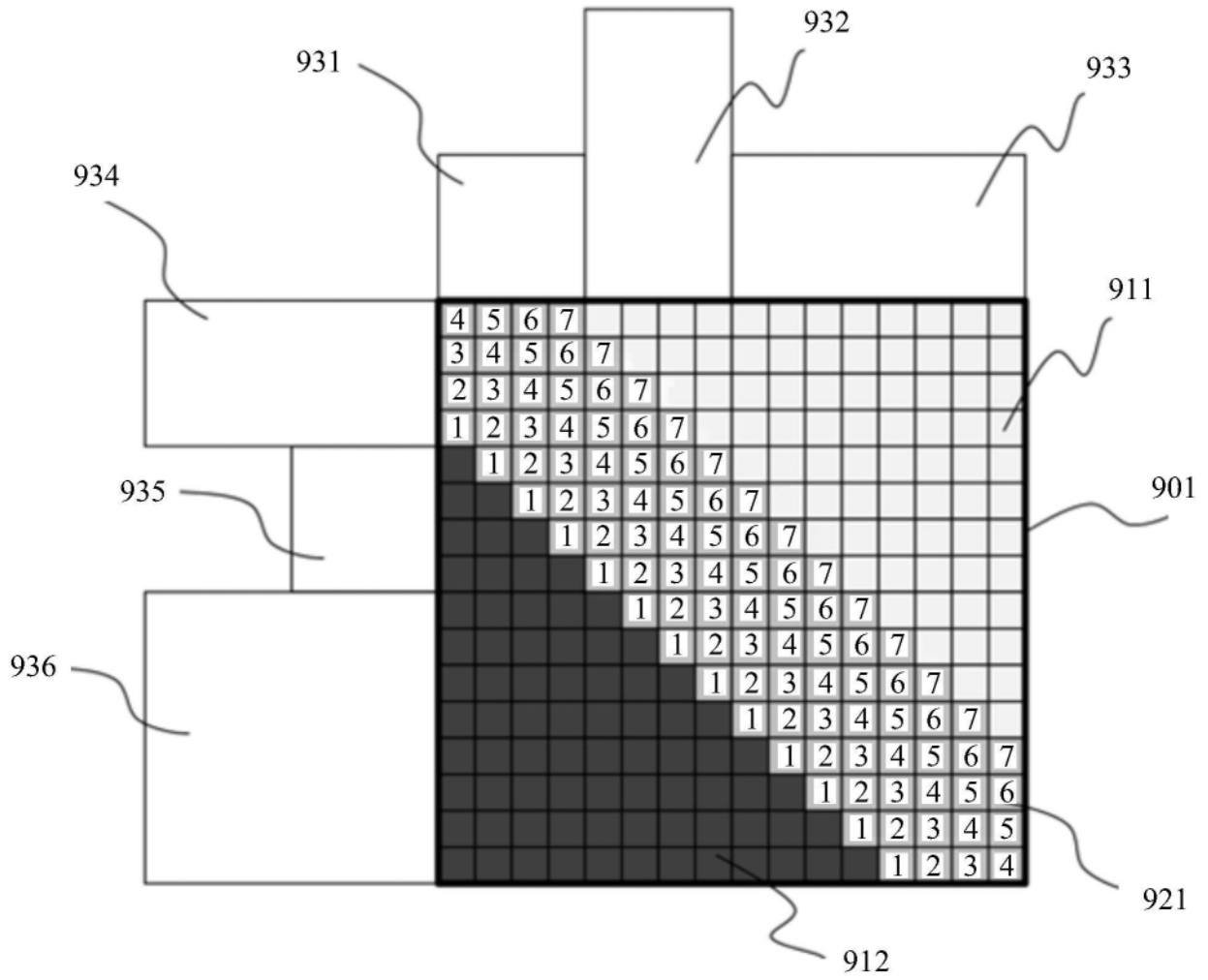


图9

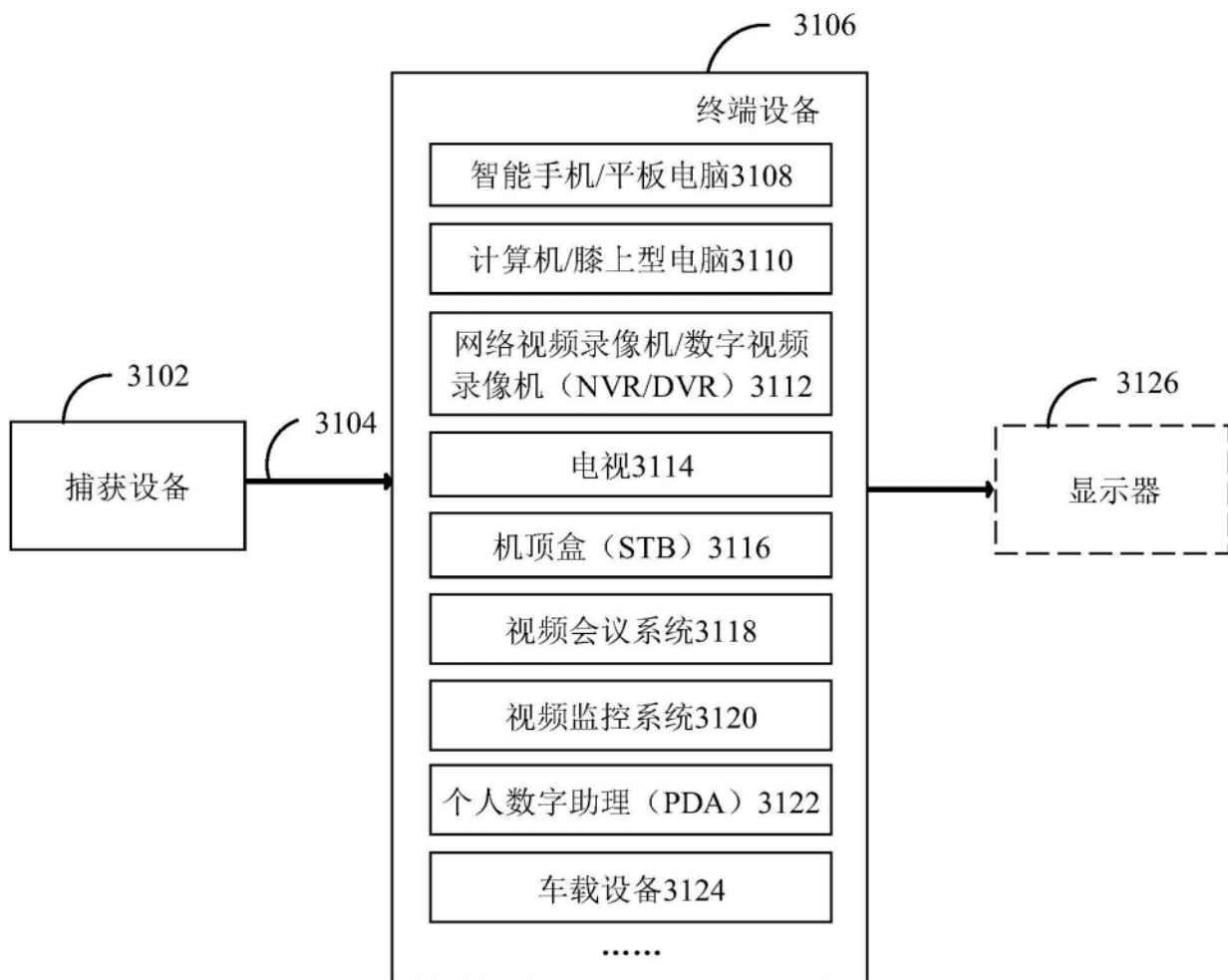


图10

