(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 115086681 B
(45)授权公告日 2023.10.13

(21)申请号 202210734580.1

(22)申请日 2020.10.12

(65)同一申请的已公布的文献号
申请公布号 CN 115086681 A

(43)申请公布日 2022.09.20

(30)优先权数据
62/914282 2019.10.11 US
62/923390 2019.10.18 US

(62)分案原申请数据
202080083162.X 2020.10.12

(73)专利权人 北京达佳互联信息技术有限公司
地址 100085 北京市海淀区上地西路6号1幢1层101D1-7

(72)发明人 修晓宇 陈漪纹 马宗全 朱弘正 王祥林 于冰

(74)专利代理机构 中国专利代理(香港)有限公司 72001
专利代理师 王洪斌 吕传奇

(51)Int.Cl.
H04N 19/60(2014.01)
H04N 19/70(2014.01)
H04N 19/186(2014.01)

(56)对比文件
TW 201709738 A,2017.03.01
US 2015264374 A1,2015.09.17
WO 2019101973 A1,2019.05.31
CN 106105203 A,2016.11.09
US 2012301041 A1,2012.11.29
US 2017318301 A1,2017.11.02
US 2018343471 A1,2018.11.29
US 2015264354 A1,2015.09.17
US 2018262763 A1,2018.09.13
CN 101335897 A,2008.12.31
CN 106105202 A,2016.11.09
US 2017118473 A1,2017.04.27
US 2015358645 A1,2015.12.10
US 2017201769 A1,2017.07.13
US 2012213434 A1,2012.08.23
Xiaoyu Xiu.Support of adaptive color transform for 444 video coding in VVC.《Joint Video Experts Team (JVET)of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 16th Meeting: Geneva, CH, 1-11 October 2019,Document: JVET-P0517》.2019,全文.(续)

审查员 杜乾敏

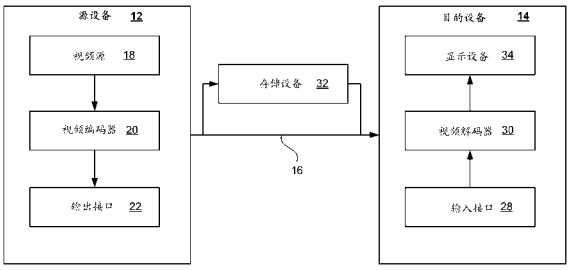
权利要求书1页 说明书32页 附图15页

(54)发明名称

4:4:4色度格式的视频编解码的方法和装置

(57)摘要

一种电子装置执行解码视频数据的方法,包括:从比特流接收与编码单元相对应的视频数据,编码单元是通过帧内预测模式编码的;从视频数据接收第一语法元素,其中第一语法元素指示是否已经使用自适应颜色空间变换ACT对编码单元进行编码;根据第一语法元素具有零值的确定:从视频数据接收一个或多个语法元素,其中一个或多个语法元素指示是否已经使用块差分脉冲编码调制BDPCM对编码单元的色度分量进行编码;根据第一语法元素具有非零值的确定:将默认值指派给与BDPCM相关联的一个或多个语法元素;根据与ACT相关联的第一语法元素和与BDPCM相关联的一个或多个语法元素从视频数据解码编码单元。



CN 115086681 B

[转续页]

[接上页]

(56)对比文件

Gordon Clare.CE8-4.1: BDPCM and Transform skip for Chroma.《Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11,16th Meeting: Geneva, CH, 1-11 October 2019, Document: JVET-P0059-v1》.2019,全文.

1.一种解码视频数据的方法,包括:

从比特流接收与编码单元相对应的视频数据,其中所述编码单元是通过帧内预测模式编码的;

从所述比特流接收第一语法元素,其中所述第一语法元素指示是否已经使用自适应颜色空间变换ACT对所述编码单元进行编码;

根据所述第一语法元素具有零值的确定,其中所述第一语法元素具有零值指示未使用ACT对所述编码单元进行编码:

从所述比特流接收一个或多个语法元素,其中所述一个或多个语法元素指示是否已经使用块差分脉冲编码调制BDPCM对所述编码单元的色度分量进行编码;

根据所述第一语法元素具有非零值的确定,其中所述第一语法元素具有非零值指示已经使用ACT对所述编码单元进行编码:

将默认值指派给与所述BDPCM相关联的所述一个或多个语法元素;

根据与所述ACT相关联的所述第一语法元素和与所述BDPCM相关联的所述一个或多个语法元素从所述视频数据解码所述编码单元的色度分量。

2.根据权利要求1所述的方法,进一步包括:

在接收所述第一语法元素之前,从所述比特流接收第二语法元素,其中所述第二语法元素指示所述视频数据是否具有预定义色度格式。

3.根据权利要求2所述的方法,其中所述预定义色度格式是4:4:4色度格式。

4.根据权利要求2所述的方法,其中仅当所述视频数据具有所述预定义色度格式时,所述第一语法元素才存在于所述比特流中。

5.根据权利要求1所述的方法,其中所述默认值指示在不使用逆BDPCM的情况下对所述编码单元的色度分量进行解码。

6.根据权利要求1所述的方法,其中所述默认值指示使用逆BDPCM对所述编码单元的色度分量进行解码。

7.根据权利要求1所述的方法,其中当所述编码单元是在不应用变换的情况下被编码时,使用逆BDPCM对所述编码单元进行解码。

8.一种电子装置,包括:

一个或多个处理单元;

耦合到所述一个或多个处理单元的存储器;以及

存储在所述存储器中的多个程序,所述多个程序当由所述一个或多个处理单元执行时使得所述电子装置执行根据权利要求1-7中任一项所述的方法。

9.一种非暂时性计算机可读存储介质,存储有多个程序以供具有一个或多个处理单元的电子装置来执行,其中所述多个程序当由所述一个或多个处理单元执行时使得所述电子装置执行根据权利要求1-7中任一项所述的方法。

10.一种非暂时性计算机可读存储介质,存储有多个程序以供具有一个或多个处理单元的电子装置来执行,其中所述多个程序当由所述一个或多个处理单元执行时使得所述电子装置执行根据权利要求1-7中任一项所述的方法;

其中所述存储介质存储所述电子装置执行根据权利要求1-7中任一项方法所解码的比特流。

4:4:4色度格式的视频编解码的方法和装置

[0001] 本申请是分案申请,其母案申请的发明名称是“4:4:4色度格式的视频编解码的方法和装置”,其母案申请的国际申请日是2020年10月12日,其母案申请的申请号是:202080083162.X,其母案申请的国际申请号是:PCT/US2020/055264。

[0002] 相关申请

[0003] 本申请要求2019年10月11日提交的题为“Methods and Apparatus of video CODING in 4:4:4 chroma format”的美国临时专利申请第62/914,282号的优先权,并且还要求2019年10月18日提交的题为“Methods and Apparatus of video CODING in 4:4:4 chroma format”的美国临时专利申请第62/923,390号的优先权,这两个专利申请的全部内容通过引用并入本文。

技术领域

[0004] 本申请总体上涉及视频数据编解码和压缩,并且特别地涉及利用色度残差缩放执行自适应颜色空间变换ACT的方法和系统。

背景技术

[0005] 数字视频由各种电子设备来支持,这些电子设备例如数字电视、膝上型或台式计算机、平板计算机、数字相机、数字记录设备、数字媒体播放器、视频游戏控制台、智能电话、视频电话会议设备、视频流送设备等。电子设备通过实现由MPEG-4、ITU-TH.263、ITU-TH.264/MPEG-4Part 10、高级视频编解码AVC、高效视频编解码HEVC和通用视频编解码VVC标准定义的视频压缩/解压缩标准传输、接收、编码、解码和/或存储数字视频数据。视频压缩通常包括执行空间(帧内)预测和/或时间(帧间)预测,以减少或去除视频数据中固有的冗余。对于基于块的视频编码,视频帧被分区(partition)成一个或多个切片,每个切片具有多个视频块,这些视频块也可以被称为编码树单元CTU。每个CTU可以包含一个编码单元CU,或者递归地被分割(split)成更小的CU,直到达到预定义的最小CU大小。每个CU(也称为叶CU)包含一个或多个变换单元TU,并且每个CU还包含一个或多个预测单元PU。每个CU可以以帧内、帧间或IBC模式编码。视频帧的帧内编码(I)切片中的视频块使用关于同一视频帧内的相邻块中的参考样本的空间预测而编码。视频帧的帧间编码(P或B)切片中的视频块可以使用关于同一视频帧内的相邻块中的参考样本的空间预测、或关于其他先前和/或未来参考视频帧中的参考样本的时间预测。

[0006] 基于先前已经编码的参考块(例如,相邻块)的空间或时间预测得到了针对待编码的当前视频块的预测性块。找到参考块的过程可以通过块匹配算法完成。表示待编码的当前块与预测性块之间的像素差异的残差数据被称为残差块或预测误差。经帧间编码的块是根据指向形成预测性块的参考帧中的参考块以及残差块的运动向量编码的。确定运动向量的过程通常被称为运动估计。经帧内编码的块是根据帧内预测模式和残差块编码的。为了进一步压缩,将残差块从像素域变换到变换域,例如频域,从而得到残差变换系数,所述系数然后可以被量化。初始地以二维数组布置的量化变换系数可以被扫描以产生变换系数的

一维向量,并且然后被熵编码成视频比特流以实现更大的压缩。

[0007] 然后,经编码的视频比特流被保存在计算机可读存储介质(例如,闪速存储器)中,以便由具有数字视频能力的另一个电子设备访问,或者被有线或无线地直接传输到电子设备。电子设备然后通过例如解析经编码的视频比特流以从所述比特流获得语法元素、并且至少部分地基于从所述比特流获得的语法元素从经编码的视频比特流将数字视频数据重建成其原始格式,来执行视频解压缩(这是与上述视频压缩相反的过程),并且在电子设备的显示器上呈现重建的数字视频数据。

[0008] 随着数字视频质量从高清晰度到4Kx2K或甚至8Kx4K,待编码/解码的视频数据的量呈指数级增长。就如何在维持经解码的视频数据的图像质量的同时更高效地编码/解码视频数据而言,这是持续的挑战。

[0009] 某些视频内容(例如,屏幕内容视频)以4:4:4色度格式编码,其中所有三个分量(亮度分量和两个色度分量)具有相同的分辨率。虽然与4:2:0色度格式和4:2:2色度格式相比,4:4:4色度格式包括更多的冗余(这对于实现良好的压缩效率是不友好的),但是4:4:4色度格式对于如下许多应用仍然是优选的编码格式,在这些应用中,需要高保真度来保留经解码的视频中的颜色信息,诸如锐利边缘。在给定4:4:4色度格式视频中存在的冗余的情况下,存在如下证据:即,通过利用4:4:4视频的三个颜色分量(例如,YCbCr域中的Y、Cb和Cr;或RGB域中的G、B和R)之间的相关性,可以实现显著的编码改进。由于这些相关性,在HEVC屏幕内容编码SCC扩展的开发期间,采用自适应颜色空间变换ACT工具来利用这三个颜色分量之间的相关性。

发明内容

[0010] 本申请描述了与视频数据编码和解码相关的实现方式,更特别地,与利用亮度映射和色度缩放LMCS执行自适应颜色空间变换ACT的系统和方法相关的实现方式。

[0011] 根据本申请的第一方面,一种解码视频数据的方法包括:从比特流接收切片的切片头部中的第一语法元素,所述第一语法元素指示具有色度缩放的亮度映射LMCS是否被应用于所述切片中的编码单元;接收针对所述编码单元的第二语法元素,所述第二语法元素指示是否已经使用自适应颜色空间变换ACT对所述编码单元进行编码;如果所述第二语法元素具有非零值,则通过应用逆ACT以将所述编码单元的亮度残差和色度残差从视频数据的经变换的颜色空间转换到原始颜色空间解码所述编码单元;以及在执行逆ACT之后,如果所述第一语法元素具有非零值,则通过对所述编码单元的亮度样本执行逆亮度映射以及对所述编码单元的色度残差执行逆缩放解码所述编码单元。

[0012] 根据本申请的第二方面,一种解码视频数据的方法包括:从比特流接收与编码单元相对应的视频数据,其中所述编码单元是通过帧内预测模式编码的;从所述视频数据接收第一语法元素,其中所述第一语法元素指示是否已经使用自适应颜色空间变换ACT对所述编码单元进行编码;如果所述第一语法元素具有零值,则从所述视频数据接收一个或多个语法元素,其中所述一个或多个语法元素指示是否已经使用块差分脉冲编码调制BDPCM对所述编码单元的色度分量进行了编码;如果所述第一语法元素具有非零值,则将默认值指派给与所述BDPCM相关联的所述一个或多个语法元素;根据与所述ACT相关联的所述第一语法元素和与所述BDPCM相关联的所述一个或多个语法元素从所述视频数据解码所述编码

单元。

[0013] 根据本申请的第三方面，一种电子装置包括一个或多个处理单元、存储器、以及存储在所述存储器中的多个程序。所述程序当由所述一个或多个处理单元执行时使得所述电子装置执行如上所描述的解码视频数据的方法。

[0014] 根据本申请的第四方面，一种非暂时性计算机可读存储介质，存储有多个程序以供具有一个或多个处理单元的电子装置来执行。所述程序当由所述一个或多个处理单元执行时使得所述电子装置执行如上所描述的解码视频数据的方法。

附图说明

[0015] 被包括以提供对实现方式的进一步理解并且被并入本文中并构成说明书的一部分的附图图示了所描述的实现方式，并且与本描述一起用于解释基本原理。相似的参考数字指代对应的部分。

[0016] 图1是图示了根据本公开的一些实现方式的示例性视频编码和解码系统的框图。

[0017] 图2是图示了根据本公开的一些实现方式的示例性视频编码器的框图。

[0018] 图3是图示了根据本公开的一些实现方式的示例性视频解码器的框图。

[0019] 图4A到4E是图示了根据本公开的一些实现方式如何将帧递归地分区成不同大小和形状的多个视频块的框图。

[0020] 图5A和5B是图示了根据本公开的一些实现方式的应用自适应颜色空间变换ACT技术来变换RGB颜色空间与YCgCo颜色空间之间的残差的示例的框图。

[0021] 图6是根据本公开的一些实现方式的在示例性视频数据解码过程中应用具有色度缩放的亮度映射LMCS技术的框图。

[0022] 图7是图示了根据本公开的一些实现方式的示例性视频解码过程的框图，通过所述过程，视频解码器实现逆自适应颜色空间变换ACT技术。

[0023] 图8A和8B是图示了根据本公开的一些实现方式的示例性视频解码过程的框图，通过所述视频解码过程，视频解码器实现逆自适应颜色空间变换ACT和具有色度缩放的亮度映射LMCS技术。

[0024] 图9是图示了根据本公开的一些实现方式的在执行自适应颜色空间变换ACT和块差分脉冲编码调制BDPCM之间的示例性解码逻辑的框图。

[0025] 图10是图示了根据本公开的一些实现方式的示例性过程的流程图，通过所述过程，视频解码器通过执行具有色度缩放的逆自适应颜色空间变换来解码视频数据。

具体实施方式

[0026] 现在将详细地参考具体实现方式，其示例在附图中图示。在以下详细描述中，阐述了许多非限制性的具体细节，以便帮助理解本文中呈现的主题。但是对于本领域的普通技术人员来说将明显的是，在不脱离权利要求的范围的情况下，可以使用各种替代方案，并且可以在没有这些具体细节的情况下实践本主题。例如，对于本领域的普通技术人员来说将明显的是，本文中呈现的主题可以在具有数字视频能力的许多类型的电子设备上实现。

[0027] 图1是图示了根据本公开的一些实现方式的用于并行地编码和解码视频块的示例性系统10的框图。如图1中所示，系统10包含源设备12，源设备12生成并编码视频数据以供

目的设备14稍后解码。源设备12和目的设备14可以包括多种电子设备中的任一个，包括台式或膝上型计算机、平板计算机、智能电话、机顶盒、数字电视、相机、显示设备、数字媒体播放器、视频游戏控制台、视频流送设备等。在一些实现方式中，源设备12和目的设备14被配备有无线通信能力。

[0028] 在一些实现方式中，目的设备14可以经由链路16接收待解码的经编码的视频数据。链路16可以包括能够将经编码的视频数据从源设备12移动到目的设备14的任何类型的通信介质或设备。在一个示例中，链路16可以包括通信介质，以使得源设备12能够实时地将经编码的视频数据直接传输到目的设备14。可以根据通信标准(诸如，无线通信协议)调制经编码的视频数据，并且将其传输到目的设备14。通信介质可以包括任何无线或有线通信介质，诸如射频RF频谱、或一个或多个物理传输线。通信介质可以形成基于分组的网络(诸如，局域网、广域网、或诸如互联网之类的全球网络)的一部分。通信介质可以包括路由器、交换机、基站、或可以用于便于从源设备12到目的设备14的通信的任何其他装备。

[0029] 在一些其他实现方式中，经编码的视频数据可以从输出接口22传输到存储设备32。随后，目的设备14可以经由输入接口28访问存储设备32中的经编码的视频数据。存储设备32可以包括多种分布式或本地访问的数据存储介质中的任一个，诸如硬盘驱动器、蓝光盘、DVD、CD-ROM、闪速存储器、易失性或非易失性存储器、或用于存储经编码的视频数据的任何其他合适的数字存储介质。在另外的示例中，存储设备32可以对应于可保存由源设备12生成的经编码的视频数据的文件服务器或另一个中间存储设备。目的设备14可以经由流送或下载来从存储设备32访问所存储的视频数据。文件服务器可以是能够存储经编码的视频数据并将经编码的视频数据传输到目的设备14的任何类型的计算机。示例性文件服务器包括web服务器(例如，用于网站)、FTP服务器、网络附加存储(NAS)设备或本地盘驱动器。目的设备14可以通过适合于访问存储在文件服务器上的经编码的视频数据的任何标准数据连接访问经编码的视频数据，所述标准数据连接包括无线信道(例如，Wi-Fi连接)、有线连接(例如，DSL、线缆调制解调器等)或两者的组合。来自存储设备32的经编码的视频数据的传输可以是流送传输、下载传输或两者的组合。

[0030] 如图1中所示，源设备12包括视频源18、视频编码器20和输出接口22。视频源18可以包括诸如视频捕获设备之类的源，例如视频相机、包含先前捕获的视频的视频存档、从视频内容提供商接收视频的视频馈送接口、和/或用于生成计算机图形数据作为源视频的计算机图形系统、或这样的源的组合。作为一个示例，如果视频源18是安全监控系统的视频相机，则源设备12和目的设备14可以形成相机电话或视频电话。然而，在本申请中描述的实现方式通常可以适用于视频编解码，并且可以适用于无线和/或有线应用。

