(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 119110081 A
(43)申请公布日 2024.12.10

(21)申请号 202411176692.5
(22)申请日 2020.01.20
(30)优先权数据
2019201653 2019.03.11 AU
(62)分案原申请数据
202080020292.9 2020.01.20
(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
(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 19/176(2014.01)
H04N 19/20(2014.01)
H04N 19/60(2014.01)
H04N 19/42(2014.01)

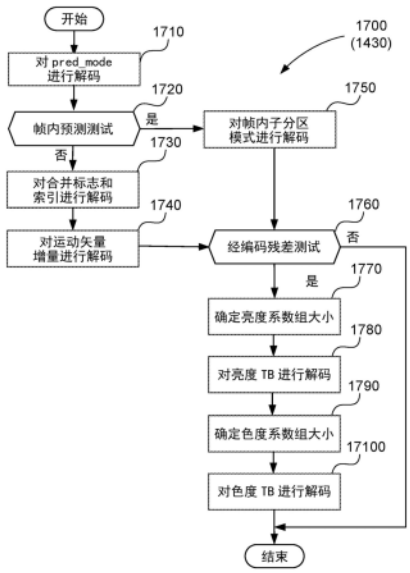
权利要求书4页 说明书35页 附图19页

(54)发明名称

用于编码和解码变换块的方法和设备

(57)摘要

本发明涉及一种用于编码和解码变换块的方法和设备。还涉及一种从视频位流中解码图像帧的颜色通道的变换块的系统和方法。该方法包括:确定所述图像帧的色度格式,所述色度格式使得相对于所述图像帧的亮度通道对所述图像帧的色度通道进行子采样;确定所述变换块的系数组大小,所述系数组大小是多达16个样本的变换块的最大区域,所述系数组大小是仅基于变换块大小、并且与(i)所述变换块的颜色平面和(ii)由于所确定的色度格式而引起的颜色平面子采样这两者无关地来确定的;以及使用所确定的大小的系数组来从所述视频位流中解码所述变换块。



CN 119110081 A

1.一种从位流中解码图像帧的变换块的方法,所述方法包括:

从包括4:2:0色度格式和4:2:2色度格式的多个色度格式中确定所述图像帧的色度格式,所述4:2:0色度格式和所述4:2:2色度格式中的各色度格式使得相对于所述图像帧的亮度通道对所述图像帧的色度通道进行子采样;

将编码树单元拆分为一个或多于一个编码单元,所述一个或多于一个编码单元各自包括亮度编码块和色度编码块,其中在通过水平二元拆分将所述编码树单元中的区域拆分成编码单元的情况下,所述编码单元由第一编码单元以及与所述第一编码单元的底部相邻的第二编码单元组成;

确定所述变换块的子块,所述子块是亮度编码块的亮度变换块或色度编码块的色度变换块,所述子块是用于扫描所述变换块中的系数的单元;以及

使用所述子块从所述位流中解码所述变换块,

其中,在包括具有所述4:2:0色度格式的所述编码树单元中的亮度编码块和色度编码块的给定编码单元中的色度编码块的块大小为8×2的情况下,即使对所述给定编码单元中的所述亮度编码块进行垂直三元拆分,也不允许对所述给定编码单元中的所述色度编码块进行垂直三元拆分,

其中,在通过垂直三元拆分对所述给定编码单元中的所述亮度编码块进行拆分、并且未通过垂直三元拆分对所述给定编码单元中的所述色度编码块进行拆分的情况下,所述给定编码单元中的所述色度编码块被排列在与通过对所述给定编码单元中的所述亮度编码块进行垂直三元拆分所获得的三个亮度编码块相对应的位置处,

其中,基于(i)所述变换块的宽度和高度、以及(ii)所述变换块是亮度变换块还是色度变换块来解码所述变换块,以及

其中,在所述变换块的宽度的log2是4并且所述变换块的高度的log2是4的情况下,在确定所述变换块的子块的宽度的log2和所述变换块的子块的高度的log2时,从(i)所述变换块的宽度的log2和所述变换块的高度的log2,并且与(ii)所述变换块是亮度变换块还是色度变换块以及(iii)所述图像帧的色度格式这两者无关地,确定所述变换块的子块的宽度的log2和所述变换块的子块的高度的log2。

2.根据权利要求1所述的方法,

其中,在具有所述4:2:0色度格式的所述编码树单元中的所述给定编码单元中的所述色度编码块的所述块大小是预定大小的情况下,即使所述给定编码单元中的所述亮度编码块被拆分,也不允许对所述给定编码单元中的所述色度编码块进行进一步拆分,以及

其中,在所述给定编码单元中的所述色度编码块的所述块大小是所述预定大小、并且所述给定编码单元的当前拆分模式是四叉拆分的情况下,将帧内预测确定为用于所述给定编码单元的所述色度编码块。

3.根据权利要求1所述的方法,其中,在所述变换块的宽度的log2是4并且所述变换块的高度的log2是4的情况下,所确定的所述变换块的子块的宽度的log2是2并且所确定的所述变换块的子块的高度的log2是2。

4.根据权利要求1所述的方法,其中,所述8×2指示宽度8和高度2。

5.根据权利要求1所述的方法,其中,在包括具有所述4:2:0色度格式的所述编码树单元中的所述亮度编码块和所述色度编码块的所述给定编码单元中的色度编码块的所述块

大小为8×2的情况下,所述给定编码单元中的所述亮度编码块的块大小是16×4,其中8×2指示宽度8和高度2,16×4指示宽度16和高度4。

6.根据权利要求1所述的方法,其中,所述亮度通道是Y分量,并且所述色度通道是包括Cb分量和Cr分量的色度分量。

7.根据权利要求1所述的方法,其中,所述第一编码单元和所述第二编码单元之间的边界是水平的。

8.一种从位流中解码图像帧的变换块的设备,所述设备包括:

第一确定单元,其被配置为从包括4:2:0色度格式和4:2:2色度格式的多个色度格式中确定所述图像帧的色度格式,所述4:2:0色度格式和所述4:2:2色度格式中的各色度格式使得相对于所述图像帧的亮度通道对所述图像帧的色度通道进行子采样;

拆分单元,其被配置为将编码树单元拆分为一个或多于一个编码单元,所述一个或多于一个编码单元各自包括亮度编码块和色度编码块,其中在通过水平二元拆分将所述编码树单元中的区域拆分成编码单元的情况下,所述编码单元由第一编码单元以及与所述第一编码单元的底部相邻的第二编码单元组成;

第二确定单元,其被配置为确定所述变换块的子块,所述子块是亮度编码块的亮度变换块或色度编码块的色度变换块,所述子块是用于扫描所述变换块中的系数的单元;以及

解码单元,其被配置为使用所述子块从所述位流中解码所述变换块,

其中,在包括具有所述4:2:0色度格式的所述编码树单元中的亮度编码块和色度编码块的给定编码单元中的色度编码块的块大小为8×2的情况下,即使对所述给定编码单元中的所述亮度编码块进行垂直三元拆分,也不允许对所述给定编码单元中的所述色度编码块进行垂直三元拆分,

其中,在通过垂直三元拆分对所述给定编码单元中的所述亮度编码块进行拆分、并且未通过垂直三元拆分对所述给定编码单元中的所述色度编码块进行拆分的情况下,所述给定编码单元中的所述色度编码块被排列在与通过对所述给定编码单元中的所述亮度编码块进行垂直三元拆分所获得的三个亮度编码块相对应的位置处,

其中,基于(i)所述变换块的宽度和高度、以及(ii)所述变换块是亮度变换块还是色度变换块来解码所述变换块,以及

其中,在所述变换块的宽度的log2是4并且所述变换块的高度的log2是4的情况下,在确定所述变换块的子块的宽度的log2和所述变换块的子块的高度的log2时,从(i)所述变换块的宽度的log2和所述变换块的高度的log2,并且与(ii)所述变换块是亮度变换块还是色度变换块以及(iii)所述图像帧的色度格式这两者无关地,确定所述变换块的子块的宽度的log2和所述变换块的子块的高度的log2。

9.一种将图像帧的变换块编码到位流中的方法,所述方法包括:

从包括4:2:0色度格式和4:2:2色度格式的多个色度格式中确定所述图像帧的色度格式,所述4:2:0色度格式和所述4:2:2色度格式中的各色度格式使得相对于所述图像帧的亮度通道对所述图像帧的色度通道进行子采样;

将编码树单元拆分为一个或多于一个编码单元,所述一个或多于一个编码单元各自包括亮度编码块和色度编码块,其中在通过水平二元拆分将所述编码树单元中的区域拆分成编码单元的情况下,所述编码单元由第一编码单元以及与所述第一编码单元的底部相邻的

第二编码单元组成;

确定所述变换块的子块,所述子块是亮度编码块的亮度变换块或色度编码块的色度变换块,所述子块是用于扫描所述变换块中的系数的单元;以及

使用所述子块将所述变换块编码到所述位流中,

其中,在包括具有所述4:2:0色度格式的所述编码树单元中的亮度编码块和色度编码块的给定编码单元中的色度编码块的块大小为8×2的情况下,即使对所述给定编码单元中的所述亮度编码块进行垂直三元拆分,也不允许对所述给定编码单元中的所述色度编码块进行垂直三元拆分,

其中,在通过垂直三元拆分对所述给定编码单元中的所述亮度编码块进行拆分、并且未通过垂直三元拆分对所述给定编码单元中的所述色度编码块进行拆分的情况下,所述给定编码单元中的所述色度编码块被排列在与通过对所述给定编码单元中的所述亮度编码块进行垂直三元拆分所获得的三个亮度编码块相对应的位置处,

其中,基于(i)所述变换块的宽度和高度、以及(ii)所述变换块是亮度变换块还是色度变换块来编码所述变换块,以及

其中,在所述变换块的宽度的log2是4并且所述变换块的高度的log2是4的情况下,在确定所述变换块的子块的宽度的log2和所述变换块的子块的高度的log2时,从(i)所述变换块的宽度的log2和所述变换块的高度的log2,并且与(ii)所述变换块是亮度变换块还是色度变换块以及(iii)所述图像帧的色度格式这两者无关地,确定所述变换块的子块的宽度的log2和所述变换块的子块的高度的log2。

10.根据权利要求9所述的方法,

其中,在具有所述4:2:0色度格式的所述编码树单元中的所述给定编码单元中的所述色度编码块的所述块大小是预定大小的情况下,即使所述给定编码单元中的所述亮度编码块被拆分,也不允许对所述给定编码单元中的所述色度编码块进行进一步拆分,以及

其中,在所述给定编码单元中的所述色度编码块的所述块大小是所述预定大小、并且所述给定编码单元的当前拆分模式是四叉拆分的情况下,将帧内预测确定为用于所述给定编码单元的所述色度编码块。

11.根据权利要求9所述的方法,其中,在所述变换块的宽度的log2是4并且所述变换块的高度的log2是4的情况下,所确定的所述变换块的子块的宽度的log2是2并且所确定的所述变换块的子块的高度的log2是2。

12.根据权利要求9所述的方法,其中,所述8×2指示宽度8和高度2。

13.根据权利要求9所述的方法,其中,在包括具有所述4:2:0色度格式的所述编码树单元中的所述亮度编码块和所述色度编码块的所述给定编码单元中的色度编码块的所述块大小为8×2的情况下,所述给定编码单元中的所述亮度编码块的块大小是16×4,其中8×2指示宽度8和高度2,16×4指示宽度16和高度4。

14.根据权利要求9所述的方法,其中,所述亮度通道是Y分量,并且所述色度通道是包括Cb分量和Cr分量的色度分量。

15.根据权利要求9所述的方法,其中,顶部的所述第一编码单元和底部的所述第二编码单元之间的边界是水平的。

16.一种将图像帧的变换块编码到位流中的设备,所述设备包括:

第一确定单元,其被配置为从包括4:2:0色度格式和4:2:2色度格式的多个色度格式中确定所述图像帧的色度格式,所述4:2:0色度格式和所述4:2:2色度格式中的各色度格式使得相对于所述图像帧的亮度通道对所述图像帧的色度通道进行子采样;

拆分单元,其被配置为将编码树单元拆分为一个或多于一个编码单元,所述一个或多于一个编码单元各自包括亮度编码块和色度编码块,其中在通过水平二元拆分将所述编码树单元中的区域拆分成编码单元的情况下,所述编码单元由第一编码单元以及与所述第一编码单元的底部相邻的第二编码单元组成;

第二确定单元,其被配置为确定所述变换块的子块,所述子块是亮度编码块的亮度变换块或色度编码块的色度变换块,所述子块是用于扫描所述变换块中的系数的单元;以及

编码单元,其被配置为使用所述子块将所述变换块编码到所述位流中,

其中,在包括具有所述4:2:0色度格式的所述编码树单元中的亮度编码块和色度编码块的给定编码单元中的色度编码块的块大小为8×2的情况下,即使对所述给定编码单元中的所述亮度编码块进行垂直三元拆分,也不允许对所述给定编码单元中的所述色度编码块进行垂直三元拆分,

其中,在通过垂直三元拆分对所述给定编码单元中的所述亮度编码块进行拆分、并且未通过垂直三元拆分对所述给定编码单元中的所述色度编码块进行拆分的情况下,所述给定编码单元中的所述色度编码块被排列在与通过对所述给定编码单元中的所述亮度编码块进行垂直三元拆分所获得的三个亮度编码块相对应的位置处,

其中,基于(i)所述变换块的宽度和高度、以及(ii)所述变换块是亮度变换块还是色度变换块来编码所述变换块,以及

其中,在所述变换块的宽度的log2是4并且所述变换块的高度的log2是4的情况下,在确定所述变换块的子块的宽度的log2和所述变换块的子块的高度的log2时,从(i)所述变换块的宽度的log2和所述变换块的高度的log2,并且与(ii)所述变换块是亮度变换块还是色度变换块以及(iii)所述图像帧的色度格式这两者无关地,确定所述变换块的子块的宽度的log2和所述变换块的子块的高度的log2。

用于编码和解码变换块的方法和设备

[0001] (本申请是申请日为2020年1月20日、申请号为2020800202929、发明名称为“用于编码和解码视频样本的块树的方法、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块树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块树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、存储器利用率和带宽方面)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,需要一个主颜色通道和两个二次颜色通道。主颜色通道通常被称为“亮度”通道,并且(一个或多个)二次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)通道中,并且将色度集中在Cb和Cr(二次)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。4:2:0色度格式通常用于“消费者”应用中,诸如互联网视频流式传输、广播电视以及蓝光™盘上的存储。在水平方向上以半速率对Cb和Cr通道进行子采样而不是垂直地进行子采样被称为“4:2:2色度格式”。4:2:2色度格式通常用于专业应用中,包括用于电影制作等的镜头的捕获。4:2:2色度格式的较高采样速率使得所得视频对编辑操作(诸如颜色分级等)更有弹性。在分发到消费者之前,4:2:2色度格式材料经常被转换为4:2:0色度格式,然后被编码以分发到消费者。除了色度格式之外,视频还由分辨率和

帧频来表征。示例分辨率是具有3840×2160的分辨率的超高清晰度(UD)或具有7680×4320的分辨率的“8K”,并且示例帧频是60Hz或120Hz。亮度样本速率的范围可以从约500兆样本/秒到数千兆样本/秒。对于4:2:0色度格式,各个色度通道的采样率是亮度采样率的四分之一,并且对于4:2:2色度格式,各个色度通道的采样率是亮度采样率的一半。

[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“编码树单元”(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相关联的是针对亮度通道的“编码树”和针对色度通道的附加编码树。编码树定义了将CTU的区域分解为一组区域,也称为“编码块”(CB)。还可以针对单个编码树指定亮度通道和色度通道这两者的块,在这种情况下,并置的编码块的集合被称为“编码单元”(CU),即,各个CU具有用于各个颜色通道的编码块。按特定顺序处理CB以进行编码或解码。作为使用4:2:0色度格式的结果,包括128×128亮度样本区域的亮度编码树的CTU具有与128×128亮度样本区域并置的64×64色度样本区域的相应色度编码树。当单个编码树用于亮度通道和色度通道时,给定区域的并置块的集合通常称为“单元”,例如上述CU以及“预测单元”(PU)和“变换单元”(TU)。当针对给定区域使用单独的编码树时,将使用上述CB以及“预测块”(PB)和“变换块”(TB)。

[0007] 尽管在“单元”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)的预测单元(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并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)的变换。量化变换系数以用于将熵编码在位流中。

[0009] 当空间预测(“帧内预测”)用于生成PB时,一组参考样本用于生成当前PB的预测样本。参考样本包括已经“重建”的与PB相邻的样本(将残差样本添加到帧内预测样本)。这些相邻样本形成PB上方的行和PB左侧的列。行和列还延伸超过PB边界以包括附加的附近样本。由于以Z顺序扫描来扫描块,一些参考样本将在紧挨着的先前块中被重建。使用来自紧挨着的先前块的样本导致反馈依赖,这可能限制通过视频编码器或解码器的块的吞吐量。另外,在从其它帧预测相对小的块(“帧间预测”)的情况下,用于提取参考样本的存储器带宽可能变得过大,尤其是考虑容纳子像素插值滤波所需的附加样本。

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] 本发明的一个方面提供一种从视频位流中解码图像帧的颜色通道的变换块的方法,所述方法包括:确定所述图像帧的色度格式,所述色度格式使得相对于所述图像帧的亮度通道对所述图像帧的色度通道进行子采样;确定所述变换块的系数组大小,所述系数组大小是多达16个样本的变换块的最大区域,所述系数组大小是仅基于变换块大小、并且与

(i)所述变换块的颜色平面和(ii)由于所确定的色度格式而引起的颜色平面子采样这两者无关地来确定的;以及使用所确定的大小的系数组来从所述视频位流中解码所述变换块。

[0012] 根据另一方面,单个表用于属于所述位流的图像帧的亮度颜色平面和色度颜色平面的变换块。

[0013] 根据另一方面,所述系数组大小被选择为在所述变换块的宽度和高度的约束内具有最接近1:1的长宽比。

[0014] 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视频位流中解码图像帧的颜色通道的变换块的方法,所述程序包括:用于确定所述图像帧的色度格式的代码,所述色度格式使得相对于所述图像帧的亮度通道对所述图像帧的色度通道进行子采样;用于确定所述变换块的系数组大小的代码,所述系数组大小是多达16个样本的变换块的最大区域,所述系数组大小是仅基于变换块大小、并且与(i)所述变换块的颜色平面和(ii)由于所确定的色度格式而引起的颜色平面子采样这两者无关地来确定的;以及用于使用所确定的大小的系数组来从所述视频位流中解码所述变换块的代码。

[0015] 本发明的另一方面提供一种视频解码器,其被配置为:从视频位流中接收图像帧的颜色通道的变换块;确定所述图像帧的色度格式,所述色度格式使得相对于所述图像帧的亮度通道对所述图像帧的色度通道进行子采样;确定所述变换块的系数组大小,所述系数组大小是多达16个样本的变换块的最大区域,所述系数组大小是仅基于变换块大小、并且与(i)所述变换块的颜色平面和(ii)由于所确定的色度格式而引起的颜色平面子采样这两者无关地来确定的;以及使用所确定的大小的系数组来从所述视频位流中解码所述变换块。

[0016] 本发明的另一方面提供一种系统,包括:存储器;以及处理器,其中,所述处理器被配置为执行存储在所述存储器上的代码以实现从视频位流中解码图像帧的颜色通道的变换块的方法,所述方法包括:确定所述图像帧的色度格式,所述色度格式使得相对于所述图像帧的亮度通道对所述图像帧的色度通道进行子采样;确定所述变换块的系数组大小,所述系数组大小是多达16个样本的变换块的最大区域,所述系数组大小是仅基于变换块大小、并且与(i)所述变换块的颜色平面和(ii)由于所确定的色度格式而引起的颜色平面子采样这两者无关地来确定的;以及使用所确定的大小的系数组来从所述视频位流中解码所述变换块。

[0017] 还公开了其它方面。

附图说明

[0018] 现在将参考以下附图和附录描述本发明的至少一个示例实施例,其中:

[0019] 图1是示出视频编码和解码系统的示意性框图;

[0020] 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;

[0021] 图3是示出视频编码器的功能模块的示意框图;

[0022] 图4是示出视频解码器的功能模块的示意框图;

[0023] 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;

[0024] 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;

[0025] 图7A和7B示出编码树单元(CTU)向多个编码单元(CU)的示例分割;

[0026] 图8A、8B和8C示出亮度和色度通道中的编码树单元(CTU)向多个编码块(CB)的示例分割;

[0027] 图9示出变换块大小和相关联的扫描模式的集合;

[0028] 图10示出用于生成亮度编码树和色度编码树中的允许拆分列表的规则集;

[0029] 图11示出将图像帧的编码树编码到视频位流的方法;

[0030] 图12示出用于从视频位流解码图像帧的编码树的方法;

[0031] 图13示出将图像帧的编码树编码到视频位流的方法;

[0032] 图14示出用于从视频位流中解码图像帧的编码树的方法;

[0033] 图15示出帧内预测编码单元的变换块分区的集合;

[0034] 图16示出将图像帧的编码单元编码到视频位流中的方法;以及

[0035] 图17示出从视频位流中解码图像帧的编码单元的方法。

具体实施方式

[0036] 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。

[0037] 如上所述,使用来自紧挨着的先前块的样本导致反馈依赖,这可能限制通过视频编码器或解码器的块的吞吐量。期望用于减轻所导致的反馈依赖循环的严重性的方法,以确保能够维持处理块的高速率,如典型实时编码和解码应用所需要的那样。对于现代视频格式的高采样率(例如500-4000个样本/秒),反馈依赖循环尤其成问题,而ASIC(专用集成电路)时钟频率通常为数百MHz。

