(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 118803250 A
(43)申请公布日 2024.10.18

(21)申请号 202411160571.1

(22)申请日 2019.06.25

(30)优先权数据
2018217333 2018.08.17 AU

(62)分案原申请数据
201980054493.8 2019.06.25

(71)申请人 佳能株式会社
地址 日本

(72)发明人 克里斯托弗·詹姆斯·罗斯沃恩
安德鲁·J·朵莱尔

(74)专利代理机构 北京魏启学律师事务所 11398
专利代理师 陈涛

(51)Int.Cl.
H04N 19/174(2014.01)
H04N 19/122(2014.01)
H04N 19/42(2014.01)
H04N 19/44(2014.01)

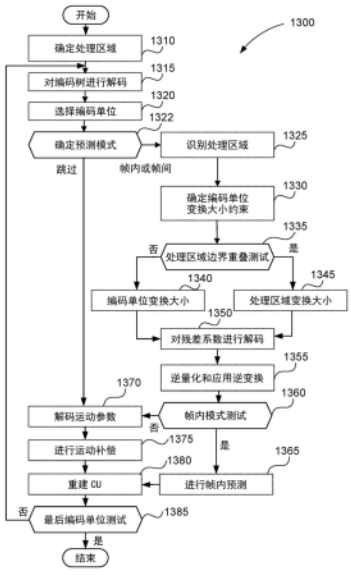
权利要求书3页 说明书33页 附图17页

(54)发明名称

图像解码方法和设备、图像编码方法和设备以及存储介质

(57)摘要

本发明涉及一种图像解码方法和设备、图像编码方法和设备以及存储介质。另外,涉及一种从位流中解码图像帧中的编码单位的系统和方法。该方法包括:从位流确定编码单位的大小;以及将图像帧分割成多个大小相等的处理区域,各个大小相等的处理区域是在对位流进行解码的流水线的单个阶段期间被处理的块。在编码单位与所确定的处理区域之间的边界重叠的情况下,该方法包括:从多个变换大小中选择用于编码单位的变换大小,该变换大小被选择以适配在该编码单位内并且大小与处理区域不同;以及通过对该编码单位中的各个变换单位的残差系数应用逆变换来对该编码单位进行解码,各个变换单位具有所选择的变换大小。



CN 118803250 A

1.一种图像解码方法,用于根据预定方式从位流中解码编码单位,所述方法包括:

从所述位流中解码用于确定编码树单位中的编码单位的信息,其中,在所述预定方式中,所述编码单位的至少一个边能够是128个样本;

在应用第一约束的情况下,以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是32个样本;

在应用第二约束的情况下,以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是64个样本;以及

通过基于所确定的变换单位进行逆变换来对所述编码单位进行解码,

其中,在所述预定方式中,能够将水平边和垂直边中的一个边长于另一个边的形状用作变换单位的形状。

2.根据权利要求1所述的方法,其中,所述编码单位的至少一个边等于128个样本。

3.根据权利要求1所述的方法,其中,三元拆分能够用于确定所述编码树单位中的编码单位。

4.根据权利要求1所述的方法,其中,所述变换单位中的各个变换单位的形状与所述编码单位的形状不同。

5.根据权利要求1所述的方法,其中,所述变换单位中的各个变换单位的长宽比与所述编码单位的长宽比不同。

6.根据权利要求1所述的方法,其中,所述第一约束和所述第二约束被替代地应用。

7.一种图像编码方法,用于根据预定方式将编码单位编码到位流中,所述方法包括:

确定编码树单位中的编码单位,其中,在所述预定方式中,所述编码单位的至少一个边能够是128个样本;

在应用第一约束的情况下,以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是32个样本;

在应用第二约束的情况下,以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是64个样本;以及

通过基于所确定的变换单位进行变换来对所述编码单位进行编码,

其中,在所述预定方式中,能够将水平边和垂直边中的一个边长于另一个边的形状用作变换单位的形状。

8.根据权利要求7所述的方法,其中,所述编码单位的至少一个边等于128个样本。

9.根据权利要求7所述的方法,其中,三元拆分能够用于确定所述编码树单位中的编码单位。

10.根据权利要求7所述的方法,其中,所述变换单位中的各个变换单位的形状与所述编码单位的形状不同。

11.根据权利要求7所述的方法,其中,所述变换单位中的各个变换单位的长宽比与所述编码单位的长宽比不同。

12.根据权利要求7所述的方法,其中,所述第一约束和所述第二约束被替代地应用。

13.一种图像解码设备,用于根据预定方式从位流中解码编码单位,所述设备包括:

解码单元,其被配置为从所述位流中解码用于确定编码树单位中的编码单位的信息,其中,在所述预定方式中,所述编码单位的至少一个边能够是128个样本;以及

确定单元,其被配置为在应用第一约束的情况下,以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是32个样本,

其中,在应用第二约束的情况下,所述确定单元被配置为以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是64个样本,

其中,所述解码单元被配置为通过基于所确定的变换单位进行逆变换来对所述编码单位进行解码,以及

其中,在所述预定方式中,能够将水平边和垂直边中的一个边长于另一个边的形状用作变换单位的形状。

14.一种非暂时性计算机可读存储介质,其存储有用于使计算机执行用于根据预定方式从位流中解码编码单位的图像解码方法的程序,所述方法包括:

从所述位流中解码用于确定编码树单位中的编码单位的信息,其中,在所述预定方式中,所述编码单位的至少一个边能够是128个样本;

在应用第一约束的情况下,以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是32个样本;

在应用第二约束的情况下,以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是64个样本;以及

通过基于所确定的变换单位进行逆变换来对所述编码单位进行解码,

其中,在所述预定方式中,能够将水平边和垂直边中的一个边长于另一个边的形状用作变换单位的形状。

15.一种图像编码设备,用于根据预定方式将编码单位编码到位流中,所述设备包括:

第一确定单元,其被配置为确定编码树单位中的编码单位,其中,在所述预定方式中,所述编码单位的至少一个边能够是128个样本;

第二确定单元,其被配置为在应用第一约束的情况下,以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是32个样本,

其中,在应用第二约束的情况下,所述第二确定单元被配置为以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是64个样本;以及

编码单元,其被配置为通过基于所确定的变换单位进行变换来对所述编码单位进行编码,

其中,在所述预定方式中,能够将水平边和垂直边中的一个边长于另一个边的形状用

作变换单位的形状。

16.一种非暂时性计算机可读存储介质,其存储有用于使计算机执行用于根据预定方式将编码单位编码到位流中的图像编码方法的程序,所述方法包括:

确定编码树单位中的编码单位,其中,在所述预定方式中,所述编码单位的至少一个边能够是128个样本;

在应用第一约束的情况下,以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是32个样本;

在应用第二约束的情况下,以如下方式确定所述编码单位中的变换单位:使得即使在所述编码单位的至少一个边是128个样本的情况下,针对所述变换单位能够选择作为亮度分量的大小的最大大小也是64个样本;以及

通过基于所确定的变换单位进行变换来对所述编码单位进行编码,

其中,在所述预定方式中,能够将水平边和垂直边中的一个边长于另一个边的形状用作变换单位的形状。

图像解码方法和设备、图像编码方法和设备以及存储介质

[0001] (本申请是申请日为2019年6月25日、申请号为2019800544938、发明名称为“对视频样本的变换块编码和解码的方法、设备和系统”的申请的分案申请。)

技术领域

[0002] 本发明通常涉及数字视频信号处理,尤其涉及用于对视频样本的变换块进行编码和解码的方法、设备和系统。本发明还涉及包括记录有用于对视频样本的变换块进行编码和解码的计算机程序的计算机可读介质的计算机程序产品。

背景技术

[0003] 当前存在包括用于传输和存储视频数据的应用的许多视频编码用的应用。还开发了许多视频编码标准并且其它视频编码标准当前正在开发中。视频编码标准化的最新进展已导致形成被称为“联合视频专家组”(JVET)的组。该联合视频专家组(JVET)包括:还已知为“视频编码专家组”(VCEG)的国际电信联盟(ITU)的电信标准化部门(ITU-T)的研究组16、问题6(SG16/Q6)的成员;以及还已知为“运动图片专家组”(MPEG)的ISO/IEC JTC1/SC29/WG11的成员。

[0004] 联合视频专家组(JVET)发布了提案征集(CfP),并在美国圣地亚哥市举行的第10次会议上对答复进行了分析。所提交的答复表明,视频压缩能力明显优于当前最先进的视频压缩标准(即,“高效率视频编码”(HEVC))的视频压缩能力。基于该优异表现,决定开始用以开发命名为“通用视频编码”(VVC)的新视频压缩标准的项目。预计VVC将特别是随着视频格式的能力的增加(例如,具有更高的分辨率和更高的帧频)解决针对甚至更高的压缩性能的持续需求、以及解决针对通过WAN的服务提供(其中,带宽成本相对较高)的日益增长的市场需求。同时,VVC必须可在当代硅工艺中实现,并且在所实现的性能与实现成本之间(例如,在硅面积、CPU处理器负荷、内存利用率和带宽方面)提供可接受的折衷。

[0005] 视频数据包括各自包括一个或多个颜色通道的图像数据的帧序列。通常,需要一个主要颜色通道和两个次要颜色通道。主要颜色通道通常被称为“亮度”通道,并且(一个或多个)次要颜色通道通常被称为“色度”通道。尽管视频数据通常在RGB(红-绿-蓝)颜色空间中显示,但该颜色空间在三个相应分量之间具有高度相关性。编码器或解码器所看到的视频数据表示通常使用诸如YCbCr等的颜色空间。YCbCr将发光度(根据变换方程映射到“亮度”)集中在Y(主要)通道中,并且将色度集中在Cb和Cr(次要)通道中。此外,可以以与亮度通道相比更低的速率(例如,在水平方向上为一半且在垂直方向上为一半(被称为“4:2:0色度格式”))对Cb和Cr通道进行空间采样。

[0006] VVC标准是一种“基于块”的编解码器,其中,帧首先被分割成称为“编码树单位”(CTU)的正方形区域阵列。CTU通常占据相对大的区域,诸如128×128个亮度样本等。然而,各帧的右和底边缘的CTU的区域可能较小。与各CTU相关联的是“编码树”,其定义了将CTU的区域分解为一组区域,也称为“编码单位”(CU)。按特定顺序处理CU以进行编码或解码。由于编码树和4:2:0色度格式的使用,帧中的给定区域与跨颜色通道的同位置块的集合相关联。

亮度块的尺寸为宽度×高度,而对于各色度块,色度块的尺寸为宽度/2×高度/2。给定区域的同位置块的集合通常称为“单位”,例如上述CU以及“预测单位”(PU)和“转换单位”(TU)。

[0007] 尽管针对相同区域,色度块与亮度块的尺寸不同,但是给定“单位”的大小通常就单位的亮度块的尺寸而言来描述。各个块通常由与这些块相关联的单位的类型来识别。例如,“编码块”(CB),“变换块”(TB)和预测块(PB)是针对一个颜色通道的块并且分别与CU、TU和PU相关联。尽管在“单位”和“块”之间有上述区别,但是术语“块”可以用作针对将操作应用于所有颜色通道的帧的区域(area)或区(region)的通用术语。

[0008] 对于各CU,生成帧数据的相应区域的内容(样本值)的预测(PU)(“预测单位”)。此外,形成了在编码器的输入处看到的预测与区域内容之间的差(或在空间域中的“残差”)的表示。各颜色通道的差可以被变换编码为残差系数的序列,从而形成给定CU的一个或多个TU。所应用的变换可以是应用于残差值的各个块的离散余弦变换(DCT)或其它变换。该主要变换是分开应用的,即分两遍进行二维变换。首先通过对块中的各行样本应用一维变换来对块进行变换。然后,通过对部分结果的各列应用一维变换来对部分结果进行变换,以产生基本上对残差样本进行去相关的变换系数的最终块。VVC标准支持各种大小的变换,包括矩形块(各边尺寸为2的幂次方)的变换。量化变换系数以用于将熵编码在位流中。

[0009] VVC标准的实现通常使用流水线将处理分割成阶段序列。各个阶段同步地操作,并且在输出完全处理(即,编码或解码)的块之前,将部分处理的块从一个阶段传递到下一阶段。需要在流水线架构的上下文中有效处理变换后的块,以避免VVC标准的实施成本过高。在存储器消耗方面和在处理“最坏情况”所需的功能模块方面这两者,就流水线阶段完成所需的速度和各阶段处理的数据的大小这两者而言,都需要过高的实施成本。

发明内容

[0010] 本发明的目的是基本上克服或至少改善现有布置的一个或多个缺点。

[0011] 根据本发明的一个方面,提供一种从位流中解码图像帧中的编码单位的方法,所述方法包括:

[0012] 从所述位流确定所述编码单位的大小;

[0013] 将所述图像帧分割成多个大小相等的处理区域,各个大小相等的处理区域是在对所述位流进行解码的流水线的单个阶段期间被处理的块;

[0014] 在所述编码单位与所确定的处理区域之间的边界重叠的情况下,从多个变换大小中选择用于所述编码单位的变换大小,该变换大小被选择以适配在所述编码单位内并且大小与所述处理区域不同;以及

[0015] 通过对所述编码单位中的各个变换单位的残差系数应用逆变换来对所述编码单位进行解码,所述变换单位中的各个变换单位具有所选择的变换大小。

[0016] 根据本发明的另一方面,提供一种非暂时性计算机可读介质,其上存储有计算机程序,以实现从位流中解码图像帧中的编码单位的方法,所述程序包括:

[0017] 用于从所述位流确定所述编码单位的大小的代码;

[0018] 用于将所述图像帧分割成多个大小相等的处理区域的代码,各个大小相等的处理区域是在对所述位流进行解码的流水线的单个阶段期间被处理的块;

[0019] 用于在所述编码单位与所确定的处理区域之间的边界重叠的情况下、从多个变换

大小中选择用于所述编码单位的变换大小的代码,该变换大小被选择以适配在所述编码单位内并且大小与所述处理区域不同;以及

[0020] 用于通过对所述编码单位中的各个变换单位的残差系数应用逆变换来对所述编码单位进行解码的代码,所述变换单位中的各个变换单位具有所选择的变换大小。

[0021] 根据本发明的又一方面,提供一种系统,包括:

[0022] 存储器;以及

[0023] 处理器,其中,所述处理器被配置为执行所述存储器上所存储的代码,以实现从位流中解码图像帧中的编码单位的方法,所述方法包括:

[0024] 从所述位流确定所述编码单位的大小;

[0025] 将所述图像帧分割成多个大小相等的处理区域,各个大小相等的处理区域是在对所述位流进行解码的流水线的单个阶段期间被处理的块;

[0026] 在所述编码单位与所确定的处理区域之间的边界重叠的情况下,从多个变换大小中选择用于所述编码单位的变换大小,该变换大小被选择以适配在所述编码单位内并且大小与所述处理区域不同;以及

[0027] 通过对所述编码单位中的各个变换单位的残差系数应用逆变换来对所述编码单位进行解码,所述变换单位中的各个变换单位具有所选择的变换大小。

[0028] 根据本发明的又一方面,提供一种视频解码器,其被配置为:

[0029] 从位流接收图像帧;

[0030] 从所述位流确定编码单位的大小;

[0031] 将所述图像帧分割成多个大小相等的处理区域,各个大小相等的处理区域是在对所述位流进行解码的流水线的单个阶段期间被处理的块;

[0032] 在所述编码单位与所确定的处理区域之间的边界重叠的情况下,从多个变换大小中选择用于所述编码单位的变换大小,该变换大小被选择以适配在所述编码单位内并且大小与所述处理区域不同;以及

[0033] 通过对所述编码单位中的各个变换单位的残差系数应用逆变换来对所述编码单位进行解码,所述变换单位中的各个变换单位具有所选择的变换大小。

[0034] 根据本发明的又一方面,提供一种从位流中解码图像帧中的编码单位的方法,所述方法包括:

[0035] 从所述位流确定所述编码单位的大小;

[0036] 将所述图像帧分割成多个大小相等的处理区域,各个大小相等的处理区域小于最大可用编码单位大小;

[0037] 从候选运动矢量列表中选择与所述编码单位相对应的运动矢量,选择所述运动矢量包括:(i)在所述编码单位大于或等于所确定的处理区域其中之一的大小的情况下,对合并索引进行解码,或者(ii)在所述编码单位不大于或等于所确定的处理区域其中之一的大小的情况下,对跳过标志进行解码以解码所述合并索引;以及

[0038] 根据针对所述编码单位的所选择的运动矢量对所述编码单位进行解码。

[0039] 根据本发明的又一方面,提供一种非暂时性计算机可读介质,其上存储有计算机程序,以实现用于从位流中解码图像帧中的编码单位的方法,所述程序包括:

[0040] 用于从所述位流确定所述编码单位的大小的代码;

[0041] 用于将所述图像帧分割成多个大小相等的处理区域的代码,各个大小相等的处理区域小于最大可用编码单位大小;

[0042] 用于从候选运动矢量列表中选择与所述编码单位相对应的运动矢量的代码,选择所述运动矢量包括:(i)在所述编码单位大于或等于所确定的处理区域其中之一的大小的情况下,对合并索引进行解码,或者(ii)在所述编码单位不大于或等于所确定的处理区域其中之一的大小的情况下,对跳过标志进行解码以解码所述合并索引;以及

[0043] 用于根据针对所述编码单位的所选择的运动矢量对所述编码单位进行解码的代码。

[0044] 根据本发明的又一方面,提供一种系统,包括:

[0045] 存储器;以及

[0046] 处理器,其中,所述处理器被配置为执行所述存储器上所存储的代码以实现从位流中解码图像帧中的编码单位的方法,所述方法包括:

[0047] 从所述位流确定所述编码单位的大小;

[0048] 将所述图像帧分割成多个大小相等的处理区域,各个大小相等的处理区域小于最大可用编码单位大小;

[0049] 从候选运动矢量列表中选择与所述编码单位相对应的运动矢量,选择所述运动矢量包括:(i)在所述编码单位大于或等于所确定的处理区域其中之一的大小的情况下,对合并索引进行解码,或者(ii)在所述编码单位不大于或等于所确定的处理区域其中之一的大小的情况下,对跳过标志进行解码以解码所述合并索引;以及

[0050] 根据针对所述编码单位的所选择的运动矢量对所述编码单位进行解码。

[0051] 根据本发明的又一方面,提供一种视频解码器,其被配置为:

[0052] 从位流中接收图像帧;

[0053] 从所述位流确定编码单位的大小;

[0054] 将所述图像帧分割成多个大小相等的处理区域,各个大小相等的处理区域小于最大可用编码单位大小;

[0055] 从候选运动矢量列表中选择与所述编码单位相对应的运动矢量,选择所述运动矢量包括:(i)在所述编码单位大于或等于所确定的处理区域其中之一的大小的情况下,对合并索引进行解码,或者(ii)在所述编码单位不大于或等于所确定的处理区域其中之一的大小的情况下,对跳过标志进行解码以解码所述合并索引;以及

[0056] 根据针对所述编码单位的所选择的运动矢量对所述编码单位进行解码。

[0057] 还公开了其它方面。

附图说明

[0058] 现在将参考以下附图和附录描述本发明的至少一个实施例,其中:

[0059] 图1是示出视频编码和解码系统的示意性框图;

[0060] 图2A和2B构成可以实践图1的视频编码和解码系统的其中一个或这两者的通用计算机系统的示意框图;

[0061] 图3是示出视频编码器的功能模块的示意框图;

[0062] 图4是示出视频解码器的功能模块的示意框图;

[0063] 图5是示出通用视频编码的树结构中的块向一个或多个块的可用分割的示意框图;

[0064] 图6是用以在通用视频编码的树结构中实现块向一个或多个块的许可分割的数据流的示意图;

[0065] 图7A和7B示出编码树单位(CTU)向多个编码单位(CU)的示例分割;

[0066] 图8A示出根据流水线架构处理的编码树单位(CTU)的示例序列;

[0067] 图8B示出视频中的帧的示例"随机访问"图片组结构;

[0068] 图9是示出VVC标准的变换大小的图;

[0069] 图10A是示出在编码树的顶层具有三元拆分的编码树单位(CTU)的编码单位的图;

[0070] 图10B是示出与图10A的编码树相关联的替代变换单位的图;

[0071] 图10C是示出与具有在相反方向上的两个三元拆分的编码树相关联的变换单位的图;

[0072] 图10D是示出与具有垂直三元拆分、水平二元拆分和垂直三元拆分的编码树相关联的变换单位的图;

[0073] 图10E是示出与具有两个垂直三元拆分的编码树相关联的变换单位的图;

[0074] 图10F是示出图10E的与具有两个垂直三元拆分的编码树相关联的变换单位的替代的图;

[0075] 图11是用于确定编码树单位的编码树中的编码单位的预测模式的方法的流程图;

[0076] 图12是用于使用变换对编码单位进行编码的方法的流程图,该方法使得能够实现视频编码器的流水线化实现;以及

[0077] 图13是用于使用变换对编码单位进行解码的方法的流程图,该变换大小是根据图12的方法选择的。

具体实施方式

[0078] 在任一个或多个附图中参考具有相同附图标记的步骤和/或特征的情况下,除非出现相反意图,否则这些步骤和/或特征为了本说明书的目的而具有相同的(一个或多个)功能或(一个或多个)操作。

[0079] 图1是示出视频编码和解码系统100的功能模块的示意框图。系统100可以利用将大块或编码单位(CU)隐含地分割成多个较小的块或变换单位(TU),以使得能够在比CTU大小更小的区域(或"流水线处理区域")中处理编码树单位(CTU)。例如,系统100可以将CTU处理为四个象限,各象限可以包含许多CU和/或可以包含跨越多个区域的CU的一部分。

[0080] 系统100包括源装置110和目的地装置130。通信通道120用于从源装置110向目的地装置130通信编码视频信息。在一些配置中,源装置110和目的地装置130中的一个或两个分别可以包括移动电话手机或"智能电话",其中在这种情况下,通信通道120是无线通道。在其它配置中,源装置110和目的地装置130可以包括视频会议设备,其中在这种情况下,通信通道120通常是诸如因特网连接等的有线通道。此外,源装置110和目的地装置130可以包括范围广泛的任意装置,其中这些装置包括支持空中电视广播、有线电视应用、因特网视频应用(包括流传输)、以及在一些计算机可读存储介质(诸如文件服务器中的硬盘驱动器等)上捕获编码视频数据的应用的装置。

[0081] 如图1所示,源装置110包括视频源112、视频编码器114和发送器116。视频源112通常包括所捕获视频帧数据(表示为113)的源,诸如摄像传感器、存储在非暂时性记录介质上的先前捕获到的视频序列、或者来自远程摄像传感器的视频馈送。视频源112也可以是计算机显卡的输出(例如,显示操作系统和在计算装置(例如,平板计算机)上执行的各种应用的视频输出)。可以包括摄像传感器作为视频源112的源装置110的示例包括智能电话、视频摄录机、专业摄像机和网络视频照相机。

[0082] 视频编码器114将来自视频源112的(由箭头113指示的)所捕获帧数据转换(或"编码")成(由箭头115指示的)位流。位流115由发送器116经由通信通道120作为编码视频数据(或"编码视频信息")进行发送。位流115也可以存储在诸如"闪速"存储器或硬盘驱动器等的非暂时性存储装置122中,直到随后通过通信通道120发送或者作为通过通信通道120的发送的代替为止。

[0083] 目的地装置130包括接收器132、视频解码器134和显示装置136。接收器132从通信通道120接收编码视频数据并将所接收到的视频数据作为(由箭头133指示的)位流传递至视频解码器134。然后,视频解码器134将(由箭头135指示的)解码后的帧数据输出至显示装置136。显示装置136的示例包括阴极射线管、液晶显示器(诸如在智能电话、平板计算机、计算机监视器、或者单机型电视机中等)。也可以将源装置110和目的地装置130各自的功能体现在单个装置中,该单个装置的示例包括移动电话手机和平板计算机。