[0031] 所捕获的、预捕获的或计算机生成的视频可以由视频编码器20编码。经编码的视频数据可以经由源设备12的输出接口22直接传输到目的设备14。经编码的视频数据还(或替代地)可以被存储在存储设备32上，以供目的设备14或其他设备稍后访问，以用于解码和/或回放。输出接口22可以进一步包括调制解调器和/或发射器。

[0032] 目的设备14包括输入接口28、视频解码器30和显示设备34。输入接口28可以包括接收器和/或调制解调器，并且通过链路16接收经编码的视频数据。通过链路16传送或者在存储设备32上提供的经编码的视频数据可以包括由视频编码器20生成的多种语法元素，以供视频解码器30在解码视频数据时使用。这种语法元素可以被包括在通信介质上传输的经

编码的视频数据内、被存储在存储介质上或者被存储在文件服务器中。

[0033] 在一些实现方式中，目的设备14可以包括显示设备34，显示设备34可以是被配置成与目的设备14通信的集成显示设备和外部显示设备。显示设备34向用户显示经解码的视频数据，并且可以包括多种显示设备中的任一个，诸如液晶显示器LCD、等离子显示器、有机发光二极管OLED显示器或另一种类型的显示设备。

[0034] 视频编码器20和视频解码器30可以根据专有或行业标准操作，这些标准诸如VVC、HEVC、MPEG-4Part 10、高级视频编解码AVC或这样的标准的扩展。应当理解的是，本申请不限于特定的视频编码/解码标准，并且可以适用于其他视频编码/解码标准。通常预期的是，源设备12的视频编码器20可以被配置成根据这些当前或未来标准中的任一个编码视频数据。类似地，通常还预期的是，目的设备14的视频解码器30可以被配置成根据这些当前或未来标准中的任一个解码视频数据。

[0035] 视频编码器20和视频解码器30均可以实现为多种合适的编码器电路中的任一个，诸如一个或多个微处理器、数字信号处理器DSP、专用集成电路ASIC、现场可编程门阵列FPGA、离散逻辑、软件、硬件、固件或其任何组合。当部分地在软件中实现时，电子设备可以将用于软件的指令存储在合适的非暂时性计算机可读介质中，并且使用一个或多个处理器在硬件中执行所述指令，以执行本公开中公开的视频编码/解码操作。视频编码器20和视频解码器30中的每一个可以被包括在一个或多个编码器或解码器中，所述编码器或解码器中的任一者可以集成为相应设备中的组合编码器/解码器CODEC的一部分。

[0036] 图2是图示了根据本申请中描述的一些实现方式的示例性视频编码器20的框图。视频编码器20可以对视频帧内的视频块执行帧内和帧间预测性编码。帧内预测性编码依赖于空间预测减少或去除给定视频帧或图片内的视频数据中的空间冗余。帧间预测性编码依赖于时间预测减少或去除视频序列的邻近视频帧或图片内的视频数据中的时间冗余。

[0037] 如图2中所示，视频编码器20包括视频数据存储器40、预测处理单元41、解码图片缓冲器DPB64、加法器50、变换处理单元52、量化单元54和熵编码单元56。预测处理单元41进一步包括运动估计单元42、运动补偿单元44、分区单元45、帧内预测处理单元46和帧内块复制BC单元48。在一些实现方式中，视频编码器20还包括逆量化单元58、逆变换处理单元60和加法器62以用于视频块重建。去块(deblocking)滤波器(未示出)可以定位在加法器62和DPB 64之间，以对块边界进行滤波，从而从重建的视频中去除块效应伪像(blockiness artifact)。除了去块滤波器之外，还可以使用环路内滤波器(未示出)对加法器62的输出进行滤波。视频编码器20可以采取固定或可编程硬件单元的形式，或者可以在一个或多个说明的固定或可编程硬件单元之间被划分。

[0038] 视频数据存储器40可以存储将由视频编码器20的组件编码的视频数据。视频数据存储器40中的视频数据可以例如从视频源18获得。DPB 64是存储用于在由视频编码器20(例如，在帧内或帧间预测性编码模式中)编码视频数据时使用的参考视频数据的缓冲器。视频数据存储器40和DPB 64可以由多种存储器设备中的任一个形成。在各种示例中，视频数据存储器40可以与视频编码器20的其他组件一起在芯片上，或相对于那些组件在芯片外。

[0039] 如图2中所示，在接收到视频数据之后，预测处理单元41内的分区单元45将视频数据分区成视频块。所述分区还可以包括根据预定义的分割结构(诸如，与视频数据相关联的

四叉树结构)将视频帧分区成切片、图块(tile)或其他较大的编码单元CU。视频帧可以被划分成多个视频块(或被称为图块的视频块集合)。预测处理单元41可以基于误差结果(例如,编码率和失真水平)针对当前视频块选择多个可能的预测性编码模式中的一个,诸如多个帧内预测性编码模式中的一个或多个帧间预测性编码模式中的一个。预测处理单元41可以将所得的帧内或帧间预测编码块提供给加法器50以生成残差块,并且提供给加法器62以重建经编码的块以在随后用作参考帧的一部分。预测处理单元41还向熵编码单元56提供语法元素,诸如运动向量、帧内模式指示符、分区信息和其他这种语法信息。

[0040] 为了针对当前视频块选择合适的帧内预测性编码模式,预测处理单元41内的帧内预测处理单元46可以相对于与待编码的当前块在同一帧中的一个或多个相邻块对当前视频块执行帧内预测性编码,以提供空间预测。预测处理单元41内的运动估计单元42和运动补偿单元44相对于一个或多个参考帧中的一个或多个预测性块对当前视频块执行帧间预测性编码,以提供时间预测。视频编码器20可以将编码执行多遍,例如针对视频数据的每个块选择适当的编码模式。

[0041] 在一些实现方式中,运动估计单元42通过根据视频帧序列内的预定模式生成运动向量来确定用于当前视频帧的帧间预测模式,所述运动向量指示当前视频帧内的视频块的预测单元PU相对于参考视频帧内的预测性块的位移。由运动估计单元42执行的运动估计是生成运动向量的过程,所述运动向量估计视频块的运动。例如,运动向量可以指示当前视频帧或图片内的视频块的PU相对于参考帧(或其他编码单元)内的预测性块、相对于当前帧(或其他编码单元)内正被编码的当前块的位移。预定模式可以将序列中的视频帧指定为P帧或B帧。帧内BC单元48可以以与运动估计单元42针对帧间预测确定运动向量类似的方式来确定用于帧内BC编码的向量(例如,块向量),或者可以利用运动估计单元42来确定块向量。

[0042] 预测性块是参考帧的块,所述块被视为在像素差异方面与待编码的视频块的PU紧密地匹配,所述像素差异可以由绝对差异的总和SAD、平方差异的总和SSD或其他差异度量来确定。在一些实现方式中,视频编码器20可以计算存储在DPB 64中的参考帧的亚整数像素位置的值。例如,视频编码器20可以内插参考帧的四分之一像素位置、八分之一像素位置或其他分数像素位置的值。因此,运动估计单元42可以相对于全像素位置和分数像素位置执行运动搜索,并且输出具有分数像素精度的运动向量。

[0043] 运动估计单元42通过将PU的位置与从第一参考帧列表(列表0)或第二参考帧列表(列表1)中选择的参考帧的预测性块的位置进行比较来计算帧间预测编码帧中的视频块的PU的运动向量,第一参考帧列表或第二参考帧列表中的每一个标识存储在DPB 64中的一个或多个参考帧。运动估计单元42将计算的运动向量发送到运动补偿单元44,并且然后发送到熵编码单元56。

[0044] 由运动补偿单元44执行的运动补偿可以涉及基于由运动估计单元42确定的运动向量来获取或生成预测性块。在接收到当前视频块的PU的运动向量时,运动补偿单元44可以在参考帧列表中的一个中定位所述运动向量指向的预测性块,从DPB 64中检索预测性块,并且将预测性块转发到加法器50。加法器50然后通过从正被编码的当前视频块的像素值中减去由运动补偿单元44提供的预测性块的像素值来形成像素差异值的残差视频块。形成了残差视频块的像素差异值可以包括亮度或色度差异分量或其两者。运动补偿单元44还

可以生成与视频帧的视频块相关联的语法元素，以供视频解码器30在解码视频帧的视频块时使用。语法元素可以包括(例如)定义了用于预测性块的运动向量的语法元素、指示预测模式的任何标志、或本文中描述的任何其他语法信息。要注意的是，运动估计单元42和运动补偿单元44可以是高度集成的，但出于概念目的而分离地图示。

[0045] 在一些实现方式中，帧内BC单元48可以与上面结合运动估计单元42和运动补偿单元44描述的方式类似的方式来生成向量并且获取预测性块，但是预测性块与正被编码的当前块在同一帧中，并且其中向量被称为块向量，而不是运动向量。特别地，帧内BC单元48可以确定用于编码当前块的帧内预测模式。在一些示例中，帧内BC单元48可以使用各种帧内预测模式、例如在单独的编码过程期间编码当前块，并且通过速率-失真分析测试其性能。接下来，帧内BC单元48可以在各种经测试的帧内预测模式当中选择适当的帧内预测模式来使用，并且相应地生成帧内模式指示符。例如，帧内BC单元48可以使用针对各种经测试的帧内预测模式的速率-失真分析来计算速率-失真值，并且在经测试的模式当中选择具有最佳速率-失真特性的帧内预测模式作为适当的帧内预测模式来使用。速率-失真分析通常确定经编码的块与原始未编码的块——其被编码以产生经编码的块——之间的失真(或误差)量、以及用于产生经编码的块的比特速率(即，比特数量)。帧内BC单元48可以根据各种经编码的块的失真和速率来计算比率，以确定哪种帧内预测模式展现出针对所述块的最佳速率-失真值。

[0046] 在其他示例中，帧内BC单元48可以全部地或部分地使用运动估计单元42和运动补偿单元44根据本文中描述的实现方式来执行用于帧内BC预测的这样的功能。在任一种情况下，对于帧内块复制，预测性块可以是被视为在像素差异方面与待编码的块紧密地匹配的块，所述像素差异可以由绝对差异的总和SAD、平方差异的总和SSD或其他差异度量来确定，并且预测性块的标识可以包括亚整数像素位置的值的计算。

[0047] 无论预测性块是来自根据帧内预测的同一帧还是来自根据帧间预测的不同帧，视频编码器20都可以通过从正被编码的当前视频块的像素值中减去预测性块的像素值来形成残差视频块，从而形成像素差异值。形成了残差视频块的像素差异值可以包括亮度和色度分量差异两者。

[0048] 如上所描述，帧内预测处理单元46可以对当前视频块进行帧内预测，作为由运动估计单元42和运动补偿单元44执行的帧间预测、或由帧内BC单元48执行的帧内块复制预测的替代方案。特别地，帧内预测处理单元46可以确定用于编码当前块的帧内预测模式。为此，帧内预测处理单元46可以使用各种帧内预测模式、例如在单独的编码过程期间编码当前块，并且帧内预测处理单元46(或在一些示例中为模式选择单元)可以从经测试的帧内预测模式中选择适当的帧内预测模式来使用。帧内预测处理单元46可以向熵编码单元56提供指示所述块的所选帧内预测模式的信息。熵编码单元56可以编码比特流中指示所选帧内预测模式的信息。

[0049] 在预测处理单元41经由帧间预测或帧内预测确定了当前视频块的预测性块之后，加法器50通过从当前视频块中减去预测性块来形成残差视频块。残差块中的残差视频数据可以被包括在一个或多个变换单元TU中，并且被提供到变换处理单元52。变换处理单元52使用变换(诸如，离散余弦变换DCT或概念上类似的变换)将残差视频数据变换成残差变换系数。

[0050] 变换处理单元52可以将所得的变换系数发送到量化单元54。量化单元54量化所述变换系数以进一步降低比特速率。量化过程还可以减小与一些或所有系数相关联的比特深度。可以通过调整量化参数修改量化的程度。在一些示例中，然后，量化单元54可以对包括经量化的变换系数的矩阵执行扫描。替代地，熵编码单元56可以执行所述扫描。

[0051] 在量化之后，熵编码单元56使用(例如)上下文自适应可变长度编码CAVLC、上下文自适应二进制算术编码CABAC、基于语法的上下文自适应二进制算术编码SBAC、概率区间分区熵PIPE编码或另一个熵编码方法或技术将经量化的变换系数熵编码成视频比特流。然后，经编码的比特流可以被传输到视频解码器30、或者存档在存储设备32中以供稍后传输到视频解码器30或由视频解码器30来检索。熵编码单元56还可以对正被编码的当前视频帧的运动向量和其他语法元素进行熵编码。

[0052] 逆量化单元58和逆变换处理单元60分别应用逆量化和逆变换以在像素域中重建残差视频块，以便生成用于预测其他视频块的参考块。如上所指出，运动补偿单元44可以从存储在DPB 64中的帧的一个或多个参考块来生成运动补偿预测性块。运动补偿单元44还可以将一个或多个内插滤波器应用于预测性块，以计算用于运动估计的亚整数像素值。

[0053] 加法器62将经重建的残差块加到由运动补偿单元44产生的经运动补偿的预测性块，以产生用于存储在DPB 64中的参考块。然后，参考块可以由帧内BC单元48、运动估计单元42和运动补偿单元44用作预测性块，以对随后视频帧中的另一个视频块进行帧间预测。

[0054] 图3是图示了根据本申请的一些实现方式的示例性视频解码器30的框图。视频解码器30包括视频数据存储器79、熵解码单元80、预测处理单元81、逆量化单元86、逆变换处理单元88、加法器90和DPB 92。预测处理单元81进一步包括运动补偿单元82、帧内预测单元84和帧内BC单元85。视频解码器30可以执行通常与上面结合图2关于视频编码器20描述的编码过程互逆的解码过程。例如，运动补偿单元82可以基于从熵解码单元80接收到的运动向量来生成预测数据，而帧内预测单元84可以基于从熵解码单元80接收到的帧内预测模式指示符来生成预测数据。

[0055] 在一些示例中，视频解码器30的单元的任务可以是执行本申请的实现方式。此外，在一些示例中，本公开的实现方式可以在视频解码器30的一个或多个单元之间被划分。例如，帧内BC单元85可以单独地或结合视频解码器30的其他单元(例如，运动补偿单元82、帧内预测单元84和熵解码单元80)来执行本申请的实现方式。在一些示例中，视频解码器30可以不包括帧内BC单元85，并且帧内BC单元85的功能可以由预测处理单元81的其他组件(诸如，运动补偿单元82)来执行。

[0056] 视频数据存储器79可以存储将由视频解码器30的其他组件解码的视频数据，诸如经编码的视频比特流。存储在视频数据存储器79中的视频数据可以例如从存储设备32、从诸如相机之类的本地视频源、经由视频数据的有线或无线网络通信、或者通过访问物理数据存储介质(例如，闪存驱动器或硬盘)来获得。视频数据存储器79可以包括编码图片缓冲器CPB，所述编码图片缓冲器CPB存储来自经编码的视频比特流的经编码的视频数据。视频解码器30的解码图片缓冲器DPB92存储参考视频数据，以供视频解码器30(例如，在帧内或帧间预测性编码模式中)在解码视频数据时使用。视频数据存储器79和DPB 92可以由多种存储器设备中的任一个形成，诸如动态随机存取存储器DRAM，包括同步DRAM～SDRAM、磁阻式RAM～MRAM、电阻式RAM～RRAM或其他类型的存储器设备。出于说明的目的，视频数据存储

器79和DPB 92在图3中被描绘为视频解码器30的两个不同的组件。但是对于本领域技术人员来说将明显的是，视频数据存储器79和DPB 92可以由相同的存储器设备或分离的存储器设备来提供。在一些示例中，视频数据存储器79可以与视频解码器30的其他组件一起在芯片上，或者相对于那些组件在芯片外。

[0057] 在解码过程期间，视频解码器30接收经编码的视频比特流，所述经编码的视频比特流表示经编码的视频帧的视频块以及相关联的语法元素。视频解码器30可以接收视频帧级别和/或视频块级别下的语法元素。视频解码器30的熵解码单元80对比特流进行熵解码以生成经量化的系数、运动向量或帧内预测模式指示符以及其他语法元素。然后，熵解码单元80将运动向量和其他语法元素转发到预测处理单元81。

[0058] 当视频帧被编码为帧内预测编码(I)帧或针对其他类型帧中的帧内编码预测性块被编码时，预测处理单元81的帧内预测单元84可以基于发信令通知的帧内预测模式和来自当前帧的先前解码的块的参考数据来生成当前视频帧的视频块的预测数据。

[0059] 当视频帧被编码为帧间预测编码(即，B或P)帧时，预测处理单元81的运动补偿单元82基于从熵解码单元80接收到的运动向量和其他语法元素来产生当前视频帧的视频块的一个或多个预测性块。每个预测性块可以从参考帧列表中的一个内的参考帧来产生。视频解码器30可以基于存储在DPB 92中的参考帧使用默认构造技术来构造参考帧列表，列表0和列表1。

[0060] 在一些示例中，当根据本文中描述的帧内BC模式对视频块进行编码时，预测处理单元81的帧内BC单元85基于从熵解码单元80接收到的块向量和其他语法元素来产生当前视频块的预测性块。预测性块可以在与由视频编码器20定义的当前视频块相同的图片的重建区域内。

[0061] 运动补偿单元82和/或帧内BC单元85通过解析运动向量和其他语法元素来确定当前视频帧的视频块的预测信息，并且然后使用所述预测信息来产生正被解码的当前视频块的预测性块。例如，运动补偿单元82使用接收到的语法元素中的一些来确定用于编码视频帧的视频块的预测模式(例如，帧内或帧间预测)、帧间预测帧类型(例如，B或P)、针对所述帧的参考帧列表中的一个或多个的构造信息、所述帧的每一个帧间预测编码视频块的运动向量、所述帧的每一个帧间预测编码视频块的帧间预测状态、以及用于解码当前视频帧中的视频块的其他信息。

[0062] 类似地，帧内BC单元85可以使用接收到的语法元素中的一些(例如，标志)来确定当前视频块是使用帧内BC模式来预测的、所述帧的哪些视频块在重建区域内并且应当被存储在DPB 92中的构造信息、所述帧的每一个帧内BC预测视频块的块向量、所述帧的每一个帧内BC预测视频块的帧内BC预测状态、以及用于解码当前视频帧中的视频块的其他信息。

[0063] 运动补偿单元82还可以使用如视频编码器20在视频块的编码期间所使用的内插滤波器来执行内插，以计算参考块的亚整数像素的内插值。在所述情况下，运动补偿单元82可以根据接收到的语法元素来确定视频编码器20使用的内插滤波器，并且使用所述内插滤波器来产生预测性块。

[0064] 逆量化单元86使用由视频编码器20针对视频帧中的每个视频块计算的、用于确定量化程度的相同量化参数来对比特流中提供的并且由熵解码单元80熵解码的经量化的变换系数进行逆量化。逆变换处理单元88将逆变换(例如，逆DCT、逆整数变换或概念上类似的

