(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 109417627 A
(43)申请公布日 2019.03.01

(21)申请号 201780042282.3

(22)申请日 2017.07.12

(30)优先权数据
16305918.1 2016.07.15 EP

(85)PCT国际申请进入国家阶段日
2019.01.07

(86)PCT国际申请的申请数据
PCT/EP2017/067602 2017.07.12

(87)PCT国际申请的公布数据
WO2018/011295 EN 2018.01.18

(71)申请人 交互数字VC控股公司
地址 美国特拉华州

(72)发明人 S.拉瑟尔 S.普里 P.勒卡莱特

(74)专利代理机构 北京市柳沈律师事务所 11105
代理人 叶齐峰

(51)Int.Cl.
H04N 19/176(2006.01)
H04N 19/147(2006.01)
H04N 19/13(2006.01)
H04N 19/12(2006.01)
H04N 19/463(2006.01)
H04N 19/132(2006.01)
H04N 19/18(2006.01)

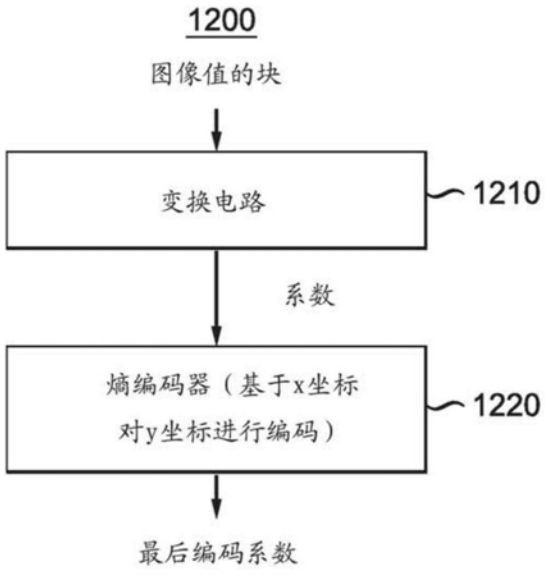
权利要求书2页 说明书8页 附图9页

(54)发明名称

用于最后系数编码的高级CABAC上下文自适应的方法和装置

(57)摘要

通过基于最后编码系数的x坐标值对最后编码系数的位置的y坐标进行编码来执行对最后编码系数位置的编码。这使得对最后编码系数参数的上下文自适应编码能够更加高效。在实施例中，使用部分变换来对图像值的块进行编码。部分变换使得对最后编码系数的编码能够更高效。



CN 109417627 A

1.一种用于对变换系数的块进行编码的方法,包括:

对图像值的块进行变换以获得变换系数;

对两个坐标进行熵编码,所述两个坐标定义通过变换系数的扫描确定的最后非零变换系数在块中的位置,

其中,所述两个坐标中的一个坐标(y)的熵编码取决于另一个坐标(x)的值。

2.如权利要求1所述的方法,其中,对所述一个坐标(y)进行熵编码使用取决于所述另一个坐标(x)的上下文。

3.如前述任一项权利要求所述的方法,还包括:

将所述两个坐标中的每一个分解为前缀和后缀;以及,

从x坐标推断y坐标的前缀的值。

4.如前述任一项权利要求所述的方法,其中,一组自适应正交变换包括用于变换图像值的块的部分变换。

5.一种用于对一组变换系数进行编码的装置,包括:

变换电路,其对图像值的块进行操作以获得变换系数;

熵编码器,其从所述变换系数生成表示最后编码系数的位置的代码,

其中,所述最后编码系数的位置由两个坐标(x,y)提供,其特征在于,y坐标的所述熵编码取决于x坐标的值。

6.如权利要求5所述的装置,其中,对所述一个坐标(y)进行熵编码使用取决于另一个坐标(x)的上下文。

7.如权利要求5或6所述的装置,其中,所述熵编码器将所述两个坐标中的每一个分解为前缀和后缀;以及从x坐标推断y坐标的前缀的值。

8.一种用于对一组变换系数进行解码的方法,包括:

对变换系数进行熵解码以生成最后编码系数的位置,

其中,最后编码系数的位置由两个坐标(x,y)提供,其特征在于,y坐标的所述熵解码取决于x坐标的值;

对变换系数进行逆变换以获得图像值的块。

9.如权利要求8所述的方法,其中,对所述一个坐标(y)进行熵编码使用取决于另一个坐标(x)的上下文。

10.如权利要求8或9所述的方法,还包括:

将所述两个坐标中的每一个分解为前缀和后缀;以及,

从x坐标推断y坐标的前缀的值。

11.一种用于对一组变换系数进行解码的装置,包括:

熵解码器,其根据变换系数对表示最后编码系数的位置的代码进行操作,

其中,最后编码系数的位置由两个坐标(x,y)提供,其特征在于,y坐标的熵编码取决于x坐标的值;

逆变换电路,其对变换系数进行操作以获得图像值的块。

12.如权利要求11所述的装置,其中,对所述一个坐标(y)进行熵编码使用取决于另一个坐标(x)的上下文。

13.如权利要求11或12所述的装置,其中,所述熵解码器将所述两个坐标中的每一个分

解为前缀和后缀;以及从x坐标推断y坐标的前缀的值。

14.一种非暂时性计算机可读存储介质,其上存储有根据权利要求8-10中任一项所述的方法对一组变换系数进行解码的指令或根据权利要求1-3中任一项所述的方法对一组变换系数进行编码的指令。

15.一种非暂时性计算机可读存储介质,其上存储有根据权利要求8-10中任一项所述的方法的比特流。

用于最后系数编码的高级CABAC上下文自适应的方法和装置

技术领域

[0001] 本原理通常涉及视频压缩和解压缩系统,更具体地涉及基于块的变换。

背景技术

[0002] 变换T将n个像素的块变换为n个变换系数。通过对变换系数应用逆变换T^{-1}以复原像素值,该过程是可逆的。

[0003] 在一些视频编码标准中,典型地从高频到低频,将变换系数放入与扫描顺序相关联的二维(2D)块中。如图1所示,在系数的量化之后,相对于高频到低频扫描顺序,第一个非零量化系数被称为最后编码系数。

[0004] 在与系数相关联的2D块拓扑中,对于大小为n=N×N的块,最后编码系数具有两个坐标(x,y),如图2所示。根据定义,0≤x,y≤N-1。这两个坐标被编码到比特流作为用以在解码器侧确定非零编码系数的信息。至少,这些坐标指示坐标(x,y)的系数不为零,并且相对于低到高扫描顺序,在该系数之后的系数都是零。

[0005] 可以添加指示变换系数的重要性的附加信息,以用信号通知剩余系数(图1中的问号标记)是否为零。

[0006] 传统上,在过去几十年中开发的许多视频和图像编解码器中,固定变换,例如,诸如离散余弦变换或离散正弦变换,被应用于每个块的像素以获得变换系数。然后,例如,这些系数被量化器Q量化,以获得由诸如VLC、算术编码器或上下文自适应二进制算术编码(CABAC)的熵编码器编码的量化系数。