[0084] 尽管以上说明了示例装置,但源装置110和目的地装置130各自通常经由硬件组件和软件组件的组合可以配置在通用计算机系统内。图2A示出这种计算机系统200,该计算机系统200包括:计算机模块201;诸如键盘202、鼠标指示器装置203、扫描器226、可被配置为视频源112的照相机227、以及麦克风280等的输入装置;以及包括打印机215、可被配置为显示装置136的显示装置214、以及扬声器217的输出装置。计算机模块201可以使用外部调制器-解调器(调制解调器)收发器装置216来经由接线221与通信网络220进行通信。可以表示通信通道120的通信网络220可以是广域网(WAN),诸如因特网、蜂窝电信网络或私有WAN等。在接线221是电话线的情况下,调制解调器216可以是传统的"拨号上网"调制解调器。可替代地,在接线221是高容量(例如,线缆或光学的)接线的情况下,调制解调器216可以是宽带调制解调器。还可以使用无线调制解调器来进行向通信网络220的无线连接。收发器装置216可以提供发送器116和接收器132的功能,并且通信通道120可以体现在接线221中。

[0085] 计算机模块201通常包括至少一个处理器单元205和存储器单元206。例如,存储器单元206可以具有半导体随机存取存储器(RAM)和半导体只读存储器(ROM)。计算机模块201还包括多个输入/输出(I/O)接口,其中这多个输入/输出(I/O)接口包括:音频-视频接口207,其连接至视频显示器214、扬声器217和麦克风280;I/O接口213,其连接至键盘202、鼠标203、扫描器226、照相机227以及可选的操纵杆或其它人机接口装置(未示出);以及外部调制解调器216和打印机215所用的接口208。从音频-视频接口207向计算机监视器214的信号通常是计算机显卡的输出。在一些实现中,调制解调器216可以内置于计算机模块201内,例如内置于接口208内。计算机模块201还具有本地网络接口211,其中该本地网络接口211允许计算机系统200经由接线223连接至已知为局域网(LAN)的局域通信网络222。如图2A所示,局域通信网络222还可以经由接线224连接至广域网220,其中该局域通信网络222通常包括所谓的"防火墙"装置或具有相似功能的装置。本地网络接口211可以包括以太网

(EthernetTM)电路卡、蓝牙(BluetoothTM)无线配置或IEEE 802.11无线配置;然而,对于接口211,可以实践多种其它类型的接口。本地网络接口211还可以提供发送器116和接收器132的功能,并且通信通道120也可以体现在局域通信网络222中。

[0086] I/O接口208和213可以提供串行连接和并行连接中的任一个或这两者,其中前者通常根据通用串行总线(USB)标准来实现并且具有相应的USB连接器(未示出)。设置有存储装置209,并且存储装置209通常包括硬盘驱动器(HDD)210。还可以使用诸如软盘驱动器和磁带驱动器等的其它存储装置(未示出)。通常设置有光盘驱动器212以用作数据的非易失性源。可以使用例如光盘(例如,CD-ROM、DVD、蓝光盘(Blu ray DiscTM))、USB-RAM、便携式外部硬盘驱动器和软盘等的便携式存储器装置作为针对计算机系统200的数据的适当源。通常,HDD 210、光盘驱动器212、网络220和222中的任意还可被配置成作为视频源112进行工作、或者作为为了经由显示器214进行再现所要存储的解码视频数据的目的地而进行工作。系统100的源装置110和目的地装置130可以体现在计算机系统200中。

[0087] 计算机模块201的组件205-213通常经由互连总线204并且以得到相关领域技术人员已知的计算机系统200的传统操作模式的方式进行通信。例如,处理器205使用接线218连接至系统总线204。同样,存储器206和光盘驱动器212通过接线219连接至系统总线204。可以实践所述配置的计算机的示例包括IBM-PC和兼容机、Sun SPARCstation、Apple MacTM或相似的计算机系统。

[0088] 在适当或期望的情况下,可以使用计算机系统200来实现视频编码器114和视频解码器134以及以下所述的方法。特别地,可以将视频编码器114、视频解码器134和要说明的方法作为在计算机系统200内可执行的一个或多个软件应用程序233来实现。特别地,利用软件233中的在计算机系统200内执行的指令231(参考图2B)来实现视频编码器114、视频解码器134和所述方法的步骤。可以将软件指令231形成为各自用于进行一个或多个特定任务的一个或多个代码模块。还可以将软件分割成两个单独部分,其中第一部分和相应的代码模块进行所述方法,并且第二部分和相应的代码模块管理第一部分和用户之间的用户界面。

[0089] 例如,可以将软件存储在包括以下所述的存储装置的计算机可读介质中。将软件从计算机可读介质载入计算机系统200,然后由计算机系统200来执行。具有这样的软件的计算机可读介质或者该计算机可读介质上所记录的计算机程序是计算机程序产品。在计算机系统200中使用该计算机程序产品优选地实现了用于实施视频编码器114、视频解码器134和所述方法的有利设备。

[0090] 通常将软件233存储在HDD 210或存储器206中。将该软件从计算机可读介质载入计算机系统200,并且由计算机系统200来执行。因而,例如,可以将软件233存储在光盘驱动器212所读取的光学可读盘存储介质(例如,CD-ROM)225上。

[0091] 在一些实例中,将应用程序233以编码在一个或多个CD-ROM 225上并且经由相应的驱动器212进行读取的方式供给至用户,或者可替代地,可以由用户从网络220或222读取应用程序233。更进一步地,还可以将软件从其它计算机可读介质载入计算机系统200。计算机可读存储介质是指将所记录的指令和/或数据提供至计算机系统200以供执行和/或处理的任何非暂时性有形存储介质。这种存储介质的示例包括软盘、磁带、CD-ROM、DVD、蓝光盘(Blu-ray DiscTM)、硬盘驱动器、ROM或集成电路、USB存储器、磁光盘、或者诸如PCMCIA卡等

的计算机可读卡等,而与这些装置在计算机模块201的内部还是外部无关。还可以参与将软件、应用程序、指令和/或视频数据或编码视频数据提供至计算机模块401的暂时性或非有形计算机可读传输介质的示例包括:无线电或红外线传输通道及向着其它计算机或联网装置的网络接线、以及包括电子邮件发送和网站上所记录的信息等的因特网或内联网。

[0092] 可以执行上述的应用程序233的第二部分和相应的代码模块来实现要绘制或以其它方式呈现在显示器214上的一个或多个图形用户界面(GUI)。通过典型地对键盘202和鼠标203进行操作,计算机系统200的用户和应用可以以在功能上可适用的方式对界面进行操作,以将控制命令和/或输入提供至与这些(一个或多个)GUI相关联的应用。还可以实现在功能上可适用的其它形式的用户界面,诸如利用经由扬声器217所输出的语音提示和经由麦克风280所输入的用户声音命令的音频界面等。

[0093] 图2B是处理器205和“存储器”234的详细示意框图。存储器234表示图2A中的计算机模块201可以访问的(包括HDD 209和半导体存储器206的)所有存储器模块的逻辑聚合。

[0094] 在初始对计算机模块201通电的情况下,执行上电自检(power-on self-test,POST)程序250。通常将POST程序250存储在图2A的半导体存储器206的ROM 249中。有时将诸如存储有软件的ROM 249等的硬件装置称为固件。POST程序250检查计算机模块201内的硬件以确保适当工作,并且通常检查处理器205、存储器234(209,206)和通常还存储在ROM 249中的基本输入-输出系统软件(BIOS)模块251,以进行正确操作。一旦POST程序250成功运行,BIOS 251启动图2A的硬盘驱动器210。启动硬盘驱动器210使得经由处理器205执行驻留在硬盘驱动器210上的引导装入程序252。这样将操作系统253载入RAM存储器206,其中在该RAM存储器206上,操作系统253开始工作。操作系统253是处理器205可执行的系统级应用,以实现包括处理器管理、存储器管理、装置管理、存储管理、软件应用接口和通用用户界面等的各种高级功能。

[0095] 操作系统253管理存储器234(209,206),以确保计算机模块201上运行的各处理或应用具有在不会与分配至其它处理的内存冲突的情况下执行的充足内存。此外,必须适当使用图2A的计算机系统200中可用的不同类型的存储器,以使得各处理可以高效地运行。因此,聚合存储器234并不意图例示如何分配存储器的特定分段(除非另外说明),而是提供计算机系统200可访问的存储器的概述图以及如何使用该存储器。

[0096] 如图2B所示,处理器205包括多个功能模块,其中这多个功能模块包括控制单元239、算术逻辑单元(ALU)240和有时称为高速缓冲存储器的本地或内部存储器248。高速缓冲存储器248在寄存器区段中通常包括多个存储寄存器244-246。一个或多个内部总线241从功能上使这些功能模块相互连接。处理器205通常还具有用于使用接线218经由系统总线204与外部装置进行通信的一个或多个接口242。存储器234使用接线219连接至总线204。

[0097] 应用程序233包括可以包含条件分支指令和循环指令的指令序列231。程序233还可以包括执行程序233时所使用的数据232。将指令231和数据232分别存储在存储器位置228、229、230和235、236、237中。根据指令231和存储器位置228-230的相对大小,如存储器位置230中示出的指令所描述的,可以将特定指令存储在单个存储器位置中。可选地,如存储器位置228和229中示出的指令段所描述的,可以将指令分割成各自被存储在单独的存储器位置的多个部分。

[0098] 通常,向处理器205赋予一组指令,其中在该处理器205内执行该组指令。处理器

205等待随后输入,其中处理器205通过执行另一组指令来对该随后输入作出反应。可以从多个源中的一个或多个源提供各输入,其中该输入包括输入装置202、203中的一个或多个所生成的数据、从外部源经由网络220、202其中之一所接收到的数据、从存储装置206、209其中之一所检索到的数据或者从插入相应的读取器212内的存储介质225所检索到的数据(所有这些均在图2A中示出)。执行一组指令在一些情况下可能会导致输出数据。执行还可能涉及将数据或变量存储至存储器234。

[0099] 视频编码器114、视频解码器134和所述方法可以使用存储器234内的相应存储器位置255、256、257中所存储的输入变量254。视频编码器114、视频解码器134和所述方法产生存储器234内的相应存储器位置262、263、264中所存储的输出变量261。可以将中间变量258存储在存储器位置259、260、266和267中。

[0100] 参考图2B的处理器205,寄存器244、245、246、算术逻辑单元(ALU)240和控制单元239一起工作以进行微操作序列,其中这些微操作序列是针对构成程序233的指令集中的各指令进行“提取、解码和执行”周期所需的。各提取、解码和执行周期包括:

[0101] (a)提取操作,用于从存储器位置228、229、230提取或读取指令231;

[0102] (b)解码操作,其中在该解码操作中,控制单元239判断提取了哪个指令;以及

[0103] (c)执行操作,其中在该执行操作中,控制单元239和/或ALU 240执行该指令。

[0104] 之后,可以执行针对下一指令的进一步提取、解码和执行周期。同样,可以进行存储周期,其中通过该存储周期,控制单元239将值存储至或写入存储器位置232。

[0105] 要说明的图12和图13的方法中的各步骤或子处理与程序233的一个或多个区段相关联,并且通常通过处理器205中的寄存器部244、245、247、ALU 240和控制单元239一起工作以针对程序233的所述分段的指令集中的各指令进行提取、解码和执行周期,来进行该步骤或子处理。

[0106] 图3是示出视频编码器114的功能模块的示意框图。图4是示出视频解码器134的功能模块的示意框图。通常,数据以样本或系数的组(诸如块向固定大小的子块的分割等)或者作为阵列在视频编码器114和视频解码器134内的功能模块之间传递。如图2A和2B所示,可以使用通用计算机系统200来实现视频编码器114和视频解码器134,其中可以利用计算机系统200内的专用硬件、利用计算机系统200内可执行的软件(诸如驻留在硬盘驱动器205上并且由处理器205控制其执行的软件应用程序233的一个或多个软件代码模块等),来实现各种功能模块。可替代地,可以利用在计算机系统200内可执行的专用硬件和软件的组合来实现视频编码器114和视频解码器134。可以可替代地在诸如进行所述方法的功能或子功能的一个或多个集成电路等的专用硬件中实现视频编码器114、视频解码器134和所述方法。这种专用硬件可以包括图形处理单元(GPU)、数字信号处理器(DSP)、专用标准产品(ASSP)、专用集成电路(ASIC)、现场可编程门阵列(FPGA)或者一个或多个微处理器和关联存储器。特别地,视频编码器114包括模块310-386,并且视频解码器134包括模块420-496,其中这些模块各自可被实现为软件应用程序233的一个或多个软件代码模块。

[0107] 尽管图3的视频编码器114是通用视频编码(VVC)视频编码流水线的示例,但也可以使用其它视频编解码器来进行这里所述的处理阶段。视频编码器114接收诸如一系列帧(各帧包括一个或多个颜色通道)等的所捕获帧数据113。块分区器310首先将帧数据113分割成CTU,CTU的形状通常为正方形,并且被配置成使得使用CTU的特定大小。例如,CTU的大

小可以是64×64、128×128或256×256个亮度样本。块分区器310进一步将各CTU分割成一个或多个CU,其中CU具有可以包括正方形和非正方形长宽比这两者的各种大小。然而,在VVC标准中,CU、PU和TU总是具有2的幂次方的边长。因而,从块分区器310输出当前CU(表示为312),从而根据对CTU的一个或多个块的迭代、根据CTU的编码树而前进。以下参考图5和6来进一步说明用于将CTU分区为CU的选项。

[0108] 从帧数据113的第一次分割得到的CTU可以是按光栅扫描顺序扫描的,并可以被分组成一个或多个“条带(slice)”。条带可以是“帧内”(或“I”)条带。帧内条带(I条带)指示条带中的每个CU都是帧内预测的。可选地,条带可以是单预测或双预测的(分别为“P”或“B”条带),分别指示条带中的单预测和双预测的附加可用性。由于帧数据113通常包括多个颜色通道,CTU和CU与来自同从块分区器310的操作定义的块区域重叠的所有颜色通道的样本相关联。CU针对帧数据113的各颜色通道包括一个编码块(CB)。由于色度通道与亮度通道的采样速率潜在地不同,因此色度通道的CB的尺寸可能不同于亮度通道的CB的尺寸。当使用4:2:0色度格式时,CU的色度通道的CB的尺寸是CU的亮度通道的CB的宽度和高度的一半。

[0109] 对于各CTU,视频编码器114在两个阶段中操作。在第一阶段(称为“搜索”阶段),块分区器310测试编码树的各种潜在配置。编码树的各个潜在配置具有关联的“候选”CU。第一阶段涉及测试各种候选CU,以选择提供高压缩效率和低失真的CU。该测试通常涉及拉格朗日优化,由此基于速率(编码成本)和失真(关于输入帧数据113的误差)的加权组合来评价候选CU。选择“最佳”候选CU(具有最低速率/失真的CU)以用于随后编码在位流115中。候选CU的评价中包括如下选项:将CU用于给定区域,或者根据各种拆分选项来拆分该区域并利用其它CU来编码各个较小所得区域或更进一步拆分区域。结果,在搜索阶段中选择CU和编码树本身这两者。

[0110] 视频编码器114针对各CU(例如,CU 312)产生由箭头320指示的预测单位(PU)。PU 320是关联的CU 312的内容的预测。减法器模块322产生PU 320和CU 312之间的表示为324的差(或“残差”,其是指差在空间域中)。差324是PU 320和CU 312中的相应样本之间的块大小差。差324被变换、量化和表示为由箭头336指示的变换单位(TU)。PU 320和关联的TU 336通常被选择为多个可能的候选CU中的“最佳”CU。

[0111] 候选编码单位(CU)是针对关联的PU和所得到的残差从视频编码器114可用的预测模式其中之一得到的CU。各候选CU得到一个或多个相应的TU,如在下文中参考图10A至12所述。TU 336是差324的量化和变换表示。当与解码器114中的预测PU组合时,TU 336以位流中的附加信令为代价来减少解码CU和原始CU 312之间的差。

[0112] 因而,各候选编码单位(CU)(即预测单位(PU)与变换单位(TU)的组合)具有关联的编码成本(或“速率”)和关联的差(或“失真”)。速率通常是以位为单位测量的。CU的失真通常被估计为样本值的差,诸如绝对差和(SAD)或平方差和(SSD)等。模式选择器386使用差324来确定从各候选PU得到的估计,以确定帧内预测模式(由箭头388表示)。可以以与残差的熵编码相比明显更低的成本来进行与各候选预测模式相关联的编码成本和相应的残差编码的估计。因此,可以评价多个候选模式,以确定率失真意义上的最佳模式。

[0113] 确定最佳模式通常是使用拉格朗日优化的变形来实现的。选择帧内预测模式388通常涉及确定从应用特定帧内预测模式得到的残差数据的编码成本。可以通过使用“绝对变换差和”(SATD)来近似编码成本,由此使用诸如Hadamard变换等的相对简单的变换来获

得估计变换残差成本。在使用相对简单的变换的一些实现中,从简化估计方法得到的成本与否则将从完整评价确定的实际成本单调相关。在具有单调相关的估计成本的实现中,可以使用该简化估计方法来作出相同的决定(即,帧内预测模式),其中视频编码器114的复杂度降低。为了允许估计成本和实际成本之间的关系的可能非单调性,可以使用该简化估计方法来生成最佳候选的列表。例如,可以从可用于残差数据的编码的其它模式决策得到非单调性。最佳候选的列表可以具有任意数量。可以使用最佳候选来进行更完整的搜索,以建立用于对各个候选的残差数据进行编码的最模式选择,从而允许最终选择帧内预测模式以及其它模式决策。

[0114] 其它模式决策包括跳过正向变换的能力(被称为“变换跳过”)。跳过变换适合于缺乏足够的相关性来经由作为变换基础函数的表达式降低编码成本的残差数据。诸如相对简单的计算机生成图形等的某些类型的内容可以表现出类似的行为。对于“跳过的变换”,即使不进行变换本身,仍会对残差系数进行编码。

[0115] 可以采用拉格朗日或类似的优化处理来进行(利用块分区器310的)CTU向CU的分区的选择以及从多个可能性中的最佳预测模式的选择这两者。通过将候选模式的拉格朗日优化处理应用在模式选择器模块386中,选择成本测量最低的帧内预测模式作为最佳模式。最佳模式是所选择的帧内预测模式388,并且也由熵编码器338编码在位流115中。通过模式选择器模块386的操作对帧内预测模式388的选择扩展到块分区器310的操作。例如,帧内预测模式388的选择的候选可以包括可应用于给定块的模式和可应用于整体与给定块同位置的多个更小块的附加模式。在包括可应用于给定块和更小同位置块的模式的情况下,候选的选择的处理隐含地也是用于确定CTU向CU的最佳层级分解的处理。

[0116] 在视频编码器114的操作的第二阶段(称为“编码”阶段)中,在视频编码器114中进行对所选择的编码树(以及因而对各个所选择的CU)的迭代。如这里进一步描述的,在迭代中,将CU编码在位流115中。

[0117] 熵编码器338支持句法元素的可变长度编码和句法元素的算术编码这两者。使用上下文自适应二进制算术编码处理来支持算术编码。算术编码的句法元素由一个或多个“bin(二进制文件)”的序列组成。与位一样,bin的值为“0”或“1”。然而,bin未作为离散位编码在位流115中。bin具有关联预测(或“可能”或“最大概率”)值和关联概率(被称为“上下文”)。当要编码的实际bin与预测值匹配时,对“最大概率符号”(MPS)进行编码。对最大概率符号进行编码在消耗位方面相对便宜。当要编码的实际bin与可能值不匹配时,对“最小概率符号”(LPS)进行编码。对最小概率符号进行编码在消耗位方面具有相对高的成本。bin编码技术使得能够使“0”vs“1”的概率歪斜的bin进行高效编码。对于具有两个可能值的句法元素(即,“flag(标志)”),单个bin就足够了。对于具有许多可能值的句法元素,需要bin的序列。

[0118] 可以基于该序列中的较前bin的值来确定该序列中的较后bin的存在。另外,各bin可以与多于一个的上下文相关联。可以根据句法元素中的较前bin和相邻句法元素的bin值(即,来自相邻块的bin值)等来选择特定上下文。每次对上下文编码bin进行编码时,以反映出新bin值的方式来更新针对该bin(如果有)选择的上下文。如此,二进制算术编码方案被认为是自适应的。

[0119] 视频编码器114也支持缺少上下文的bin(“旁路bin”)。假定“0”和“1”之间的等概

率分布来对旁路bin进行编码。因而,各个bin占据位流115中的一位。不存在上下文节省了内存并降低了复杂度,因而使用特定bin的值的分布未歪斜的旁路bin。采用上下文和自适应的熵编码器的一个示例在本领域中被称为CABAC(上下文自适应二进制算术编码器),并且在视频编码中采用了该编码器的许多变体。

[0120] 熵编码器338使用上下文编码的和旁路编码的bin的组合来对帧内预测模式388进行编码。通常,在视频编码器114中生成“最大概率模式”的列表。最大概率模式的列表通常是诸如三个或六个模式等的固定长度,并且可以包括在早期块中遇到的模式。上下文编码bin对指示帧内预测模式是否是最大概率模式其中之一的flag进行编码。如果帧内预测模式388是最大概率模式其中之一,则使用旁路编码bin对进一步信令进行编码。例如,编码后的进一步信令使用截断一元bin串来指示哪个最大概率模式与帧内预测模式388相对应。否则,将帧内预测模式388编码为“剩余模式”。编码为剩余模式使用诸如固定长度代码等的替代句法,也使用旁路编码bin来编码,以表示除存在于最大概率模式列表中的帧内预测模式以外的帧内预测模式。

[0121] 多路复用器模块384根据选自各候选CU的所测试的预测模式中的所确定的最佳帧内预测模式388来输出PU 320。候选预测模式无需包括视频编码器114所支持的每个可想到的预测模式。

[0122] 预测模式大致分为两个类别。第一个类别是“帧内预测(intra-frame prediction)”(也称为“帧内预测(intra prediction)”)。在帧内预测中,生成块的预测,并且生成方法可以使用从当前帧中获得的其它样本。对于帧内预测PU,有可能将不同的帧内预测模式用于亮度和色度,因而主要就对PB而非PU的操作而言来描述帧内预测。

[0123] 预测模式的第二个类别是“帧间预测(inter-frame prediction)”(也称为“帧间预测(inter prediction)”)。在帧间预测中,使用来自按位流中的编码帧的顺序在当前帧之前的一个或两个帧的样本来产生块的预测。位流中的编码帧的顺序可能不同于当捕获或显示时的帧的顺序。当一个帧用于预测时,该块被称为“单预测”并且具有两个关联的运动矢量。当两个帧用于预测时,该块被称为“双预测”并且具有两个关联的运动矢量。对于P条带,各CU可以被帧内预测或单预测。对于B条带,各CU可以被帧内预测、单预测或双预测。通常使用“图片组”结构来编码帧,从而实现帧的时间层级结构。帧的时间层级结构允许帧参考按显示帧的顺序的先前和后续的图片。图像按确保满足解码各帧的依赖关系所必需的顺序来编码。

[0124] 帧间预测的子类别被称为“跳过模式”。帧间预测和跳过模式被描述为两个不同的模式。然而,帧间预测模式和跳过模式这两者涉及参考来自先前帧的样本的块的运动矢量。帧间预测涉及编码运动矢量增量(delta),从而指定相对于运动矢量预测器的运动矢量。从利用“合并索引”选择的一个或多个候选运动矢量的列表中获得运动矢量预测器。编码运动矢量增量提供相对于所选择的运动矢量预测的空间偏移。帧间预测还使用位流133中的编码残差。跳过模式仅使用索引(也称为“合并索引”)来从若干运动矢量候选中选择一个。在没有任何进一步信令的情况下使用所选择的候选。而且,跳过模式不支持对任何残差系数进行编码。当使用跳过模式时不存在编码残差系数意味着不需要进行跳过模式的变换。因此,跳过模式通常不会导致流水线处理问题。流水线处理问题可能是针对帧内预测CU和帧间预测CU的情况。由于跳过模式的有限信令,当相对高质量的参考帧可用时,跳过模式对于

实现非常高的压缩性能是有用的。随机访问图片组结构的较高时间层中的双预测CU通常具有准确反映基础运动的高质量参考图片和运动矢量候选。结果,跳过模式对于将参考图8B描述的随机访问图片组结构中的较高时间层处的帧中的双预测块是有用的。