逆变换过程)应用于变换系数,以便在像素域中重建残差块。

[0065] 在运动补偿单元82或帧内BC单元85基于向量和其他语法元素生成了当前视频块的预测性块之后,加法器90通过将来自逆变换处理单元88的残差块与由运动补偿单元82和帧内BC单元85生成的对应预测性块进行求和来重建当前视频块的经解码的视频块。环路内滤波器(未在图中绘出)可以定位在加法器90和DPB 92之间,以进一步处理经解码的视频块。然后,给定帧中的经解码的视频块被存储在DPB 92中,所述DPB 92存储用于接下来的视频块的后续运动补偿的参考帧。DPB 92或与DPB 92分离的存储器设备也可以存储经解码的视频以供稍后在显示设备(诸如,图1的显示设备34)上呈现。

[0066] 在典型的视频编解码过程中,视频序列通常包括一组有序的帧或图片。每个帧可以包括三个样本数组,被表示为SL、SCb和SCr。SL是亮度样本的二维数组。SCb是Cb色度样本的二维数组。SCr是Cr色度样本的二维数组。在其他实例中,帧可以是单色的,并且因此仅包括亮度样本的一个二维数组。

[0067] 如图4A中所示,视频编码器20(或更具体地,分区单元45)通过首先将帧分区成一组编码树单元CTU来生成所述帧的经编码的表示。视频帧可以包括整数数量的CTU,这些CTU按照光栅扫描次序从左到右和从上到下连续地定序。每个CTU是最大的逻辑编码单元,并且CTU的宽度和高度由视频编码器20在序列参数集中发信令通知,使得视频序列中的所有CTU具有相同的大小,大小为128×128、64×64、32×32和16×16之一。但是应当注意,本申请不一定限于特定的大小。如图4B中所示,每个CTU可以包括亮度样本的一个编码树块CTB、色度样本的两个对应的编码树块、以及用于对这些编码树块的样本进行编码的语法元素。语法元素描述了经编码的像素块的不同类型的单元的性质、以及如何在视频解码器30处重建所述视频序列,包括帧间或帧内预测、帧内预测模式、运动向量和其他参数。在单色图片或具有三个单独的颜色平面的图片中,CTU可以包括单个编码树块、以及用于对编码树块的样本进行编码的语法元素。编码树块可以是N×N个样本块。

[0068] 为了实现更好的性能,视频编码器20可以递归地对CTU的编码树块执行树分区,例如二叉树分区、三叉树分区、四叉树分区或两者的组合,并且将CTU划分成更小的编码单元CU。如图4C中所描绘的,64×64的CTU 400首先被划分成四个较小的CU,每个CU具有32×32的块大小。在四个较小的CU当中,CU 410和CU 420均按照块大小被划分成四个16×16的CU。两个16×16的CU 430和440均按照块大小被进一步划分成4个8×8的CU。图4D描绘了一种四叉树数据结构,所述结构图示了如图4C中描绘的CTU 400的分区过程的最终结果,四叉树的每个叶节点对应于一个CU,所述CU具有范围从32×32到8×8的相应大小。类似于图4B中描绘的CTU,每个CU可以包括相同大小的帧的亮度样本的编码块CB和色度样本的两个对应的编码块、以及用于对编码块的样本进行编码的语法元素。在单色图片或具有三个单独的颜色平面的图片中,CU可以包括单个编码块、以及用于对编码块的样本进行编码的语法结构。应当注意的是,图4C和4D中描绘的四叉树分区仅仅是为了说明的目的,并且一个CTU可以基于四叉/三叉/二叉树分区被划分成CU以适应变化的局部特性。在多类型树结构中,一个CTU通过四叉树结构来分区,并且每个四叉树的叶CU可以进一步通过二叉和三叉树结构来分区。如图4E中所示,存在五种分区类型,即四元分区、水平二元分区、垂直二元分区、水平三元分区和垂直三元分区。

[0069] 在一些实现方式中,视频编码器20可以进一步将CU的编码块分区成一个或多个

MxN预测块PB。预测块是在其上应用相同预测(帧间或帧内)的矩形(正方形或非正方形)样本块。CU的预测单元PU可以包括亮度样本的预测块、色度样本的两个对应的预测块、以及用于预测所述预测块的语法元素。在单色图片或具有三个单独的颜色平面的图片中,PU可以包括单个预测块、以及用于预测所述预测块的语法结构。视频编码器20可以针对CU的每个PU的亮度、Cb和Cr预测块来生成预测性亮度、Cb和Cr块。

[0070] 视频编码器20可以使用帧内预测或帧间预测来生成PU的预测性块。如果视频编码器20使用帧内预测来生成PU的预测性块,则视频编码器20可以基于与PU相关联的帧的经解码的样本来生成PU的预测性块。如果视频编码器20使用帧间预测来生成PU的预测性块,则视频编码器20可以基于除了与PU相关联的帧之外的一个或多个帧的经解码的样本来生成PU的预测性块。

[0071] 在视频编码器20生成了CU的一个或多个PU的预测性亮度、Cb和Cr块之后,视频编码器20可以通过从CU的原始亮度编码块中减去CU的预测性亮度块来生成CU的亮度残差块,使得CU的亮度残差块中的每一样本指示CU的预测性亮度块之一中的亮度样本与CU的原始亮度编码块中的对应样本之间的差异。类似地,视频编码器20可以分别生成CU的Cb残差块和Cr残差块,使得CU的Cb残差块中的每一个样本指示CU的预测性Cb块之一中的Cb样本与CU的原始Cb编码块中的对应样本之间的差异,并且CU的Cr残差块中的每一个样本可以指示CU的预测性Cr块之一中的Cr样本与CU的原始Cr编码块中的对应样本之间的差异。

[0072] 此外,如图4C中所图示,视频编码器20可以使用四叉树分区来将CU的亮度、Cb和Cr残差块分解成一个或多个亮度、Cb和Cr变换块。变换块是在其上应用相同变换的矩形(正方形或非正方形)样本块。CU的变换单元TU可以包括亮度样本的变换块、色度样本的两个对应的变换块、以及用于对变换块的样本进行变换的语法元素。因此,CU的每个TU可以与亮度变换块、Cb变换块和Cr变换块相关联。在一些示例中,与TU相关联的亮度变换块可以是CU的亮度残差块的子块。Cb变换块可以是CU的Cb残差块的子块。Cr变换块可以是CU的Cr残差块的子块。在单色图片或具有三个单独的颜色平面的图片中,TU可以包括单个变换块、以及用于对所述变换块的样本进行变换的语法结构。

[0073] 视频编码器20可以将一个或多个变换应用于TU的亮度变换块,以生成TU的亮度系数块。系数块可以是变换系数的二维数组。变换系数可以是标量。视频编码器20可以将一个或多个变换应用于TU的Cb变换块,以生成TU的Cb系数块。视频编码器20可以将一个或多个变换应用于TU的Cr变换块,以生成TU的Cr系数块。

[0074] 在生成了系数块(例如,亮度系数块、Cb系数块或Cr系数块)之后,视频编码器20可以量化系数块。量化通常指代如下过程:其中对变换系数进行量化以尽可能减少用于表示变换系数的数据的量,从而提供进一步的压缩。在视频编码器20量化了系数块之后,视频编码器20可以对指示经量化的变换系数的语法元素进行熵编码。例如,视频编码器20可以对指示经量化的变换系数的语法元素执行上下文自适应二进制算术编码CABAC。最后,视频编码器20可以输出包括形成了经编码的帧和相关联数据的表示的比特序列的比特流,所述比特流要么被保存在存储设备32中、要么被传输到目的设备14。

[0075] 在接收到由视频编码器20生成的比特流之后,视频解码器30可以解析所述比特流以从所述比特流获得语法元素。视频解码器30可以至少部分地基于从所述比特流获得的语法元素来重建视频数据的帧。重建视频数据的过程通常与由视频编码器20执行的编码过程

互逆。例如,视频解码器30可以对与当前CU的TU相关联的系数块执行逆变换,以重建与当前CU的TU相关联的残差块。视频解码器30还通过将当前CU的PU的预测性块的样本添加到当前CU的TU的变换块的对应样本来重建当前CU的编码块。在重建了帧的每个CU的编码块之后,视频解码器30可以重建所述帧。

[0076] 如上所指出,视频编解码主要使用两种模式来实现视频压缩,两种模式即帧内的预测(或帧内预测)和帧间的预测(或帧间预测)。基于调色板的编码是已经被许多视频编解码标准采用的另一种编码方案。在可能特别适合于屏幕生成的内容编码的基于调色板的编码中,视频编解码器(例如,视频编码器20或视频解码器30)形成表示给定块的视频数据的颜色的调色板表格。所述调色板表格包括给定块中最主要的(例如,频繁使用的)像素值。在给定块的视频数据中不经常表示的像素值要么不被包括在调色板表格中,要么作为转义颜色(escape color)被包括在调色板表格中。

[0077] 调色板表格中的每个条目包括调色板表格中的对应像素值的索引。所述块中的样本的调色板索引可以被编码以指示来自调色板表格的哪个条目将被用于预测或重建哪个样本。所述调色板模式开始于针对图片、切片、图块或其他这种视频块分组的第一个块来生成调色板预测器(predictor)的过程。如下面将解释的,通常通过更新先前使用的调色板预测器来生成随后视频块的调色板预测器。出于说明的目的,假设调色板预测器是在图片级别定义的。换句话说,图片可以包括多个编码块,每个编码块具有其自己的调色板表格,但是针对整个图片存在一个调色板预测器。

[0078] 为了减少在视频比特流中发信令通知调色板条目需要的比特,视频解码器可以利用调色板预测器以用于确定调色板表格中的用于重建视频块的新调色板条目。例如,调色板预测器可以包括来自先前使用的调色板表格的调色板条目,或者甚至通过包括最近使用的调色板表格的所有条目而利用最近使用的调色板表格来初始化。在一些实现方式中,调色板预测器可以包括少于来自最近使用的调色板表格的所有条目,并且然后并入来自其他先前使用的调色板表格的一些条目。调色板预测器可以具有与用于编码不同块的调色板表格相同的大小,或者可以大于或小于用于编码不同块的调色板表格。在一个示例中,调色板预测器被实现为包括64个调色板条目的先进先出FIFO表格。

[0079] 为了从调色板预测器生成针对视频数据块的调色板表格,视频解码器可以从经编码的视频比特流接收调色板预测器的每个条目的一比特标志。所述一比特标志可以具有指示调色板预测器的相关联条目将被包括在调色板表格中的第一个值(例如,二进制一)、或指示调色板预测器的相关联条目将不被包括在调色板表格中的第二个值(例如,二进制零)。如果调色板预测器的大小大于用于视频数据块的调色板表格,则一旦达到调色板表格的最大大小,视频解码器就可以停止接收更多的标志。

[0080] 在一些实现方式中,调色板表格中的一些条目可以在经编码的视频比特流中被直接发信令通知,而不是使用调色板预测器来被确定。对于这种条目,视频解码器可以从经编码的视频比特流接收三个单独的m比特值,所述m比特值指示与所述条目相关联的亮度和两个色度分量的像素值,其中m表示视频数据的比特深度。与直接发信令通知的调色板条目需要的多个m比特值相比,从调色板预测器导出的那些调色板条目仅需要一比特标志。因此,使用调色板预测器来发信令通知一些或所有调色板条目可以显著减少发信令通知新调色板表格的条目需要的比特数量,从而改进调色板模式编码的总体编码效率。

[0081] 在许多实例中,基于用于编码一个或多个先前编码的块的调色板表格来确定一个块的调色板预测器。但是当对图片、切片或图块中的第一编码树单元进行编码时,先前编码的块的调色板表格可能不可用。因此,不能够使用先前使用的调色板表格的条目来生成调色板预测器。在这种情况下,可以在序列参数集SPS和/或图片参数集PPS中发信令通知调色板预测器初始化器的序列,所述序列参数集SPS和/或图片参数集PPS是当先前使用的调色板表格不可用时用于生成调色板预测器的值。SPS通常指代应用于被称为编码视频序列CVS的一系列连续编码视频图片的语法元素的语法结构,如由在PPS中找到的语法元素的内容确定的那样,所述PPS由在每个切片片段头部中找到的语法元素来指代。PPS通常指代应用于CVS内的一个或多个个体图片的语法元素的语法结构,如由在切片片段头部中找到的语法元素确定的那样。因此,SPS通常被认为是比PPS更高级别的语法结构,这意味着与PPS中包含的语法元素相比,SPS中包含的语法元素通常较不频繁地改变并且应用于视频数据的较大部分。

[0082] 图5A至5B是图示了根据本公开的一些实现方式的应用自适应颜色空间变换ACT技术来变换RGB颜色空间和YCgCo颜色空间之间的残差的示例的框图。

[0083] 在HEVC屏幕内容编码扩展中,应用ACT以自适应地将残差从一个颜色空间(例如,RGB)变换到另一个颜色空间(例如,YCgCo),使得三个颜色分量(例如,R、G和B)之间的相关性(例如,冗余)在YCgCo颜色空间中显著降低。此外,在现有的ACT设计中,通过针对每个TU发信令通知一个标志tu_act_enabled_flag从而在变换单元TU级别执行不同颜色空间的适配。当标志tu_act_enabled_flag等于1时,它指示当前TU的残差是在YCgCo空间中编码的;否则(即,所述标志等于0),它指示当前TU的残差是在原始颜色空间(即,没有颜色空间转换)中编码的。此外,取决于当前TU是以无损模式编码还是以有损模式编码,应用不同的颜色空间变换公式。具体地,在图5A中定义了针对有损模式的RGB颜色空间与YCgCo颜色空间之间的正向和逆向颜色空间变换公式。

[0084] 对于无损模式,使用RGB-YCgCo变换的可逆版本(也被称为YCgCo-LS)。RGB-YCgCo变换的可逆版本是基于图5B中描绘的提升操作和相关描述来实现的。

[0085] 如图5A中所示,在有损模式中使用的正向和逆向颜色变换矩阵没有被归一化。因此,在应用颜色变换之后,YCgCo信号的幅度小于原始信号的幅度。为了补偿由正向颜色变换引起的量值降低,将经调整的量化参数应用于YCgCo域中的残差。具体地,当应用颜色空间变换时,用于量化YCgCo域残差的QP值QP_Y、QP_{Cg}和QP_{Co}分别被设置为QP-5、QP-5和QP-3,其中QP是原始颜色空间中使用的量化参数。

[0086] 图6是根据本公开的一些实现方式的在示例性视频数据解码过程中应用具有色度缩放的亮度映射LMCS技术的框图。

[0087] 在VVC中,LMCS被用作在环路内滤波器(例如,去块滤波器、SAO和ALF)之前应用的新编解码工具。通常,LMCS具有两个主要模块:1)基于自适应逐段线性模型的亮度分量的环路内映射;2)亮度相关的色度残差缩放。图6示出了其中应用了LMCS的经修改的解码过程。在图6中,在映射域中进行的解码模块包括熵解码模块、逆量化模块、逆变换模块、亮度帧内预测模块和亮度样本重建模块(即,亮度预测样本和亮度残差样本的相加)。在原始(即,非映射)域中进行的解码模块包括运动补偿预测模块、色度帧内预测模块、色度样本重建模块(即,色度预测样本和色度残差样本的相加)以及所有的环路内滤波器模块,诸如去块模块、

SAO模块和ALF模块。由LMCS引入的新操作模块包括亮度样本的正向映射模块610、亮度样本的逆向映射模块620和色度残差缩放模块630。

[0088] LMCS的环路内映射可以调整输入信号的动态范围，以改进编码效率。现有LMCS设计中的亮度样本的环路内映射建立在两个映射函数上：一个正向映射函数FwdMap和一个对应的逆向映射函数InvMap。正向映射函数使用具有16个相同大小的段的一个逐段线性模型从编码器被发信令通知到解码器。逆向映射函数可以直接从正向映射函数中导出，并且因此不需要被发信令通知。

[0089] 亮度映射模型的参数在切片级别被发信令通知。首先，发信令通知存在标志，以指示是否要针对当前切片发信令通知亮度映射模型。如果亮度映射模型存在于当前切片中，则进一步发信令通知对应的逐段线性模型参数。此外，在切片级别，发信令通知另一个LMCS控制标志以启用/禁用切片的LMCS。

[0090] 色度残差缩放模块630被设计成当环路内映射被应用于亮度信号时补偿亮度信号与其对应的色度信号之间的量化精度的相互作用。还在切片头部中发信令通知针对当前切片是启用还是禁用色度残差缩放。如果启用亮度映射，则发信令通知附加标志，以指示是否应用亮度相关的色度残留缩放。当不使用亮度映射时，亮度相关的色度残差缩放始终被禁用，并且不需要附加标志。此外，针对包含少于或等于四个色度样本的CU，始终禁用色度残差缩放。

[0091] 图7是图示了根据本公开的一些实现方式的示例性视频解码过程的框图，通过所述解码过程，视频解码器实现逆自适应颜色空间变换ACT的技术。

[0092] 类似于HEVC SCC中的ACT设计，VVC中的ACT将采用4:4:4色度格式的一个CU的帧内/帧间预测残差从原始颜色空间(例如，RGB颜色空间)转换到YCgCo颜色空间中。作为结果，为了更好的编码效率，可以减少三个颜色分量之间的冗余。图7描绘了其中如何通过添加逆ACT模块710在VVC框架中应用逆ACT的解码流程图。当处理在ACT启用的情况下编码的CU时，熵解码、逆量化以及基于DCT/DST的逆变换首先应用于CU。此后，如图7中所描绘，调用逆ACT来将经解码的残差从YCgCo颜色空间转换到原始颜色空间(例如，RGB和YCbCr)。此外，因为有损模式中的ACT没有被归一化，所以将(-5，-5，-3)的QP调整应用于Y、Cg和Co分量，以补偿经变换的残差的改变的量值。

[0093] 在一些实施例中，ACT方法重新使用HEVC的相同ACT核心变换来进行不同颜色空间之间的颜色转换。具体地，取决于当前CU是以有损还是以无损方式编码，应用两种不同版本的颜色变换。针对有损情况的正向和逆向颜色变换使用如图5A中所描绘的不可逆YCgCo变换矩阵。针对无损情况，应用如图5B中所示的可逆颜色变换YCgCo-LS。此外，与现有的ACT设计不同，以下改变被引入到提议的ACT方案，以处理其与VVC标准中的其他编解码工具的相互作用。

[0094] 例如，因为HEVC中的一个CU的残差可以被分区成多个TU，所以针对每个TU分离地发信令通知ACT控制标志，以指示是否需要应用颜色空间转换。然而，如上面结合图4E描述的，在VVC中应用了嵌套有二元和三元分区结构的一个四叉树来代替多分区类型概念，从而去除了HEVC中的分离的CU、PU和TU分区。这意味着，在大多数情况下，一个CU叶节点也用作预测和变换处理的单元，而无需进一步分区，除非最大支持变换大小小于CU的一个分量的宽度或高度。基于这种分区结构，在本公开中提出了在CU级别自适应地启用和禁用ACT。具