[0007] 期望尽可能高效地编码最后编码系数位置。

[0008] HEVC/H.265标准通过使用两个坐标(x,y)引入了对最后编码系数位置的编码。使用截断的一元码对每个坐标进行二进制化,然后使用具有基于上下文的通道自适应的CABAC对每个比特或HEVC术语中的“bin(二进制位)”进行编码。在HEVC中,两个坐标x和y被分别编码和解码。

发明内容

[0009] 本原理解决了现有技术的这些和其它不足和缺点,本原理针对用于最后系数编码的高级CABAC上下文自适应的方法和装置。

[0010] 根据本原理的一个方面,提供了一种用于对一组变换系数进行编码的方法,包括:对图像值的块进行变换以获得变换系数的步骤;以及对最后编码系数的位置进行熵编码的步骤,其中,最后编码系数的位置由两个坐标提供,使得y坐标的熵编码取决于x坐标的值。

[0011] 根据本原理的另一方面,提供了一种用于对一组变换系数进行编码的装置,包括:变换电路,其对图像值的块进行操作以获得变换系数;以及熵编码器,其中,变换系数的最后编码系数的位置由两个坐标提供,使得y坐标的熵编码取决于x坐标的值。

[0012] 根据本原理的另一方面,提供了一种用于对一组变换系数进行解码的方法。该方法包括对变换系数进行熵解码以生成最后编码系数的位置,其中最后编码系数的位置由两

个坐标(x,y)提供,其特征在于,y坐标的熵解码取决于x坐标的值;以及对变换系数进行逆变换以获得图像值的块的步骤。

[0013] 根据本原理的另一方面,提供了一种用于对一组变换系数进行解码的装置。该装置包括熵解码器,其对表示来自变换系数的最后编码系数的位置的代码进行操作,其中最后编码系数的位置由两个坐标(x,y)提供,其特征在于,y坐标的熵编码取决于x坐标的值。该装置还包括对变换系数进行操作以获得图像值的块的逆变换电路。

[0014] 根据本原理的另一方面,提供了一种非暂时性计算机可读存储介质,其上存储有用于对一组变换系数进行解码的指令,使得最后编码系数的位置由两个坐标(x,y)提供,其特征在于,y坐标的熵编码取决于x坐标的值。

[0015] 根据本原理的另一方面,提供了一种非暂时性计算机可读存储介质,其上存储有用于对一组变换系数进行解码的比特流,使得最后编码系数的位置由两个坐标(x,y)提供,其特征在于,y坐标的熵编码取决于x坐标的值。

[0016] 从以下结合附图阅读的示例性实施例的详细描述,本原理的这些和其他方面、特征和优点将变得明显。

附图说明

[0017] 图1示出了变换单元的扫描顺序。

[0018] 图2示出了最后编码系数的坐标的示例。

[0019] 图3示出了具有12个系数的4×4变换单元中的编码的示例。

[0020] 图4示出了坐标值的后缀和前缀确定。

[0021] 图5示出了即时学习方案。

[0022] 图6示出了BD速率增益与各种场景的变换矢量的数量。

[0023] 图7示出了具有部分变换的8×8块的编码。

[0024] 图8示出了上下文值的结构。

[0025] 图9示出了上下文值的演变。

[0026] 图10示出了取决于相邻通道的上下文选择的示例。

[0027] 图11示出了使用本原理对一组变换系数进行编码的方法的一个实施例。

[0028] 图12示出了使用本原理对一组变换系数进行编码的装置的一个实施例。

[0029] 图13示出了使用本原理对一组变换系数进行解码的方法的一个实施例。

[0030] 图14示出了使用本原理对一组变换系数进行解码的装置的一个实施例。

具体实施方式

[0031] 通过以下实施例解决的技术问题是降低在已经应用了2D变换的像素的变换块中对最后编码系数的位置进行编码的成本。

[0032] 这些实施例是视频编码标准中常用的熵编码方案的改进。一种这样的视频编码标准是HEVC/H.265标准,但是实施例不限于该标准。

[0033] 本原理的主要思想是选择用于编码取决于值x的坐标y的上下文。这显然是可解码的,因为首先解码x,然后是y。这导致更好的编码性能,因为现有方法没有考虑x和y之间的依赖性。例如,相对于在x=0且y>0的情况下具有更多系数的块,所描述的构思特别有助于

编码仅具有一个DC系数(x＝y＝0)的块。简而言之,y通道统计被更好地建模。

[0034] 特别地,如果使用部分变换,则隐含地考虑了并非所有坐标对(x,y)都是可接受的事实,使得(x,y)的编码更高效。在具有部分变换的该特定实施例中,已经使用部分变换对修改的HEVC标准进行了测试,并且已经表明(x,y)的改进编码导致约-0.5%的压缩增益。

[0035] 这里呈现的构思提出推断用于编码取决于值x的y的bins的上下文。

[0036] 例如,如图3所示,考虑4x4的TU,其最后编码系数位置为c_{11}。在这种情况下,x＝2且y＝2。

[0037] 一旦x按照HEVC被编码,y的上下文将取决于x。在上面的示例中,使用一元码将值y＝2二进制化为001,并且使用专用上下文C_0,C_1和C_2对三个bins中的每一个进行编码。在HEVC中,上下文不取决于x,使得C_0表示y为零的概率P(y＝0)。在本原理下,该上下文被替换为取决于x的$C_{0,x=2}$,使得它表示概率P(y＝0|x＝2)。

[0038] 这需要增加由编解码器使用的上下文数量。在一个变体中,这些增加可以通过推断y的上下文,不是通过x的精确值而是通过x的值的范围来限制。例如,取决于x是否为零来选择$C_{i,x=0}$。而且,这种推断对于有限数量的二进制位可以是高效的,例如仅仅是前几个索引i。

[0039] 在HEVC中,如图4所示,坐标x或y被分解为前缀和后缀。

[0040] 例如,如果坐标值是14,则前缀是7,并且后缀用2比特进行编码。后缀是从坐标值中减去第一个值后的余数,14-12＝2。如果坐标值为3,则前缀为3,但没有后缀。

[0041] 使用截断的一元码对前缀进行二进制化。基于提供用于前缀值的上限的块大小的知识来执行截断。使用CABAC和专用上下文对二进制化的前缀的每个比特进行编码。

[0042] 例如,在4×4块中,可能的前缀是0,1,2或3,它们分别被二进制化为1,01,001和000(截断的)。在另一示例中,在8×8块中,可能的前缀是0到5,并被二进制化为1,01,001,0001,00001和00000(截断的)。

[0043] 使用固定长度编码对后缀进行二进制化,并且在没有上下文的情况下,在旁路模式下使用CABAC来对固定长度的后缀进行编码。

[0044] 例如,如果坐标值在16×16的块中为14,则前缀为7(二进制化为0000000),后缀为2(二进制化为10并编码为2比特)。

[0045] 如同在HEVC中,保留前缀和后缀的二进制化过程,仅对前缀使用上下文。本原理的主要特征在于,y的上下文取决于x的值。当然,这仅适用于y的前缀,因为后缀是在没有使用任何上下文的情况下进行编码的(而是使用旁路的CABAC)。没有对后缀进行任何改变。