[0038] 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可以利用不同的规则来允许对亮度和色度编码树中的区域进行细分以降低所遇到的最坏情况块处理速率。例如,系统100可以操作使得块的大小总是被设置为16(十六)个样本的倍数,而不管块的长宽比如何。此外,在编码树包括指示存在小亮度编码块的拆分的情况下,可以在色度通道中禁止拆分,使得单个色度CB与多个亮度CB并置。独立于各个并置亮度CB的预测模式(包括其中一个或多个亮度CB使用帧间预测的情况),色度CB可以使用单一预测模式(诸如一种帧内预测模式等)。残差系数编码也可以利用16的倍数的块大小,包括块具有两个样本的宽度或高度的情况。

[0039] 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。

[0040] 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。

[0041] 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。

[0042] 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。解码后的帧数据135具有与帧数据113相同的色度格式。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。

[0043] 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。

[0044] 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。

[0045] 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。

[0046] 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。

[0047] 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。

[0048] 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。

[0049] 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。

[0050] 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。

[0051] 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。

[0052] 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。

[0053] 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。

[0054] 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存储器冲突的情况下执行的充足存储器。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。

[0055] 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。

[0056] 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。

[0057] 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。

[0058] 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。

[0059] 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:

[0060] 提取操作,用于从存储器位置228、229、230提取或读取指令231;

[0061] 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

[0062] 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。

[0063] 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。

[0064] 要说明的图10和图11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。

[0065] 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。

[0066] 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。帧数据113可以处于4:2:0色度格式或4:2:2色度格式。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分割成与亮度编码树或色度编码树相对应的一个或多个CB。CB具有各种大小并且可以包括正方形和非正方形长宽比这两者的。参考图10进一步描述块分区器310的操作。然而,在VVC标准中,CB、CU、PU和TU总是具有2的幂的边长。因而,从块分区器310输出当前CB(表示为312),从而根据对CTU的一个或多个块的迭代、根据CTU的亮度编码树和色度编码树而前进。以下参考图5和6来进一步说明用于将CTU分区为CB的选项。

[0067] 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。

[0068] 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”CB。第一阶段涉及测试各种候选CB,以选择提供高压缩效率和低失真的CB。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CB。选择“最佳”候选CB(具有最低评价速率/失真的CB)以用于随后编码在位流115中。候选CB的评价中包括如下选项:将CB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CB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CB和编码树本身这两者。

[0069] 视频编码器114针对各CB(例如,CB 312)产生由箭头320指示的预测块(PB)。PB 320是关联的CB 312的内容的预测。减法器模块322产生PB 320和CB 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PB 320和CB 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指示的变换块(TB)。通常从多个可能候选CB之一(例如,基于所评价的成本或失真)选择PB 320和关联的TB 336。

[0070] 候选编码块(CB)是针对关联的PB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CB。各候选CB得到一个或多个相应的TB,如在下文中参考图8所述。TB 336是差324的量化和变换表示。当与视频解码器114中的预测PB组合时,TB 336以位流中的附加信令为代价来减少解码CB和原始CB 312之间的差。

[0071] 因而,各候选编码块(CB)(即预测块(PB)与变换块(TB)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。CB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PB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。

[0072] 确定率失真方面的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佳模式选择,从而允许最终选择帧内预测模式以及其它模式决策。

[0073] 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。

[0074] 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CB的最佳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。最低成本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CB的最佳层级分解的处理。

[0075] 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中进行对所选择的亮度编码树和所选择的色度编码树(以及因而对各个所选择的CB)的迭代。如这里进一步描述的,在迭代中,将CB编码在位流115中。

[0076] 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。

[0077] 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。

[0078] 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存储器并降低了复杂度,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。

[0079] 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。

[0080] 多路复用器模块384根据选自各候选CB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PB 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。

[0081] 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PB,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的操作而言来描述帧内预测。

[0082] 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。此外,对于帧间预测,单个编码树通常用于亮度通道和色度通道这两者。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。

[0083] 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。

[0084] 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。

[0085] 在确定并选择了PB 320、并在减法器322处从原始样本块中减去了PB 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。正向主变换模块326对差324应用正向变换,从而将差324从空间阈转换到频率域并且产生由箭头328表示的主变换系数。主变换系数328被传递到正向二次变换模块330以通过进行不可分离二次变换(NSST)操作来产生由箭头332表示的变换系数。正向主变换通常是可分离的,从而通常使用DCT-2(但也可以使用DST-7和DCT-8)例如水平方向上针对不超过16个样本的块宽度和垂直方向上针对不超过16个样本的块高度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。正向二次变换通常是不可分离变换,其仅适用于帧内预测CU的残差,但也可能被绕过。正向二次变换对16个样本(排列为主变换系数328的左上4×4子块)或64个样本(排列为左上8×8系数,排列为主变换系数328的四个4×4子块)进行操作。此外,正向二次变换的矩阵系数是根据CU的帧内预测模式从多组中选择的,使得有两组系数可供使用。使用一组矩阵系数或绕过正向二次变换是以使用截断的一元二值化编码以表达值零(未应用二次变换)、一(所选择的第一组矩阵系数)或二(所选择的第二组矩阵系数)的“nsst_index”句法元素用信号通知的。

[0086] 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。

[0087] 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通过逆二次变换模块344以产生由箭头346表示的中间逆变换系数。中间逆变换系数346被传递到逆主变换模块348以产生TU的由箭头350表示的残差样本。逆

二次变换模块344进行的逆变换类型与由正向二次变换模块330进行的正向变换类型相对应。逆主变换模块348进行的逆变换类型与由主变换模块326进行的主变换类型相对应。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)。

[0088] 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。

[0089] 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。

[0090] 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的访问在存储器带宽方面是昂贵的。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。

[0091] 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CB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,PB 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。

[0092] 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)。

[0093] 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。

[0094] 位流133被输入到熵解码器模块420。熵解码器模块420通过解码“bin”序列从位流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使用算术解码引擎来将各个句法元素解码为一个或多个bin的序列。各个bin可以使用一个或多个“上下文”,其中上下文描述用于为bin编码“一”和“零”值的概率水平。在多个上下文可用于给定bin的情况下,进行“上下文建模”或“上下文选择”步骤以选择可用上下文之一来解码bin。对bin进行解码的处理形成了一个顺序反馈循环。反馈循环中的操作数量优选地被最小化以使得熵解码器420能够实现以bin/秒为单位的高吞吐量。上下文建模取决于在选择上下文时视频解码器134已知的位流的其它特性,即当前bin之前的特性。例如,可以基于编码树中当前CU的四叉树深度来选择上下文。依赖性优选地基于在解码bin之前熟知的特性,或者在不需要长的顺序处理的情况下确定。编码树的四叉树深度是容易知道的上下文建模的依赖性的示例。帧内预测模式是确定起来相对困难或计算密集的上下文建模的依赖性的示例。帧内预测模式被编码为“最大概率模式”(MPM)列表的索引或“剩余模式”列表的索引,其中根据所解码的“intra_luma_mpm_flag”在MPM和剩余模式之间进行选择。当使用MPM时,解码“intra_luma_mpm_idx”句法元素以选择要使用最大概率模式中的哪个。通常存在六个MPM。当使用剩余模式时,解码“intra_luma_remainder”句法元素以选择要使用剩余(非MPM)模式中的哪个。确定最大概率模式和剩余模式这两者需要大量的操作并且包括对相邻块的帧内预测模式的依赖。例如,相邻块可以是当前块上方和左侧的块。期望地,可以确定各个CU的bin的上下文,从而能够由算术编码引擎进行解析,而无需知道用信号通知的帧内预测模式。因此,存在于算术编码引擎中用于顺序bin解码的反馈循环避免了对帧内预测模式的依赖。利用由于MPM列表构建对相邻块的帧内预测模式的依赖而得到的单独反馈循环,帧内预测模式确定可以推迟到后续处理阶段。因此,熵解码器模块420的算术解码引擎能够解析intra_luma_mpm_flag、intra_luma_mpm_idx、intra_luma_remainder,而无需知道任何较早(例如,相邻)块的帧内预测模式。熵解码器模块420应用算术编码算法,例如“上下文自适应二进制算术编码”(CABAC),以从位流133解码句法元素。所解码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CB的分区。参数用于通常与来自先前解码的CB的样本数据组合生成PB。

[0095] 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中间变换系数(由箭头432表示)。根据所解码的“nsst_index”句法元素,重建中间变换系数432被传递到逆二次变换模块436,在该二次变换模块处应用二次变换或无操作(旁路)。在处理器205的执行下,熵解码器420从位流133解码“nsst_index”。如参考图3所述,“nsst_index”从位流133解码为具有值零至二的截断的一元句法元素。逆二次变换模块436产生重建变换系数440。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

432。

[0096] 重建变换系数440被传递到逆主变换模块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PB,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。

[0097] 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CB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。

[0098] 当在位流133中指示CB的预测模式是帧内预测时,帧内预测样本480经由复用器模块484形成解码的PB 452。帧内预测产生样本的预测块(PB),即,使用同一颜色分量中的“相邻样本”导出的一个颜色分量中的块。相邻样本是与当前块相邻的样本,并且由于在块解码顺序中处于前面因此已经被重建。在亮度和色度块并置的情况下,亮度和色度块可以使用不同的帧内预测模式。然而,两个色度通道各自共享相同的帧内预测模式。帧内预测分为三种类型。“DC帧内预测”涉及利用表示相邻样本的平均值的单个值填充PB。“平面帧内预测”涉及利用与平面相对应的样本填充PB,其中,DC偏移以及垂直和水平梯度源自相邻样本。“角度帧内预测”涉及利用以特定方向(或“角度”)在PB上滤波和传播的相邻样本填充PB。在VVC中,支持65个角度,其中矩形块能够利用正方形块不可用的附加角度来产生总共87个角度。第四种类型的帧内预测可用于色度PB,由此根据“交叉分量线性模型”(CCLM)模式从并置的亮度重建样本生成PB。三种不同的CCLM模式是可用的,各种模式使用从相邻亮度和色度样本导出的不同模型。然后使用导出模型来从并置的亮度样本生成色度PB的样本块。

[0099] 当在位流133中指示CB的预测模式是帧内预测时,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PB 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。通常,运动矢量被应用于亮度和色度通道这两者,但用于子样本插值亮度和色度通道的滤波处理是不同的。当编码树中的拆分导致相对小的亮度块的集合并且相应的色度区域没有被分割成相应的小色度块时,分别如参考图13和14所描述的那样对块进行编码和解码。特别地,如果使用帧间预测来预测小亮度块中的任一个,则仅针对亮度CB而不针对相应色度CB的任何部分进行帧间预测操作。环内滤波模块368从重建样本456产生滤波块数据492。

[0100] 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或CB。

[0101] 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”。

[0102] 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约束为禁止块区域小于例如16个样本的预定最小大小之外,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是最大可用编码单元大小。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7A和图7B进一步描述大于处理区域大小的CU的示例。由于使用经子采样色度格式(诸如4:2:0等),视频编码器114和视频解码器134的布置可以比亮度通道中更早地终止色度通道中的区域的拆分。

[0103] 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。在编码树的各个叶节点处,针对各个颜色通道存在一个编码块。针对亮度和色度这两者在相同的深度处终止的拆分得到三个并置CB。针对亮度比针对色度在更深的深度处终止的拆分得到与色度通道的CB并置的多个亮度CB。

[0104] 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)。

[0105] 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”。树的根包括零个或多个四叉树拆分(树的“QT”部分)。一旦QT部分终止,可以发生零个或多个二元或三元拆分(“多树”或树的“MT”部分),最终在树的叶节点处的CB或CU中结束。在树描述所有颜色通道的情况下,树的叶节点为CU。在树描述亮度通道或色度通道的情况下,树的叶节点为CB。

[0106] 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(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,在所描述的布置中,约束可以单独应用于色度通道的块。拆分选项对色度通道的约束的应用可能针对亮度vs色度(例如,在帧数据采用4:2:0色度格式或4:2:2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，因此所有CU的边尺寸也为2的幂。

[0107] 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。

[0108] 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。

[0109] 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。

[0110] 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。

[0111] 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。

[0112] 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。

[0113] 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。

[0114] 参考图9进一步描述在编码树的各个节点处的允许和不允许的拆分的集合。

[0115] 图7A和7B提供CTU 710向多个CU或CB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。

[0116] 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。

[0117] 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。

[0118] 图7A和7B的示例描述适用于亮度通道和色度通道这两者的编码树。然而,图7A和7B的示例还说明关于遍历仅适用于亮度通道的编码树或仅适用于色度通道的编码树的行为。对于具有许多嵌套拆分的编码树,较深层级处的可用拆分选项受到对用于相应小区域的可用块大小的限制的约束。施加对小区域的可用块大小的限制以防止块处理速率的最差情况如此高以致对实现施加不合理的负担。特别地,块大小在色度中将为16(十六)个样本的倍数的约束使得实现能够以16(十六)个样本的粒度处理样本。将块大小约束为十六个样本的倍数特别地与“帧内重建”反馈循环相关,即,图4的视频解码器134中的涉及模块450、460、468、476和484的路径以及视频编码器114中的等效路径。特别地,将块大小约束为16(十六)个样本的倍数帮助维持帧内预测模式中的吞吐量。例如,“同时数据多指令”(SIMD)微处理器架构通常对可以包含16个样本的宽字进行操作。此外,硬件架构可以使用宽总线(诸如具有16个样本的宽度的总线)来沿着帧内重建反馈循环传送样本。如果使用更小的块大小(例如,四个样本),则总线将未被充分使用,例如,仅总线宽度的四分之一包含样本数据。虽然未充分使用的总线可以处理较小的块(即,少于十六个样本),但是在最坏情况的场景中,诸如许多或所有块具有相对小的大小,未充分使用可能导致阻止编码器(114)或解码器(134)的实时操作。对于帧间预测,各个块取决于从帧缓冲器(诸如缓冲器372或496等)获得的参考样本。由于在处理先前帧时帧缓冲器填充有参考样本,因此不存在影响用于产生经帧间预测块的逐块操作的反馈依赖循环。除了与帧内帧重建相关的反馈依赖循环之外,还存在与帧内预测模式458的确定相关的附加且并发的反馈循环。通过从最大概率模式列表选择模式或从剩余模式列表选择模式来确定帧内预测模式458。最大概率模式列表和剩余模式列表的确定需要相邻块的帧内预测模式。当使用相对小的块大小时,最大概率模式列表和剩余模式列表需要较频繁地确定,即,在由样本中的块大小和通道的采样速率支配的频率下确定。

[0119] 图8A、8B和8C提供根据编码树820(图8B)的CTU 800(8A)的示例分割,其中在亮度拆分之前终止色度拆分且使用4:2:0色度格式。在色度拆分终止的情况下,使用CB对,各色度通道使用一个CB。为了说明方便,CTU 800的大小为64×64个亮度样本。CTU 800等效于128×128的CTU大小和包括一个附加的四叉树拆分的编码树。将四叉树拆分应用于8×8亮度区域814。8×8亮度区域814被拆分成四个4×4亮度CB,然而在色度通道中不发生拆分。代替地,使用预定最小大小(在所述示例中为16)对的色度CB,一对对应于各色度通道。色度CB对通常具有与可期望地同时处理的样本的数量的最小粒度相对应的最小大小。例如,视频

编码器114和视频解码器134的许多实现将对16个样本的集合进行操作,例如这是由于在硬件实现中使用相应宽的内部总线。此外,由拆分得到的各个亮度CB至少部分地与色度CB对重叠,且整体亮度CB完全与色度CB对重叠。在区域814的示例中,生成4×4色度CB对。图8C示出所得亮度CB和色度CB如何相关的示例。

[0120] 返回参考8A,将垂直二元拆分被应用于16×4亮度区域810。16×4亮度区域810被拆分成两个8×4亮度CB,然而,在色度通道中未发生拆分,从而得到8×2色度CB对。将垂直三元拆分应用于16×4亮度区域812。16×4亮度区域812被拆分成4×4、4×8和4×4亮度CB,然而,在色度通道中未发生拆分,从而得到8×2色度CB对。将水平二元拆分应用于8×16亮度区域816。8×16亮度区域816被拆分成8×4、8×8和8×4亮度CB,然而,在色度通道中未发生拆分,从而得到4×8色度CB对。因此,色度CB在区域中为至少16个样本。

[0121] 图8C示出CTU 800的一部分,该部分具有以“分解”(或分离)方式示出的三个颜色平面以例示不同平面中的不同块结构。示出了亮度样本平面850、第一色度样本平面852和第二色度样本平面854。当使用“YCbCr”颜色空间时,亮度样本平面850包含图像帧的Y样本,第一色度样本平面852包含图像帧的Cb样本,并且第二色度样本平面854包含图像帧的Cr样本。使用4:2:0色度格式得到第一色度样本平面852和第二色度样本平面854相对于亮度样本平面850在水平方向和垂直方向上具有样本密度的一半。因此,样本中的色度块的CB尺寸通常为相应亮度CB的尺寸的一半。即,对于4:2:0色度格式,色度CB宽度和高度各自为并置的亮度CB的一半。对于4:2:2色度格式,色度CB高度为并置的亮度CB的一半,而宽度与并置的亮度CB的宽度相同。为了清楚起见,仅示出8×16亮度区域816的编码树中的父拆分,并且仅在亮度样本平面850中示出了拆分。当色度拆分终止时,多个亮度CB与色度CB对并置。例如,CTU 800的编码树包括应用于8×16亮度区域816的水平三元拆分。水平三元拆分得到存在于亮度样本平面850中的8×4亮度CB 860、8×8亮度CB 862及8×4亮度CB 864。由于8×16亮度区域816与色度样本平面(852和854)中的4×8色度样本的区域相对应,因此编码树的三元拆分不应用于色度样本平面(852和854)。因此,4×8色度样本的区域形成色度的叶节点,从而得到色度CB对,即,用于第一色度样本平面852的色度CB 866和用于第二色度样本平面854的色度CB 868。在仅在亮度平面中应用水平三元拆分的示例中,实现32个样本的最小色度CB大小。其它示例亮度区域(810、812和814)得到16的最小色度CB大小,其对应于最小亮度块大小和样本处理的期望粒度。

[0122] 图9示出从使用4:2:0色度格式得到的色度通道的变换块大小和关联的扫描模式的集合900。还可以针对4:2:2色度格式使用集合900。所描述的布置适于用于具有如下色度格式的图像帧:其中相对于图像帧的亮度通道对图像帧的色度通道进行子采样,特别是对于4:2:0和4:2:2格式。集合900不包括所有可能的色度变换块大小。图9中仅示出具有小于或等于十六的宽度或者小于或等于八的高度的色度变换块。可发生具有较大宽度和高度的色度块,但为了便于参考,图9中未示出。

[0123] 一组禁止变换大小910包括变换块大小2×2、2×4和4×2,其全部具有小于十六个样本的区域。换句话说,在图9的示例中,16(十六)个色度样本的最小变换大小源自所描述的布置的操作,特别是对于帧内预测CB。通过如参考图10所描述那样确定拆分选项来避免禁止变换大小910的实例。以双层方式扫描变换中的残差系数,其中变换被分割成“子块”(或“系数组”)。扫描沿着从最后有效(非零)系数返回朝向DC(左上)系数的扫描路径进行。

扫描路径被定义为各子块(“下层”)内的进展和从一个子块到下一子块(“上层”)的进展。在集合900中,8×2TB 920使用8×2子块,即,包含十六个残差系数的子块。2×8TB 922使用2×8子块,即,也包含十六个残差系数。

[0124] 宽度或高度为2并且另一尺寸为8的倍数的TB使用多个2×8或8×2子块。因此,在一些情况下具有两个样本的宽度的色度块使用块到子块的分割来编码,子块各自大小为2×8个样本,并且在一些实例中,具有两个样本的高度的色度块使用块到子块的分割来编码,子块各自大小为8×2个样本。例如,16×2TB 916具有两个8×2子块,如针对TB 920所示,扫描各子块。从一个子块到下一子块的扫描的进展如子块进展917中所示。

[0125] 2×32TB(图9中未示出)使用布置为一乘四阵列的四个2×8子块。如针对2×8TB 922所示来扫描各子块中的残差系数,其中子块从一乘四阵列的最低子块向上前进到最高子块。

[0126] 较大的TB遵循类似的扫描进展。对于宽度和高度各自均大于或等于四的所有TB,使用4×4子块扫描。例如,4×8TB 923使用4×4子块扫描924,具有从下方子块到上方子块的进展。可以用类似方式扫描4×4TB 925。8×8TB 929使用针对四个4×4子块的进展930。在所有情况下,子块内的扫描和从子块到子块的进展遵循反向对角线扫描,即,扫描从TB的“最后”有效残差系数返回朝向左上方残差系数而前进。图9还示出跨越例如8×4TB 932、16×4TB 934和16×8TB 936的扫描顺序。此外,取决于最后有效系数沿着扫描路径的位置,仅需要扫描子块的包含从子块的最后有效系数位置起的最后有效残差系数直到左上方残差系数的部分。不需要扫描在正向方向上进一步沿扫描路径的子块(即,更靠近块的右下方)。集合900并且特定是禁止变换大小910对用于将色度中的编码树的区域(或节点)拆分成子区域(或子节点)的能力施加限制,如参考图10所描述。