体地，针对每个CU发信令通知一个标志cu_act_enabled_flag，以在原始颜色空间与YCgCo颜色空间之间进行选择，以用于对CU的残差进行编码。如果所述标志等于1，则它指示CU内的所有UT的残差是在YCgCo颜色空间中编码的。否则，如果所述标志cu_act_enabled_flag等于0，则CU的所有残差是在原始颜色空间中编码的。

[0095] 在一些实施例中，语法元素(例如，sps_act_enabled_flag)被添加到序列参数集(SPS)以指示ACT是否在序列级别被启用。此外，当颜色空间转换被应用于其亮度和色度分量具有相同分辨率(例如，4:4:4色度格式4:4:4)的视频内容时，需要添加一个比特流一致性要求，使得ACT仅能够针对4:4:4色度格式被启用。表1说明了具有添加的上述语法的经修改的SPS语法表。

[0096]

seq_parameter_set_rbsp( ) {	描述符
...	
sps_act_enabled_flag	u(1)
...	
}	

[0097] 表1经修改的SPS语法表。

[0098] 具体地，sps_act_enabled_flag等于1指示ACT被启用，并且sps_act_enabled_flag等于0指示ACT被禁用，从而对于CU，不会发信令通知所述标志cu_act_enabled_flag，它指代SPS但是被推断为0。当ChromaArrayType不等于3时，比特流一致性的要求是sps_act_enabled_flag的值应等于0。

[0099] 在另一个实施例中，代替于始终发信令通知sps_act_enabled_flag，而是对所述标志发信令通知以输入信号的色度类型为条件。具体地，给定ACT仅能够在亮度和色度分量处于相同分辨率时被应用，则仅当输入视频是以4:4:4色度格式被捕获时才发信令通知所述标志sps_act_enabled_flag。在这种改变的情况下，经修改的SPS语法表是：

[0100]

seq_parameter_set_rbsp( ) {	描述符
...	
if(chroma_format_idc == 3)	
sps_act_enabled_flag	u(1)
...	
}	

[0101] 表2具有信令条件的经修改的SPS语法表。

[0102] 在一些实施例中，针对使用ACT来解码视频数据的语法设计规范在下表中说明。

[0103]

coding_unit(x0, y0, cbWidth, cbHeight, cqtDepth, treeType, modeType) {	描述符
chType = treeType = = DUAL_TREE_CHROMA? 1 : 0	
if(slice_type != I || sps_ibc_enabled_flag || sps_palette_enabled_flag) {	
if(treeType != DUAL_TREE_CHROMA && !((cbWidth = = 4 && cbHeight = = 4) || modeType = = MODE_TYPE_INTRA) && !sps_ibc_enabled_flag))	

[0104]

cu_skip_flag[x0][y0]	ae(v)
if(cu_skip_flag[x0][y0] = = 0 && slice_type != I && !(cbWidth = = 4 && cbHeight = = 4) && modeType = = MODE_TYPE_ALL)	
pred_mode_flag	ae(v)
if(((slice_type = = I && cu_skip_flag[x0][y0] = = 0) || (slice_type != I && (CuPredMode[chType][x0][y0] != MODE_INTRA || (cbWidth = = 4 && cbHeight = = 4 && cu_skip_flag[x0][y0] = = 0)))) && cbWidth <= 64 && cbHeight <= 64 && modeType != MODE_TYPE_INTER && sps_ibc_enabled_flag && treeType != DUAL_TREE_CHROMA)	
pred_mode_ibc_flag	ae(v)
if((((slice_type = = I) (cbWidth = = 4 && cbHeight = = 4)) sps_ibc_enabled_flag) && CuPredMode[x0][y0] = = MODE_INTRA) || (slice_type != I && !(cbWidth = = 4 && cbHeight = = 4) && !sps_ibc_enabled_flag && CuPredMode[x0][y0] != MODE_INTRA)) && sps_palette_enabled_flag && cbWidth <= 64 && cbHeight <= 64 && && cu_skip_flag[x0][y0] = = 0 && modeType != MODE_INTER)	
pred_mode_plt_flag	ae(v)
}	
if(CuPredMode[chType][x0][y0] == MODE_INTRA)	
cu_act_enabled_flag	ae(v)
if(CuPredMode[chType][x0][y0] = = MODE_INTRA || CuPredMode[chType][x0][y0] = = MODE_PLT) {	
if(treeType = = SINGLE_TREE) treeType = = DUAL_TREE_LUMA) {	
if(pred_mode_plt_flag) {	
if(treeType = = DUAL_TREE_LUMA)	
palette_coding(x0, y0, cbWidth, cbHeight, 0, 1)	
else /* SINGLE_TREE */	
palette_coding(x0, y0, cbWidth, cbHeight, 0, 3)	
} else {	
if(sps_bdpcm_enabled_flag && cbWidth <= MaxTsSize && cbHeight <= MaxTsSize)	
intra_bdpcm_flag	ae(v)

[0105]

if(intra_bdpcm_flag)	
intra_bdpcm_dir_flag	ae(v)
else {	
if(sps_mip_enabled_flag && (Abs(Log2(cbWidth) - Log2(cbHeight)) <= 2) && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY)	
intra_mip_flag[x0][y0]	ae(v)
if(intra_mip_flag[x0][y0])	
intra_mip_mode[x0][y0]	ae(v)
else {	
if(sps_mrl_enabled_flag && ((y0 % CtbSizeY) > 0))	
intra_luma_ref_idx[x0][y0]	ae(v)
if (sps_isp_enabled_flag && intra_luma_ref_idx[x0][y0] = = 0 && (cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY) && !cu_act_enabled_flag)	
intra_subpartitions_mode_flag[x0][y0]	ae(v)
if(intra_subpartitions_mode_flag[x0][y0] = = 1)	
intra_subpartitions_split_flag[x0][y0]	ae(v)
if(intra_luma_ref_idx[x0][y0] = = 0)	
intra_luma_mpm_flag[x0][y0]	ae(v)
if(intra_luma_mpm_flag[x0][y0]) {	
if(intra_luma_ref_idx[x0][y0] = = 0)	
intra_luma_not_planar_flag[x0][y0]	ae(v)
if(intra_luma_not_planar_flag[x0][y0])	
intra_luma_mpm_idx[x0][y0]	ae(v)
} else	
intra_luma_mpm_remainder[x0][y0]	ae(v)
}	
}	
}	
}	
if((treeType = = SINGLE_TREE || treeType = = DUAL_TREE_CHROMA) && ChromaArrayType != 0) {	
if (pred_mode_plt_flag && treeType = = DUAL_TREE_CHROMA)	
palette_coding(x0, y0, cbWidth / SubWidthC, cbHeight / SubHeightC, 1, 2)	

[0106]

else {	
if(!cu_act_enabled_flag) {	
if(CclmEnabled)	
cclm_mode_flag	ae(v)
if(cclm_mode_flag)	
cclm_mode_idx	ae(v)
else	
intra_chroma_pred_mode	ae(v)
}	
}	
}	
} else if(treeType != DUAL_TREE_CHROMA) { /* MODE_INTER or MODE_IBC */	
if(cu_skip_flag[x0][y0] = = 0)	
general_merge_flag[x0][y0]	ae(v)
if(general_merge_flag[x0][y0]) {	
merge_data(x0, y0, cbWidth, cbHeight, chType)	
} else if (CuPredMode[chType][x0][y0] = = MODE_IBC) {	
mvd_coding(x0, y0, 0, 0)	
if(MaxNumIbcMergeCand > 1)	
mvp_l0_flag[x0][y0]	ae(v)
if(sps_amvr_enabled_flag && (MvdL0[x0][y0][0] != 0 || MvdL0[x0][y0][1] != 0)) {	
amvr_precision_idx[x0][y0]	ae(v)
}	
} else {	
if(slice_type = = B)	
inter_pred_idc[x0][y0]	ae(v)
if(sps_affine_enabled_flag && cbWidth >= 16 && cbHeight >= 16) {	
inter_affine_flag[x0][y0]	ae(v)
if(sps_affine_type_flag && inter_affine_flag[x0][y0])	
cu_affine_type_flag[x0][y0]	ae(v)
}	
if(sps_smvd_enabled_flag && !mvd_l1_zero_flag && inter_pred_idc[x0][y0] = = PRED_BI && !inter_affine_flag[x0][y0] && RefIdxSymL0 > -1 && RefIdxSymL1 > -1)	

[0107]

sym_mvd_flag[x0][y0]	ae(v)
if(inter_pred_idc[x0][y0] != PRED_L1) {	
if(NumRefIdxActive[0] > 1 && !sym_mvd_flag[x0][y0])	
ref_idx_l0[x0][y0]	ae(v)
mvd_coding(x0, y0, 0, 0)	
if(MotionModelIdc[x0][y0] > 0)	
mvd_coding(x0, y0, 0, 1)	
if(MotionModelIdc[x0][y0] > 1)	
mvd_coding(x0, y0, 0, 2)	
mvp_l0_flag[x0][y0]	ae(v)
} else {	
MvdL0[x0][y0][0] = 0	
MvdL0[x0][y0][1] = 0	
}	
if(inter_pred_idc[x0][y0] != PRED_L0) {	
if(NumRefIdxActive[1] > 1 && !sym_mvd_flag[x0][y0])	
ref_idx_l1[x0][y0]	ae(v)
if(mvd_l1_zero_flag && inter_pred_idc[x0][y0] == PRED_BI) {	
MvdL1[x0][y0][0] = 0	
MvdL1[x0][y0][1] = 0	
MvdCpL1[x0][y0][0][0] = 0	
MvdCpL1[x0][y0][0][1] = 0	
MvdCpL1[x0][y0][1][0] = 0	
MvdCpL1[x0][y0][1][1] = 0	
MvdCpL1[x0][y0][2][0] = 0	
MvdCpL1[x0][y0][2][1] = 0	
} else {	
if(sym_mvd_flag[x0][y0]) {	
MvdL1[x0][y0][0] = -MvdL0[x0][y0][0]	
MvdL1[x0][y0][1] = -MvdL0[x0][y0][1]	
} else	
mvd_coding(x0, y0, 1, 0)	
if(MotionModelIdc[x0][y0] > 0)	
mvd_coding(x0, y0, 1, 1)	
if(MotionModelIdc[x0][y0] > 1)	

[0108]

mvd_coding(x0, y0, 1, 2)	
mvp_l1_flag[x0 **][** y0 **]**	ae(v)
}	
} else {	
MvdL1[x0][y0][0] = 0	
MvdL1[x0][y0][1] = 0	
}	
if((sps_amvr_enabled_flag && inter_affine_flag[x0][y0] == 0 && (MvdL0[x0][y0][0] != 0 || MvdL0[x0][y0][1] != 0 || MvdL1[x0][y0][0] != 0 || MvdL1[x0][y0][1] != 0)) || (sps_affine_amvr_enabled_flag && inter_affine_flag[x0][y0] == 1 && (MvdCpL0[x0][y0][0][0] != 0 || MvdCpL0[x0][y0][0][1] != 0 || MvdCpL1[x0][y0][0][0] != 0 || MvdCpL1[x0][y0][0][1] != 0 || MvdCpL0[x0][y0][1][0] != 0 || MvdCpL0[x0][y0][1][1] != 0 || MvdCpL1[x0][y0][1][0] != 0 || MvdCpL1[x0][y0][1][1] != 0 || MvdCpL0[x0][y0][2][0] != 0 || MvdCpL0[x0][y0][2][1] != 0 || MvdCpL1[x0][y0][2][0] != 0 || MvdCpL1[x0][y0][2][1] != 0))) {	
amvr_flag[x0 **][** y0 **]**	ae(v)
if(amvr_flag[x0][y0])	
amvr_precision_idx[x0 **][** y0 **]**	ae(v)
}	
if(sps_bcw_enabled_flag && inter_pred_idc[x0][y0] == PRED_BI && luma_weight_l0_flag[ref_idx_l0 [x0][y0]] == 0 && luma_weight_l1_flag[ref_idx_l1 [x0][y0]] == 0 && chroma_weight_l0_flag[ref_idx_l0 [x0][y0]] == 0 && chroma_weight_l1_flag[ref_idx_l1 [x0][y0]] == 0 && cbWidth * cbHeight >= 256)	
bcw_idx[x0 **][** y0 **]**	ae(v)
}	
}	
if(CuPredMode[chType][x0][y0] != MODE_INTRA && !pred_mode_plt_flag && general_merge_flag[x0][y0] == 0)	
cu_cbf	ae(v)

[0109]

if(cu_cbf) {	
if(CuPredMode[chType][x0][y0] == MODE_INTER && sps_sbt_enabled_flag && !ciip_flag[x0][y0] && !MergeTriangleFlag[x0][y0]) {	
if(cbWidth <= MaxSbtSize && cbHeight <= MaxSbtSize) {	
allowSbtVerH = cbWidth >= 8	
allowSbtVerQ = cbWidth >= 16	
allowSbtHorH = cbHeight >= 8	
allowSbtHorQ = cbHeight >= 16	
if(allowSbtVerH || allowSbtHorH || allowSbtVerQ || allowSbtHorQ)	
cu_sbt_flag	ae(v)
}	
if(cu_sbt_flag) {	
if((allowSbtVerH || allowSbtHorH) && (allowSbtVerQ || allowSbtHorQ))	
cu_sbt_quad_flag	ae(v)
if((cu_sbt_quad_flag && allowSbtVerQ && allowSbtHorQ) || (!cu_sbt_quad_flag && allowSbtVerH && allowSbtHorH))	
cu_sbt_horizontal_flag	ae(v)
cu_sbt_pos_flag	ae(v)
}	
if(sps_act_enabled_flag && CuPredMode[chType][x0][y0] != MODE_INTRA)	
cu_act_enabled_flag	ae(v)
}	
LfnstDcOnly = 1	
LfnstZeroOutSigCoeffFlag = 1	
transform_tree(x0, y0, cbWidth, cbHeight, treeType)	
lfnstWidth = (treeType == DUAL_TREE_CHROMA) ? cbWidth / SubWidthC : cbWidth	
lfnstHeight = (treeType == DUAL_TREE_CHROMA) ? cbHeight / SubHeightC : cbHeight	
if(Min(lfnstWidth, lfnstHeight) >= 4 && sps_lfnst_enabled_flag == 1 && CuPredMode[chType][x0][y0] == MODE_INTRA && IntraSubPartitionsSplitType == ISP_NO_SPLIT && (!intra_mip_flag[x0][y0] || Min(lfnstWidth, lfnstHeight) >= 16) && tu_mts_idx[x0][y0] == 0 && Max(cbWidth, cbHeight) <= MaxTbSizeY) {	
if(LfnstDcOnly == 0 && LfnstZeroOutSigCoeffFlag == 1)	

[0110]

lfnst_idx[x0][y0]	ae(v)
}	
}	

[0111] 表3针对发信令通知ACT模式的规范。

[0112] 标志cu_act_enabled_flag等于1指示编码单元的残差是在YCgCo颜色空间中编码的，并且标志cu_act_enabled_flag等于0指示编码单元的残差是在原始颜色空间(例如，RGB或YCbCr)中编码的。当标志cu_act_enabled_flag不存在时，它被推断为等于0。

[0113] 图8A和8B是图示了根据本公开的一些实现方式的示例性视频解码过程的框图，通过所述解码过程，视频解码器实现逆自适应颜色空间变换ACT和具有色度缩放的亮度映射的技术。在一些实施例中，使用两个ACT(例如，图7中的逆ACT)和色度残差缩放(例如，图6中的色度残差缩放)来编码视频比特流。在一些其他实施例中，使用色度残差缩放但不使用两个ACT来编码视频比特流，从而不需要逆ACT。

[0114] 更具体地，图8A描绘了一实施例，其中视频编解码器在逆ACT之前执行色度残差缩放。作为结果，视频编解码器在颜色空间变换域中利用色度残差缩放来执行亮度映射。例如，假设输入视频以RGB格式被捕获并且被变换到YCgCo颜色空间中，则视频编解码器根据YCgCo颜色空间中的亮度残差Y对色度残差Cg和Co执行色度残差缩放。

[0115] 图8B描绘了替代实施例，其中视频编解码器在逆ACT之后执行色度残差缩放。作为结果，视频编解码器在原始颜色空间域中利用色度残差缩放来执行亮度映射。例如，假设输入视频是以RGB格式被捕获的，则视频编解码器对B和R分量应用色度残差缩放。

[0116] 图9是图示了根据本公开的一些实现方式的在执行自适应颜色空间变换ACT与块差分脉冲编码调制BDPCM之间的示例性解码逻辑的框图。

[0117] BDPCM是用于屏幕内容编码的编解码工具。在一些实施例中，在序列级别，在SPS中发信令通知BDPCM启用标志。仅当SPS中启用了变换跳过模式时，才发信令通知BDPCM启用标志。

[0118] 当BDPCM被启用时，如果CU大小在亮度样本方面小于或等于MaxTsSize×MaxTsSize并且如果CU是帧内编码的，则在CU级别发信令通知一标志，其中MaxTsSize是变换跳过模式被允许的最大块大小。所述标志指示是使用常规帧内编码还是BDPCM。如果使用BDPCM，则进一步传输另一个BDPCM预测方向标志，以指示预测是水平的还是垂直的。然后，使用常规的水平或垂直帧内预测过程、利用未滤波的参考样本来预测所述块。对残差进行量化，并且对每个经量化的残差与其预测器——即，水平或垂直(取决于BDPCM预测方向)相邻位置的先前编码的残差——之间的差异进行编码。

[0119] 对于大小为M(高度)×N(宽度)的块，令$r_{i,j}$(0≤i≤M-1，0≤j≤N-1)为预测残差。令Q($r_{i,j}$)(0≤i≤M-1，0≤j≤N-1)表示残差$r_{i,j}$的量化版本。BDPCM被应用于经量化的残差值，从而得到具有元素$\tilde{r}_{i,j}$的经修改的M×N数组$\tilde{R}$，其中$\tilde{r}_{i,j}$是从其相邻的经量化的残差值来预测的。对于垂直BDPCM预测模式，对于0≤j≤(N-1)，以下内容用于导出$\tilde{r}_{i,j}$：

[0120] $$\tilde{r}_{i,j}=\begin{cases}Q(r_{i,j}), & i=0\\ Q(r_{i,j})-Q(r_{(i-1),j}), & 1\le i\le (M-1)\end{cases} \quad (1)$$ 。

[0121] 对于水平BDPCM预测模式，对于0≤i≤(M-1)，以下内容用于导出$\tilde{r}_{i,j}$：

[0122] $$\tilde{r}_{i,j}=\begin{cases}Q(r_{i,j}), & j=0\\ Q(r_{i,j})-Q(r_{i,(j-1)}), & 1\le j\le (N-1)\end{cases} \quad (2)$$ 。