[0046] 通常,变换T将n个像素的块变换为m＝n个变换系数。通过对变换系数应用逆变换T^{-1}以复原像素值,该过程是可逆的。在部分变换P的情况下,n个像素被变换为较少的(m<n)个变换系数。等效地,可以假设缺少的m-n个系数被设置为零。对变换系数应用“逆”变换P’(当然不是数学逆,因为部分变换是不可逆的)以获得初始像素值的近似。典型地,部分变换系数表示像素块的低频信息。

[0047] 如果使用部分变换,则已知某些m-n个变换系数必然为零,因此对最后编码系数的位置和坐标(x,y)施加一些约束。在这种情况下,通过隐式地使用这些约束,示出了在使用部分变换将TU像素变换为变换系数的情况下,该实施例自动处理两个坐标(x,y)的高效编码。

[0048] 替代诸如DCT或DST的系统变换，而是可以使用一组自适应的正交变换，使用不同的分类和变换优化方案在大训练集上离线学习该正交变换。将这组变换馈送到编解码器，并且在率失真优化(RDO)循环中选择组中的最佳变换。更加适应性的手段是对序列的特定帧内帧学习一组正交变换。这在本说明书的其余部分中被称为基于即时块的变换学习方案。该方案如图5所示。

[0049] 图5的框图示出了典型的即时方案，其中算法被分解为两部分，即视频/图像编解码器内的残差块的分类和新的变换组的生成。第一步将残差块分类为K个不同的类($S_1 \ldots S_K$)。在第二步中，使用针对特定类的重构误差的最小化来获得针对每个类的新变换。通常，奇异值分解(SVD)和(Karhunen-Loève变换)KLT被用于生成一组正交变换。迭代这两个步骤直到达到解的收敛或停止准则。如框图所见，系统的输入是帧内帧或图像以及一些初始的不可分离的正交变换组(T1...TK)。系统输出一组学习变换(T'1…T'K)以及需要被编码到发送到解码器的比特流中的语法信息。通常，与编码帧所需的比特相比，编码这些变换基矢量所需的开销比特相当大。

[0050] 由于SVD的能量压缩特性，观察到通过推导学习变换的不完整表示可以显著降低开销成本，其中只有前'm'个矢量被传输到解码器而剩余的(n-m)个变换矢量不是使用类似于Gram-Schmidt方法的完整算法生成就是被强制为零，从而导致部分变换。

[0051] 为了说明丢弃变换的最后几个矢量对Bjontegaard失真率(BD率)(Bjontegaard Distortion rate)的影响，在4K序列'PeopleOnStreet'和'Traffic'上学习了四个大小为64×64的不可分离的优化的变换。对这些序列执行编码测试，测试中保留前'm'个矢量，然后使用完整算法完成其余的基矢量。图6示出了性能增益相对于编码基矢量的数量的变化。垂直轴是相对于锚(HEVC测试软件HM15.0)的在不考虑变换成本的情况下的增益百分比。

[0052] 在图2中观察到，通过仅保留变换矢量的前一半，即m＝32，就BD率而言，性能下降可忽略不计。对于m＝16的情况，即当仅编码前16个基矢量时，与通过编码所有'n'个基矢量获得的总比特率相比，性能下降1％，但是变换矢量的开销成本降低至总开销的四分之一。对于m＝8，就BD率性能而言，有显著的性能损失，但也进一步降低了开销。总之，这表明在性能损失和开销成本之间存在折衷。

[0053] 由于当仅编码前'm'个变换矢量时，就BD率而言的性能下降取决于视频内容，因此有必要用内容自适应方法来估计'm'的最优值。直观地，在低比特率下，大多数系数被量化为零，在高比特率下，系数的能量即使在高频率下也是显著的。因此，'m'的值取决于内容，也取决于量化参数QP。

[0054] 平均的残差信号能量更多地集中在前几个系数中，其中DC系数平均具有最大能量，并且随着我们增高频率系数而降低。因此，大多数高频系数被量化为零。可以应用简单的基于阈值的方法来计算'm'的最佳值，该最佳值需要作为开销与帧一起被编码。

[0055] 设E是DCT系数的能量之和。阈值t被定义为参数p与E的乘积，获得，

[0056] $t = p \cdot E$

[0057] 'm'的值可以简单地从平均能量大于该阈值的系数的数量中计算。'p'的值可以通过实验找到。就被编码的矢量的总数的百分比而言，表1示出了对于所选择的'p'的值，超过该阈值的矢量的数量'm'的变化。从表1中观察到，与低QP所需的相比，在高QP下所需的矢量的平均数量要少得多。此外，矢量的数量'm'也随内容而变化。

[0058] 表1:针对不同序列的编码的矢量的数量和阈值

[0059]

QP	'p'	编码的矢量的总数量的百分比(%)			
		PeopleOnStreet	Traffic	Nebuta	SteamLocomotive
22	0.01	0.23	0.24	0.30	0.32
	0.005	0.27	0.29	0.32	0.35
	0.001	0.45	0.48	0.46	0.46
27	0.01	0.20	0.22	0.28	0.31
	0.005	0.25	0.27	0.31	0.34
	0.001	0.38	0.39	0.38	0.43
32	0.01	0.18	0.19	0.26	0.28
	0.005	0.21	0.22	0.29	0.30
	0.001	0.29	0.30	0.39	0.36
37	0.01	0.14	0.15	0.23	0.23
	0.005	0.18	0.18	0.26	0.25
	0.001	0.23	0.23	0.30	0.30

[0060] 当使用部分变换应用本原理时,y前缀的范围取决于x并且y的上下文自动适应。

[0061] 再次考虑图3的示例,使用m=12的部分变换,使得最后的16-12=4总是为零。现在,让我们假设对于特定的TU,最后编码系数是c_9。那么,如果在HEVC中编码,则坐标(x,y)=(3,0)变为(000,0)。坐标x如HEVC中那样被编码。关于坐标y,使用CABAC上下文对二进制化的唯一比特0进行编码。

[0062] 在HEVC中,此上下文不取决于x的值。相反,遵循这里描述的原理,用于对y前缀的二进制位进行编码的上下文却取决于x的值。所以,在本示例中,当x=3时,因为在x=3的情况下强制使其他系数在最后一列为0的部分变换,针对y前缀唯一可能的值是0,且针对y的编码的第一个比特始终为0。

[0063] 因此,依赖于x=3的上下文看到只有零的通道,并且它适应通道的统计,使得在TU的一些编码之后,编码0的成本就比特率而言变得可忽略不计。这就是上下文如何隐式自适应以高效地编码y的第一比特,理想情况下,根本不应该对所述y的第一比特进行编码,因为可从隐式变换得知x=3推断出y=0。