[0127] 在使用2×2、2×4和4×2TB(一组TB 910)的VVC系统中,针对宽度和/或高度为两个样本的TB,可以采用2×2子块。如上所述,TB 910的使用增加了帧内重建反馈依赖循环中的吞吐量约束。此外,仅具有四个系数的子块的使用增加了在较高吞吐量下解析残差系数的难度。特别地,对于各子块,“有效图”指示所包含的各残差系数的有效性。对1值有效性标志进行编码,这使得将残差系数的量值建立为至少一,并且对零值标志进行编码,这使得将残差系数的量值建立为零。仅针对“有效”残差系数对残差系数量值(从一向前)和符号进行编码。对于DC系数,无有效位被编码且量值(从零起)总是被编码。高吞吐量编码器和解码器可能需要每个时钟周期编码或解码多个有效图bin以维持实时操作。当bin间依赖性较多时(例如,当使用较小子块大小时),增加了每个周期的多bin编码和解码的难度。在系统100中,子块大小为16(尽管包含最后有效系数的子块除外),而不管块大小如何。

[0128] 图10示出用于生成色度编码树中的允许拆分的列表的规则集1000。其它帧可以允许帧间预测块与帧内预测块的混合。尽管已参考图6描述了编码树的可用拆分的完整集,但是针对给定区域大小,对可用变换大小的限制对特定拆分选项施加了约束。如下所述,亮度通道和色度通道各自的拆分选项根据对应编码树单元的区域的尺寸来确定。

[0129] 色度区域的规则1020示出不同区域的允许拆分。即使色度通道在考虑中,由于可能使用不同的色度格式,因此规则1020的允许拆分以亮度样本的单位来表示。

[0130] 在遍历编码树的节点时,通过利用编码树的区域大小检查一组拆分选项的可用性来获得用于色度的允许拆分的列表。得到可使用CB编码的区域的拆分选项被添加到允许拆

分的列表。对于使用CB编码的区域,区域大小必须使得能够利用来自集合900的特定大小的整数数量的变换进行编码。特定大小被选择为不超过区域大小(考虑宽度和高度这两者)的最大大小。如此,对于较小区域,使用单个变换。在区域大小超过最大可用变换的大小的情况下,对最大可用变换进行拼接以占据整个区域。

[0131] 当考虑编码树中具有给定面积(以亮度样本表示)的节点时,根据拆分类型和色度区域面积来确定进行给定类型拆分的能力。如图10所示,对照区域大小来测试各拆分选项以判断拆分选项是否将导致禁止大小的子区域。得到允许大小的子区域的拆分选项被视为是允许色度拆分1070。

[0132] 例如,如色度区域的规则1021a所示,在QT模式(对应于图6的决策610)的情况下,如果区域在4:2:0格式中具有大小8×8或在4:2:2格式中具有大小8×8,则不允许四叉树拆分,因为拆分将导致分别针对色度通道的2×2或2×4的变换大小。可允许的区域大小由箭头1021指示。类似地,色度规则集1020的其它可允许拆分由箭头1022、1023、1024、1025和1026指示并且如下文关于图13和14所论述。箭头1021、1022、1023、1024、1025和1026各自引用允许的色度拆分列表1070。

[0133] 就亮度样本栅格方面描述色度通道的区域大小。例如,当使用4:2:0色度格式时,8×4区域与用于色度通道的4×2变换相对应。当使用4:2:2色度格式时,8×4区域与色度中的4×4变换相对应。当使用4:4:4色度格式时,不相对于亮度对色度进行子采样,因此色度中的变换大小与区域大小相对应。

[0134] 可允许的拆分选项将在下面关于图13和14进一步描述。

[0135] 图11示出用于将图像帧的编码树编码在视频位流中的方法1100。方法1100可以由诸如配置的FPGA、ASIC或ASSP等的设备来实现。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。因此,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。方法1100开始于确定色度格式步骤1105。

[0136] 在确定色度格式步骤1105处,处理器205将帧数据113的色度格式确定为4:2:0色度格式或4:2:2色度格式之一。色度格式是帧数据的属性,并且在方法1100的操作期间不改变。方法1100在处理器205的控制下从步骤1105继续到将帧分割成CTU的步骤1110。

[0137] 在将帧分割成CTU的步骤1110处,在处理器205的执行下,块分区器310将帧数据113的当前帧分割成CTU阵列。开始进行随着从分割得到的CTU的编码。处理器中的控制从步骤1110前进到确定编码树的步骤1120。

[0138] 在确定编码树的步骤1120处,在处理器205的执行下,视频编码器114组合测试各种预测模式和拆分选项以到达CTU的编码树。还导出了CTU的编码树的各个CU的预测模式和残差系数。通常,进行拉格朗日优化以选择CTU的最佳编码树和CU。当评价使用帧间预测时,从一组候选运动矢量中选择运动矢量。根据搜索模式生成候选运动矢量。在对评价中的候选运动矢量的所提取的参考块进行失真测试时,考虑了编码树中的禁止色度拆分的应用。当在色度中禁止拆分并且在亮度中允许拆分时,所得到的亮度CB可以使用帧间预测。仅对亮度通道应用运动补偿,因此失真计算考虑了亮度失真而不考虑色度失真。由于在色度拆分被禁止时在色度通道中不进行运动补偿,因此不考虑色度失真。对于色度,考虑由所考虑的帧内预测模式和编码色度TB(如果有的话)引起的失真。当考虑亮度和色度这两者时,帧间预测搜索可以首先基于亮度失真选择运动矢量,然后也通过考虑色度失真来“细化”运动

矢量。细化通常考虑运动矢量值的小变化,诸如子像素位移等。当色度拆分被禁止并且进行针对小亮度块的帧间预测的评价时,不需要色度细化。处理器205中的控制从步骤1120前进到对编码树进行编码的步骤1130。

[0139] 在对编码树进行编码的步骤1130处,在处理器205的执行下,视频编码器114进行关于图1300描述的方法1300以将当前CTU的编码树编码在位流115中。执行步骤1130以将当前CTU编码在位流中。处理器205中的控制从步骤1130前进到最后CTU测试步骤1140。

[0140] 在最后CTU测试步骤1140处,处理器205测试当前CTU是否是条带或帧中的最后CTU。如果不是(步骤1140处为“否”),则视频编码器114前进到帧中的下一个CTU,并且处理器205中的控制从步骤1140前进回到步骤1120,以继续处理帧中的剩余CTU。如果CTU是帧或条带中的最后一个,则步骤1140返回“是”,并且方法1100终止。作为方法1100的结果,将整个图像帧作为CTU序列编码在位流中。

[0141] 图12示出用于从视频位流解码图像帧的编码树的方法1200。方法1200可以由诸如配置的FPGA、ASIC或ASSP等的设备来实现。另外,方法1200可以在处理器205的执行下由视频解码器134进行。因此,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开始于确定色度格式步骤1205。

[0142] 在确定色度格式步骤1205处,处理器205将帧数据113的色度格式确定为4:2:0色度格式或4:2:2色度格式之一。色度格式是帧数据的属性,并且在方法1200的操作期间不改变。视频解码器134可以借助位流133的简档来确定色度格式。简档定义了可以由特定位流133使用的一组编码工具,并且可以将色度格式限制为特定值,诸如4:2:0等。通过从位流133解码例如“profile_idc”句法元素,或者通过从位流133解码一个或多个约束标志来确定简档,其中各个约束标志约束特定工具在位流133中的使用。在色度格式不是由简档完全指定的情况下,可以解码诸如“chroma_format_idc”等的其它句法以确定色度格式。方法1200在处理器205的执行下从步骤1205继续到将帧分割成CTU的步骤1210。

[0143] 在将帧分割成CTU的步骤1210处,在处理器205的执行下,视频解码器134确定要解码为CTU阵列的帧数据133的当前帧的分割。开始进行随着从确定的分割所得到的CTU的解码。处理器中的控制从步骤1210前进到对编码树进行解码的步骤1220。

[0144] 在对编码树进行解码的步骤1220处,在处理器205的执行下,视频解码器134针对当前CTU进行方法1400以从位流133解码当前CTU的编码树。当前CTU是从执行步骤1210得到的CTU中选择的一个CTU。处理器205中的控制从步骤1220前进到最后CTU测试步骤1240。

[0145] 在最后CTU测试步骤1240处,处理器205测试当前CTU是否是条带或帧中的最后一个。如果不是(步骤1240处为“否”),则视频解码器134前进到帧中的下一个CTU,并且处理器205中的控制从步骤1240前进回到步骤1220,以从位流继续解码CTU。如果CTU是帧或条带中的最后一个,则步骤1240返回“是”,并且方法1200终止。

[0146] 图13示出将图像帧的编码树编码在视频位流中的方法1300。方法1300可以由诸如配置的FPGA、ASIC或ASSP等的设备来实现。另外,方法1300可以在处理器205的执行下由视频编码器114进行。因此,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使得将块编码在位流115中,以使得各个块是最小的最小区域。所描述的布置使用预定最小大小的样本。在所描述的示例中使用的最小大小是16个样本,这在一些硬件和软件实现方面是优选的。然而,仍然可以使用不同的最小大小。例如,处理粒度为32或

64并且分别为32或64个样本的相应最小块区域是可能的。在硬件和软件实现这两者中,编码具有最小区域的块对于实现可行性是有利的。对于软件实现,16个样本的最小区域与典型的单指令多数据(SIMD)指令集(诸如AVX-2和SSE4等)对齐。最初在当前CTU的编码树的根节点处调用的方法1300在编码拆分模式步骤1310处开始。

[0147] 在编码拆分模式步骤1310处,在处理器205的执行下,熵编码器338将编码树的当前节点处的拆分模式编码在位流115中。拆分模式是参考图5描述的拆分模式之一,并且编码拆分模式的步骤仅允许对可能的拆分进行编码。例如,四叉树拆分512仅可能在编码树的根节点处或在编码树中的其它四叉树拆分之下。如关于组910所示,禁止得到宽度或高度小于四个样本的亮度CB的拆分。例如基于规则集1010,关于二元和/或三元拆分的最大深度的其它约束也可能是有效的。处理器205中的控制从步骤1310前进到无拆分测试步骤1320。

[0148] 在无拆分测试步骤1320处,处理器205测试当前拆分是否为“无拆分”(即510)。如果当前拆分是无拆分510(步骤1320处为“是”),则处理器205中的控制从步骤1320前进到编码CU步骤1330。否则,如果当前拆分不是510(步骤1320处为“否”),则处理器205中的控制前进到色度拆分禁止测试步骤1340。

[0149] 在编码CU步骤1330处,在处理器205的执行下,熵编码器338将CU的预测模式和CU的残差编码在位流115中。在步骤1330到达编码树的各个叶节点时,方法1300在完成步骤1330时终止,返回到编码树遍历中的父调用。一旦编码树的所有节点已经被遍历,整个CTU被编码在位流115中,并且控制返回到方法1100,前进到图像帧中的下一个CTU。

[0150] 在色度拆分禁止测试步骤1340处,处理器205根据图10的色度区域1020拆分规则集,确定是否允许按照步骤1310将编码树中的当前节点的拆分应用于色度通道。如果编码树中的当前节点覆盖128个亮度样本(32×4或4×32或16×8或8×16)的亮度区域,则相应色度区域(分别为16×2、2×16、8×4、4×8个色度样本)中的三元拆分被禁止,如规则集1020中所示。如果允许三元拆分,则得到的块大小将包括禁止的块大小(例如2×4或4×2)。当编码树中的当前节点覆盖64个亮度样本的亮度区域时,禁止二元、三元和四叉树拆分,如规则集1020中所示。对于64个亮度样本的亮度区域实现二元、三元和四叉树拆分将得到禁止的色度块大小(2×2、2×4、4×2)。如果不禁止拆分(即,拆分是列表1070的允许色度拆分),则步骤1340返回“否”,并且处理器205中的控制从步骤1340前进到进行亮度和色度拆分步骤1350。否则,如果禁止拆分(1340处为“是”),则处理器205中的控制前进到进行亮度拆分步骤13100。

[0151] 在进行亮度和色度拆分步骤1350处,处理器205应用该拆分以将与编码树的当前节点相关联的当前区域分割成与编码树的子节点相关联的子区域。根据图5和6的说明来应用拆分。处理器205中的控制从步骤1350前进到选择区域步骤1360。

[0152] 在选择区域步骤1360处,处理器选择由步骤1350得到的子区域之一。根据区域的Z顺序扫描来选择子区域。在步骤1360的后续迭代中,选择进展通过子区域。处理器205中的控制从步骤1360前进到对编码树进行编码的步骤1370。

[0153] 在对编码树进行编码的步骤1370处,处理器205针对步骤1360所得到的所选区域递归地调用方法1300。步骤1370进一步操作以针对各个区域将亮度和色度块以及相关联的预测模式和残差系数编码到位流。处理器205中的控制从步骤1370进行到最后区域测试步骤1380。

[0154] 在最后区域测试步骤1380处,处理器205测试如在步骤1360中选择的所选区域是否是如步骤1350中实现的拆分模式分割所得到的区域中的最后一个。如果该区域不是最后区域(步骤1380处为“否”),则处理器205中的控制从步骤1380前进到步骤1360,以继续进展通过拆分的区域。否则,步骤1380返回“是”,方法1300终止,并且处理器205中的控制进行到方法1300的父调用。

[0155] 在进行亮度拆分步骤13100处,如在步骤1310处编码的拆分模式仅由处理器205在亮度通道中进行。结果,根据拆分模式将编码树的当前节点分割成多个亮度CB。仅生成色度CB对,即每个色度通道一个色度CB。各个所得到的亮度CB与色度CB对以及共同地所得到的亮度CB部分地重叠(并置)。整体亮度CB精确地覆盖色度CB对的区域,具有色度CB对的区域。此外,各个亮度CB和色度CB的最小区域是最小大小,例如16个样本。

[0156] 步骤13100和1350各自操作以确定色度通道Cb和Cr的色度编码块的大小。在步骤1350处,基于在步骤1310处确定的拆分模式来确定色度通道的色度编码块大小。在步骤13100处,基于预定最小色度块大小来确定色度通道的色度编码块大小。如上所述,基于对于编码树单元禁止的色度拆分来实现步骤1350。如图10的规则集1020所示,基于在步骤1105中确定的色度格式来确定允许的拆分以及相应的色度编码块的大小。

[0157] 处理器205中的控制从步骤13100前进到选择亮度CB步骤13110。

[0158] 在选择亮度CB步骤13110处,处理器205选择由步骤13100得到的CB的下一个亮度CB。方法13100最初选择第一CB,即由亮度拆分得到的CB的左上亮度CB。在随后调用步骤13110时,根据对从步骤13100得到的亮度CB的Z顺序扫描来选择各个“下一个”亮度CB。处理器205中的控制从步骤13110前进到编码亮度CB步骤13120。

[0159] 在编码亮度CB步骤13120处,在处理器205的执行下,熵编码器338将所选择的亮度CB编码在位流115中。通常,针对所选择的亮度CB来编码预测模式和残差系数。针对亮度CB编码的预测模式可以使用帧间预测或帧内预测。例如,“cu_skip_flag”被编码以指示在没有任何残差的情况下使用帧间预测,否则“pred_mode_flag”和可选的“pred_mode_ibc_flag”被编码以指示使用帧内预测、帧间预测或块内复制,各自具有可选的残差系数。当可能存在残差时,“cu_cbf”标志用信号通知在CB的任何TB中存在至少一个有效(非零)残差系数。当指示CB使用帧间预测时,相关联的运动矢量仅适用于亮度CB。即,运动矢量也不应用于生成与任何部分并置的色度CB相关联的任何PB。当指示CB使用块内复制时,相关联的块矢量仅与亮度CB相关联,而不与任何部分并置的色度CB相关联。处理器205中的控制从步骤13120前进到最后亮度CB测试步骤13130。

[0160] 在最后亮度CB测试步骤13130处,处理器205根据在步骤13100进行的拆分的亮度CB的Z顺序迭代来测试在步骤13110处选择的亮度CB是否是最后亮度CB。如果所选择的亮度CB不是最后一个(步骤13130处为“否”),则处理器205中的控制从步骤13130前进到步骤13120。否则,步骤13130返回“是”,处理器205中的控制前进到确定色度帧内预测模式步骤13140。

[0161] 在确定色度帧内预测模式13140处,在处理器205的执行下,视频编码器114确定与步骤13100的亮度CB并置的色度CB对的帧内预测模式。步骤13140有效率地确定为使用帧内预测对色度块进行编码。确定色度CB占用的区域是否在亮度通道中被进一步拆分成多个亮度CB。通道的色度块的大小是通过步骤1350的操作确定的预定最小值(例如16个样本)。即

使在步骤13120处使用帧间预测对相应的亮度CB进行编码,也确定色度CB对的帧内预测模式。在一种配置中,对各个色度CB应用单一预测模式,诸如DC帧内预测等。单一预测模式的使用允许借助于对拆分色度的禁止来确定该模式(步骤1340处为“是”的结果),并且不涉及额外的搜索来确定要使用多个可能模式中的哪个模式。此外,对于这种情况,位流115不需要附加的信号通知,即,不需要对附加的“intra_chroma_pred_mode”句法元素进行编码。然而,当色度拆分已经被禁止时(步骤1340处为“是”),通过在位流115中包括“intra_chroma_pred_mode”句法元素,用信号通知几种可能的帧内预测模式中的一种帧内预测模式来实现更高的压缩性能。视频编码器114确定要使用哪个帧内预测模式。帧内预测模式通常根据与失真相比的编码成本的考虑来确定。然而,与针对这种色度CB使用单一帧内预测模式相比,通常获得更高的压缩性能。处理器205中的控制从步骤13140前进到编码色度CB步骤13150。

[0162] 在编码色度CB步骤13150处,在处理器205的执行下,熵编码器338在可使用多个帧内预测模式时使用“intra_chroma_pred_mode”句法元素将色度CB的帧内预测模式编码在位流115中。当一个帧内预测模式(例如DC帧内预测)是可能的时,“intra_chroma_pred_mode”不被编码在位流115中。用于色度帧内预测的可用帧内预测模式可以包括DC、平面和以下角度预测模式:水平、垂直、右上对角线。可用的帧内预测模式还可以包括“直接模式”(DM_CHROMA),由此从并置的亮度CB(通常是由步骤13100得到的亮度CB的最下和最右侧)获得色度帧内预测模式。当“交叉分量线性模型”帧内预测可用时,可以从来自亮度CB的样本来预测色度CB。与色度CB相关联的色度TB的残差系数也可以被编码在位流115中,如参考图14的步骤14150所描述的。一旦步骤13150由处理器205执行,方法1300终止,并且处理器205中的控制返回到方法1300的父调用。

[0163] 图14示出在方法1200的步骤1220中实现的从视频位流解码图像帧的编码树的方法1400。方法1400可以由诸如配置的FPGA、ASIC或ASSP等的设备来实现。另外,方法1400可以在处理器205的执行下由视频解码器134进行。因此,方法1400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1400得到如下:从位流133解码块,以使得各个块不小于最小区域,诸如16个样本等,这在硬件情况和软件情况这两者下对于实现可行性是有利的。对于软件情况,16个样本的最小区域与典型的单指令多数据(SIMD)指令集(诸如AVX-2和SSE4等)对齐。最初在当前CTU的编码树的根节点处调用的方法1400在解码拆分模式步骤1410处开始。

[0164] 在解码拆分模式步骤1410处,在处理器205的执行下,熵解码器420从位流133中解码编码树的当前节点处的拆分模式。拆分模式是参考图5描述的拆分之一并且编码拆分模式的方法仅允许对被允许的拆分进行编码,即,即使在色度通道中禁止拆分的情况下在亮度通道中也被是允许的。例如,四叉树拆分512仅可能在编码树的根节点处或在编码树中的其它四叉树拆分之下。禁止了得到宽度或高度小于四个样本的亮度CB的拆分。因此,最小亮度CB大小是16个样本。关于二元和/或三元拆分的最大深度的其它约束也可以是有效的。处理器205中的控制从步骤1410前进到未拆分测试步骤1420。

[0165] 在无拆分测试步骤1420处,处理器205测试当前拆分是否为“无拆分”(即,510)。如果当前拆分是无拆分510(1420处为“是”),则处理器205中的控制从步骤1420前进到解码CU步骤1430。否则,步骤1420返回“否”,并且处理器205中的控制前进到色度拆分禁止测试步骤1440。

[0166] 在解码CU步骤1430处,在处理器205的执行下,熵解码器420解码位流115的CU的预测模式和CU的残差系数。步骤1430进行操作以使用由熵解码器420从位流确定的残差系数和预测模式来解码编码单元。当步骤1430到达编码树的各个叶节点时,方法1400在步骤1430完成时终止,返回到编码树遍历中的父调用。一旦已经遍历了编码树的所有节点,则从位流133解码整个CTU,并且控制返回到方法1200,前进到图像帧中的下一个CTU。

[0167] 在色度拆分禁止测试步骤1440处,处理器205根据图10的色度区域1020拆分规则集,确定是否按照步骤1410允许将编码树中的当前节点的拆分应用于色度通道。步骤1440以与方法1300的步骤1340类似的方式确定是否禁止拆分测试。步骤1440的操作防止发生被禁止的块大小。当色度区域已经处于最小大小(例如16个色度样本)时,不允许任何类型的进一步拆分,因为所得到的区域将小于允许的最小值。当色度区域大小是32个样本并且相应的拆分是三元拆分(无论是水平还是垂直三元拆分)时,也不允许进一步拆分以避免区域8个色度样本的色度块。如果不禁止拆分(即,允许拆分),则步骤1450返回“否”,并且处理器205中的控制从步骤1440前进到进行亮度和色度拆分步骤1450。否则,如果禁止拆分(步骤1450处为“是”),则处理器205中的控制前进到确定色度帧内预测模式的步骤14100。

