



(21) 申请号 202080020348.0

(22) 申请日 2020.03.12

(65) 同一申请的已公布的文献号
申请公布号 CN 113632491 A

(43) 申请公布日 2021.11.09

(30) 优先权数据

62/817,475 2019.03.12 US

62/834,297 2019.04.15 US

62/871,519 2019.07.08 US

16/816,016 2020.03.11 US

(85) PCT国际申请进入国家阶段日
2021.09.10

(86) PCT国际申请的申请数据
PCT/US2020/022355 2020.03.12

(87) PCT国际申请的公布数据
W02020/186038 EN 2020.09.17

(73) 专利权人 高通股份有限公司

地址 美国加利福尼亚

(72) 发明人 董洁 A·赛义德 V·谢廖金
M·卡切夫维茨

(74) 专利代理机构 永新专利商标代理有限公司
72002

专利代理师 贾丽萍

(51) Int.Cl.

H04N 19/91 (2006.01)

H04N 19/70 (2006.01)

(56) 对比文件

CN 107333141 A, 2017.11.07

CN 107667530 A, 2018.02.06

US 2014092985 A1, 2014.04.03

US 9484952 B2, 2016.11.01

审查员 刘丽媛

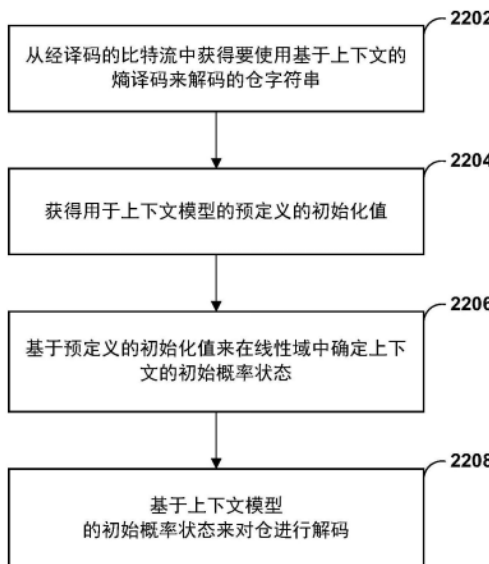
权利要求书4页 说明书32页 附图33页

(54) 发明名称

针对视频译码的概率初始化

(57) 摘要

一种用于对视频数据进行熵解码的示例方法包括：检索用于在对用于视频数据的可独立译码单元的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值；基于预定义的初始化值并且在线性域中确定上下文的初始概率状态；以及从比特流中并且基于上下文的初始概率状态来对用于语法元素的值的仓进行熵解码。



1. 一种用于对视频数据进行熵解码的方法,所述方法包括:

从存储器中检索用于在对用于视频数据的可独立译码单元的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值;

基于所述预定义的初始化值来确定斜率索引值和偏移索引值;

基于所述斜率索引值来确定参数m的值;

基于所述偏移索引值,根据以下等式来确定参数n的值: $n = (\text{OffsetIdx} * 18) + 1$,其中,OffsetIdx是所述偏移索引值,

获得与所述可独立译码单元相关联的量化参数的初始值;

基于所述预定义的初始化值、所述参数m、所述参数n以及所述量化参数的所述初始值,确定所述上下文的初始概率状态,

其中,确定所述初始概率状态还基于在与所述可独立译码单元相关联的所述量化参数的所述初始值与定义量化参数锚定点的非零值之间的差;以及

从视频比特流中并且基于所述上下文的所述初始概率状态来对用于所述语法元素的值的仓进行熵解码。

2. 根据权利要求1所述的方法,其中,确定所述上下文的所述初始概率状态包括:在线性域中确定所述上下文的所述初始概率状态,而不在中间在对数域中确定所述上下文的所述初始概率状态。

3. 根据权利要求1所述的方法,其中,所述量化参数的所述初始值为SliceQP_y。

4. 根据权利要求1所述的方法,其中,定义所述量化参数锚定点的值为16。

5. 根据权利要求1所述的方法,其中,确定所述初始概率状态包括根据以下等式来确定所述初始概率状态:

$$\text{InitProbState} = \text{clip3}(1, 127, ((m * (\text{SliceQP}_y - \text{QP}_{\text{anchor}})) \gg \text{rshift}) + n)$$

其中,InitProbState是所述初始概率状态,SliceQP_y是所述量化参数的所述初始值,QP_{anchor}是所述量化参数锚定点,并且rshift是右移位值。

6. 根据权利要求1所述的方法,其中,进行熵解码包括:使用上下文自适应二进制算术译码(CABAC)来进行熵解码。

7. 一种视频解码设备,包括:

存储视频数据的存储器;以及

一个或多个处理器,其在电路中实现并且被配置为:

从所述存储器中检索用于在对用于视频数据的可独立译码单元的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值;

基于所述预定义的初始化值来确定斜率索引值和偏移索引值;

基于所述斜率索引值来确定参数m的值;

基于所述偏移索引值,根据以下等式来确定参数n的值:

$$n = (\text{OffsetIdx} * 18) + 1, \text{其中, OffsetIdx是所述偏移索引值,}$$

获得与所述可独立译码单元相关联的量化参数的初始值;

基于所述预定义的初始化值、所述参数m、所述参数n以及所述量化参数的所述初始值,确定所述上下文的初始概率状态,

其中,为了确定所述初始概率状态,所述处理器还被配置为基于在与所述可独立译码

单元相关联的所述量化参数的所述初始值与定义量化参数锚定点的非零值之间的差来确定所述初始概率状态;以及

从视频比特流中并且基于所述上下文的所述初始概率状态来对用于所述语法元素的值的仓进行熵解码。

8. 根据权利要求7所述的设备,其中,为了确定所述上下文的所述初始概率状态,所述一个或多个处理器被配置为:在线性域中确定所述上下文的所述初始概率状态,而不在中间在对数域中确定所述上下文的所述初始概率状态。

9. 根据权利要求7所述的设备,其中,所述量化参数的所述初始值为SliceQP_Y。

10. 根据权利要求7所述的设备,其中,所述量化参数锚定点的值为16。

11. 根据权利要求7所述的设备,其中,为了确定所述初始概率状态,所述一个或多个处理器被配置为根据以下等式来确定所述初始概率状态:

$$\text{InitProbState} = \text{clip3}(1, 127, ((m * (\text{SliceQP}_Y - \text{QP}_{\text{anchor}})) \gg \text{rshift}) + n)$$

其中,InitProbState是所述初始概率状态,SliceQP_Y是所述量化参数的所述初始值,QP_{anchor}是所述量化参数锚定点,并且rshift是右移位值。

12. 根据权利要求7所述的设备,其中,为了对所述仓进行熵解码,所述一个或多个处理器被配置为:使用上下文自适应二进制算术译码(CABAC)来对所述仓进行解码。

13. 一种用于对视频数据进行熵编码的方法,所述方法包括:

从存储器中检索用于在对用于视频数据的可独立译码单元的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值;

基于所述预定义的初始化值来确定斜率索引值和偏移索引值;

基于所述斜率索引值来确定参数m的值;

基于所述偏移索引值,根据以下等式来确定参数n的值: $n = (\text{OffsetIdx} * 18) + 1$,其中,OffsetIdx是所述偏移索引值,

获得与所述可独立译码单元相关联的量化参数的初始值;

基于所述预定义的初始化值、所述参数m、所述参数n以及所述量化参数的所述初始值,确定所述上下文的初始概率状态,

其中,确定所述初始概率状态还基于在与所述可独立译码单元相关联的所述量化参数的所述初始值与定义量化参数锚定点的非零值之间的差;以及

在视频比特流中并且基于所述上下文的所述初始概率状态来对用于所述语法元素的值的仓进行熵编码。

14. 根据权利要求13所述的方法,其中,确定所述上下文的所述初始概率状态包括:在线性域中确定所述上下文的所述初始概率状态,而不在中间在对数域中确定所述上下文的所述初始概率状态。

15. 根据权利要求13所述的方法,其中,所述量化参数的所述初始值为SliceQP_Y。

16. 根据权利要求13所述的方法,其中,所述量化参数锚定点的值为16。

17. 根据权利要求13所述的方法,其中,确定所述初始概率状态包括根据以下等式来确定所述初始概率状态:

$$\text{InitProbState} = \text{clip3}(1, 127, ((m * (\text{SliceQP}_Y - \text{QP}_{\text{anchor}})) \gg \text{rshift}) + n)$$

其中,InitProbState是所述初始概率状态,SliceQP_Y是所述量化参数的所述初始值,

QP_{anchor} 是所述量化参数锚定点, 并且 $rshift$ 是右移位值。

18. 一种视频编码设备, 包括:

存储视频数据的存储器; 以及

一个或多个处理器, 其在电路中实现并且被配置为:

从所述存储器中检索用于在对用于视频数据的可独立译码单元的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值;

基于所述预定义的初始化值来确定斜率索引值和偏移索引值;

基于所述斜率索引值来确定参数 m 的值;

基于所述偏移索引值, 根据以下等式来确定参数 n 的值:

$n = (\text{OffsetIdx} * 18) + 1$, 其中, OffsetIdx 是所述偏移索引值,

获得与所述可独立译码单元相关联的量化参数的初始值;

基于所述预定义的初始化值、所述参数 m 、所述参数 n 以及所述量化参数的所述初始值, 确定所述上下文的初始概率状态,

其中, 为了确定所述初始概率状态, 所述处理器还被配置为基于在与所述可独立译码单元相关联的所述量化参数的所述初始值与定义量化参数锚定点的非零值之间的差来确定所述初始概率状态; 以及

在视频比特流中并且基于所述上下文的所述初始概率状态来对用于所述语法元素的值的仓进行熵编码。

19. 根据权利要求18所述的设备, 其中, 为了确定所述上下文的所述初始概率状态, 所述一个或多个处理器被配置为: 在线性域中确定所述上下文的所述初始概率状态, 而不在中间在对数域中确定所述上下文的所述初始概率状态。

20. 根据权利要求18所述的设备, 其中, 所述量化参数的所述初始值为 SliceQP_Y 。

21. 根据权利要求18所述的设备, 其中, 所述量化参数锚定点的值为16。

22. 根据权利要求18所述的设备, 其中, 为了确定所述初始概率状态, 所述一个或多个处理器被配置为根据以下等式来确定所述初始概率状态:

$\text{InitProbState} = \text{clip3}(1, 127, ((m * (\text{SliceQP}_Y - QP_{\text{anchor}})) \gg rshift) + n)$

其中, InitProbState 是所述初始概率状态, SliceQP_Y 是所述量化参数的所述初始值, QP_{anchor} 是所述量化参数锚定点, 并且 $rshift$ 是右移位值。

23. 一种设备, 包括:

用于获得用于在对用于视频数据的可独立译码单元的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值的单元;

用于基于所述预定义的初始化值来确定斜率索引值和偏移索引值的单元;

用于基于所述斜率索引值来确定参数 m 的值的单元;

用于基于所述偏移索引值, 根据以下等式来确定参数 n 的值的单元: $n = (\text{OffsetIdx} * 18) + 1$, 其中, OffsetIdx 是所述偏移索引值,

用于获得与所述可独立译码单元相关联的量化参数的初始值的单元;

用于基于所述预定义的初始化值、所述参数 m 、所述参数 n 以及所述量化参数的所述初始值, 确定所述上下文的初始概率状态的单元,

其中, 所述用于确定所述初始概率状态的单元包括用于基于在与所述可独立译码单元

相关联的所述量化参数的所述初始值与定义量化参数锚定点的非零值之间的差来确定所述初始概率状态的单元;以及

用于从比特流中并且基于所述上下文的所述初始概率状态来对用于所述语法元素的值的仓进行熵解码的单元。

24. 一种存储指令的非暂时性计算机可读存储介质,所述指令在被执行时使得视频译码器的一个或多个处理器进行以下操作:

获得用于在对用于视频数据的可独立译码单元的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值;

基于所述预定义的初始化值来确定斜率索引值和偏移索引值;

基于所述斜率索引值来确定参数m的值;

基于所述偏移索引值,根据以下等式来确定参数n的值: $n = (\text{OffsetIdx} * 18) + 1$,其中,OffsetIdx是所述偏移索引值,

获得与所述可独立译码单元相关联的量化参数的初始值;

基于所述预定义的初始化值、所述参数m、所述参数n以及所述量化参数的所述初始值,确定所述上下文的初始概率状态,

其中,为了确定所述初始概率状态,所述处理器还被配置为基于在与所述可独立译码单元相关联的所述量化参数的所述初始值与定义量化参数锚定点的非零值之间的差来确定所述初始概率状态;以及

从比特流中并且基于所述上下文的所述初始概率状态来对用于所述语法元素的值的仓进行熵编码。

针对视频译码的概率初始化

[0001] 本申请要求享受于2020年3月11日递交的美国申请No.16/816,016的优先权,该美国申请要求享受于2019年3月12日递交的美国临时申请No.62/817,475、于2019年4月15日递交的美国临时申请No.62/834,297、以及于2019年7月8日递交的美国临时申请No.62/871,519的权益,上述申请的每个申请的全部内容通过引用的方式被并入本文中。

技术领域

[0002] 本公开内容涉及视频编码和视频解码。

背景技术

[0003] 数字视频能力可以被合并到宽范围的设备中,包括数字电视机、数字直播系统、无线广播系统、个人数字助理(PDA)、膝上型计算机或台式计算机、平板计算机、电子书阅读器、数字相机、数字记录设备、数字媒体播放器、视频游戏设备、视频游戏控制台、蜂窝或卫星无线电话、所谓的“智能电话”、视频电话会议设备、视频流式传输设备等。数字视频设备实现视频译码技术(诸如在通过MPEG-2、MPEG-4、ITU-T H.263、ITU-T H.264/MPEG-4(第10部分,高级视频译码(AVC))、ITU-T H.265/高效率视频译码(HEVC)定义的标准以及这样的标准的扩展中描述的那些技术)。通过实现这样的视频译码技术,视频设备可以更高效地发送、接收、编码、解码和/或存储数字视频信息。

[0004] 视频译码技术包括空间(图片内(intra-picture))预测和/或时间(图片间(inter-picture))预测以减少或去除在视频序列中固有的冗余。对于基于块的视频译码,视频切片(例如,视频图片或视频图片的一部分)可以被分割为视频块,视频块还可以被称为译码树单元(CTU)、译码单元(CU)和/或译码节点。图片的经帧内译码(I)的切片中的视频块是使用相对于相同图片中的相邻块中的参考样本的空间预测来编码的。图片的经帧间译码(P或B)的切片中的视频块可以使用相对于相同图片中的相邻块中的参考样本的空间预测或者相对于其它参考图片中的参考样本的时间预测。图片可以被称为帧,以及参考图片可以被称为参考帧。

发明内容

[0005] 概括而言,本公开内容描述了针对用于视频压缩的算术译码的概率初始化的技术。例如,视频译码器可以设置在二进制算术译码器中使用的每个上下文模型的初始概率状态。该过程可以在熵编码或解码可独立解码单元(诸如切片或瓦片)的开始时调用。

[0006] 作为一个示例,一种用于对视频数据进行熵译码的方法包括:获得用于在对用于视频数据的可独立译码单元的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值;基于所述预定义的初始化值并且在线性域中确定所述上下文的初始概率状态;以及基于所述上下文的所述初始概率状态来对用于所述语法元素的值的仓进行熵译码。

[0007] 作为另一示例,一种设备包括:存储视频数据的存储器;以及一个或多个处理器,

其在电路中实现并且被配置为：获得用于在对用于视频数据的可独立译码单元的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值；基于所述预定义的初始化值并且在线性域中确定所述上下文的初始概率状态；以及基于所述上下文的所述初始概率状态来对用于所述语法元素的值的仓进行熵译码。

[0008] 作为另一示例，一种设备包括：用于获得用于在对用于视频数据的可独立译码单元的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值的单元；用于基于所述预定义的初始化值并且在线性域中确定所述上下文的初始概率状态的单元；以及用于基于所述上下文的所述初始概率状态来对用于所述语法元素的值的仓进行熵译码的单元。

[0009] 作为另一示例，一种计算机可读存储介质存储指令，所述指令在被执行时使得视频译码器的一个或多个处理器进行以下操作：获得用于在对用于视频数据的可独立译码单元的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值；基于所述预定义的初始化值并且在线性域中确定所述上下文的初始概率状态；以及基于所述上下文的所述初始概率状态来对用于所述语法元素的值的仓进行熵译码。

[0010] 在附图和下文的描述中阐述了本公开内容的一个或多个方面的细节。根据说明书和附图以及根据权利要求，在本公开内容中描述的技术的其它特征、目的和优势将是显而易见的。

附图说明

[0011] 图1是示出可以执行本公开内容的技术的示例视频编码和解码系统的框图。

[0012] 图2A和2B是示出二进制算术译码中的范围更新过程的概念图。

[0013] 图3是示出二进制算术译码中的输出过程的概念图。

[0014] 图4A和4B是示出示例四叉树二叉树 (QTBT) 结构和相应的译码树单元 (CTU) 的概念图。

[0015] 图5是示出可以执行本公开内容的技术的示例视频编码器的框图。

[0016] 图6是示出视频编码器中的上下文自适应二进制算术译码器的框图。

[0017] 图7是示出可以执行本公开内容的技术的示例视频解码器的框图。

[0018] 图8是示出视频解码器中的上下文自适应二进制算术译码器的框图。

[0019] 图9是示出如在HEVC中的用于表示为1的概率的概率状态的图。

[0020] 图10A和10B是示出如在VVC中的用于表示为1的概率的概率状态的图，其中图10A具有1024个概率状态 (10比特的精度)，并且图10B具有16384个概率状态 (14比特的精度)。

[0021] 图11A和11B是示出从InitProbState到可以在通用视频译码 (VVC) 的算术译码引擎中使用的概率状态的映射的图，其中，图11A具有1024个概率状态 (10比特的精度)，并且图11B具有16384个概率状态 (14比特的精度)。

[0022] 图12是示出在I切片中使用的语法元素SaoMergeFlag (1个仓) 的概率分布的先验知识的图。

[0023] 图13A是示出在图12中的由对数域中的概率状态表示的概率分布的先验知识的图，并且图13B是示出模型概率状态和具有线性函数的SliceQP_y的图。

[0024] 图14A是示出在图12中的由线性域中的概率状态表示的概率分布的先验知识的图;图14B是示出在不增加在HEVC和当前VVC中定义的斜率和偏移的范围的情况下的模型概率状态和SliceQP_y以及线性函数的图,而图14C是示出在增加斜率和偏移的范围的情况下的模型概率状态和SliceQP_y以及线性函数的图。

[0025] 图15A是示出在B切片中使用的语法元素SplitFlag的概率分布的先验知识的图;图15B是示出模型对数概率状态和SliceQP_y以及线性函数的图;图15C是示出模型线性概率状态和SliceQP_y以及使用在HEVC和当前VVC中定义的斜率和偏移的线性函数的图,而图15D是示出模型线性概率状态和SliceQP_y以及具有在SliceQP_y等于32处定义的交点的线性函数的图。

[0026] 图16A-16C是示出使用由多个交点确定的分段线性函数来拟合概率分布的图。

[0027] 图17A和17B是示出将二次域InitProbState映射到可以在VVC的算术译码引擎中使用的概率状态的示例的图,其中图17A具有1024个概率状态(10比特的精度),并且图17B具有16384个概率状态(14比特的精度)。

[0028] 图18A和18B是示出极端概率分布的示例的图。

[0029] 图19是示出用于对当前块进行编码的示例方法的流程图。

[0030] 图20是示出根据本公开内容的一种或多种技术的用于执行基于上下文的熵编码的示例过程的流程图。

[0031] 图21是示出用于对当前块进行解码的示例方法的流程图。

[0032] 图22是示出根据本公开内容的一种或多种技术的用于执行基于上下文的熵解码的示例过程的流程图。

具体实施方式

[0033] 概括而言,本公开内容描述了用于针对用于视频压缩的算术译码的概率初始化的技术。例如,视频译码器可以设置在二进制算术译码器中使用的每个上下文模型的初始概率状态。可以在对可独立解码单元(诸如切片)进行熵编码或解码的开始时调用该过程。

[0034] 为了执行初始化,视频译码器可以获得用于在对用于视频数据的切片的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的初始化值(例如,initValue)。视频译码器可以基于预定义的初始化值并且在对数域中确定上下文的初始概率状态(例如,pStateIdx)。视频译码器可以基于上下文的初始概率状态来对用于语法元素的值的仓进行熵译码。

[0035] 当根据一些视频标准对视频数据进行译码时,视频译码器可以在线性域中保持概率状态。例如,如下文进一步详细讨论的,在通用视频译码(VVC)中采用的上下文自适应二进制算术译码器(CABAC)可以在线性域中保持上下文的概率状态。在对数域中确定初始概率状态的情况下,视频译码器可以执行各种操作以将所确定的初始概率状态转换在线性域中。例如,视频译码器可以使用查找表(LUT)将初始概率状态从对数域转换到线性域。

[0036] 根据本公开内容的一种或多种技术,视频译码器可以直接在线性域中确定初始概率状态。例如,视频可以基于预定义的初始化值来在线性域中确定上下文的初始概率状态(例如,pStateIdx),而不使用LUT将初始概率状态从对数域转换到线性域。以这种方式,视频编码器可以避免必须存储和/或访问在线性域和对数域中的初始概率状态之间转换的

LUT。

[0037] 图1是示出可以执行本公开内容的技术的示例视频编码和解码系统100的框图。概括而言,本公开内容的技术涉及对视频数据进行译码(coding)(编码(encoding)和/或解码(decoding))。通常,视频数据包括用于处理视频的任何数据。因此,视频数据可以包括原始的未经译码的视频、经编码的视频、经解码(例如,经重构)的视频、以及视频元数据(例如,信令数据)。

[0038] 如图1所示,在该示例中,系统100包括源设备102,源设备102提供要被目的地设备116解码和显示的、经编码的视频数据。具体地,源设备102经由计算机可读介质110来将视频数据提供给目的地设备116。源设备102和目的地设备116可以包括宽范围的设备中的任何设备,包括台式计算机、笔记本计算机(即,膝上型计算机)、平板计算机、机顶盒、电话手机(诸如智能电话)、电视机、相机、显示设备、数字媒体播放器、视频游戏控制台、视频流式传输设备等。在一些情况下,源设备102和目的地设备116可以被配备用于无线通信,以及因此可以被称为无线通信设备。

[0039] 在图1的示例中,源设备102包括视频源104、存储器106、视频编码器200以及输出接口108。目的地设备116包括输入接口122、视频解码器300、存储器120以及显示设备118。根据本公开内容,源设备102的视频编码器200和目的地设备116的视频解码器300可以被配置为应用用于针对用于视频压缩的算术译码的概率初始化的技术。因此,源设备102表示视频编码设备的示例,而目的地设备116表示视频解码设备的示例。在其它示例中,源设备和目的地设备可以包括其它组件或排列。例如,源设备102可以从诸如外部相机的外部视频源接收视频数据。同样,目的地设备116可以与外部显示设备对接,而不是包括集成的显示设备。

[0040] 在图1中所示的系统100仅是一个示例。通常,任何数字视频编码和/或解码设备可以执行用于针对用于视频压缩的算术译码的概率初始化的技术。源设备102和目的地设备116仅是这样的译码设备的示例,其中,源设备102生成经编码的视频数据以用于传输给目的地设备116。本公开内容将“译码”设备指代为执行对数据的译码(例如,编码和/或解码)的设备。因此,视频编码器200和视频解码器300分别表示译码设备(具体地,视频编码器和视频解码器)的示例。在一些示例中,设备102、116可以以基本上对称的方式进行操作,使得设备102、116中的每一者包括视频编码和解码组件。因此,系统100可以支持在视频设备102、116之间的单向或双向视频传输,例如,以用于视频流式传输、视频回放、视频广播或视频电话。

[0041] 通常,视频源104表示视频数据(即原始的未经译码的视频数据)的源,以及将视频数据的顺序的一系列图片(还被称为“帧”)提供给视频编码器200,视频编码器200对用于图片的数据进行编码。源设备102的视频源104可以包括视频捕获设备,诸如摄像机、包含先前捕获的原始视频的视频存档单元、和/或用于从视频内容提供者接收视频的视频馈送接口。作为另外的替代方式,视频源104可以生成基于计算机图形的数据作为源视频,或者生成实时视频、被存档的视频和计算机生成的视频的组合。在每种情况下,视频编码器200可以对被捕获的、预捕获的或计算机生成的视频数据进行编码。视频编码器200可以将图片从所接收的次序(有时被称为“显示次序”)重新排列为用于译码的译码次序。视频编码器200可以生成包括经编码的视频数据的比特流。然后,源设备102可以经由输出接口108将经编码的

视频数据输出到计算机可读介质110上,以便由例如目的地设备116的输入接口122接收和/或取回。

[0042] 源设备102的存储器106和目的地设备116的存储器120表示通用存储器。在一些示例中,存储器106、120可以存储原始视频数据,例如,来自视频源104的原始视频以及来自视频解码器300的原始的经解码的视频数据。另外或替代地,存储器106、120可以存储可由例如视频编码器200和视频解码器300分别执行的软件指令。尽管在该示例中被示为与视频编码器200和视频解码器300分开,但是应当理解的是,视频编码器200和视频解码器300还可以包括用于在功能上类似或等效目的的内部存储器。此外,存储器106、120可以存储例如从视频编码器200输出并且输入到视频解码器300的经编码的视频数据。在一些示例中,存储器106、120的部分可以被分配为一个或多个视频缓冲器,例如,以存储原始的经解码和/或经编码的视频数据。

[0043] 计算机可读介质110可以表示能够将经编码的视频数据从源设备102传送给目的地设备116的任何类型的介质或设备。在一个示例中,计算机可读介质110表示通信介质,其使得源设备102能够例如经由射频网络或基于计算机的网络,来实时地向目的地设备116直接地发送经编码的视频数据。输出接口108可以根据诸如无线通信协议的通信标准来对包括经编码的视频数据的传输信号进行调整,以及输入接口122可以根据诸如无线通信协议的通信标准来对所接收的传输信息进行解调。通信介质可以包括任何无线或有线通信介质,例如,射频(RF)频谱或一条或多条物理传输线。通信介质可以形成诸如以下各项的基于分组的网络的一部分:局域网、广域网、或诸如互联网的全球网络。通信介质可以包括路由器、交换机、基站、或对于促进从源设备102到目的地设备116的通信而言可以有用的任何其它设备。

[0044] 在一些示例中,计算机可读介质110可以包括存储设备112。源设备102可以将经编码的数据从输出接口108输出到存储设备112。类似地,目的地设备116可以经由输入接口122从存储设备112存取经编码的数据。存储设备112可以包括各种各样的分布式或本地存取的数据存储介质中的任何一者,诸如硬盘驱动器、蓝光光盘、DVD、CD-ROM、闪存、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它适当的数字存储介质。

[0045] 在一些示例中,计算机可读介质110可以包括文件服务器114或可以存储由源设备102生成的经编码的视频数据的另一中间存储设备。源设备102可以将经编码的视频数据输出到文件服务器114或者可以存储由源设备102生成的经编码的视频数据的另一中间存储设备。目的地设备116可以经由流式传输或下载来从文件服务器114存取被存储的视频数据。文件服务器114可以是能够存储经编码的视频数据并且将该经编码的视频数据发送给目的地设备116的任何类型的服务器设备。文件服务器114可以表示网页服务器(例如,用于网站)、文件传输协议(FTP)服务器、内容递送网络设备、或网络附加存储(NAS)设备。目的地设备116可以通过任何标准数据连接(包括互联网连接)来从文件服务器114存取经编码的视频数据。这可以包括适于存取被存储在文件服务器114上的经编码的视频数据的无线信道(例如,Wi-Fi连接)、有线连接(例如,DSL、电缆调制解调器等)、或这两者的组合。文件服务器114和输入接口122可以被配置为根据流式传输协议、下载传输协议、或其组合来操作。

[0046] 输出接口108和输入接口122可以表示无线发射机/接收机、调制解调器、有线联网组件(例如,以太网卡)、根据各种各样的IEEE 802.11标准中的任何一种标准进行操作的无