[0125] 根据运动矢量和参考图片索引来选择样本。运动矢量和参考图片索引适用于所有颜色通道,因而主要就对PU而非PB的操作来描述帧间预测。在各类别内(即,帧内和帧间预测),可以应用不同技术来生成PU。例如,帧内预测可以使用来自先前重建样本的相邻行和列的值组合方向来根据规定的滤波和生成处理生成PU。可替代地,可以使用少量参数来描述PU。帧间预测方法在运动参数的数量及其精度上可能变化。运动参数通常包括参考帧索引(其指示将使用来自参考帧列表的哪些参考帧加上参考帧各自的空间平移),但可以包括更多帧、特殊帧、或者诸如缩放和旋转等的复杂仿射参数。另外,可以应用预先确定的运动细化处理以基于参考样本块生成密集运动估计。

[0126] 在确定并选择了最佳PU 320、并在减法器322处从原始样本块中减去了PU 320的情况下,获得编码成本最低的残差(表示为324)并对该残差进行有损压缩。有损压缩处理包括变换、量化和熵编码的步骤。变换模块326对差324应用正向变换,从而将差324转换到频率域并且产生由箭头332表示的变换系数。正向变换通常是可分离的,从而对各块的一组行然后对一组列进行变换。通过首先对块的各行应用一维变换以产生部分结果然后对部分结果的各列应用一维变换以产生最终结果来进行各组行和列的变换。

[0127] 变换系数332被传递到量化器模块334。在模块334处,进行根据“量化参数”的量化以产生由箭头336表示的残差系数。量化参数对于给定TB是恒定的,因而得到针对TB的残差系数的产生的均匀缩放。通过应用“量化矩阵”也可以实现非均匀缩放,由此对各残差系数应用的缩放因子是从量化参数和大小通常等于TB的大小的缩放矩阵中的相应条目的组合导出的。残差系数336被供给至熵编码器338以编码在位流115中。通常,根据扫描模式对各TB的残差系数以及TU的至少一个有效残差系数进行扫描以生成值的有序列表。扫描模式通常将TB扫描为4×4个“子块”的序列,从而按4×4的残差系数集合的粒度提供常规扫描操作,其中子块的配置取决于TB的大小。另外,预测模式388和相应块分区也编码在位流115中。

[0128] 如上所述,视频编码器114需要访问与在视频解码器134中看到的帧表示相对应的帧表示。因而,残差系数336也由去量化器模块340进行逆量化以产生由箭头342表示的逆变换系数。逆变换系数342通过逆变换模块348以产生TU的由箭头350表示的残差样本。求和模块352将残差样本350和PU 320相加以产生CU的重建样本(由箭头354指示)。

[0129] 重建样本354被传递到参考样本高速缓冲存储器356和环内滤波器模块368。通常使用ASIC上的静态RAM实现(因此避免了昂贵的片外存储器访问)的参考样本高速缓冲存储器356提供了满足用于为帧中的后续CU生成帧内PB的依赖关系所需的最小样本存储。最小依赖关系通常包括沿着一行CTU的底部的样本的“线缓冲器”,以供下一行CTU以及范围由CTU的高度设置的列缓冲使用。参考样本高速缓冲存储器356将参考样本(由箭头358表示)供给至参考样本滤波器360。样本滤波器360应用平滑操作以产生滤波参考样本(由箭头362指示)。滤波参考样本362由帧内预测模块364使用以产生由箭头366表示的样本的帧内预测块。对于各候选帧内预测模式,帧内预测模块364产生样本块即366。

[0130] 环内滤波器模块368对重建样本354应用数个滤波阶段。滤波阶段包括“去块滤波

器”(DBF),该DBF应用与CU边界对齐的平滑化,以减少由不连续而产生的伪影。环内滤波器模块368中存在的另一滤波阶段是“自适应环路滤波器”(ALF),该ALF应用基于Wiener的自适应滤波器以进一步降低失真。环内滤波器模块368中的另一可用滤波阶段是“样本自适应偏移”(SAO)滤波器。SAO滤波器通过首先将重建样本分类为一个或多个类别、并且根据所分配的类别在样本级别应用偏移来工作。

[0131] 从环内滤波器模块368输出由箭头370表示的滤波样本。滤波样本370被存储在帧缓冲器372中。帧缓冲器372通常具有存储数个(例如,多达16个)图片的容量,因而存储在存储器206中。由于所需的大存储器消耗,因此帧缓冲器372通常不使用片上存储器来存储。如此,对帧缓冲器372的访问在内存带宽方面是昂贵的。帧缓冲器372将参考帧(由箭头374表示)提供至运动估计模块376和运动补偿模块380。

[0132] 运动估计模块376估计多个“运动矢量”(表示为378),其各自是相对于当前CU的位置的笛卡尔空间偏移,从而参考帧缓冲器372中的参考帧其中之一中的块。针对各运动矢量产生参考样本的滤波块(表示为382)。滤波参考样本382形成可供模式选择器386的潜在选择用的进一步候选模式。此外,对于给定CU,PU 320可以使用一个参考块(“单预测”)形成,或者可以使用两个参考块(“双预测”)形成。对于所选择的运动矢量,运动补偿模块380根据支持运动矢量中的子像素精度的滤波处理来产生PU 320。如此,运动估计模块376(其对许多候选运动矢量进行操作)与运动补偿模块380(其仅对所选择的候选进行操作)相比可以进行简化滤波处理,以实现降低的计算复杂度。

[0133] 尽管参考通用视频编码(VVC)说明了图3的视频编码器114,但其它视频编码标准或实现也可以采用模块310-386的处理阶段。帧数据113(和位流115)也可以从存储器206、硬盘驱动器210、CD-ROM、蓝光盘(Blue-ray diskTM)或其它计算机可读存储介质中读取(或者被写入存储器206、硬盘驱动器210、CD-ROM、蓝光盘或其它计算机可读存储介质)。另外,帧数据113(和位流115)可以从外部源(诸如连接至通信网络220的服务器或者射频接收器等)接收(或者被发送至该外部源)。

[0134] 在图4中示出视频解码器134。尽管图4的视频解码器134是通用视频编码(VVC)视频解码流水线的示例,但其它视频编解码器也可用于进行本文所述的处理阶段。如图4所示,位流133被输入到视频解码器134。位流133可以从存储器206、硬盘驱动器210、CD-ROM、蓝光盘或其它非暂时性计算机可读存储介质中读取。可替代地,位流133可以从外部源(诸如连接至通信网络220的服务器或者射频接收器等)接收。位流133包含表示要解码的所捕获帧数据的编码句法元素。

[0135] 位流133被输入到熵解码器模块420。熵解码器模块420从位流133中提取句法元素并将句法元素的值传递到视频解码器134中的其它模块。熵解码器模块420应用CABAC算法以从位流133解码句法元素。解码得到的句法元素用于重建视频解码器134内的参数。参数包括残差系数(由箭头424表示)以及诸如帧内预测模式等的模式选择信息(由箭头458表示)。模式选择信息还包括诸如运动矢量等的信息、以及各CTU向一个或多个CU的分区。参数用于通常与来自先前解码的CU的样本数据组合生成PU。

[0136] 残差系数424被输入到去量化器模块428。去量化器模块428对残差系数424进行逆量化(或“缩放”),以根据量化参数创建重建变换系数(由箭头440表示)。如果在位流133中指示使用非均匀逆量化矩阵,则视频解码器134从位流133读取量化矩阵作为缩放因子序

列,并且根据将缩放因子排列成矩阵。逆缩放将量化矩阵与量化参数组合使用以创建重建中间变换系数。

[0137] 重建变换系数440被传递到逆变换模块444。模块444将系数从频率域变换回到空间域。TB有效地基于有效残差系数和非有效残差系数值。模块444的操作的结果是由箭头448表示的残差样本的块。残差样本448在大小上等于相应的CU。残差样本448被供给至求和模块450。在求和模块450处,将残差样本448加到表示为452的解码PU,以产生由箭头456表示的重建样本的块。重建样本456被供给至重建样本高速缓冲存储器460和环内滤波模块488。环内滤波模块488产生表示为492的帧样本的重建块。帧样本492被写入帧缓冲器496。

[0138] 重建样本高速缓冲存储器460以类似于视频编码器114的重建样本高速缓冲存储器356的方式操作。重建样本高速缓冲存储器460在无存储器206的情况下(例如,通过作为代替使用通常是片上存储器的数据232)为对后续CU进行帧内预测所需的重建样本提供存储。由箭头464表示的参考样本是从重建样本高速缓冲存储器460获得的,并被供给至参考样本滤波器468以产生由箭头472表示的滤波参考样本。滤波参考样本472被供给至帧内预测模块476。模块476根据在位流133中表示的并由熵解码器420解码的帧内预测模式参数458,产生由箭头480表示的帧内预测样本的块。

[0139] 当针对当前CU在位流133中指示帧内预测时,帧内预测样本480经由多路复用器模块484形成解码PU 452。

[0140] 当针对当前CU在位流133中指示帧间预测时,运动补偿模块434使用运动矢量和参考帧索引从帧缓冲器496中选择和过滤样本块,来产生表示为438的帧间预测样本的块。样本块498是从帧缓冲器496中所存储的先前解码帧获得的。为了进行双预测,产生两个样本块并将这两个样本块混合在一起以产生解码PU 452的样本。帧缓冲器496由来自环内滤波模块488的滤波块数据492填充。与视频编码器114的环内滤波模块368一样,环内滤波模块488应用任何、至少或全部的DBF、ALF和SAO滤波操作。环内滤波模块368从重建样本456产生滤波块数据492。

[0141] 图5是示出通用视频编码的树结构中的区域向一个或多个子区域的可用分割或拆分的集合500的示意框图。如参考图3所述,集合500中示出的分割可供编码器114的块分区器310利用以根据如通过拉格朗日优化所确定的编码数将各CTU分割成一个或多个CU。

[0142] 尽管集合500仅示出将正方形区域分割成其它可能非正方形子区域,但应当理解,图500正示出潜在分割、而并未要求包含区域为正方形。如果包含区域为非正方形,则根据包含块的长宽比来对从分割得到的块的尺寸进行缩放。一旦区域未被进一步拆分,即,在编码树的叶节点处,CU占据该区域。利用块分区器310的CTU向一个或多个CU的特定子分割被称为CTU的“编码树”。将区域子分割成子区域的处理必须在所得到的子区域达到最小CU大小时终止。除了将CU约束为禁止小于例如4×4的大小之外,将CU约束为具有为四的最小宽度或高度。就宽度和高度或者就宽度或高度这两而言其它最小值也是可能的。子分割处理也可以在分解的最深层之前终止,从而得到大于最小CU大小的CU。有可能不发生拆分,从而得到占据整个CTU的单个CU。占据整个CTU的单个CU是最大可用编码单位大小。此外,未发生拆分的CU大于处理区域大小。作为在编码树的最高层处的二元或三元拆分的结果,诸如64×128、128×64、32×128和128×32等的CU大小是可能的,其各自也大于处理区域大小。参考图10A-10F进一步描述大于处理区域大小的CU的示例。

[0143] 在不存在进一步子分割的情况下,在编码树的叶节点处存在CU。例如,叶节点510包含一个CU。在编码树的非叶节点处,存在向两个或更多个其它节点的拆分,其中各节点可以包含叶节点(因而一个CU)、或者包含向更小区域的进一步拆分。

[0144] 如图5所示,四叉树拆分512将包含区域分割成四个大小相等的区域。与HEVC相比,通用视频编码(VVC)通过添加水平二元拆分514和垂直二元拆分516实现了附加的灵活性。拆分514和516各自将包含区域分割成两个大小相等的区域。分割沿着包含块内的水平边界(514)或垂直边界(516)。

[0145] 在通用视频编码中通过添加三元水平拆分518和三元垂直拆分520实现了进一步的灵活性。三元拆分518和520将块分割成沿着包含区域宽度或高度的1/4和3/4在水平方向(518)或垂直方向(520)上形成界限的三个区域。四叉树、二叉树和三叉树的组合被称为“QTBTTT”或可替代地被称为多叉树(MT)。

[0146] 与仅支持四叉树、因而仅支持正方形块的HEVC相比,QTBTTT特别是考虑到二叉树和/或三叉树拆分的可能递归应用而得到更多可能的CU大小。可以通过约束拆分选项以消除将得到小于四个样本的块宽度或高度或者将得到不是四个样本的倍数的拆分来降低异常(例如,非正方形)块大小的可能性。通常,约束将在考虑亮度样本时适用。然而,约束也可单独应用于色度通道的块,从而可能针对亮度vs色度(例如,在帧数据采用4:2:0色度格式时)得到不同的最小块大小。各拆分产生具有相对于包含区域不变的、二等分的或四等分的边尺寸的子区域。然后,由于CTU大小为2的幂次方,因此所有CU的边尺寸也为2的幂次方。

[0147] 图6是示出在通用视频编码中使用的QTBTTT(或“编码树”)结构的数据流600的示意流程图。将QTBTTT结构用于各CTU以定义CTU向一个或多个CU的分割。各CTU的QTBTTT结构由视频编码器114中的块分区器310确定,并被编码到位流115中或者由视频解码器134中的熵解码器420从位流133解码。根据图5所示的分割,数据流600进一步表现可供块分区器310将CTU分割成一个或多个CU用的许可组合的特征。

[0148] 从层级结构的顶层开始、即在CTU处,首先进行零个或更多个四叉树分割。具体地,由块分区器310作出四叉树(QT)拆分决策610。610处的决策返回“1”符号,这表明决定根据四叉树拆分512将当前节点拆分成四个子节点。结果是诸如在620处等生成四个新节点,并且针对各新节点,递归回到QT拆分决策610。各新节点均是按光栅(或Z扫描)顺序考虑的。可替代地,如果QT拆分决策610指示不进行进一步拆分(返回“0”符号),则四叉树分区停止,并且随后考虑多树(MT)拆分。

[0149] 首先,由块分区器310作出MT拆分决策612。在612处,指示进行MT拆分的决策。在决策612处返回“0”符号,这表明将不进行节点向子节点的进一步拆分。如果将不进行节点的进一步拆分,则节点是编码树的叶节点并且对应于CU。在622处输出叶节点。可替代地,如果MT拆分612指示决定进行MT拆分(返回“1”符号),则块分区器310进入方向决策614。

[0150] 方向决策614将MT拆分的方向指示为水平(“H”或“0”)或垂直(“V”或“1”)。如果决策614返回指示水平方向的“0”,则块分区器310进入决策616。如果决策614返回指示垂直方向的“1”,则块分区器310进入决策618。

[0151] 在决策616和618各自中,在BT/TT拆分时将MT拆分的分区数量指示为两个(二元拆分或“BT”节点)或三个(三元拆分或“TT”)。也就是说,当从614指示的方向为水平时,由块分区器310作出BT/TT拆分决策616,并且当从614指示的方向为垂直时,由块分区器310作出

BT/TT拆分决策618。

[0152] BT/TT拆分决策616指示水平拆分是通过返回“0”所指示的二元拆分514、还是通过返回“1”所指示的三元拆分518。当BT/TT拆分决策616指示二元拆分时,在生成HBT CTU节点的步骤625处,块分区器310根据二元水平拆分514生成两个节点。当BT/TT拆分616指示三元拆分时,在生成HTT CTU节点的步骤626处,块分区器310根据三元水平拆分518生成三个节点。

[0153] BT/TT拆分决策618指示垂直拆分是通过返回“0”所指示的二元拆分516、还是通过返回“1”所指示的三元拆分520。当BT/TT拆分618指示二元拆分时,在生成VBT CTU节点的步骤627处,块分区器310根据垂直二元拆分516生成两个节点。当BT/TT拆分618指示三元拆分时,在生成VTT CTU节点的步骤628处,块分区器310根据垂直三元拆分520生成三个节点。对于从步骤625-628得到的各节点,根据方向614按从左到右或从上到下的顺序来应用数据流600返回到MT拆分决策612的递归。结果,可以应用二叉树和三叉树拆分以生成具有各种大小的CU。

[0154] 图7A和7B提供CTU 710向多个CU的示例分割700。在图7A中示出示例CU 712。图7A示出CTU 710中的CU的空间排列。示例分割700在图7B中也被示出为编码树720。

[0155] 在图7A的CTU 710中的各非叶节点(例如,节点714、716和718)处,按“Z顺序”扫描或遍历所包含的节点(其可以是进一步分割的或者可以是CU),以创建在编码树720中表示为列的节点列表。对于四叉树拆分,Z顺序扫描得到从左上方到右方之后是从左下方到右方的顺序。对于水平和垂直拆分,Z顺序扫描(遍历)分别简化为从上方到下方的扫描和从左方到右方的扫描。图7B的编码树720根据所应用的扫描顺序列出所有的节点和CU。各拆分在树的下一级别生成二、三或四个新节点的列表,直到到达叶节点(CU)为止。

[0156] 在如参考图3所述利用块分区器310将图像分解为CTU并进一步分解为CU、并且使用CU生成各残差块(324)的情况下,利用视频编码器114对残差块进行正向变换和量化。随后扫描如此得到的TB 336以形成残差系数的顺序列表,作为熵编码模块338的操作的一部分。在视频解码器134中进行等效处理以从位流133获得TB。

[0157] 图8A示出包括CTU序列(例如,CTU 812,接着是后续CTU)的示例帧800。各CTU具有128×128个亮度样本的大小。如果帧800要使用处理器205的本地存储器或集成电路以CTU为单位来处理,则由于CTU大小的128×128个亮度样本,所得到的本地存储器要求将是禁止的。这里所描述的视频编码器114和视频解码器134的实现可以通过在比CTU的区域小的区域中处理图像数据或位流来减少片上存储器消耗。片上存储器特别昂贵,因为片上存储器消耗裸晶上的大的面积。软件实现还可以通过将更多存储器访问限制到低级高速缓冲存储器(例如,L1和L2高速缓冲存储器)而受益,从而减少对访问外部存储器的需要。由此,为了减少存储器消耗,视频编码器114和视频解码器134的实现可以一次以比一个CTU的粒度小的粒度处理数据。

[0158] 较小的粒度可以是64×64个亮度样本的区域(或“流水线处理区域”)大小,类似于CTU的一个四叉树细分割。此外,较小的粒度定义被视为不可分割区域的区域。不可分割区域通过流水线架构的各处理阶段。在区域定义与帧(诸如帧800等)上的特定区域相对应的数据的一个聚集或区块(chunk)(诸如样本、块和系数的集合、位流的一部分等)并穿过流水线的意义上,流水线化的处理区域被视为不可分割的。在区域内,可以存在CU的各种布置,

且CU可以跨越多个较小粒度区域。这些区域允许各流水线处理阶段仅本地存储与较小区域(例如,64×64个亮度样本或更少)相关联的数据,而不是与128×128的完全CTU大小相关联的数据。

[0159] 还使用所述的流水线处理区域来实现色度数据的相应本地存储器减少。在各CTU内,按Z顺序处理区域。该处理以光栅扫描方式从CTU前进到CTU,如区域扫描810所示。从视频解码器134的角度来看,第一流水线阶段是熵解码器420。尽管位流133被依次解析,但是解析的句法元素可以根据区域被分组。例如,图8A的区域814首先由熵解码器420来处理。一旦熵解码器420处理一个区域,关联的句法元素被传递到第二流水线阶段。第二流水线阶段可以是逆量化器428和逆变换444。模块428和444对区域中的所有CU进行以产生区域的残差样本448。一旦第二阶段完成,区域的残差样本448被传递到第三阶段。第三阶段可以包括求和450(帧内重建)、参考样本高速缓冲存储器460、参考样本滤波器468和帧内预测模块476。第三阶段模块形成反馈环路,如图4中所示和参考图4所述。反馈环路存在于相邻CU之间,因而存在于区域内以及存在于从一个区域到下一区域。该反馈环路需要在一个流水线阶段内进行第三阶段模块。环内滤波488通常在一个或多个后续流水线阶段中进行。

[0160] 还可以实施涉及帧缓冲器496和运动补偿模块434的用于帧间预测的单独反馈环路。然而,用于帧间预测的反馈环路是从当前帧到先前帧,因而不影响CTU级别处的流水线化操作。图8A的尚待处理的区域(例如816)以浅阴影示出。

[0161] 根据各CTU的编码树,区域可以包含不同大小的不同CU,例如,如关于图7A和7B所描述的。图7A和7B的示例包括编码树的顶层处的四叉树拆分,在图7B中表示为726。四叉树分割成四个64×64区域(其进一步分割成不同大小的CU),与64×64的流水线化处理区域大小对齐。当对包含高度详细纹理且没有可用参考图片的图像帧进行编码时,如“帧内”帧的情况那样,将发生至少一个四叉树拆分的可能性高。由此,对于帧内编码,可以使用与128×128的帧间预测帧所用的CTU大小相比较小的CTU大小(例如,64×64),而不对用户强加不可接受级别的压缩性能损失。

[0162] 使用较小的CTU大小(特定是不超过流水线化处理所用的区域大小的大小)确保没有CU且因而没有TU跨越多个区域。变换是需要在一个流水线阶段内进行的操作,因为变换内的数据依赖关系跨越TB,使得各残差系数影响来自TB的每个输出样本。结果,变换表示流水线操作必须为“原子的”(未进一步分割)的最小级别,因此给定变换的数据必须完全包含在流水线中的一个流水线处理区域或数据区块内。

[0163] 然而,在流水线处理区域内,多个CU且因而多个变换是可能的,因为各变换完全包含在处理区域内。对变换大小的限制设置了处理区域内可能遇到的变换的数量的最差情况。例如,对于64×64处理区域且考虑亮度通道,处理区域内可能遇到的4×4变换的最大数量为16×16=256。处理区域内可能遇到的8×8变换的最大数量为8×8=64等等,直到64×64变换,对于64×64变换,在一个流水线处理区域中可能仅进行一个变换。类似的计算适用于色度通道。

[0164] 对于帧间预测CU,相对较大的CU是可能的。较大CU的可能性由于可以包含高度匹配的参考块的一个或多个参考帧的可用性而发生。视频编码器114可以选择大的CU,且在这样做时将大的样本块从参考帧复制到当前帧。

[0165] 对帧间预测CU进行编码的一种方法是经由“跳过模式”。指示以跳过模式编码的CU

不具有有效残差系数并且从使用“合并索引”选择的空间或时间邻居获得对应的运动矢量。不存在任何有效残差系数意味着不需要进行任何逆量化或逆变换步骤。如此,CTU内的跳过模式CU的放置不由于流水线化而受到约束,并且不需要确定跳过模式CU的TU大小。跳过模式CU不会在相对于流水线处理区域的特定CU对齐的处理中引入困难。在运动参数容易建模的状态下高质量参考帧可用的情况下,通常使用跳过模式CU。因而,针对解码器可以准确地预测运动矢量的块来选择跳过模式CU。运动矢量选择与CU处的期望输出高度匹配的参考块。

[0166] 跳过模式的使用不限于包含相对低细节的图像的部分。包含高度详细纹理的图像部分可以以低成本来复制,因为编码树终止于大的CU大小,并且用于指定空间位移的运动矢量的编码是高度有效的,尤其是经由合并索引编码。随机访问图片组结构的较高时间层中的帧是跳过模式提供高度压缩性能的示例。使用图6描述的灵活的块结构结合例如128×128的相对大的CTU大小使得大的CU能够相当灵活地放置在各CTU内。因此,解码器或编码器可以适应例如通常发生在前景和背景对象之间的边界处的运动场的改变。大的CU通常在低位速率下变得普遍。进一步,这些大的CU可以跨越多个流水线处理区域且不需要受约束以避免跨越流水线处理区域。应用约束以避免跨越流水线处理区域将等效于CTU大小的减小。减小CTU大小将限制CU大小和各CTU内的放置这两者的灵活性,从而不期望地降低压缩效率。

[0167] 图8B是视频中的帧的示例“随机访问”图片组(GOP)结构850。示出了十七个帧的序列。对于各帧,在结构850中示出显示顺序851、编码顺序852及时间层ID 853。视频以帧内帧860开始。由于参考图片缓冲器为空,因此帧内帧860仅可以包括帧内预测CU。要编码的第二帧是单预测帧(P条带)870(因为编码顺序852的第二值是“16”)。单预测帧870在帧内帧860之后被编码并且可以仅参考帧860,如箭头872所示。然而,帧870的显示顺序是16,因此帧860和870之间的内容的相对大的改变是可能的。因而,帧870的编码成本相对高。然而,帧870的编码成本小于帧内帧860的编码成本,在帧内帧860中没有参考帧可用。