[0168] 在进行亮度和色度拆分步骤1450处,处理器205应用该拆分以将与编码树的当前节点相关联的当前区域拆分成与编码树的子节点相关联的子区域。如关于图5和6所述来应用拆分。

[0169] 步骤14100和1450各自进行操作以确定色度通道Cb和Cr的色度编码块的大小。在步骤1450处,基于在步骤1410处解码的拆分模式来确定色度通道的色度编码块大小。在步骤14100处,基于预定最小色度块大小来确定色度通道的色度编码块大小。如上所述,基于对编码树单元禁止的色度拆分来实现步骤1450,该色度拆分对应于最小色度CB大小16(并且在亮度区域128个样本的三元拆分的情况下为32)。如图10的规则集1020所示,基于在步骤1205处确定的色度格式,确定可允许的拆分以及相应的色度编码块的大小。

[0170] 处理器205中的控制从步骤1450前进到选择区域步骤1460。

[0171] 在选择区域步骤1460处,处理器205根据区域的Z顺序扫描来选择从步骤1450得到的子区域之一。步骤1460进行操作以在后续迭代中使选择进展通过子区域。处理器205中的控制从步骤1460前进到对编码树进行解码的步骤1470。

[0172] 在对编码树进行解码的步骤1470处,处理器205针对从步骤1460的操作所得到的所选择区域递归地调用方法1400。步骤1470进一步操作以使用从位流确定的残差系数和预测模式对编码树的各个区域进行解码。处理器205中的控制从步骤1470前进到最后区域测试步骤1480。

[0173] 在最后区域测试步骤1480处,处理器205测试在步骤1460的最后一次迭代中预选的所选区域是否是在步骤1450处实现的拆分模式分割所得到的区域中的最后一个。如果该区域不是最后区域(步骤1480处为“否”),则处理器205中的控制从步骤1480前进到步骤1460,以继续前进通过拆分的区域。否则,步骤1480返回“是”,方法1400终止,并且处理器205中的控制前进到方法1400的父调用。

[0174] 在进行亮度拆分步骤14100处,仅由处理器205在亮度通道中进行步骤1410处解码的拆分模式。结果,编码树的当前节点根据拆分模式被分割成多个亮度CB。步骤14100进行操作以仅生成色度CB对,即每个色度通道一个色度CB。各个所得到的亮度CB与色度CB对部

分地重叠(至少部分地并置),并且共同地亮度CB与色度CB对完全地重叠。此外,各个亮度CB和色度CB的最小区域是16个样本。处理器205中的控制从步骤14100前进到选择亮度CB步骤14110。

[0175] 在选择亮度CB步骤14110处,处理器205选择从步骤14100得到的CB中的下一个亮度CB。下一个亮度CB的选择从第一个CB开始,第一个CB是从亮度拆分所得到的CB中的左上亮度CB。在随后调用步骤14110时,根据对从步骤14100得到的亮度CB的Z顺序扫描来选择各个“下一个”亮度CB。处理器205中的控制从步骤14110前进到解码亮度CB步骤14120。

[0176] 在解码亮度CB步骤14120处,在处理器205的执行下,熵解码器420从位流115解码所选择的亮度CB。通常,针对所选择的亮度CB来解码预测模式和残差。例如,“cu_skip_flag”被解码以指示在没有任何残差的情况下使用帧间预测,否则“pred_mode_flag”和可选的“pred_mode_ibc_flag”被解码以指示使用帧内预测、帧间预测或块内复制,各自具有可选的残差系数。当可能存在残差时,“cu_cbf”标志用信号通知在CB的任何TB中存在至少一个有效(非零)残差系数。当指示CB使用帧间预测时,相关联的运动矢量仅适用于亮度CB,即,运动矢量也不被应用来生成与任何部分并置的色度CB相关联的任何PB。当指示CB使用块内复制时,相关联的块矢量仅与亮度CB相关联,而不与任何部分并置的色度CB相关联。处理器205中的控制从步骤14120前进到最后亮度CB测试步骤14130。

[0177] 在最后亮度CB测试步骤14130处,处理器205根据在步骤14100处进行的拆分的亮度CB的Z顺序迭代来测试在步骤14110处选择的亮度CB是否是最后亮度CB。如果所选择的亮度CB不是最后一个,则处理器205中的控制从步骤14130前进到步骤14110。否则,处理器205中的控制前进到确定色度帧内预测模式步骤14140。

[0178] 在确定色度帧内预测模式14140处,在处理器205的执行下,视频解码器134确定与步骤14100的亮度CB并置的色度CB对的帧内预测模式。如果如通过步骤1440的操作所确定的,色度块是在发生对亮度的编码树的拆分的同时停止拆分色度的编码树的结果,则步骤14140有效地确定为色度块已经使用帧内预测来编码,并且因此将使用帧内预测来解码。即使在步骤14120处使用帧间预测对相应的亮度CB进行解码,也确定色度CB对的帧内预测模式。在一种布置中,对各个色度CB应用单一预测模式,诸如DC帧内预测等。单一预测模式的使用允许借助于对拆分色度的禁止来确定该模式(步骤1440处为“是”的结果),并且不涉及额外的搜索来确定要使用多个可能模式中的哪个模式。此外,对于这种情况,位流134不需要附加的信号通知,即,不需要对附加的“intra_chroma_pred_mode”句法元素进行编码。然而,当色度拆分已经被禁止时(步骤1440处为“是”),通过在位流134中包括“intra_chroma_pred_mode”句法元素,用信号通知几种可能的帧内预测模式中的一种帧内预测模式来实现更高的压缩性能。视频解码器134需要使用熵解码器420确定要使用哪个帧内预测模式以从位流134解码“intra_chroma_pred_mode”句法元素。处理器205中的控制从步骤14140前进到解码色度CB步骤14150。

[0179] 在解码色度CB步骤14150处,在处理器205的执行下,熵解码器420通常根据所解码的“intra_chroma_pred_mode”句法元素从位流420确定色度CB的帧内预测模式。当多种帧内预测模式可用时,对“intra_chroma_pred_mode”进行解码。当仅一种帧内预测模式可用(例如DC帧内预测)时,在不从位流133解码附加句法元素的情况下推断该模式。用于色度帧内预测的可用帧内预测模式可以包括DC、平面、以下角度预测模式:水平、垂直、右上对角

线。可用的帧内预测模式还可以包括“直接模式”(DM_CHROMA),由此从并置的亮度CB(通常是从步骤14100得到的亮度CB的最下方和最右方)获得色度帧内预测模式。当“交叉分量线性模型”帧内预测可用时,可以从来自亮度CB的样本来预测色度CB。对于色度CB对,“cu_cbf”标志用信号通知色度CB对的任意一个中存在至少一个有效残差系数。如果在色度CB对的任意一个中存在至少一个有效残差系数,则“tu_cbf_cb”和“tu_cbf_cr”分别用信号通知Cb和Cr通道的色度CB中存在至少一个有效系数。对于具有至少一个有效残差系数的色度CB,对句法元素的“residual_coding”序列进行解码以确定相应色度CB的残差系数。残差编码句法根据后向对角线扫描将残差系数编码为使变换块从最后有效系数位置填充到左上(“DC”)系数位置的值序列。后向对角线扫描进行作为“子块”(通常为大小4×4,但大小2×2、2×4、2×8、8×2、4×2也是可能的)序列(或“系数组”)的变换块的扫描。在各个系数组内的扫描是在后向对角线方向上,并且从一个子块到下一个子块的扫描也是在后向对角线方向上。一旦步骤14150由处理器205执行,则方法1400终止,并且处理器205中的控制返回到方法1400的父调用。

[0180] 方法1300和方法1400的编码树方法(其中针对4:2:0色度格式的视频数据保持16个样本的最少块区域)便于在软件和硬件这两者中实现高吞吐量。此外,针对小CB大小的对亮度CB的帧间预测的限制通过避免对还提取用于产生运动补偿色度CB的样本的需求来减小运动补偿存储器带宽的这种最坏情况的存储器带宽。特别地,如果最小色度CB大小为2×2,并且需要附加样本来提供对色度CB的子样本插值的滤波支持,则与仅针对小块大小在亮度通道中进行帧间预测相比,将看到存储器带宽的显著增加。运动补偿的编码增益基本上出现在亮度通道中,因此同样被运动补偿的小块的省略实现了存储器带宽的减小,从而对编码性能的影响相对较少。此外,存储器带宽的减小有助于进行对4×4亮度CB的运动补偿并实现所得编码增益的可行性。

[0181] 在视频编码器114和视频解码器134的一个布置中,从编码树的色度拆分终止的点开始,编码树中可以发生多于一个亮度拆分。例如,在色度通道中不拆分8×16亮度区域,从而得到4×8色度CB对。在亮度通道中,首先用水平三元拆分来拆分8×16亮度区域,然后进一步拆分所得到的亮度CB中的一个。例如,所得到的8×4亮度CB被垂直二元拆分成两个4×4亮度CB。编码树中从编码树的色度拆分终止的点起具有多于一个亮度拆分的布置在色度拆分禁止区域内分别在视频编码器114和视频解码器134中重新调用方法1300和1400,该调用具有在随后调用中不需要其它色度CB这样的修改。在创建了色度CB对的方法1300和1400的调用中,整个色度区域被所创建的色度CB覆盖,因此方法1300和1400的递归调用不需要创建附加色度CB。

[0182] 图15示出帧内预测的编码单元的变换块分区的集合1500。亮度CB可以被分区成相同大小的一个亮度TB(“ISP_NO_SPLIT”)。大小为4×4的亮度CB具有16个样本的区域且不进一步被分区,从而得到大小也为4×4的一个亮度TB。具有32个样本的区域的亮度CB可以分区成两个分区。例如,8×4亮度CB 1510可以水平地(“ISP_HOR_SPLIT”)分区成两个8×2亮度TB 1520或垂直地(“ISP_VER_SPLIT”)分区成两个4×4亮度TB 1530。如果亮度CB 1510为4×8亮度CB,则块可以在1520处水平地分区成两个4×4亮度TB或在1530处垂直地分区成两个2×8亮度TB。

[0183] 区域64或更大样本的亮度CB被分区成一个分区或分区成四个分区。具有64个或更

大样本的区域的宽度W和高度H的亮度CB 1550可以被水平地分区成大小为W×(H/4)的四个亮度TB 1560,或者可以被垂直地分区成四个(W/4)×H亮度TB 1570。如集合1500中所示,将亮度CB分割成多个分区,这得到更多和更小的亮度TB。进行帧内预测以生成用于各个亮度TB的PB,并且在亮度CB内从一个分区到下一个分区进行帧内重建处理。

[0184] 图16示出用于将图像帧的编码单元编码到视频位流115中的方法1600。方法1600可以由诸如配置的FPGA、ASIC或ASSP等的设备来实现。另外,方法1600可以由视频编码器114在处理器205的执行下进行。因此,方法1600可存储在计算机可读存储介质上和/或存储器206中。方法1600使得将块编码到位流115中,以使得仅基于变换块大小来确定系数组大小,并且不进一步在亮度通道和色度通道之间做区分。由于熵编码是视频编码器114中的关键反馈环路,因此减少系数组大小确定所需的存储器存取或计算是有利的。针对编码树中的各个编码单元而调用(即,在图13的步骤1330处调用)的方法1600在编码pred_mode步骤1610处开始。如上所述,当步骤1320确定为当前拆分是无拆分510时,执行步骤1330。

[0185] 在编码pred_mode步骤1610处,在处理器205的执行下,熵编码器338将CU的预测模式编码到位流115中。处理器205中的控制从步骤1610前进到帧内预测测试步骤1620。

[0186] 在帧内预测测试步骤1620处,处理器205测试CU的预测模式。如果预测模式是帧内预测(步骤1620处为“是”),则处理器205中的控制从步骤1620前进到编码帧内子分区模式步骤1650。否则,如果预测模式不是帧内预测(步骤1620处为“否”),则处理器205中的控制从步骤1620前进到编码合并标志和索引步骤1630。

[0187] 在编码合并标志和索引步骤1630处,在处理器205的执行下,熵编码器338将合并标志编码到位流115中以用信号通知“合并模式”用于(或不用于)帧间预测。合并模式使得从候选空间(或时间)相邻块的集合中的空间(或时间)相邻块获得CU的运动矢量。如果使用合并模式,则利用相应“合并索引”选择一个候选。合并索引与合并标志一起被编码在位流115中。如果使用“运动矢量预测”,则执行类似的编码,由此使用标志将若干可能的候选运动矢量之一作为预测子来用信号通知。处理器205中的控制从步骤1630前进到编码运动矢量增量步骤1640。

[0188] 在编码运动矢量增量步骤1640处,在处理器205的执行下,熵编码器338将运动矢量增量编码到位流115中。当针对CU使用运动矢量预测时,进行步骤1640。运动矢量增量指定在步骤1630处编码的运动矢量预测子和要用于运动补偿的运动矢量之间的增量。处理器205中的控制从步骤1640前进到编码残差测试步骤1660。如果针对CU未使用运动矢量预测,则不实现步骤1640且方法1600直接前进到步骤1660。

[0189] 在编码帧内子分区模式步骤1650处,在处理器205的执行下,熵编码器338利用上下文编码的“intra_subpartitions_mode_flag”句法元素将是否使用帧内子分区的决策编码到位流115中。当亮度CB大小大于最小亮度变换块大小(即,大于16个亮度样本)时,帧内子分区可用于亮度通道。帧内子分区将编码单元分割成多个亮度变换块,如集合1500中所示。如果亮度CB被分区成多个TB,则“intra_subpartitions_split_flag”用信号通知亮度CB分割成多个亮度TB是水平还是垂直发生的。共同地,“intra_subpartitions_mode_flag”和“intra_subpartitions_split_flag”对枚举为“ISP_NO_SPLIT”、“ISP_HOR_SPLIT”和“ISP_VER_SPLIT”的三个可能的分区进行编码。处理器205中的控制从步骤1650前进到经编码残差测试步骤1660。

[0190] 在经编码残差测试步骤1660处,处理器205确定编码块的任何变换块中的至少一个残差系数是否有效。该确定包括由帧内子分区的应用所得到的所有亮度TB以及与两个色度通道相关联的色度TB对。如果任意的亮度TB和色度TB中的至少一个残差系数有效,则熵编码器338在处理器205的执行下针对“cu_cbf”句法元素来算术编码“一”,步骤1660返回“是”并且处理器205前进到确定亮度系数组大小步骤1670。如果在CU的任意TB中不存在有效残差系数,则步骤1660返回“否”,针对cu_cbf来算术编码“零”并且方法1600终止,其中处理器205前进到CTU中的下一个CU。

[0191] 在确定亮度系数组大小步骤1670处,处理器确定与CU相关联的一个或多个亮度TB(变换块)的系数组大小。如果帧内子分区不处于使用中,则存在一个亮度TB。如果帧内子分区处于使用中,则存在两个或四个亮度TB。亮度TB的大小取决于水平或垂直进行的帧内子分区以及亮度TB的数量,并且相应地取决于亮度CU大小,如集合1500中所示。

[0192] 使用亮度TB宽度和高度来确定系数组大小,如以下表1中所示。表1示出用于亮度和色度通道的变换块(TB)大小到系数组映射表(具有针对TB的相同大小的系数组,而不管TB是用于亮度通道还是色度通道)。TB宽度和高度是2的幂,并且因此表1考虑TB宽度和高度的log2,即“log2TBwidth”和“log2TBheight”形成表1的三维中的前两个索引。表的最后维度区分系数组的宽度和高度。系数组维度被存储为log2width和log2height。例如,大小为16×16的TB得到表1中的编索引(4,4),其返回(2,2),从而指示系数组大小为4×4。大小(2×32)的TB得到表1中的编索引(1,5),其返回(1,3),从而指示系数组大小为2×8。亮度TB的最小区域是16个样本,并且因此不访问表1中log2width+log2height小于四的情况。在帧内子分区用于CU的情况下,各个亮度TB具有相同大小,并且因此对于CU进行一次针对亮度TB的系数组大小确定。

[0193] 以下表2示出针对亮度和色度通道的变换块(TB)大小到系数组大小的映射(具有针对与色度相比在亮度中的相同大小TB的不同系数组大小)。如果要使用表2,将需要附加维度,即,区分亮度与色度,并且表大小与表1相比是两倍。如表1中那样定义的系数组大小使得大小是适合TB宽度和高度同时在区域中不超过16个样本的最大可能大小。表1提供从中选择系数组大小的一组系数组大小。在TB宽度和高度的约束内,宽度vs高度的所选系数组长宽比保持尽可能接近1:1。处理器205中的控制从步骤1670前进到编码亮度TB步骤1680。

[0194] 在编码亮度TB步骤1680处,在处理器205的执行下,熵编码器338将CU的一个或多个亮度TB的残差系数编码到位流115中。步骤1670的所确定的系数组大小被用于各个亮度TB。对于各个亮度TB,经编码块标志被编码到位流115中,从而指示亮度TB中存在至少一个有效系数。如果亮度TB中存在至少一个有效系数,则将最后有效位置编码到位流中。最后有效位置被定义为沿着从TB的DC(左上)系数向右下系数前进的扫描路径的最后有效系数。扫描路径被定义为TB分割成非重叠子块的阵列内的对角线扫描,各个子块的大小为系数组大小,并且占据整个TB。按扫描顺序从一个子块到下一个子块的前进也遵循对角线扫描。对于除了左上系数组和包含最后有效系数的系数组之外的各个系数组,熵编码器338对“经编码子块标志”进行编码。经编码子块标志指示子块中存在至少一个有效残差系数。如果在子块中不存在有效残差系数,则TB中的残差系数的对角线扫描跳过该子块。如果在子块中存在至少一个有效残差系数,则扫描该子块内的所有位置,并且对各个残差系数的量值进行编

码,并且对各个有效残差系数的符号进行编码。处理器205中的控制从步骤1680前进到确定色度系数组大小步骤1690。

[0195] 在确定色度系数组大小步骤1690处,处理器205确定与CU相关联的色度变换块对的系数组大小。用于各个色度通道的一个色度CB与CU相关联,而与亮度CB是否被分割成多个亮度TB无关。如表1中所示,使用色度TB宽度和高度来确定系数组大小。TB宽度和高度是2的幂,并且因此表1考虑TB宽度和高度的log2,即,“log2TBwidth”和“log2TBheight”形成表1的三维中的前两个索引。表的最后维度区分系数组的宽度和高度。系数组维度被存储为log2width和log2height。例如,大小为16×16的TB得到表1中的编索引(4,4),其返回(2,2),从而指示系数组大小为4×4。大小(2×32)的TB得到表1中的编索引(1,5),其返回(1,3),从而指示系数组大小为2×8。各个色度TB具有相同大小,并且因此对于CU进行一次针对色度TB对的系数组大小确定。如果要使用表2,将需要附加维度,即区分亮度与色度,并且表大小与表1的表大小相比是两倍。

[0196] 如关于步骤1670和1690所描述,仅基于变换块大小来确定系数组大小,并且不进一步在亮度通道与色度通道之间做区分。因此与色度格式是4:2:2还是4:2:0无关地,确定系数组大小。如关于表1所述,系数组大小是基于多达16个样本的系数组的最大区域。步骤1690进行操作以确定针对TB的系数组大小,而不管变换块的颜色平面(Y或Cb或Cr)或由于色度格式引起的颜色平面(适用于Cb和Cr通道)中的子采样如何。在步骤1670和步骤1690这两者中使用表1。因此,单个表用于属于亮度平面和各个色度颜色平面的变换块。处理器205中的控制从步骤1690前进到编码色度TB步骤16100。

[0197] 在编码色度TB步骤16100处,在处理器205的执行下,熵编码器338将CU的色度TB对的残差系数编码到位流115中。步骤1690的所确定的系数组大小被用于色度TB对。对于各个色度TB,经编码块标志被编码到位流115中,从而指示色度TB中存在至少一个有效系数。如参考步骤1680所述,针对各个色度TB的编码步骤的剩余部分符合针对亮度TB的编码处理。在执行步骤16100时,方法1600终止,并且处理器205中的控制前进到CTU中的下一个CU。

[0198] 图17示出从视频位流133中解码图像帧的编码单元的方法1700。方法1700可以由诸如配置的FPGA、ASIC或ASSP等的设备来实现。另外,方法1700可以在处理器205的执行下由视频解码器134进行。因此,方法1700可以存储在计算机可读存储介质上和/或存储器206中。方法1700使得从位流133解码块,以使得仅基于变换块大小来确定系数组大小,并且不进一步在亮度通道与色度通道之间做区分。由于熵解码是视频解码器134中的关键反馈环路,因此减少系数组大小确定所需的存储器存取或计算是有利的。针对编码树中的各个编码单元调用(即,在图14的步骤1430处调用)方法1700。如上所述,如果当前拆分是无拆分510,则执行步骤1430。方法1700开始于解码pred_mode步骤1710。

[0199] 在解码pred_mode步骤1710处,在处理器205的执行下,熵解码器420从位流133中解码CU的预测模式。处理器205中的控制从步骤1710前进到帧内预测测试步骤1720。

[0200] 在帧内预测测试步骤1720处,处理器205测试CU的预测模式(如在步骤1710处解码的)。如果预测模式是帧内预测,则步骤1720返回“是”并且处理器205中的控制从步骤1720前进到解码帧内子分区模式步骤1750。否则,如果不是帧内预测,则步骤1720返回“否”并且处理器205中的控制从步骤1720前进到解码合并标志和索引步骤1730。