线通信组件、或其它物理组件。在其中输出接口108和输入接口122包括无线组件的示例中,输出接口108和输入接口122可以被配置为根据蜂窝通信标准(诸如4G、4G-LTE(长期演进)、改进的LTE、5G等)来传输数据(诸如经编码的视频数据)。在其中输出接口108包括无线发射机的一些示例中,输出接口108和输入接口122可以被配置为根据其它无线标准(诸如IEEE 802.11规范、IEEE 802.15规范(例如,ZigBee™)、Bluetooth™标准等)来传输数据(诸如经编码的视频数据)。在一些示例中,源设备102和/或目的地设备116可以包括相应的片上系统(SoC)设备。例如,源设备102可以包括用于执行被赋予视频编码器200和/或输出接口108的功能的SoC设备,以及目的地设备116可以包括用于执行被赋予视频解码器300和/或输入接口122的功能的SoC设备。

[0047] 本公开内容的技术可以应用于视频译码,以支持各种各样的多媒体应用中的任何一者,诸如空中电视广播、有线电视传输、卫星电视传输、互联网流式视频传输(诸如基于HTTP的动态自适应流式传输(DASH))、被编码到数据存储介质上的数字视频、对被存储在数据存储介质上的数字视频的解码、或其它应用。

[0048] 目的地设备116的输入接口122从计算机可读介质110(例如,存储设备112、文件服务器114等)接收经编码的视频比特流。经编码的视频比特流可以包括由视频编码器200定义的诸如以下语法元素的信令信息(其还被视频解码器300使用):所述语法元素具有描述视频块或其它译码单元(例如,切片、图片、图片组、序列等)的特性和/或处理的值。显示设备118将经解码的视频数据的经解码的图片显示给用户。显示设备118可以表示各种各样的显示设备中的任何一者,诸如阴极射线管(CRT)、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。

[0049] 尽管在图1中未示出,但是在一些示例中,视频编码器200和视频解码器300可以各自与音频编码器和/或音频解码器集成,以及可以包括适当的MUX-DEMUX单元或其它硬件和/或软件,以处理包括公共数据流中的音频和视频两者的经复用的流。如果适用,MUX-DEMUX单元可以遵循ITU H.223复用器协议或其它协议(诸如用户数据报协议(UDP))。

[0050] 视频编码器200和视频解码器300各自可以被实现为各种各样的适当的编码器和/或解码器电路中的任何一者,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)、分立逻辑、软件、硬件、固件、或其任何组合。当所述技术部分地在软件中实现时,设备可以将用于软件的指令存储在适当的非暂时性计算机可读介质中,以及使用一个或多个处理器在硬件中执行指令以执行本公开内容的技术。视频编码器200和视频解码器300中的每一者可以被包括在一个或多个编码器或解码器中,编码器或解码器中的任一者可以被集成为相应设备中的组合编码器/解码器(CODEC)的一部分。包括视频编码器200和/或视频解码器300的设备可以包括集成电路、微处理器、和/或无线通信设备(诸如蜂窝电话)。

[0051] 视频编码器200和视频解码器300可以根据视频译码标准(诸如ITU-T H.265(还被称为高效率视频译码(HEVC)标准)或对其的扩展(诸如多视图和/或可缩放视频译码扩展))进行操作。替代地,视频编码器200和视频解码器300可以根据其它专有或行业标准(诸如联合探索测试模型(JEM)或ITU-T H.266标准,还被称为通用视频译码(VVC))进行操作。VVC标准的最新草案是在以下文档中描述的:Bross等人,“Versatile Video Coding (Draft 4)”,ITU-T SG 16WP 3和ISO/IEC JTC 1/SC 29/WG 11的联合视频专家组(JVET),第13次会议:

摩洛哥马拉喀什,2019年1月9-18日,JVET-M1001-v6(下文中被称为“VVC草案4”)。然而,本公开内容的技术不限于任何特定的译码标准。

[0052] 通常,视频编码器200和视频解码器300可以执行对图片的基于块的译码。术语“块”通常指代包括要被处理的(例如,在编码和/或解码过程中要被编码、被解码或以其它方式使用的)数据的结构。例如,块可以包括亮度和/或色度数据的样本的二维矩阵。通常,视频编码器200和视频解码器300可以对以YUV(例如,Y,Cb,Cr)格式表示的视频数据进行译码。也就是说,并不是对用于图片的样本的红色、绿色和蓝色(RGB)数据进行译码,视频编码器200和视频解码器300可以对亮度和色度分量进行译码,其中,色度分量可以包括红色色相和蓝色色相色度分量两者。在一些示例中,视频编码器200在进行编码之前将所接收的经RGB格式化的数据转换为YUV表示,以及视频解码器300将YUV表示转换为RGB格式。替代地,预处理和后处理单元(未示出)可以执行这些转换。

[0053] 概括而言,本公开内容可以涉及对图片的译码(例如,编码和解码)以包括对图片的数据进行编码或解码的过程。类似地,本公开内容可以涉及对图片的块的译码以包括对用于块的数据进行编码或解码(例如,预测和/或残差译码)的过程。经编码的视频比特流通常包括用于表示译码决策(例如,译码模式)以及将图片分割为块的语法元素的一系列值。因此,关于对图片或块进行译码的参考通常应当被理解为对用于形成图片或块的语法元素的值进行译码。

[0054] HEVC定义了各种块,包括译码单元(CU)、预测单元(PU)和变换单元(TU)。根据HEVC,视频译码器(诸如视频编码器200)根据四叉树结构来将译码树单元(CTU)分割为CU。也就是说,视频译码器将CTU和CU分割为四个相等的、不重叠的正方形,以及四叉树的每个节点具有零个或四个子节点。没有子节点的节点可以被称为“叶节点”,以及这样的叶节点的CU可以包括一个或多个PU和/或一个或多个TU。视频译码器可以进一步分割PU和TU。例如,在HEVC中,残差四叉树(RQT)表示对TU的分区。在HEVC中,PU表示帧间预测数据,而TU表示残差数据。经帧内预测的CU包括帧内预测信息,诸如帧内模式指示。

[0055] 作为另一示例,视频编码器200和视频解码器300可以被配置为根据JEM或VVC进行操作。根据JEM或VVC,视频译码器(诸如视频编码器200)将图片分割为多个译码树单元(CTU)。视频编码器200可以根据树结构(诸如四叉树-二叉树(QTBT)结构或多类型树(MTT)结构)分割CTU。QTBT结构去除了多种分割类型的概念,诸如在HEVC的CU、PU和TU之间的分隔。QTBT结构包括两个级别:根据四叉树分割而被分割的第一级别、以及根据二叉树分割而被分割的第二级别。QTBT结构的根节点对应于CTU。二叉树的叶节点对应于译码单元(CU)。

[0056] 在MTT分割结构中,可以使用四叉树(QT)分割、二叉树(BT)分割以及一个或多个类型的三叉树(TT)分割来对块进行分割。三叉树分割是其中块被分为三个子块的分割。在一些示例中,三叉树分割将块划分为三个子块,而不通过中心划分原始块。MTT中的分割类型(例如,QT、BT和TT)可以是对称的或不对称的。

[0057] 在一些示例中,视频编码器200和视频解码器300可以使用单个QTBT或MTT结构来表示亮度分量和色度分量中的每一者,而在其它示例中,视频编码器200和视频解码器300可以使用两个或更多个QTBT或MTT结构,诸如用于亮度分量的一个QTBT/MTT结构以及用于两个色度分量的另一个QTBT/MTT结构(或者用于相应色度分量的两个QTBT/MTT结构)。

[0058] 视频编码器200和视频解码器300可以被配置为使用根据HEVC的四叉树分割、QTBT

分割、MTT分割、或其它分割结构。为了解释的目的,关于QTBT分割给出了本公开内容的技术的描述。然而,应当理解的是,本公开内容的技术还可以应用于被配置为使用二叉树分割或者还使用其它类型的分割的视频译码器。

[0059] 本公开内容可以互换地使用“ $N \times N$ ”和“ N 乘 N ”来指代块(诸如CU或其它视频块)在垂直和水平维度方面的样本大小,例如,16x16个样本或16乘16个样本。通常,16x16 CU将在垂直方向上具有16个样本($y=16$),以及将在水平方向上具有16个样本($x=16$)。同样地, $N \times N$ CU通常在垂直方向上具有 N 个样本,以及在水平方向上具有 N 个样本,其中 N 表示非负整数值。CU中的样本可以按行和列来排列。此外,CU不一定需要在水平方向上具有与在垂直方向上相同的数量的样本。例如,CU可以包括 $N \times M$ 个样本,其中 M 不一定等于 N 。

[0060] 视频编码器200对用于CU的表示预测和/或残差信息以及其它信息的视频数据进行编码。预测信息指示将如何预测CU以便形成用于CU的预测块。残差信息通常表示在编码之前的CU的样本与预测块之间的逐样本差。

[0061] 为了预测CU,视频编码器200通常可以通过帧间预测或帧内预测来形成用于CU的预测块。帧间预测通常指代根据先前译码的图片的数据来预测CU,而帧内预测通常指代根据相同图片的先前译码的数据来预测CU。为了执行帧间预测,视频编码器200可以使用一个或多个运动矢量来生成预测块。视频编码器200通常可以执行运动搜索,以识别例如在CU与参考块之间的差方面与CU紧密匹配的参考块。视频编码器200可以使用绝对差之和(SAD)、平方差之和(SSD)、平均绝对差(MAD)、均方差(MSD)、或其它这样的差计算来计算差度量,以确定参考块是否与当前CU紧密匹配。在一些示例中,视频编码器200可以使用单向预测或双向预测来预测当前CU。

[0062] JEM或VVC的一些示例还提供仿射运动补偿模式,其可以被认为是帧间预测模式。在仿射运动补偿模式下,视频编码器200可以确定表示非平移运动(诸如放大或缩小、旋转、透视运动或其它不规则的运动类型)的两个或更多个运动矢量。

[0063] 为了执行帧内预测,视频编码器200可以选择帧内预测模式来生成预测块。JEM或VVC的一些示例提供六十七种帧内预测模式,包括各种方向性模式、以及平面模式和DC模式。通常,视频编码器200选择帧内预测模式,帧内预测模式描述要根据其来预测当前块(例如,CU的块)的样本的、当前块的相邻样本。假设视频编码器200以光栅扫描次序(从左到右、从上到下)对CTU和CU进行译码,则这样的样本通常可以是在与当前块相同的图片中在当前块的上方、左上方或左侧。

[0064] 视频编码器200对表示用于当前块的预测模式的数据进行编码。例如,对于帧间预测模式,视频编码器200可以对表示使用各种可用帧间预测模式中的哪一种帧间预测模式的数据以及用于对应模式的运动信息进行编码。对于单向或双向帧间预测,例如,视频编码器200可以使用高级运动矢量预测(AMVP)或合并模式来对运动矢量进行编码。视频编码器200可以使用类似的模式来对用于仿射运动补偿模式的运动矢量进行编码。

[0065] 在诸如对块的帧内预测或帧间预测的预测之后,视频编码器200可以计算用于该块的残差数据。残差数据(诸如残差块)表示在块与用于该块的预测块之间的逐样本差,该预测块是使用对应的预测模式来形成的。视频编码器200可以将一个或多个变换应用于残差块,以在变换域中而非在样本域中产生经变换的数据。例如,视频编码器200可以将离散余弦变换(DCT)、整数变换、小波变换或概念上类似的变换应用于残差视频数据。另外,视频

编码器200可以在第一变换之后应用二次变换,诸如模式相关的不可分离二次变换(MDNSST)、信号相关变换、Karhunen-Loeve变换(KLT)等。视频编码器200在应用一个或多个变换之后产生变换系数。

[0066] 如上所述,在任何变换以产生变换系数之后,视频编码器200可以执行对变换系数的量化。量化通常指代如下的过程:在该过程中,对变换系数进行量化以可能减少用于表示系数的数据量,从而提供进一步的压缩。通过执行量化过程,视频编码器200可以减少与系数中的一些或全部系数相关联的比特深度。例如,视频编码器200可以在量化期间将n比特的值向下舍入为m比特的值,其中n大于m。在一些示例中,为了执行量化,视频编码器200可以执行对要被量化的值的按位右移位。

[0067] 在量化之后,视频编码器200可以扫描变换系数,从而根据包括经量化的变换系数的二维矩阵产生一维矢量。可以将扫描设计为将较高能量(以及因此较低频率)的系数放在矢量的前面,以及将较低能量(以及因此较高频率)的变换系数放在矢量的后面。在一些示例中,视频编码器200可以利用预定义的扫描顺序来扫描经量化的变换系数以产生经串行化的矢量,以及以及然后对矢量的经量化的变换系数进行熵编码。在其它示例中,视频编码器200可以执行自适应扫描。在扫描经量化的变换系数以形成一维矢量之后,视频编码器200可以例如根据上下文自适应二进制算术译码(CABAC)来对一维矢量进行熵编码。视频编码器200还可以对用于描述与经编码的视频数据相关联的元数据的语法元素的值进行熵编码,以供视频解码器300在对视频数据进行解码时使用。

[0068] 为了执行CABAC,视频编码器200可以将上下文模型内的上下文指派给要被发送的符号。上下文可以涉及例如符号的相邻值是否为零值。概率确定可以是基于被指派给符号的上下文的。

[0069] 视频编码器200还可以例如在图片报头、块报头、切片报头中生成去往视频解码器300的语法数据(诸如基于块的语法数据、基于图片的语法数据和基于序列的语法数据)、或其它语法数据(诸如序列参数集合(SPS)、图片参数集合(PPS)或视频参数集合(VPS))。同样地,视频解码器300可以对这样的语法数据进行解码以确定如何解码对应的视频数据。

[0070] 以这种方式,视频编码器200可以生成比特流,其包括经编码的视频数据,例如,描述将图片分割为块(例如,CU)以及用于该块的预测和/或残差信息的语法元素。最终,视频解码器300可以接收比特流以及对经编码的视频数据进行解码。

[0071] 以下章节将更详细地描述二进制算术译码(BAC)和CABAC技术。通常,BAC是递归的区间细分过程。在H.264/AVC和H.265/HEVC视频译码标准中,BAC用于对CABAC过程中的仓进行编码。BAC译码器的输出是二进制流,其表示最终译码的概率区间内的概率的值或指针。概率区间由范围(“range”)和下限值(“low”)指定。range是概率区间的延伸。low是译码/概率区间的下限。

[0072] 在以下文档中描述了对视频译码应用算术译码:D.Marpe,H.Schwarz,和T.Wiegand,“Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard”,IEEE Trans.Circuits and Systems for Video Technology,第13卷,第7期,2003年7月。CABAC涉及三个主要功能,即二值化、上下文建模和算术译码。二值化指代将语法元素映射到二进制符号(或“仓”)的功能。二进制符号也可以被称为“仓字符串”。上下文建模指代估计各个仓的概率的功能。算术译码指代基于所估计

的概率来将仓压缩为比特的后续功能。其各种设备和/或其模块(诸如二进制算术译码器)可以执行算术译码的功能。

[0073] 在HEVC中使用了若干不同的二值化过程,包括一元(U)、截断一元(TU)、k阶指数哥伦布(EGk)和固定长度(FL)。在以下文档中描述了各种二值化过程的细节:V.Sze和M.Budagavi,“High throughput CABAC entropy coding in HEVC”,IEEE Transactions on Circuits and Systems for Video Technology(TCSVT),第22卷,第12期,第1778-1791页,2012年12月。

[0074] CABAC中的每个上下文(即概率模型)由一个状态表示。每个状态(σ)隐式地表示特定符号(例如,仓)为最不可能符号(LPS)的概率(p_σ)。符号可以是LPS或最可能符号(MPS)。符号是二进制的,并且因此,MPS和LPS可以是0或1。概率是针对对应的上下文估计的,并且用于(隐式地)使用算术译码器对符号进行熵译码。

[0075] BAC过程由状态机处理,状态机根据要译码的上下文和正在译码的仓的值来改变其内部值“range”和“low”。根据上下文的状态(即其概率),range被分为rangeMPS $_\sigma$ (state $_\sigma$ 中最可能符号的范围)和rangeLPS $_\sigma$ (state $_\sigma$ 中最不可能符号的范围)。理论上,概率state $_\sigma$ 的rangeLPS $_\sigma$ 值通过乘法来推导:

[0076] $\text{rangeLPS}_\sigma = \text{range} \times p_\sigma$

[0077] 其中, p_σ 是选择LPS的概率。当然,MPS的概率是 $1 - p_\sigma$ 。等效地,rangeMPS $_\sigma$ 等于range减去rangeLPS $_\sigma$ 。BAC根据要译码的上下文仓的状态、当前range和正被译码的仓的值(即仓等于LPS或MPS)来迭代地更新range。

[0078] 图2A和2B示出了该过程在仓n处的示例。在图2A的示例201中,在仓n处的range包括RangeMPS和RangeLPS,其由在给定特定上下文状态(σ)情况下的LPS(p_σ)的概率给出。示例201示出了当仓n的值等于MPS时,在仓n+1处的range的更新。在该示例中,low保持不变,但是在仓n+1处的range值被减小为在仓n处的RangeMPS值。图2B的示例203示出当仓n的值不等于MPS(即,等于LPS)时在仓n+1处的range的更新。在该示例中,low被移到在仓n处的RangeLPS的较低range值。此外,在仓n+1处的range的值被减小为在仓n处的RangeLPS的值。

[0079] 在HEVC中,range利用9比特来表达,并且low利用10比特来表达。存在重归一化过程,以便以足够精度来维持range和low值。每当range小于256时发生重归一化。因此,在重归一化之后的range始终等于或大于256。根据range和low的值,BAC向比特流输出“0”或“1”,或更新内部变量(被称为B0:未完成比特),以保留供将来输出。图3示出了根据range的BAC输出的示例。例如,当range和low高于某个门限(例如,512)时,向比特流输出“1”。当range和low低于某个门限(例如,512)时,向比特流输出“0”。当range和low在某些门限之间时,不会向比特流输出任何内容。相反,B0值将递增,并且对下一个仓进行编码。

[0080] 在本公开内容中描述的技术可以例如在视频编码器、视频解码器或组合视频编码器-解码器(编解码器)内执行。具体地,这些技术可以在视频编码器的熵编码单元和/或视频解码器的熵解码单元中执行。这些技术可以例如在CABAC过程内执行,CABAC过程可以被配置为支持视频译码(诸如根据HEVC标准的各方面的视频译码)。熵编码和解码单元可以以反向或逆向的方式应用译码过程,例如,以对各种视频数据(诸如与残差视频数据相关联的变换系数、运动矢量信息、语法元素、以及在视频编码和/或视频解码过程中可以有用的其它类型的信息)中的任何一种进行编码或解码。

[0081] 通常,视频解码器300执行与由视频编码器200执行的过程反向的过程,以对比特流的经编码的视频数据进行解码。例如,视频解码器300可以使用CABAC,以与视频编码器200的CABAC编码过程基本上类似的、但是互易的方式来对用于比特流的语法元素的值进行解码。语法元素可以定义用于将图片分割为CTU、以及根据对应的分割结构(诸如QTBT结构)对每个CTU进行分割以定义CTU的CU的分割信息。语法元素还可以定义用于视频数据的块(例如,CU)的预测和残差信息。

[0082] 残差信息可以通过例如经量化的变换系数来表示。视频解码器300可以对块的经量化的变换系数进行逆量化和逆变换以重新产生用于该块的残差块。视频解码器300使用经信号通知的预测模式(帧内预测或帧间预测)和相关的预测信息(例如,用于帧间预测的运动信息)来形成用于该块的预测块。视频解码器300然后可以对预测块和残差块(在逐个样本的基础上)进行组合以重新产生原始块。视频解码器300可以执行额外处理,诸如执行去块过程以减少沿着块的边界的视觉伪影。

[0083] 根据本公开内容的技术,视频编码器200和/或视频解码器300可以获得用于在对用于视频数据的切片中的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的斜率索引(例如,SlopeIdx)和预定义的偏移索引(例如,OffsetIdx);基于预定义的斜率索引和预定义的偏移索引,在线性域中确定用于视频数据的切片的上下文的初始概率状态;以及基于上下文的初始概率状态来对用于语法元素的值的仓进行熵译码。

[0084] 概括而言,本公开内容可能涉及“用信号通知”某些信息(诸如语法元素)。术语“用信号通知”通常可以指代对语法元素的值和/或用以对经编码的视频数据进行解码的其它数据的传送。也就是说,视频编码器200可以在比特流中用信号通知用于语法元素的值。通常,用信号通知指代在比特流中生成值。如上所述,源设备102可以基本上实时地或不是实时地(诸如可能在将语法元素存储到存储设备112以供目的地设备116稍后取回时发生)将比特流传输给目的地设备116。

[0085] 本公开内容的技术适用于二进制算术译码器(诸如在通用视频编码(VVC)中采用的上下文自适应二进制算术译码器(CABAC)),其中,概率状态表示线性域中的真实概率。

[0086] 在二进制算术译码中,首先将视频序列转换为数据元素(或语法元素),其中时空冗余被去除;然后通过熵译码来将语法元素无损地转换为二进制表示(或比特流)。最新的视频译码标准(例如,HEVC和VVC)可以利用二进制算术译码来执行熵译码。

[0087] 二进制算术译码过程具有三个主要阶段,即二值化、自适应概率估计和算术译码。在二值化阶段,每个要被译码的非二进制语法元素都被转换成二进制数据符号(或仓)的字符串。

[0088] 在自适应概率估计阶段,每个仓(无论是二进制语法元素还是从非二进制语法元素转换的二进制字符串的一个元素)都具有其概率分布的估计(即,为0或1的概率)。分布可以被分为两个类别:(1)平稳和均匀分布(即始终 $p=0.5$)、以及(2)时变或非均匀分布。具有类别(2)分布的仓可以被指派概率模型(或上下文模型),该模型基于仓的先前仓值和其它上下文统计来跟踪该仓的实时分布。

[0089] 在算术译码阶段,具有类别(1)分布的仓可以在旁路模式下进行译码,旁路模式是低复杂度且高度并行化的模式;具有类别(2)分布的仓以常规模式进行译码,其中使用通过

相关联的上下文模型估计的仓值及其概率。

[0090] 当在用于视频译码的二进制算术译码器中使用时,在理论上为实值并且范围从0到1的概率被数字化,并且因此通常被称为概率状态。例如,在HEVC中,概率具有7比特的精度,其对应于128个概率状态。图9示出了在概率与概率状态之间的映射。可以看出,HEVC中的概率状态表示对数域中的真实概率。再例如,在VVC中,特定仓的概率估计是在该仓的相关联的上下文模型中跟踪并且分别以快速和慢速更新的两个概率的平均值。以快速更新的一者具有10比特的精度,其对应于1024个概率状态;以慢速更新的另一者具有14比特的精度,其对应于16384个概率状态。与HEVC不同,VVC采用在概率状态与概率之间的线性映射(参见图10A和10B)。

[0091] 在HEVC和VVC中,视频比特流包括多个可独立解码单元(例如,切片)或者由其组成,这意味着在这样的单元的开始处,必须将所有上下文模型的概率状态重置为一些预定义值。通常,在没有源的统计性质的任何先验知识的情况下,每个上下文模型都应当假设均匀分布($p=0.5$)。然而,为了衔接自适应概率估计的学习阶段并且在不同的译码条件下实现预适应,发现为每个概率模型提供比等概率状态更合适的初始概率状态(初始化过程)是有益的。

[0092] HEVC中的CABAC具有在每个切片的开始处调用的取决于量化参数(QP)的初始化过程。在给定用于切片的亮度QP的初始值 SliceQP_Y 的情况下,特定上下文模型的初始概率状态(被表示为 InitProbState)是通过等式(1)到(3)来生成的,

$$[0093] \quad m = \text{SlopeIdx} * 5 - 45 \quad (1)$$

$$[0094] \quad n = (\text{OffsetIdx} \ll 3) - 16 \quad (2)$$

$$[0095] \quad \text{InitProbState} = \text{Clip3}(1, 127, ((m * \text{SliceQP}_Y) \gg 4) + n) \quad (3)$$

[0096] 其中, SlopeIdx 和 OffsetIdx (两者都是范围从0到15(含)的整数)是为每个上下文模型预定义和存储的初始化参数。等式(3)意指 InitProbState 是通过 SliceQP_Y 的线性函数来建模的,在 $\text{SliceQP}_Y=0$ 处具有大约 $m \gg 4$ 的斜率以及交点 n 。可以分别在表1和2中看到从 SlopeIdx 到斜率以及从 OffsetIdx 到交点的映射。

[0097] 表1-从 SlopeIdx 到斜率的映射

[0098]

SlopeIdx	0	1	2	3	4	5	6	7
斜率	-2.81	-2.50	-2.19	-1.88	-1.56	-1.25	-0.94	-0.63
SlopeIdx	8	9	10	11	12	13	14	15
斜率	-0.31	0	0.31	0.63	0.94	1.25	1.56	1.88

[0099] 表2-从 OffsetIdx 到交点的映射

[0100]

OffsetIdx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
交点	-16	-8	0	8	16	24	32	40	48	56	64	72	80	88	96	104

[0101] 换句话说,上下文模型可以不直接存储初始概率状态;相反,其存储联合地确定线性函数的两个初始化参数,该线性函数在每个切片的开始时使用 SliceQP_Y 作为自变量来推导概率状态。

[0102] 两者都具有4比特的精度的 SlopeIdx 和 OffsetIdx 被打包到单个8比特初始化值中,其高半字节和低半字节分别为 SlopeIdx 和 OffsetIdx 。

[0103] 当前VVC的CABAC使用与HEVC中基本相同的方式来推导 InitProbState ,除了在等

式(3)中的限幅为从0到127。然而,在推导InitProbState(其表示对数域中的概率(参见图9))之后,VVC中的初始化过程需要再一个步骤来将InitProbState转换为表示线性域中的概率的概率状态(参见图)10A和10B),以便在VVC的算术译码引擎中使用。在图11A和11B中所示的转换(或映射)通过如下使用查找表(LUT)(参见表3)来实现:

[0104] 1.使用InitProbState作为搜索索引来在LUT中找到对应的probability_state值。

[0105] 2.-在等式(4)中推导较低精度(10比特)的概率状态(被表示为ProbabilityStateL)。

[0106] $\text{ProbabilityStateL} = \text{probability_state} \gg 5$ (4)

[0107] -在等式(5)中推导较高精度(14比特)的概率状态(被表示为ProbabilityStateH)。

[0108] $\text{ProbabilityStateH} = \text{probability_state} \gg 1$ (5)

[0109] 表3-从InitProbState到probability_state的映射

[0110]

InitProbState	0	1	2	3	4	5	6	7
probability_state	614	647	681	718	756	797	839	884
InitProbState	8	9	10	11	12	13	14	15
probability_state	932	982	1034	1089	1148	1209	1274	1342
InitProbState	16	17	18	19	20	21	22	23
probability_state	1414	1490	1569	1653	1742	1835	1933	2037
InitProbState	24	25	26	27	28	29	30	31
probability_state	2146	2261	2382	2509	2643	2785	2934	3091
InitProbState	32	33	34	35	36	37	38	39
probability_state	3256	3430	3614	3807	4011	4225	4452	4690
InitProbState	40	41	42	43	44	45	46	47
probability_state	4941	5205	5483	5777	6086	6412	6755	7116
InitProbState	48	49	50	51	52	53	54	55
probability_state	7497	7898	8320	8766	9235	9729	10249	10798
InitProbState	56	57	58	59	60	61	62	63
probability_state	11375	11984	12625	13300	14012	14762	15551	16384
InitProbState	64	65	66	67	68	69	70	71
probability_state	16384	17216	18005	18755	19467	20142	20783	21392
InitProbState	72	73	74	75	76	77	78	79
probability_state	21969	22518	23038	23532	24001	24447	24869	25270
InitProbState	80	81	82	83	84	85	86	87
probability_state	25651	26012	26355	26681	26990	27284	27562	27826
InitProbState	88	89	90	91	92	93	94	95
probability_state	28077	28315	28542	28756	28960	29153	29337	29511
InitProbState	96	97	98	99	100	101	102	103
probability_state	29676	29833	29982	30124	30258	30385	30506	30621
InitProbState	104	105	106	107	108	109	110	111
probability_state	30730	30834	30932	31025	31114	31198	31277	31353
InitProbState	112	113	114	115	116	117	118	119

probability_state	31425	31493	31558	31619	31678	31733	31785	31835
InitProbState	120	121	122	123	124	125	126	127
probability_state	31883	31928	31970	32011	32049	32086	32120	32153

[0111] 如上所讨论的,当前VVC的CABAC采用表示线性域中的概率的概率状态(参见图10A和10B),但是仍然使用HEVC中的传统初始化过程,其中所推导出的初始概率状态表示对数域中的概率。为了衔接初始化过程的输出和算术译码引擎的输入,使用128个条目的LUT进行映射。