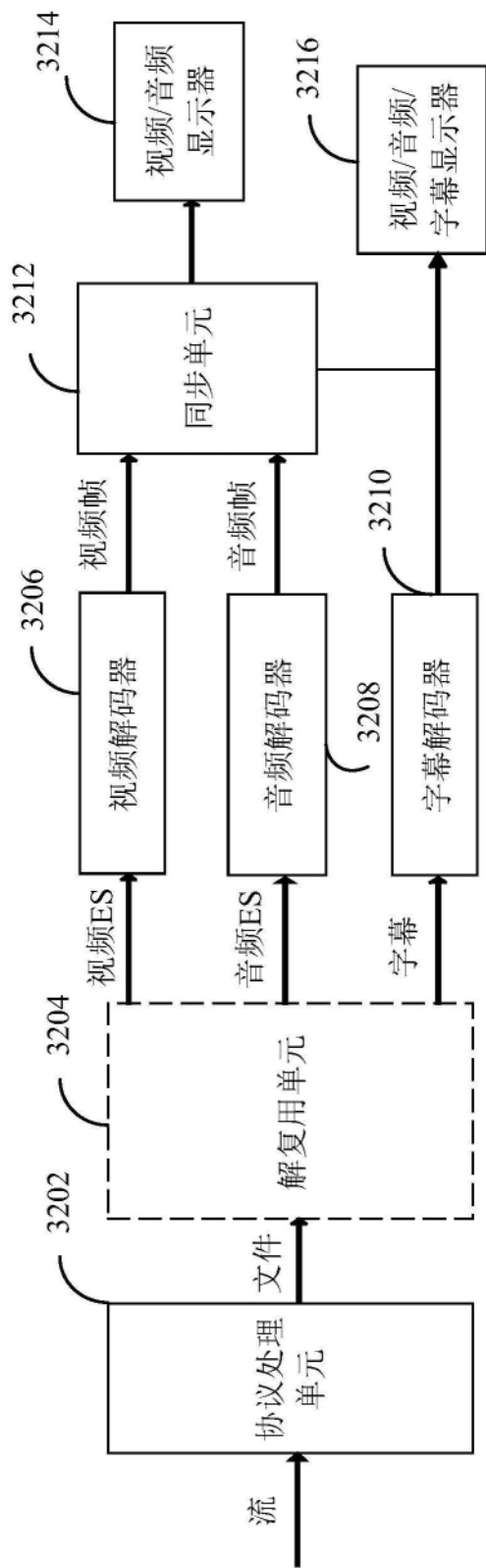


图11

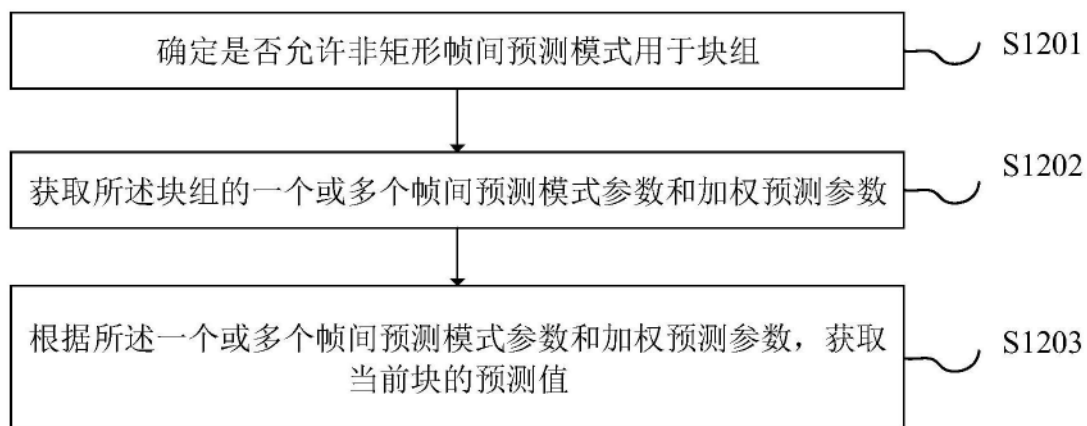


图12