[0201] 在解码合并标志和索引步骤1730处,在处理器205的执行下,熵解码器420从位流

133中解码合并标志,从而用信号通知“合并模式”是否在位流中用于帧间预测。合并模式使得从候选空间或时间相邻块的集合中的空间(或时间)相邻块获得CU的运动矢量。如果使用合并模式,则利用(也从位流133解码的)“合并索引”选择一个候选。如果使用“运动矢量预测”,则进行类似的解码,从而通过位流中的标志将若干可能的候选运动矢量之一作为预测子来用信号通知。处理器205中的控制从步骤1730前进到解码运动矢量增量步骤1740。

[0202] 在解码运动矢量增量步骤1740处,在处理器205的执行下,熵解码器420从位流133中解码运动矢量增量。当针对CU使用运动矢量预测时,进行步骤1740。运动矢量增量指定在步骤1730处编码的运动矢量预测子和要用于运动补偿的运动矢量之间的增量。处理器205中的控制从步骤1740前进到经编码残差测试步骤1760。如果针对CU未使用运动矢量预测,则不实现步骤1740,并且处理器205中的控制直接前进到步骤1760。

[0203] 在解码帧内子分区模式步骤1750处,在处理器205的执行下,熵解码器420使用上下文编码的“intra_subpartitions_mode_flag”句法元素来从位流133中解码是否使用帧内子分区的决策。当亮度CB大小大于最小亮度变换块大小(即,大于16个亮度样本)时,帧内子分区可用于亮度通道。帧内子分区将编码单元分割成多个亮度变换块,如集合1500中所示。如果亮度CB被分区成多个TB,则“intra_subpartitions_split_flag”用信号通知亮度CB分割成多个亮度TB是水平还是垂直发生的。共同地,“intra_subpartitions_mode_flag”和“intra_subpartitions_split_flag”对枚举为“ISP_NO_SPLIT”、“ISP_HOR_SPLIT”和“ISP_VER_SPLIT”的三个可能的分区进行编码。处理器205中的控制从步骤1750前进到经编码残差测试步骤1760。

[0204] 在经编码残差测试步骤1760处,处理器205确定编码块的任何变换块中的至少一个残差系数是否有效。该确定包括由帧内子分区的应用所得到的所有亮度TB以及与两个色度通道相关联的色度TB对。在处理器205的执行下,熵编码器420对“cu_cbf”句法元素进行算术解码,并且处理器205确定CU的任意TB中的至少一个残差系数是否有效。如果任意点亮度TB和色度TB中的至少一个残差系数是有效的,则步骤1760返回“是”并且处理器205中的控制前进到确定亮度系数组大小步骤1770。如果在CU的任意TB中不存在有效残差系数(如由针对cu_cbf进行算术解码的“零”所指示的),则步骤1760返回“否”并且方法1700终止,其中处理器205前进到CTU中的下一个CU。

[0205] 在确定亮度系数组大小步骤1770处,处理器205确定与CU相关联的一个或多个亮度变换块的系数组大小。步骤1770的确定以与步骤1670的确定相同的方式来操作。处理器205中的控制从步骤1770前进到解码亮度TB步骤1780。

[0206] 在解码亮度TB步骤1780处,在处理器205的执行下,熵解码器420从位流133中解码CU的一或多个亮度TB的残差系数。步骤1770的所确定的系数组大小被用于各个亮度TB。对于各个亮度TB,从位流133中解码经编码块标志,其指示亮度TB中存在至少一个有效系数。如果亮度TB中存在至少一个有效系数,则从位流中解码最后有效位置。最后有效位置被定义为沿着从TB的DC(左上)系数向右下系数前进的扫描路径的最后有效系数。扫描路径被定义为TB分割成非重叠子块的阵列内的对角线扫描,各个子块的大小为系数组大小,并且占据整个TB。按扫描顺序从一个子块到下一子块的前进也遵循对角线扫描。对于除了左上系数组和包含最后有效系数的系数组之外的各个系数组,熵编码器338对“经编码子块标志”进行编码。经编码子块标志指示子块中存在至少一个有效残差系数。如果在子块中不存在

有效残差系数,则TB中的残差系数的对角线扫描跳过该子块。如果在子块中存在至少一个有效残差系数,则扫描该子块内的所有位置,并且对各个残差系数的量值进行编码,并且对各个有效残差系数的符号进行编码。处理器205中的控制从步骤1780前进到确定色度系数组大小步骤1790。

[0207] 在确定色度系数组大小步骤1790处,处理器205确定与CU相关联的色度变换块对的系数组大小。在步骤1790中进行的确定以与在步骤1690中进行的确定相同的方式来操作。

[0208] 类似于步骤1690,在步骤1790处基于变换块大小来确定系数组大小,并且不进一步在亮度通道与色度通道之间做区分。因此与色度格式是4:2:2还是4:2:0或者各个颜色平面中的相应子采样无关地,确定系数组大小。如关于表1所述,系数组大小是基于多达16个样本的TB的最大区域。步骤1690进行操作以确定TB的系数组大小,而不管变换块的颜色平面(Cb或Cr)如何。处理器205中的控制从步骤1790前进到解码色度TB步骤17100。

[0209] 在解码色度TB步骤17100处,在处理器205的执行下,熵解码器420从位流133中解码CU的色度TB对的残差系数。步骤1790的所确定的系数组大小用于色度TB对。对于各个色度TB,从位流133中解码经编码块标志,其指示在色度TB中存在至少一个有效系数。如参考步骤1780所描述的,针对各个色度TB的解码处理的剩余部分以与针对亮度TB相同的方式来操作。方法1700在步骤17100的执行时终止并且处理器205中的控制前进到CTU中的下一个CU。

[0210] 表3示出在JVET“通用测试条件”(CTC)-“全帧内(All Intra)Main10”配置下使用表1时获得的编码性能结果。表3的结果是与未实现方法1600和1700的基线VTM-4.0相比使用实现方法1600和1700的“VVC测试模型”(VTM)软件所获得的。总的来说,该改变没有编码影响,并且甚至在色度通道中看到一些小的增益,从而证明简化变换块大小到系数组大小映射表对编码性能无害。

[0211] 视频编码器115和视频解码器134分别使用方法1600和1700,通过协调亮度TB和色度TB的系数组大小来实现残差编码/解码处理中的存储器减少。因此,色度TB有权使用诸如2×8、8×2、2×4、4×2代替仅是2×2和4×4的系数组大小。对于亮度TB,当使用帧内子分区时,16×1和1×16的大小是可能的。尽管大小16×1和1×16可用于色度(因为它们在表1中存在),色度块的最小宽度和高度是两个样本,因此大小16×1和1×16不用于色度TB。由于残差编码和解码在设计中是反馈环路的一部分,因此存储器减少对应于例如软件实现中的高速缓存性能或硬件实现中的关键路径减少的改进。

[0212]
```
uint32_t g_log2SbbSize[MAX_CU_DEPTH + 1][MAX_CU_DEPTH + 1][2] =
//===== 亮度/色度 =====
{
  { { 0,0 },{ 0,1 },{ 0,2 },{ 0,3 },{ 0,4 },{ 0,4 },{ 0,4 },{ 0,4 } },
  { { 1,0 },{ 1,1 },{ 1,2 },{ 1,3 },{ 1,3 },{ 1,3 },{ 1,3 },{ 1,3 } },
  { { 2,0 },{ 2,1 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 } },
  { { 3,0 },{ 3,1 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 } },
  { { 4,0 },{ 3,1 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 } },
  { { 4,0 },{ 3,1 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 } },
  { { 4,0 },{ 3,1 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 } },
  { { 4,0 },{ 3,1 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 },{ 2,2 } }
}
```

[0213] 表1:针对亮度和色度通道的变换块大小到系数组映射表(具有针对TB的相同大小的系数组,而不管TB是用于亮度通道还是色度通道)

[0214]
```
uint32_t g_log2SbbSize[2][MAX_CU_DEPTH+1][MAX_CU_DEPTH+1][2] =
{
  //===== 亮度 =====
  {
    { {0,0}, {0,1}, {0,2}, {0,3}, {0,4}, {0,4}, {0,4}, {0,4} },
    { {1,0}, {1,1}, {1,2}, {1,3}, {1,3}, {1,3}, {1,3}, {1,3} },
    { {2,0}, {2,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} },
    { {3,0}, {3,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} },
    { {4,0}, {3,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} },
    { {4,0}, {3,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} },
    { {4,0}, {3,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} },
    { {4,0}, {3,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} }
  },
```

[0215]
```
  //===== 色度 =====
  {
    { {0,0}, {0,0}, {0,0}, {0,0}, {0,0}, {0,0}, {0,0}, {0,0} },
    { {0,0}, {1,1}, {1,1}, {1,1}, {1,1}, {1,1}, {1,1}, {1,1} },
    { {0,0}, {1,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} },
    { {0,0}, {1,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} },
    { {0,0}, {1,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} },
    { {0,0}, {1,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} },
    { {0,0}, {1,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} },
    { {0,0}, {1,1}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2}, {2,2} }
  },
}
```

[0216] 表2:针对亮度和色度通道的变换块大小到系数组大小的常规映射(具有针对亮度vs色度中的相同大小TB的不同系数组大小)

[0217]

	全帧内 Main10				
	相对于 VTM-4.0				
	Y	U	V	EncT	DecT
类 A1	-0.01%	-0.03%	0.00%	97%	97%
类 A2	-0.02%	0.05%	-0.04%	95%	91%
类 B	0.00%	-0.02%	-0.03%	95%	93%
类 C	0.01%	-0.16%	-0.03%	97%	94%
类 E	0.00%	-0.01%	-0.14%	97%	95%
总体	0.00%	-0.04%	-0.05%	96%	94%
类 D	0.01%	-0.06%	-0.23%	97%	91%
类 F	0.01%	-0.07%	-0.13%	97%	94%

[0218] 表3:从具有针对TB(无论TB是用于亮度通道还是色度通道)的相同大小系数组所得到的编码性能

[0219] 产业上的可利用性

[0220] 所描述的布置适用于计算机和数据处理行业,并且特别地适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,从而实现高压缩效率。

[0221] 与HEVC对比,VVC系统允许使用针对亮度和色度通道的单独编码树,以提高灵活性。然而,如上所述,由于使用较小的色度块会影响吞吐量,因此可能出现所导致的问题。本文所描述的布置确定各编码树单元被处理时的适当规则以帮助避免吞吐量问题。另外,如上所述,在考虑避免吞吐量问题的规则的情况下,所描述的布置可以帮助提供用于描述各编码树的上下文编码bin的算术编码的改进的效率和准确性。

[0222] 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。

[0223] 相关专利申请的引用

[0224] 本申请根据35U.S.C.§119要求于2019年3月11日提交的澳大利亚专利申请2019201653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。

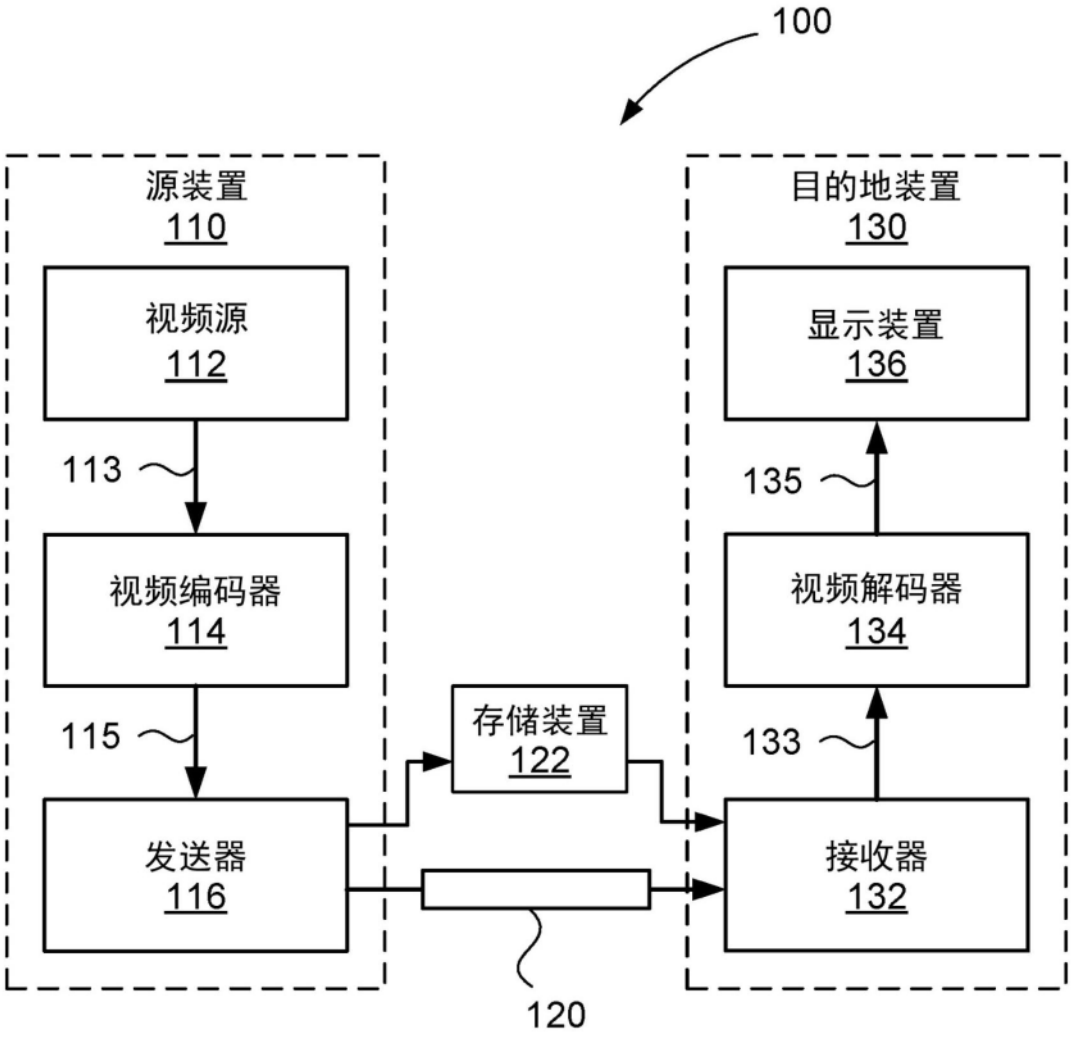

100
源装置
110
视频源
112
113
视频编码器
114
115
发送器
116
存储装置
122
120
目的地装置
130
显示装置
136
135
视频解码器
134
133
接收器
132

图1

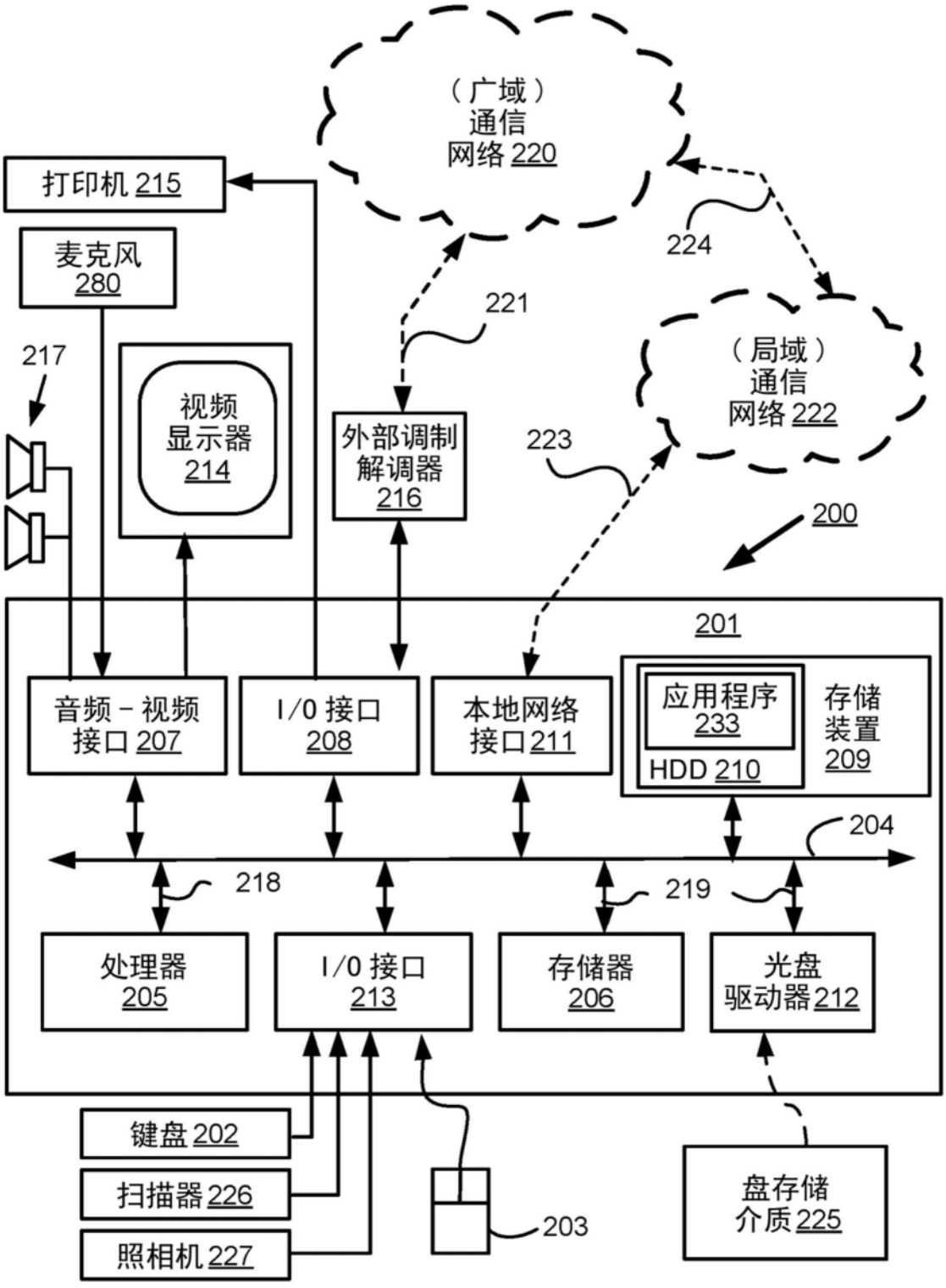

（广域）
通信
网络 220
打印机 215
麦克风
280
224
221
217
视频
显示器
214
外部调制
解调器
216
（局域）
通信
网络 222
223
200
201
音频－视频
接口 207
I/0 接口
208
本地网络
接口 211
应用程序
233
HDD 210
存储
装置
209
204
218
219
处理器
205
I/0 接口
213
存储器
206
光盘
驱动器 212
键盘 202
扫描器 226
照相机 227
203
盘存储
介质 225

图2A

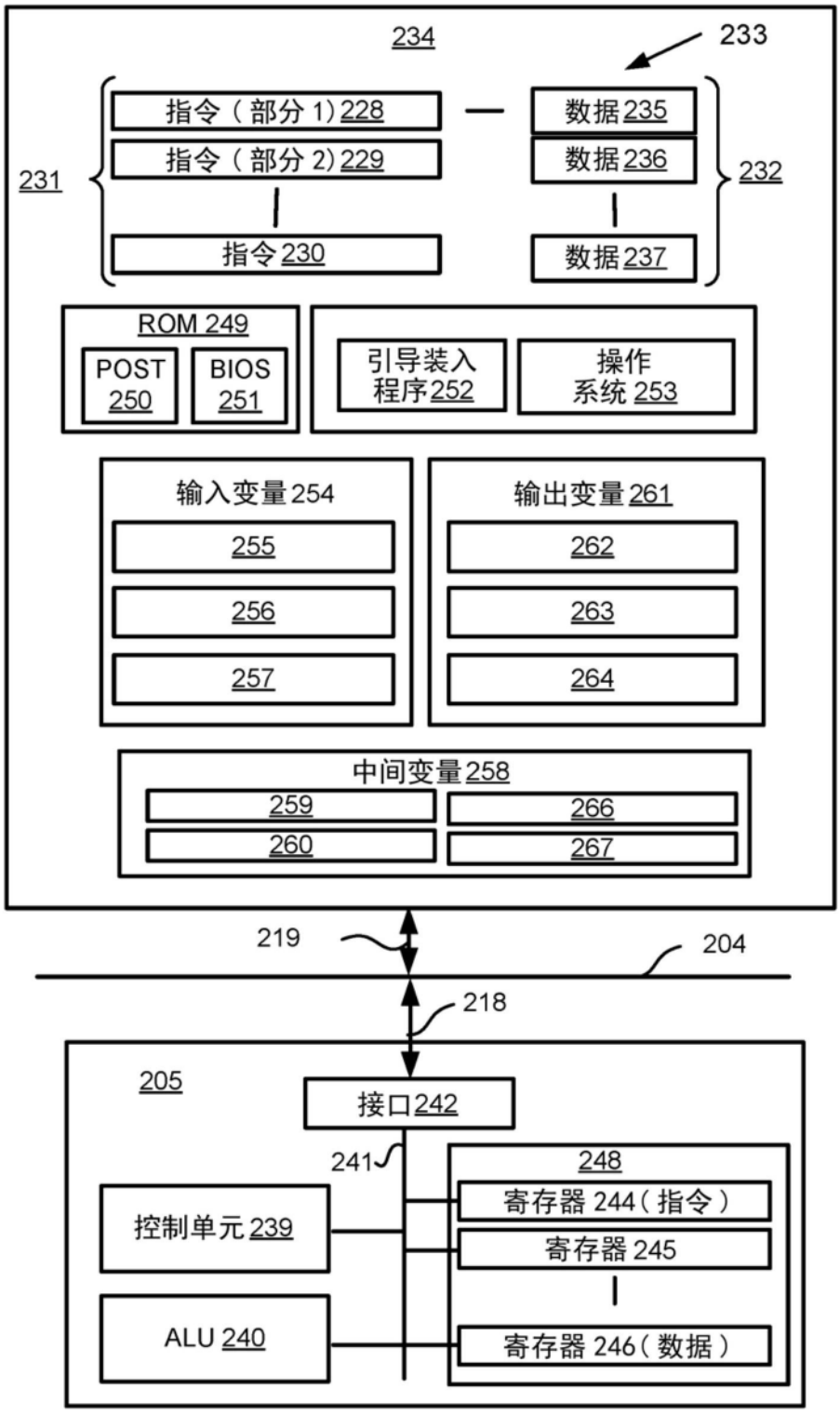

234
233
指令（部分 1）228
指令（部分 2）229
指令 230
231
数据 235
数据 236
数据 237
232
ROM 249
POST 250
BIOS 251
引导装入程序 252
操作系统 253
输入变量 254
255
256
257
输出变量 261
262
263
264
中间变量 258
259
260
266
267
219
204
218
205
接口 242
241
248
寄存器 244（指令）
寄存器 245
寄存器 246（数据）
控制单元 239
ALU 240