[0168] 中间帧能够使用双预测,如具有相对于可用参考帧的两个箭头(例如,图8B中的箭头891和892)的各帧所示。在较高时间层处,从帧到对应参考帧的距离(显示顺序中的增量)较小。通常,当从帧到对应参考帧的距离较小时,压缩性能较高,因为基础图像数据在帧之间变化较小。较高时间层处的帧通常能够使用较大CU并且更频繁地利用跳过模式这两者。例外情况是被遮挡对象变得可见的情况。当被遮挡对象变得可见时,通常在图片组(GOP)结构的约束内不存在可用参考块,并且具有帧内预测的较小CU的使用变得更可能。

[0169] 图9示出用于亮度通道的VVC标准的所支持变换大小的集合900。亮度通道的所支持变换大小如下:4×4、4×8、4×16、4×32、8×4、8×8、8×16、8×32、16×4、16×8、16×16、16×32、32×4、32×8、32×16、32×32、以及64×64。对于使用4:2:0色度格式的色度通道,对于各亮度变换大小,对应的色度变换大小是可用的。色度变换大小具有亮度变换大小的宽度和高度的一半。二维(2D)变换是可分离的,其中在水平和垂直这两者上进行一维(1D)DCT-2变换。可替代地,可以进行对于变换的水平和垂直阶段可独立控制的1D DCT-7变换与1D DST-7变换之间的选择。

[0170] 一般来说,一个TU与CU相关联。对于各颜色通道,一个TB与TU相关联。然而,当不存在用于特定颜色通道的变换的有效系数时,颜色通道的TB可以说是不存在的。颜色通道可

以说是不存在的,因为不需要对残差系数的全零阵列进行变换。尽管一维(1D)变换通常在矩阵乘法运算方面来定义,但使用蝶形阶跃和提升阶跃的实现通常用于复杂性的减小。由于应用的水平和垂直阶段,在各1D变换内和跨越2D块这两者都存在依赖关系。因此,各残差样本受到各残差系数(从逆变换的角度来看)的影响,并且对于正向变换存在对应关系。进行正向变换或逆变换的工作不能被分割成区段(例如以变换TB的一半并且稍后变换另一半)。确定TB的一半的计算成本与确定整个TB的成本几乎相同。因而,确定区段中的变换的架构具有比将变换(从处理流水线的角度来看)确定为"原子"(不可分割)操作的架构高得多的复杂性。相比之下,如下文所描述,当在区段中处理时,PU具有与区段大小vs总PU大小近似成比例的各区段的成本。

[0171] 为了支持跨越多个流水线处理区域的大的CU,考虑帧内预测和帧间预测情况的数据依赖关系。对于帧内预测情况,使用空间上相邻的参考样本和帧内预测模式来生成预测块。当CU大于流水线处理区域大小时,可以在多个部分中确定PU,使得对流水线处理区域操作的预测流水线阶段计算部分PB(针对一个区域)且确定附加部分PB(针对后续区域),这些部分PB共同形成整个PB。

[0172] 对于帧内预测块,确定部分PB需要使用PB的参考样本。参考样本不需要与部分PB相邻。例如,将128×64PB分割并处理为两个64×64部分PB。所得的第二(最右)64×64部分PB使用与原始(完整)128×64PB相对应的参考样本,而非被使用的参考样本是存在于第二64×64部分预测块的位置处的64×64预测块。结果,以比CTU大小的粒度小的粒度操作的流水线化处理架构能够对大于流水线处理区域大小的PB进行帧内预测,其具有缓冲部分PB的额外参考样本的附加成本。附加缓冲参考样本为根据CTU的宽度而确定了大小的样本的行和列,但不需要附加帧宽行缓冲器。

[0173] 对于帧间预测块,将PU分割成多个部分PU是相对简单的,因为所使用的共同信息包括(一个或多个)运动矢量和(一个或多个)参考帧索引。如此,PU可以跨越多个流水线处理区域并且处理为多个部分PU,各部分PU包含在单独流水线处理区域内。即使数个PU跨越多个流水线处理阶段,存储关联的运动矢量以供跨越多个流水线处理区域使用的成本也较低。使用大的PU用于帧间预测对于低位速率应用并且尤其是在使用诸如"随机访问"等的图片组(GOP)结构时的较高层级处是高度有益的。在这种图片组结构中并且尤其在低运动区域中,可以使用相对大的PU。以存在于位流中的最小句法使用大的PU来编码整个图片的相对大的部分。

[0174] 图10A示出CTU 1000的CU和TU,该CTU 1000在编码树的顶层处具有垂直三元拆分,且未进一步拆分。拆分编码树得到三个CU 1020、1022和1024分别具有大小32×128、64×128和32×128。CU 1020、1022和1024分别以偏移(0,0)、(32,0)和(96,0)位于CTU内。对于各CU,存在相同大小的对应PU,并且在CTU 1000中,对应PU跨越多个流水线处理区域。一或多个TU还与各CU相关联。当CU大小等于变换大小其中之一时,一个TU与CU相关联且具有等于对应大小的变换的大小。

[0175] 图10B示出具有与图10A的编码树的CU相关联的TU的替代布置的CTU 1040。当CU大小大于任何变换大小时,以"拼接"方式布置多个TU以占据整个CU。在给定宽度和高度约束的情况下,拼接使用"适配"在CU内的最大可用变换。例如,如图10B中所示,32×128CU 1042及32×128CU 1046以拼接方式使用四个32×32TU。64×128CU 1044以拼接方式使用两个64

×64TU,因为64×64是可用于CU 1044的最大变换大小。如上所述,拆分CTU 1040对于帧内预测或帧间预测操作的性能不会造成问题。然而,关联的TU的处理需要适应流水线处理区域大小。

[0176] 如参考图9所述,可用变换大小限于特定大小集合。从流水线化处理架构的角度来看,用于编码或解码的各TB的处理是不可分割的操作。由于需要适应CU在CTU中的不同放置且需要完全在一个流水线阶段内针对区域进行各变换,因此按照如下描述两种方法。

[0177] 在第一方法中,流水线处理区域不总是固定大小(例如,64×64)。代替地,流水线处理区域的大小适应于各CTU的编码树。因此第一方法可以被称为"灵活流水线处理区域"。术语"灵活流水线处理区域"用于与固定大小的区域的情况区分开,因而流水线处理区域的固定栅格存在于图像帧中,如在本发明的其它地方所讨论的。特别地,使用灵活流水线处理区域,可以如下处理CTU 1040:

[0178] 区域0:32×64(CU 1042的上半部分,包含两个32×32TU)。

[0179] 区域1:64×64(CU 1044的上半部分,包含一个64×64TU)。

[0180] 区域2:32×64(CU 1046的上半部分,包含两个32×32TU)。

[0181] 区域3:32×64(CU 1042的下半部分,包含两个32×32TU)。

[0182] 区域4:64×64(CU 1044的下半部分,包含一个64×64TU)。

[0183] 区域5:32×64(CU 1046的下半部分,包含两个32×32TU)。

[0184] 如此,CTU 1040中的CU的布置得到大小为32×64和64×64的六个灵活流水线处理区域。这是灵活流水线处理区域的数量的最坏情况。

[0185] 当TU未跨越灵活流水线处理区域边界(例如,由于四叉树拆分,如在图7A的示例中)时,灵活流水线处理区域的数量为四个且各区域具有64×64个亮度样本的大小。尽管在流水线化实现中灵活流水线处理区域方法确实实现了TU在CTU中的灵活放置,但与流水线处理区域在各CTU上且因而在图像帧上具有固定放置的架构相比,该示例的流水线的最差情况处理速率增加50%。尽管总采样率未改变,但在流水线架构中,各体区域的处理速率不一定仅与区域大小相关联,因此较小的区域不一定以与其较小的大小相称的较高速率来处理,而是由于处理各区域的开销而以较低速率来处理。因而,设计的最差情况高于所有区域具有相同大小的系统。此外,如将参考图10C所述,还存在灵活处理区域需要每CTU七个区域的情况,从而进一步增加这种架构的最差情况区域速率。

[0186] 在第二方法中,改变CU和TU之间的关系,使得"大"CU(超过可用变换的宽度或高度的CU)的拼接方法被扩展。拼接方法经扩展而同样应用于原本将具有跨越多个流水线处理区域的TU的CU。例如,CU 1022被分割成32×32TU的二乘四阵列。CU(1022)向较小TU的分割的隐含之处在于:借助于编码树和编码树的CU在CTU内的放置来确定该分割,而不需要在位流中存在进一步信令(例如,附加标志)。然而,可以基于作为编码单位的尺寸的编码单位自身的隐含性质来确定该分割。编码单位1022的分割实现了图10A的CTU 1000的以下流水线化处理:

[0187] 区域0:64×64(CU 1020的上半部分、CU 1022的左上四分之一,总共四个32×32TU)。

[0188] 区域1:64×64(CU 1022的右上四分之一、CU 1024的上半部分,总共四个32×32TU)。

[0189] 区域2:64×64(CU 1020的下半部分、CU 1022的左下四分之一,总共四个32×32TU)。

[0190] 区域3:64×64(CU 1020的右下四分之一、CU 1022的下半部分,总共四个32×32TU)。

[0191] 如此,使用第二方法,不管编码树如何,始终使用四个64×64流水线处理区域来处理128×128CTU。变换单位的形状和/或长宽比不同于CU的形状或大小。因此,流水线处理区域的处理速率恒定,而不管CTU内的CU的布置如何。在第二方法中,64×64TU仅可能用于左上位置对齐到相对于图像帧的左上角64×64栅格的CU。对于具有无拆分操作的编码树(其具有256×256CU)、或在水平方向和垂直方向中的各方向上具有最多一个二元拆分(给出了大小为128×64、64×128或64×64的CU)或单个四叉树拆分(给出了各自大小为64×64的四个CU)的编码树的CTU,发生了所需要的编码单位条件。

[0192] 在第三方法中,64×64TU不存在于视频编码器114和视频解码器134中,因而不存在于可用TU大小的集合中。64×64TU相对很少使用。然而,64×64TU在非常低的位速率下可以是有益的,因此64×64变换大小的不存在确实对VVC标准造成压缩性能损失。然而,即使移除64×64变换,仍需要在CU跨越多个流水线处理区域时的TU的拼接。例如,图10E中示出的CTU 10300具有包含两个垂直三元拆分结果的编码树。两个垂直三元拆分得到沿着CTU的中心垂直定向的大小32×128的CU 10320。CU 10320位于相对于CTU 10300的左上角的偏移(48,0)处。CU 10320占据四个64×64流水线处理区域中的每一个且使用相对小的TU。

[0193] 图10E示出根据灵活流水线处理区域的(第一)方法分割成区域的CTU 10300。编码树具有可以使用32×64个亮度样本的两个附加区域来适应的两个垂直三元拆分。各附加区域可以包含两个32×32TU(在CU 10320中整体布置为一乘四列),其中四个区域占据48×64个亮度样本。CTU 10320的所得分割由图10E中的区域边界10330示出。在第一方法中,CU内的TU的拼接仅受变换大小的可用性的约束。使用最大可用大小来以TU拼接CU。

[0194] 图10F示出CTU 10400。CTU 10400涉及根据进一步拼接TU的第二方法被分割成多个区域的CTU 10300。在图10F的示例中,32×128CU 10420(对应于CU 10320)与16×32TU的二乘四阵列而非32×32TU的一乘四阵列相关联。16×32TU的二乘四阵列使得各TU包含在CTU的四个流水线处理区域其中之一内,如区域边界10430所示。在图10F的示例中,没有TU跨越边界10430,意味着没有TU与两个或更多个流水线处理区域重叠或跨越两个或更多个流水线处理区域。具有两个垂直三元拆分的编码树的示例示出使得将CU分割成多个TU以避免TU跨越多个流水线处理区域的不同编码树是可能的。此外,示例情况不限于涉及64×64TU的情况。在(第二)方法中,CU内的TU的拼接进一步受各TU不跨越多个流水线处理区域的要求的约束。结果,使用比最大可用变换大小将使用的TU小的TU,这限制了CU内的TU的拼接。由于使用变换来去相关给定大小的块上的残差样本,因此可预期使用较小TU导致较低压缩性能。然而,对于帧间预测CU,在一些情况下不发生较低压缩性能。单个变换在对包含映射到变换的相对少的基函数的值的残差样本进行去相关时是有效的。因此,单个变换在对低频的残差样本进行去相关时通常是有效的,低频的残差样本是位于频域中朝向TB的左上角的样本。然而,包含高度不连续内容(诸如计算机图形或文本等)或包含集中在块的一部分中的内容的残差样本是较差去相关的,因为所有基函数跨越整个TB。

[0195] 例如,残差样本集中在CU的一部分中的情况对于帧间预测块是常见的并且源自诸

如前景与背景之间的边界等的差。一般来说,较大块的使用得到改进的压缩性能,但也得到跨越前景和背景(或其它类似边界)的块。当前景和背景具有不同运动时,通常从(具有零或接近零值的残差样本的)参考图片良好地预测CU的一部分。相比之下,CU的其它部分是从参考图片较差地预测的。

[0196] 一种解决方案是视频编码器114进一步拆分编码树,从而得到较小的CU。各个较小的CU可以被分配不同预测模式以更好地适应基础图像特征。在对象边界处或被遮挡区域变得可见的位置,运动搜索通常不能找到足够的帧间预测CU。替代地,通常使用帧内预测。CU内的TU的拼接将各个残差系数的空间影响程度限制于对应TB的大小。如此,CU内的拼接TU的使用可以允许选择较大的CU。较大的CU可以是跨越图像帧中的被遮挡的前景和/或背景对象的CU。接着,由于流水线处理区域边界而导致的TU的进一步拼接可能不会过度地降级压缩性能,并且可能甚至由于选择比视频编码器114原本的情况大的CU而提供优点。

[0197] 图10C示出与具有在相反方向上的两个三元拆分的编码树10100相关联的TU。第一三元拆分垂直地得到三个区域10101、10102和10103。中间区域10102进一步被水平地三元拆分,从而得到附加区域。具体地,64×64CU 10122源自区域10102中的进一步三元拆分。CU 10122位于相对于CTU 10100的左上角的偏移(32,32)处。CU 10122跨越四个流水线处理区域,如从处理区域边界10110所见。

[0198] 如果64×64变换将用于CU 10122,则将需要七个处理区域,其由CU 10122的大小为64×64的一个区域和另外六个区域(大小为32×64的两个和大小为64×32的四个)组成以处理CTU中的剩余区域。随着超过通常的四个流水线处理区域的实质性增加的发生,64×64变换大小不适于CU 10122。因此,针对CU 10122使用四个32×32变换来编码残差。

[0199] 图10D示出与具有同一方向上的两个三元拆分以及相反方向上的中间二元拆分的CTU 10200的编码树相关联的TU。作为CTU 10200的编码树的结果,大小为32×64的CU 10222位于相对于CTU 10200的左上角的位置(0,48)处。CU 10222跨越两个流水线处理区域,使得各区域包括CU 10222的16×64部分。基于图9的可用变换大小,使用16×32的大小,以2乘2的方式拼接以占据CU 10200。然后,流水线化实现可以处理如由边界10210所示描绘的区域中的CTU。此外,流水线化实现可以通过单独地进行针对作为一个区域的一部分的CU 10222的最左边两个变换的处理以及进行针对作为另一区域的一部分的CU 10222的最右边两个变换的处理,来处理CU 10200。

[0200] 帧内的拼接TU可以进行各TU边界处的重建。进行各TU边界处的重建由于各CU内的附加反馈环路而增加复杂性。然而,进行各TU边界处的重建不增加最差情况复杂性,因为选择较小的CU的替代方案将得到相同严重程度的反馈环路。对于帧间预测CU,不存在附加反馈环路。

[0201] 图11示出用于确定编码树单位的编码树中的编码单位的预测模式的方法1100。方法1100使得能够实现视频编码器114和视频解码器134的流水线化实现。在确定CTU的编码树时,视频编码器114进行搜索以确定编码树,如关于图6所描述。通过生成叶节点步骤622来测试CTU的各“候选区域”以包含CU而不进一步拆分的选项。在步骤622处,调用方法1100以生成一个或多个候选CU。各候选CU被评价且基于CTU的区域的最佳候选CU与区域的进一步拆分的比较。在考虑源自拆分的最佳候选CU时,确定所得CU的编码树和预测模式这两者。即,“最佳”候选CU被选择用于位流115中。视频编码器114和视频解码器134的布置可能基于

诸如CU大小和/或时间层ID等的方面而限制特定CU的可用预测模式。限制影响了预测模式的确定。此外,该限制还减少用于受影响CU的必要预测模式信令,如参考图12和13所描述,特别是针对步骤1222和1322所描述。方法1100可以由诸如经配置FPGA、ASIC或ASSP等的设备体现。另外,方法1100可以在处理器205的执行下由视频编码器114进行。如此,方法1100可以存储在计算机可读存储介质上和/或存储在存储器206中。针对各候选区域调用的方法1100在条带类型测试步骤1110处开始。

[0202] 在条带类型测试步骤1110处,处理器205测试视频数据113的当前帧的条带。通常,条带类型源自GOP结构(例如,图8B的随机接入GOP结构)。如果条带类型是帧内("I")条带,则添加帧内候选CU 1150以用于评价。如果条带类型是帧间("P"或"B")条带,则方法1100从步骤1110前进到候选CU大小测试步骤1120。

[0203] 在候选CU大小测试步骤1120处,处理器205测试CU的大小以判断是否仅能够使用跳过模式来编码CU,或者是否还可以使用其它预测模式。如果CU的任一边长(即,宽度或高度)超过64,则步骤1120返回"大",并且仅添加跳过模式候选1170以用于评价。在步骤1120返回"大"推断出CU与处理区域重叠,将关于步骤1210和1310进行描述。如果CU是垂直三元拆分的结果,则重叠可能是垂直的。如果CU是垂直三元拆分的结果,则重叠可能是垂直的。

[0204] 由于仅添加一个模式用于针对任一边长超过64的CU的评价,因此不必编码或解码边长超过64的CU的跳过标志。由于CU的大小足以供视频解码器134确定CU的预测模式,因此不需要编码或解码跳过标志。此外,对于128×128的CTU大小,将边长超过64的CU编码为跳过仅防止了针对这样大的CU出现的流水线问题。针对较小大小的CU仍可能出现流水线问题,如由CU内的附加TU拼接所解决的那样。尽管仅允许针对大的CU的跳过编码将似乎限制视频编码器114选择CU的预测模式的灵活性,但进行进一步拆分从而得到较大CU原本将位于的位置处的较小CU的能力是可用的。此外,在P或B条带中期望帧内CU的位置可能是针对被遮挡对象在视频序列中变得可见的位置,因为被遮挡对象趋向于需要较小大小的CU来跟随被遮挡区域的轮廓。如果CU的边长均不超过64,则步骤1120返回"常规"。结果,针对候选CU评价帧内候选模式1150、帧间候选模式1160和跳过模式候选1170中的各个。

[0205] 在CU的候选预测模式的评价(即,1150、1160和1170中的一个或多个的评价)之后,选择最佳模式作为候选CU。基于最低速率或失真成本选择"最佳"候选预测模式。借助于如参考图6所描述的对编码树的遍历,由拆分得到的CU的聚合成本与包含区域中的一个CU的比较使得能够确定编码树。

[0206] 在方法1100的另一布置中,对给定CU的仅测试跳过模式的限制限于CU是CTU的三元拆分的结果的情况。例如,如果CU大小为32×128、64×128(拆分的中心CU,跨越四个流水线处理区域)、128×32或128×64(拆分的中心CU,跨越四个流水线处理区域),则对测试模式进行限制。对测试跳过模式的限制减少了推断跳过模式的情况的数量,因为测试了多个预测模式,并且针对CU用信号通知的最佳者源自CTU的二元拆分。换句话说,使用大小为64×128或128×64的CU,其各自都不会得到跨越多个流水线处理区域的TU。下文描述的图12的步骤1222和图13的步骤1322相应地变化。

[0207] 在方法1100的又一布置中,对仅测试跳过模式的限制限于CTU的三元拆分的中心CU。即,该限制适用于64×128CU(拆分的中心CU,跨越四个流水线处理区域)或128×64CU(分割的中心CU,跨越四个流水线处理区域)。下文描述的图12的步骤1222和图13的步骤

1322相应地变化。

[0208] 图12是用于将CTU的编码树的所得CU编码到位流115中的方法1200的流程图,其中变换大小经选择以使得该方法可以在流水线架构中进行,其中流水线处理区域的大小小于CTU大小。在方法1200中,选择变换大小以使得各变换可以在根据处理栅格定义的区域内完整地处理。方法1200可以由诸如经配置的FPGA、ASIC或ASSP等的设备体现。另外,方法1200可以在处理器205的执行下由视频编码器114进行。如此,方法1200可以存储在计算机可读存储介质上和/或存储在存储器206中。方法1200在确定处理区域步骤1210处由处理器205开始。

[0209] 在确定处理区域步骤1210处,视频编码器114在处理器205的执行下确定图像帧向占据图像帧整体的大小相等且正方形处理区域的栅格的分割。处理区域定义各图像帧的具有比CTU的大小小的大小的部分。例如,在128×128的CTU大小的情况下,处理区域大小可以是64×64,或者在64×64的CTU大小的情况下,处理区域大小可以是32×32。在这些情况中的各情况下,各CTU被分割成以二乘二阵列布置的四个流水线处理区域。处理区域的处理顺序被设置为Z顺序。Z顺序扫描的使用与CTU中的CU的扫描顺序一致,因而与从一个流水线处理区域前进到下一流水线处理区域时确保满足数据依赖关系所必需的顺序对齐。步骤1210用于将图像帧分割成多个大小相等的处理区域,各个大小相等的处理区域是在对位流进行编码的流水线的单个阶段期间处理的块。处理器205中的控制从步骤1210前进到对编码树进行编码的步骤1215。

[0210] 在对编码树进行编码的步骤1215处,块分区器310在处理器205的执行下将由方法1100确定的CTU的编码树编码到位流115中。如参考图5和6描述且使用图7A和7B的示例,编码树根据一系列拆分将CTU分解成一个或多个CU。在方法1100中,块分区器310测试了拆分的许多不同组合以到达特定编码树,该特定编码树使得CTU能够以高压缩比来编码,同时维持解码图像的保真度,如参考图3所描述。方法1100通过确定编码树来有效地确定各编码单位(CU)的大小。处理器205中的控制从步骤1215前进到选择CU步骤1220。

[0211] 在选择CU步骤1220处,块分区器在处理器205的执行下选择CTU的编码树的一个CU。当对CU进行编码时,针对步骤1215的编码的编码树中的各CU进行步骤1220。所选择的CU具有特定大小和在图像帧中的位置,因而具有相对于包含CTU的左上角的位置。由此,可以说所选择的CU占据包含CTU内的给定区域。处理器205中的控制从步骤1220前进到预测模式测试和编码步骤1222。

[0212] 在预测模式测试和编码步骤1222处,处理器205测试如方法1100中所确定的所选择的CU的预测模式。如果帧的条带类型是(i)“P”或“B”,或者(ii)如果“跳过标志”(或“cu_skip_flag”)被编码。跳过标志指示CU是否是使用跳过模式编码的。如果CU不是使用跳过模式编码的,则编码“pred_mode”标志,从而指示帧间预测或帧内预测中的哪一个用于CU。

[0213] 如关于图11的步骤1120所描述,可以基于CU的大小来推断跳过模式。因此,对于“大”CU(针对每个步骤1120),CU大于或等于处理区域其中之一的大小,并且通过CU的隐含大小性质来推断跳过模式。CU将有效地与处理区域其中之一的边界重叠。相应地推断跳过模式,并且不将跳过标志编码到位流中。此外,基于CU的隐含性质来确定合并索引。可选地,如果所推断的跳过是基于三元拆分或三元拆分的中心CU,则合并索引是基于CU的隐含性质(其为CU的形状和/或位置)来确定的。如果未推断跳过模式,则步骤1322判断为CU不大于或

等于处理区域其中之一的大小且包括跳过标志。在具有所推断的跳过模式的布置中,仅在CU大小小于预定阈值的情况下编码跳过标志,如参考图11的步骤1120所描述。例如,边长都不超过64个样本。跳过标志的编码可以另外仅在帧的时间层ID(例如,853)低于预定阈值时进行编码。例如,时间ID的阈值可能低于GOP结构大小的最大时间层,诸如在GOP大小为十六个图片时低于4等。如果满足阈值(例如,任一边长大于64的CU),则具有所推断的跳过模式的布置不需要对跳过标志进行编码,因为在方法1100中针对这样的情况仅测试跳过编码。此外,步骤1222在预测模式信息仅在针对CU测试多于一个预测模式时被编码的方面与方法1100一致。因此可以实现位流115中存在的信令的相应减少且因而更高的压缩性能。