[0064] 具有部分变换的实施例容易扩展到使用后缀的坐标y,如图7中对8×8TU中的示例,使用m=57的部分变换。前57个变换系数可以是非零的,但最后64-57=7个系数必须为零。让我们假设最后编码系数位置是55。这给出了坐标(x,y)=(5,6)。y的前缀为5,并且后缀为0。在HEVC中,后缀应该被编码为1比特,但在我们的情况下,我们知道后缀不能是1,因为在系数55正下方的系数58在这种情况下已知必然是零,所以没有对后缀进行编码并有一比特的编码的增益。

[0065] 在HEVC/H.265标准中,在算术编码器中已经提出了用于编码二进制数据的新工具,即上下文自适应二进制算术编码(或CABAC)。取值为0或1的二进制符号s以遵循概率p被编码为1以及概率1-p被编码为0。此概率是从上下文推导出来的,并且在每个符号编码之后进行适应。

[0066] 上下文值是8比特值,参见图8。起始比特表示最可能的符号(Most Probable Symbol)(或MPS),并且接下来的7个比特表示可由其推导出概率p的概率p'(或状态)。

[0067] 取决于编码符号是否等于MPS,遵循图9中描述的过程对上下文值进行更新。

[0068] 通过两个表进行演进:如果编码符号是MPS,为transIdxMPS,如果编码符号不是MPS,则为transIdxLPS,即,它是最小可能符号(Least Probable Symbol)(LPS)。表2提供了针对条目p′的这些表,其名称为pStateIdx。

[0069] 表2:上下文状态演进表

[0070] 表9-41状态转换表

[0071]

pStateIdx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
transIdxLps	0	0	1	2	2	4	4	5	6	7	8	9	9	11	11	12
transIdxMps	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
pStateIdx	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
transIdxLps	13	13	15	15	16	16	18	18	19	19	21	21	22	22	23	24
transIdxMps	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
pStateIdx	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
transIdxLps	24	25	26	26	27	27	28	29	29	30	30	30	31	32	32	33
transIdxMps	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
pStateIdx	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
transIdxLps	33	33	34	34	35	35	35	36	36	36	37	37	37	38	38	63
transIdxMps	49	50	51	52	53	54	55	56	57	58	59	60	61	62	62	63

[0072] 符号s为MPS的概率P_{MPS}在8比特上线性量化,从0到127。它从上下文值中推导出来

[0073] $P_{MPS}=(p'+64)/127=(pStateIdx+64)/127$

[0074] 并且取决于MPS的值,符号s为1的概率p显而易见地从P_{MPS}推导出。

[0075] $p=p_{MPS}$ 如果MPS=1,

[0076] $p=1-P_{MPS}$ 如果MPS=0。

[0077] 上下文自适应编码是一种允许编码动态地遵循符号所属通道的统计的强大的工具。而且,为了避免混淆统计以及失去该过程的好处,每个通道都应该有它自己的上下文。这已经导致在HEVC/H.265中许多上下文的广泛使用,其使用数百个上下文,以便为许多通道建模。例如,在使用上下文的所有通道中,有

[0078] • 运动矢量残差,

[0079] • TU编码标志,

[0080] • 最后显著系数位置,

[0081] • 编码组编码标志,

[0082] • 变换系数显著标志,

[0083] • 变换系数幅度(大于1和大于2)标志,

[0084] • SAO数据,

[0085] • 其他数据。

[0086] 所有这些上下文/通道也很大程度上取决于颜色通道(即,通道是亮度还是色度)、变换单元大小、变换系数的位置、相邻符号值和其他因素。

[0087] 作为示例,编码组(CG)编码标志取决于当前CG下方和右侧的CG编码标志是否为1来选择,如图10所示。

[0088] 总而言之,上下文选择取决于很多因素,因此有大量的上下文。当然,解码器必须

更新上下文值以与在编码器侧执行的相对应,以确保与编码器的同步和流的解析。

[0089] 这里描述的实施例提出对最后编码系数的y坐标的上下文选择添加依赖性,这种依赖性是所述x坐标的值。应注意,该上下文选择还取决于(如HEVC中标准化的)颜色通道(亮度或色度)、TU大小和扫描顺序。

[0090] 除了可以在比特流中用信号通知的部分变换的情况之外,没有实现所描述原理所需的特定语法。

[0091] 关于解码过程,这里描述的原理影响与最后编码系数位置和解码最后编码系数位置的过程相关联的上下文的数量。

[0092] 在具有部分变换的实施例中,y后缀的改进的二进制化和截断的一元编码也受到影响。

[0093] 除了本原理的上述特征之外,所描述的实施例还提供了y坐标的熵编码由CABAC和相关联的上下文执行,依赖性是通过x的值对上下文的选择的推断;y坐标的熵编码通过使用前缀和后缀来执行,依赖性是通过x的值对用于y的前缀进行编码的上下文的选择的推断,并且作为变型,变换是部分变换,且对y后缀的二进制化使用部分变换的大小m的知识来截断。

[0094] 所提出的构思在其呈现在视频流的语法中的意义上是规范的并且隐含了一种要应用的解码方法。因此,本构思可以在诸如HEVC的后继者的视频标准中实现。

[0095] 图11中示出了用于编码一组变换系数的方法1100的一个实施例。该方法在开始框1101处开始,并且控制前进到框1110,用于变换图像值的块以产生变换系数值。控制从框1110前进到框1120,以使用x坐标对最后编码系数进行熵编码以对y坐标进行编码。

[0096] 图12中示出了用于编码一组变换系数的装置1200的一个实施例。该装置包括变换电路1210,变换电路在其输入端口上接收图像值的块并在其输出端口上产生变换系数。变换电路1210的该输出端口与熵编码器1220的输入在信号上连接。熵编码器1220基于最后编码系数位置的x坐标对最后编码系数位置的y坐标进行编码,以产生最后编码系数值。

[0097] 图13中示出了用于解码一组变换系数的方法1300的实施例。该方法在开始框1301处开始,并且控制前进到框1310,以使用x坐标值对最后编码系数进行熵解码以解码y坐标值。控制从框1310前进到框1320,用于对系数进行逆变换以产生图像值。

[0098] 图14中示出了用于解码一组变换系数的装置1400的实施例。该装置包括熵解码器1410,熵解码器接收包括针对最后编码系数的编码的变换系数并基于x坐标对最后编码系数的y坐标进行解码。熵解码器1410的输出与逆变换电路1420的输入在信号上连接。逆变换电路1420在其输入端口上接收变换系数并对它们进行逆变换以在其输出端口上产生图像值的块。

[0099] 前述实施例可以在机顶盒(STB)、调制解调器、网关或执行视频编码或解码的其他设备中实现。

[0100] 可以通过使用专用硬件以及与适当软件相关联的能够运行软件的硬件来提供图中所示的各个元件的功能。当由处理器提供时,可以由单个专用处理器、由单个共享处理器或由其中某些可以被共享的多个单独处理器提供功能。此外,术语“处理器”或“控制器”的明确使用不应被解释为专指能够运行软件的硬件,并且可以隐含地包括但不限于数字信号处理器(“DSP”)硬件、用于存储软件的只读存储器(“ROM”)、随机存取存储器(“RAM”)和非易