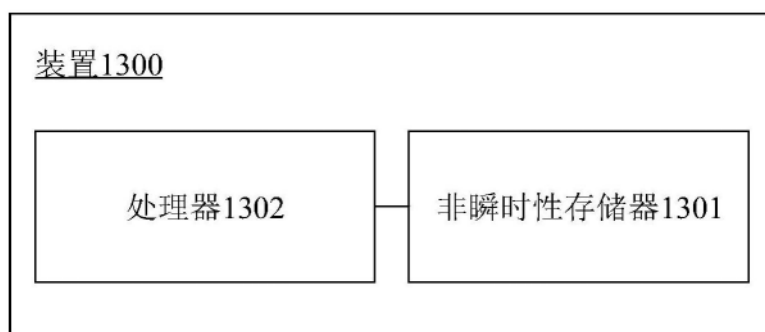


图13

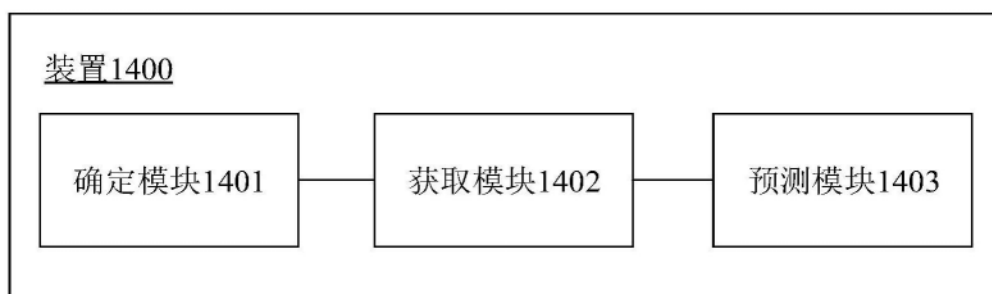


图14

