[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
H04B 1/10
H04B 1/66 G06F 11/00
H04N 5/21 H04N 7/68

[12] 发明专利说明书

[21] ZL 专利号 98812593.5

[45] 授权公告日 2003 年 10 月 29 日

[11] 授权公告号 CN 1126274C

[22] 申请日 1998.10.23 [21] 申请号 98812593.5

[30] 优先权

[32] 1997.10.23 [33] US [31] 08/956870
[32] 1997.10.23 [33] US [31] 08/956632
[32] 1997.10.23 [33] US [31] 08/957555
[32] 1998. 1. 2 [33] US [31] 09/002553
[32] 1998. 1. 2 [33] US [31] 09/002547
[32] 1998. 1. 2 [33] US [31] 09/002470
[32] 1998. 1.30 [33] US [31] 09/016083
[32] 1998. 7. 6 [33] US [31] 09/111357

[86] 国际申请 PCT/US98/22532 1998.10.23

[87] 国际公布 WO99/21286 英 1999.4.29

[85] 进入国家阶段日期 2000.6.23

[71] 专利权人 索尼电子有限公司

地址 美国新泽西州

[72] 发明人 T·康多 J·J·卡里格
Y·弗吉莫里 S·戈萨尔

审查员 李 明

[74] 专利代理机构 中国专利代理(香港)有限公司

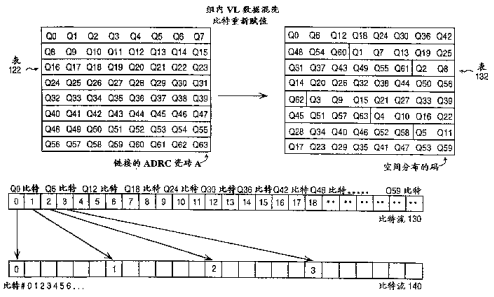
代理人 栾本生 陈景峻

权利要求书 6 页 说明书 40 页 附图 44 页

[54] 发明名称 用于在有损耗传输环境中定位传输差错以提供强有力的差错恢复的设备与方法

[57] 摘要

公开用于源编码信号以便使传输差错定位于一组抽样值的一种系统与方法。此信号包括多个信号码元(SE)，而每个 SE 具有多个分量(132)。此信号划分为多个数据组(122)，而每一个数据组具有一组 SE。数据组的每个 SE 分量组合为多个分区，每个 SE 分量具有多个比特(140)。SE 分量的多个比特从多个分区中分布在生成的比特流上。在一个实施例中，这用于可能有损耗通信信道上视频信号的传输。



ISSN1008-4274

1. 一种用于源编码信号以便使传输差错定位于一组抽样值的方法，所述方法包括以下步骤：

产生多个信号码元（SE）分量，其中每个 SE 分量包括抽样值分组，所述抽样分组包括所述一组抽样值中的至少一个抽样值；

将每个 SE 分量分布到多个分区，每个 SE 分量具有多个比特；以及

将所述 SE 分量的所述多个比特从所述多个分区分布在生成的比特流上。

2. 如权利要求 1 所述的用于源编码的方法，其中所述信号是视频信号。

3. 如权利要求 1 所述的用于源编码的方法，其中 SE 包括所述信号的编码表示。

4. 如权利要求 1 所述的用于源编码的方法，其中所述信号是音频信号。

5. 如权利要求 1 所述的用于源编码的方法，其中所述 SE 分量包括可变长度数据。

6. 如权利要求 1 所述的用于源编码的方法，其中将所述 SE 分量分布到所述多个分区的所述步骤还包括以规则顺序将 SE 分量分配到所述多个分区。

7. 如权利要求 1 所述的用于源编码的方法，其中产生所述多个 SE 分量的所述步骤包括链接信号码元（SE）的一个分组，其中 SE 包括所述一组抽样值中已编码的抽样值。

8. 如权利要求 1 所述的用于源编码的方法，其中多个分区包括 N 个 SE 分量，并且多个分区包括 Y 个分区，分布所述 SE 分量的所述步骤包括以下步骤：

（1）将第（X+Y）个 SE 分量变换到所述第 Y 分区，其中 X 是一个整数；

（2）使 X 递增一设定值；

（3）将所述第（X + Y）个 SE 分量变换到所述第 Y 分区；以及

重复步骤 1 - 3，直到所有 N 个 SE 分量被变换到所述 Y 个分区中

的一个分区。

9. 如权利要求 8 所述的用于源编码的方法，其中 N 与 Y 是整数。

10. 如权利要求 1 所述的用于源编码的方法，其中分布所述多个比特的所述步骤还包括根据所述 SE 分量分配顺序将所述多个比特分配到所述比特流中的规则位置的步骤。