失性存储器。

[0101] 还可以包括其他硬件、传统的和/或定制的。类似地，图中所示的任何开关仅是概念性的。它们的功能可以通过程序逻辑的操作、通过专用逻辑、通过程序控制和专用逻辑的交互，或甚至手动地执行，特定技术可由实施者选择，如从上下文中更具体地理解的。

[0102] 本说明书说明了本原理。因此，应当理解，本领域技术人员将能够设计出各种布置，这些布置虽然未在本文中明确描述或示出，但体现了本原理并且包括在其精神和范围内。

[0103] 本文所引用的所有示例和条件语言旨在用于教学目的，以帮助读者理解由本发明人提供的本原理和概念，以促进本领域，并且应被解释为不限于这些具体引用的示例和条件。

[0104] 此外，这里叙述本原理的原理、方面和实施例的所有陈述以及其具体示例旨在包含其结构和功能等同方式。另外，其旨在这些等同方式包括当前已知的等同方式以及将来开发的等同方式，即，开发的执行相同功能的任何元件，而不管结构如何。

[0105] 因此，例如，本领域技术人员将理解，这里给出的框图表示体现本原理的说明性电路的概念图。类似地，应当理解，任何流程图、流程示意图、状态转换图、伪代码等表示可以实质上在计算机可读介质中表示并且由计算机或处理器运行的各种过程，无论这样的计算机或处理器是否被明确显示。

[0106] 在权利要求中，表示为用于执行指定功能的部件的任何元件旨在包含执行该功能的任何方式，包括例如：a)执行该功能的电路元件的组合，或b)与用于运行软件以执行该功能的适当电路相结合的任何形式的软件，因此包括固件、微代码等。由这些权利要求限定的本原理在于，由各种所述装置提供的功能以权利要求所要求的方式组合和结合在一起。因此认为任何能够提供这些功能的部件都等同于这里所示的部件。

[0107] 说明书中对本原理的“一个实施例”或“实施例”以及其其他变型的引用意味着结合该实施例描述的特定特征、结构、特性等被包括在本原理的至少一个实施例中。因此，出现在在整个说明书中的各个地方的短语“在一个实施例中”或“在实施例中”以及任何其他变型的出现不一定都指代相同的实施例。

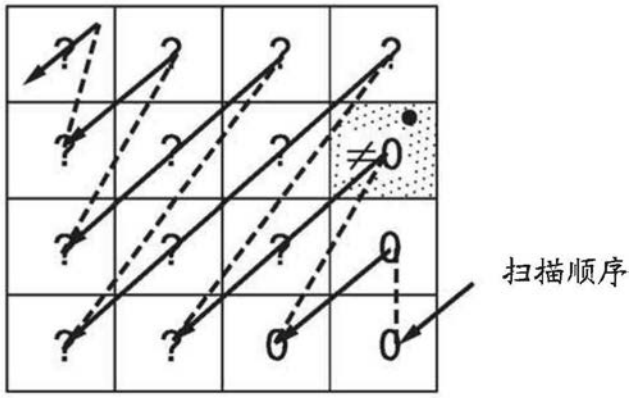

?
≠0
0
扫描顺序

图1

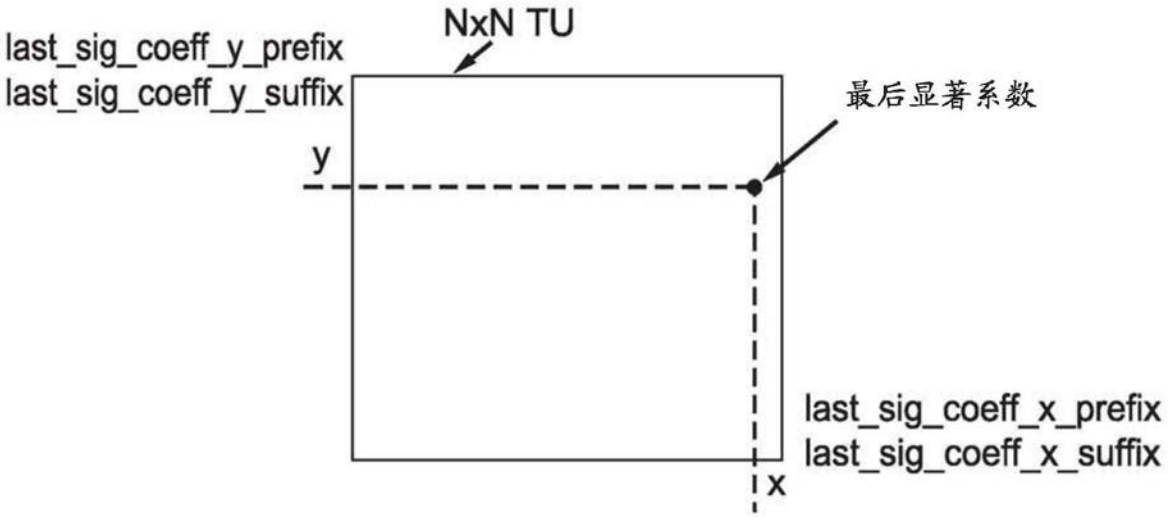

NxN TU
last_sig_coeff_y_prefix
last_sig_coeff_y_suffix
最后显著系数
y
last_sig_coeff_x_prefix
last_sig_coeff_x_suffix
x