图2B

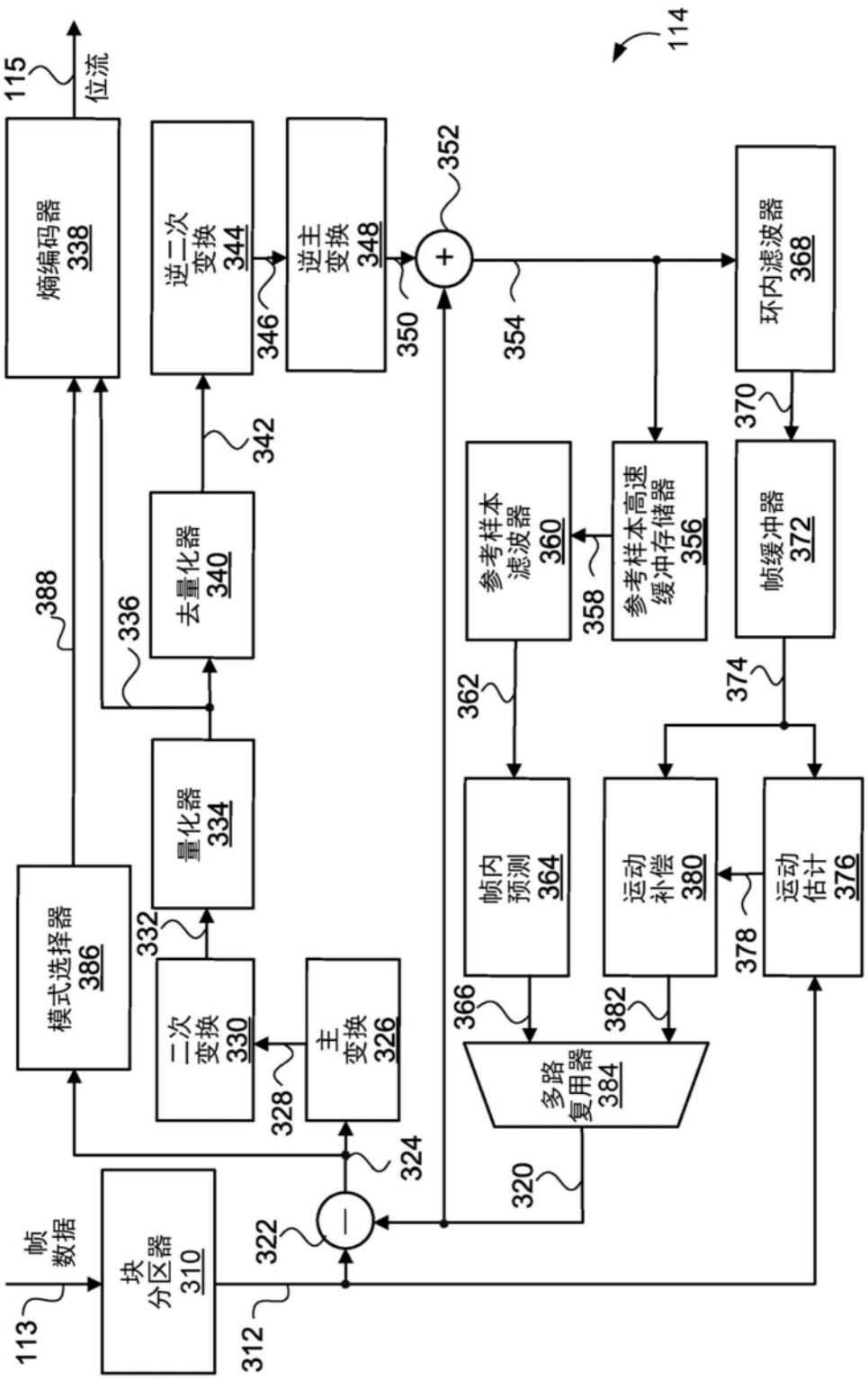

114
115
位流
熵编码器 338
逆二次变换 344
逆主变换 348
352
环内滤波器 368
370
帧缓冲器 372
参考样本高速缓冲存储器 356
参考样本滤波器 360
358
388
336
去量化器 340
342
346
350
354
362
374
量化器 334
模式选择器 386
二次变换 330
主变换 326
332
328
帧内预测 364
运动补偿 380
运动估计 376
366
382
378
多路复用器 384
320
324
322
帧数据
113
块分区器 310
312

图3

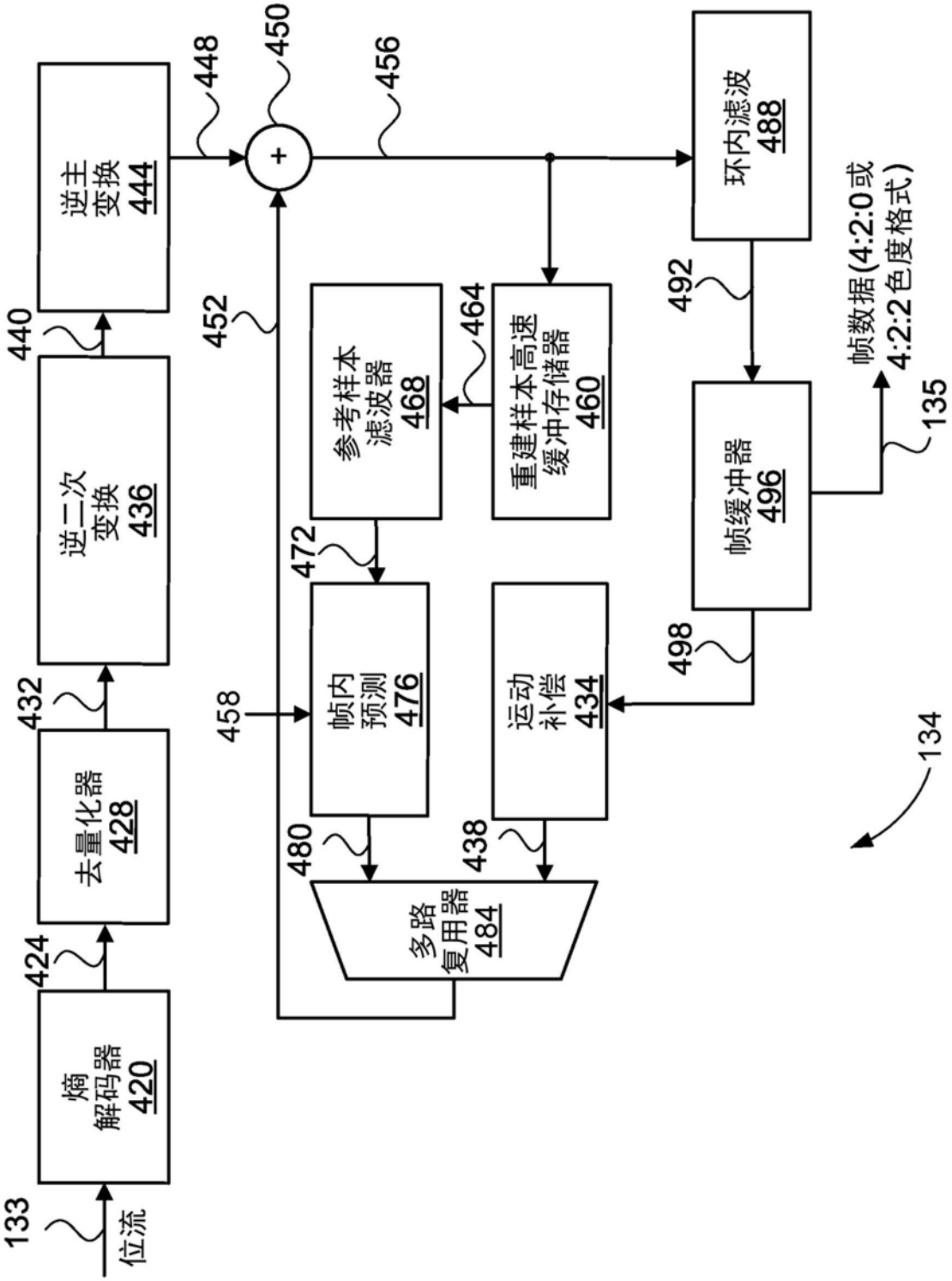

133
位流
熵
解码器
420
424
去量化器
428
432
逆二次
变换
436
440
逆主
变换
444
448
450
+
456
环内滤波
488
492
帧数据(4:2:0或
4:2:2色度格式)
135
帧缓冲器
496
498
452
参考样本
滤波器
468
464
重建样本高速
缓冲存储器
460
472
458
帧内
预测
476
运动
补偿
434
480
438
多路
复用器
484
134

图4

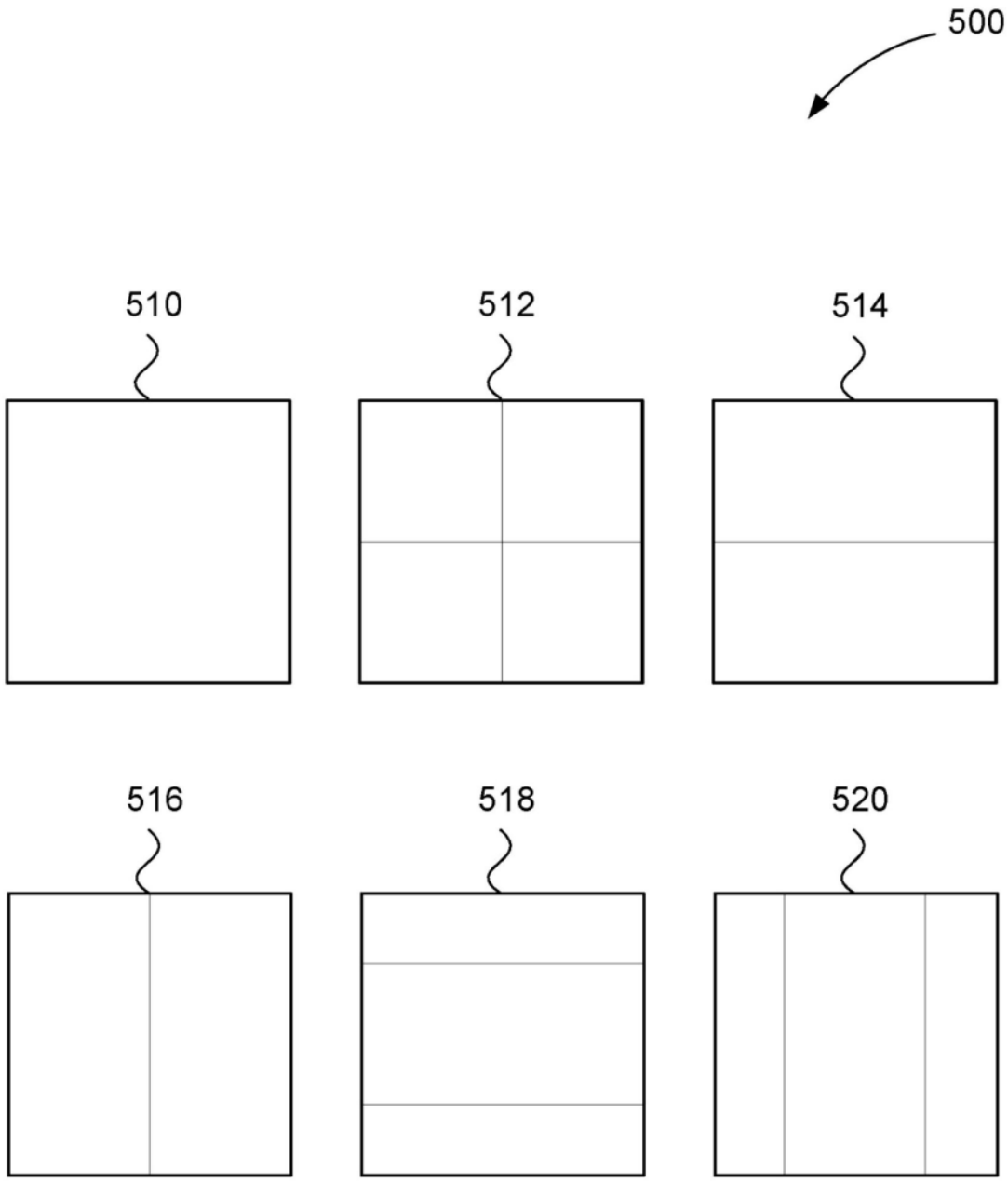


图5

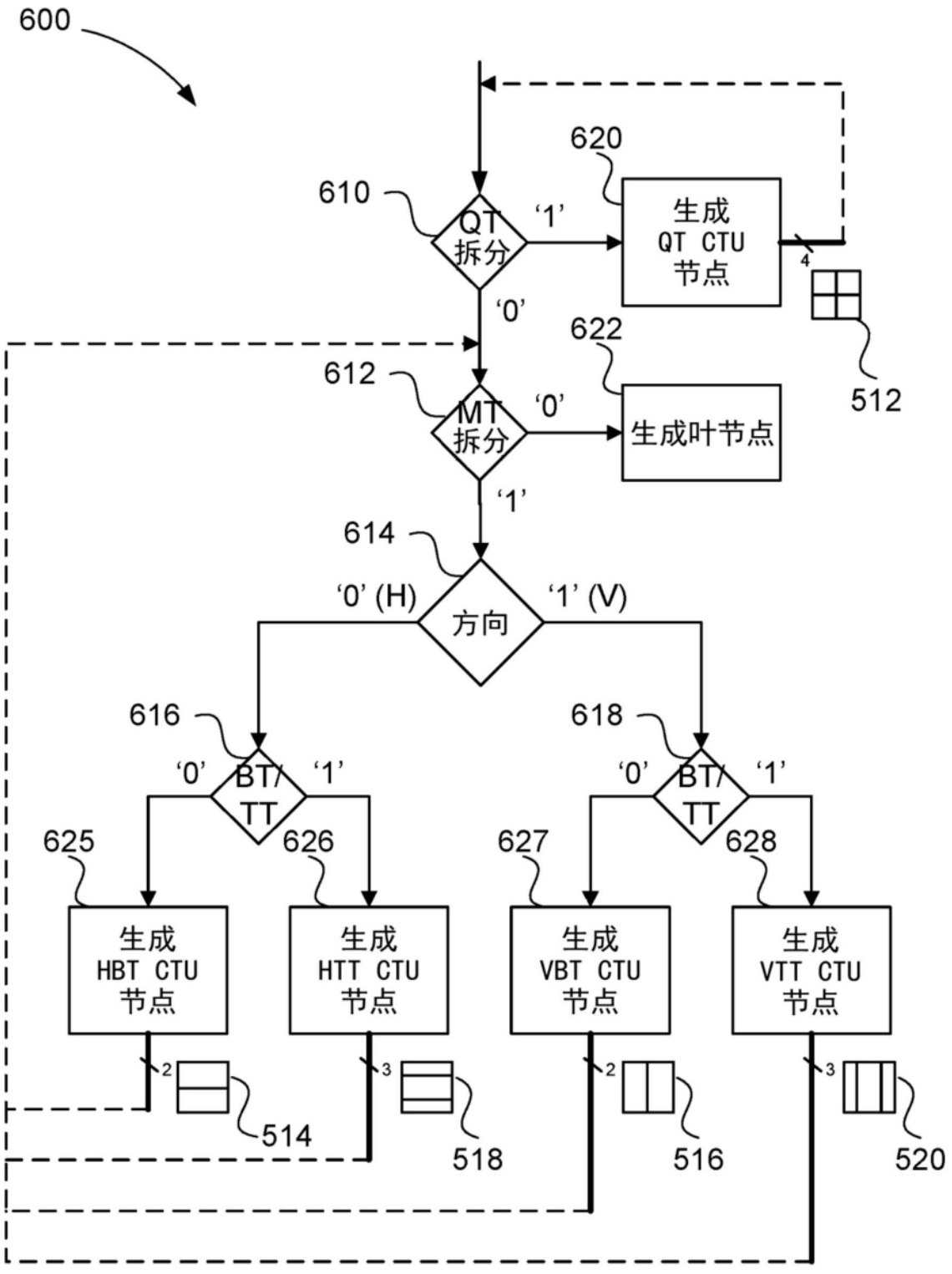

600
610
QT
拆分
'1'
620
生成
QT CTU
节点
4
512
'0'
612
MT
拆分
'0'
622
生成叶节点
'1'
614
'0' (H)
方向
'1' (V)
616
'0'
BT/
TT
'1'
618
'0'
BT/
TT
'1'
625
626
627
628
生成
HBT CTU
节点
生成
HTT CTU
节点
生成
VBT CTU
节点
生成
VTT CTU
节点
2
3
2
3
514
518
516
520