[0112] 根据本公开内容的一种或多种技术,可以针对VVC修改初始化过程(例如,这些技术可以通过适当地修改VVC中的初始化过程来改进该表查找过程)。提出了用于修改的两种方法。在一种方法中,输出概率状态InitProbState表示线性域中的概率。通过这样做,映射过程被减少为移位操作,仅用于精度调整目的。在另一种方法中,输出概率状态InitProbState表示二次域中的概率,并且映射是通过使用等式而不是LUT来实现的。

[0113] 增加斜率和交点的范围。为了找到用于给定的上下文模型的合适的初始化参数,需要通过某种手段(诸如源的统计分析)来获取使用该上下文模型的仓的概率分布的先验知识。图12示出了在I切片中使用的语法元素SaoMergeFlag的仓的示例。

[0114] 在HEVC和当前VVC中,概率的先验知识被转换为对数域中的概率状态(图13A是从图12转换的),并且最佳初始化参数(通过该参数,所确定的线性函数最佳拟合概率状态和SliceQP_y之间的相关性)是通过穷举搜索或其它更聪明的方法找到的。对于同一示例,SlopedIdx和OffsetIdx分别为13和6(斜率大约为1.25和交点32),并且在图13B中绘制了对应的最佳拟合线性函数。

[0115] 根据本公开内容的技术,初始概率状态在线性域中,以与算术译码引擎协调。对于同一示例,在图14A中示出了由线性域中的概率状态表示的先验知识,这仅是图12的缩放版本。注意的是,为了阐明的简单,针对对数和线性概率状态两者都使用7比特的精度。然而,在其它示例中,可以使用其他比特精度。

[0116] 图13A和图14A的比较表明,概率状态在对数域中伴随着SliceQP_y而变化得更加轻微(与其在线性域中相比),这意味着很好地拟合对数域中的概率状态的平坦线在线性域中可能变得相当陡峭。关于在图14A中所示的概率状态,在不改变在HEVC和当前VVC中定义的初始化过程(参见等式(1)到(3))的情况下,通过穷举搜索可以找到的最佳初始化参数是等于15的SlopedIdx以及等于4的OffsetIdx,利用其达到最大有效斜率1.88,如图14B所示。然而,线可能还没有陡峭到足以实现良好的拟合。随着斜率和交点的搜索范围的增加,可以找到更好的拟合,如图14C所示,其中,斜率为3.25并且交点为-23,两者都超过了HEVC和当前VVC所允许的限制。

[0117] 注意的是,示例(图12到14C)是典型的,而不是异常的-大量的仓具有这种概率分布。因此,应当增加斜率和偏移的范围,以便使用线性模型来拟合SliceQP_y和现在处于线性域中的概率状态的相关性。

[0118] 在一个示例中,可能的斜率和交点值的数量没有改变(即,SlopeIdx和OffsetIdx的范围仍然从0到15),但是它们之间的步长大小增加,这意味着分别地,等式(6)中的常数a大于等式(1)中的5,并且等式(7)中的b大于等式(2)的3。

[0119] $m = \text{SlopeIdx} * a - c$ (6)

[0120] $n = (\text{OffsetIdx} \ll b) - d$ (7)

[0121] 在另一示例中,可能的斜率和交点值的数量增加。例如,SlopeIdx的范围可以从0到31,从而导致32个可能的斜率值。以上两个示例可以联合地使用或者可以独立地使用。

[0122] 定义在SliceQP_Y处的交点不等于0。除了满足具有斜率和偏移的增加的范围的线性概率状态之外,定义在SliceQP_Y处的交点不等于0是一种替代或额外方法。例如,图15A示出了在B切片中使用的SplitFlag的概率分布。在对数域中,如图15B所示,关于切片SliceQP_Y的概率状态可以通过具有等于-0.63的斜率和等于88的交点的线性函数来很好地建模(即,如HEVC和当前VVC中定义的SlopedIX 7和OffsetIdx 13)。然而,在线性域中,在HEVC和当前VVC中定义的有效斜率/交点组合无法很好地对概率状态建模。图15C示出了通过穷举搜索找到的最近拟合,其中斜率等于-0.94,交点等于104(即SlopedIdx 6和OffsetIdx 15)。如图15D所示,另一个有效斜率-1.88(SlopeIdx 3)确实实现了更好的拟合,但是到y轴的投影(即,在SliceQP_Y处的交点等于0)为130,远远超出了104的上限。

[0123] 如以上在等式(3)中所示,在一些示例中,初始概率状态可以部分通过将变量m乘以SliceQP_Y来确定。在等式(3)的示例中,概率状态的线性模型是在SliceQP_Y为零的点处定义的。

[0124] 根据本公开内容的一种或多种技术,视频译码器可以利用偏移,使得概率状态的线性模型被定义在SliceQP_Y的值处而不是零。在线性模型上定义概率状态的点可以被称为量化参数锚定点(QP_{anchor})。例如,视频译码器可以从SliceQP_Y中减去偏移,并且将结果乘以变量m。在一个示例中,这可以通过将等式(3)修改为如下所示的等式(8)来实现。

[0125] $\text{InitProbState} = \text{Clip3}(1, 127, ((m * (\text{SliceQP}_Y - \text{QP}_{\text{anchor}})) \gg 4) + n)$ (8)

[0126] 其中,QP_{anchor}是定义交点的QP值。以这种方式,交点可以不会超过概率状态的有效范围(在该示例中为1到127),并且消除了对在y轴处的大投影的考虑,尤其是当坡度陡峭时。在图15D中,SlopeIdx等于3,QP_{anchor}为32,并且在QP_{anchor}处的交点为72。用于QP_{anchor}的其它值包括但不限于8、16、64、128等。

[0127] 使用针对线性函数的替代表示。在上面讨论的示例中,用于利用SliceQP_Y来对概率状态建模的线性函数是通过由斜率和交点对来确定的,这与HEVC和当前VVC相同。本节提出了针对线性函数的替代表示,其中直线函数通过在两个不同QP处的两个交点来确定。与斜率/交点表示相比,这种双交点表示在拟合方面提供了更多的灵活性,因为其允许不是算术序列形式的更多斜率值。

[0128] 将两个交点和两个对应的QP分别表示为int1、int2、QP1和QP2。QP1和QP2可以是任意的,只要它们在当前VVC中定义的有效范围(0到63)内。然而,由于InitProbState是通过如下等式(9)或等式(10)计算的(注意的是,为了阐明的简单,此处不考虑中间限幅和移位操作),

$$[0129] \quad \frac{\text{int1} - \text{int2}}{\text{QP1} - \text{QP2}} (\text{SliceQP}_Y - \text{QP1}) + \text{int1} \quad (9)$$

$$[0130] \quad \frac{\text{int1} - \text{int2}}{\text{QP1} - \text{QP2}} (\text{SliceQP}_Y - \text{QP2}) + \text{int2} \quad (10)$$

[0131] 所以,QP1和QP2的差最好是2的整数次幂,使得等式(9)和等式(10)中的除法可以用右移位来代替。

[0132] 双交点表示可以进一步扩展为确定分段线性函数的多交点表示。一个这样的示例是来自亮度SigFlag的仓(参见图16A-16C),其中,在大约30或31的QP处出现转折点,而通过斜率/交点对或双交点确定的直线并没有良好到足以进行拟合(参见图16A)。在这种情况下,提出了三重交点表示,其使用三个初始化参数:在等于31的QP处的交点、以及分别在小于和大于31的QP处的两个其它交点(参见图16B)。为了甚至更精确地拟合概率分布,可以使用四重交点表示,其中,使用表示在等于37的QP处的交点的额外初始化参数来捕获另一个次要转折点(参见图16C)。

[0133] 保留较高的中间精度。以上示例介绍了对等式(1)到(3)的修改,并且推导线性域中的InitProbState。由于HEVC传统,InitProbState仍然具有7比特的精度(0到127),并且需要被左移位3比特和7比特,以便与在当前VVC中定义的ProbabilityStateL和ProbabilityStateH的精度对齐,如在等式(11)和(12)中。

[0134] $\text{ProbabilityStateL} = \text{InitProbState} \ll 3$ (11)

[0135] $\text{ProbabilityStateH} = \text{InitProbState} \ll 7$ (12)

[0136] 注意的是,在该技术中保存了正好在等式(4)和(5)之前的表格查找步骤。

[0137] 应用右移位以推导InitProbState(参见等式(3))、之后是左移位(参见等式(11)和(12))肯定比将所有(或部分)右移位移到最后一个步骤损失更多的信息,因为后者保留了较高的中间精度。以下两个示例将所有右移位移到最后一个步骤。

[0138] 在一个示例中:

[0139] $\text{InitProbState} = \text{Clip3}(1, 2047, ((m * \text{SliceQP}_Y) + (n \ll 4)))$ (13)

[0140] $\text{ProbabilityStateL} = \text{InitProbState} \gg 1$ (14)

[0141] $\text{ProbabilityStateH} = \text{InitProbState} \ll 3$ (15)

[0142] 在另一示例中:

[0143] $\text{InitProbState} = \text{Clip3}(0, 2047, ((m * \text{SliceQP}_Y) + (n \ll 4)))$ (16)

[0144] $\text{ProbabilityStateL} = \text{InitProbState} \gg 1$ (17)

[0145] $\text{ProbabilityStateH} = (\text{InitProbState} \ll 3) + 4$ (18)

[0146] CE1相关:以下文档提出了用于初始化的以下技术:Simplification of JVET-00191 using 4or 6bit per initialization value,JVET-00946,瑞典哥德堡,2019年7月3-12日(下文称为“JVET-00946”):

[0147] $q = \text{Clip3}(9, 23, \text{SliceQP}_Y \gg 1) - 8$

[0148] $a = (\text{initValue} \gg 3) * 9$

[0149] $b = (\text{initValue} \& 7) * 9$

[0150] $\text{ProbabilityStateL} = 16 * a + (b - a) * q + 8$

[0151] $\text{ProbabilityStateH} = \text{ProbabilityStateL} \ll 4$

[0152] 然而,在JVET-00946中提出的技术可能存在一个或多个缺点。例如,在JVET-00946中提出的技术可能损失一些中间精度。

[0153] 根据本公开内容的一种或多种技术,视频译码器可以如下执行初始化:

[0154] $q = \text{Clip3}(18, 46, \text{SliceQP}_Y) - 16$

[0155] $a = (\text{initValue} \gg 3) * 9$

[0156] $b = (\text{initValue} \& 7) * 9$

[0157] $\text{ProbabilityStateL} = 16 * a + ((b - a) * q \gg 1) + 8$

[0158] $\text{ProbabilityStateH} = \text{ProbabilityStateL} \ll 4$

[0159] 可以看出,通过移动右移位直到最后一个步骤(即,计算ProbabilityStateL),以上技术允许高中间精度。

[0160] 在二次域中的初始化。以上示例可以用于其中初始化过程的输出表示线性域中的概率状态的场景。然而,对于一些输入源,可能期望使概率状态表示对数域中的概率,其对应于表示线性域中的熵。为了实现对数域中的表示,本公开内容提出了使用二次域中的概率状态。图17A和17B示出了从二次域InitProbState到在VVC的算术译码引擎中使用的线性概率的映射的示例,其看起来类似于用于对数InitProbState的映射(参见图11A-11B),但是可以通过乘法来实现(注意的是,存储所有可能的乘法结果的LUT可以是提前生成的以供使用(如果特定实现所优选的话))。下面是一个示例,

[0161] $\text{slope} = \text{int}(\text{initId} \& 0x0F) - 8;$

[0162] $\text{offset} = \text{int}(\text{initId} \& 0xF0) - 128;$

[0163] $\text{shp} = \text{offset} + ((\text{slope} * (\text{qp} - 32)) \gg 1);$

[0164] $\text{idx} = \text{Clip3}(-127, 127, \text{shp});$

[0165] $\text{val} = 128 - \text{abs}(\text{idx});$

[0166] $\text{pl} = (\text{idx} < 0 ? 32768 - \text{val}^2 : \text{val}^2);$

[0167] $\text{ProbabilityStateL} = \text{pl} \gg 1$

[0168] $\text{ProbabilityStateH} = \text{pl} \gg 5$

[0169] 其中,initId是初始化值,其中高半字节和低半字节分别表示斜率和偏移,并且idx意指具有8比特的精度且以0为中心的的概率状态。

[0170] 注意的是,在本节中提出的方法可以与以上介绍的较高中间精度相结合。

[0171] 针对极端概率分布的初始化。大量仓可能具有极端概率分布,这意味着这些仓具有为1或0的非常高的概率,而与SliceQP_y无关。图18A和18B是示出极端概率分布的示例的图。

[0172] 为了对这样的极端概率分布建模,斜率可能必须为0,并且初始概率状态可以仅取决于n(参见等式(2)和(7))。因此,初始概率状态可以不随着量化参数(QP)而变化。推导n的公式(如在等式(2)或等式(7)中)可以被设计为使得接近0和1的概率都可以高效地表示。以下示例是基于概率状态的7比特表示(0到127)的,并且可以扩展到其它精度。

[0173] 以等式(2)作为示例。在OffsetIdx的范围从0到15的情况下,n的可以达到的最大值为104,其表示概率0.8189(即104/128),其距1.0很远。

[0174] 在以下文档中,以10比特的精度来推导n,其中,OffsetIdx的范围从0到7,如在等式(19)中:“Simplification of the initialization process for context variables”, JVET-N0301,瑞士日内瓦,2019年3月19-27日。

[0175] $n = ((\text{OffsetIdx} * 5) \ll 5) + 32 \quad (19)$

[0176] 在等式(20)中示出了其n的等效7比特表示,

[0177] $n = ((\text{OffsetIdx} * 5) \ll 2) + 4 \quad (20)$

[0178] 其中,其可以达到的最接近0的概率是0.0313(OffsetIdx等于0,并且n等于4, $4/128 = 0.0313$),而其可以达到的最接近1.0的概率是0.9688(OffsetIdx等于6,并且n等于

124,124/128=0.9688)。然而,这种近似可能没有精确到足以表示极端概率分布。

[0179] 根据本公开内容的一种或多种技术,视频译码器可以利用以下公式中的一个或多个公式来推导初始化参数(例如,n)。

[0180] 1.如果OffsetIdx是3比特(从0到7),则n可以如在等式(21)中所示地推导。

[0181] $n = (\text{OffsetIdx} * 18) + 1$ (21)

[0182] 2.如果OffsetIdx是4比特(从0到15),则n可以如在等式(22)中所示地推导。

[0183] $n = (\text{OffsetIdx} * 9) + 1$ (22)

[0184] 可以看出,在任一种情况下,n都可以达到1和127,这意味着可以达到0.0078(即1/128)和0.9922(即127/128)的概率。这样的概率可以分别足够接近0和1。以这种方式,本公开内容的技术使得视频译码器能够更精确地对极端概率分布进行初始化,这可以降低经译码的视频数据的比特率。

[0185] 在一些示例中,m、n和/或initState可以根据以下等式之一来确定:

[0186] (23)

[0187] $m = \text{slopeIdx} * 4 - 36$

[0188] $n = \text{OffsetIdx} * 9 + 1$

[0189] $\text{initState} = (m * (\text{qp} - 24) >> 4) + n$

[0190] (24)

[0191] $m = \text{slopeIdx} * 8 - 32$

[0192] $n = \text{OffsetIdx} * 18 + 1$

[0193] $\text{initState} = (m * (\text{qp} - 16) >> 4) + n$

[0194] (25)

[0195] $m = \text{slopeIdx} * 5 - 45$

[0196] $n = \text{OffsetIdx} * 8 + 7$

[0197] 图4A和4B是示出示例四叉树二叉树(QTBT)结构130以及对应的译码树单元(CTU)132的概念图。实线表示四叉树拆分,以及虚线指示二叉树拆分。在二叉树的每个拆分(即非叶)节点中,用信号通知一个标志以指示使用哪种拆分类型(即,水平或垂直),其中,在该示例中,0指示水平拆分,以及1指示垂直拆分。对于四叉树拆分,由于四叉树节点将块水平地并且垂直地拆分为具有相等大小的4个子块,因此无需指示拆分类型。相应地,视频编码器200可以对以下各项进行编码,以及视频解码器300可以对以下各项进行解码:用于QTBT结构130的区域树级别(即第一级别)(即实线)的语法元素(诸如拆分信息)、以及用于QTBT结构130的预测树级别(即第二级别)(即虚线)的语法元素(诸如拆分信息)。视频编码器200可以对用于通过QTBT结构130的终端叶节点表示的CU的视频数据(诸如预测和变换数据)进行编码,以及视频解码器300可以对该视频数据进行解码。

[0198] 通常,图4B的CTU 132可以与定义与QTBT结构130的处于第一和第二级别的节点相对应的块的大小的参数相关联。这些参数可以包括CTU大小(表示样本中的CTU 132的大小)、最小四叉树大小(MinQTSIZE,其表示最小允许四叉树叶节点大小)、最大二叉树大小(MaxBTSIZE,其表示最大允许二叉树根节点大小)、最大二叉树深度(MaxBTDepth,其表示最大允许二叉树深度)、以及最小二叉树大小(MinBTSize,其表示最小允许二叉树叶节点大小)。

[0199] QTBT结构的与CTU相对应的根节点可以在QTBT结构的第一级别处具有四个子节点,其中的每个子节点可以是根据四叉树分割来分割的。也就是说,第一级别的节点是叶节点(没有子节点)或者具有四个子节点。QTBT结构130的示例将这样的节点表示为包括具有实线分支的父节点和子节点。如果第一级别的节点不大于最大允许二叉树根节点大小(MaxBTSIZE),则可以通过相应的二叉树进一步对它们进行分割。可以对一个节点的二叉树拆分进行迭代,直到从拆分产生的节点达到最小允许二叉树叶节点大小(MinBTSIZE)或最大允许二叉树深度(MaxBTDepth)。QTBT结构130的示例将这样的节点表示为具有虚线分支。二叉树叶节点被称为译码单元(CU),其用于预测(例如,图片内或图片间预测)和变换,而不进行任何进一步分割。如上所讨论的,CU还可以被称为“视频块”或“块”。

[0200] 在QTBT分割结构的一个示例中,CTU大小被设置为128x128(亮度样本和两个对应的64x64色度样本),MinQTSIZE被设置为16x16,MaxBTSIZE被设置为64x64,MinBTSIZE(对于宽度和高度两者)被设置为4,以及MaxBTDepth被设置为4。首先对CTU应用四叉树分割以生成四叉树叶节点。四叉树叶节点可以具有从16x16(即MinQTSIZE)到128x128(即CTU大小)的大小。如果四叉树叶节点为128x128,则由于该大小超过MaxBTSIZE(即,在该示例中为64x64),因此其将不被二叉树进一步拆分。否则,四叉树叶节点将被二叉树进一步分割。因此,四叉树叶节点还是用于二叉树的根节点,以及具有为0的二叉树深度。当二叉树深度达到MaxBTDepth(在该示例中为4)时,不允许进一步拆分。当二叉树节点具有等于MinBTSIZE(在该示例中为4)的宽度时,这意味着不允许进行进一步的水平拆分。类似地,具有等于MinBTSIZE的高度的二叉树节点意味着不允许针对该二叉树节点进行进一步的垂直拆分。如上所述,二叉树的叶节点被称为CU,以及根据预测和变换而被进一步处理,而无需进一步分割。

[0201] 图5是示出可以执行本公开内容的技术的示例视频编码器200的框图。图5是出于解释的目的而提供的,以及不应当被认为对在本公开内容中广泛地举例说明和描述的技术进行限制。出于解释的目的,本公开内容在视频译码标准(诸如HEVC视频译码标准以及处于开发中的H.266视频译码标准)的背景下描述了视频编码器200。然而,本公开内容的技术不限于这些视频译码标准,并且通常适用于视频编码和解码。

[0202] 在图5的示例中,视频编码器200包括视频数据存储器230、模式选择单元202、残差生成单元204、变换处理单元206、量化单元208、逆量化单元210、逆变换处理单元212、重构单元214、滤波器单元216、解码图片缓冲器(DPB)218和熵编码单元220。视频数据存储器230、模式选择单元202、残差生成单元204、变换处理单元206、量化单元208、逆量化单元210、逆变换处理单元212、重构单元214、滤波器单元216、DPB 218和熵编码单元220中的任何一者或全部可以在一个或多个处理器中或者在处理电路中实现。此外,视频编码器200可以包括额外或替代的处理器或处理电路以执行这些和其它功能。

[0203] 视频数据存储器230可以存储要由视频编码器200的组件来编码的视频数据。视频编码器200可以从例如视频源104(图1)接收被存储在视频数据存储器230中的视频数据。DPB 218可以充当参考图片存储器,其存储参考视频数据以在由视频编码器200对后续视频数据进行预测时使用。视频数据存储器230和DPB 218可以由各种各样的存储器设备中的任何一者形成,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)、或其它类型的存储器设备。视频数据存储器230和DPB 218可以由相同的

存储器设备或单独的存储器设备来提供。在各个示例中,视频数据存储器230可以与视频编码器200的其它组件在芯片上(如图所示),或者相对于那些组件在芯片外。

[0204] 在本公开内容中,对视频数据存储器230的引用不应当被解释为限于在视频编码器200内部的存储器(除非如此具体地描述)或者在视频编码器200外部的存储器(除非如此具体地描述)。确切而言,对视频数据存储器230的引用应当被理解为存储视频编码器200接收以用于编码的视频数据(例如,用于要被编码的当前块的视频数据)的参考存储器。图1的存储器106还可以提供对来自视频编码器200的各个单元的输出的临时存储。

[0205] 说明了图5的各个单元以帮助理解由视频编码器200执行的操作。该单元可以被实现为固定功能电路、可编程电路、或其组合。固定功能电路指代提供特定功能并且关于可以执行的操作而预先设置的电路。可编程电路指代可以被编程以执行各种任务并且以可以执行的操作来提供灵活功能的电路。例如,可编程电路可以执行软件或固件,软件或固件使得可编程电路以软件或固件的指令所定义的方式进行操作。固定功能电路可以执行软件指令(例如,以接收参数或输出参数),但是固定功能电路执行的操作类型通常是不可变的。在一些示例中,该单元中的一个或多个单元可以是不同的电路块(固定功能或可编程),以及在一些示例中,一个或多个单元可以是集成电路。

[0206] 视频编码器200可以包括根据可编程电路形成的算术逻辑单元(ALU)、基本功能单元(EFU)、数字电路、模拟电路和/或可编程内核。在其中使用由可编程电路执行的软件来执行视频编码器200的操作的示例中,存储器106(图1)可以存储视频编码器200接收并且执行的软件的目标代码,或者视频编码器200内的另一存储器(未示出)可以存储这样的指令。

[0207] 视频数据存储器230被配置为存储所接收的视频数据。视频编码器200可以从视频数据存储器230取回视频数据的图片,以及将视频数据提供给残差生成单元204和模式选择单元202。视频数据存储器230中的视频数据可以是要被编码的原始视频数据。

[0208] 模式选择单元202包括运动估计单元222、运动补偿单元224和帧内预测单元226。模式选择单元202可以包括额外功能单元,其根据其它预测模式来执行视频预测。作为示例,模式选择单元202可以包括调色板单元、块内复制单元(其可以是运动估计单元222和/或运动补偿单元224的一部分)、仿射单元、线性模型(LM)单元等。

[0209] 模式选择单元202通常协调多个编码通路(pass),以测试编码参数的组合以及针对这样的组合的所得到的速率-失真值。编码参数可以包括将CTU分割为CU、用于CU的预测模式、用于CU的残差数据的变换类型、用于CU的残差数据的量化参数等。模式选择单元202可以最终选择编码参数的具有比其它测试的组合更好的速率-失真值的组合。

[0210] 视频编码器200可以将从视频数据存储器230中检索的图片分割为一系列CTU,以及将一个或多个CTU封装在切片内。模式选择单元202可以根据树结构(诸如上述HEVC的QTBT结构或四叉树结构)来分割图片的CTU。如上所述,视频编码器200可以通过根据树结构来分割CTU,从而形成一个或多个CU。这样的CU通常还可以被称为“视频块”或“块”。

[0211] 通常,模式选择单元202还控制其组件(例如,运动估计单元222、运动补偿单元224和帧内预测单元226)以生成用于当前块(例如,当前CU,或者在HEVC中的PU和TU的重叠部分)的预测块。为了对当前块进行帧间预测,运动估计单元222可以执行运动搜索以识别在一个或多个参考图片(例如,被存储在DPB 218中的一个或多个先前译码的图片)中的一个或多个紧密匹配的参考块。具体地,运动估计单元222可以例如根据绝对差之和(SAD)、平方

差之和 (SSD)、平均绝对差 (MAD)、均方差 (MSD) 等, 来计算表示潜在参考块将与当前块的类似程度的值。运动估计单元222通常可以使用在当前块与所考虑的参考块之间的逐样本差来执行这些计算。运动估计单元222可以识别根据这些计算所得到的具有最低值的参考块, 其指示与当前块最紧密匹配的参考块。

[0212] 运动估计单元222可以形成一个或多个运动矢量 (MV), 所述运动矢量限定相对于当前块在当前图片中的位置而言参考块在参考图片中的位置。然后, 运动估计单元222可以将运动矢量提供给运动补偿单元224。例如, 对于单向帧间预测, 运动估计单元222可以提供单个运动矢量, 而对于双向帧间预测, 运动估计单元222可以提供两个运动矢量。然后, 运动补偿单元224可以使用运动矢量来生成预测块。例如, 运动补偿单元224可以使用运动矢量来取回参考块的数据。作为另一示例, 如果运动矢量具有分数样本精度, 则运动补偿单元224可以根据一个或多个插值滤波器来对用于预测块的值进行插值。此外, 对于双向帧间预测, 运动补偿单元224可以取回用于通过相应的运动矢量标识的两个参考块的数据以及例如通过逐样本平均或加权平均来将所检索到的数据进行组合。

[0213] 作为另一示例, 对于帧内预测或帧内预测译码, 帧内预测单元226可以根据与当前块相邻的样本来生成预测块。例如, 对于方向性模式, 帧内预测单元226通常可以在数学上将相邻样本的值进行组合, 以及跨越当前块在所定义的方向上填充这些计算出的值以产生预测块。作为另一示例, 对于DC模式, 帧内预测单元226可以计算当前块的相邻样本的平均, 以及生成预测块以包括针对预测块的每个样本的该得到的平均。

[0214] 模式选择单元202将预测块提供给残差生成单元204。残差生成单元204从视频数据存储器230接收当前块的原始的未经译码的版本, 以及从模式选择单元202接收预测块。残差生成单元204计算在当前块与预测块之间的逐样本差。所得到的逐样本差定义了用于当前块的残差块。在一些示例中, 残差生成单元204还可以确定残差块中的样本值之间的差以使用残差差分脉冲码调制 (RDPCM) 来生成残差块。在一些示例中, 可以使用执行二进制减法的一个或多个减法器电路来形成残差生成单元204。

[0215] 在其中模式选择单元202将CU分割为PU的示例中, 每个PU可以与亮度预测单元和对应的色度预测单元相关联。视频编码器200和视频解码器300可以支持具有各种大小的PU。如上所指出的, CU的大小可以指代CU的亮度译码块的大小, 以及PU的大小可以指代PU的亮度预测单元的大小。假设特定CU的大小为 $2N \times 2N$, 则视频编码器200可以支持用于帧内预测的 $2N \times 2N$ 或 $N \times N$ 的PU大小、以及用于帧间预测的 $2N \times 2N$ 、 $2N \times N$ 、 $N \times 2N$ 、 $N \times N$ 或类似的对称的PU大小。视频编码器200和视频解码器300还可以支持针对用于帧间预测的 $2N \times nU$ 、 $2N \times nD$ 、 $nL \times 2N$ 和 $nR \times 2N$ 的PU大小的非对称分割。

[0216] 在其中模式选择单元不将CU进一步分割为PU的示例中, 每个CU可以与亮度译码块和对应的色度译码块相关联。如上所述, CU的大小可以指代CU的亮度译码块的大小。视频编码器200和视频解码器300可以支持 $2N \times 2N$ 、 $2N \times N$ 或 $N \times 2N$ 的CU大小。