[0214] 如以下布置所示,“大”和“常规”集合之间的CU的不同分割是可能的。在“大”集合中具有更多CU得到进行拼接以解决流水线处理问题的更少TU实例,代价是在针对这些大小的CU选择预测模式时向视频编码器114提供较少的灵活性。

[0215] 在方法1100的又一布置中,要使用的对推断跳过模式(步骤1120返回“大”)的CU的边长的限制适用于两个边均大于或等于64的任何CU。要求两个边均大于或等于64得到发生跳过推断的CU的集合为:128×128、64×128和128×64。图12的步骤1222和图13的步骤1322相应地变化。

[0216] 在方法1100的又一布置中,要使用的对推断跳过模式(步骤1120返回“大”)的CU的边长的限制适用于任一边长大于64的任何CU,从而得到发生跳过推断的CU集合为:128×128,64×128、128×64、128×32、32×128、64×64。再次,图12的步骤1222和图13的步骤1322相应地变化。“大”与“常规”集合之间的阈值(或边界)可以取决于系统100的“操作点”(例如,位流的期望位速率)。代替具有固定边界,可以在位流115中用信号通知边界作为阈值,从而允许视频编码器115选择用于系统100的边界。边界可以作为边长的log2来用信号通知,并且CU的“任一”或“两个”边必须匹配将被视为在“大”集合中的CU的用信号通知的边界这一要求也可以用信号通知。

[0217] 如果预测模式被确定(或推断)为跳过模式(步骤1222返回“跳过”),则处理器205中的控制从步骤1222前进到进行运动补偿步骤1270。否则,(预测模式是帧间预测或帧内预测并且步骤1222返回“帧内或帧间”),处理器205中的控制从步骤1222前进到识别处理区域步骤1225。

[0218] 在识别处理区域步骤1225处,处理器205使用在步骤1220处选择的CU的区域来识别哪个(哪些)处理区域与选择的CU重叠。例如,图10A的CU 1022与CTU 1000中的四个64×64处理区域重叠。处理器205中的控制从步骤1225前进到确定CU变换大小约束步骤1230。

[0219] 在确定CU变换大小约束步骤1230处,处理器205确定CU的初始变换大小。将初始变换大小设置为宽度不超过所选择的CU的宽度且高度不超过所选择的CU的高度的预定集合的变换大小(诸如图9的变换大小等)中的最大变换大小。因而,初始变换大小是“适配”到所选择的CU中的最大大小。考虑到亮度通道,通常单个变换占据整个CU。色度通道具有类似关系,其中色度变换大小与针对4:2:0色度格式调整的亮度通道的变换大小相对应,其是各变换的宽度和高度的一半。

[0220] 在单个变换未完全占据CU的情况下,在步骤1230处使用“拼接”处理来应用初始变换大小以利用变换占据整个CU。例如,对于具有32×128的大小的CU 1020,需要以一乘四拼接的32×32的初始变换大小来占据整个CU。对于大小为64×128的CU 1022,初始变换大小

为64×64,利用一乘二拼接以占据整个CU。处理器205中的控制从步骤1230进前进到处理区域边界重叠测试步骤1235。

[0221] 在处理区域边界重叠测试步骤1235处,处理器205判断初始变换大小的且与所选择的CU相关联的任何变换是否跨越两个或更多个处理区域(或者“横跨”两个或更多个处理区域的边界)。换句话说,在步骤1235处,处理器205判断编码单位是否与处理区域之间的边界重叠。例如,对于位于相对于CTU 1000的左上角的位置(0,0)处的CU 1020的32×32的初始变换大小,各个变换完全包含在64×64流水线处理区域内。顶部两个32×32变换位于一个处理区域中,并且底部两个32×32变换位于另一处理区域中。在这种情况下,步骤1235返回“否”并且处理器205中的控制前进到CU变换大小步骤1240。

[0222] 然而,对于具有64×64的初始变换大小且位于相对于CTU 1000的左上角的位置(32,0)处的CU 1022,初始变换大小占据从(32,0)到(95,64)的区域。当处理区域对齐到64×64栅格时,初始第一变换占据两个处理区域,并且占据从(32,64)到(95,127)的区域的第二变换占据另外两个处理区域。当将导致跨越两个或更多个处理区域之间的边界的这些所提议的初始变换其中至少之一(在图10A的示例中为两个)时,步骤1235返回“是”,并且处理器205中的控制从步骤1240前进到处理区域变换大小步骤1245。

[0223] 作为一般规则,变换跨越两个或更多个处理区域的可能性源自编码树的顶层处的三元拆分的应用。结果是由于CTU大小、处理区域大小、变换的边尺寸均为2的幂次方、以及CTU大小为处理区域大小的宽度和高度的两倍而引起的。因此,仅编码树的顶层处的三元拆分可能产生在空间上(水平地或垂直地)偏移处理区域的宽度或高度的一半的CU。当使用初始变换大小的变换时,偏移CU可能得到将跨越两个或更多个处理区域的变换,从而产生了针对以处理区域的粒度操作的流水线化架构的实质性实现挑战。

[0224] 在给定CTU大小、处理区域大小与变换边尺寸大小之间的关系的情况下,一个解决方案可以是禁止编码树中的边长超过64个样本的区域的三元拆分。用于边长超过64个样本的区域的剩余选项没有进一步拆分、二元拆分或四叉树拆分。如果没有进行进一步拆分,则四个64×64变换的拼接将是可能的,其中每一个完全包含在流水线处理区域内。如果在任一方向上进行128×128CTU的二元拆分,则禁止所得子区域中在相反方向上的三元拆分将防止可能的64×64CU跨越两个流水线处理区域。然而,进一步拆分三元拆分的中间(64×64)CU可以解决关于流水线处理的变换放置问题。初始禁止将防止搜索中间编码树。如果在任一方向上进行二元拆分(得到大小为64×128或大小为128×64的两个区域),则由于一个边长为128,应用在同一方向上的任一所得区域的三元拆分也将是不可能的。

[0225] 然而,同一方向上的三元拆分将不会得到跨越流水线处理区域之间的边界的任何变换。例如,128×64区域的水平三元拆分(源自CTU的水平二元拆分)将得到16×128、32×128和另一个16×128区域。沿着128的边长,通常边长32的变换被使用、拼接四次,并且不会得到跨越多个流水线处理区域的任何变换。最后,如果进行四叉树拆分,则各所得区将在单独流水线处理区域内,并且将不会导致进一步的流水线处理问题,而不管后续拆分如何。

[0226] 因此,虽然禁止边长超过64的任何区域上的三元拆分是解决流水线处理性能的一个方法,但禁止确实禁止了潜在有用块大小,从而降低压缩性能。此外,限制防止了使用“跳过模式”帧间预测CU,其在不具有残差系数时不会由于变换放置而引起流水线处理区域。因此,该限制不利地影响压缩性能,因为跳过模式CU(例如,针对大CU)的灵活放置尤其在低位

速率下是期望的。

[0227] 如所描述,基于CU大小和CTU内的位置来进行步骤1235的测试。因而步骤1235实现了隐含测试,从而不增加视频编码器114的“搜索空间”。也就是说,视频编码器114不被给予决定TU配置时的附加自由度(例如,标志的添加)。不存在决定TU配置时的附加自由度意味着不需要位流115中的附加信令来存储自由度的结果。换句话说,步骤1235的操作隐含地基于CTU的编码树的性质。步骤1235的输出独立于且不涉及与要编码在位流中的TU大小有关的显式信号的生成。

[0228] 在CU变换大小步骤1240处,处理器205将CU的变换大小选择为步骤1230的所确定的CU变换大小。由于没有所得TU跨越多个流水线处理区域,因此不需要进一步将CU分割成附加TU。处理器205中的控制从步骤1240前进到应用正向变换和量化步骤1250。

[0229] 在处理区域变换大小步骤1245处,处理器205确定所选择的CU的变换大小,使得所得变换均不跨越所选择的CU所跨越的流水线处理区域中的两个或更多个。例如,大小64×128的CU 1022位于相对于包含CTU 1000的左上的(32,0)处。如此,CU 1022跨越从(32,0)到(95,127)的区域。在水平方向上,CU以64的X偏移通过流水线处理区域。因此,在水平方向上,TU宽度需要至多32,以是适合于流水线化实现的最大TU宽度。在垂直方向上,TU宽度需要至多64,以是适合于流水线化实现的最大TU宽度。然而,如图9中所见,不存在可用的32×64TU。可以使用的最大可用TU是32×32,因此选择32×32大小的TU。通过针对CTU 1000选择32×32TU,CTU可以以如参考图10A所描述的流水线方式进行处理。步骤1245有效地操作以从(例如如图9所示的)可用的变换大小集合(多个)中选择编码单位的变换大小。变换大小经选择以适配在编码单位内且可以不同于处理区域的大小。处理器205中的控制从步骤1245前进到应用正向变换和量化步骤1250。

[0230] 在应用正向变换和量化步骤1250处,变换模块326和量化器模块334在处理器205的执行下应用步骤1240或步骤1245的所选择的变换以变换差324且产生残差系数336。如果CU大小等于变换大小,则进行单个变换。如果CU大小大于变换大小,则以拼接方式应用变换,使得对所有差324进行变换。此外,借助于在步骤1245处选择的变换大小,个体变换不覆盖跨越流水线处理区域中的两个或更多个的区域。

[0231] 诸如视频压缩标准的“参考软件”等的软件实现通常每次一个CTU地处理各帧,而不使用更细粒度的处理(诸如比CTU大小小的区域的流水线化处理等)。参考软件实现不会遇到诸如以上识别的流水线处理区域问题等的问题,因为它们通常不会实时地或在资源受限的装置中运行。实际的实现(尤其是利用流水线架构的硬件实现以及一些软件实现)受益于完全包含在不同流水线处理区域内的变换。例如,受益于完全包含在不同流水线区域内的变换的软件实现包括使用相同流水线架构来改善局部性的多核实现。完全包含在不同流水线区域内的重要益处变换是流水线处理区域的均匀大小和速率。处理器205中的控制从步骤1250前进到编码残差系数步骤1255。

[0232] 在编码残差系数步骤1255处,熵编码器338在处理器205的执行下将步骤1250的残差系数编码到位流115中。首先,“根编码块标志”被编码,从而指示存在由步骤1250的量化得到的至少一个有效残差系数。根编码块标志针对CU编码一次,并且跨CU的任何TU的任何TB的所有颜色通道用信号通知CU的任何变换的有效性。假设针对跨越CU的任何颜色通道的任何变换存在至少一个有效残差系数,则在各颜色通道内,针对在该颜色通道中应用的各

变换对单独的编码块标志进行编码。各编码块标志指示对应变换块中的至少一个有效残差系数的存在。对于具有至少一个有效残差系数的变换,还编码有效图以及有效系数的大小和符号。处理器205中的控制从步骤1255前进到帧内模式测试1260。

[0233] 在帧内模式测试1260处,通过处理器205测试所选择的CU的预测模式。如果预测模式是帧内预测(步骤1260处为“是”),则处理器205中的控制前进到进行帧内预测步骤1265。否则(预测模式是帧间预测并且步骤1260返回“否”),处理器205中的控制前进到进行运动补偿步骤1270。

[0234] 在进行帧内预测步骤1265处,帧内预测模块364在处理器205的执行下生成样本的帧内预测块(366)。根据用于所选择的CU的各PB的帧内预测模式,使用滤波参考样本362来生成样本的帧内预测块366。当由于步骤1245而导致多个TU与CU相关联时,在所选择的CU内部的各TU边界处应用帧内重建处理。除了在各个CU边界处的重建样本之外,还用在CU内部的各个TU边界处的重建样本来更新参考样本高速缓冲存储器356。在CU内部的TU边界处的重建使得CU内部的当前TU上方或左侧的TU的残差系数有助于用于生成与当前TU同位置的PB的一部分的参考样本。因此,CU内部的TU边界处的重建可以减少失真且改进压缩效率。处理器205中的控制从步骤1265前进到重建CU步骤1275。

[0235] 在进行运动补偿步骤1270处,运动补偿模块380在处理器205的执行下产生滤波块样本382。通过从帧缓冲器372提取一个或两个样本块374来产生滤波块样本382。对于各样本块,根据参考图片索引来选择帧,并且根据运动矢量来指定像素中的相对于所选择的CU的空间位移。对于从帧缓冲器372提取的各样本块,根据运动矢量的“子像素”位移部分应用滤波。运动矢量的子像素位移部分的精度可以是四分之一像素精度或十六分之一像素精度。在使用两个块的情况下,将所得滤波块混合在一起。在方法1100中确定参考图片索引和(一个或多个)运动矢量。处理器205中的控制从步骤1270前进到重建CU步骤1275。

[0236] 在重建CU步骤1275处,求和模块352在处理器205的执行下通过将残差样本350与用于帧间预测或经帧内预测CU的PU 320相加而产生重建样本354。对于跳过模式CU,不存在残差样本,且因此从PU 320导出重建样本354。重建样本354可以用于当前帧中的后续帧内预测CU的参考。在应用环内滤波(即,应用环内滤波器368)之后,将重建样本354写入到帧缓冲器372以供后续帧中的帧间预测CU的参考。环内滤波器368的解块滤波应用于CU的内部边界。即,将解块滤波应用于CU内部的TU之间的边界,其是由于CU大小和由于流水线处理区域边界这两者而通过拼接而产生的。处理器205中的控制从步骤1275前进到最后CU测试步骤1285。

[0237] 在最后CU测试步骤1285处,处理器测试所选择的CU是否为CTU中的最后一个。如果不是(在步骤1160中为“否”),则处理器205中的控制返回至步骤1215。如果所选择的CU是按CU扫描顺序即深度优先Z顺序扫描的CTU中的最后一个,则方法1200终止。在方法1200终止之后,编码下一CTU,或视频编码器114前进到视频的下一图像帧。

[0238] 图13示出用于从位流133解码CTU的CU的方法1300。在方法1300中,选择变换大小以使得方法1300可以在流水线化架构中进行。相应的流水线处理区域的大小小于CTU大小,并且流水线处理区域的速率在各CTU的编码树上是独立的。方法1300可以由诸如经配置的FPGA、ASIC或ASSP等的设备来体现。另外,方法1300可以在处理器205的执行下由视频解码器134进行。如此,方法1300可以存储在计算机可读存储介质上和/或存储在存储器206中。

方法1300在确定处理区域步骤1310处由处理器205开始。

[0239] 在确定处理区域步骤1310处,视频解码器134在处理器205的执行下确定将位流的图像帧分割成占据整个图像帧的相同大小和正方形处理区域的栅格。步骤1310以匹配步骤1210的方式确定图像帧的分割。步骤1310用于将图像帧分割成多个大小相等的处理区域,各个大小相等的处理区域是在解码位流的流水线的单个阶段期间处理的块。处理器205中的控制从步骤1310前进到对编码树进行解码的步骤1315。

[0240] 在对编码树进行解码的步骤1315处,熵解码器420在处理器205的执行下从位流133解码CTU的编码树。编码树根据一系列拆分将CTU分解成一个或多个CU,如参考图5和6且使用图7A和7B的示例所描述。从位流133解码的编码树是在图12的步骤1215确定的编码树。步骤1315通过使用编码树对CTU进行解码来有效地确定各个编码单位(CU)的大小。处理器205中的控制从步骤1315前进到选择CU步骤1320。

[0241] 在选择CU步骤1320处,视频解码器134在处理器205的执行下根据在与同编码树相关联的句法存在于位流134中的方向相对应的正向方向上迭代通过编码树来选择经解码的编码树的一个CU。正向方向涉及Z顺序扫描。所选择的CU具有特定大小和在图像帧中的位置,因而具有相对于包含CTU的左上角的位置。由此,可以说所选择的CU占据包含CTU内的给定区域。处理器205中的控制从步骤1320前进到确定预测模式测试步骤1322。

[0242] 在确定预测模式测试步骤1322处,处理器205确定所选择的CU的预测模式。如果帧的条带类型是“P”或“B”,则熵解码器420解码指示是否使用跳过模式编码CU的“跳过标志”(或“cu_skip_flag”)。如果未使用跳过模式来编码CU,则熵解码器420解码“pred_mode”标志。“pred_mode”标志指示帧间预测或帧内预测中的哪一个用于CU。如关于图11的步骤1120所描述,可以基于CU的大小来推断跳过模式。因此,对于“大”CU(针对每个步骤1120),CU大于或等于处理区域其中之一的大小,并且通过CU的隐含大小性质来推断跳过模式。相应地推断跳过模式,并且不将跳过标志编码到位流中。而是,基于CU的隐含性质确定合并索引。可选地,如果所推断的跳过是基于三元拆分或三元拆分的中心CU,则合并索引是基于CU的隐含性质(其为CU的形状和/或位置)来确定的。如果未推断跳过模式,则步骤1322判断为CU不是大于或等于处理区域其中之一的大小并且包括跳过标志。

[0243] 在具有所推断的跳过模式的布置中,仅在CU大小小于预定阈值的情况下(例如,在两个边长均不超过64个样本的情况下)才解码跳过标志。否则,CU被判断为“大CU”,并且跳过模式被推断为正被使用。跳过标志的编码可以另外仅在时间层ID低于预定阈值(例如,低于GOP结构大小的最大时间层,例如,在GOP大小为16个图片时低于四)时编码。如果满足阈值测试(例如,阈值之上的大CU大小和/或时间层ID),则具有所推断的跳过模式的布置不需要解码跳过标志i,因为在方法1100中针对这样的情况仅测试跳过编码。由此,预测模式被判断为跳过模式。此外,步骤1322在预测模式信息仅在针对CU测试多于一个预测模式时被解码的方面与方法1100一致。当针对CU仅测试一个预测模式时,视频解码器134基于例如CU大小来推断预测模式,而非显式地解码预测模式。

[0244] 如果判断为(或推断)该预测模式是跳过模式(步骤1322处的“跳过”),则处理器205中的控制从步骤1322前进到解码运动参数步骤1370。否则(预测模式是帧间预测或帧内预测),步骤1322返回“帧内或帧间”,并且处理器205中的控制前进到识别处理区域步骤1325。

[0245] 在识别处理区域步骤1325处,处理器205使用在步骤1320处选择的CU的区域来识别哪个(哪些)处理区域与所选择的CU重叠。例如,图10A的编码单位1022与CTU 1000中的四个64×64处理区域重叠。步骤1325以与图12的步骤1225类似的方式操作。处理器205中的控制从步骤1325前进到确定编码单位变换大小约束步骤1330。

[0246] 在确定编码单位变换大小约束步骤1330处,处理器205确定CU的初始变换大小。以与步骤1230的确定类似的方式设置初始变换大小。处理器205中的控制从步骤1330前进到处理区域边界重叠测试步骤1335。

[0247] 类似于重叠测试步骤1235,在处理区域边界重叠测试步骤1335处,处理器205判断初始变换大小的且与所选择的CU相关联的任何变换是否跨越两个或更多个处理区域。换句话说,步骤1335判断编码单位是否与处理区域之间的边界重叠。如果各个变换完全包含在处理区域内(步骤1335处的“否”),则处理器205中的控制前进到CU变换大小步骤1340。如果源自初始变换大小的变换其中至少之一跨越或“横跨”两个或更多个处理区域之间的边界(步骤1335处的“是”),则处理器205中的控制前进到处理区域变换大小步骤1345。测试步骤1335的结果取决于CU大小和在CTU内的位置,这些完全由CTU的编码树描述。如此,不需要从位流133解码附加信令以判断CU是否跨越两个处理区域。而是,使用CU的隐含性质(大小和位置)以测试处理区域边界是否重叠。

[0248] 在CU变换大小步骤1340处,处理器205根据步骤1240的变换大小选择将CU的变换大小选择为步骤1330的所确定的CU变换大小。处理器205中的控制从步骤1340前进到解码残差系数步骤1350。

[0249] 在处理区域变换大小步骤1345处,处理器205确定所选择的CU的变换大小,使得所得变换都不跨越所选择的CU所跨越的流水线处理区域中的两个或多个。步骤1345根据步骤1245的变换大小选择进行操作。步骤1345有效地操作以从(例如如图9所示的)可用的变换大小集合(多个)中选择编码单位的变换大小。变换大小经选择以适配在编码单位内并且可以不同于处理区域的大小。处理器205中的控制从步骤1345前进到解码残差系数步骤1350。

[0250] 在解码残差系数步骤1350处,熵解码器420在处理器205的执行下从位流115解码残差系数。编码单位通过将逆变换应用于编码单位中的各变换单位的残差系数。在解码残差系数时,首先解码“根编码块标志”。根编码块标志指示CU的任何TU中(即,跨所有颜色通道)的至少一个有效残差系数的存在。当根编码块标志指示在CU中存在有效残差系数时,在各颜色通道内,针对在该颜色通道中应用的各变换解码单独的编码块标志。各编码块标志指示对应变换中的至少一个有效残差系数的存在。对于具有至少一个有效残差系数的变换,还解码有效图以及有效系数的大小和符号。处理器205中的控制从步骤1350前进到逆量化和应用逆变换步骤1355。

[0251] 在逆量化和应用逆变换步骤1355处,解量化器模块428和逆变换模块444在处理器205的执行下逆量化残差系数以产生按比例缩放变换系数440。在步骤1355处,应用步骤1340或步骤1345中的所选择的变换来变换按比例缩放变换系数440以产生残差样本448。与步骤1250一样,根据所确定的变换大小以拼接的方式进行变换的应用。此外,借助于在步骤1345处选择的变换大小,个体变换不会覆盖跨越流水线处理区域中的两个或更多个的区域。与方法1200一样,实际实现(尤其是利用流水线架构的硬件实现以及一些软件实现)受益于完全包含在不同流水线处理区域内的变换。有益于所描述的布置的示例软件实现是可

以使用相同的流水线架构来改善数据局部性的多核实现。处理器205中的控制从步骤1355前进到帧内模式测试步骤1360。

[0252] 在帧内模式测试1360处，由处理器205来测试所选择CU的所确定预测模式。如果预测模式是帧内预测(步骤1360处的“是”)，则处理器205中的控制前进到进行帧内预测步骤1365。否则(预测模式是帧间预测)，步骤1360返回“否”并且处理器205中的控制前进到解码运动参数步骤1370。

[0253] 在进行帧内预测步骤1365处，帧内预测模块476在处理器205的执行下生成帧内预测样本块(480)。根据用于所选择CU的各个PB的帧内预测模式，使用滤波参考样本472来生成帧内预测样本块480。当由于步骤1345而使多个TU与CU相关联时，在所选择CU内部的各TU边界处应用帧内重建处理。除了各CU边界处的重建样本之外，还用CU内部的各TU边界处的重建样本来更新重建样本高速缓冲存储器460。CU内部的TU边界处的重建使得CU内部的当前TU上方或左侧的TU的残差有助于用于生成与当前TU同位置的PB的一部分的参考样本。CU内部的TU边界处的重建可以操作以减少失真且改进压缩效率。处理器205中的控制从步骤1365前进到重建CU步骤1380。

[0254] 在解码运动参数步骤1370处，熵解码器420在处理器205的执行下解码所选择的CU的运动矢量。解码运动矢量包括通过以下操作选择运动矢量：(i)如果推断出跳过模式(如在1120和1322处从CU的性质所识别的)，则解码合并索引，或者(ii)如果CU未推断出跳过模式，则解码跳过标志以解码合并索引。使用空间上和时间上相邻的块来创建候选运动矢量的列表(称为“合并列表”)。从位流133解码合并索引以从合并列表选择候选之一。可以基于CU的隐含性质(如上文关于步骤1322所描述)或通过从位流解码拆分模式标志来确定合并索引。如果使用跳过模式编码所选择的CU，则所选择的候选变为CU的运动矢量。如果使用帧间预测来编码所选择的CU，则从位流133解码运动矢量增量并且将其添加到根据经解码的合并索引而选择的候选。处理器中的控制从步骤1370前进到进行运动补偿步骤1375。