[0123] 在解码器侧处，上述过程被反转以计算$Q(r_{i,j})$，$0 \leq i \leq M-1$，$0 \leq j \leq N-1$，如下：

[0124] 如果使用垂直BDPCM，则 $Q(r_{i,j}) = \sum_{k=0}^{i} \tilde{r}_{k,j}$ (3)

[0125] 如果使用水平BDPCM，则 $Q(r_{i,j}) = \sum_{k=0}^{j} \tilde{r}_{i,k}$ (4) 。

[0126] 经逆量化的残差$Q^{-1}(Q(r_{i,j}))$被加到帧内块预测值，以产生重建的样本值。

[0127] 使用与在变换跳过模式残差编码中相同的残差编码过程将预测的量化残差值$\tilde{r}_{i,j}$发送到解码器。在用于未来帧内模式编码的MPM模式方面，如果BDPCM预测方向分别是水平或垂直的，则针对BDPCM编码的CU来存储水平或垂直预测模式。对于去块而言，如果使用BDPCM对块边界的侧上的两个块进行编码，则所述特定块边界不会被去块。根据最新的VVC工作草案，当输入视频是4:4:4色度格式时，可以通过在CU级别针对亮度和色度通道发信令通知两个单独的标志(即，intra_bdpcm_luma_flag和intra_bdpcm_chroma_flag)来将BDPCM应用于亮度和色度分量两者。

[0128] 在一些实施例中，视频编解码器执行不同的逻辑以更好地处理ACT与BDPCM之间的相互作用。例如，当ACT被应用于一个帧内CU时，针对亮度分量启用BDPCM，但是针对色度分量禁用BDPCM(910)。编码单元的对应经修改的语法表被说明如下：

[0129]

coding_unit(x0, y0, cbWidth, cbHeight, cqtDepth, treeType, modeType) {	描述符
.....	
if(CuPredMode[chType][x0][y0] == MODE_INTRA && treeType == SINGLE_TREE)	
cu_act_enabled_flag	ae(v)
if(CuPredMode[chType][x0][y0] == MODE_INTRA || CuPredMode[chType][x0][y0] == MODE_PLT) {	
if(treeType == SINGLE_TREE || treeType == DUAL_TREE_LUMA) {	
if(pred_mode_plt_flag) {	
if(treeType == DUAL_TREE_LUMA)	

[0130]

palette_coding(x0, y0, cbWidth, cbHeight, 0, 1)	
else /* SINGLE_TREE */	
palette_coding(x0, y0, cbWidth, cbHeight, 0, 3)	
} else {	
if(sps_bdpcm_enabled_flag && cbWidth <= MaxTsSize && cbHeight <= MaxTsSize)	
intra_bdpcm_flag	ae(v)
if(intra_bdpcm_flag)	
intra_bdpcm_dir_flag	ae(v)
else {	
if(sps_mip_enabled_flag && (Abs(Log2(cbWidth) - Log2(cbHeight)) <= 2) && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY)	
intra_mip_flag[x0][y0]	ae(v)
if(intra_mip_flag[x0][y0])	
intra_mip_mode[x0][y0]	ae(v)
else {	
if(sps_mrl_enabled_flag && ((y0 % CtbSizeY) > 0))	
intra_luma_ref_idx[x0][y0]	ae(v)
if (sps_isp_enabled_flag && intra_luma_ref_idx[x0][y0] = = 0 && (cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY) && !cu_act_enabled_flag)	
intra_subpartitions_mode_flag[x0][y0]	ae(v)
if(intra_subpartitions_mode_flag[x0][y0] = = 1)	
intra_subpartitions_split_flag[x0][y0]	ae(v)
if(intra_luma_ref_idx[x0][y0] = = 0)	
intra_luma_mpm_flag[x0][y0]	ae(v)
if(intra_luma_mpm_flag[x0][y0]) {	
if(intra_luma_ref_idx[x0][y0] = = 0)	
intra_luma_not_planar_flag[x0][y0]	ae(v)
if(intra_luma_not_planar_flag[x0][y0])	
intra_luma_mpm_idx[x0][y0]	ae(v)
} else	
intra_luma_mpm_remainder[x0][y0]	ae(v)
}	
}	

[0131]

}	
}	
if (treeType = = SINGLE_TREE || treeType = = DUAL_TREE_CHROMA) && ChromaArrayType != 0) {	
if (pred_mode_plt_flag && treeType = = DUAL_TREE_CHROMA)	
palette_coding(x0, y0, cbWidth / SubWidthC, cbHeight / SubHeightC, 1, 2)	
else {	
if(!cu_act_enabled_flag) {	
if(cbWidth <= MaxTsSize && cbHeight <= MaxTsSize && sps_bdpcm_chroma_enabled_flag) {	
intra_bdpcm_chroma_flag	ae(v)
if(intra_bdpcm_chroma_flag)	
intra_bdpcm_chroma_dir_flag	ae(v)
} else {	
if(CclmEnabled)	
cclm_mode_flag	ae(v)
if(cclm_mode_flag)	
cclm_mode_idx	ae(v)
else	
intra_chroma_pred_mode	ae(v)
}	
}	
}	
}	
.....	
}	

[0132] 表4仅针对亮度分量而启用的BDPCM。

[0133] 在一些实施例中，当ACT被应用于一个帧内CU时，针对亮度和色度分量两者启用BDPCM(920)。编码单元的对应经修改的语法表被说明如下：

[0134]

coding_unit(x0, y0, cbWidth, cbHeight, cqtDepth, treeType, modeType) {	描述符
.....	
if(CuPredMode[chType][x0][y0] == MODE_INTRA && treeType == SINGLE_TREE)	
cu_act_enabled_flag	ae(v)

[0135]

if(CuPredMode[chType][x0][y0] == MODE_INTRA || CuPredMode[chType][x0][y0] == MODE_PLT) {	
if(treeType = = SINGLE_TREE || treeType = = DUAL_TREE_LUMA) {	
if(pred_mode_plt_flag) {	
if(treeType = = DUAL_TREE_LUMA)	
palette_coding(x0, y0, cbWidth, cbHeight, 0, 1)	
else /* SINGLE_TREE */	
palette_coding(x0, y0, cbWidth, cbHeight, 0, 3)	
} else {	
if(sps_bdpcm_enabled_flag && cbWidth <= MaxTsSize && cbHeight <= MaxTsSize)	
intra_bdpcm_flag	ae(v)
if(intra_bdpcm_flag)	
intra_bdpcm_dir_flag	ae(v)
else {	
if(sps_mip_enabled_flag && (Abs(Log2(cbWidth) - Log2(cbHeight)) <= 2) && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY)	
intra_mip_flag[x0][y0]	ae(v)
if(intra_mip_flag[x0][y0])	
intra_mip_mode[x0][y0]	ae(v)
else {	
if(sps_mrl_enabled_flag && ((y0 % CtbSizeY) > 0))	
intra_luma_ref_idx[x0][y0]	ae(v)
if (sps_isp_enabled_flag && intra_luma_ref_idx[x0][y0] = = 0 && (cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY) && !cu_act_enabled_flag)	
intra_subpartitions_mode_flag[x0][y0]	ae(v)
if(intra_subpartitions_mode_flag[x0][y0] = = 1)	
intra_subpartitions_split_flag[x0][y0]	ae(v)
if(intra_luma_ref_idx[x0][y0] = = 0)	
intra_luma_mpm_flag[x0][y0]	ae(v)
if(intra_luma_mpm_flag[x0][y0]) {	
if(intra_luma_ref_idx[x0][y0] = = 0)	
intra_luma_not_planar_flag[x0][y0]	ae(v)
if(intra_luma_not_planar_flag[x0][y0])	

[0136]

intra_luma_mpm_idx[x0][y0]	ae(v)
} else	
intra_luma_mpm_remainder[x0][y0]	ae(v)
}	
}	
}	
}	
if((treeType = = SINGLE_TREE || treeType = = DUAL_TREE_CHROMA) && ChromaArrayType != 0) {	
if (pred_mode_plt_flag && treeType = = DUAL_TREE_CHROMA)	
palette_coding(x0, y0, cbWidth / SubWidthC, cbHeight / SubHeightC, 1, 2)	
else {	
if(cbWidth <= MaxTsSize && cbHeight <= MaxTsSize && sps_bdpcm_chroma_enabled_flag) {	
intra_bdpcm_chroma_flag	ae(v)
if(intra_bdpcm_chroma_flag)	
intra_bdpcm_chroma_dir_flag	ae(v)
} else {	
if(!cu_act_enabled_flag) {	
if(CclmEnabled)	
cclm_mode_flag	ae(v)
if(cclm_mode_flag)	
cclm_mode_idx	ae(v)
else	
intra_chroma_pred_mode	ae(v)
}	
}	
}	
}	
.....	
}	

[0137] 表5针对亮度和色度分量两者而启用的BDPCM。

[0138] 在一些实施例中，当ACT被应用于一个帧内CU时，针对亮度和色度分量两者禁用BDPCM(930)。在这种情况下，不需要发信令通知BDPCM相关的语法元素。编码单元的对应经修改的语法表被说明如下：

[0139]

coding_unit(x0, y0, cbWidth, cbHeight, cqtDepth, treeType, modeType) {	描述符
……	
if(CuPredMode[chType][x0][y0] == MODE_INTRA && treeType == SINGLE_TREE)	
cu_act_enabled_flag	ae(v)
if(CuPredMode[chType][x0][y0] == MODE_INTRA || CuPredMode[chType][x0][y0] == MODE_PLT) {	
if(treeType == SINGLE_TREE || treeType == DUAL_TREE_LUMA) {	
if(pred_mode_plt_flag) {	
if(treeType == DUAL_TREE_LUMA)	
palette_coding(x0, y0, cbWidth, cbHeight, 0, 1)	
else /* SINGLE_TREE */	
palette_coding(x0, y0, cbWidth, cbHeight, 0, 3)	
} else {	
if(sps_bdpcm_enabled_flag && cbWidth <= MaxTsSize && cbHeight <= MaxTsSize && !cu_act_enabled_flag)	
intra_bdpcm_flag	ae(v)
if(intra_bdpcm_flag)	
intra_bdpcm_dir_flag	ae(v)
else {	
if(sps_mip_enabled_flag && (Abs(Log2(cbWidth) – Log2(cbHeight)) <= 2) && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY)	
intra_mip_flag[x0][y0]	ae(v)
if(intra_mip_flag[x0][y0])	
intra_mip_mode[x0][y0]	ae(v)
else {	
if(sps_mrl_enabled_flag && ((y0 % CtbSizeY) > 0))	
intra_luma_ref_idx[x0][y0]	ae(v)
if (sps_isp_enabled_flag && intra_luma_ref_idx[x0][y0] == 0 && (cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY) && !cu_act_enabled_flag)	
intra_subpartitions_mode_flag[x0][y0]	ae(v)
if(intra_subpartitions_mode_flag[x0][y0] == 1)	

intra_subpartitions_split_flag[x0][y0]	ae(v)
if(intra_luma_ref_idx[x0][y0] = = 0)	
intra_luma_mpm_flag[x0][y0]	ae(v)
if(intra_luma_mpm_flag[x0][y0]) {	
if(intra_luma_ref_idx[x0][y0] = = 0)	
intra_luma_not_planar_flag[x0][y0]	ae(v)
if(intra_luma_not_planar_flag[x0][y0])	
intra_luma_mpm_idx[x0][y0]	ae(v)
} else	
intra_luma_mpm_remainder[x0][y0]	ae(v)
}	
}	
}	
}	
if((treeType = = SINGLE_TREE || treeType = = DUAL_TREE_CHROMA) && ChromaArrayType != 0) {	
if (pred_mode_plt_flag && treeType = = DUAL_TREE_CHROMA)	
palette_coding(x0, y0, cbWidth / SubWidthC, cbHeight / SubHeightC, 1, 2)	
else {	
if(!cu_act_enabled_flag) {	
if(cbWidth <= MaxTsSize && cbHeight <= MaxTsSize && sps_bdpcm_chroma_enabled_flag) {	
intra_bdpcm_chroma_flag	ae(v)
if(intra_bdpcm_chroma_flag)	
intra_bdpcm_chroma_dir_flag	ae(v)
} else {	
if(CclmEnabled)	
cclm_mode_flag	ae(v)
if(cclm_mode_flag)	
cclm_mode_idx	ae(v)
else	
intra_chroma_pred_mode	ae(v)
}	
}	
}	

[0140]

}	
}	

[0141]

[0142] 表6针对亮度和色度分量两者而禁用BDPCM。

[0143] 图10是图示了根据本公开的一些实现方式的示例性过程的流程图1000,通过所述过程,视频编解码器通过基于逆自适应颜色空间变换ACT与其他编解码工具(例如,LMCS和BDPCM)之间的相互作用而有条件地执行ACT技术来解码视频数据。

[0144] 作为第一步骤,视频解码器从比特流接收切片的切片头部中的第一语法元素,所述第一语法元素指示具有色度缩放的亮度映射LMCS是否被应用于所述切片中的编码单元(1010)。

[0145] 然后,视频解码器从比特流接收针对所述切片中的所述编码单元的第二语法元素(例如,cu_act_enabled_flag)(1020)。所述第二语法元素指示是否已经使用自适应颜色空间变换ACT对所述编码单元进行编码。

[0146] 如果所述第二语法元素具有非零值(例如,指示已经使用ACT对所述编码单元进行编码),则视频解码器通过应用逆ACT以将所述编码单元的亮度残差和色度残差从与所述编码单元相对应的视频数据的经变换的颜色空间转换到原始颜色空间解码所述视频数据(1030)。

[0147] 在应用逆ACT以转换所述编码单元的亮度残差和色度残差之后,如果所述第一语法元素具有非零值(例如,指示已经利用LMCS对所述编码单元进行编码),则视频解码器通过对所述编码单元的亮度样本执行逆亮度映射以及对所述编码单元的色度残差执行逆缩放来解码与所述编码单元相对应的视频数据(1040)。

[0148] 在一些实施例中,在接收所述第二语法元素之前,视频解码器从比特流中接收第三语法元素(例如,chroma_format_idc),其中所述第三语法元素指示视频数据是否具有预定义色度格式。如上所指出,ACT仅针对某些预定义色度格式(例如,4:4:4色度格式)而启用。

[0149] 在一些实施例中,仅当视频数据具有所述预定义色度格式时,所述第二语法元素(例如,cu_act_enabled_flag)才存在于比特流中(例如,当视频数据不是4:4:4色度格式时,ACT被禁用)。

[0150] 在一些实施例中,经变换的颜色空间是YCgCo颜色空间,而原始颜色空间可以是RGB颜色空间或YCbCr颜色空间。

[0151] 在一些实施例中,在执行逆ACT以转换所述编码单元的亮度残差和色度残差之后,视频解码器通过以下方式对所述编码单元的亮度样本执行逆亮度映射以及对所述编码单元的色度残差执行逆缩放:通过将所述原始颜色空间中的亮度残差添加到预测的亮度样本来生成所述编码单元的映射域中的中间亮度样本;使用一个或多个自适应逐段线性模型将所述映射域中的所述中间亮度样本转换成预映射域中的重建的亮度分量;根据所述映射域中的对应中间亮度样本对所述色度残差进行逆缩放;以及使用经缩放的色度残差和预测的色度样本来重建所述编码单元的色度分量。

[0152] 在一些实施例中,视频解码器从比特流接收与编码单元相对应的视频数据,其中所述编码单元是通过帧内预测模式编码的。视频编解码器从视频数据接收第一语法元素(例如,cu_act_enabled_flag),其中所述第一语法元素指示是否已经使用自适应颜色空间变换ACT对所述编码单元进行编码。如果所述第一语法元素具有零值,则视频编解码器从视频数据接收一个或多个语法元素(例如,intra_bdpcm_chroma_flag、intra_bdpcm_chroma_

dir_flag)，其中所述一个或多个语法元素指示是否已经使用块差分脉冲编码调制BDPCM对所述编码单元的色度分量进行编码。如果所述第一语法元素具有非零值，则视频编解码器将默认值指派给与BDPCM相关联的所述一个或多个语法元素。默认值可以指示使用逆BDPCM或以其他方式对所述编码单元的色度分量进行解码。然后，视频编解码器根据与ACT相关联的所述第一语法元素和与所述BDPCM相关联的所述一个或多个语法元素从视频数据来解码所述编码单元。

[0153] 在一些实施例中，在接收所述第一语法元素之前，视频解码器从视频数据接收第二语法元素，其中所述第二语法元素指示视频数据是否具有预定义色度格式。如上所指出，ACT仅针对某些预定义色度格式(例如，4:4:4色度格式)而启用。

[0154] 在一些实施例中，仅当视频数据具有所述预定义色度格式时，所述第一语法元素才存在于比特流中。

[0155] 在一些实施例中，当所述编码单元是在不应用颜色空间变换的情况下编码时，使用逆BDPCM对所述编码单元进行解码。

[0156] 在一个或多个示例中，所描述的功能可以用硬件、软件、固件或其任何组合来实现。如果以软件来实现，这些功能可以作为一个或多个指令或代码被存储在计算机可读介质上或在其上传输，并且由基于硬件的处理单元来执行。计算机可读介质可以包括计算机可读存储介质，所述介质对应于诸如数据存储介质之类的有形介质、或者通信介质，所述通信介质包括例如根据通信协议便于将计算机程序从一个地方传送到另一个地方的任何介质。以这种方式，计算机可读介质通常可以对应于(1)有形的非暂时性计算机可读存储介质、或(2)诸如信号或载波之类的通信介质。数据存储介质可以是能够被一个或多个计算机或一个或多个处理器访问以检索用于实现本申请中描述的实现方式的指令、代码和/或数据结构的任何可用介质。计算机程序产品可以包括计算机可读介质。

[0157] 在本文中的实现方式的描述中使用的术语仅仅是出于描述特定实现方式的目的，而不是意图限制权利要求的范围。如在实现方式和所附权利要求的描述中使用的那样，单数形式“一(a、and)”、和“所述”也意图包括复数形式，除非上下文清楚地另行指示。还将理解的是，本文中使用的术语“和/或”指代并涵盖一个或多个相关联的所列项目的任何和所有可能的组合。将进一步理解的是，术语“包括”和/或“包含”当在本说明书中使用时指定声明的特征、元件和/或组件的存在，但是不排除一个或多个其他特征、元件、组件和/或其群组的存在或添加。

[0158] 还将理解的是，尽管术语第一、第二等在本文中可用于描述各种元件，但是这些元件不应当受这些术语限制。这些术语仅用于将一个元件与另一个元件进行区分。例如，在不脱离实现方式的范围的情况下，第一电极可以被称为第二电极，并且类似地，第二电极可以被称为第一电极。第一电极和第二电极两者都是电极，但是它们不是相同的电极。