[0217] 对于其它视频译码技术 (举几个示例, 诸如块内复制模式译码、仿射模式译码和线性模型 (LM) 模式译码), 模式选择单元202经由与译码技术相关联的相应单元来生成用于正被编码的当前块的预测块。在一些示例中 (诸如调色板模式译码), 模式选择单元202可以不生成预测块, 而是替代地生成指示基于所选择的调色板来重构块的方式的语法元素。在这样的模式下, 模式选择单元202可以将这些语法元素提供给熵编码单元220以进行编码。

[0218] 如上所述,残差生成单元204接收用于当前块和对应的预测块的视频数据。然后,残差生成单元204针对当前块生成残差块。为了生成残差块,残差生成单元204计算在预测块与当前块之间的逐样本差。

[0219] 变换处理单元206将一个或多个变换应用于残差块,以生成变换系数的块(本文中被称为“变换系数块”)。变换处理单元206可以将各种变换应用于残差块,以形成变换系数块。例如,变换处理单元206可以将离散余弦变换(DCT)、方向变换、Karhunen-Loeve变换(KLT)、或概念上类似的变换应用于残差块。在一些示例中,变换处理单元206可以对残差块执行多个变换,例如,初级变换和二次变换(诸如旋转变换)。在一些示例中,变换处理单元206不对残差块应用变换。

[0220] 量化单元208可以对变换系数块中的变换系数进行量化,以产生经量化的变换系数块。量化单元208可以根据与当前块相关联的量化参数(QP)值来对变换系数块的变换系数进行量化。视频编码器200(例如,经由模式选择单元202)可以通过调整与CU相关联的QP值来调整被应用于与当前块相关联的变换系数块的量化程度。量化可能引起信息损失,以及因此,经量化的变换系数可能具有与变换处理单元206所产生的原始变换系数相比较低的精度。

[0221] 逆量化单元210和逆变换处理单元212可以将逆量化和逆变换分别应用于经量化的变换系数块,以根据变换系数块来重构残差块。重构单元214可以基于经重构的残差块和由模式选择单元202生成的预测块来产生与当前块相对应的重构块(尽管潜在地具有某种程度的失真)。例如,重构单元214可以将经重构的残差块的样本与来自模式选择单元202所生成的预测块的对应样本相加,以产生经重构的块。

[0222] 滤波器单元216可以对经重构的块执行一个或多个滤波器操作。例如,滤波器单元216可以执行去块操作以减少沿着CU的边缘的块效应伪影。在一些示例中,可以跳过滤波器单元216的操作。

[0223] 视频编码器200将经重构的块存储在DPB 218中。例如,在其中不需要滤波器单元216的操作的示例中,重构单元214可以将经重构的块存储到DPB 218中。在其中需要滤波器单元216的操作的示例中,滤波器单元216可以将经滤波的重构块存储到DPB 218中。运动估计单元222和运动补偿单元224可以从DPB 218取回根据经重构的(以及潜在地经滤波的)块形成的参考图片,以对后续编码的图片的块进行帧间预测。另外,帧内预测单元226可以使用在DPB 218中的当前图片的经重构的块来对当前图片中的其它块进行帧内预测。

[0224] 通常,熵编码单元220可以对从视频编码器200的其它功能组件接收的语法元素进行熵编码。例如,熵编码单元220可以对来自量化单元208的经量化的变换系数块进行熵编码。作为另一示例,熵编码单元220可以对来自模式选择单元202的预测语法元素(例如,用于帧间预测的运动信息或用于帧内预测的帧内模式信息)进行熵编码。熵编码单元220可以对作为视频数据的另一示例的语法元素执行一个或多个熵编码操作,以生成经熵编码的数据。例如,熵编码单元220可以执行上下文自适应变长编码(CAVLC)操作、CABAC操作、可变-可变(V2V)长度译码操作、基于语法的上下文自适应二进制算术译码(SBAC)操作、概率区间分割熵(PIPE)译码操作、指数哥伦布编码操作、或对数据的另一类型的熵编码操作。在一些示例中,熵编码单元220可以在其中语法元素未被熵编码的旁路模式下操作。

[0225] 图6是根据本公开内容的技术的可以被配置为执行CABAC的示例熵编码单元220的

框图。语法元素1180被输入到熵编码单元220中。如果语法元素已经是二进制值语法元素(例如,仅具有值0和1的标志或其它语法元素),则可以跳过二值化的步骤。如果语法元素是非二进制值语法元素(例如,可以具有除了1或0以外的值的语法元素),则由二值化器1200对非二进制值语法元素进行二值化。二值化器1200将非二进制值语法元素映射到二进制决策的序列中。这些二进制决策通常被称为“仓”。例如,对于变换系数级别,该级别的值可以被分解为连续的仓,每个仓指示系数级别的绝对值是否大于某个值。例如,仓0(有时被称为显著性标志)指示变换系数级别的绝对值是否大于0。仓1指示变换系数级别的绝对值是否大于1,以此类推。可以为每个非二进制值语法元素开发唯一的映射。

[0226] 由二值化器1200产生的每个仓被馈送到熵编码单元220的二进制算术译码侧。也就是说,对于预定的非二进制值语法元素集合,每个仓类型(例如,仓0)在下一个仓类型(例如,仓1)之前被译码。译码可以在常规模式或旁路模式下执行。在旁路模式中,旁路编码引擎1260使用固定概率模型(例如,使用Golomb-Rice或指数Golomb译码)来执行算术译码。旁路模式通常用于更加可预测的语法元素。

[0227] 在常规模式下的译码包括执行CABAC。常规模式CABAC用于对仓值进行译码,其中,仓的值的概率在给定先前译码的仓的值的情况下是可预测的。仓为LPS的概率由上下文建模器1220来确定。上下文建模器1220输出该仓值和用于上下文的概率状态(例如,概率状态 σ ,包括LPS值和LPS发生的概率)。上下文可以是用于一系列仓的初始上下文,或者可以是基于先前译码的仓的译码值来确定的。如上所述,上下文建模器1220可以基于所接收的仓是MPS还是LPS来更新状态。在上下文建模器1220确定上下文和概率状态 σ 之后,常规编码引擎1240对该仓值执行BAC。

[0228] 返回到图5,视频编码器200可以输出比特流,其包括用于重构切片或图片的块所需要的经熵编码的语法元素。具体地,熵编码单元220可以输出比特流。

[0229] 上述操作是关于块描述的。这样的描述应当被理解为用于亮度译码块和/或色度译码块的操作。如上所述,在一些示例中,亮度译码块和色度译码块是CU的亮度分量和色度分量。在一些示例中,亮度译码块和色度译码块是PU的亮度分量和色度分量。

[0230] 在一些示例中,不需要针对色度译码块重复关于亮度译码块执行的操作。作为一个示例,不需要重复用于识别用于亮度译码块的运动矢量(MV)和参考图片的操作来识别用于色度块的MV和参考图片。而是,可以对用于亮度译码块的MV进行缩放以确定用于色度块的MV,以及参考图片可以是相同的。作为另一示例,对于亮度译码块和色度译码块,帧内预测过程可以是相同的。

[0231] 视频编码器200表示一种被配置为对视频数据进行编码的设备的示例,其包括:被配置为存储视频数据的存储器;以及在电路中实现的一个或多个处理单元,并且被配置为:获得用于在对用于视频数据的切片中的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的斜率索引(例如,SlopeIdx)和预定义的偏移索引(例如,OffsetIdx);基于预定义的斜率索引和预定义的偏移索引,来确定用于视频数据的切片的上下文的初始概率状态;以及基于上下文的初始概率状态来对用于语法元素的值的仓进行熵编码。

[0232] 图7是示出可以执行本公开内容的技术的示例视频解码器300的框图。图7是出于解释的目的而提供的,以及不对在本公开内容中广泛地举例说明和描述的技术进行限制。

出于解释的目的,本公开内容根据JEM、VVC和HEVC的技术描述了视频解码器300。然而,本公开内容的技术可以由被配置为其它视频译码标准的视频译码设备来执行。

[0233] 在图7的示例中,视频解码器300包括译码图片缓冲器(CPB)存储器320、熵解码单元302、预测处理单元304、逆量化单元306、逆变换处理单元308、重构单元310、滤波器单元312和解码图片缓冲器(DPB)314。CPB存储器320、熵解码单元302、预测处理单元304、逆量化单元306、逆变换处理单元308、重构单元310、滤波器单元312和DPB 314中的任何一者或全部可以在一个或多个处理器中或者在处理电路中实现。此外,视频解码器300可以包括额外或替代的处理器或处理电路以执行这些和其它功能。

[0234] 预测处理单元304包括运动补偿单元316和帧内预测单元318。预测处理单元304可以包括加法单元,其根据其它预测模式来执行预测。作为示例,预测处理单元304可以包括调色板单元、块内复制单元(其可以形成运动补偿单元316的一部分)、仿射单元、线性模型(LM)单元等。在其它示例中,视频解码器300可以包括更多、更少或不同的功能组件。

[0235] CPB存储器320可以存储要由视频解码器300的组件解码的视频数据,诸如经编码的视频比特流。例如,可以从计算机可读介质110(图1)获得被存储在CPB存储器320中的视频数据。CPB存储器320可以包括存储来自经编码的视频比特流的经编码的视频数据(例如,语法元素)的CPB。此外,CPB存储器320可以存储除了经译码的图像的语法元素之外的视频数据,诸如表示来自视频解码器300的各个单元的输出的临时数据。DPB314通常存储经解码的图片,视频解码器300可以输出经解码的图片,和/或在解码经编码的视频比特流的后续数据或图片时使用经解码的图片作为参考视频数据。CPB存储器320和DPB 314可以由各种各样的存储器设备中的任何一者形成,诸如动态随机存取存储器(DRAM),包括(包括同步DRAM(SDRAM))、磁阻式RAM(MRAM)、电阻式RAM(RRAM)、或其它类型的存储器设备。CPB存储器320和DPB 314可以由相同的存储器设备或分别的存储器设备来提供。在各个示例中,CPB存储器320可以与视频解码器300的其它组件在芯片上,或者相对于那些组件在芯片外。

[0236] 另外或替代地,在一些示例中,视频解码器300可以从存储器120(图1)取回经译码的视频数据。也就是说,存储器120可以如上文所讨论地利用CPB存储器320来存储数据。同样,当视频解码器300的功能中的一些或全部功能是在要被视频解码器300的处理电路执行的软件中实现时,存储器120可以存储要被视频解码器300执行的指令。

[0237] 说明了在图7中示出的各个单元以帮助理解由视频解码器300执行的操作。该单元可以被实现为固定功能电路、可编程电路、或其组合。类似于图5,固定功能电路指代提供特定功能并且关于可以执行的操作而预先设置的电路。可编程电路指代可以被编程以执行各种任务并且以可以执行的操作来提供灵活功能的电路。例如,可编程电路可以执行软件或固件,软件或固件使得可编程电路以软件或固件的指令所定义的方式进行操作。固定功能电路可以执行软件指令(例如,以接收参数或输出参数),但是固定功能电路执行的操作的类型通常是不可变的。在一些示例中,该单元中的一个或多个单元可以是不同的电路块(固定功能或可编程),以及在一些示例中,一个或多个单元可以是集成电路。

[0238] 视频解码器300可以包括根据可编程电路形成的ALU、EFU、数字电路、模拟电路和/或可编程内核。在其中由在可编程电路上执行的软件执行视频解码器300的操作的示例中,片上或片外存储器可以存储视频解码器300接收并且执行的软件的指令(例如,目标代码)。

[0239] 熵解码单元302可以从CPB接收经编码的视频数据,以及对视频数据进行熵解码以

重新产生语法元素。预测处理单元304、逆量化单元306、逆变换处理单元308、重构单元310和滤波器单元312可以基于从比特流中提取的语法元素来生成经解码的视频数据。

[0240] 图8是根据本公开内容的技术的可以被配置为执行CABAC的示例熵解码单元302的框图。图8的熵解码单元302以与在图6中描述的熵编码单元220反向的方式来执行CABAC。来自比特流2180的经译码的比特被输入到熵解码单元302中。经译码的比特是基于它们是使用常规模式还是旁路模式来熵译码的,而被馈送给上下文建模器2200或旁路解码引擎2220。如果经译码的比特是以旁路模式来译码的,则旁路解码引擎将使用Golomb-Rice或指数Golomb解码,例如,以检索二进制值语法元素或非二进制语法元素的仓。

[0241] 如果经译码的比特是以常规模式来译码的,则上下文建模器2200可以确定用于经译码的比特的概率模型,并且常规解码引擎2240可以对经译码的比特进行解码以产生非二进制值语法元素的仓(或语法元素本身,如果为二进制值的话)。在上下文建模器2200确定上下文和概率状态 σ 之后,常规解码引擎2240执行BAC以对仓值进行解码。换句话说,常规解码引擎2240可以确定上下文的概率状态,并且基于先前译码的仓和当前范围来对仓值进行解码。在解码仓之后,上下文建模器2200可以基于窗口大小和经解码的仓的值来更新上下文的概率状态。

[0242] 返回到图7,通常,视频解码器300逐块地重构图片。视频解码器300可以单独地对每个块执行重构操作(其中,当前正在被重构(即,被解码)的块可以被称为“当前块”)。

[0243] 熵解码单元302可以对定义经量化的变换系数块的经量化的变换系数的语法元素以及诸如量化参数(QP)和/或变换模式指示的变换信息进行熵解码。逆量化单元306可以使用与经量化的变换系数块相关联的QP来确定量化程度,以及同样地,确定供逆量化单元306应用的逆量化程度。逆量化单元306可以例如执行按位左移位操作以对经量化的变换系数进行逆量化。逆量化单元306从而可以形成包括变换系数的变换系数块。

[0244] 在逆量化单元306形成变换系数块之后,逆变换处理单元308可以将一个或多个逆变换应用于变换系数块,以生成与当前块相关联的残差块。例如,逆变换处理单元308可以将逆DCT、逆整数变换、逆Karhunen-Loeve变换(KLT)、逆旋转变换、逆方向变换或另一逆变换应用于变换系数块。

[0245] 此外,预测处理单元304根据由熵解码单元302进行熵解码的预测信息语法元素来生成预测块。例如,如果预测信息语法元素指示当前块是经帧间预测的,则运动补偿单元316可以生成预测块。在这种情况下,预测信息语法元素可以指示在DPB 314中的要从其取回参考块的参考图片、以及标识相对于当前块在当前图片中的位置而言参考块在参考图片中的位置的运动矢量。运动补偿单元316通常可以以与关于运动补偿单元224(图5)所描述的方式基本类似的方式来执行帧间预测过程。

[0246] 作为另一示例,如果预测信息语法元素指示当前块是经帧内预测的,则帧内预测单元318可以根据通过预测信息语法元素指示的帧内预测模式来生成预测块。再次,帧内预测单元318通常可以以与关于帧内预测单元226(图5)所描述的方式基本上类似的方式来执行帧内预测过程。帧内预测单元318可以从DPB 314取回当前块的相邻样本的数据。

[0247] 重构单元310可以使用预测块和残差块来重构当前块。例如,重构单元310可以将残差块的样本与预测块的对应样本相加来重构当前块。

[0248] 滤波器单元312可以对经重构的块执行一个或多个滤波器操作。例如,滤波器单元

312可以执行去块操作以减少沿着经重构的块的边缘的块效应伪影。不必要在全部示例中执行滤波器单元312的操作。

[0249] 视频解码器300可以将经重构的块存储在DPB 314中。例如,在其中不执行滤波器单元312的操作的示例中,重构单元310可以将经重构的块存储到DPB 314中。在其中执行滤波器单元312的操作的示例中,滤波器单元312可以将经滤波的重构块存储到DPB 314中。如上所讨论的,DPB 314可以将参考信息(诸如用于帧内预测的当前图片以及用于后续运动补偿的先前解码的图片的样本)提供给预测处理单元304。此外,视频解码器300可以从DPB输出经解码的图片,以用于在诸如图1的显示设备118的显示设备上的后续呈现。

[0250] 以这种方式,视频解码器300表示视频解码设备的示例,该视频解码设备包括:被配置为存储视频数据的存储器;以及一个或多个处理单元,其在电路中实现并且被配置为:获得用于在对用于视频数据的切片中的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的斜率索引(例如,SlopeIdx)和预定义的偏移索引(例如,OffsetIdx);基于预定义的斜率索引和预定义的偏移索引,来确定用于视频数据的切片的上下文的初始概率状态;以及基于上下文的初始概率状态来对用于语法元素的值的仓进行熵解码。

[0251] 图19是示出用于对当前块进行编码的示例方法的流程图。当前块可以包括当前CU。尽管关于视频编码器200(图1和5)进行了描述,但是应当理解的是,其它设备可以被配置为执行与图19的方法类似的方法。

[0252] 在该示例中,视频编码器200最初预测当前块(350)。例如,视频编码器200可以形成用于当前块的预测块。然后,视频编码器200可以计算用于当前块的残差块(352)。为了计算残差块,视频编码器200可以计算在原始的未经译码的块与用于当前块的预测块之间的差。然后,视频编码器200可以对残差块进行变换以及对残差块的系数进行量化(354)。接下来,视频编码器200可以扫描残差块的经量化的变换系数(356)。在扫描期间或在扫描之后,视频编码器200可以对系数进行熵编码(358)。例如,视频编码器200可以使用上文并且参考图20讨论的用于概率初始化的技术,使用CAVLC或CABAC来对系数和/或其它语法元素进行编码。然后,视频编码器200可以输出块的经熵译码的数据(360)。

[0253] 图20是示出根据本公开内容的一种或多种技术的用于执行基于上下文的熵编码的示例过程的流程图。图20的技术可以由视频编码器来执行,诸如在图1、5和6中示出的视频编码器200。为了说明的目的,在图1、5和6的视频编码器200的背景下描述了图20的技术,但是具有不同于视频编码器200的配置的视频编码器可以执行图20的技术。

[0254] 视频编码器200可以获得要使用基于上下文的熵译码(例如,CABAC)来编码的仓字符串(例如,一维二进制向量)(2002)。例如,视频编码器200的熵编码单元220可以通过对从视频编码器200的模式选择单元202接收的语法元素进行二值化来获得仓字符串。

[0255] 视频编码器200可以获得用于多个上下文中的上下文的预定义的初始化值(2004)。例如,视频编码器200的熵编码单元220可以获得initValue的值,该值可以是六比特变量。

[0256] 视频编码器200可以基于预定义的初始化值来在线性域中确定用于视频数据的可独立译码单元(例如,切片、瓦片等)的上下文的初始概率状态(2006)。例如,熵编码单元220可以在线性域中确定上下文的初始概率状态,而不在中间在对数域中确定上下文的初始概

率状态。在一些示例中,熵编码单元220可以在不使用LUT将初始概率状态从对数域转换到线性域的情况下确定初始概率状态。

[0257] 为了确定初始概率状态,熵编码单元220可以获得与可独立译码单元相关联的量化参数的初始值(例如, SliceQP_Y),并且获得不为零的量化参数锚定点的值(例如, $\text{QP}_{\text{anchor}}$)。熵编码单元220可以基于用于切片的量化参数的初始值与量化参数锚定点的值之间的差来确定初始概率状态。例如,熵编码单元220可以根据以下等式来确定初始概率状态:

[0258] $\text{InitProbState} = ((m * (\text{SliceQP}_Y - \text{QP}_{\text{anchor}})) \gg \text{rshift}) + n$

[0259] 其中,InitProbState是初始概率状态, SliceQP_Y 是量化参数的初始值, $\text{QP}_{\text{anchor}}$ 是量化参数锚定点,并且rshift是右移位值。

[0260] 如上所讨论的,在一些示例中,熵编码单元220可以被配置为以增加的精度针对极端概率分布执行初始化。例如,熵编码单元220可以基于预定义的初始化值来确定斜率索引值和偏移索引值。熵编码单元220可以进行以下操作:基于斜率索引值来确定m的值;以及基于偏移索引值来确定n的值。在一些示例中,熵编码单元220可以根据以下等式来确定n的值: $n = (\text{OffsetIdx} * 18) + 1$,其中OffsetIdx是偏移索引值。如上所示,在一些示例中,熵编码单元220可以基于m的值和n的值来确定初始概率状态。

[0261] 视频编码器200可以在视频比特流中并且基于上下文的初始概率状态来对仓字符串的仓进行编码(2008)。例如,熵编码单元220可以输出表示在上下文的最终译码的概率区间内的概率的值或指针的二进制流。

[0262] 图21是示出用于对视频数据的当前块进行解码的示例方法的流程图。当前块可以包括当前CU。尽管关于视频解码器300(图1和7)进行了描述,但是应当理解的是,其它设备可以被配置为执行与图21的方法类似的方法。

[0263] 视频解码器300可以接收用于当前块的经熵译码的数据(诸如经熵译码的预测信息和用于与当前块相对应的残差块的系数的经熵译码的数据)(370)。视频解码器300可以对经熵译码的数据进行熵解码以确定用于当前块的预测信息并且重新产生残差块的系数(372)。例如,视频解码器300可以使用上文并且参考图22讨论的用于概率初始化的技术,使用CAVLC或CABAC来对系数和/或其它语法元素进行解码。视频解码器300可以例如使用如由用于当前块的预测信息所指示的帧内或帧间预测模式来预测当前块(374),以计算用于当前块的预测块。然后,视频解码器300可以对所重新产生的系数进行逆扫描(376),以创建经量化的变换系数的块。然后,视频解码器300可以对系数进行逆量化和逆变换以产生残差块(378)。最终,视频解码器300可以通过将预测块和残差块进行组合来对当前块进行解码(380)。

[0264] 图22是示出根据本公开内容的一种或多种技术的用于执行基于上下文的熵解码的示例过程的流程图。图22的技术可以由视频解码器来执行,诸如在图1、7和8中示出的视频解码器300。为了说明的目的,在图1、7和8的视频解码器300的背景下描述了图22的技术,但是具有不同于视频解码器300的配置的视频解码器可以执行图22的技术。

[0265] 视频解码器300可以从视频比特流中获得要使用基于上下文的熵译码来解码的仓字符串(例如,一维二进制向量)(2202)。例如,视频解码器300的熵解码单元302可以从CPB存储器320获得仓字符串。在一些示例中,仓字符串可以表示在上下文的最终译码的概率区

间内的概率的值或指针。在一些示例中,基于上下文的熵译码可以包括上下文自适应二进制算术译码(CABAC)。

[0266] 视频解码器300可以获得用于多个上下文中的上下文的预定义的初始化值(2204)。例如,视频解码器300的熵解码单元302可以获得initValue的值,该值可以是六比特变量。

[0267] 视频解码器300可以基于预定义的初始化值来在线性域中确定用于视频数据(例如,切片、瓦片等)的可独立译码单元的上下文的初始概率状态(2206)。例如,熵解码单元302可以在线性域中确定上下文的初始概率状态,而不在中间在对数域中确定上下文的初始概率状态。在一些示例中,熵解码单元302可以在不使用LUT将初始概率状态从对数域转换到线性域的情况下确定初始概率状态。

[0268] 为了确定初始概率状态,熵解码单元302可以获得用于可独立译码单元的量化参数的初始值(例如,SliceQP_y),并且获得不为零的量化参数锚定点的值(例如,QP_{anchor})。熵解码单元302可以基于用于可独立译码单元的量化参数的初始值与量化参数锚定点的值之间的差来确定初始概率状态。例如,熵解码单元302可以根据以下等式来确定初始概率状态:

[0269]
$$\text{InitProbState} = ((m * (\text{SliceQP}_y - \text{QP}_{\text{anchor}})) \gg \text{rshift}) + n$$

[0270] 其中,InitProbState是初始概率状态,SliceQP_y是量化参数的初始值,QP_{anchor}是量化参数锚定点,并且rshift是右移位值。

[0271] 如上所讨论的,在一些示例中,熵解码单元302可以被配置为以增加的精度针对极端概率分布执行初始化。例如,熵解码单元302可以基于预定义的初始化值来确定斜率索引值和偏移索引值。熵解码单元302可以进行以下操作:基于斜率索引值来确定m的值;以及基于偏移索引值来确定n的值。在一些示例中,熵解码单元302可以根据以下等式来确定n的值: $n = (\text{OffsetIdx} * 18) + 1$,其中OffsetIdx是偏移索引值。如上所示,在一些示例中,熵解码单元302可以基于m的值和n的值来确定初始概率状态。

[0272] 视频解码器300可以基于上下文的初始概率状态来对仓字符串的仓进行解码(2208)。视频解码器300可以基于经解码的仓和上下文的初始概率状态来确定上下文的经更新的概率状态。视频解码器300可以基于上下文的经更新的概率状态来对另一个仓进行解码(2206)。

[0273] 以下编号的示例可以说明本公开内容的一个或多个方面:

[0274] 示例1A。一种用于对视频数据进行熵译码的方法,所述方法包括:获得用于在对用于所述视频数据的切片中的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的斜率索引和预定义的偏移索引;基于所述预定义的斜率索引和所述预定义的偏移索引来确定用于所述视频数据的所述切片的所述上下文的初始概率状态;以及基于所述上下文的所述初始概率状态来对用于所述语法元素的所述值的仓进行熵译码。

[0275] 示例2A。根据示例1A所述的方法,其中,所述初始概率状态表示线性域中的初始概率。

[0276] 示例3A。根据示例1A所述的方法,其中,所述初始概率状态表示二次域中的初始概率。

[0277] 示例4A。根据示例3A所述的方法,其中,确定所述初始概率状态是不使用查找表(LUT)在所述预定义的斜率索引、所述预定义的偏移索引和所述初始概率状态之间进行映射的情况下执行的。

[0278] 示例5A。根据示例1A-4A的任何组合所述的方法,其中,所述初始概率状态由InitProbState来表示,所述斜率索引由SlopeIdx来表示,并且所述偏移索引由OffsetIdx来表示。

[0279] 示例6A。根据示例1A-5A中任一项所述的方法,其中,所述上下文自适应熵译码过程包括上下文自适应二进制算术译码(CABAC)过程或上下文自适应变长译码(CAVLC)过程。

[0280] 示例7A。根据示例1A-6A中任一项所述的方法,其中,译码包括解码。

[0281] 示例8A。根据示例1A-7A中任一项所述的方法,其中,译码包括编码。

[0282] 示例9A。一种用于对视频数据进行译码的设备,所述设备包括用于执行根据示例1A-8A中任一项所述的方法的一个或多个单元。

[0283] 示例10A。根据示例9A所述的设备,其中,所述一个或多个单元包括在电路中实现的一个或多个处理器。

[0284] 示例11A。根据示例9A和10A中任一项所述的设备,还包括:用于存储所述视频数据的存储器。

[0285] 示例12A。根据示例9A-11A中任一项所述的设备,还包括:被配置为显示经解码的视频数据的显示器。

[0286] 示例13A。根据示例9A-12A中任一项所述的设备,其中,所述设备包括相机、计算机、移动设备、广播接收机设备或机顶盒中的一者或多者。

[0287] 示例14A。根据示例9A-13A中任一项所述的设备,其中,所述设备包括视频解码器。

[0288] 示例15A。根据示例9A-14A中任一项所述的设备,其中,所述设备包括视频编码器。

[0289] 示例16A。一种具有存储在其上的指令的计算机可读存储介质,所述指令在被执行时使得一个或多个处理器执行根据示例1A-8A中任一项所述的方法。

[0290] 示例1B。一种用于对视频数据进行熵译码的方法,所述方法包括:获得用于在对用于所述视频数据的切片中的语法元素的值进行熵译码的上下文自适应熵译码过程中使用的多个上下文中的上下文的预定义的偏移索引,以;基于所述预定义的偏移索引来确定用于所述视频数据的所述切片的所述上下文的初始概率状态;以及基于所述上下文的所述初始概率状态来对用于所述语法元素的所述值的仓进行熵译码。

[0291] 示例2B。根据示例1B所述的方法,其中,所述初始概率状态表示线性域中的初始概率。

[0292] 示例3B。根据示例1B所述的方法,其中,所述初始概率状态表示二次域中的初始概率。

[0293] 示例4B。根据示例3B所述的方法,其中,确定所述初始概率状态是不使用查找表(LUT)在所述预定义的偏移索引和所述初始概率状态之间进行映射的情况下执行的。

[0294] 示例5B。根据示例1B-4B的任何组合所述的方法,其中,所述初始概率状态由InitProbState来表示,并且所述偏移索引由OffsetIdx来表示。

[0295] 示例6B。根据示例1B-5B中任一项所述的方法,其中,所述上下文自适应熵译码过程包括上下文自适应二进制算术译码(CABAC)过程或上下文自适应变长译码(CAVLC)过程。