[0255] 在进行运动补偿步骤1375处，运动补偿模块434在处理器205的执行下产生滤波块样本438。通过从帧缓冲器496提取一个或两个样本块498来产生滤波块样本438。对于各样本块498，根据参考图片索引来选择帧，并且根据运动矢量来指定像素中的相对于所选择CU的空间位移。对于从帧缓冲器372提取的各个样本块，根据运动矢量的“子像素”位移部分应用滤波。运动矢量的子像素位移部分的精度可以是四分之一像素精度或十六分之一像素精度。在使用两个块的情况下，将所得的滤波块混合在一起。参考图片索引和(一个或多个)运动矢量从位流133解码并且在方法1100中确定。处理器205中的控制从步骤1375前进到重建CU步骤1380。

[0256] 在重建CU步骤1380处，求和模块450在处理器205的执行下产生重建样本456。通过将残差样本448与用于帧间预测或帧内预测CU的PU 452相加来产生重建样本456。对于跳过模式CU，不存在残差，因此从PU 452导出重建样本456。重建样本456可以用于当前帧中的后续帧内预测CU的参考。在应用环内滤波(即，应用环内滤波器488)之后，重建样本456被写入到帧缓冲器496以供后续帧中的帧间预测CU的参考。环路滤波器488的解块滤波应用于CU的内部边界。即，将解块滤波应用于CU内部的TU之间的边界(源自由于CU大小和由于流水线处理区域边界这两者而导致的拼接)。处理器205中的控制从步骤1380前进到最后CU测试步骤1385。

[0257] 在最后CU测试步骤1385处,处理器205测试所选择的CU是否为按CU扫描顺序(深度优先Z顺序扫描)的CTU中的最后CU。如果不是(步骤1385处为“否”),则处理器205中的控制返回到步骤1315。如果所选择的CU是CTU中的最后CU(步骤1385处为“是”),则方法1300终止。在方法1300终止之后,解码下一CTU,或者视频解码器134前进到位流的下一图像帧。

[0258] 在视频编码器114和视频解码器134的替代布置中,推断跨越多个流水线处理区域的CU以按“跳过模式”编码,因此不具有任何关联的残差系数,且因而不需要进行变换来编码或解码这样的块。如此,当视频编码器114在步骤1215处确定编码树时,当测试将得到跨越多个处理区域的TU的这样的CU时,需要在没有任何关联的残差系数的情况下编码这些CU。

[0259] 产业上的可利用性

[0260] 所述的配置可应用于计算机和数据处理行业,并且特别可用于对诸如视频和图像信号等的信号进行编码或解码的数字信号处理,从而在不会由于提供具有小于最大支持块大小或CTU大小的处理区域大小的流水线化实现的可能性对硅面积造成内存消耗方面的过高成本的情况下实现高压缩效率。在一些实现中,所述的布置对于VVC标准是有用的,因为实现区域TU的拼接(例如,如在步骤1145和1245处所实现的)有助于防止流水线效率低下,特别是对于帧间预测模式而言。如上所述,这里所述的一些实现允许将三元编码树用于较大的CU,或者在减少对处理时间和/或质量的影响的情况下使用64×64CU。

[0261] 前述仅说明本发明的一些实施例,并且可以在没有背离本发明的范围和精神的情况下对本发明进行修改和/或改变,其中这些实施例仅是示例性而非限制性的。

[0262] (仅澳大利亚)在本说明书的上下文中,词语“包括”意味着“主要但未必仅包括”或“具有”或“包含”,而不是“仅由…组成”。词语“包括(comprising)”的诸如“comprise”和“comprises”等的词尾变化具有相应的变化含义。

[0263] 相关申请的交叉引用

[0264] 本申请根据35U.S.C.§119要求于2018年8月17日提交的澳大利亚专利申请2018217333的优先权益,这里出于所有目的将该专利申请的全部内容通过引用合并于此。

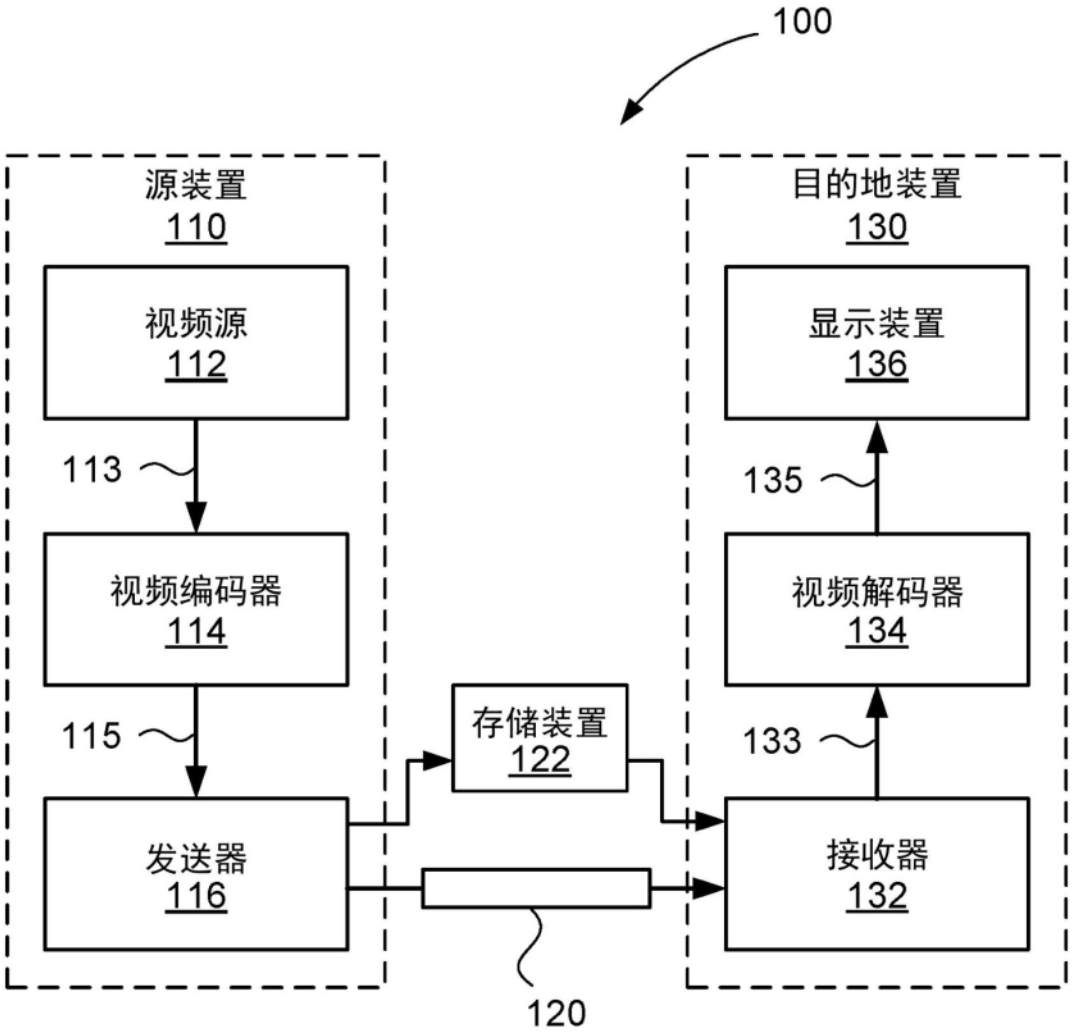

100
源装置
110
视频源
112
113
视频编码器
114
115
发送器
116
存储装置
122
120
目的地装置
130
显示装置
136
135
视频解码器
134
133
接收器
132

图1

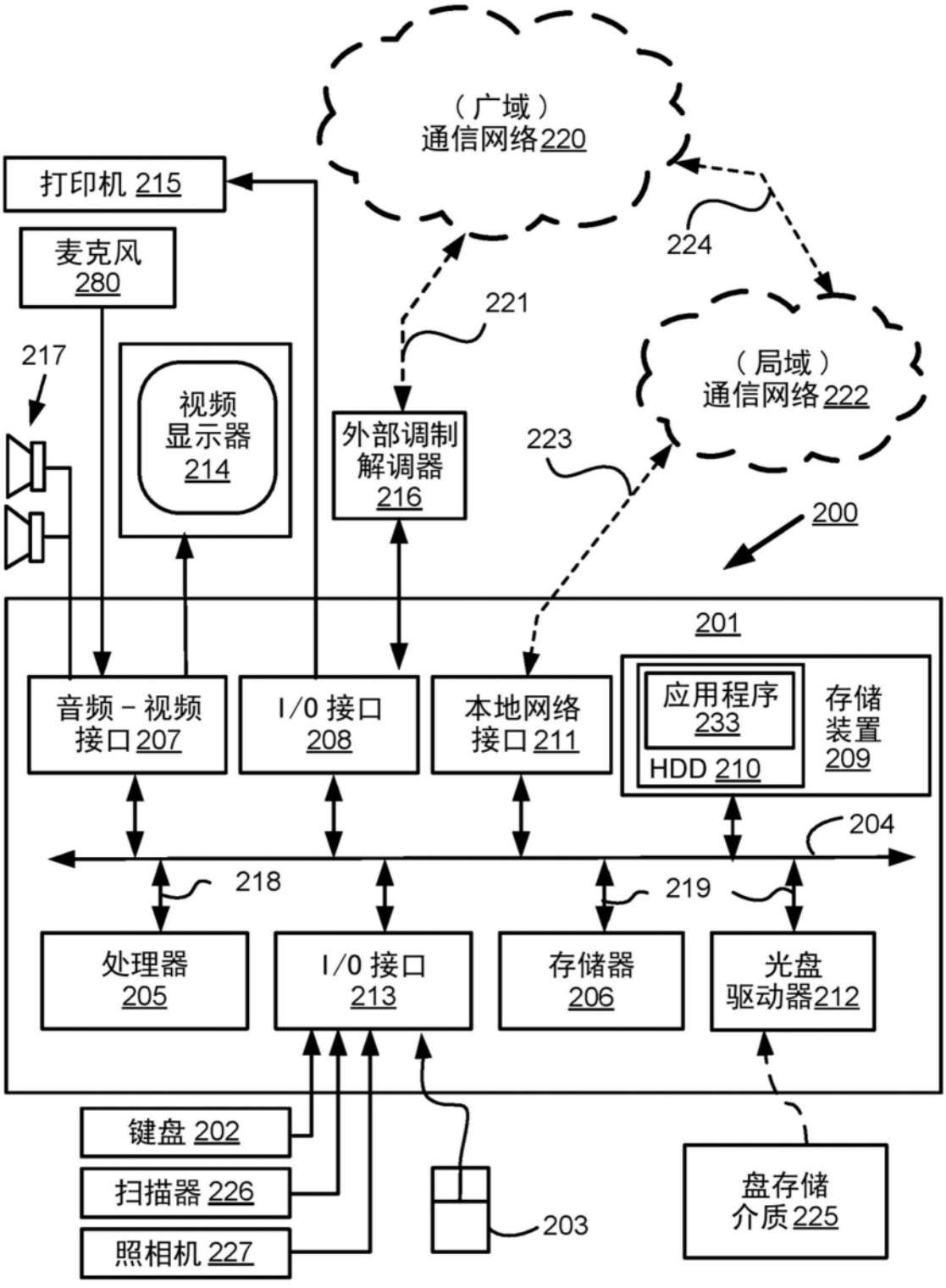

（广域）
通信网络220
打印机 215
麦克风
280
224
221
217
视频
显示器
214
外部调制
解调器
216
（局域）
通信网络222
223
200
201
音频－视频
接口207
I/0 接口
208
本地网络
接口211
应用程序
233
HDD 210
存储
装置
209
204
218
219
处理器
205
I/0 接口
213
存储器
206
光盘
驱动器212
键盘 202
扫描器 226
照相机 227
203
盘存储
介质225

图2A

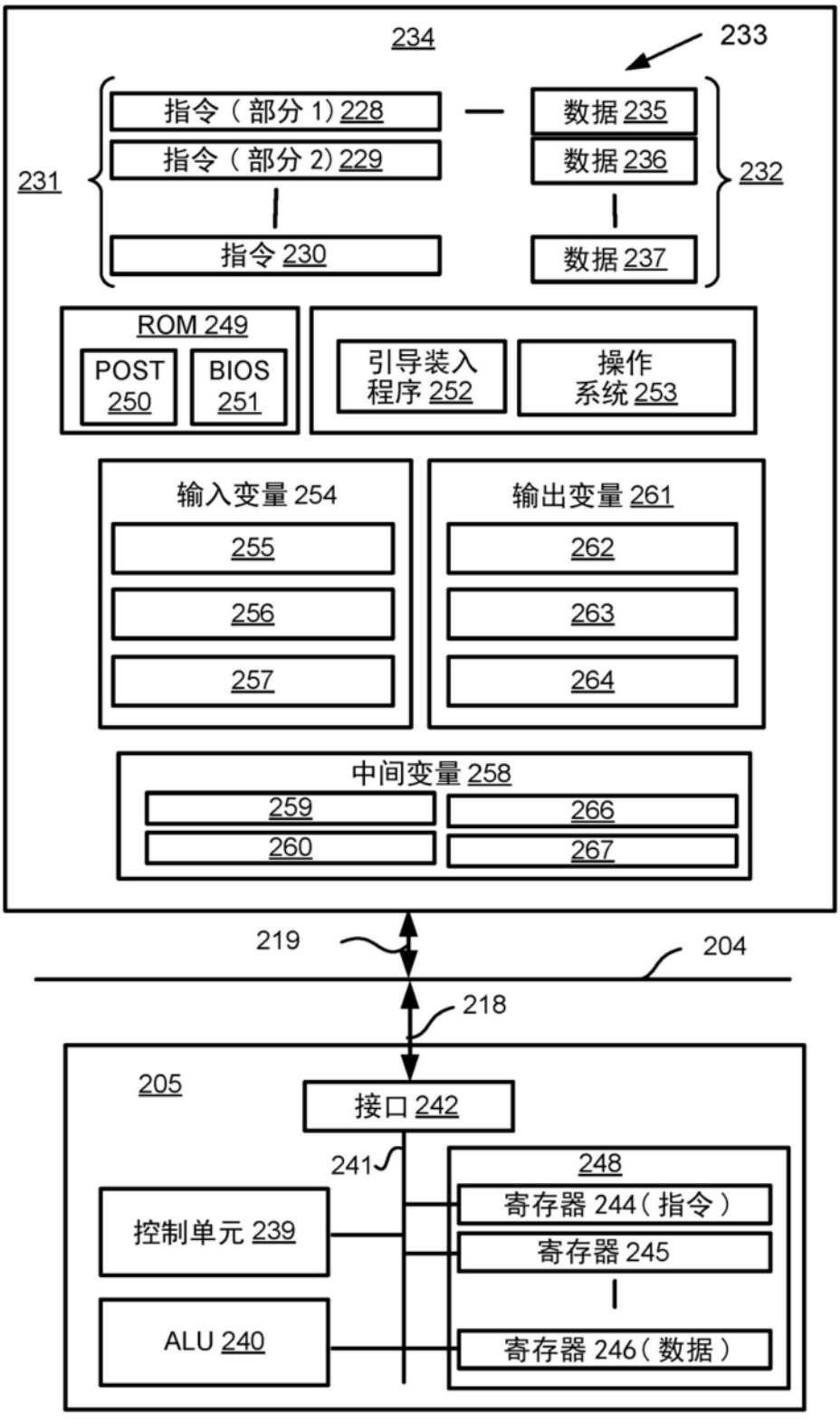

234
233
指令（部分 1）228
指令（部分 2）229
231
指令 230
数据 235
数据 236
232
数据 237
ROM 249
POST 250
BIOS 251
引导装入程序 252
操作系统 253
输入变量 254
255
256
257
输出变量 261
262
263
264
中间变量 258
259
260
266
267
219
204
218
205
接口 242
241
248
寄存器 244（指令）
寄存器 245
控制单元 239
ALU 240
寄存器 246（数据）

图2B

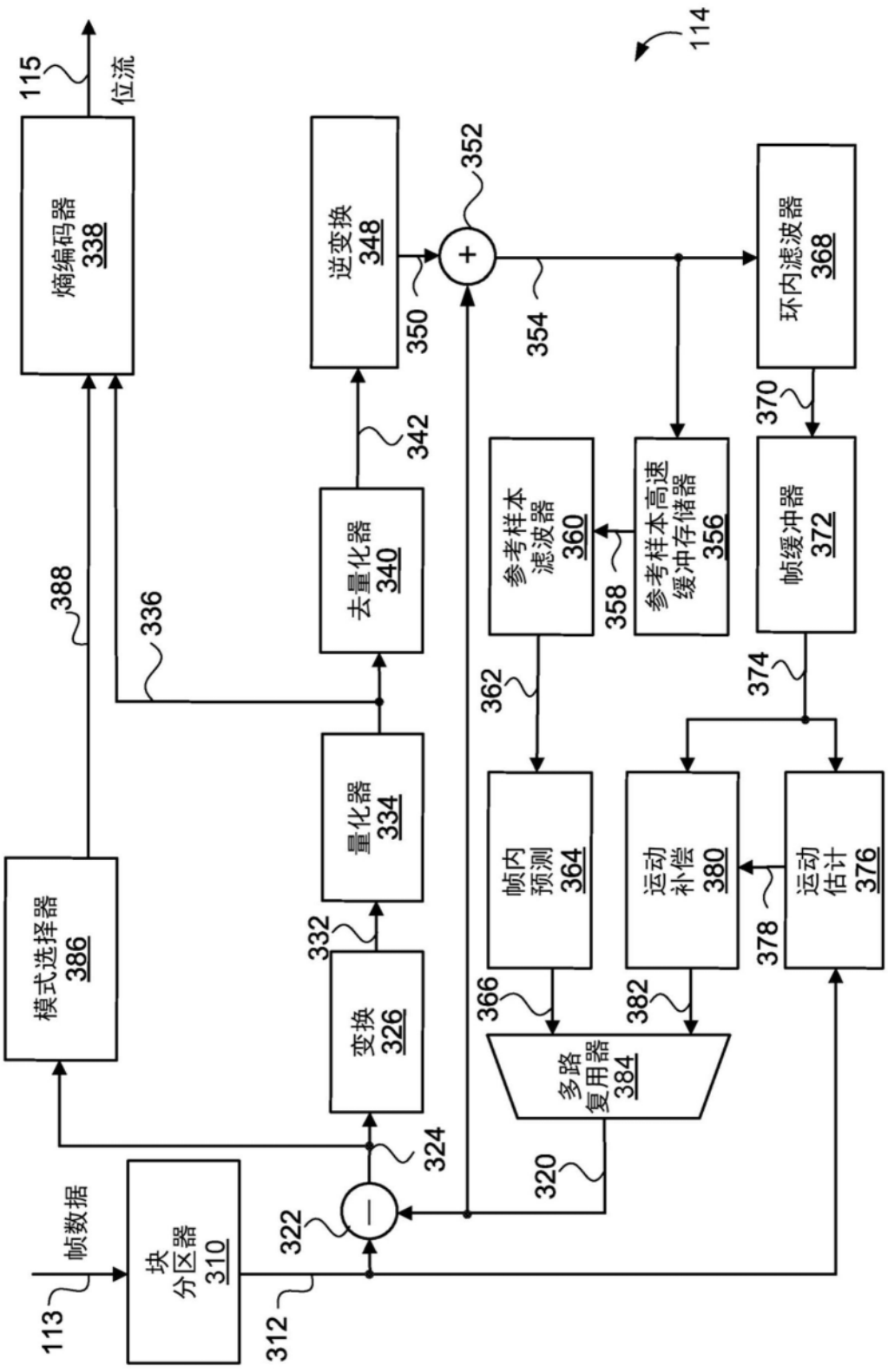

114
115
位流
熵编码器 338
逆变换 348
352
环内滤波器 368
350
354
370
388
342
参考样本滤波器 360
参考样本高速缓冲存储器 356
帧缓冲器 372
336
去量化器 340
358
374
362
量化器 334
模式选择器 386
帧内预测 364
运动补偿 380
运动估计 376
332
变换 326
366
382
378
多路复用器 384
324
320
帧数据
322
块分区器 310
113
312

图3

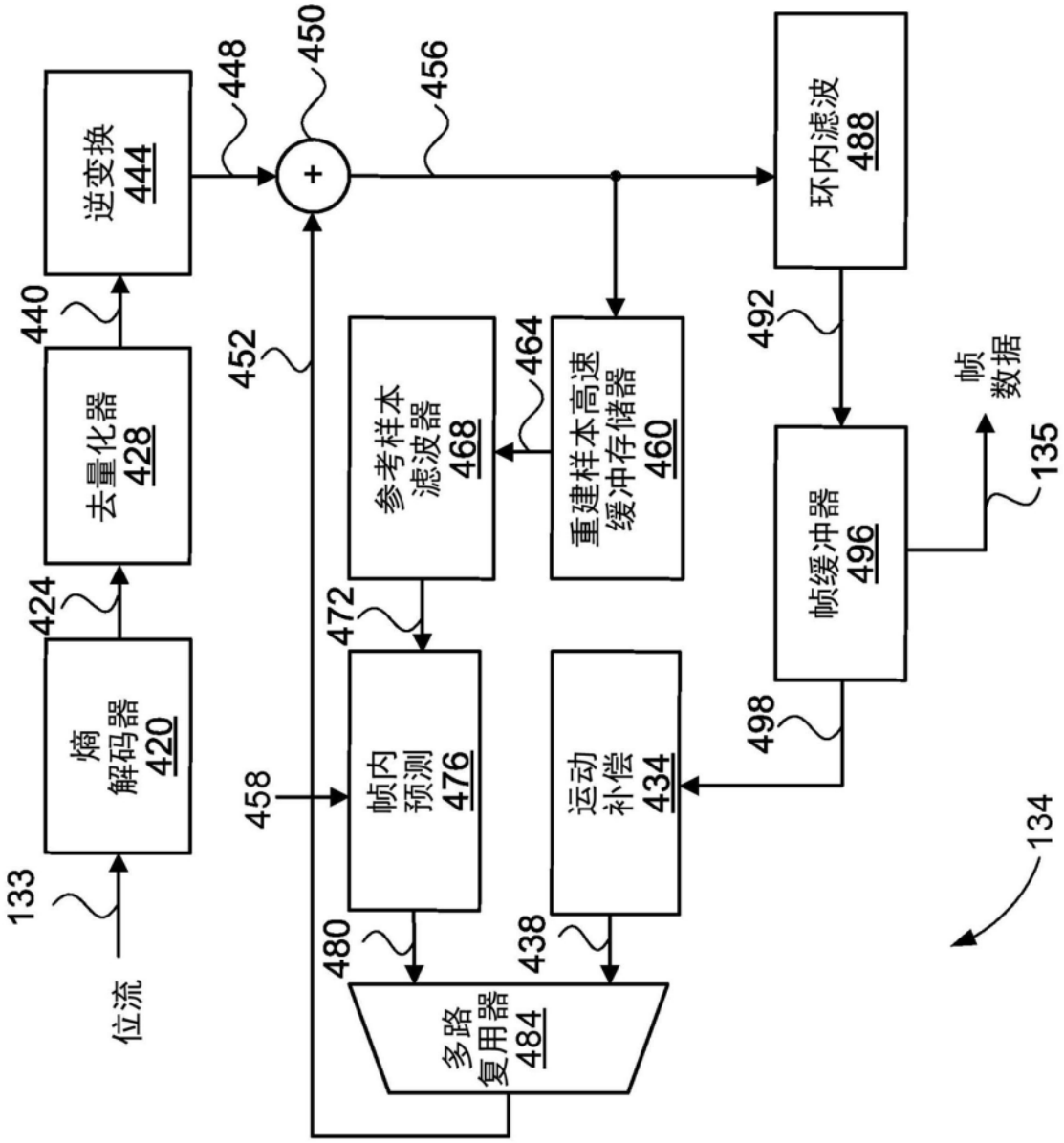

逆变换 444
448
450
456
环内滤波 488
去量化器 428
440
452
参考样本滤波器 468
464
重建样本高速缓冲存储器 460
492
帧数据
135
熵解码器 420
424
472
帧缓冲器 496
帧内预测 476
458
运动补偿 434
498
134
133
位流
480
438
多路复用器 484