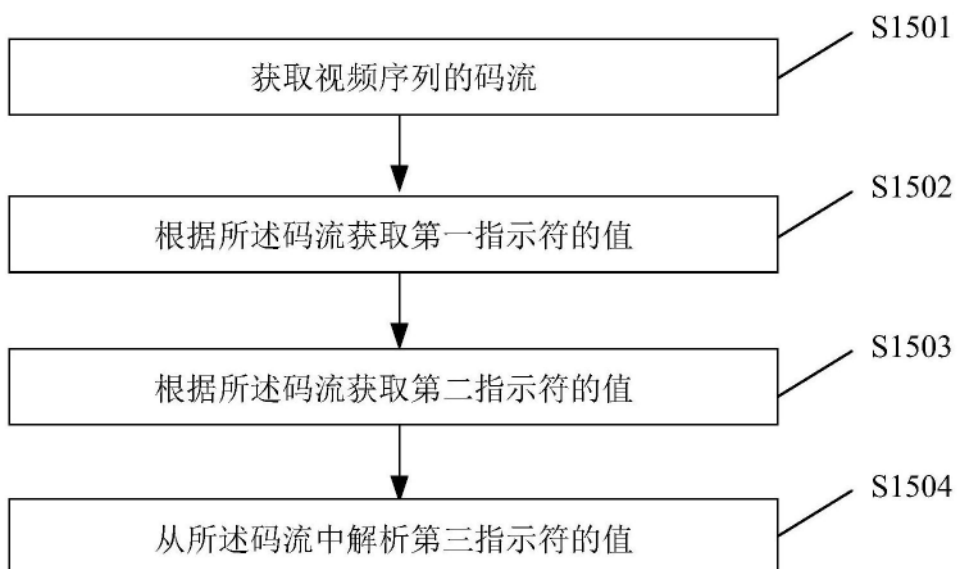


图15

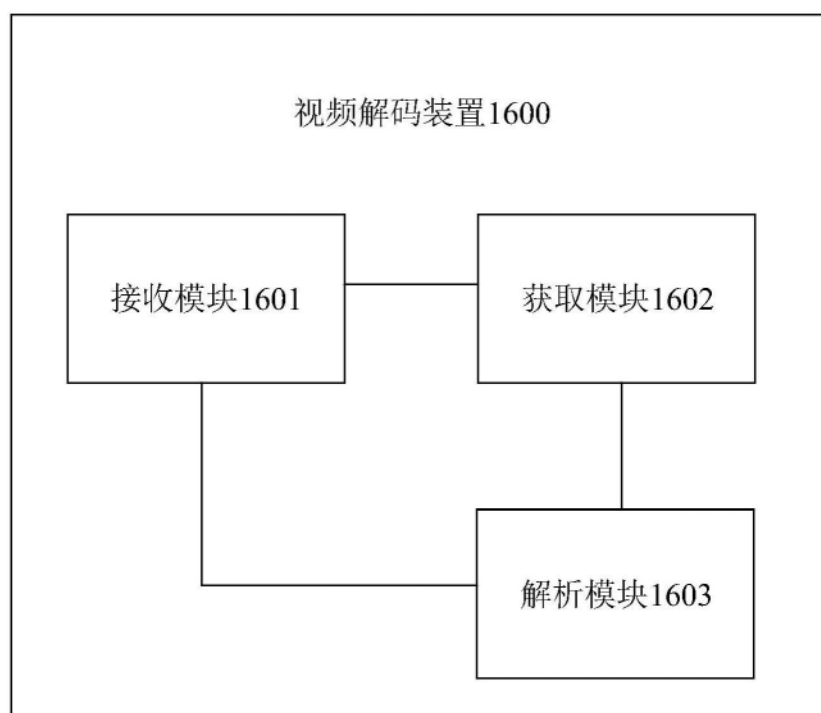


图16