[0296] 示例7B。根据示例1B-6B中任一项所述的方法,其中,译码包括解码。

[0297] 示例8B。根据示例1B-7B中任一项所述的方法,其中,译码包括编码。

[0298] 示例9B。一种用于对视频数据进行译码的设备,所述设备包括用于执行根据示例1B-8B中任一项所述的方法的一个或多个单元。

[0299] 示例10B。根据示例9B所述的设备,其中,所述一个或多个单元包括在电路中实现的一个或多个处理器。

[0300] 示例11B。根据示例9B和10B中任一项所述的设备,还包括:用于存储所述视频数据的存储器。

[0301] 示例12B。根据示例9B-11B中任一项所述的设备,还包括:被配置为显示经解码的视频数据的显示器。

[0302] 示例13B。根据示例9B-12B中任一项所述的设备,其中,所述设备包括相机、计算机、移动设备、广播接收机设备或机顶盒中的一者或多者。

[0303] 示例14B。根据示例9B-13B中任一项所述的设备,其中,所述设备包括视频解码器。

[0304] 示例15B。根据示例9B-14B中任一项所述的设备,其中,所述设备包括视频编码器。

[0305] 示例16B。一种具有存储在其上的指令的计算机可读存储介质,所述指令在被执行时使得一个或多个处理器执行根据示例1B-8B中任一项所述的方法。

[0306] 示例1C。一种用于对视频数据进行熵译码的方法,所述方法包括:基于变量的值来确定第一中间值;基于所述第一中间值来确定上下文模型的低精度概率状态的值,其中,确定所述低精度概率状态的值包括将所述第一中间值右移位;以及基于所述上下文模型的所述低精度概率状态的值来对语法元素的至少一个比特的值进行译码。

[0307] 示例2C。根据示例1C所述的方法,其中,确定所述第一中间值包括:根据以下等式来确定所述第一中间值: $q = \text{Clip3}(18, 46, \text{SliceQP}_Y) - 16$,其中,q是所述第一中间值,并且SliceQP_Y是所述变量的值。

[0308] 示例3C。根据示例1C或示例2C所述的方法,其中,确定所述上下文模型的所述低精度概率状态的值包括:根据以下等式来确定所述上下文模型的所述低精度概率状态的值: $\text{ProbabilityStateL} = 16 * a + ((b - a) * q >> 1) + 8$,其中,q是所述第一中间值,ProbabilityStateL是所述上下文模型的所述低精度概率状态的值,a是第二中间值,并且b是第三中间值。

[0309] 示例4C。根据示例3C所述的方法,还包括:根据以下等式来确定用于a和b的值: $a = (\text{initValue} >> 3) * 9$, $b = (\text{initValue} \& 7) * 9$,其中initValue是初始化值。

[0310] 示例5C。根据示例1C-4C中任一项所述的方法,还包括:基于所述上下文模型的所述低精度概率状态的值来确定所述上下文模型的高精度概率状态的值。

[0311] 示例6C。根据示例5C所述的方法,其中,确定所述上下文模型的所述高精度概率状态的值包括:根据以下等式来确定所述上下文模型的所述高精度概率状态的值: $\text{ProbabilityStateH} = \text{ProbabilityStateL} \ll 4$,其中,ProbabilityStateH是所述上下文模型的所述高精度概率状态的值,并且ProbabilityStateL是所述上下文模型的所述低精度概率状态的值。

[0312] 示例7C。根据示例1C-6C中任一项所述的方法,其中,所述上下文自适应熵译码过程包括上下文自适应二进制算术译码(CABAC)过程或上下文自适应变长译码(CAVLC)过程。

[0313] 示例8C。根据示例1C-7C中任一项所述的方法,其中,译码包括解码。

[0314] 示例9C。根据示例1C-8C中任一项所述的方法,其中,译码包括编码。

[0315] 示例10C。一种用于对视频数据进行译码的设备,所述设备包括用于执行根据示例1C-9C中任一项所述的方法的一个或多个单元。

[0316] 示例11C。根据示例10C所述的设备,其中,所述一个或多个单元包括在电路中实现的一个或多个处理器。

[0317] 示例12C。根据示例10C和11C中任一项所述的设备,还包括:用于存储所述视频数据的存储器。

[0318] 示例13C。根据示例10C-12C中任一项所述的设备,还包括:被配置为显示经解码的视频数据的显示器。

[0319] 示例14C。根据示例10C-13C中任一项所述的设备,其中,所述设备包括相机、计算机、移动设备、广播接收机设备或机顶盒中的一者或多者。

[0320] 示例15C。根据示例10C-14C中任一项所述的设备,其中,所述设备包括视频解码器。

[0321] 示例16C。根据示例10C-15C中任一项所述的设备,其中,所述设备包括视频编码器。

[0322] 示例17C。一种具有存储在其上的指令的计算机可读存储介质,所述指令在被执行时使得一个或多个处理器执行根据示例1C-9C中任一项所述的方法。

[0323] 要认识到的是,根据示例,本文描述的技术中的任何技术的某些动作或事件可以以不同的顺序执行,可以被添加、合并或完全省略(例如,并非全部描述的动作或事件对于所述技术的实践是必要的)。此外,在某些示例中,动作或事件可以例如通过多线程处理、中断处理或多个处理器并发地而不是顺序地执行。

[0324] 在一个或多个示例中,所描述的功能可以在硬件、软件、固件或其任何组合中实现。如果在软件中实现,则所述功能可以作为一个或多个指令或代码存储在计算机可读介质上或者通过其进行发送以及由基于硬件的处理单元执行。计算机可读介质可以包括计算机可读存储介质,其对应于诸如数据存储介质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来促进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)非暂时性的有形计算机可读存储介质、或者(2)诸如信号或载波的通信介质。数据存储介质可以由一个或多个计算机或者一个或多个处理器访问以取得用于实现在本公开内容中描述的技术的指令、代码和/或数据结构的任何可用的介质。计算机程序产品可以包括计算机可读介质。

[0325] 举例来说而非进行限制,这样的计算机可读存储介质可以包括RAM、ROM、EEPROM、CD-ROM或其它光盘存储、磁盘存储或其它磁存储设备、闪存、或者能够用于以指令或数据结构形式存储期望的程序代码以及能够由计算机访问的任何其它介质。此外,任何连接被适当地称为计算机可读介质。例如,如果使用同轴电缆、光纤光缆、双绞线、数字用户线(DSL)或者无线技术(诸如红外线、无线电和微波)来从网站、服务器或其它远程源发送指令,则同轴电缆、光纤光缆、双绞线、DSL或者无线技术(诸如红外线、无线电和微波)被包括在介质的定义中。然而,应当理解的是,计算机可读存储介质和数据存储介质不包括连接、载波、信号或其它暂时性介质,而是替代地针对非暂时性的有形存储介质。如本文所使用的,磁盘和光

盘包括压缩光盘(CD)、激光光盘、光盘、数字多功能光盘(DVD)、软盘和蓝光光盘,其中,磁盘通常磁性地复制数据,而光盘利用激光来光学地复制数据。上述各项的组合也应当被包括在计算机可读介质的范围之内。

[0326] 指令可以由一个或多个处理器来执行,诸如一个或多个数字信号处理器(DSP)、通用微处理器、专用集成电路(ASIC)、现场可编程门阵列(FPGA)、或其它等效的集成或分立逻辑电路。相应地,如本文所使用的术语“处理器”和“处理电路”可以指代前述结构中的任何一者或者适于实现本文描述的技术的任何其它结构。另外,在一些方面中,本文描述的功能可以在被配置用于编码和解码的专用硬件和/或软件模块内提供,或者被并入经组合的编解码器中。此外,所述技术可以充分地在一个或多个电路或逻辑元件中实现。

[0327] 本公开内容的技术可以在各种各样的设备或装置中实现,包括无线手机、集成电路(IC)或一组IC(例如,芯片组)。在本公开内容中描述了各个组件、模块或单元以强调被配置为执行所公开的技术的设备的功能性方面,但是不一定需要通过不同的硬件单元来实现。而是,如上所述,各个单元可以被组合在编解码器硬件单元中,或者由可互操作的硬件单元的集合(包括如上所述的一个或多个处理器)结合适当的软件和/或固件来提供。

[0328] 已经描述了各个示例。这些和其它示例在以下权利要求的范围内。

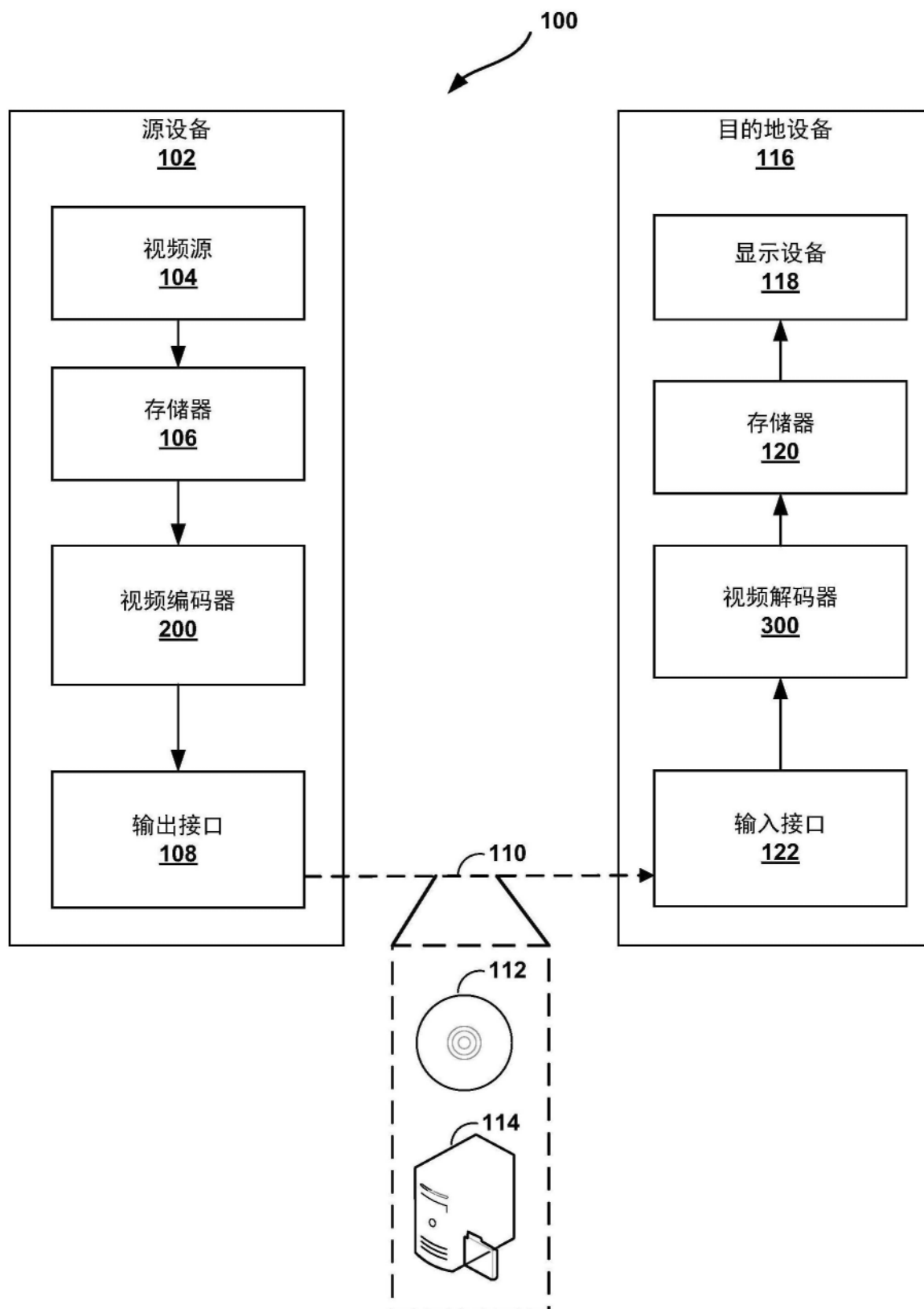


图1

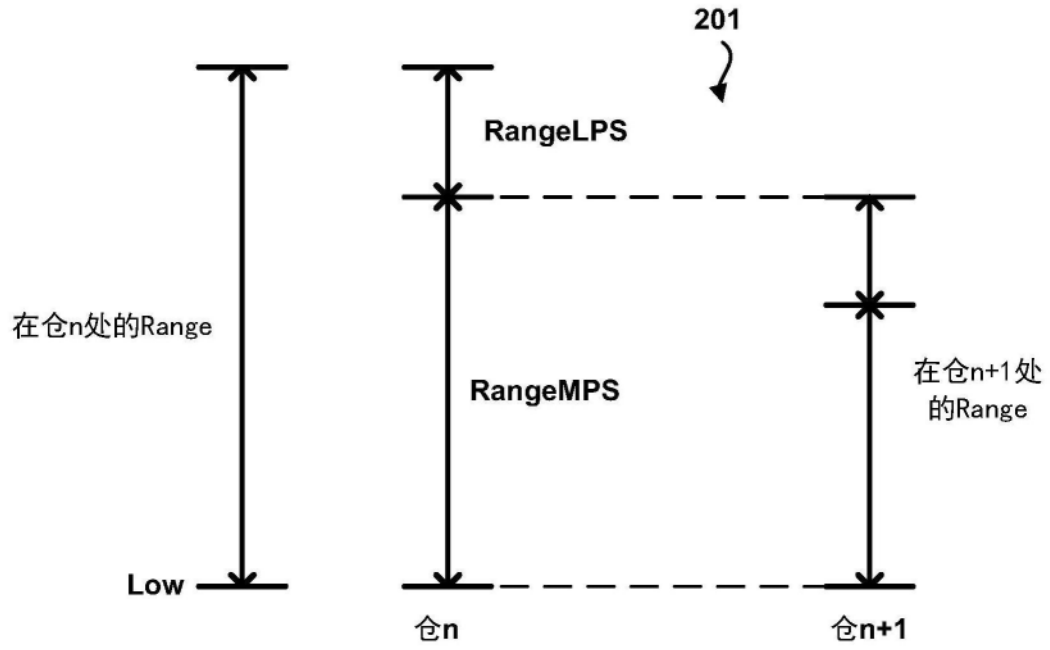


图2A

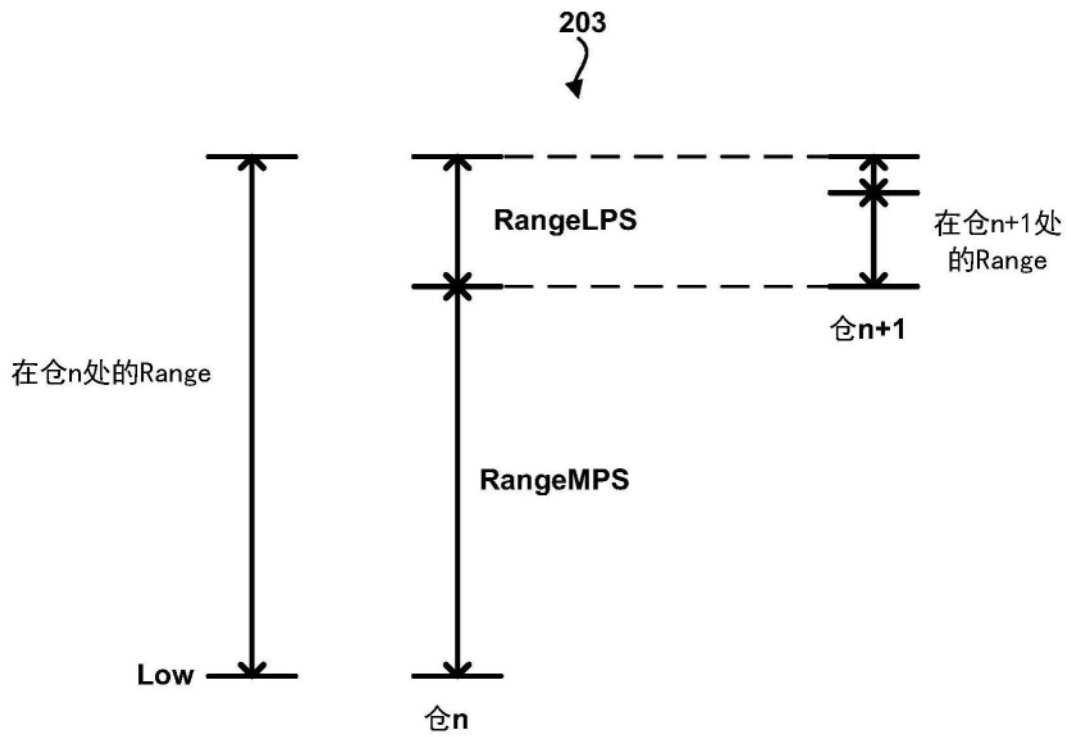


图2B

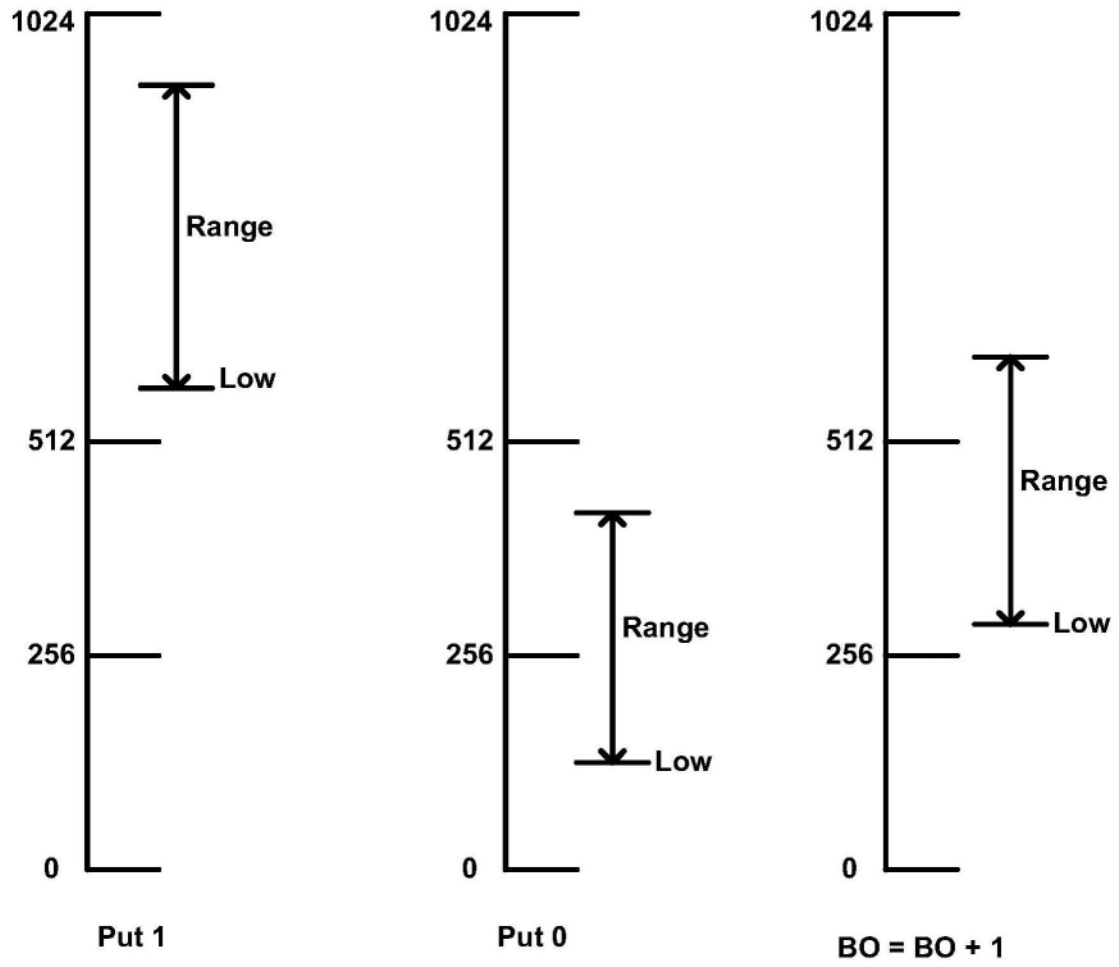


图3

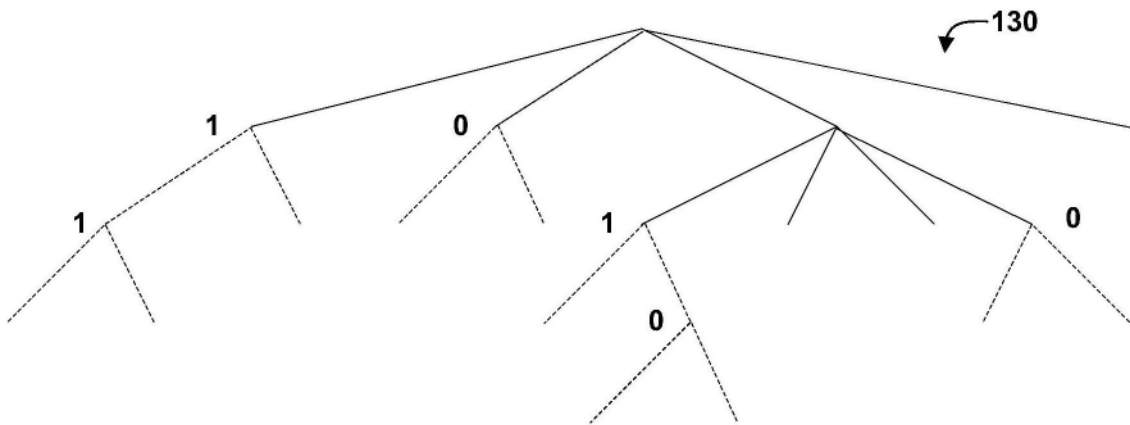


图4A

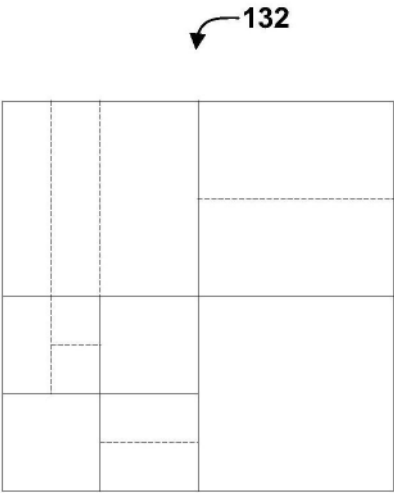


图4B

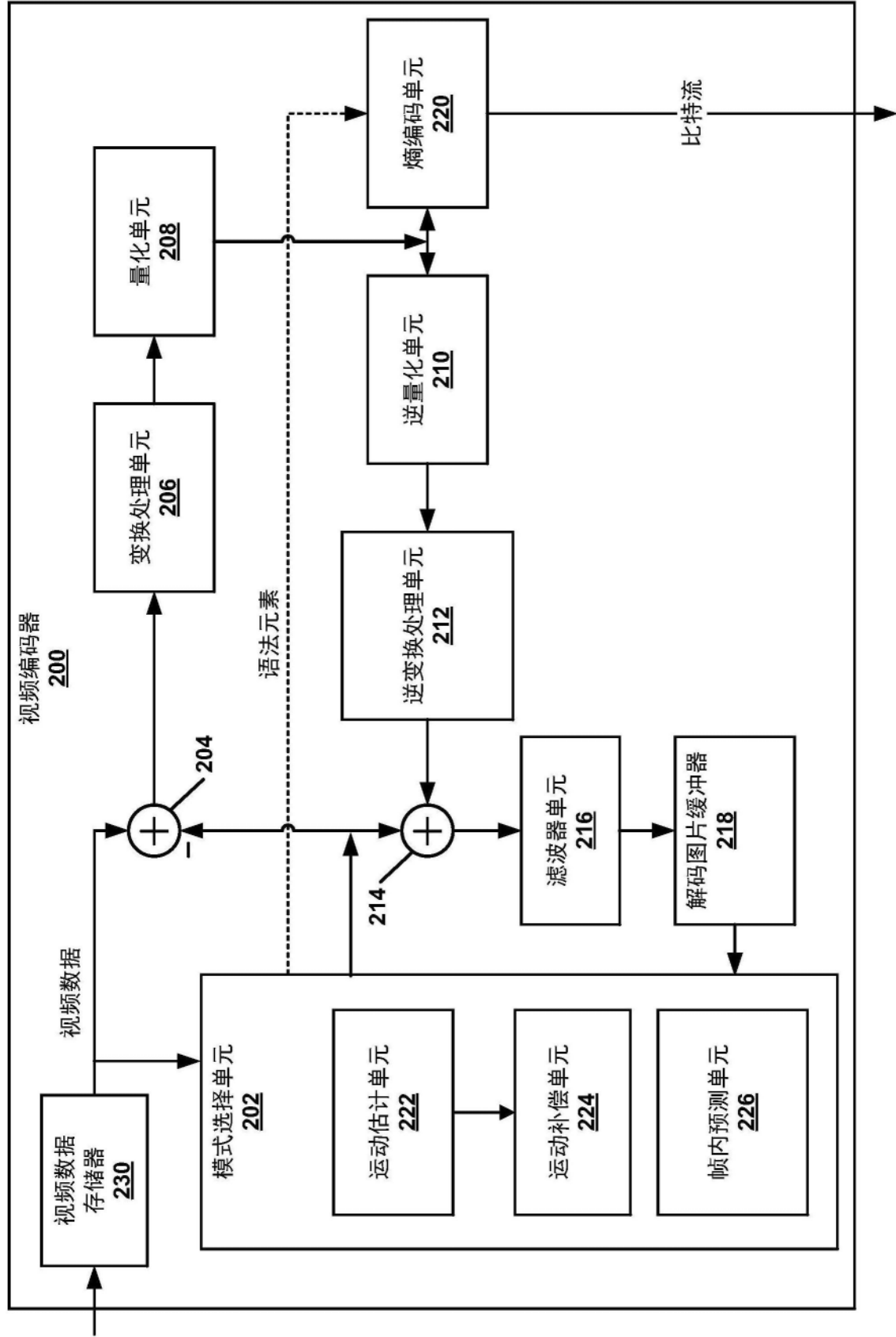
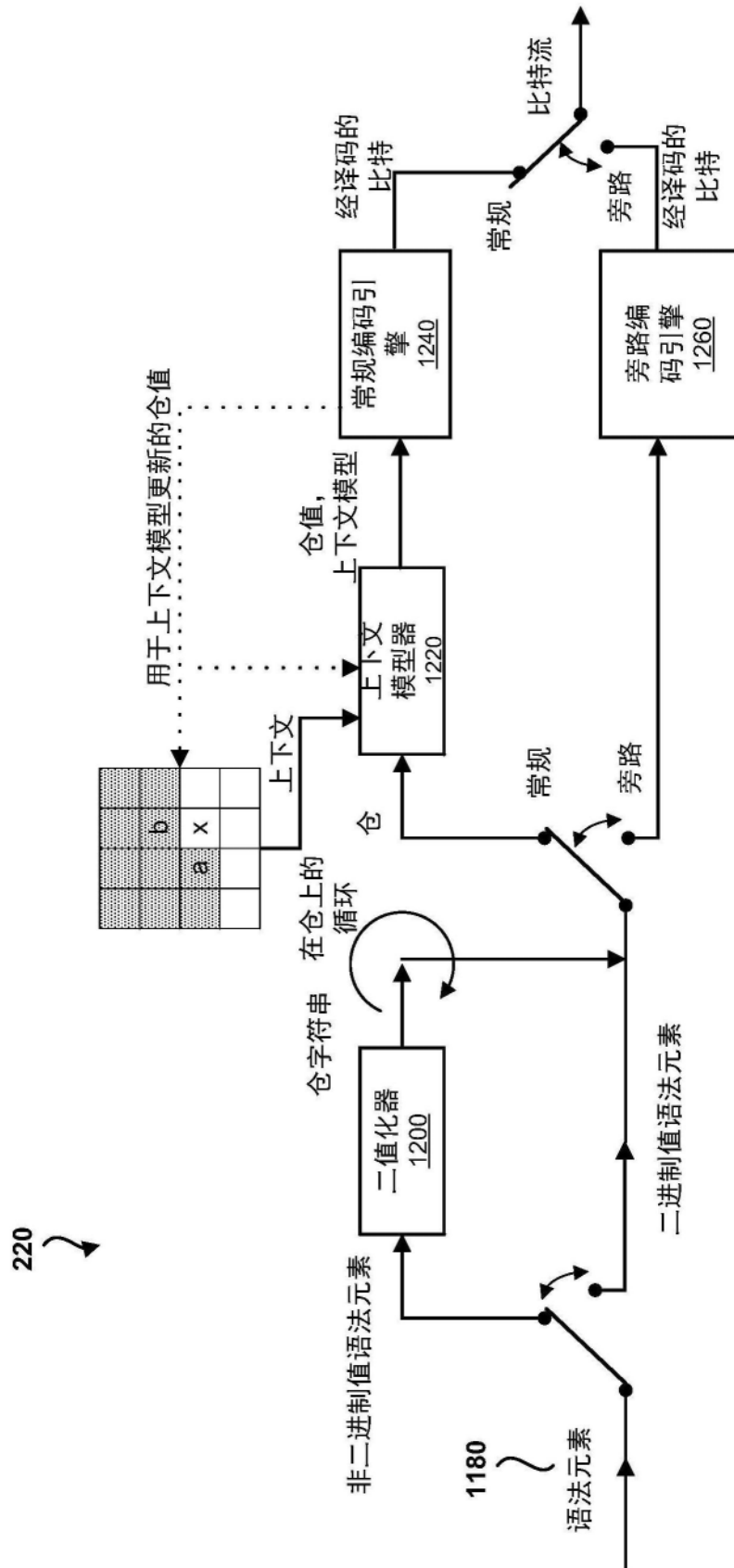