图4

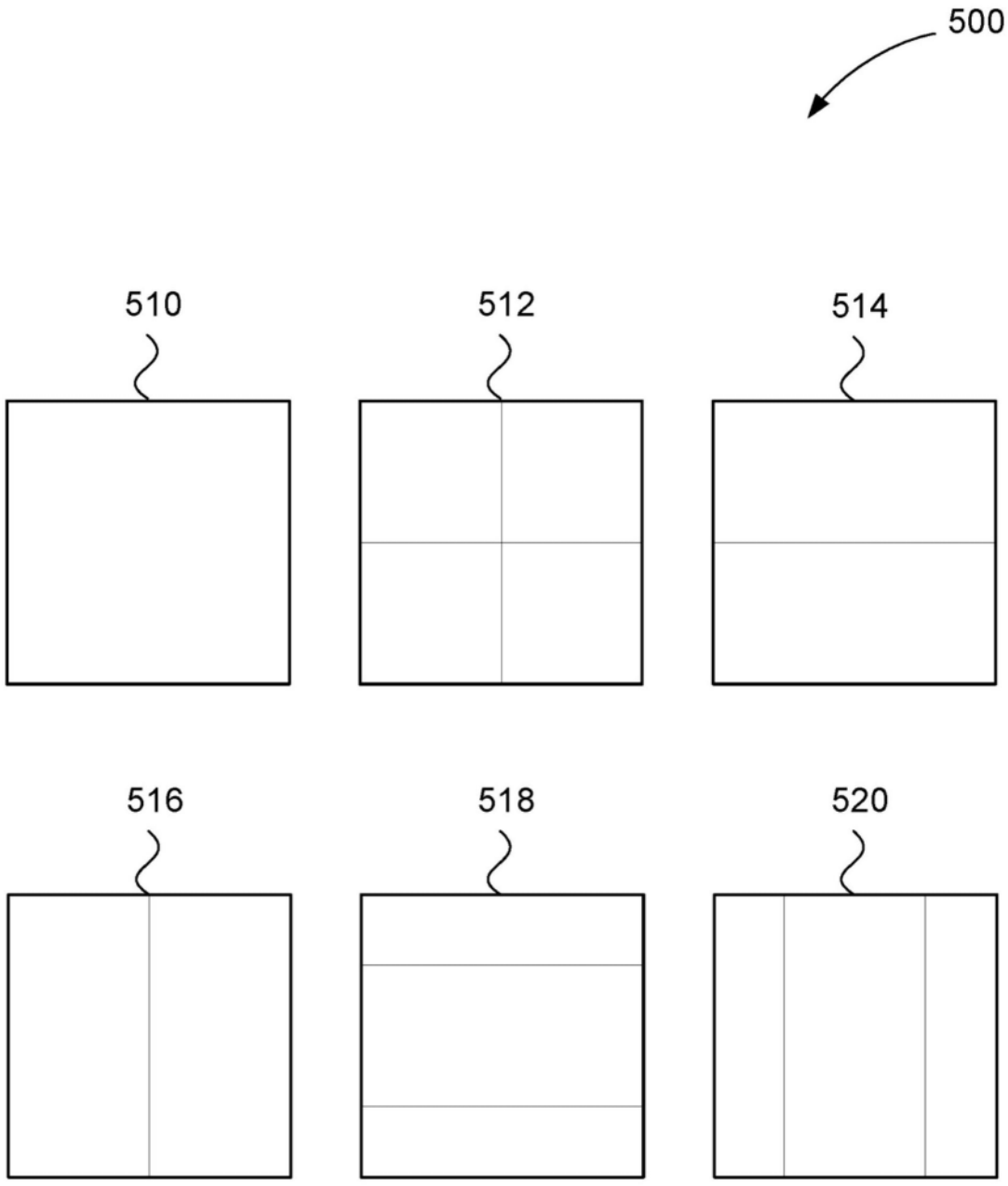


图5

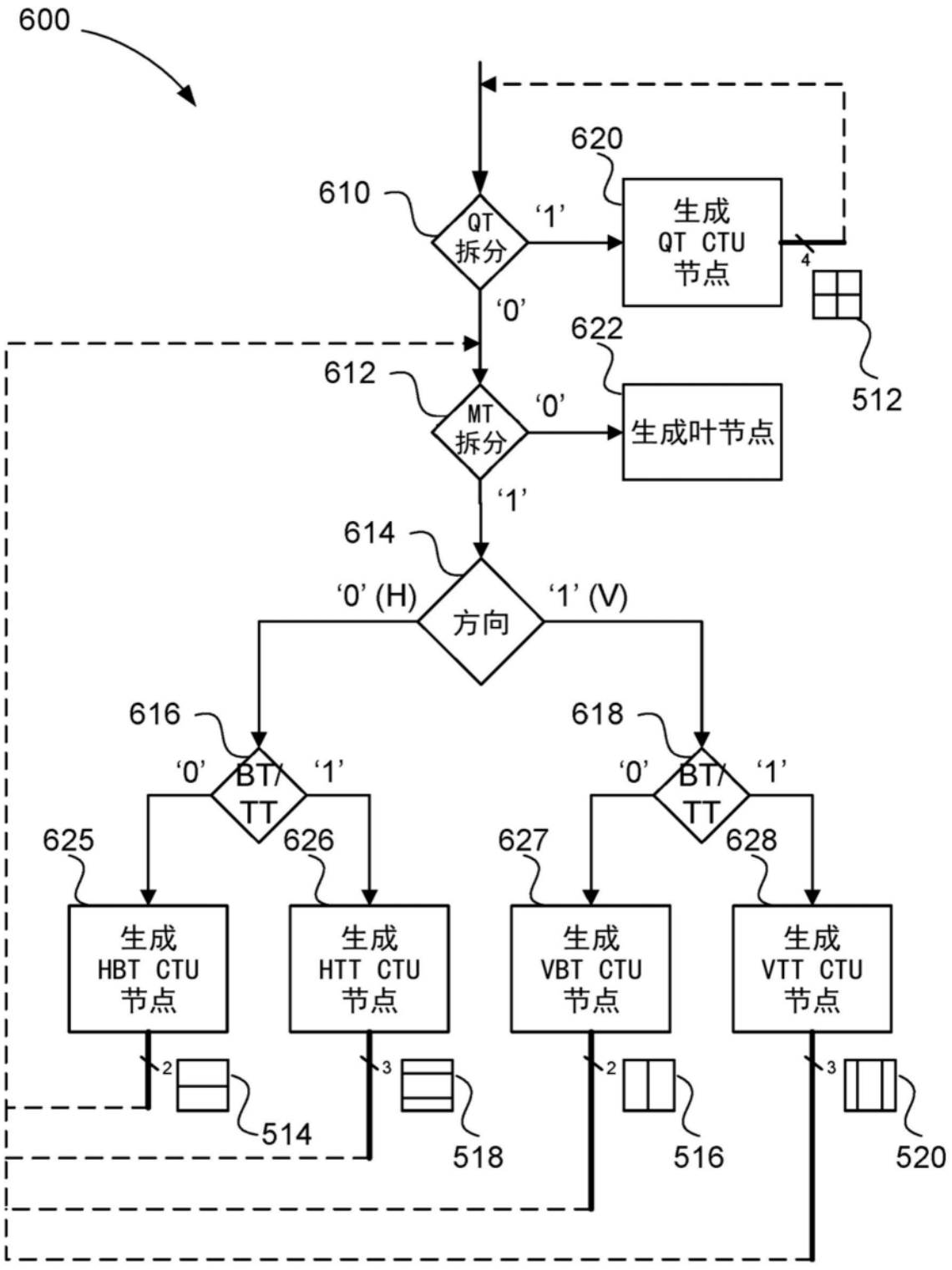

600
610
QT
拆分
'1'
620
生成
QT CTU
节点
4
512
'0'
612
MT
拆分
'0'
622
生成叶节点
'1'
614
'0' (H)
方向
'1' (V)
616
'0'
BT/
TT
'1'
618
'0'
BT/
TT
'1'
625
626
627
628
生成
HBT CTU
节点
生成
HTT CTU
节点
生成
VBT CTU
节点
生成
VTT CTU
节点
2
3
2
3
514
518
516
520

图6

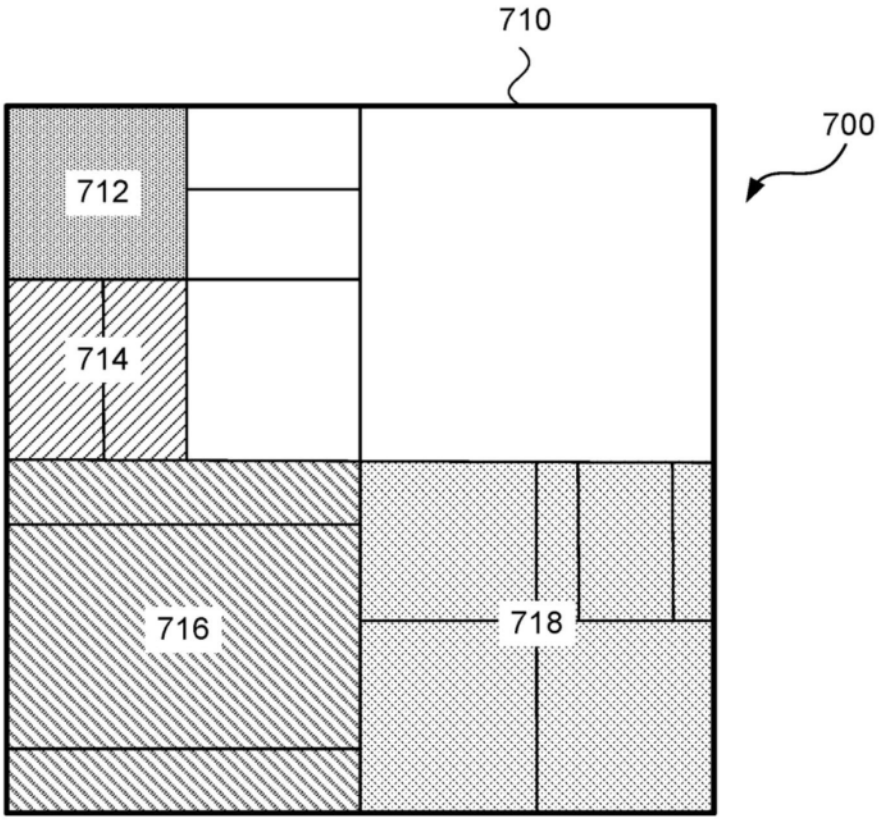

710
700
712
714
716
718

图7A

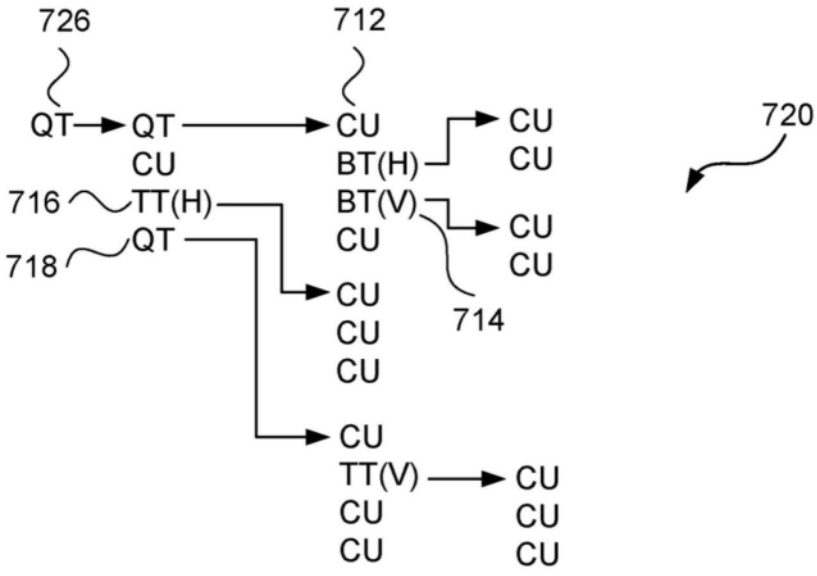

726
712
720
QT
QT
CU
CU
BT(H)
CU
CU
716
TT(H)
BT(V)
CU
CU
718
QT
CU
CU
CU
CU
714
CU
TT(V)
CU
CU
CU
CU
CU

图7B

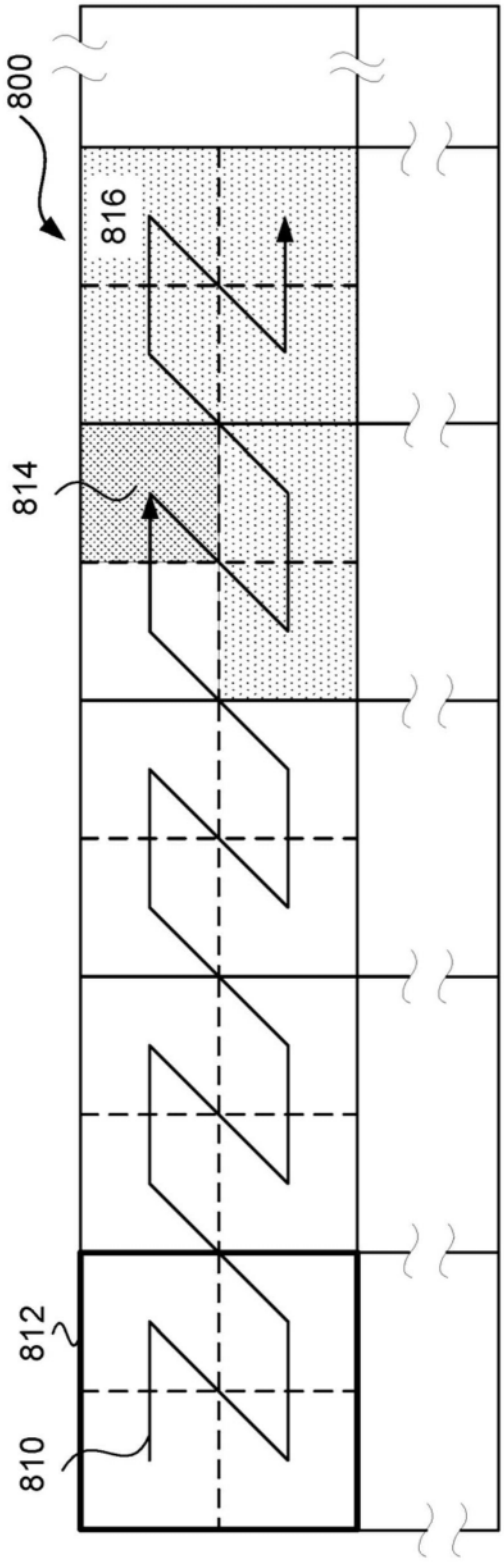


图8A

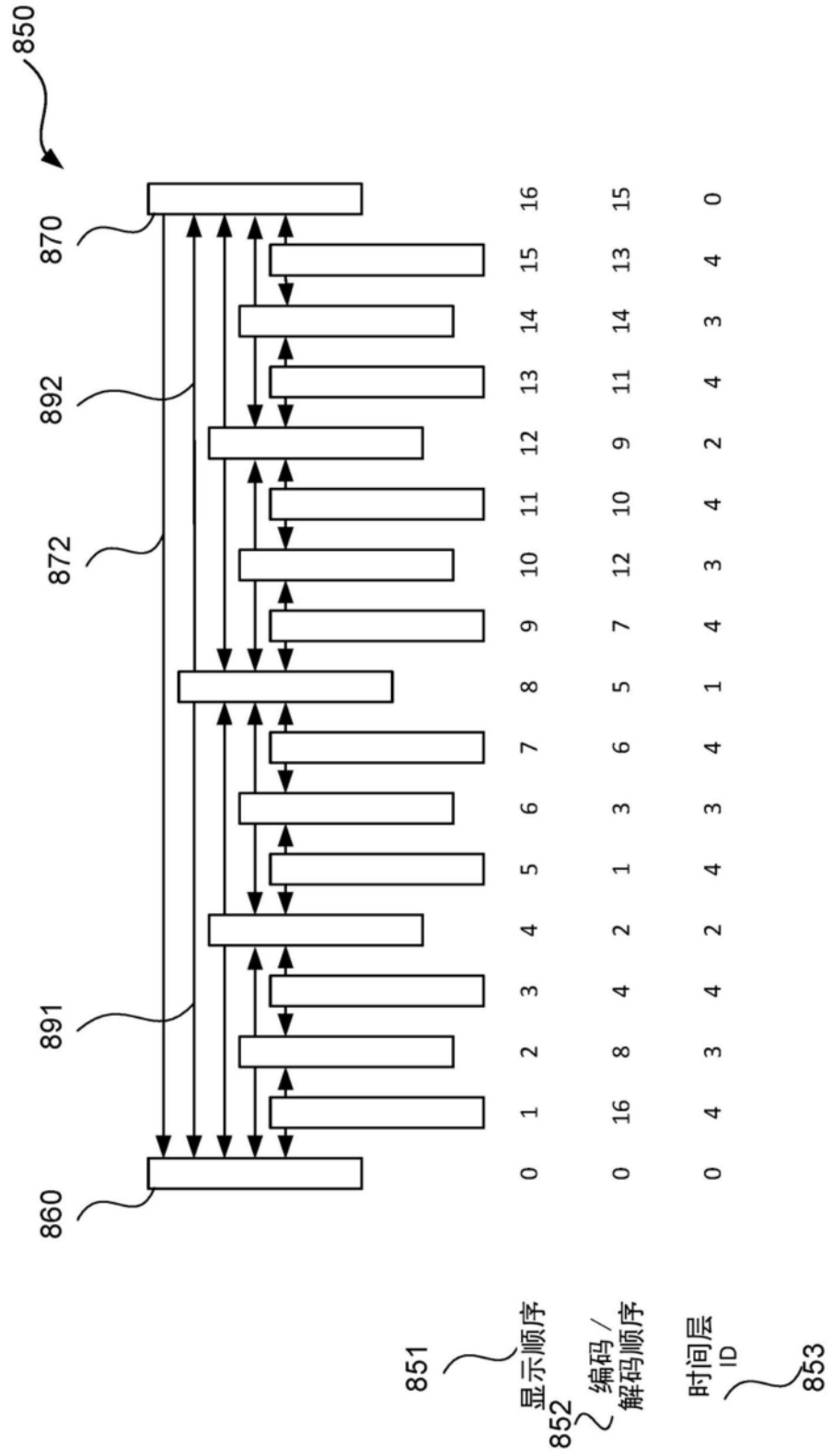

850
870
892
872
891
860
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
0 16 8 4 2 1 3 6 5 7 9 10 12 11 13 14 15
0 4 3 4 2 4 3 4 1 4 3 4 2 4 2 4 3 4 0
851
显示顺序
852
编码/解码顺序
时间层ID
853

图8B

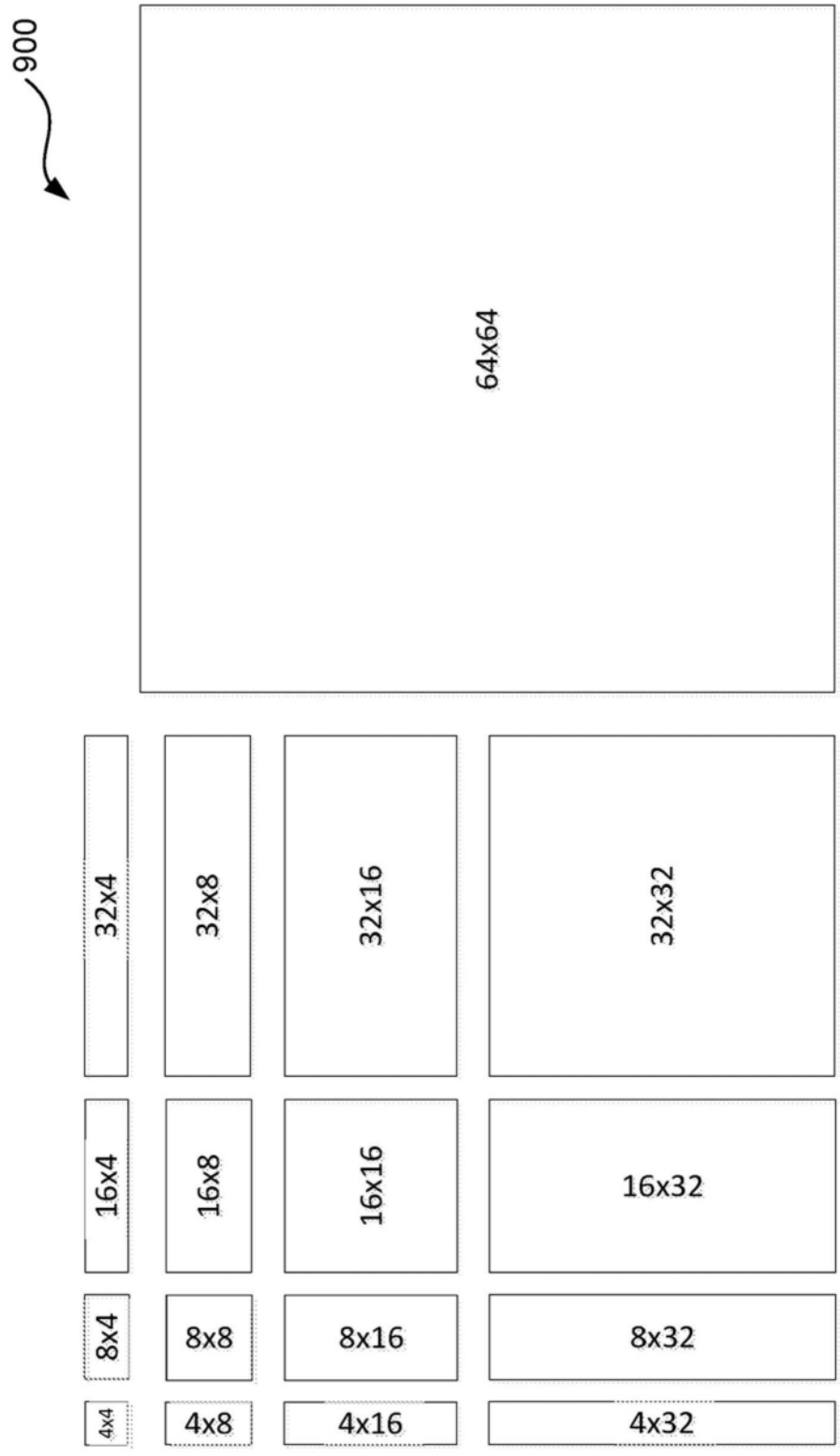

900
64x64
32x4
32x8
32x16
32x32
16x4
16x8
16x16
16x32
8x4
8x8
8x16
8x32
4x4
4x8
4x16
4x32

图9

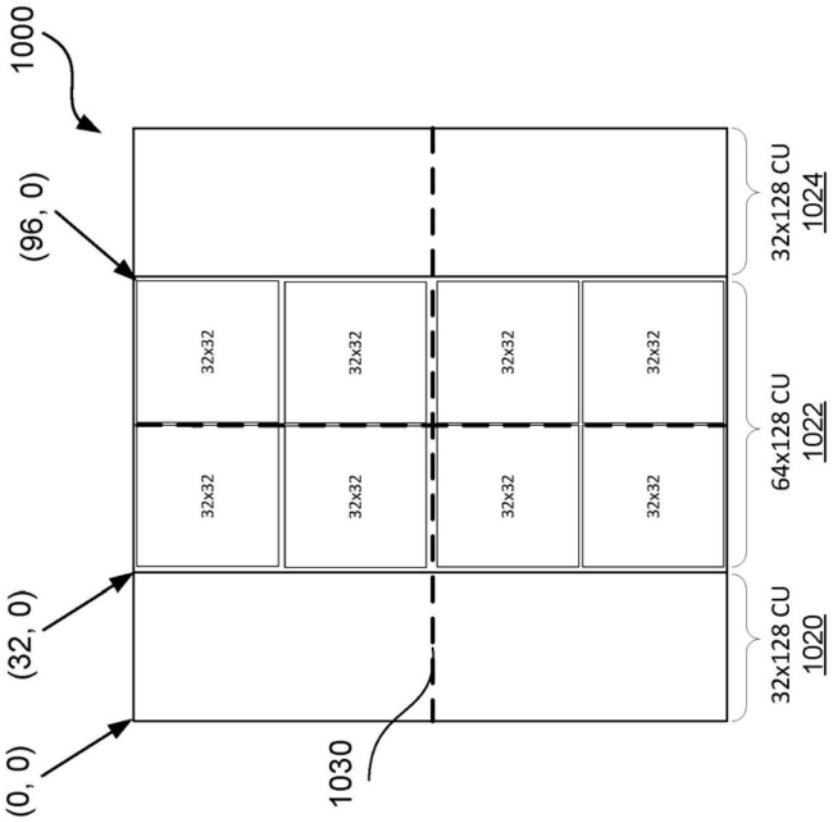

1000
(96, 0)
(32, 0)
(0, 0)
1030
32x32
32x128 CU
1024
64x128 CU
1022
32x128 CU
1020