图6

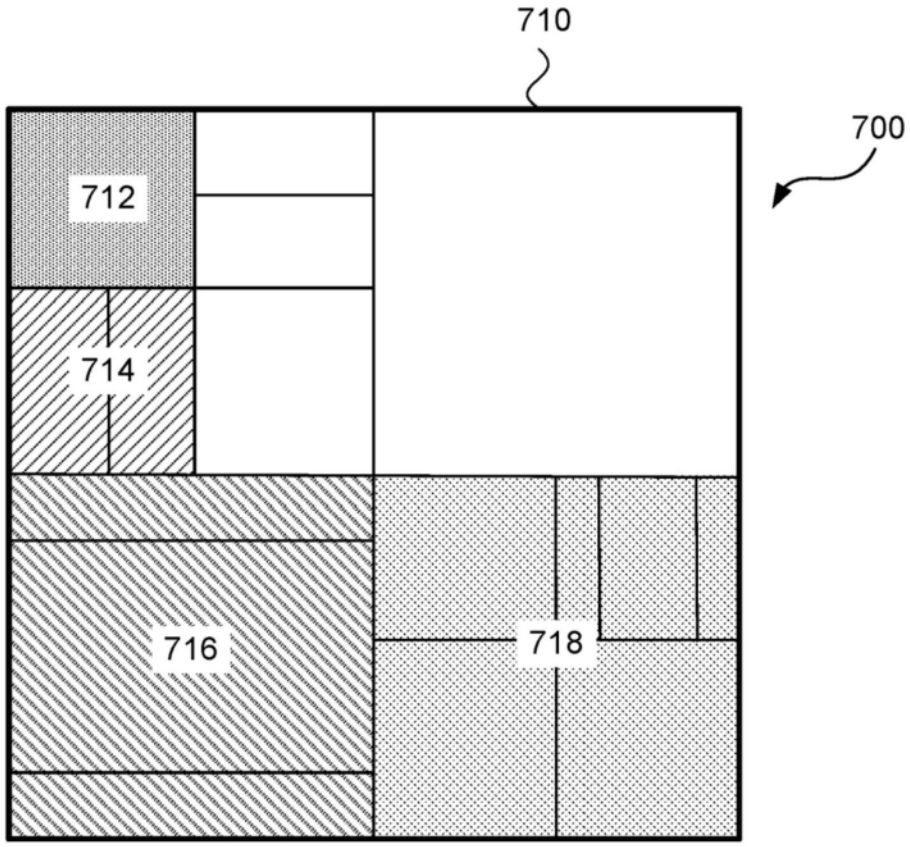

710
700
712
714
716
718

图7A

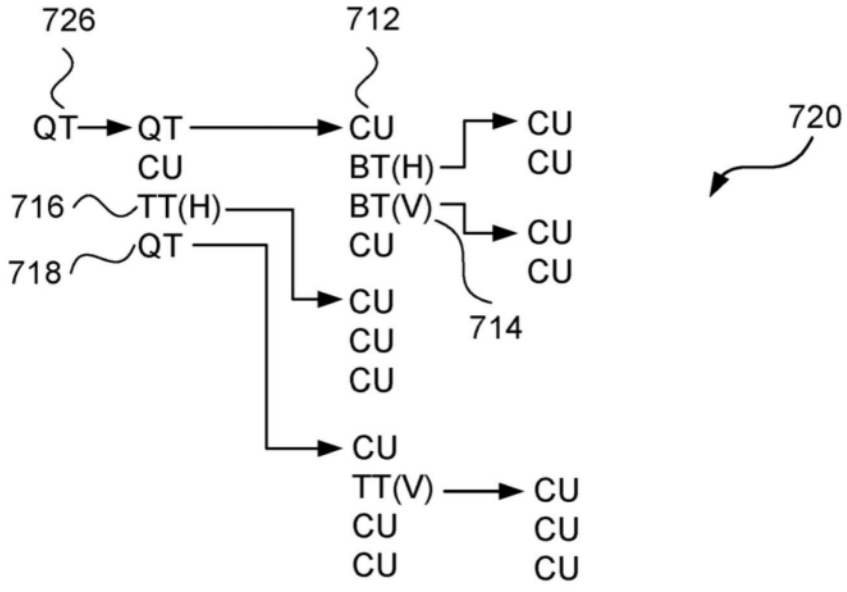

726
712
720
QT
QT
CU
CU
BT(H)
CU
CU
716
TT(H)
BT(V)
CU
CU
718
QT
CU
714
CU
CU
CU
CU
TT(V)
CU
CU
CU
CU

图7B

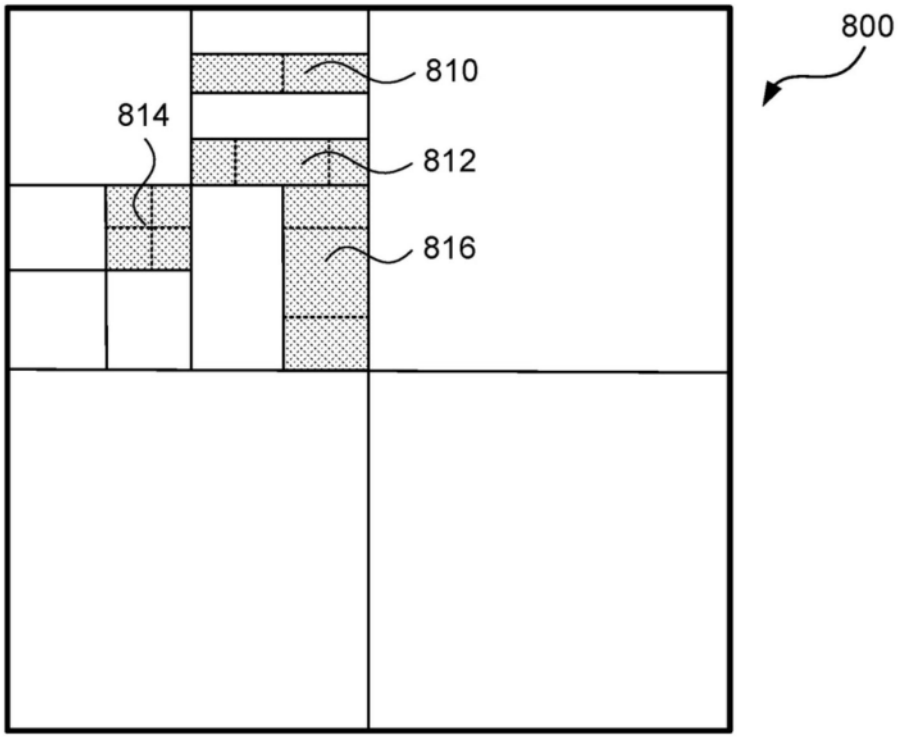

800
810
814
812
816

图8A

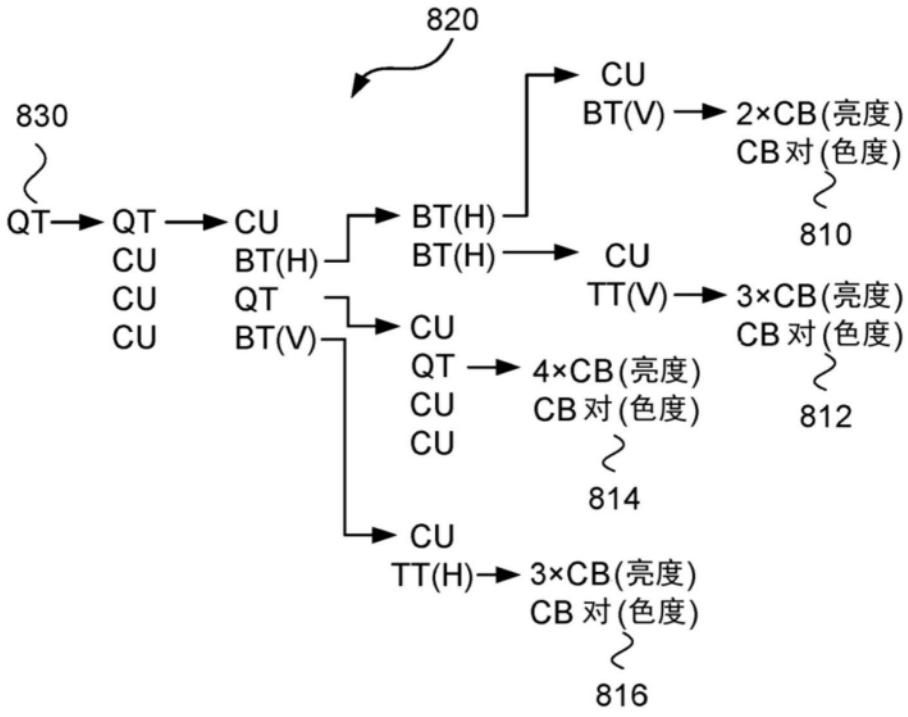

820
830
QT
QT
CU
CU
CU
CU
BT(H)
QT
BT(V)
BT(H)
BT(H)
CU
BT(V)
2×CB(亮度)
CB对(色度)
810
CU
TT(V)
3×CB(亮度)
CB对(色度)
812
CU
QT
CU
CU
4×CB(亮度)
CB对(色度)
814
CU
TT(H)
3×CB(亮度)
CB对(色度)
816

图8B

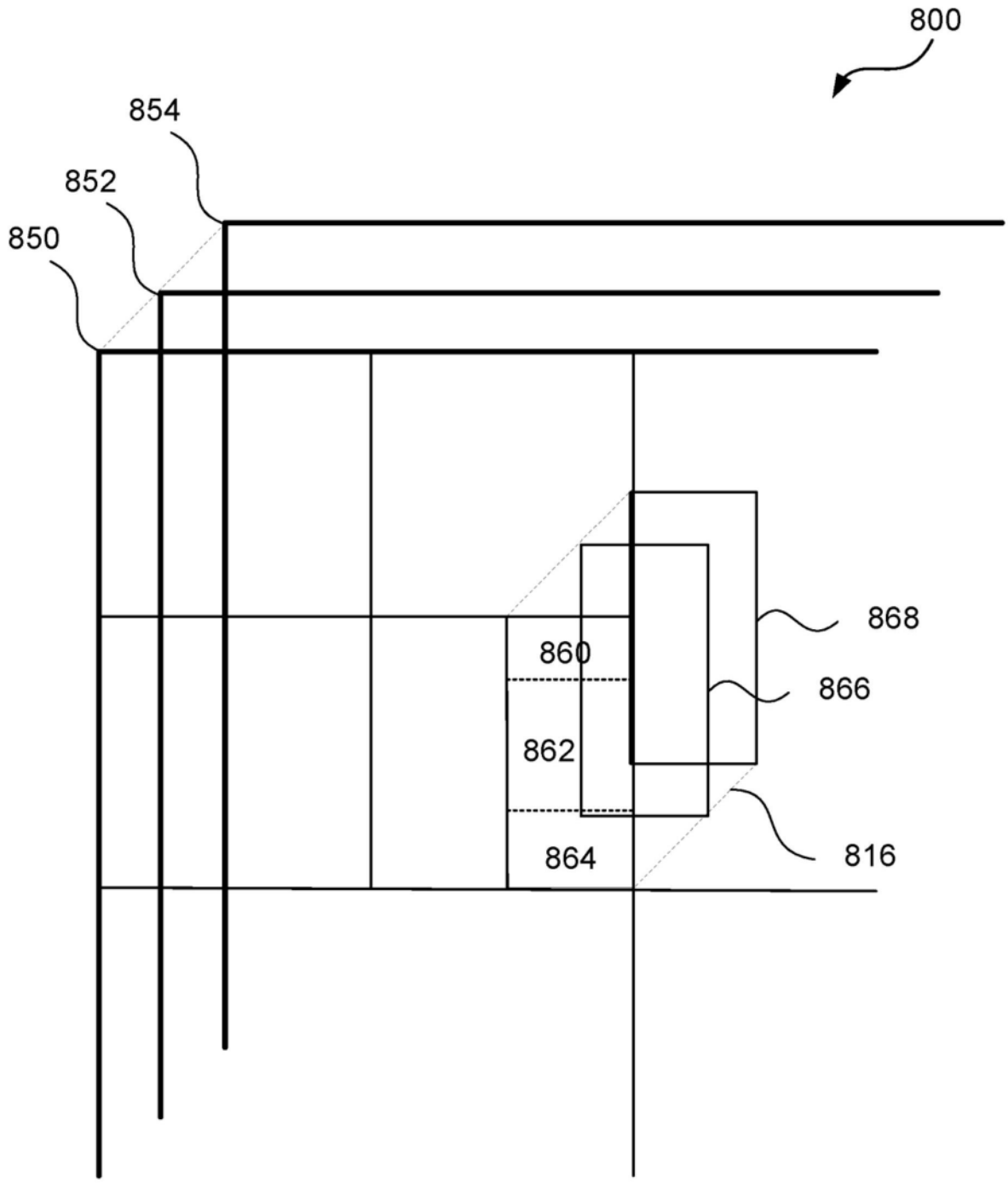


图8C

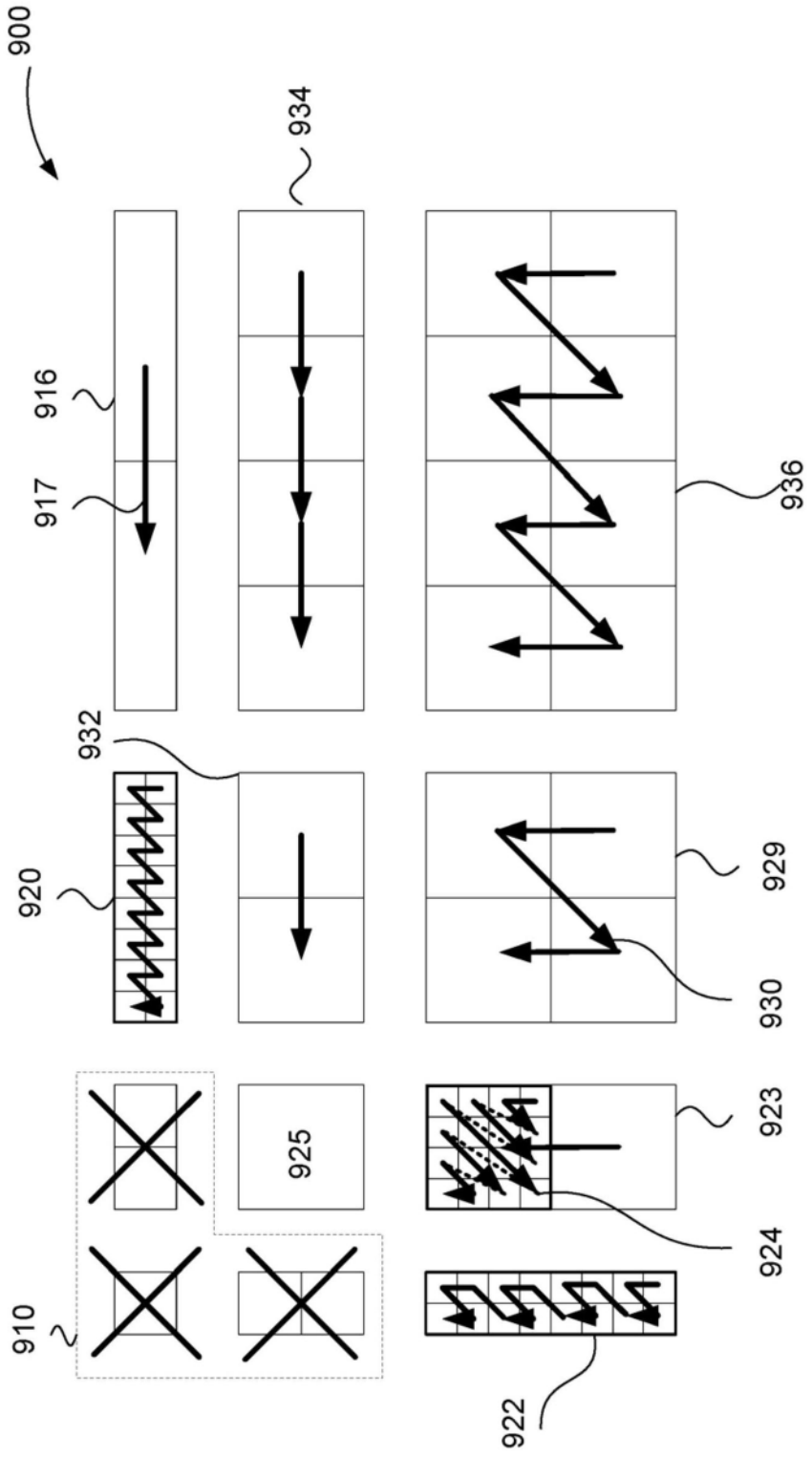


图9

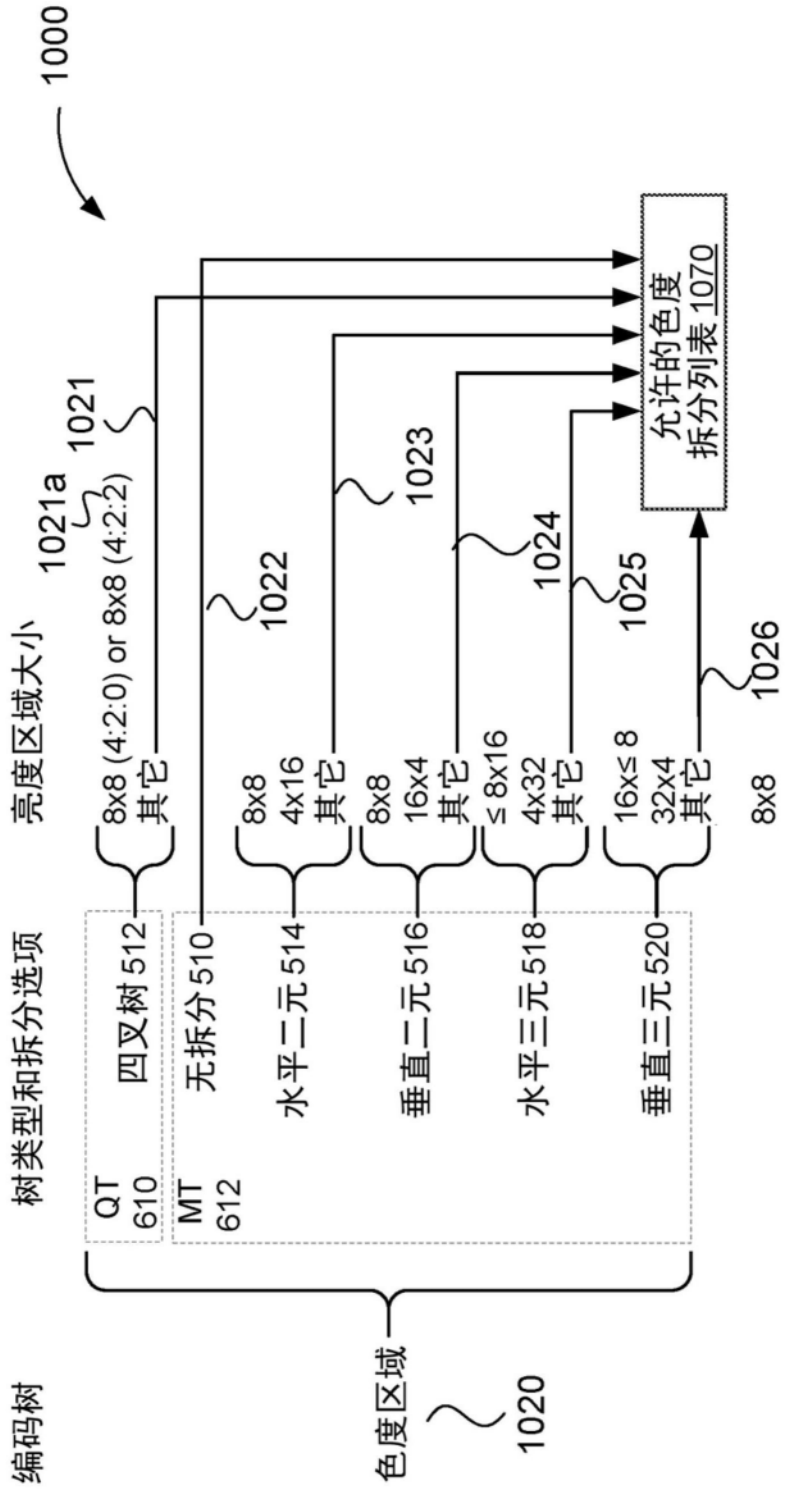

1000
编码树
色度区域
1020
树类型和拆分选项
QT
610
四叉树 512
MT
612
无拆分 510
水平二元 514
垂直二元 516
水平三元 518
垂直三元 520
亮度区域大小
1021a
1021
8x8 (4:2:0) or 8x8 (4:2:2)
其它
1022
8x8
4x16
其它
1023
8x8
16x4
其它
1024
≤ 8x16
4x32
其它
1025
16x≤ 8
32x4
其它
8x8
1026
允许的色度
拆分列表 1070

图10

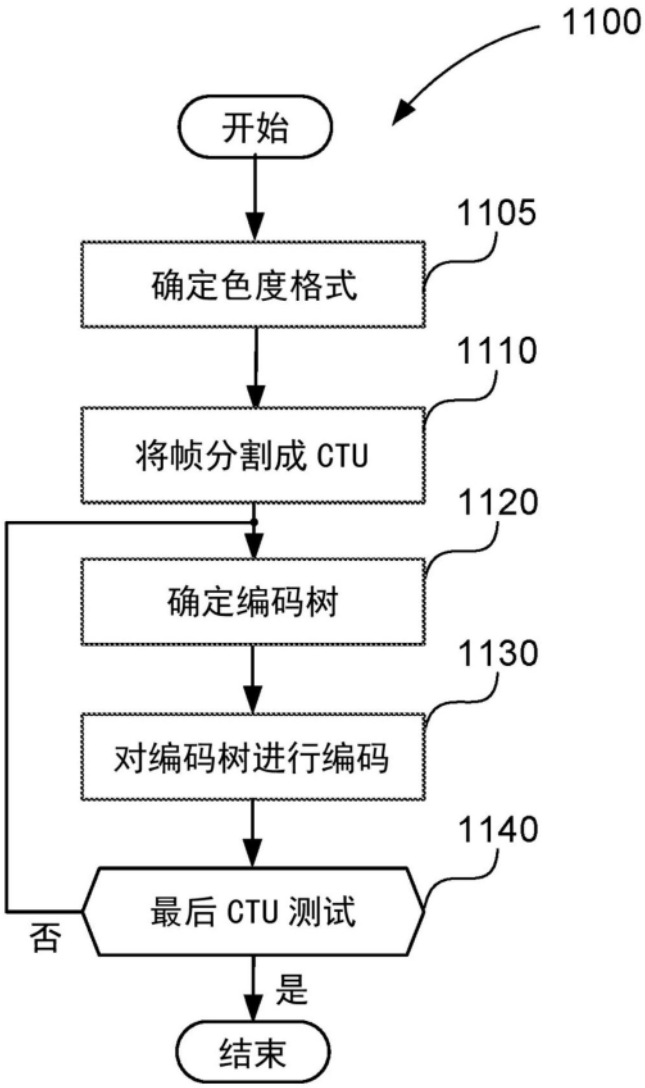

1100
开始
1105
确定色度格式
1110
将帧分割成 CTU
1120
确定编码树
1130
对编码树进行编码
1140
最后 CTU 测试
否
是
结束