图5



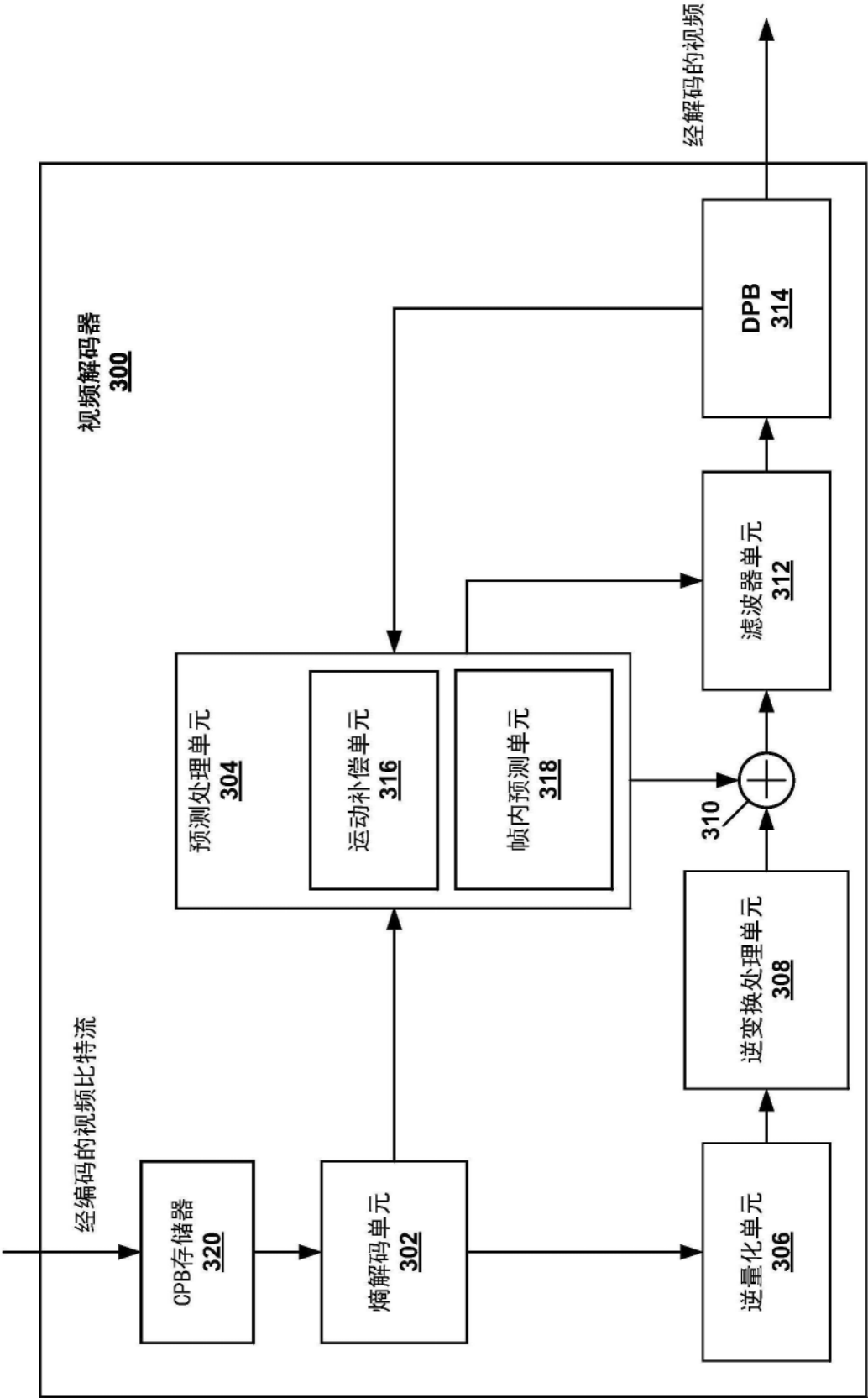


图7

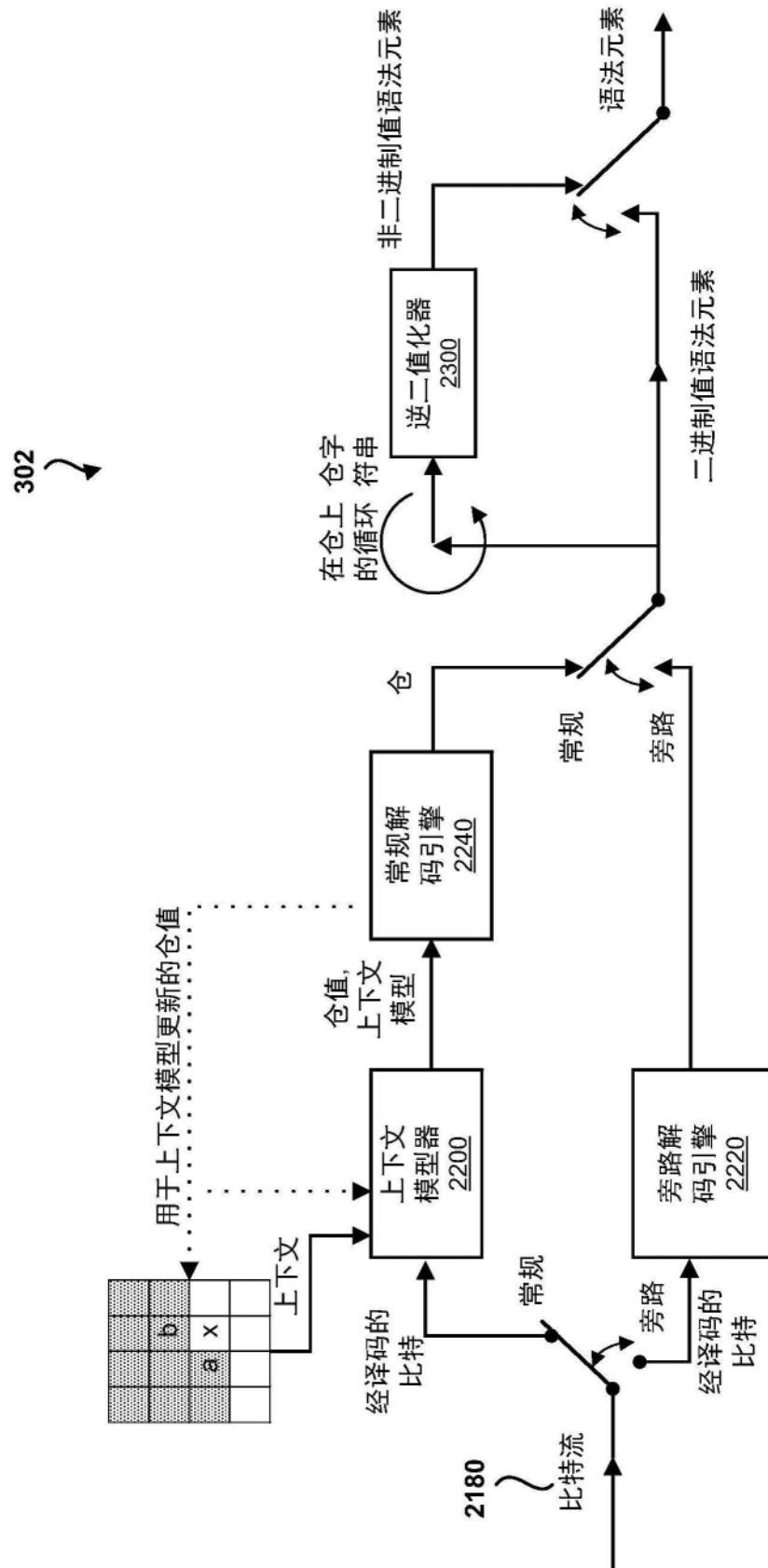


图8

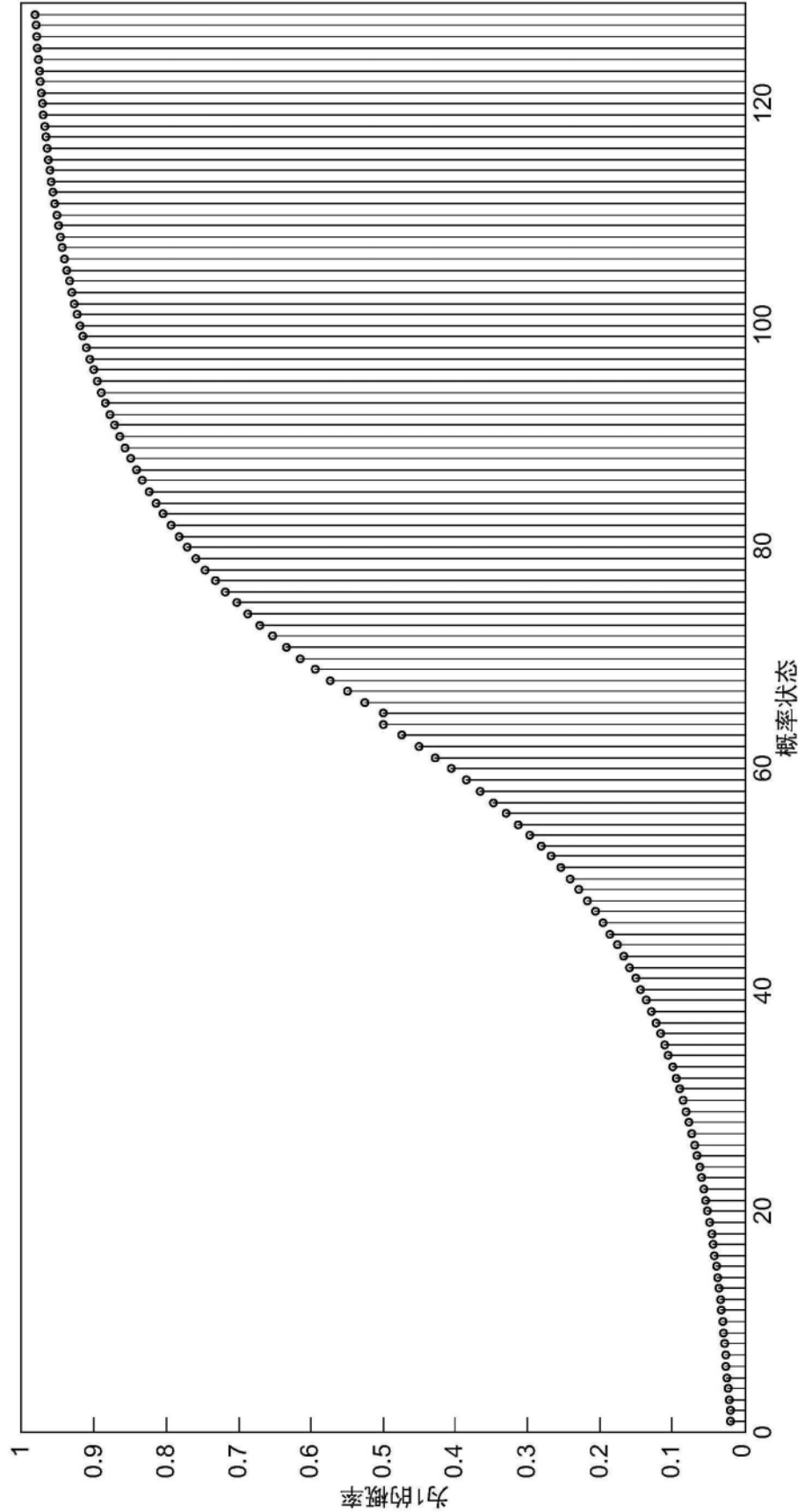


图9

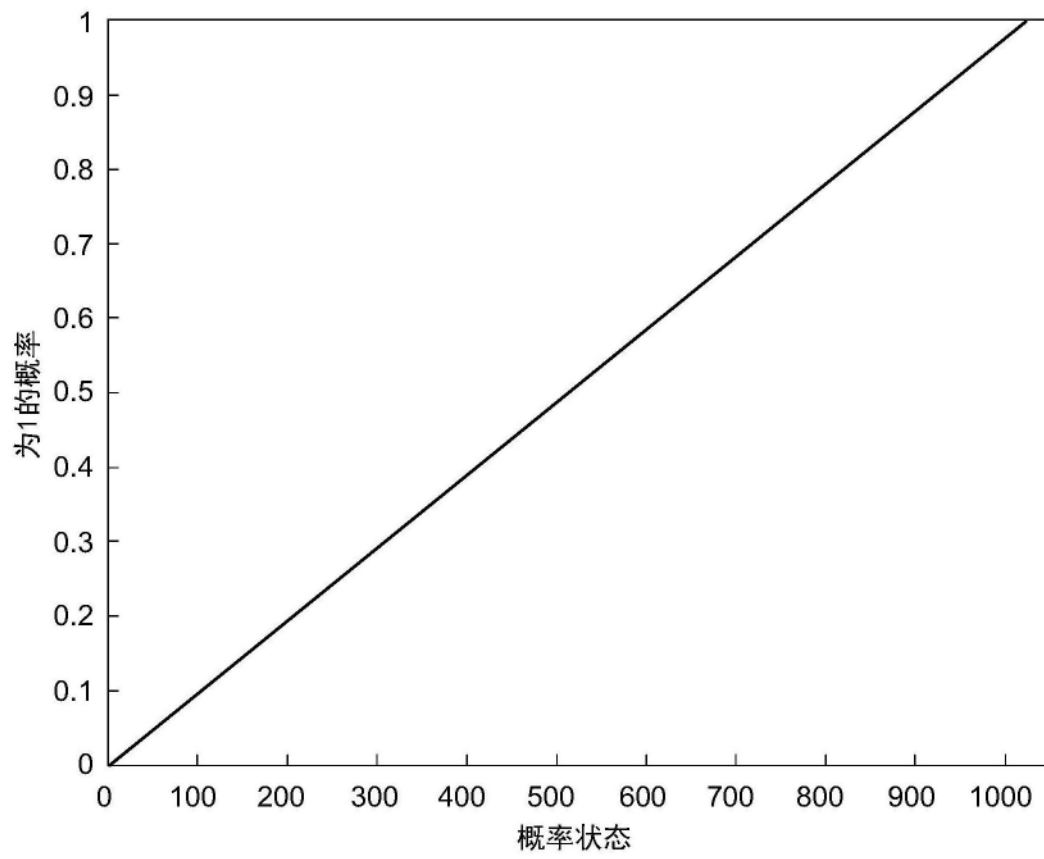


图10A

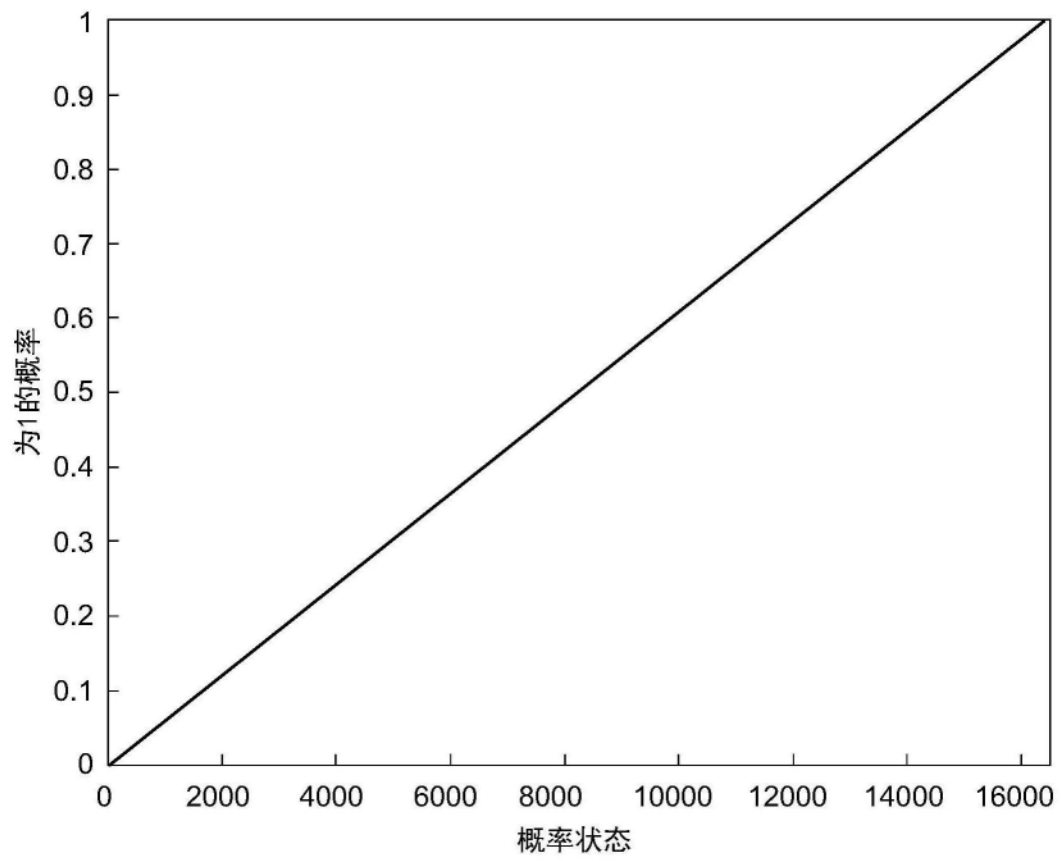


图10B

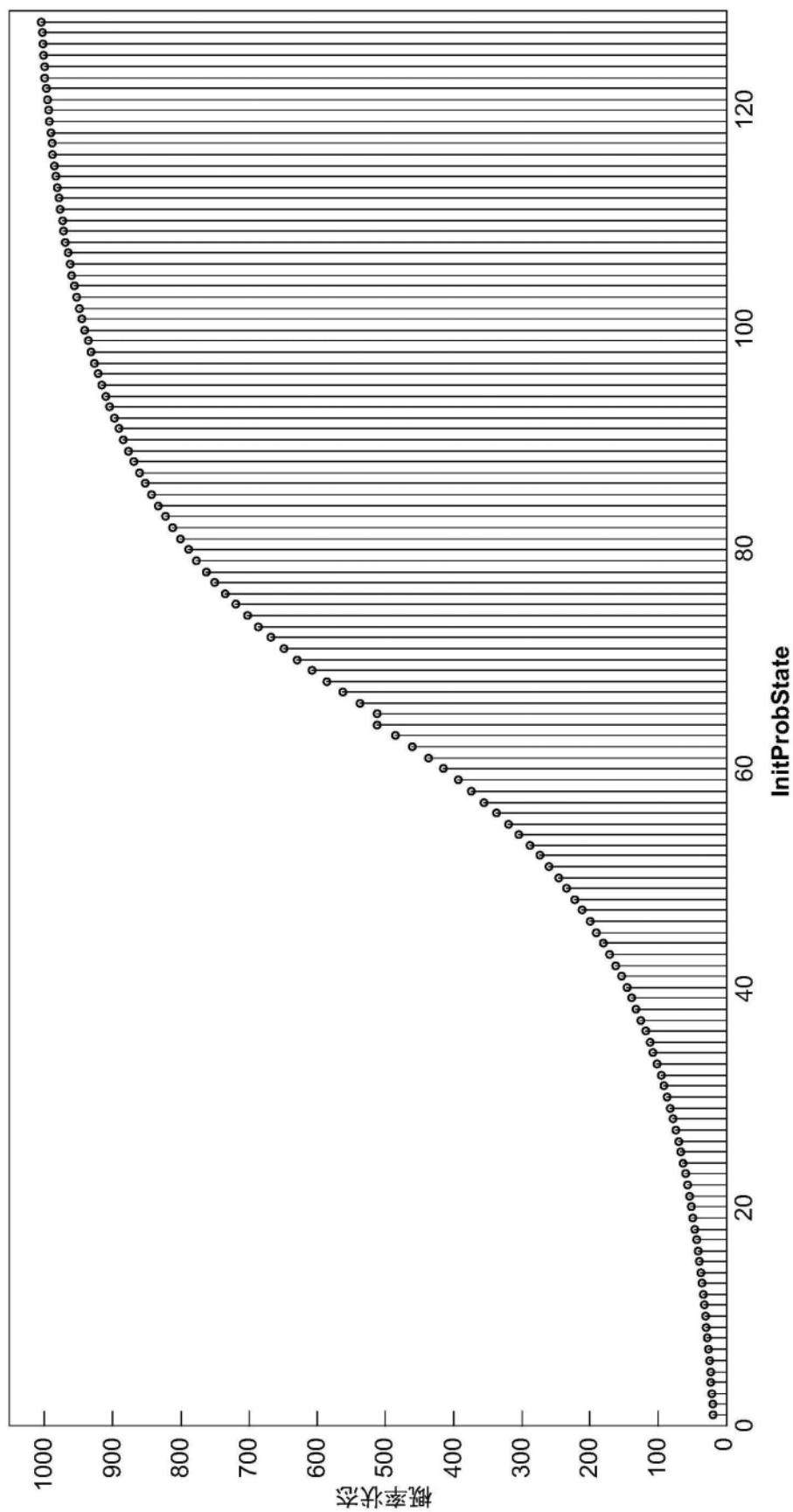


图11A

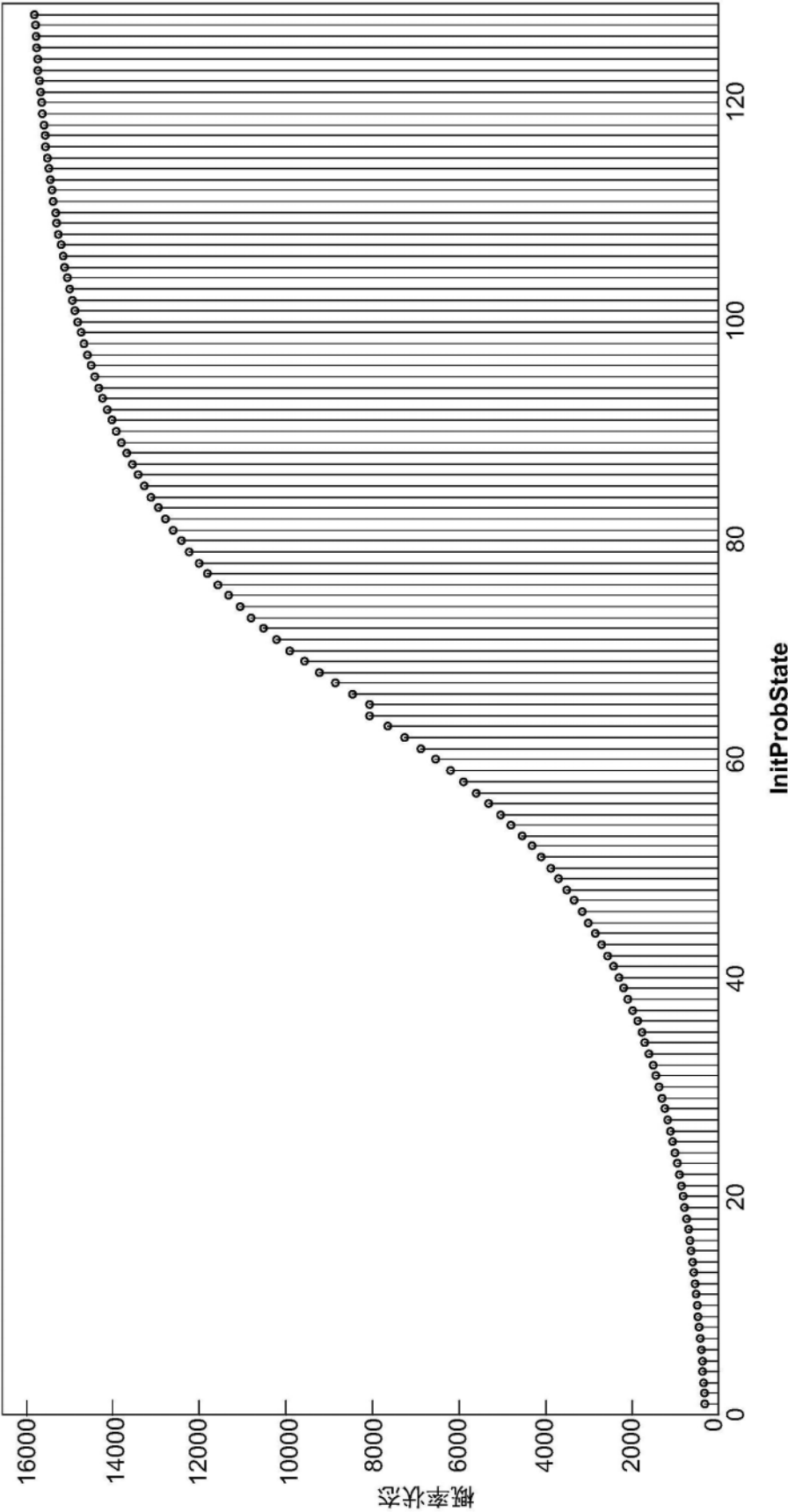


图11B

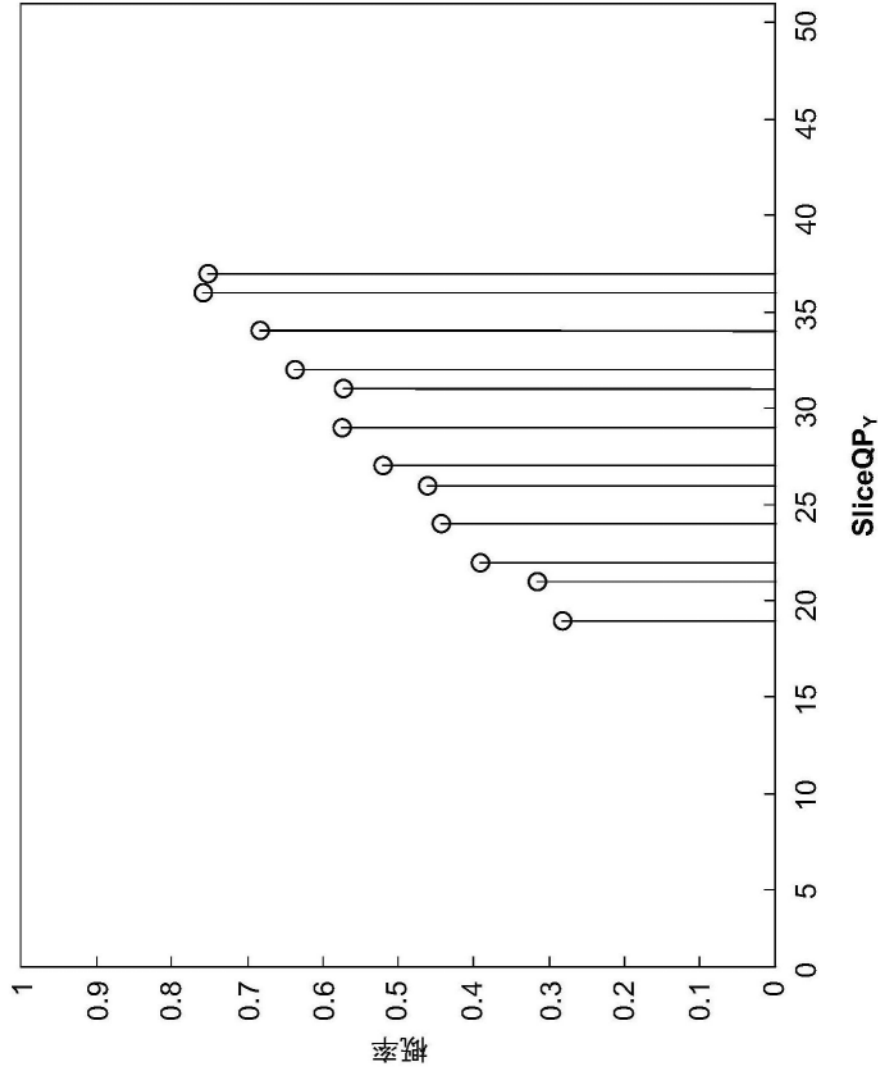


图12

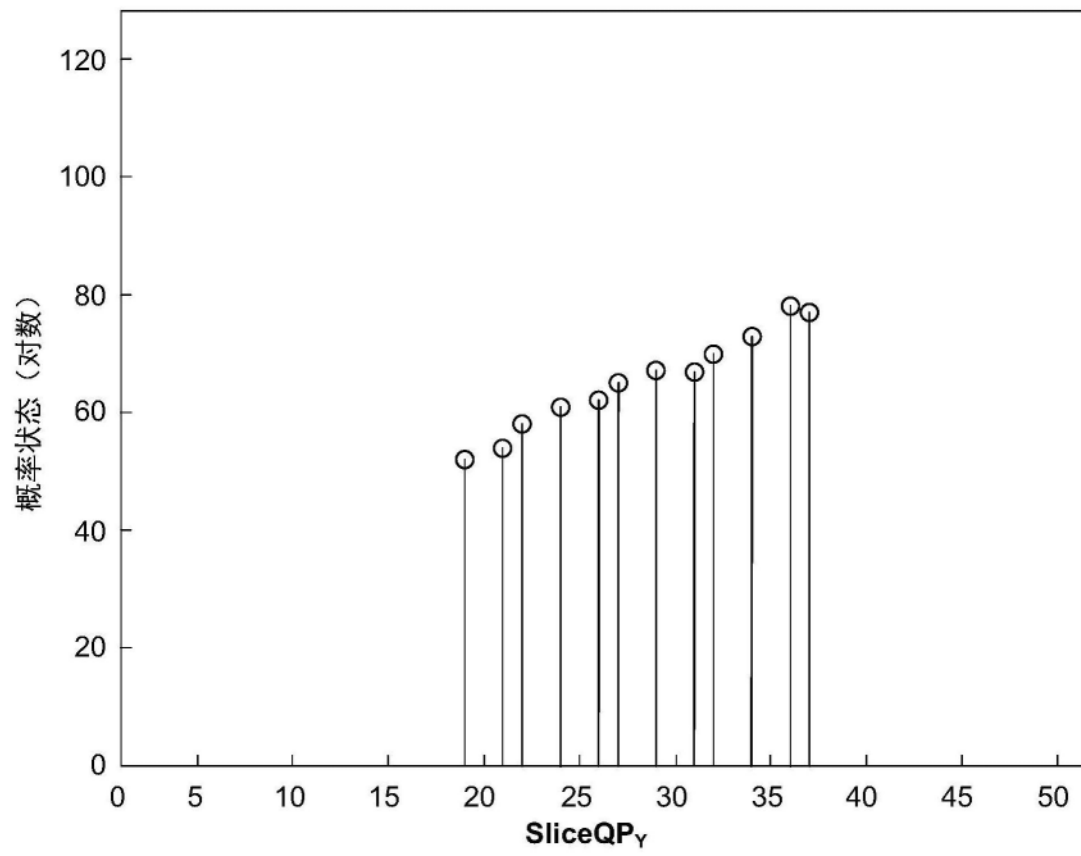


图13A