图10A

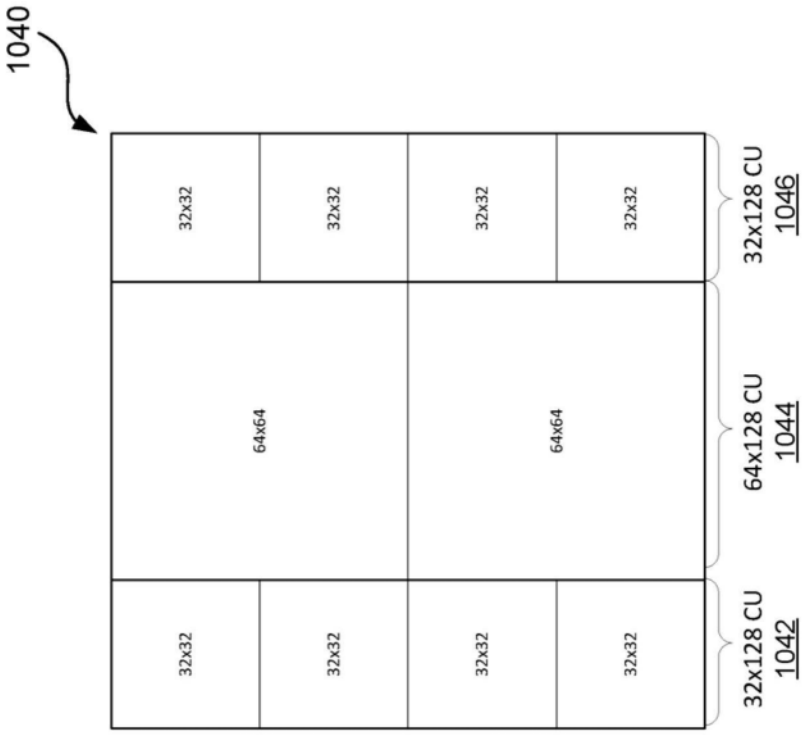

1040
32x32
64x64
32x128 CU
1046
64x128 CU
1044
32x128 CU
1042

图10B

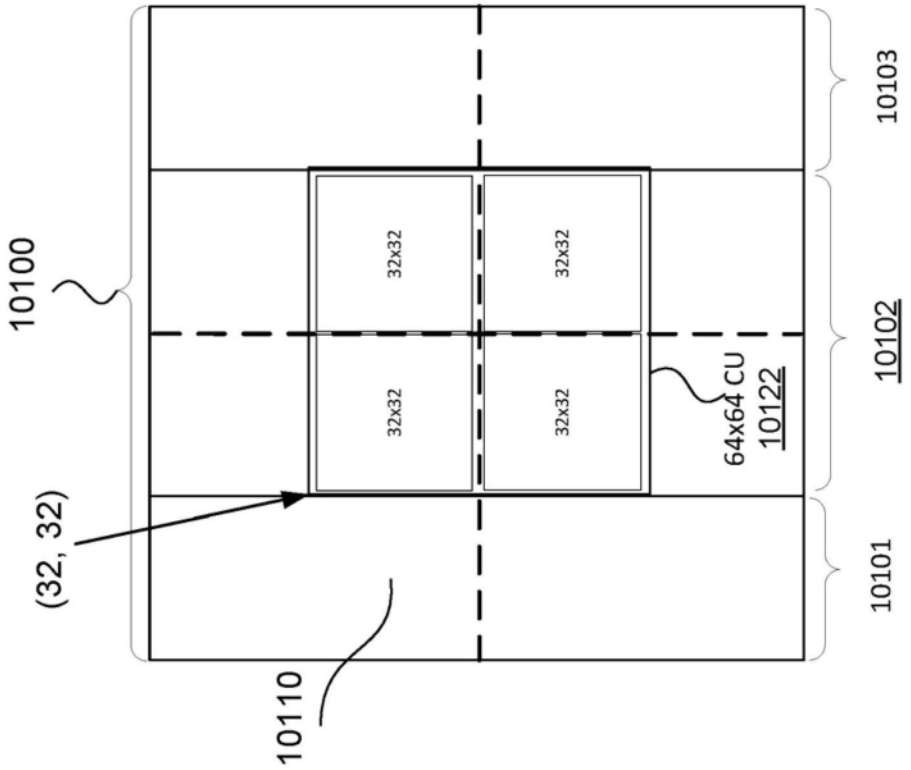

10100
10103
10102
10101
32x32
32x32
32x32
32x32
64x64 CU
10122
(32, 32)
10110

图10C

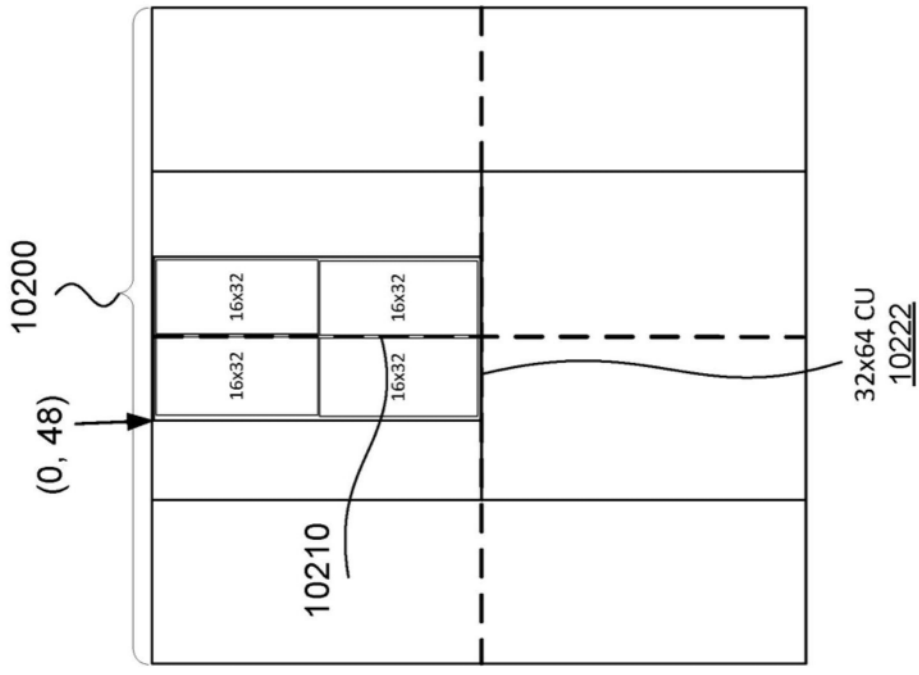

10200
16x32
16x32
16x32
16x32
32x64 CU
10222
(0, 48)
10210

图10D

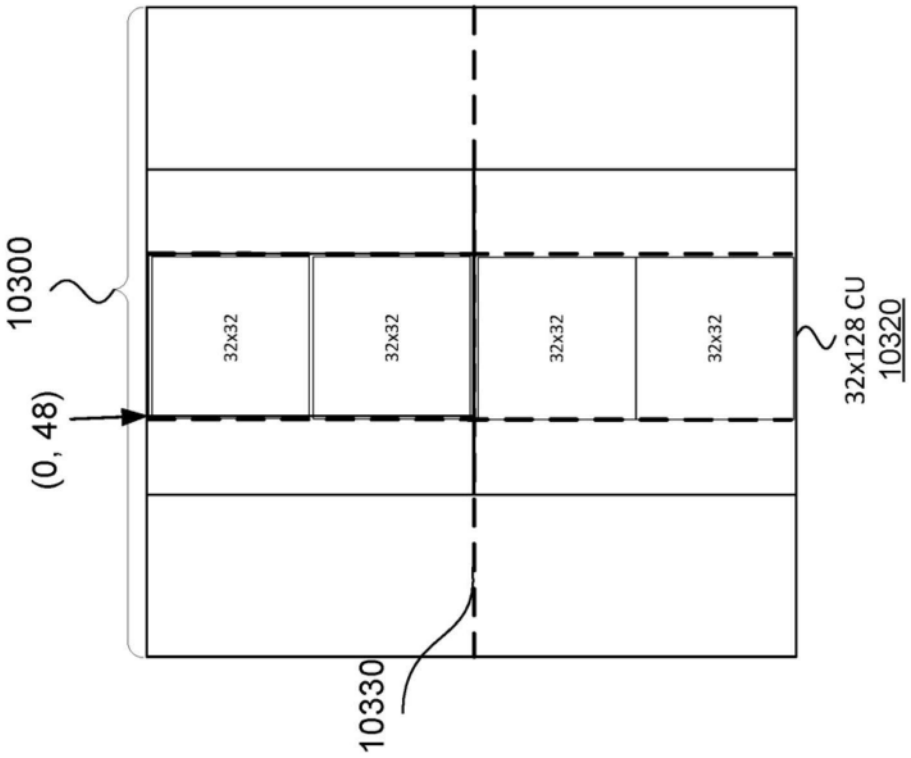

10300
(0, 48)
32x32
32x32
32x32
32x32
32x128 CU
10320
10330

图10E

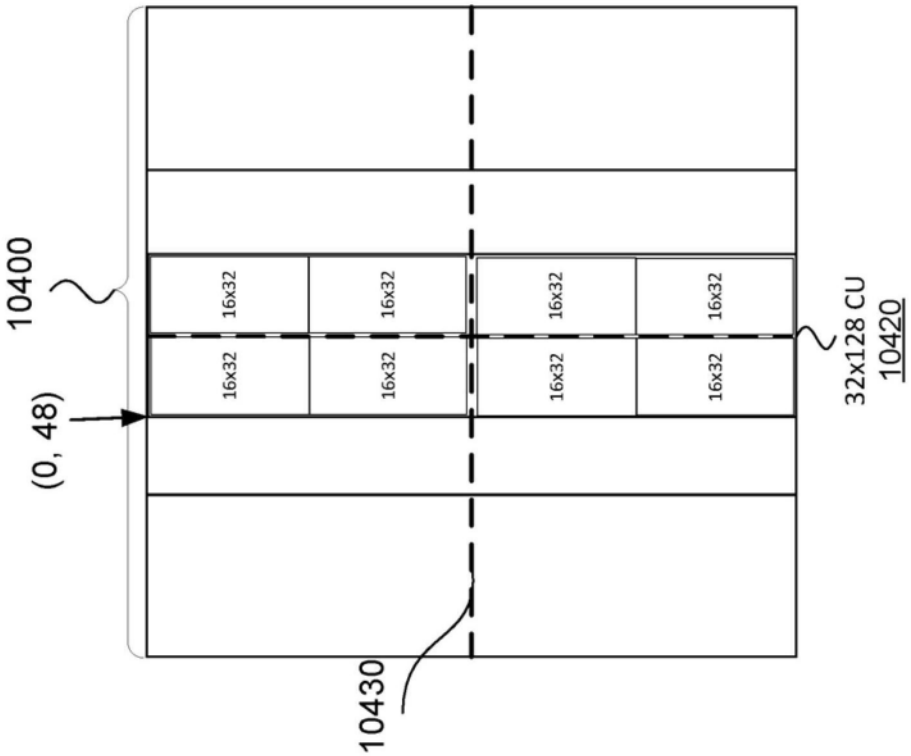

10400
(0, 48)
16x32
16x32
16x32
16x32
16x32
16x32
16x32
16x32
32x128 CU
10420
10430

图10F

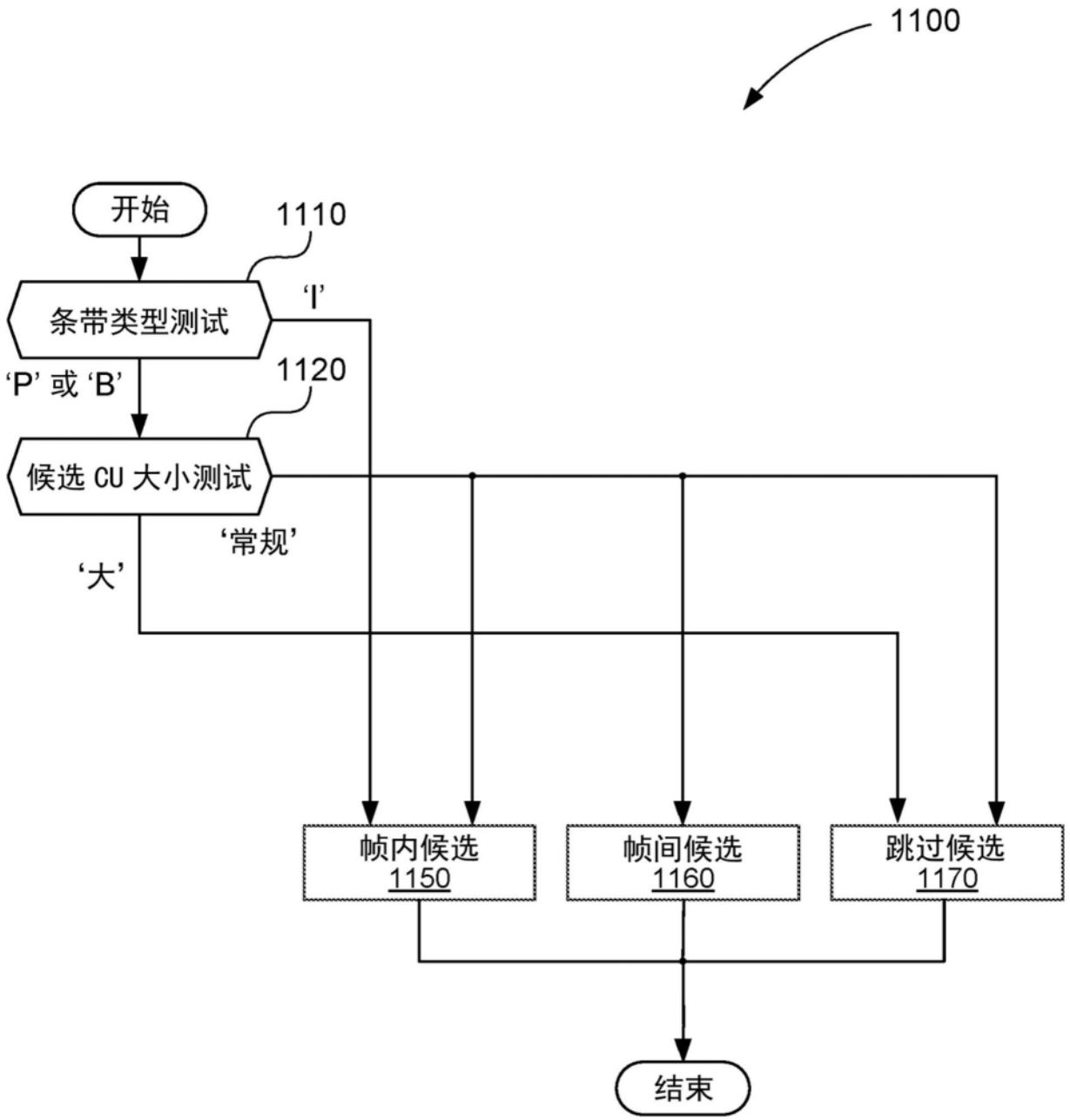

1100
开始
1110
条带类型测试
'I'
'P' 或 'B'
1120
候选 CU 大小测试
'常规'
'大'
帧内候选
1150
帧间候选
1160
跳过候选
1170
结束

图11

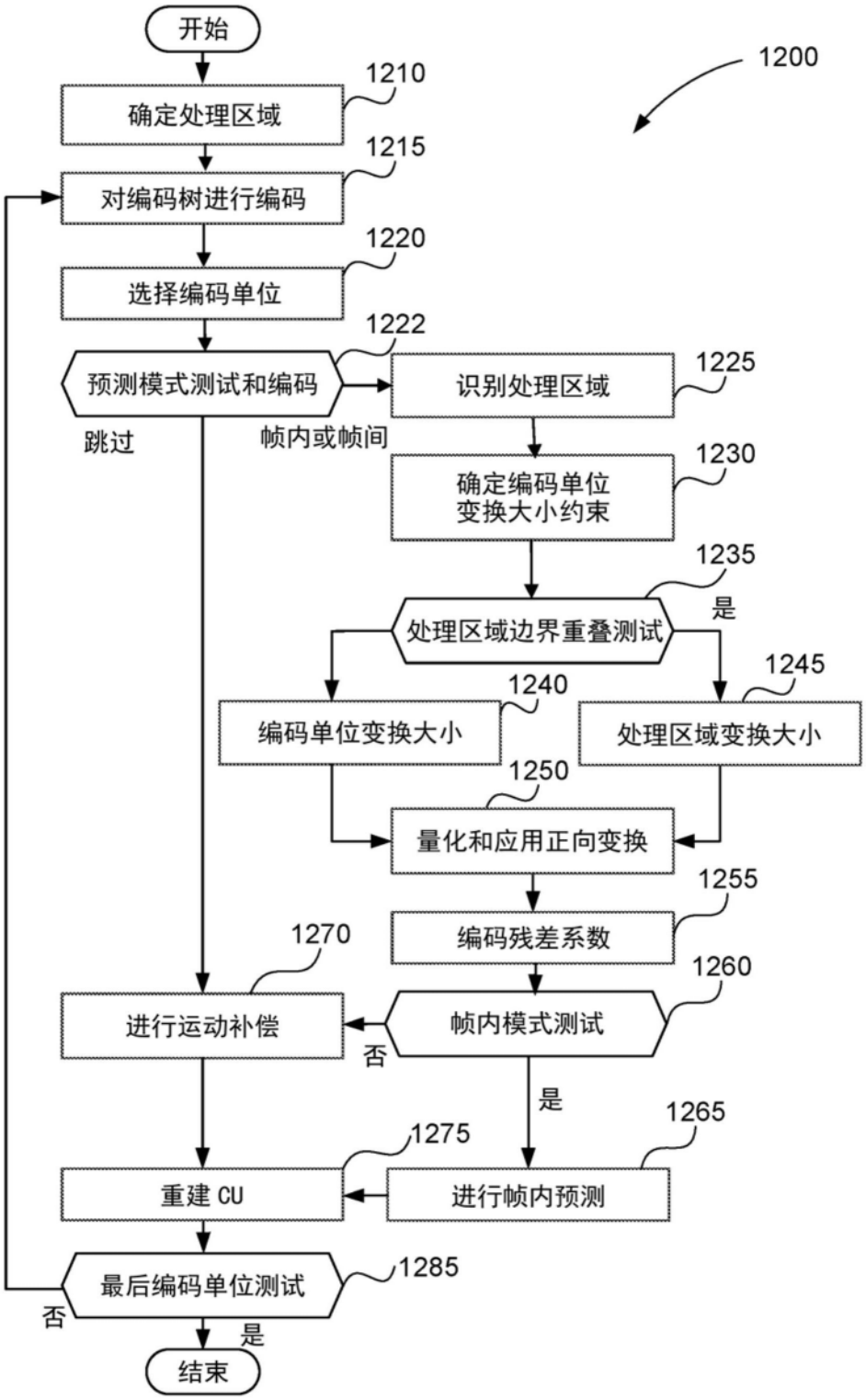

开始
1200
1210
确定处理区域
1215
对编码树进行编码
1220
选择编码单位
1222
预测模式测试和编码
跳过
帧内或帧间
1225
识别处理区域
1230
确定编码单位
变换大小约束
1235
处理区域边界重叠测试
是
1240
编码单位变换大小
1245
处理区域变换大小
1250
量化和应用正向变换
1255
编码残差系数
1260
帧内模式测试
否
是
1265
进行帧内预测
1270
进行运动补偿
1275
重建 CU
1285
最后编码单位测试
否
是
结束

图12

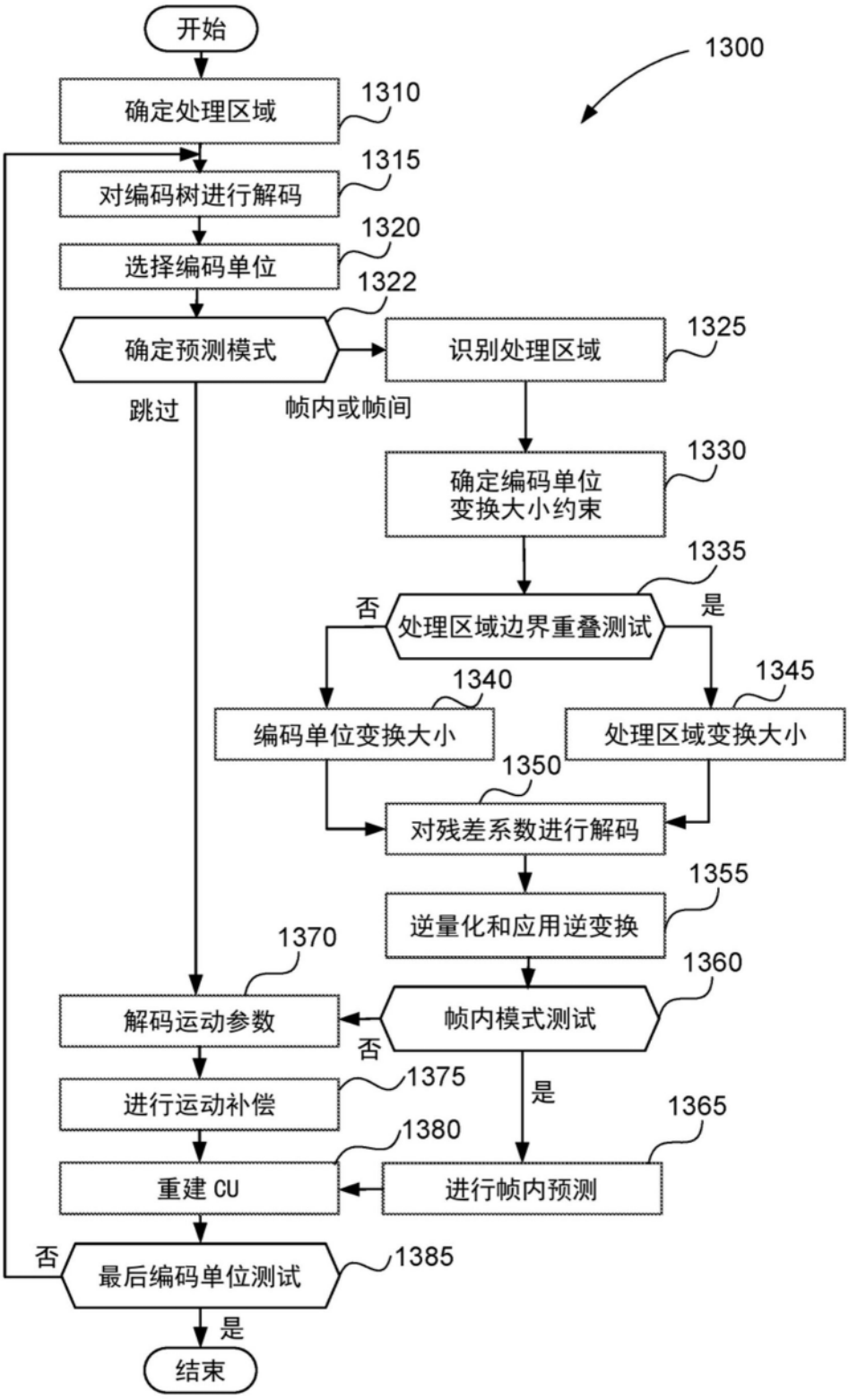

开始
1300
确定处理区域
1310
对编码树进行解码
1315
选择编码单位
1320
确定预测模式
1322
识别处理区域
1325
跳过
帧内或帧间
确定编码单位
变换大小约束
1330
处理区域边界重叠测试
1335
否
是
编码单位变换大小
1340
处理区域变换大小
1345
对残差系数进行解码
1350
逆量化和应用逆变换
1355
解码运动参数
1370
帧内模式测试
1360
否
是
进行运动补偿
1375
重建 CU
1380
进行帧内预测
1365
否
最后编码单位测试
1385
是
结束

图13