图11

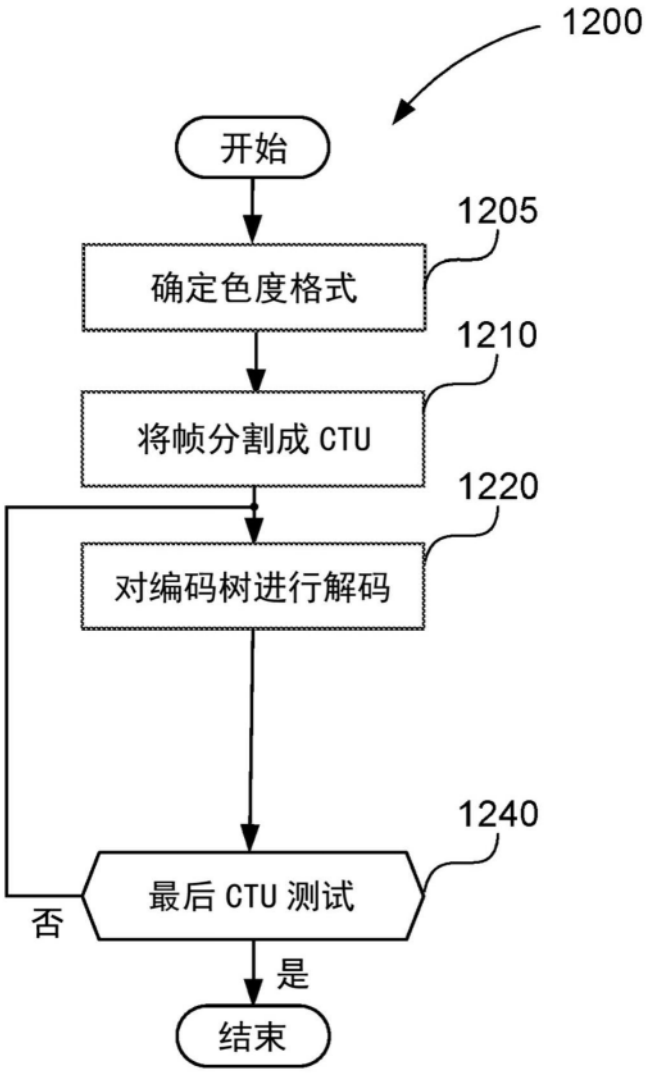

1200
开始
1205
确定色度格式
1210
将帧分割成 CTU
1220
对编码树进行解码
1240
最后 CTU 测试
否
是
结束

图12

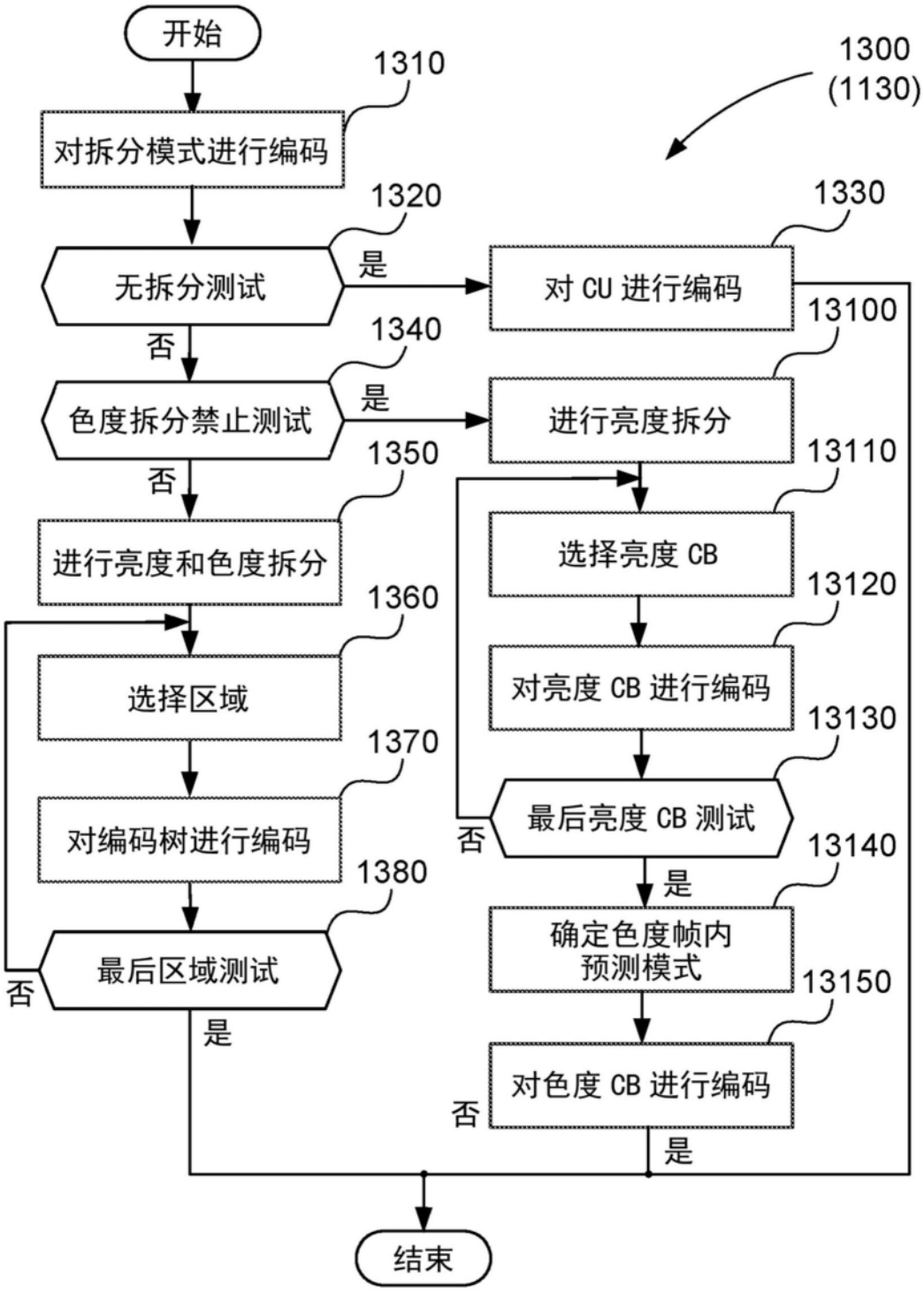

开始
1310
对拆分模式进行编码
1320
无拆分测试
是
否
1340
色度拆分禁止测试
是
否
1350
进行亮度和色度拆分
1360
选择区域
1370
对编码树进行编码
1380
最后区域测试
否
是
1300
(1130)
1330
对 CU 进行编码
13100
进行亮度拆分
13110
选择亮度 CB
13120
对亮度 CB 进行编码
13130
最后亮度 CB 测试
否
13140
是
确定色度帧内
预测模式
13150
对色度 CB 进行编码
否
是
结束

图13

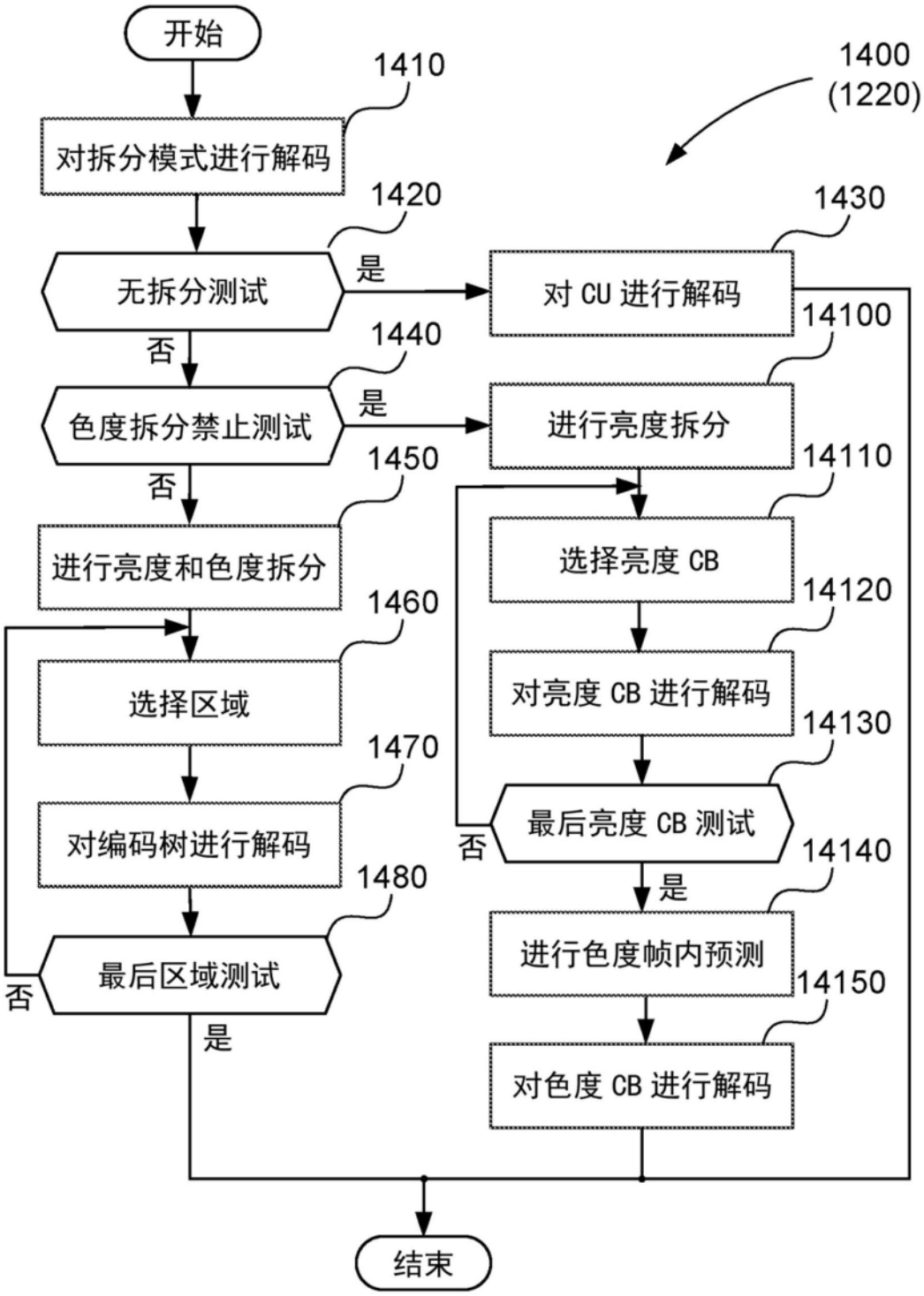

开始
1410
对拆分模式进行解码
1420
无拆分测试
是
否
1440
色度拆分禁止测试
是
否
1450
进行亮度和色度拆分
1460
选择区域
1470
对编码树进行解码
1480
最后区域测试
否
是
结束
1400
(1220)
1430
对 CU 进行解码
14100
进行亮度拆分
14110
选择亮度 CB
14120
对亮度 CB 进行解码
14130
最后亮度 CB 测试
否
是
14140
进行色度帧内预测
14150
对色度 CB 进行解码

图14

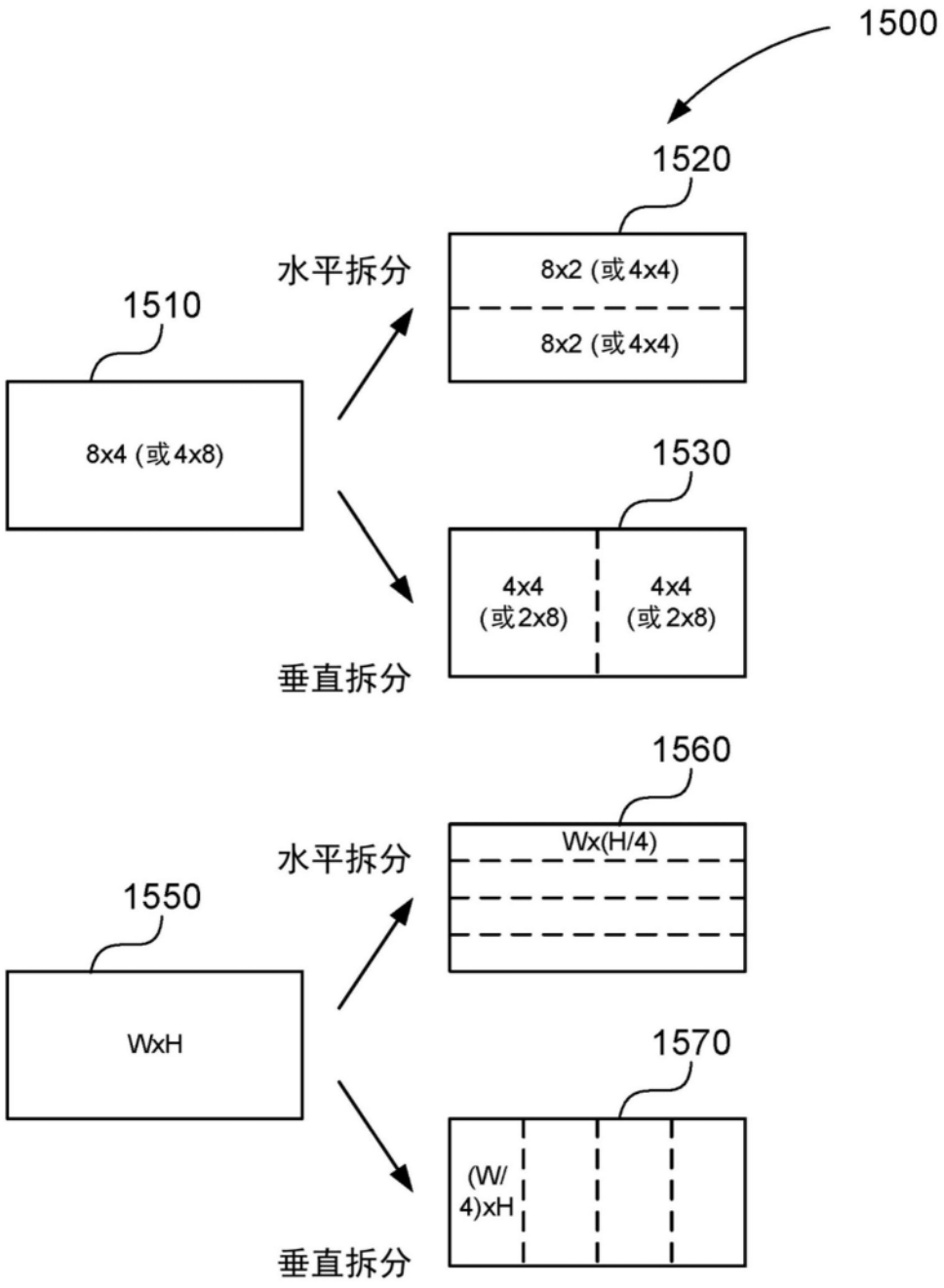

1500
1520
水平拆分
8x2 (或4x4)
8x2 (或4x4)
1510
8x4 (或4x8)
1530
4x4
(或2x8)
4x4
(或2x8)
垂直拆分
1560
Wx(H/4)
水平拆分
1550
WxH
1570
(W/
4)xH
垂直拆分

图15

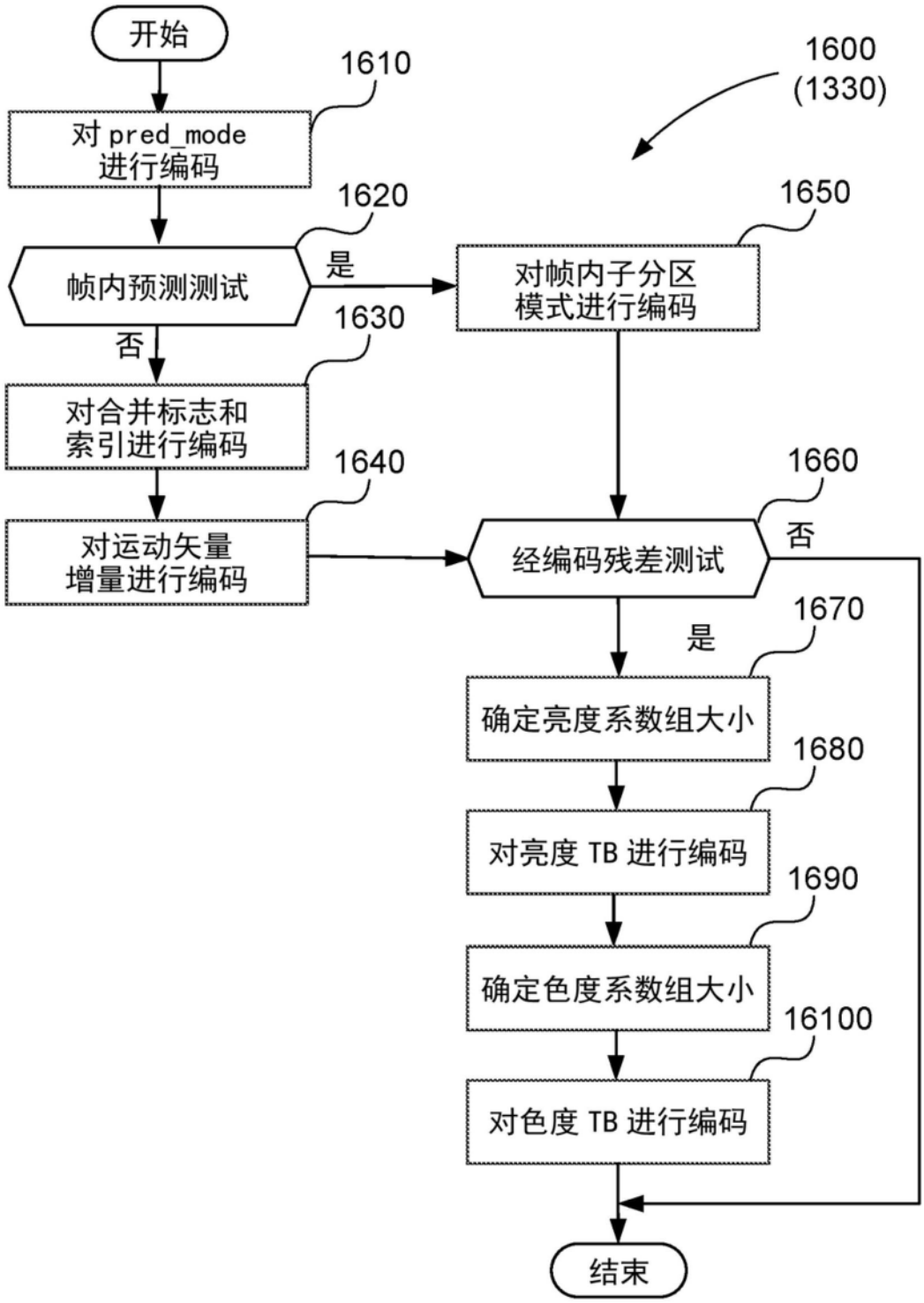

开始
1610
对 pred_mode
进行编码
1620
帧内预测测试
是
否
1630
对合并标志和
索引进行编码
1640
对运动矢量
增量进行编码
1600
(1330)
1650
对帧内子分区
模式进行编码
1660
经编码残差测试
否
是
1670
确定亮度系数组大小
1680
对亮度 TB 进行编码
1690
确定色度系数组大小
16100
对色度 TB 进行编码
结束

图16

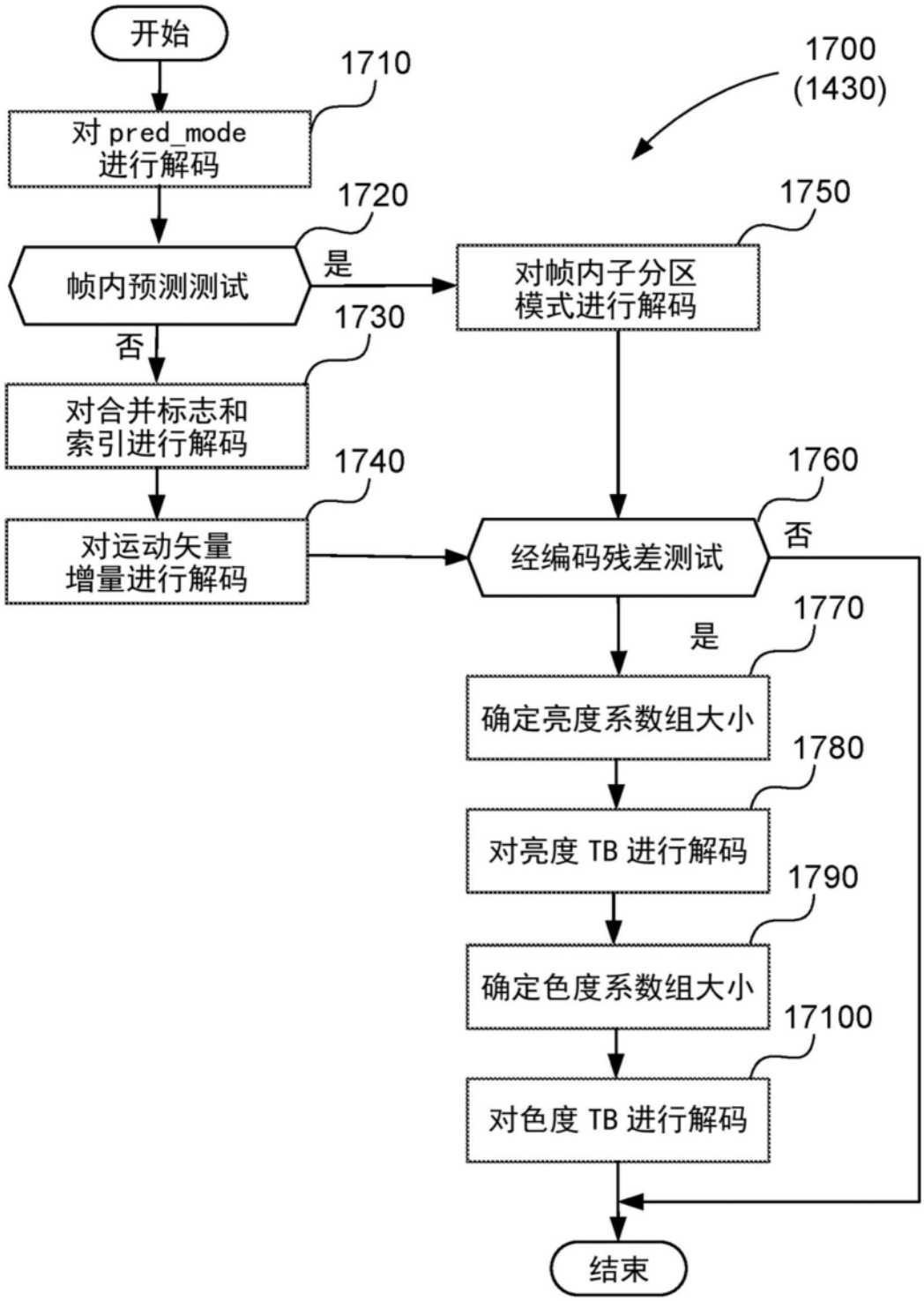

开始
1710
对 pred_mode
进行解码
1720
帧内预测测试
是
否
1730
对合并标志和
索引进行解码
1740
对运动矢量
增量进行解码
1700
(1430)
1750
对帧内子分区
模式进行解码
1760
经编码残差测试
否
是
1770
确定亮度系数组大小
1780
对亮度 TB 进行解码
1790
确定色度系数组大小
17100
对色度 TB 进行解码
结束

图17