[0159] 本申请的描述是出于说明和描述的目的而给出的，并且并不意图是穷尽的或将本发明限于公开的形式。受益于前述描述和相关附图中呈现的教导，许多修改、变化和替代实现方式对于本领域普通技术人员来说将是明显的。选择和描述所述实施例是为了最好地解释本发明的原理、实际应用，并且使得本领域的其他技术人员能够针对各种实现方式来理解本发明，并且最好地利用基本原理以及在适合于预期的特定用途时利用具有各种修改的各种实现方式。因此，要理解的是，权利要求的范围不限于公开的实现方式的具体示例，并

且修改和其他实现方式意图被包括在所附权利要求的范围内。

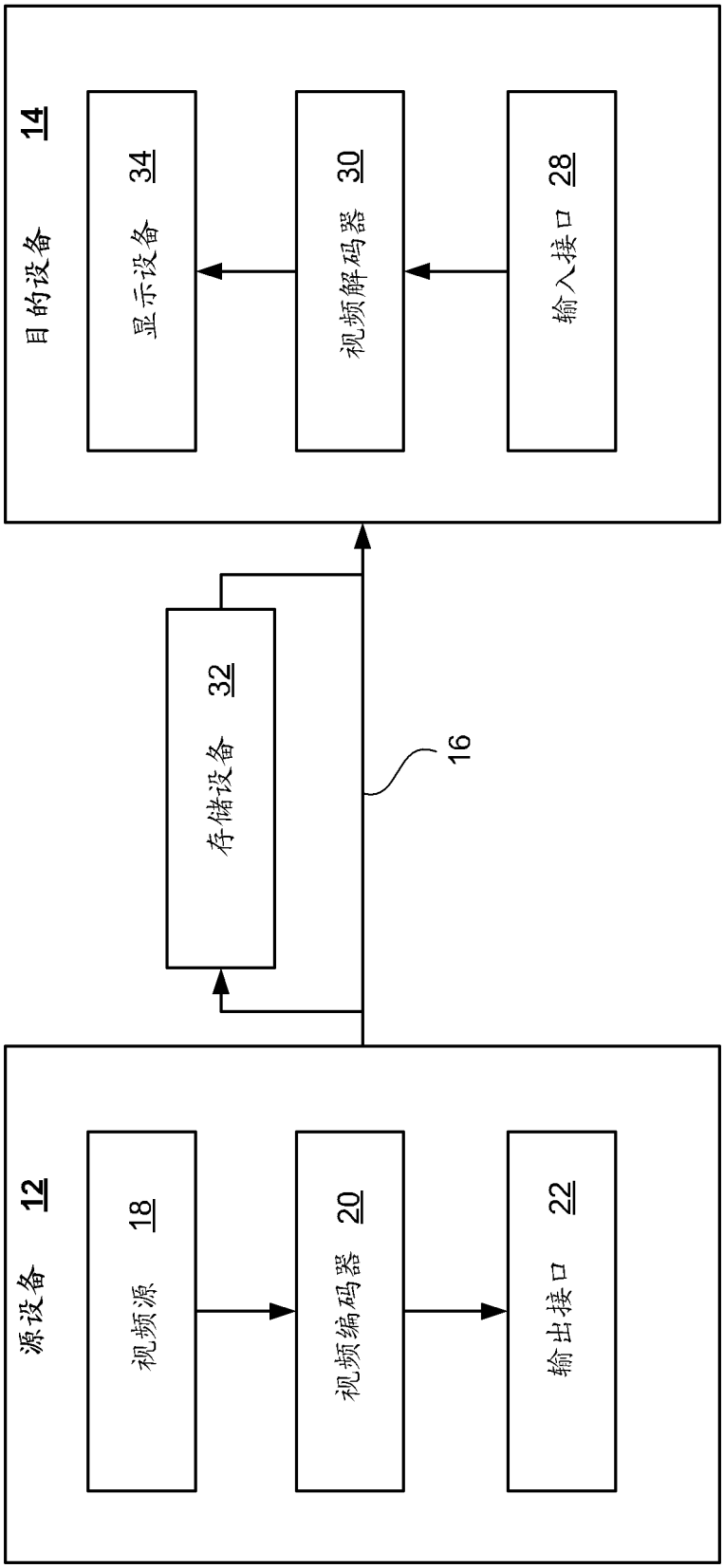

目的设备 14
显示设备 34
视频解码器 30
输入接口 28
存储设备 32
16
源设备 12
视频源 18
视频编码器 20
输出接口 22

图 1

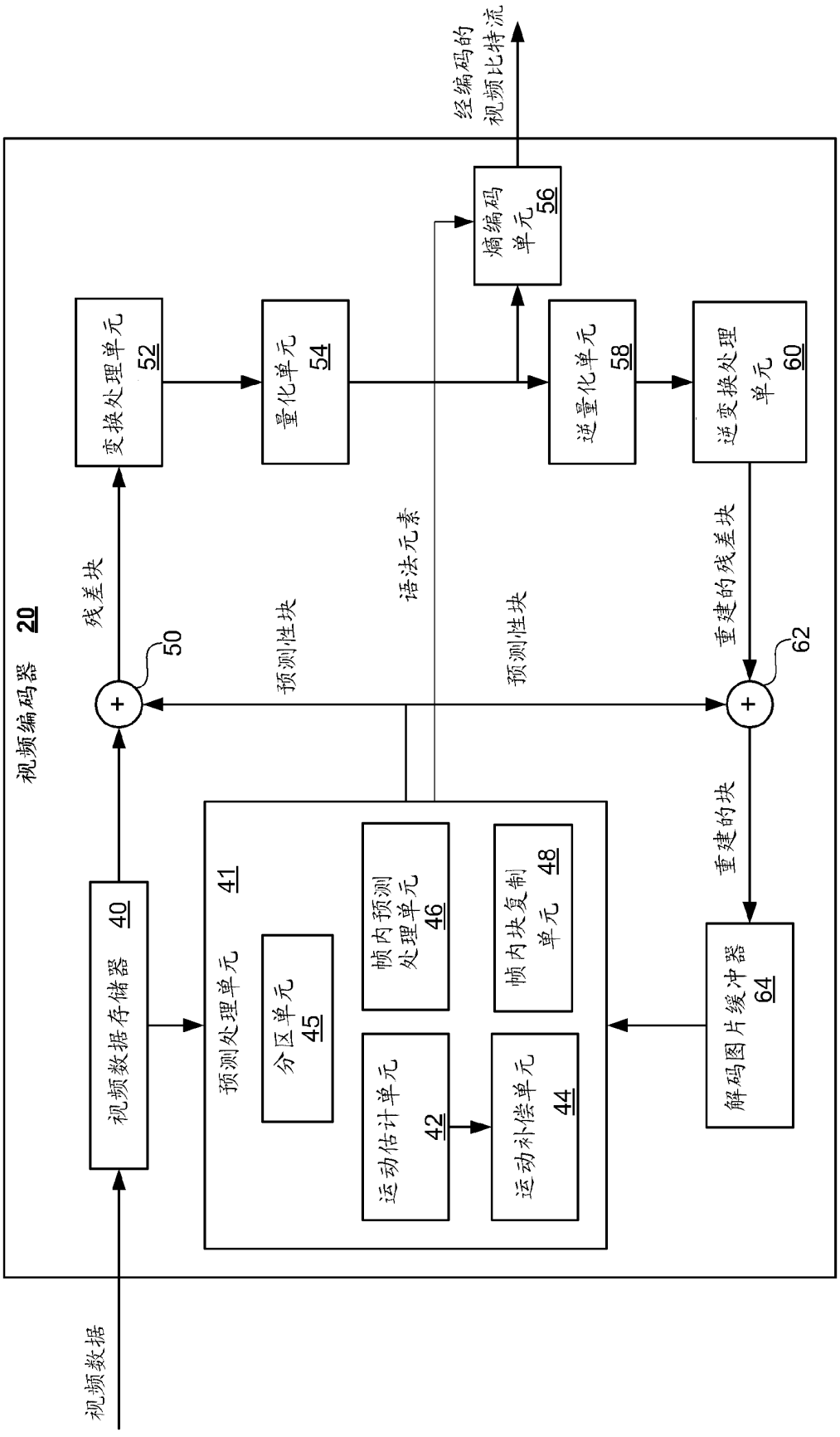

视频数据
视频编码器 20
视频数据存储器 40
预测处理单元 41
分区单元 45
帧内预测处理单元 46
帧内块复制单元 48
运动估计单元 42
运动补偿单元 44
解码图片缓冲器 64
残差块
50
变换处理单元 52
量化单元 54
熵编码单元 56
经编码的视频比特流
预测性块
语法元素
预测性块
逆量化单元 58
逆变换处理单元 60
重建的残差块
62
重建的块

图 2

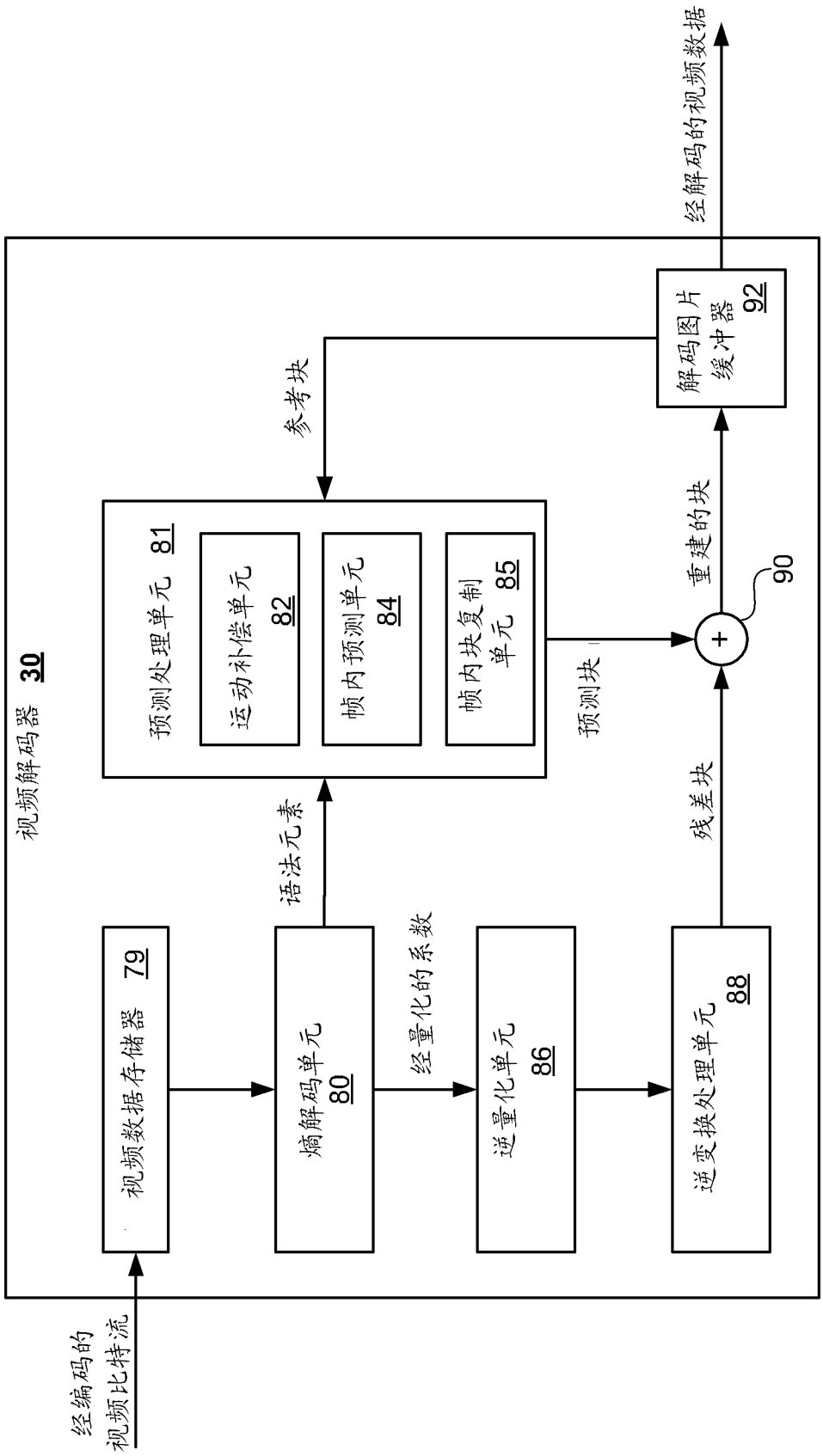

经编码的
视频比特流
视频解码器 30
视频数据存储器 79
熵解码单元
80
预测处理单元 81
运动补偿单元
82
帧内预测单元
84
帧内块复制
单元 85
逆量化单元
86
逆变换处理单元
88
解码图片
缓冲器
92
+
90
语法元素
经量化的系数
残差块
预测块
参考块
重建的块
经解码的视频数据