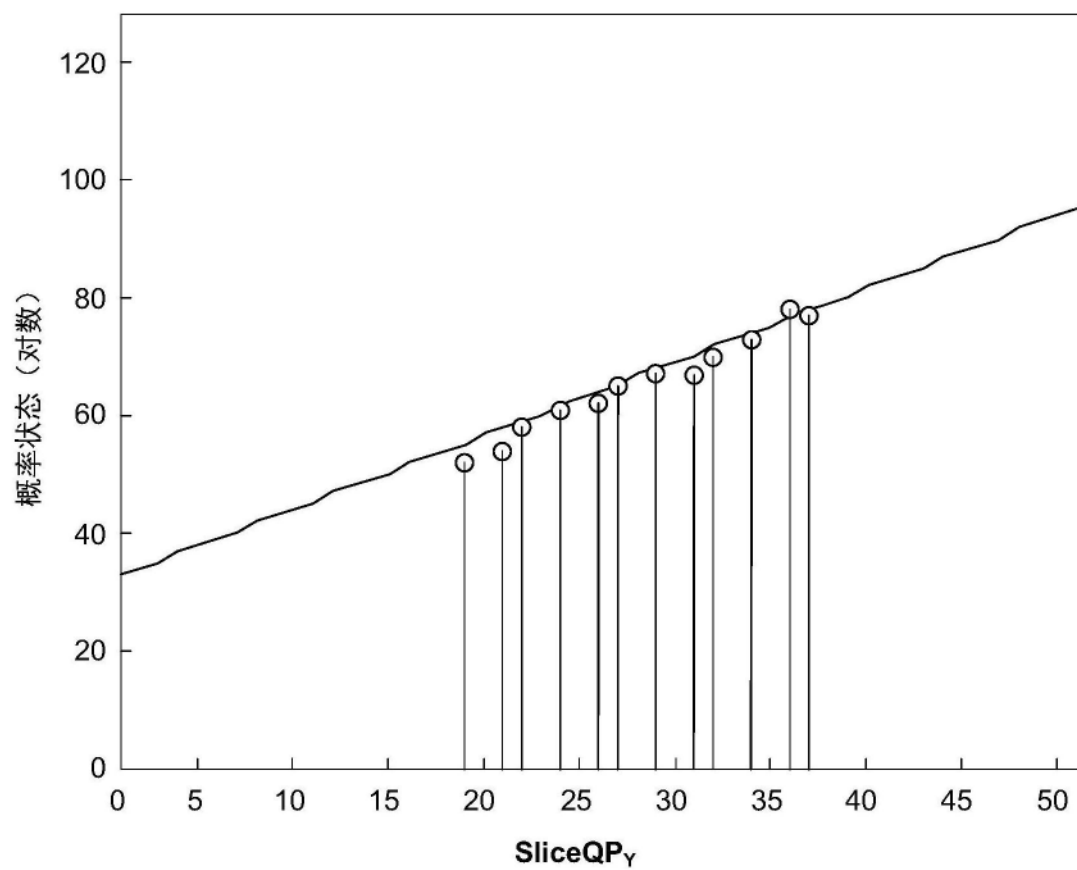


图13B

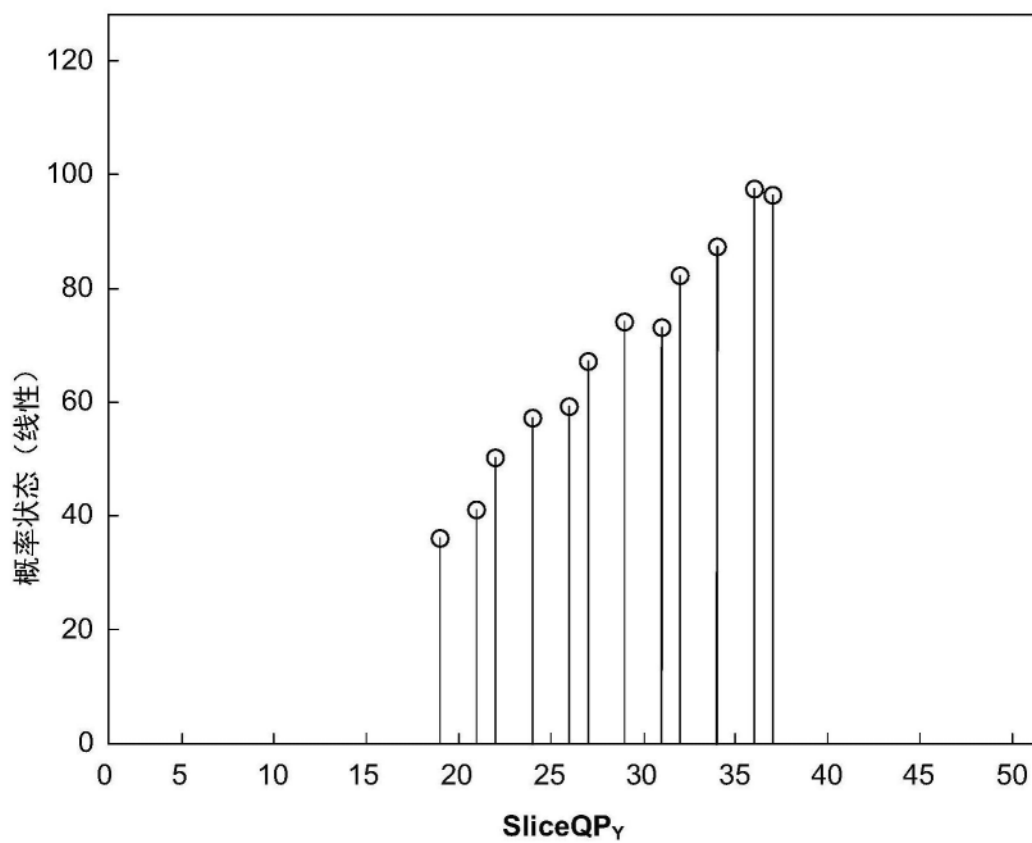


图14A

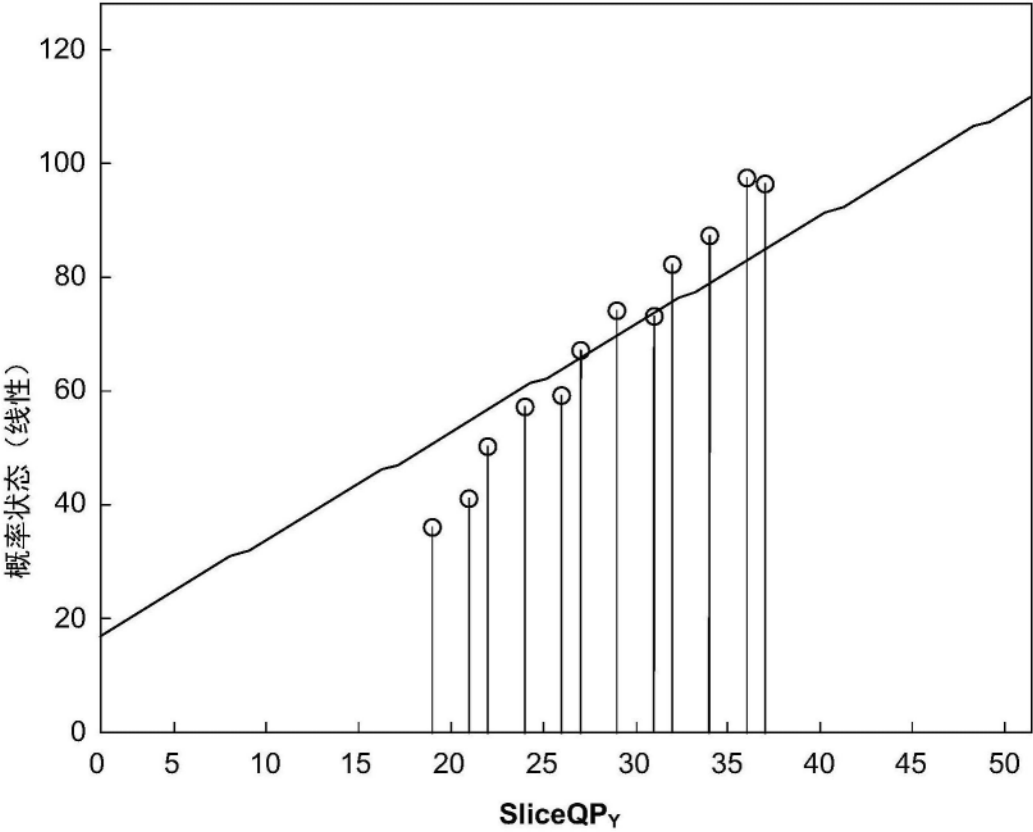


图14B

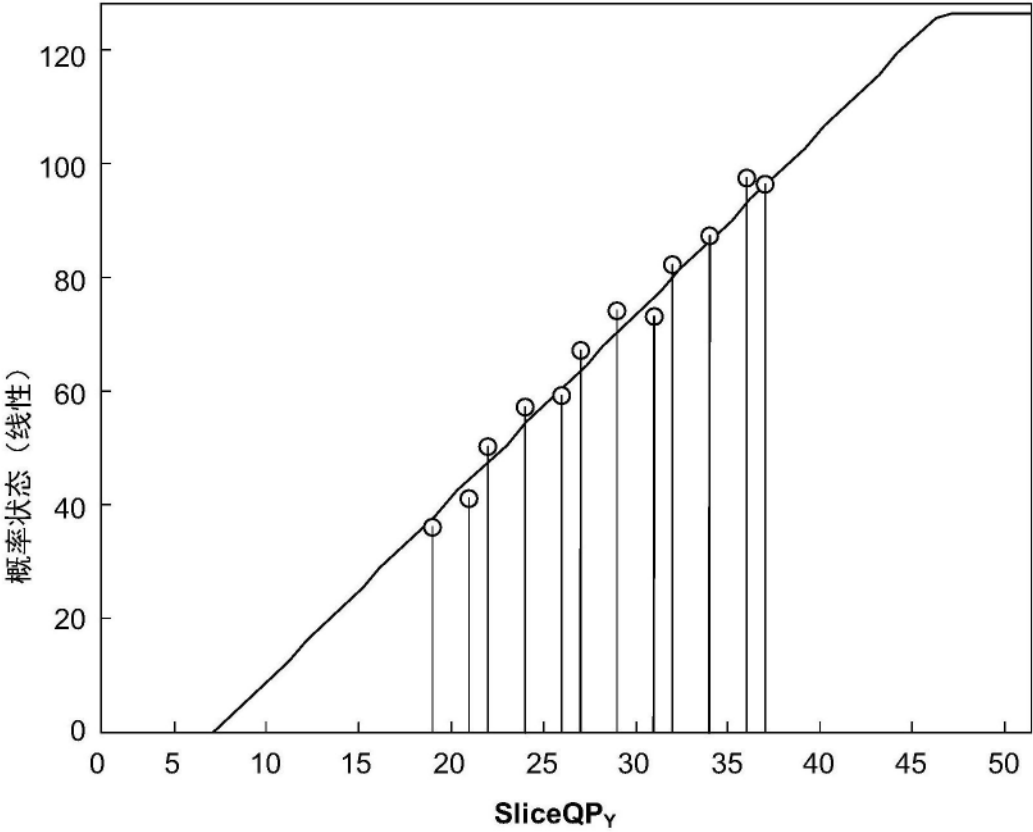


图14C

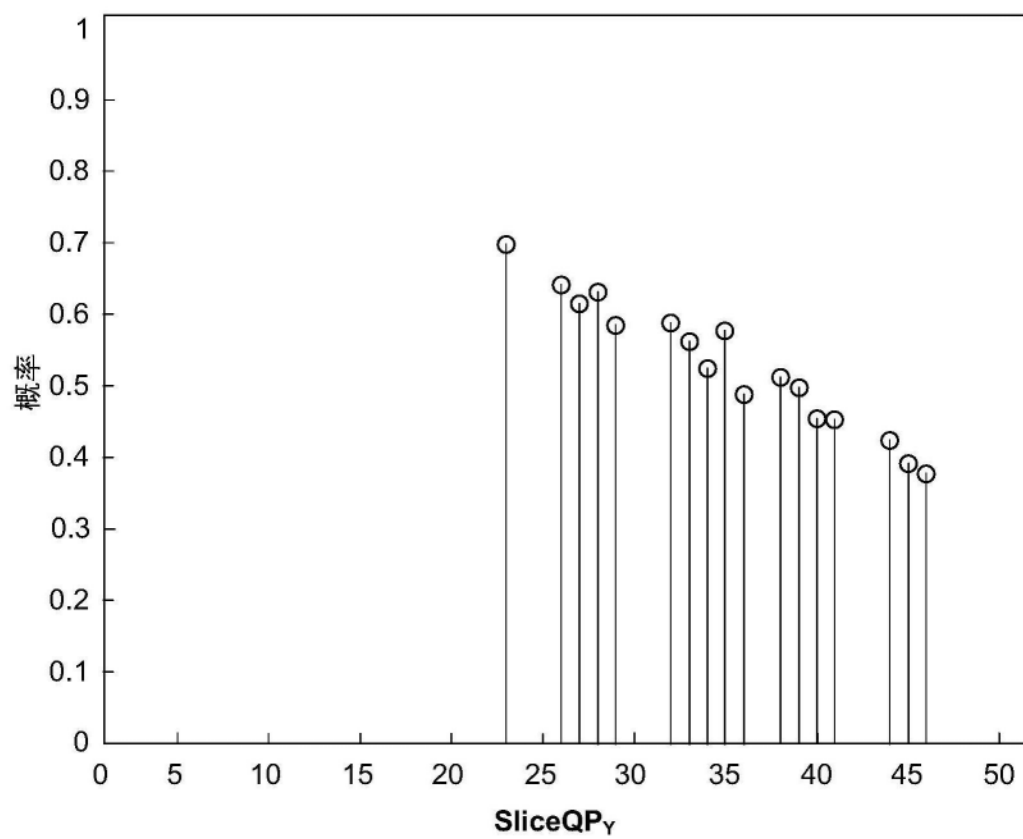


图15A

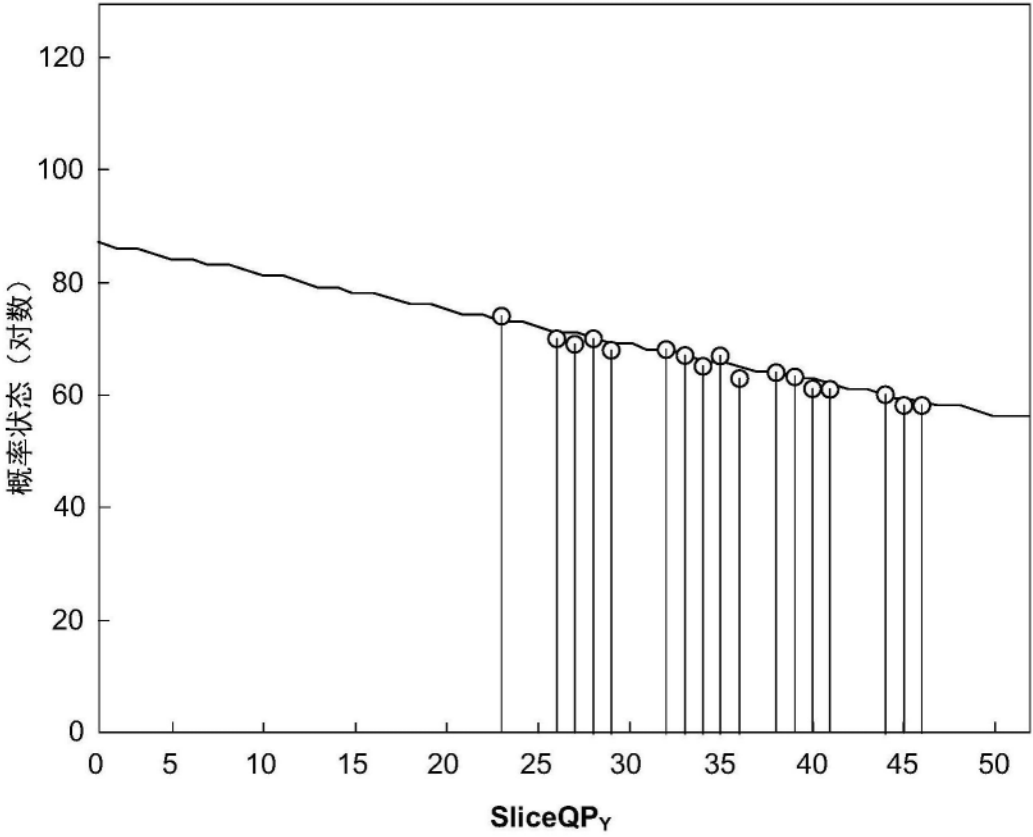


图15B

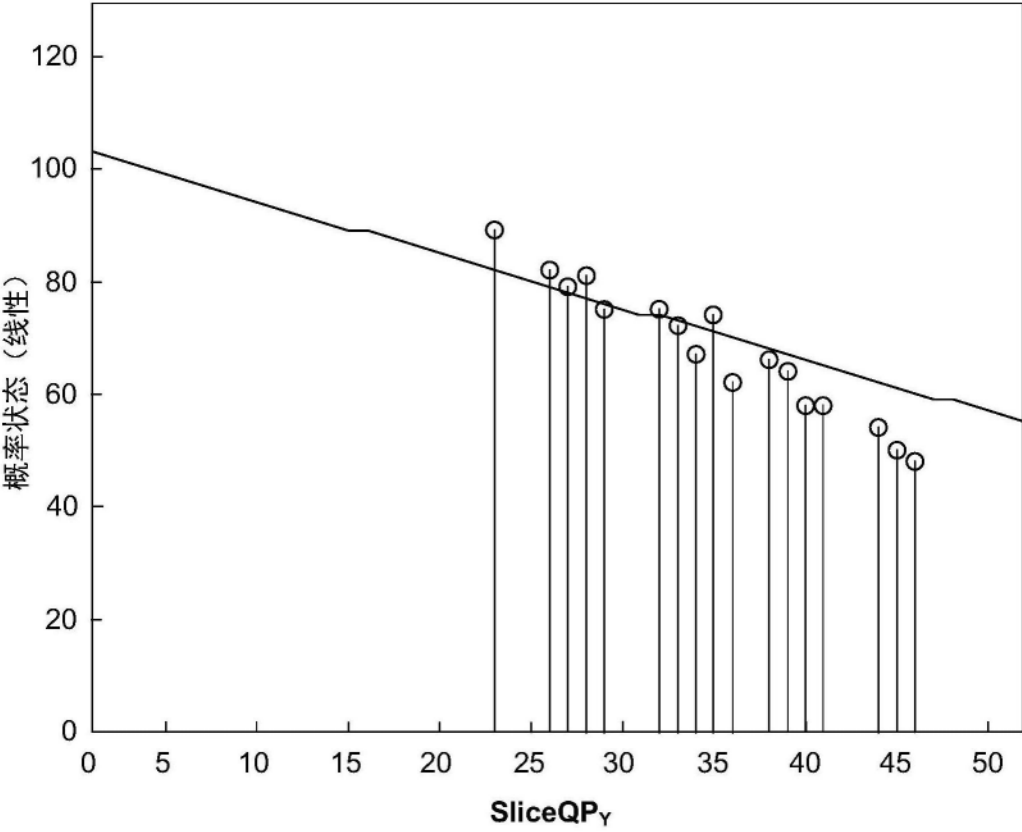


图15C

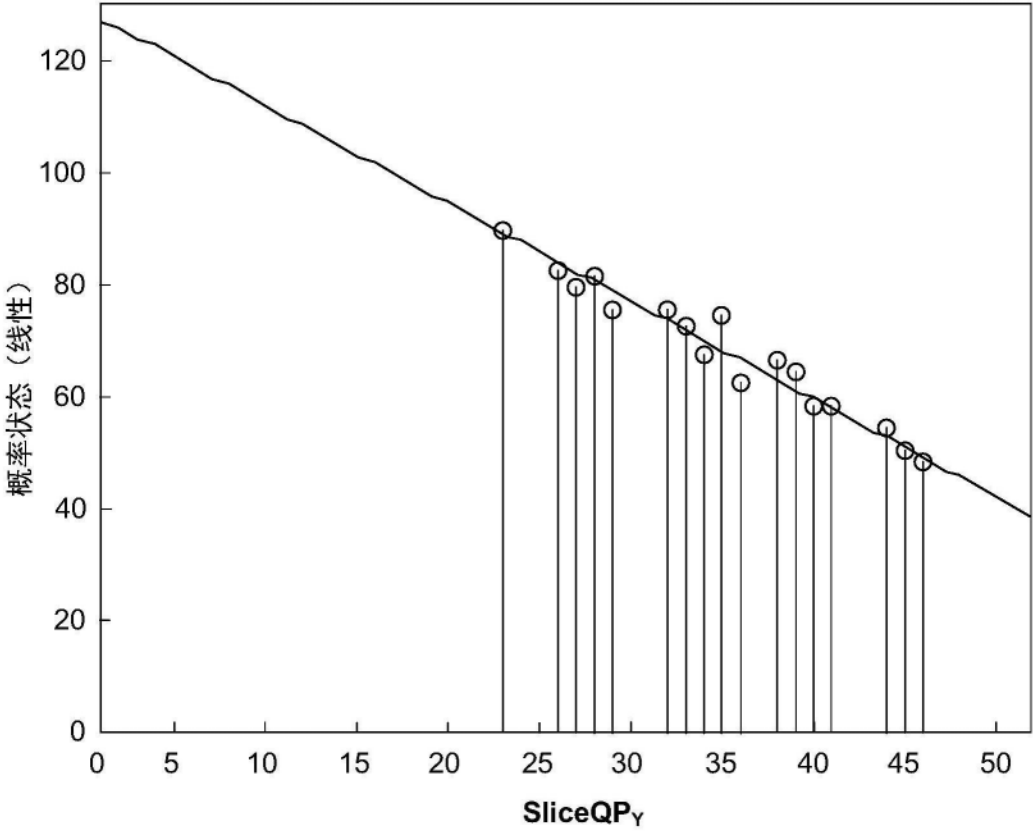


图15D

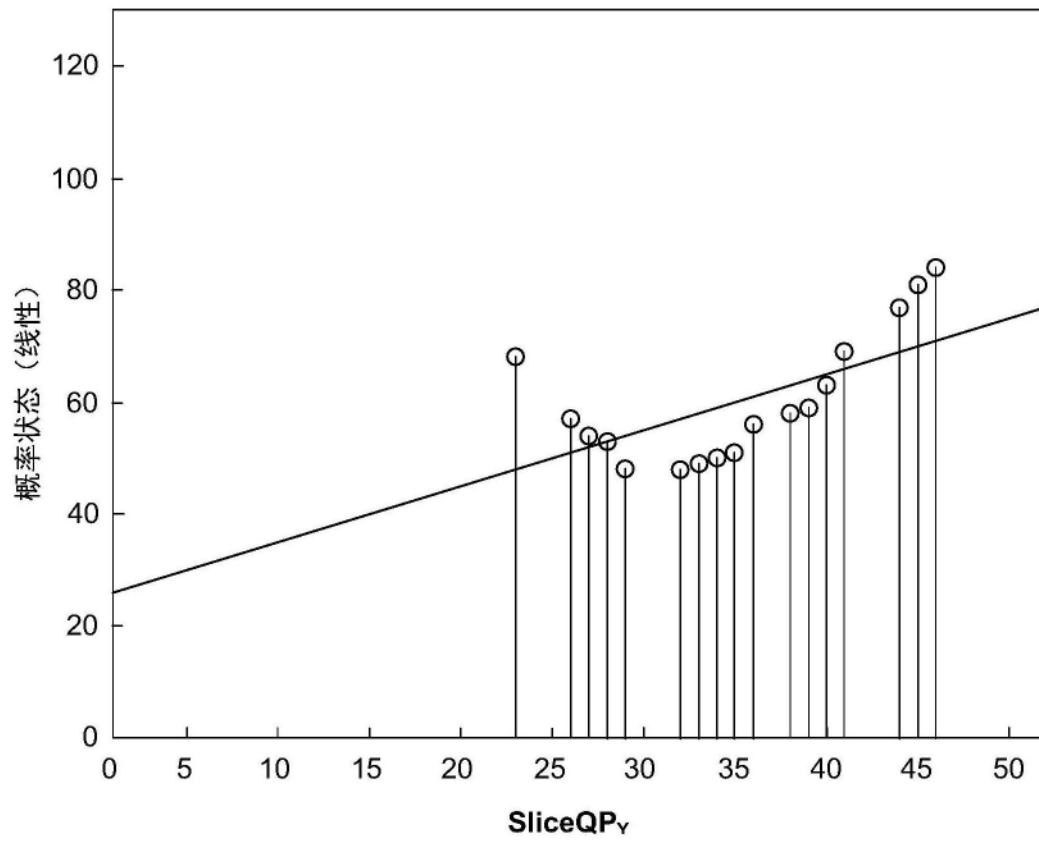


图16A

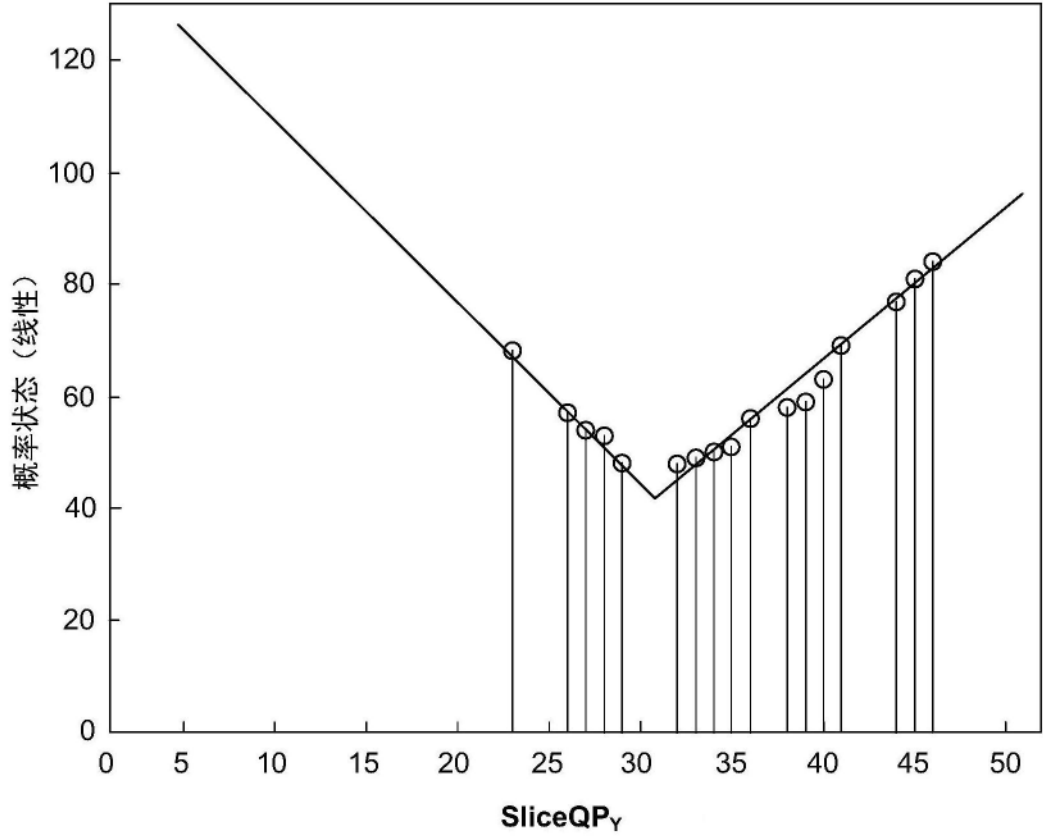