图2

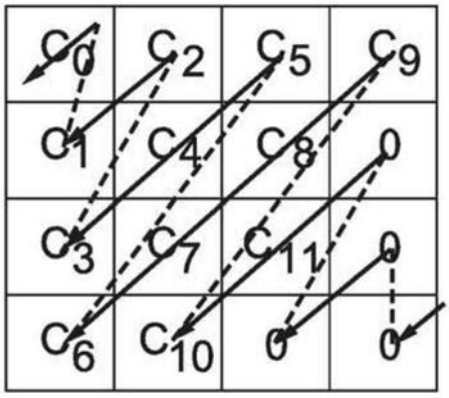

C0
C2
C5
C9
C1
C4
C8
0
C3
C7
C11
0
C6
C10
0
0

图3

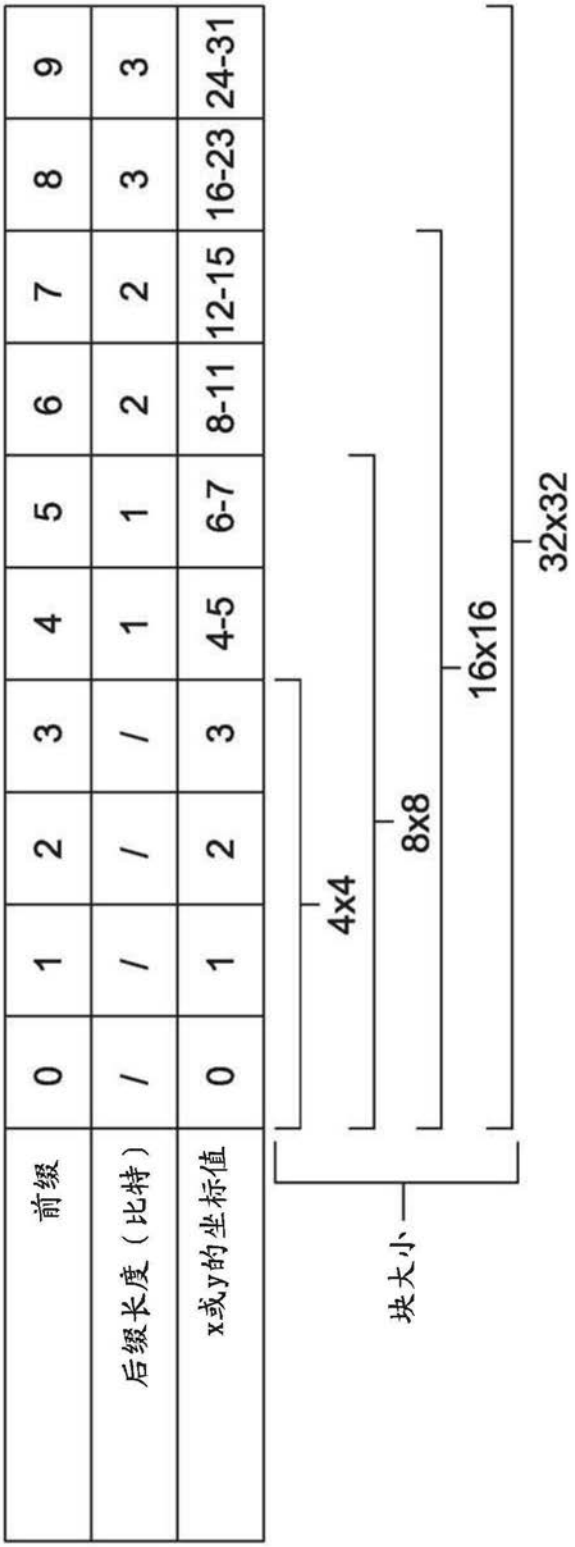

前缀
后缀长度（比特）
x或y的坐标值
0
/
0
1
/
1
2
/
2
3
/
3
4
1
4-5
5
1
6-7
6
2
8-11
7
2
12-15
8
3
16-23
9
3
24-31
4x4
8x8
16x16
32x32
块大小

图4

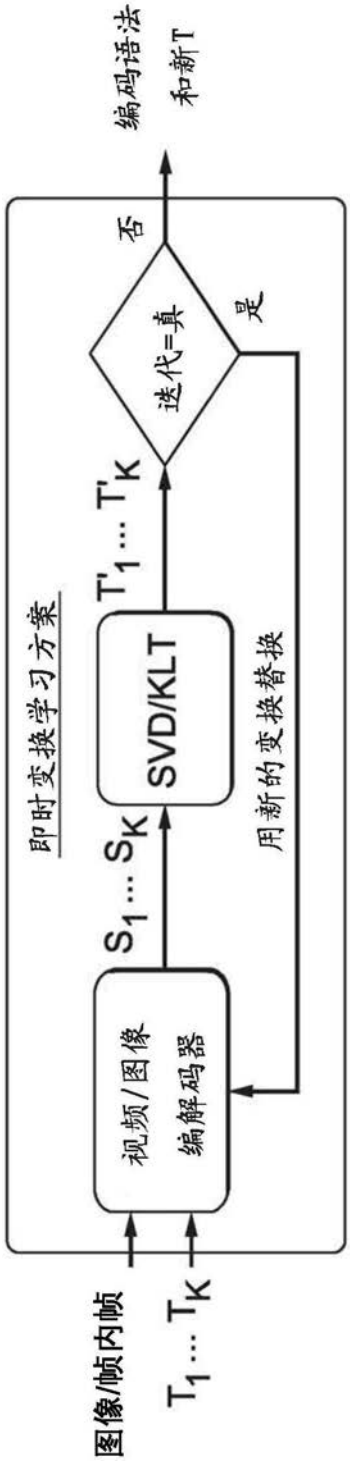

图像/帧内帧
$T_1 \dots T_K$
视频/图像
编解码器
$S_1 \dots S_K$
SVD/KLT
$T'_1 \dots T'_K$
迭代=真
否
是
编码语法
和新T
即时变换学习方案
用新的变换替换