图 3

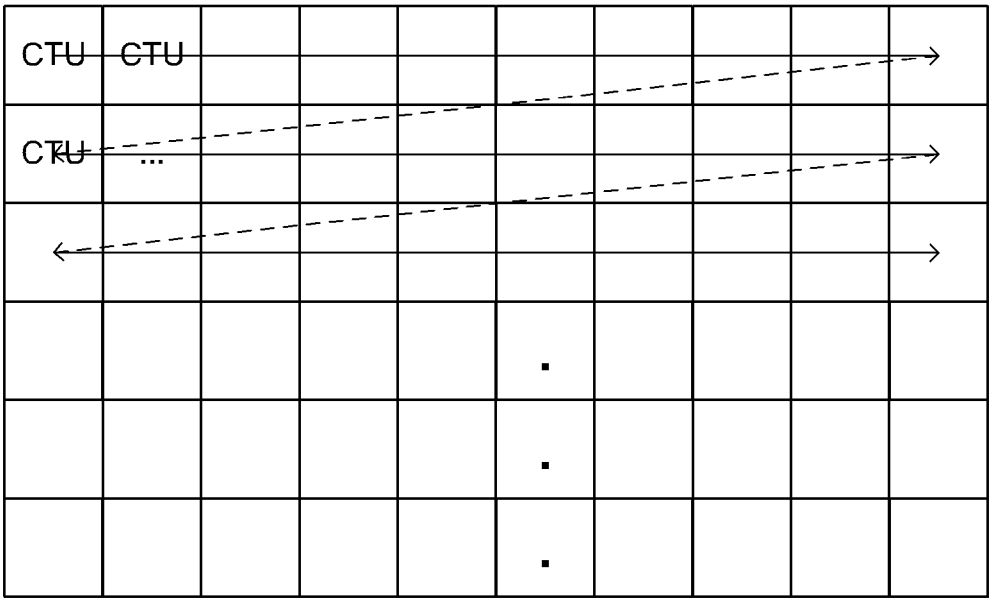


图 4A

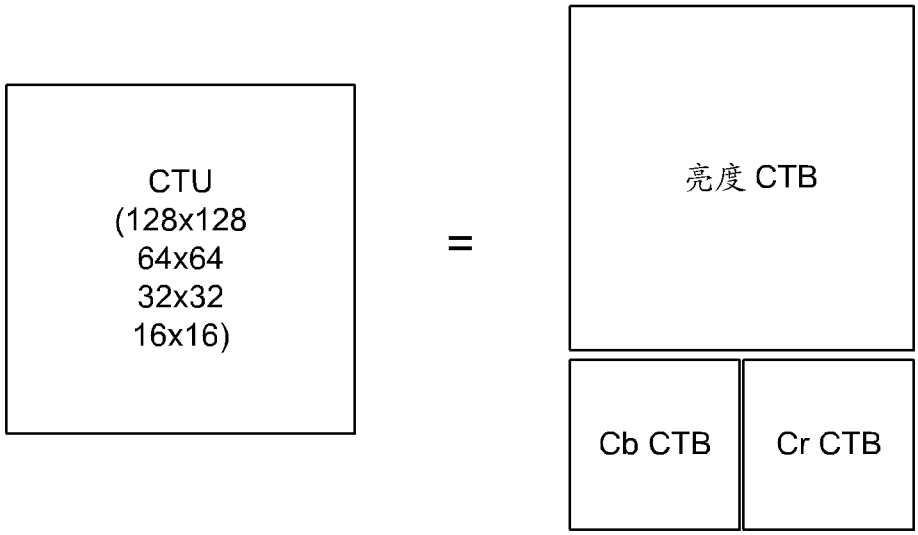


图 4B

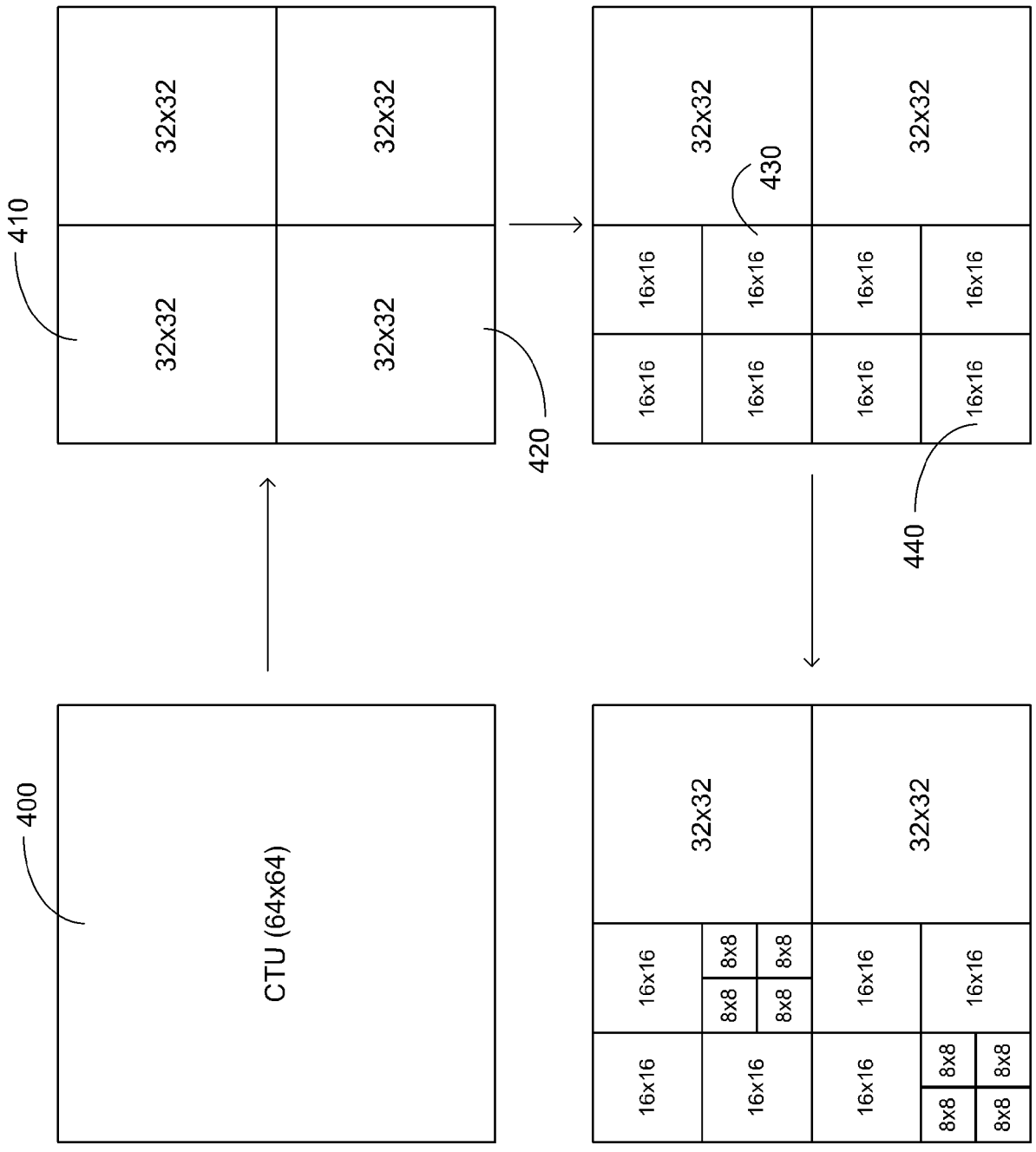

400
CTU (64x64)
410
32x32
32x32
32x32
32x32
420
430
440
16x16
8x8

图 4C

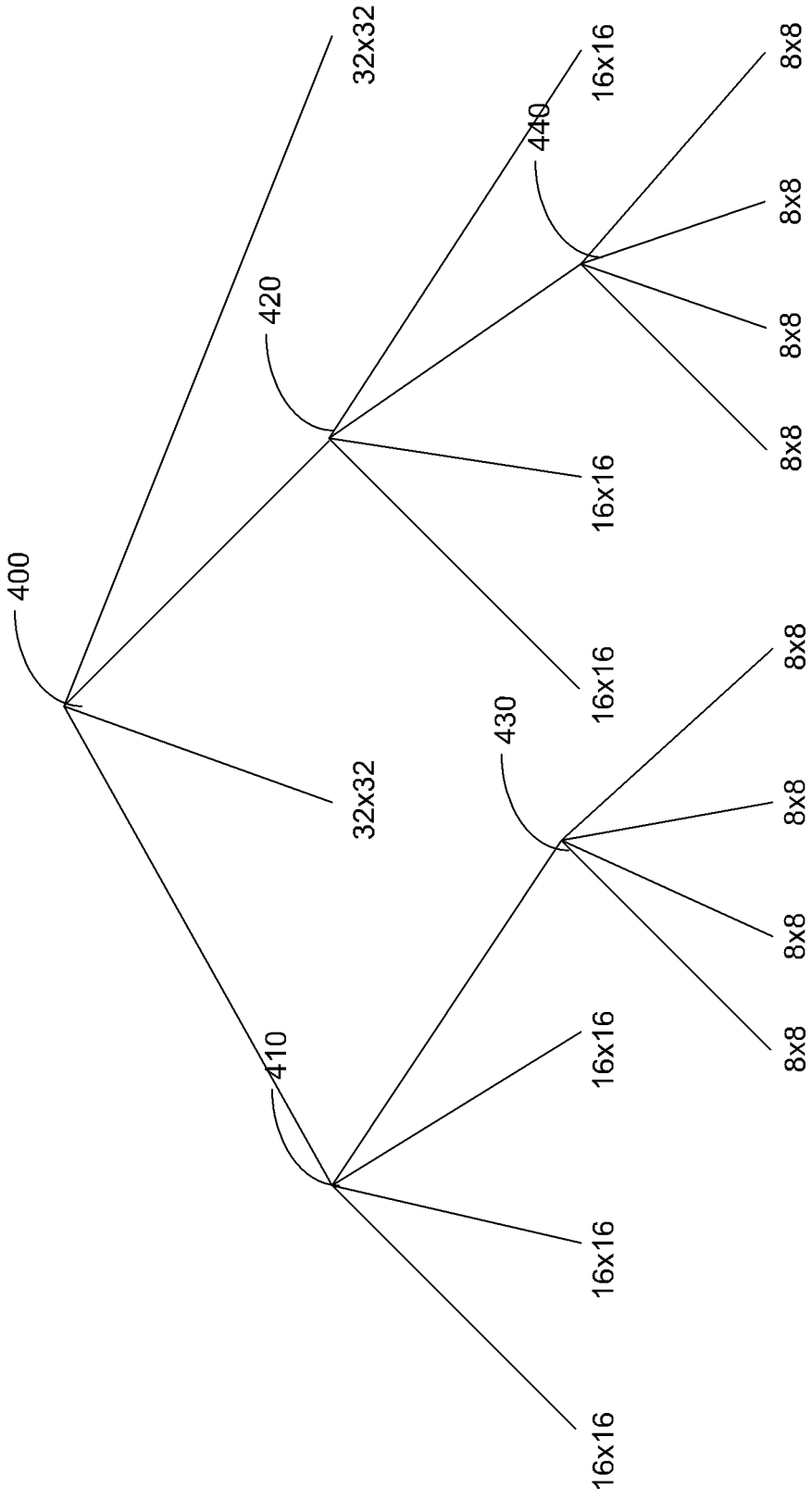

400
32x32
420
16x16
440
8x8
8x8
8x8
16x16
16x16
32x32
430
8x8
8x8
8x8
410
16x16
16x16
16x16