图16B

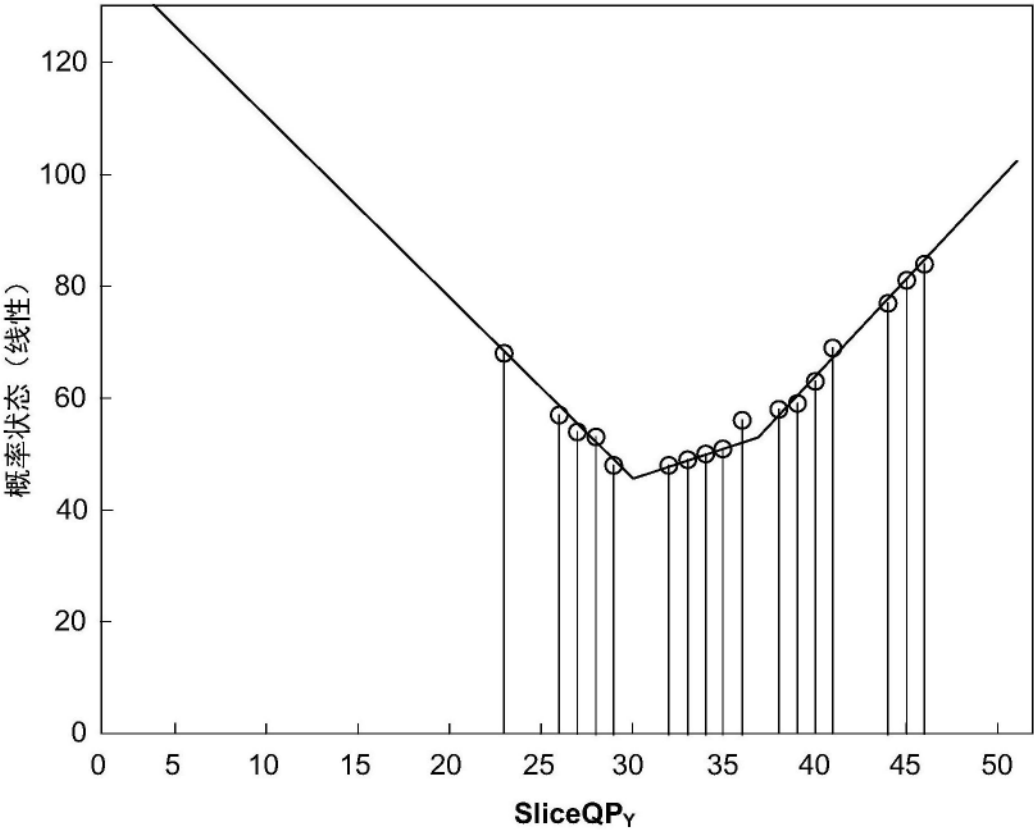


图16C

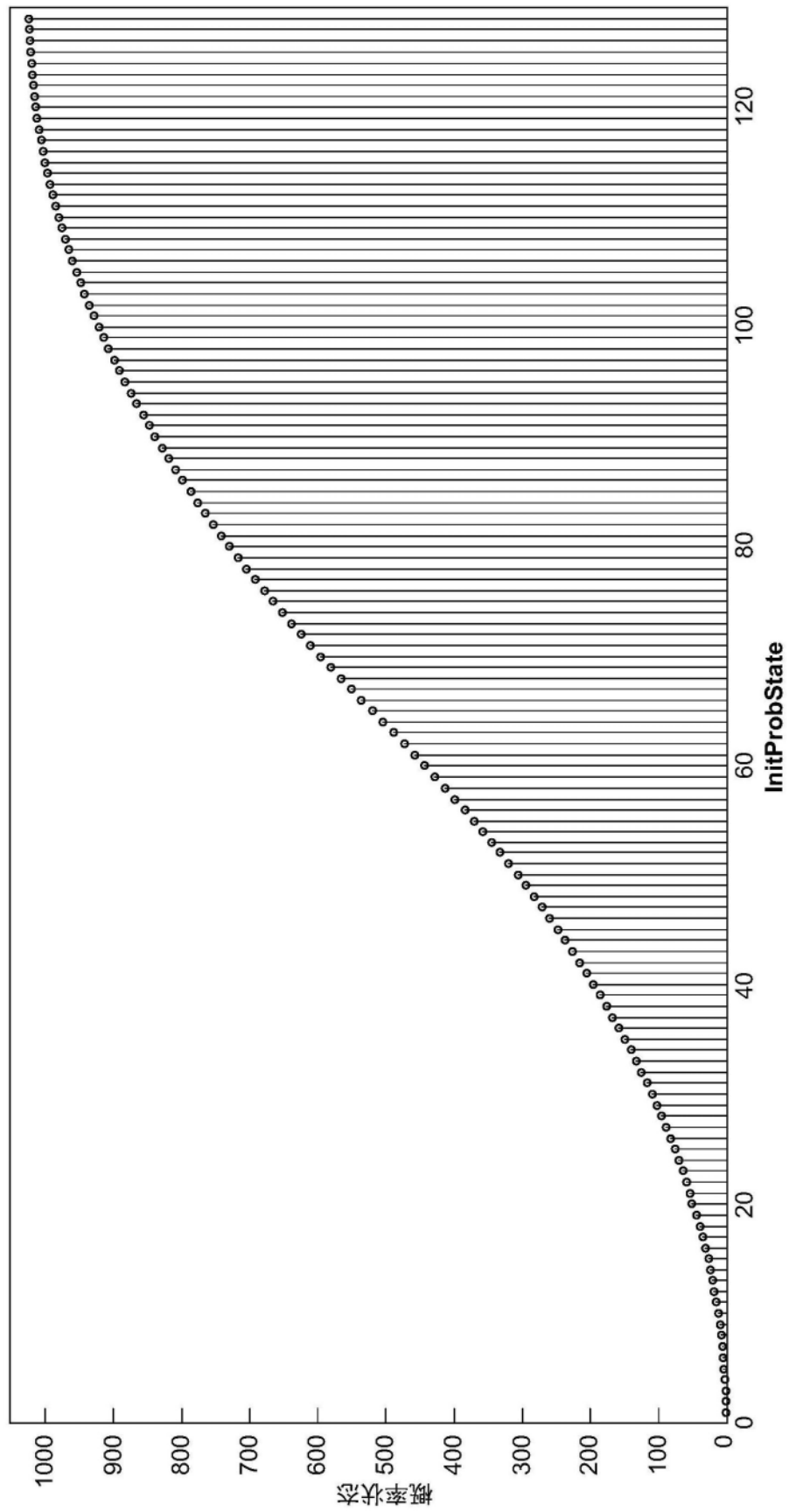


图17A

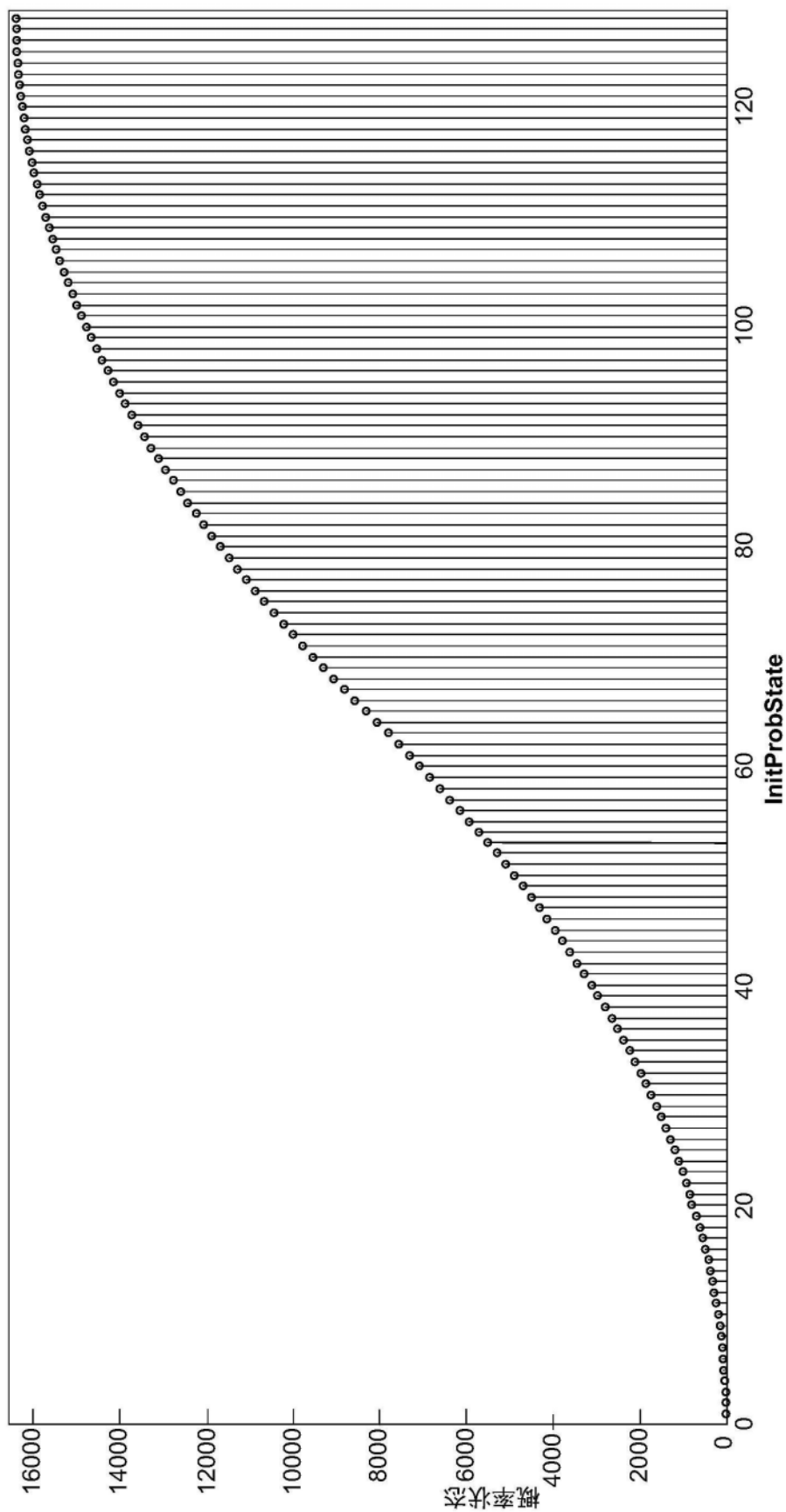


图17B

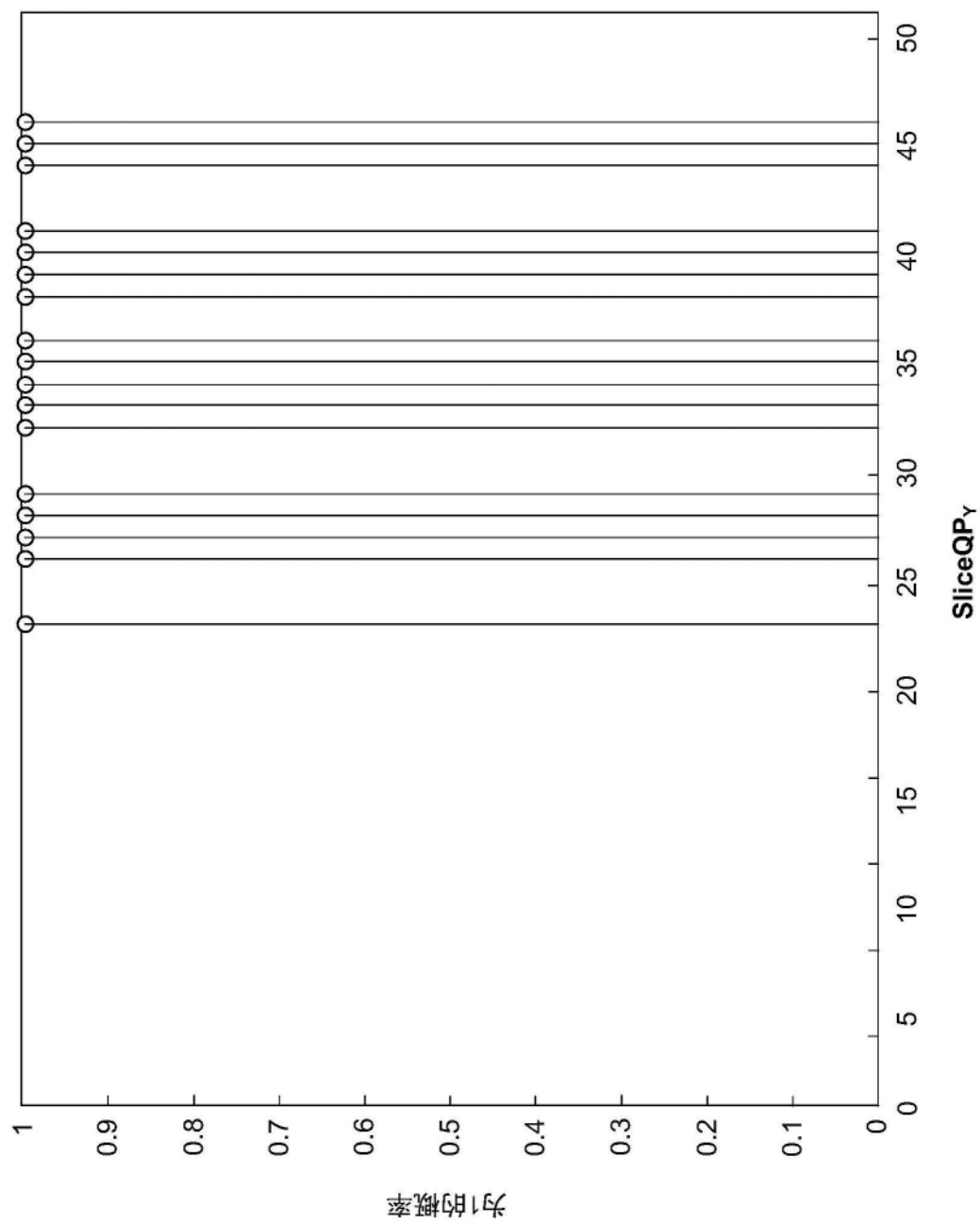


图18A

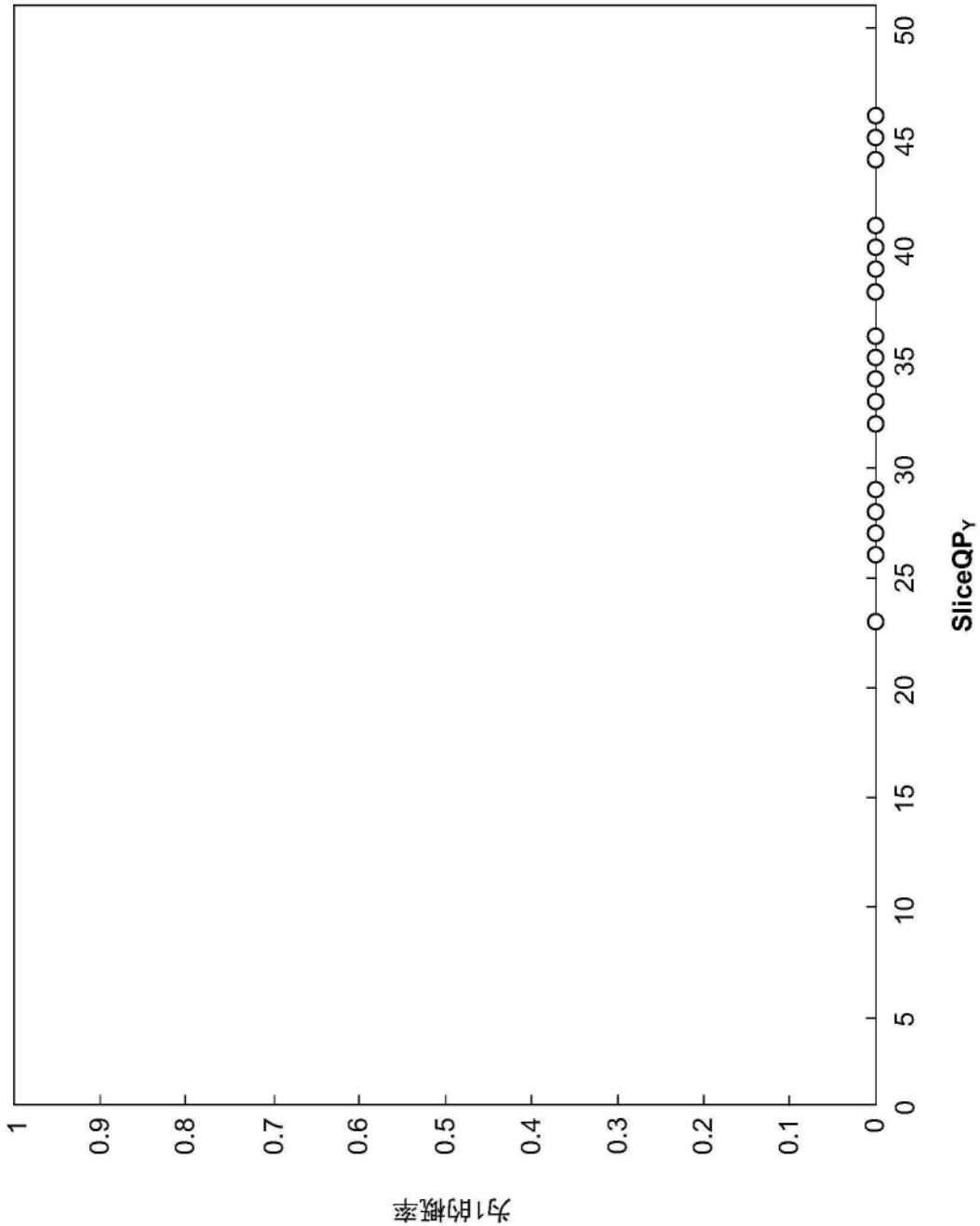


图18B

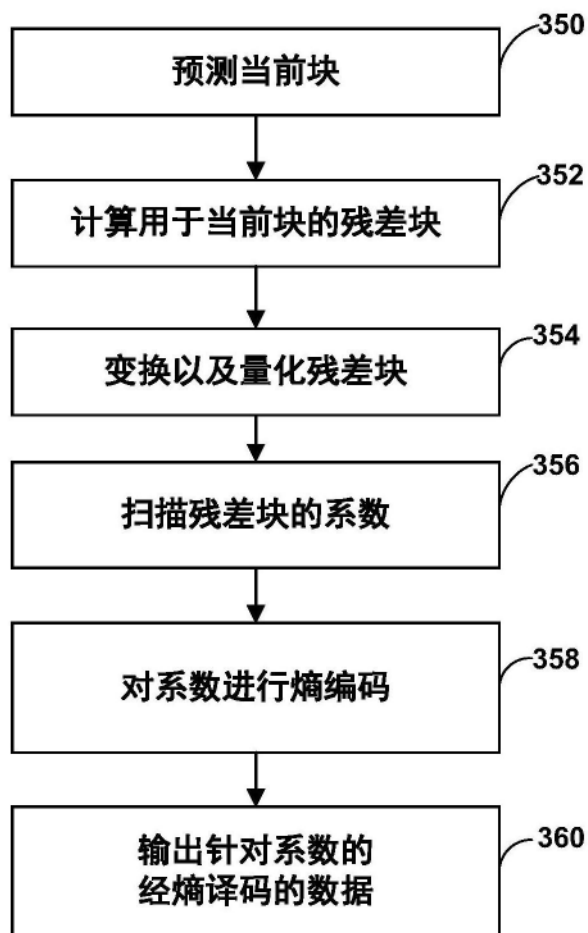


图19

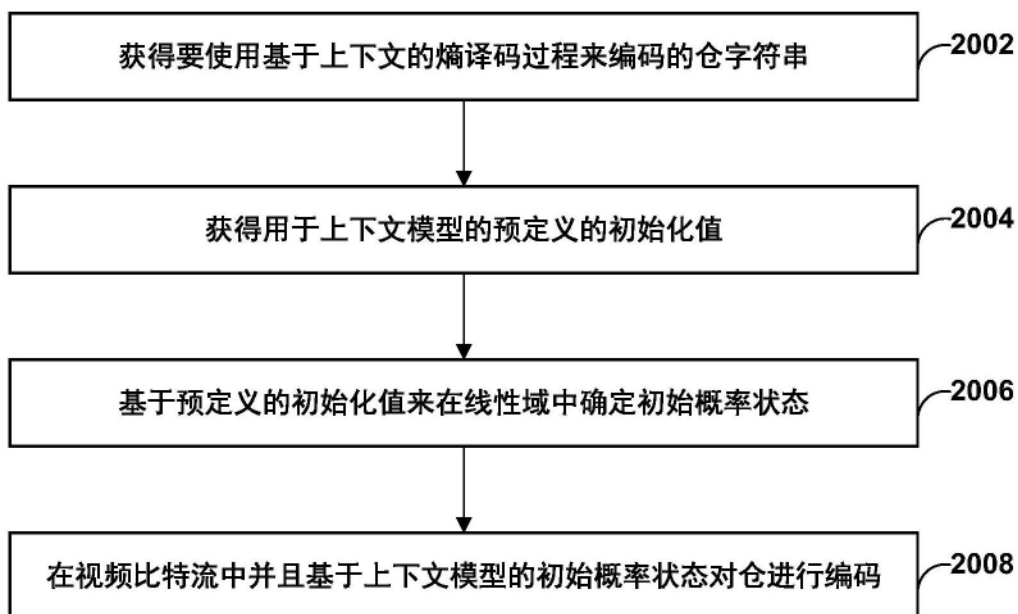


图20

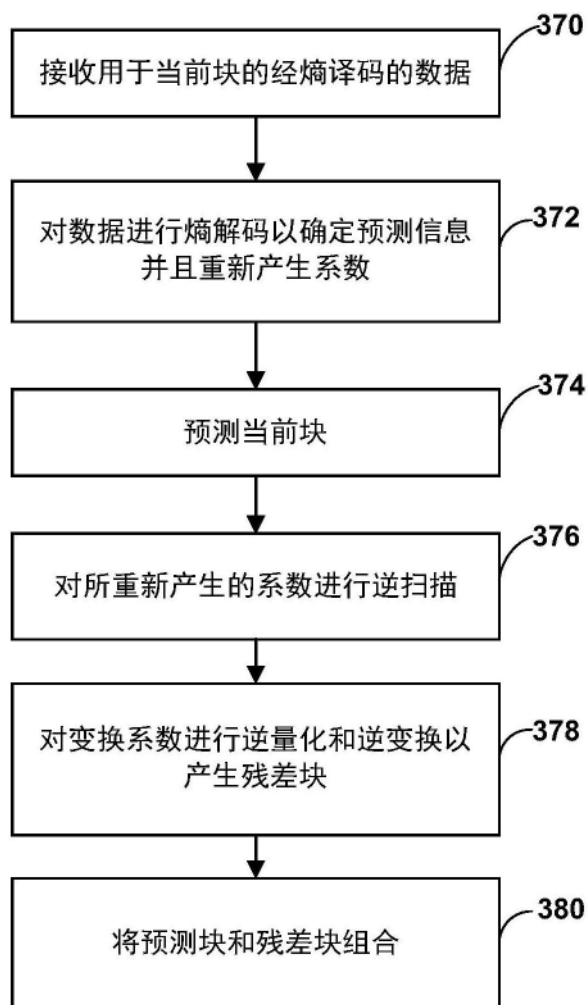


图21

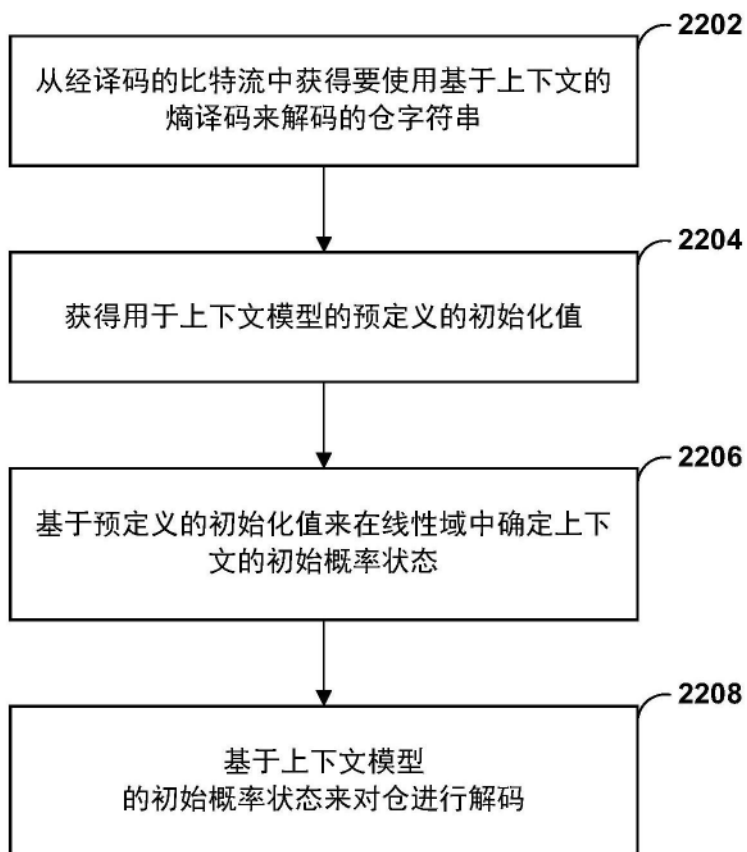


图22