图5

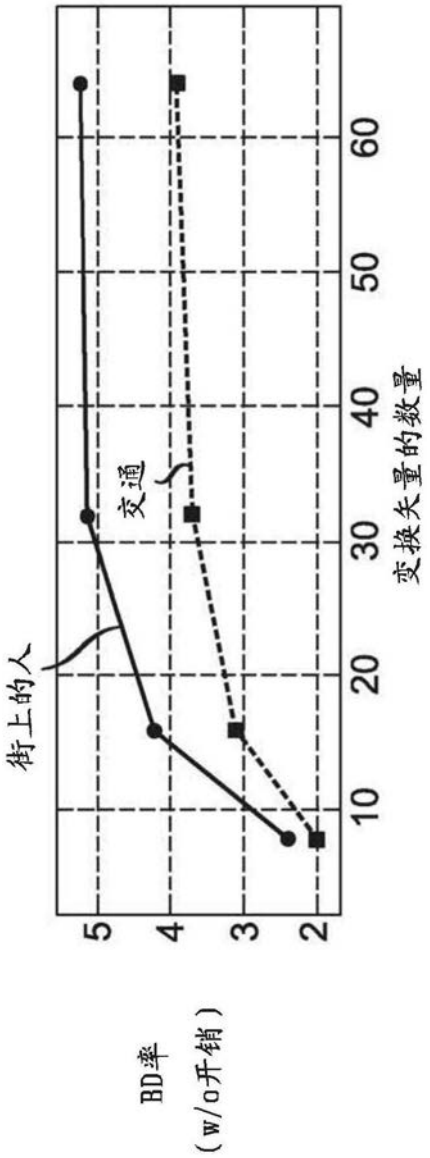


图6

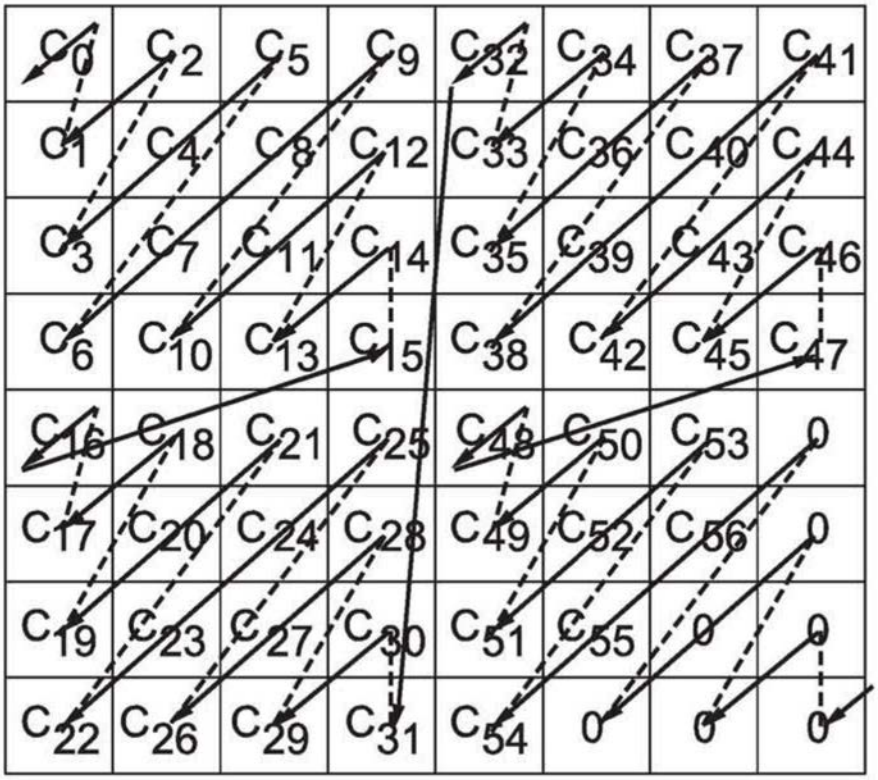

C0 C2 C5 C9 C32 C34 C37 C41
C1 C4 C8 C12 C33 C36 C40 C44
C3 C7 C11 C14 C35 C39 C43 C46
C6 C10 C13 C15 C38 C42 C45 C47
C16 C18 C21 C25 C48 C50 C53 0
C17 C20 C24 C28 C49 C52 C56 0
C19 C23 C27 C30 C51 C55 0 0
C22 C26 C29 C31 C54 0 0 0

图7

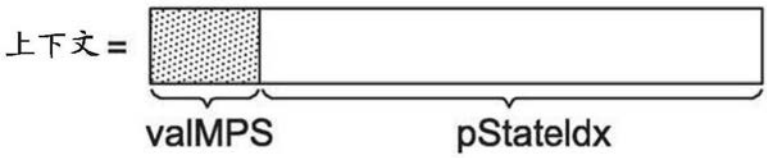

上下文=
valMPS
pStateIdx

图8

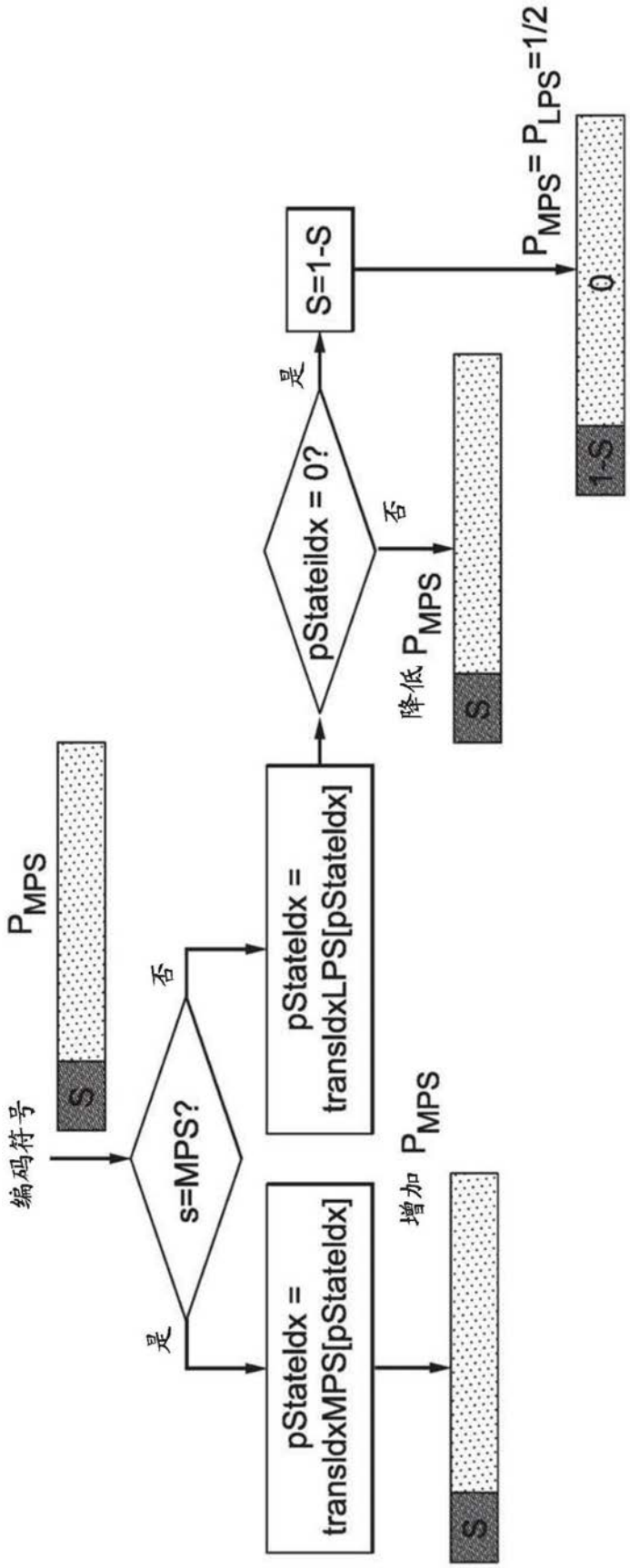

编码符号
s=MPS?
是
否
pStateIdx = transIdxMPS[pStateIdx]
pStateIdx = transIdxLPS[pStateIdx]
pStateIdx = 0?
是
否
S=1-S
增加 P_MPS
降低 P_MPS
P_MPS
P_MPS= P_LPS=1/2
S
1-S
0