图 4D

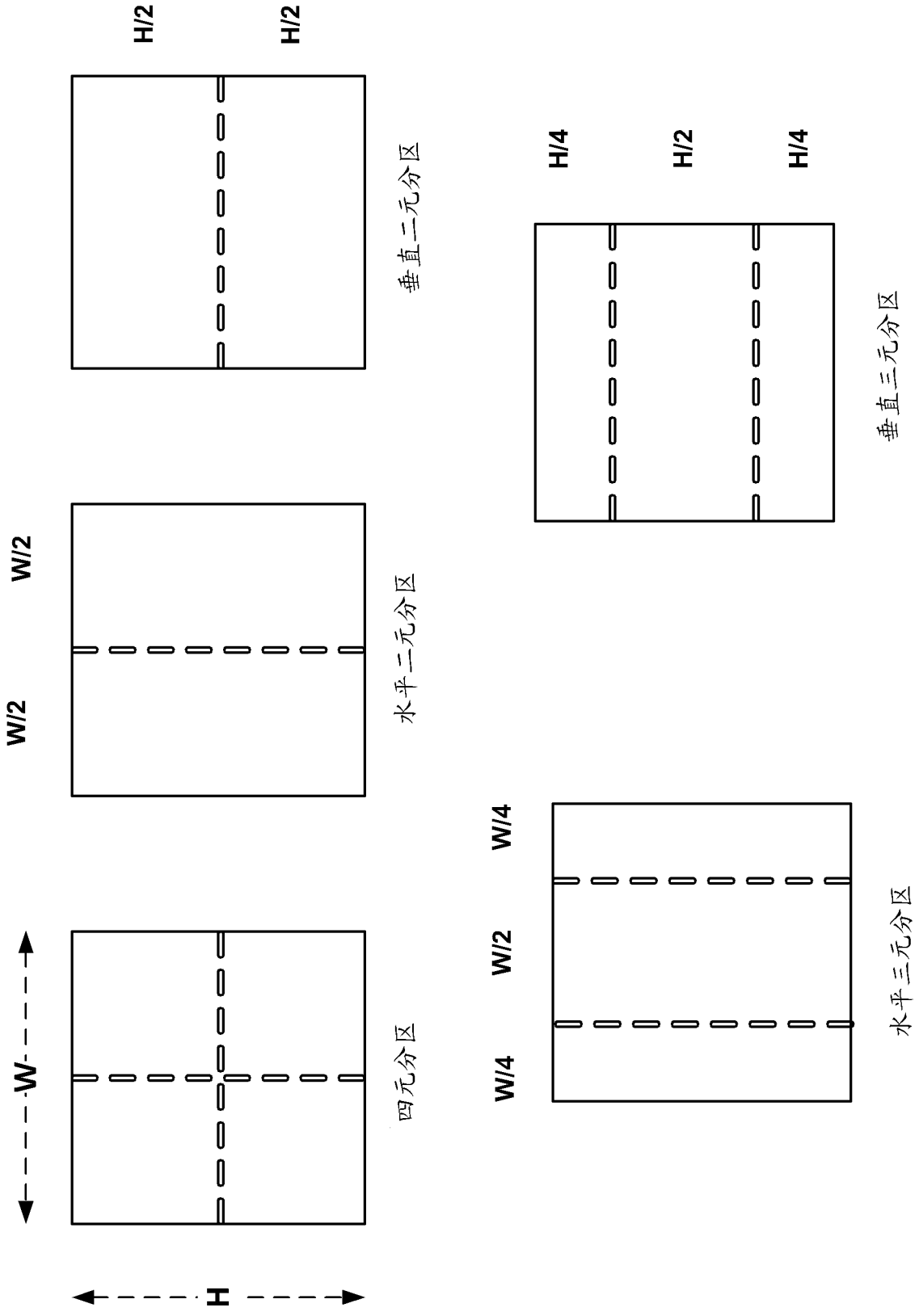

H/2
H/2
垂直二元分区
H/4
H/2
H/4
垂直三元分区
W/2
W/2
水平二元分区
W/4
W/2
W/4
水平三元分区
W
H
四元分区

图 4E

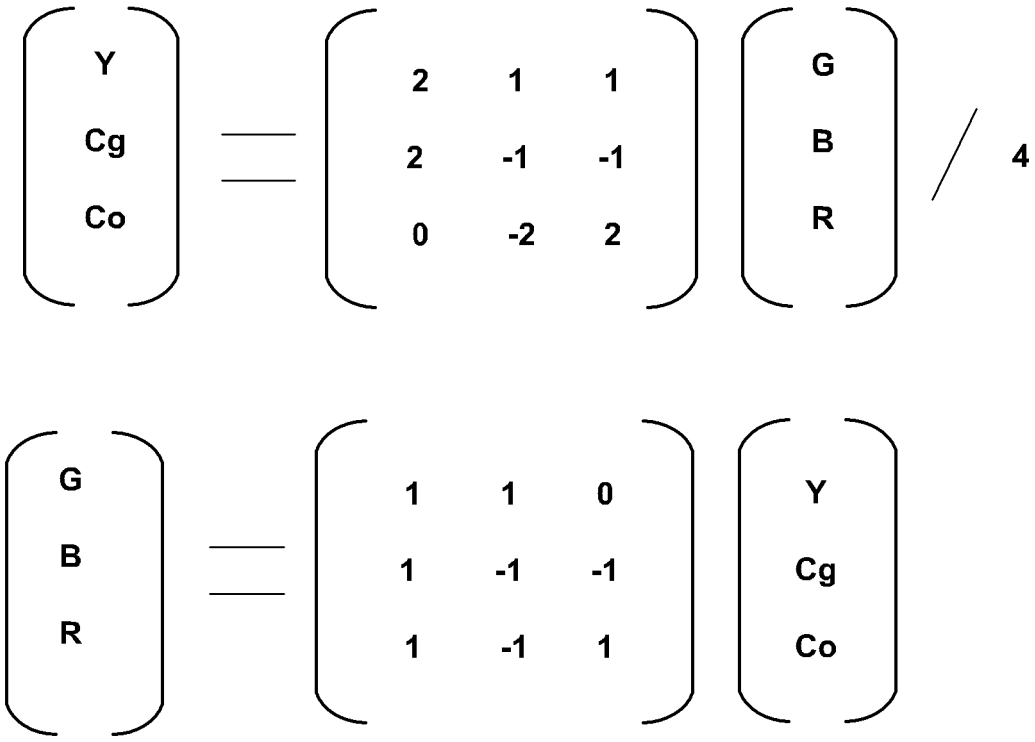


图 5A

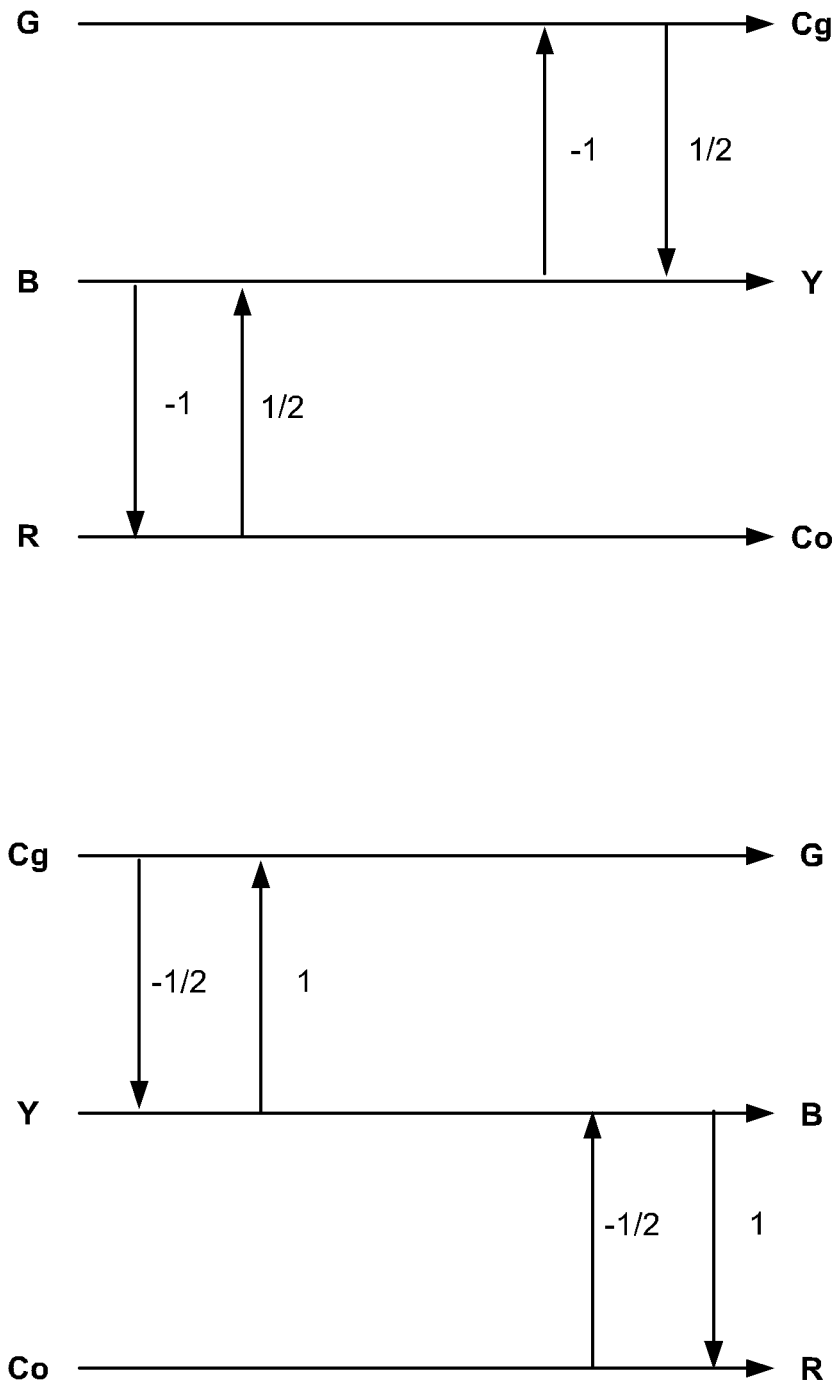


图 5B

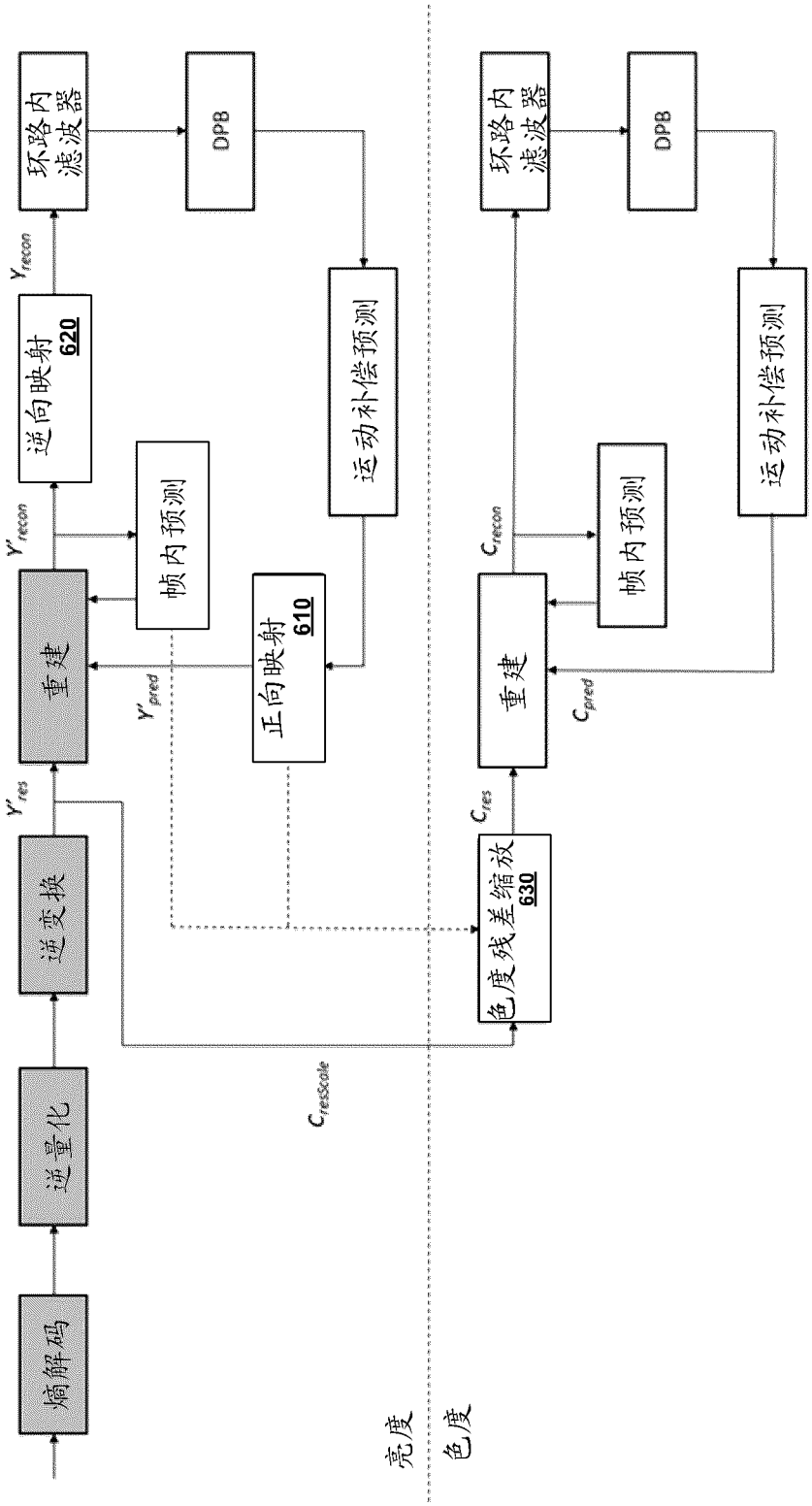

熵解码
逆量化
逆变换
重建
帧内预测
逆向映射 620
环路内滤波器
DPB
运动补偿预测
正向映射 610
色度残差缩放 630
亮度
色度

图 6

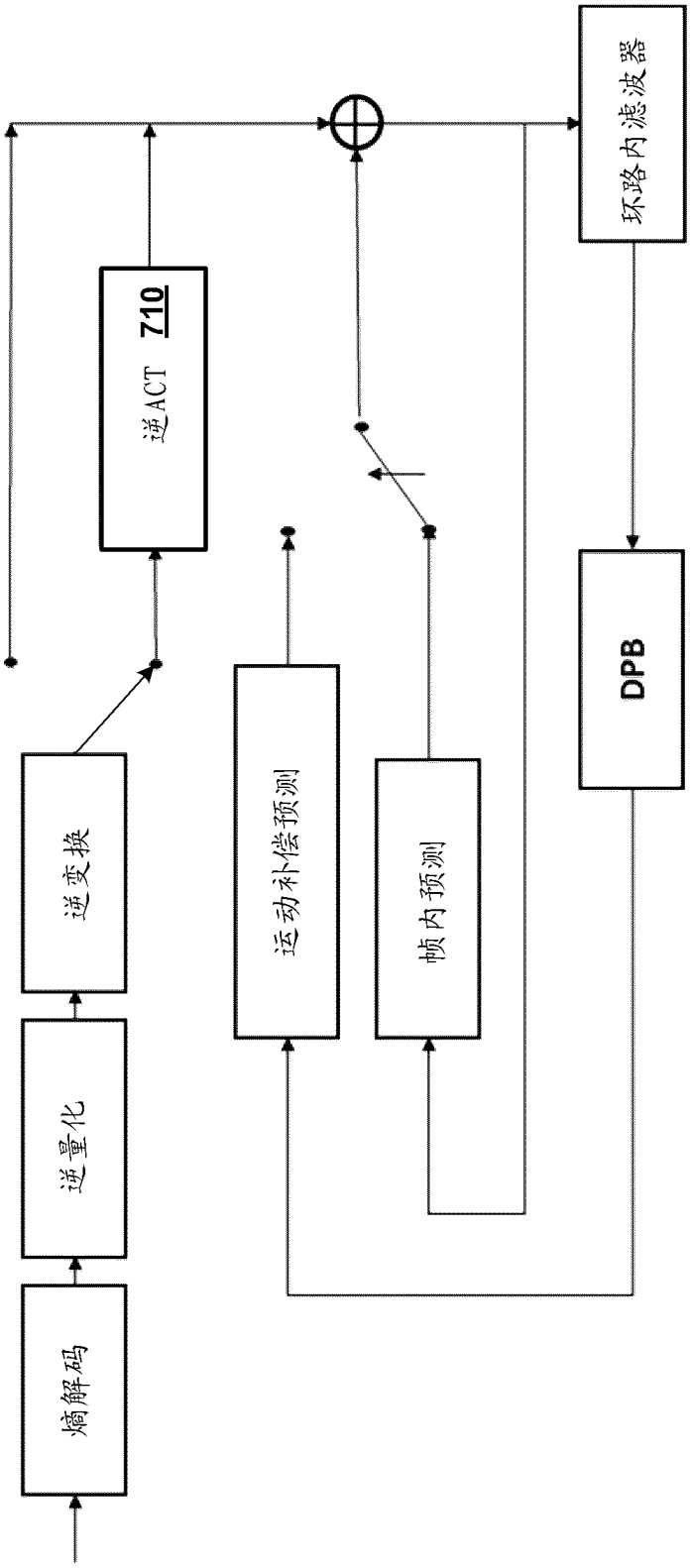


图 7

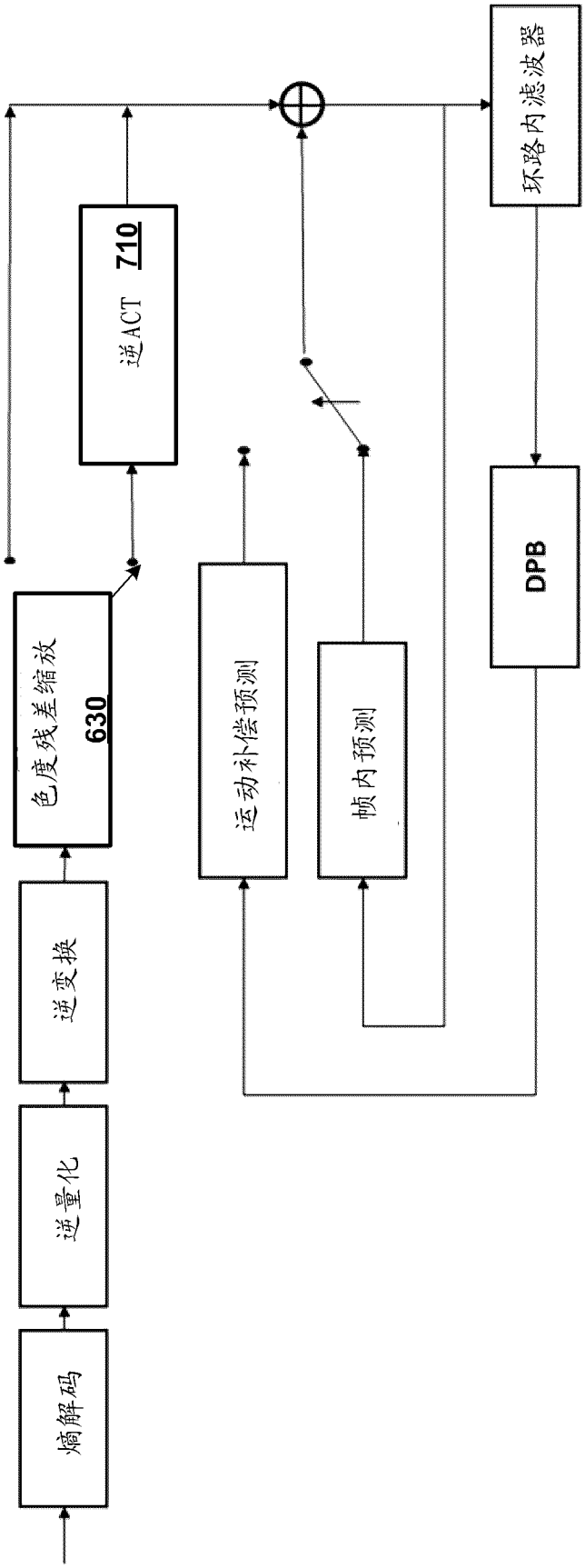


图 8A

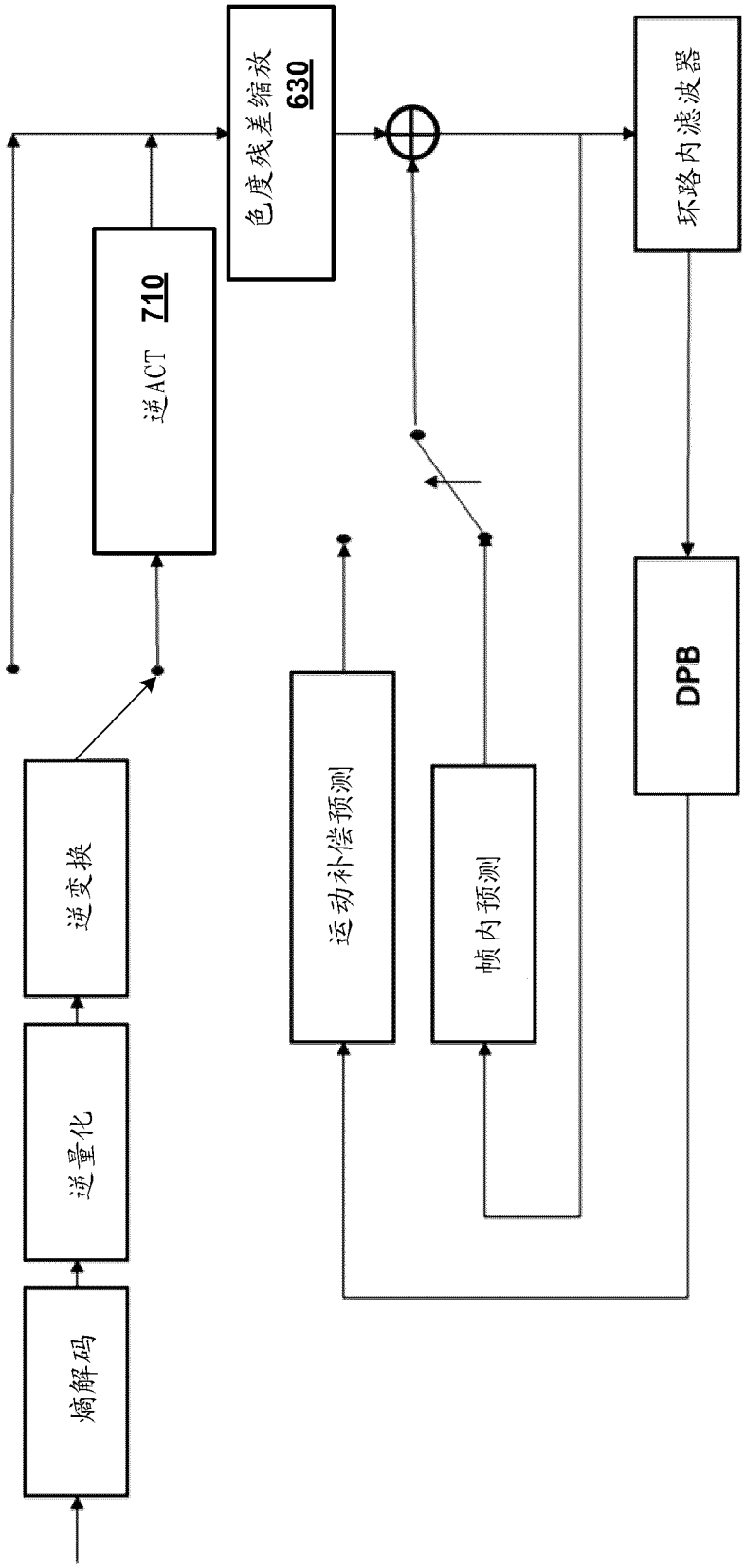


图 8B

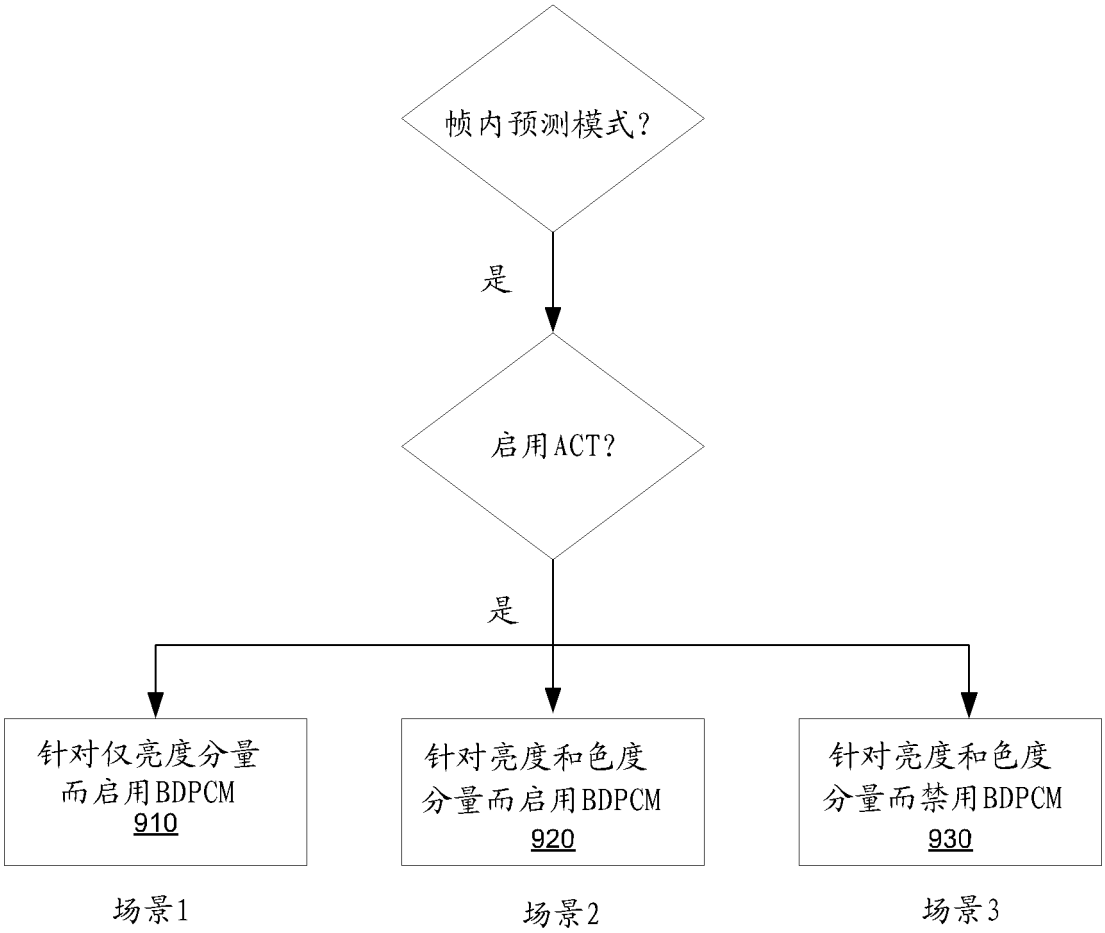

帧内预测模式?
是
启用ACT?
是
针对仅亮度分量而启用BDPCM
910
针对亮度和色度分量而启用BDPCM
920
针对亮度和色度分量而禁用BDPCM
930
场景1
场景2
场景3

图 9

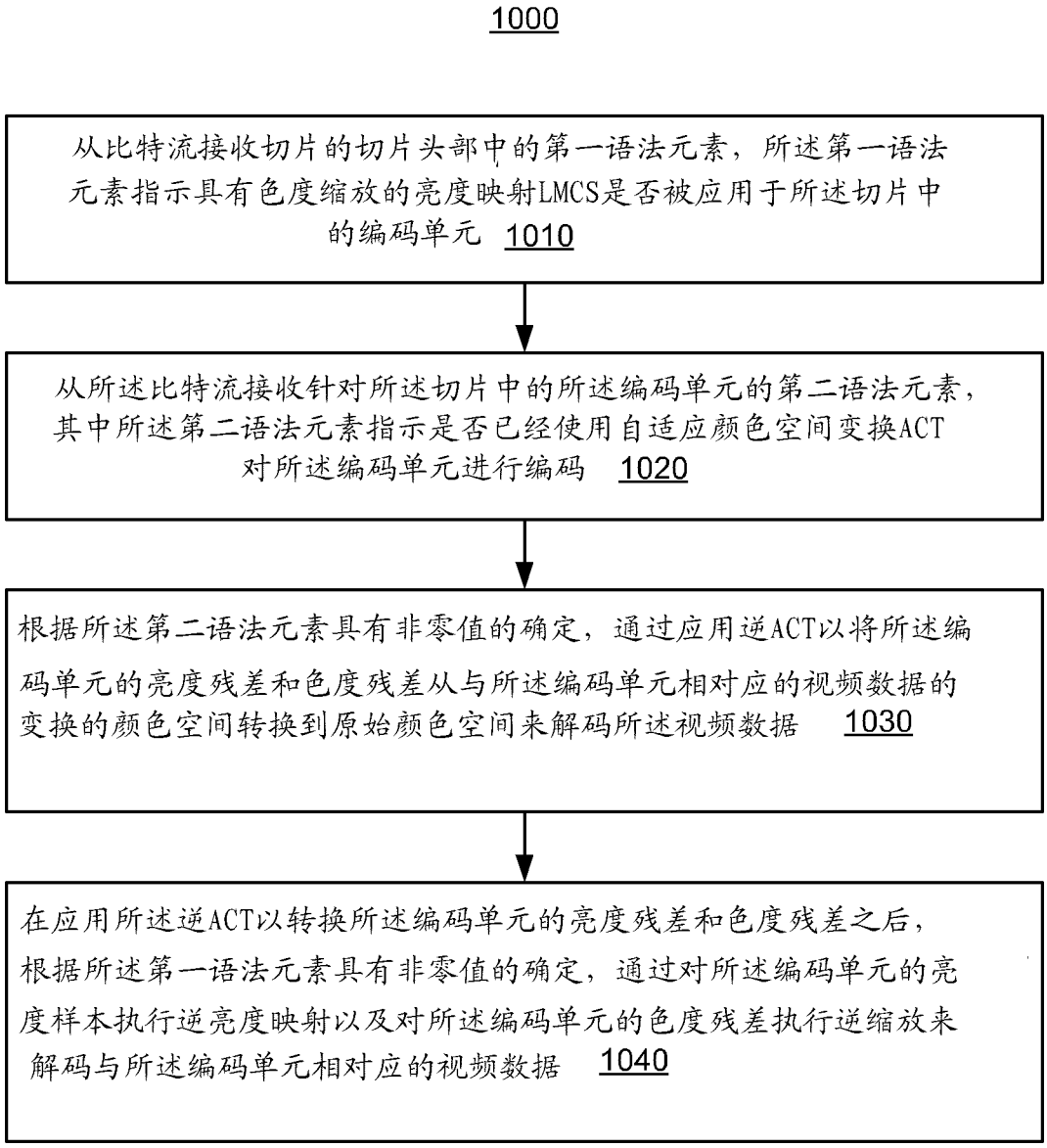

1000
从比特流接收切片的切片头部中的第一语法元素，所述第一语法元素指示具有色度缩放的亮度映射LMCS是否被应用于所述切片中的编码单元 1010
从所述比特流接收针对所述切片中的所述编码单元的第二语法元素，其中所述第二语法元素指示是否已经使用自适应颜色空间变换ACT对所述编码单元进行编码 1020
根据所述第二语法元素具有非零值的确定，通过应用逆ACT以将所述编码单元的亮度残差和色度残差从与所述编码单元相对应的视频数据的变换的颜色空间转换到原始颜色空间来解码所述视频数据 1030
在应用所述逆ACT以转换所述编码单元的亮度残差和色度残差之后，根据所述第一语法元素具有非零值的确定，通过对所述编码单元的亮度样本执行逆亮度映射以及对所述编码单元的色度残差执行逆缩放来解码与所述编码单元相对应的视频数据 1040

图 10