图9

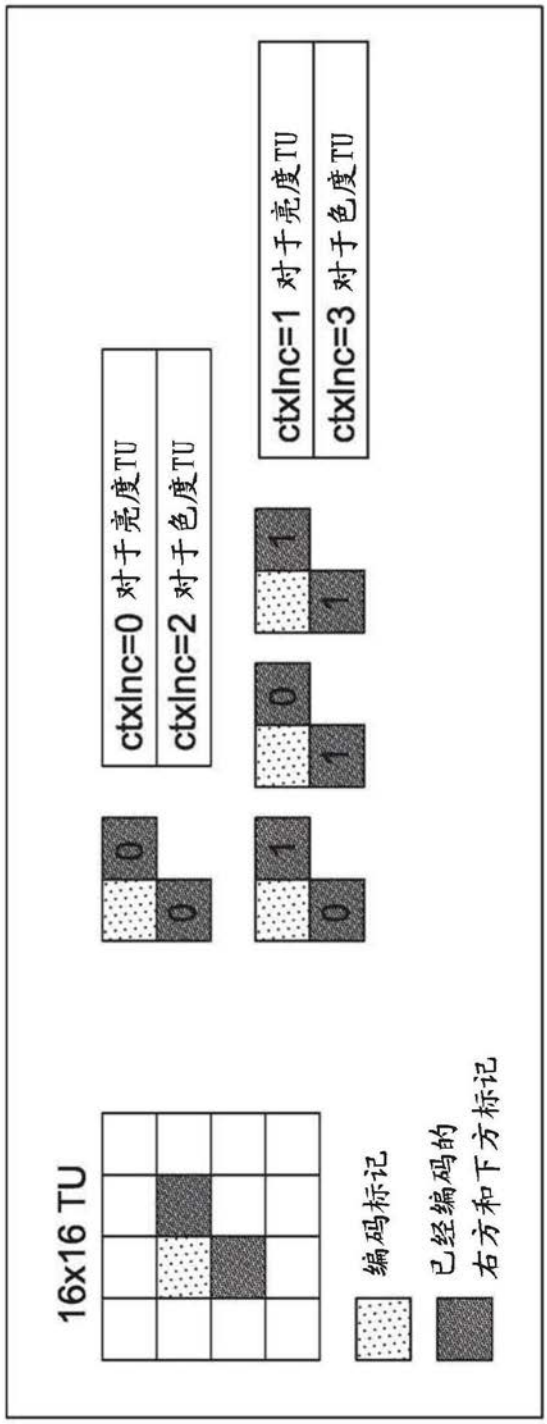

16x16 TU
编码标记
已经编码的
右方和下方标记
ctxInc=0 对于亮度TU
ctxInc=2 对于色度TU
ctxInc=1 对于亮度TU
ctxInc=3 对于色度TU
0
0
1
0
0
1
1
1

图10

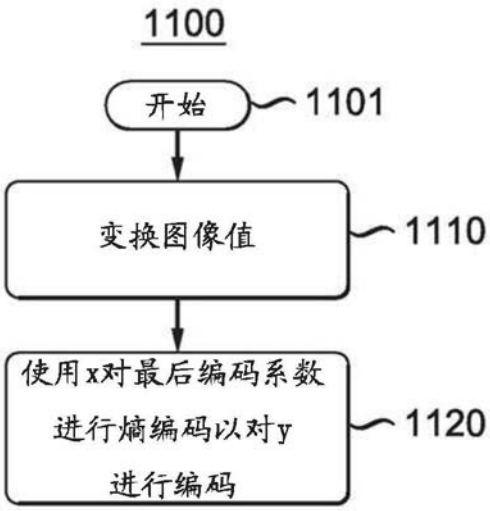

1100
开始
1101
变换图像值
1110
使用x对最后编码系数
进行熵编码以对y
进行编码
1120

图11

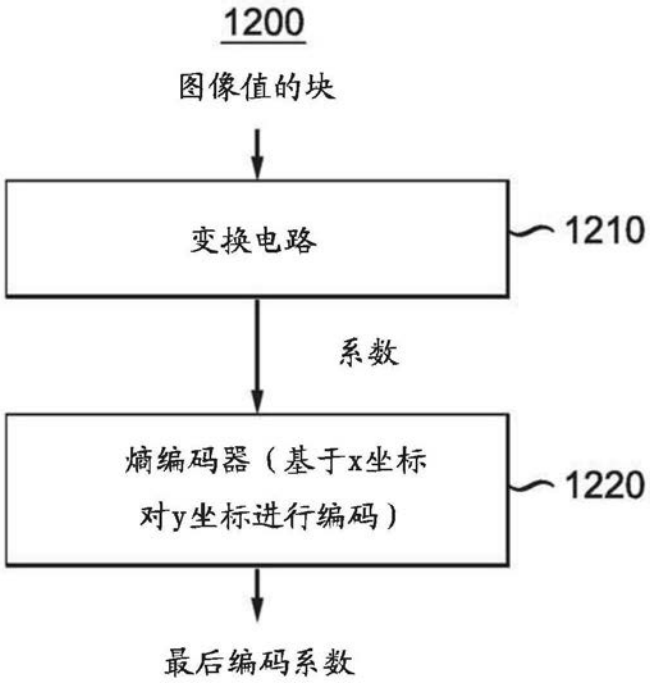

1200
图像值的块
变换电路
1210
系数
熵编码器（基于x坐标
对y坐标进行编码）
1220
最后编码系数

图12

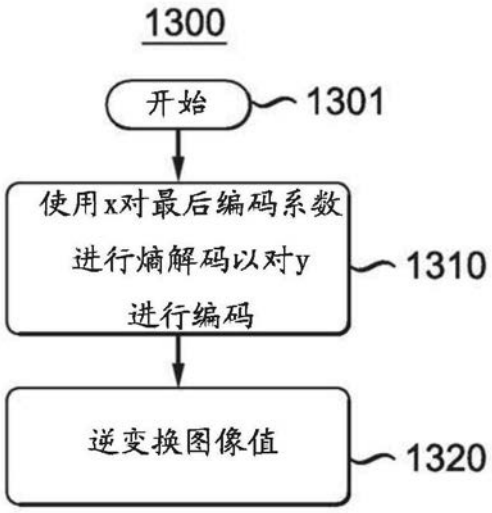

1300
开始
1301
使用x对最后编码系数进行熵解码以对y进行编码
1310
逆变换图像值
1320

图13

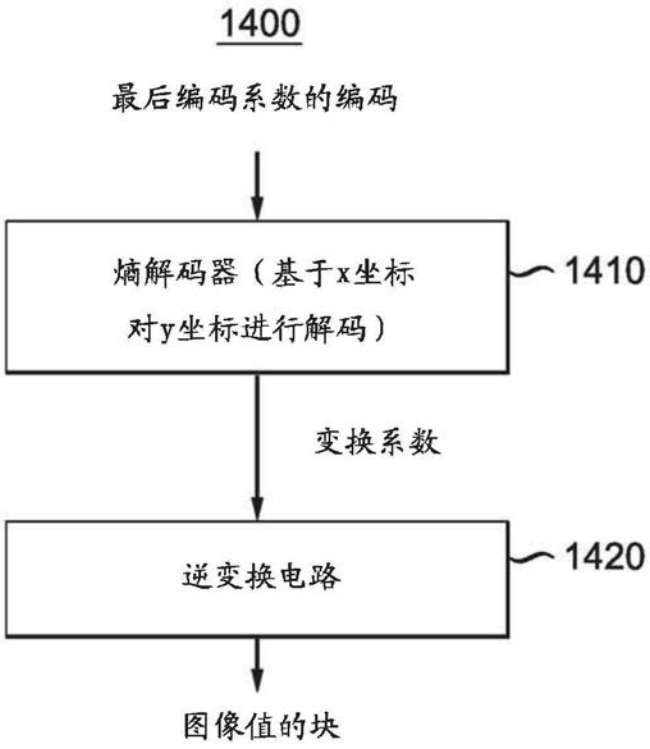

1400
最后编码系数的编码
熵解码器（基于x坐标对y坐标进行解码）
1410
变换系数
逆变换电路
1420
图像值的块

图14