11. 如权利要求 1 所述的用于源编码的方法，其中分布所述多个比特的所述步骤还包括将所述多个分区的第 Y 分区的第一比特变换到所述比特流中的第 Y 位置并将第 Y 分区的后续比特变换到第 Y 位置后面的位置的步骤。

12. 如权利要求 11 所述的用于源编码的方法，其中分布所述多个比特的步骤还包括对所有分区重复所述变换步骤。

13. 如权利要求 1 所述的用于源编码的方法，其中分布所述多个比特的步骤还包括生成单个比特流，其中打包所述比特流中的一组比特以便传输。

14. 如权利要求 6 所述的用于源编码的方法，还包括以下步骤：

重新分布所述多个比特，所述重新分布将每个所述 SE 分量的所述比特返回到所述多个分区中与所述分布步骤之前的一个位置对应的位置；和

重新组合来自所述多个分区的每个所述 SE 分量，其中所述重新组合将所述 SE 分量返回到与所述分布步骤之前的一个位置对应的位置。

15. 如权利要求 14 所述的用于源编码的方法，其中重新分布的所述步骤还包括根据所述 SE 分量重新分配顺序将每个所述 SE 分量的所述比特从所述比特流中的规则位置重新分配到所述多个分区的步骤。

16. 如权利要求 14 所述的用于源编码的方法，其中对接收的比特流进行所述重新分布，从多个接收的分组中生成所述接收的比特流。

17. 如权利要求 14 所述的用于源编码的方法，其中重新组合的所述步骤还包括以规则顺序将一个 SE 分量从所述多个分区中分离出来的步骤。

18. 一种数字处理系统，包括构造为使信号的传输差错定位于一

组抽样值的处理器，所述处理器执行执行指令，以便：

产生多个 SE 分量，其中每个 SE 分量包括抽样值分组，所述抽样值分组包括所述一组抽样值中的至少一个抽样值；

将一个数据组的每个 SE 分量分布到多个分区，每个 SE 分量具有多个比特；以及将所述 SE 分量的所述多个比特从所述多个分区分布在生成的比特流上。

19. 如权利要求 18 所述的数字处理系统，其中所述信号是视频信号。

20. 如权利要求 18 所述的数字处理系统，其中 SE 包括所述信号的编码表示。

21. 如权利要求 18 所述的数字处理系统，其中所述信号是音频信号。

22. 如权利要求 18 所述的数字处理系统，其中所述 SE 分量包括可变长度数据。

23. 如权利要求 18 所述的数字处理系统，其中所述处理器被配置成通过以规则顺序将 SE 分量分配到所述多个分区而将所述数据组的所述 SE 分量分布到所述多个分区。

24. 如权利要求 18 所述的数字处理系统，其中所述处理器被配置成通过链接信号码元（SE）的一个分组而产生所述多个 SE 分量，其中 SE 包括已编码的抽样值。

25. 如权利要求 18 所述的数字处理系统，其中多个分区包括 N 个 SE 分量，并且多个分区包括 Y 个分区，所述处理器被配置成通过以下步骤来分布所述 SE 分量：

（1）将第（X+Y）个 SE 分量变换到所述第 Y 分区，其中 X 是一个整数；

（2）使 X 递增一设定值；

（3）将所述第（X + Y）个 SE 分量变换到所述第 Y 分区；以及

重复步骤 1 - 3，直到所有 N 个 SE 分量被变换到所述 Y 个分区中的一个分区。

26. 如权利要求 25 所述的数字处理系统，其中 N 与 Y 是整数。

27. 如权利要求 23 所述的数字处理系统，其中所述处理器被配置成通过与所述 SE 分量分配顺序相对应地将所述多个比特分配到所

述比特流中的规则位置而分布所述多个比特。

28. 如权利要求 18 所述的数字处理系统，其中所述处理器被配置成通过将所述多个分区的第 Y 分区的第一比特变换到所述比特流中的第 Y 位置并将第 Y 分区的后续比特变换到第 Y 位置后面的位置而分布所述多个比特。

29. 如权利要求 28 所述的数字处理系统，其中所述处理器被配置成通过对所有分区重复所述变换而分布所述多个比特。

30. 如权利要求 18 所述的数字处理系统，其中所述处理器被配置成通过生成单个比特流而分布所述多个比特，其中打包所述比特流中的一组比特以便传输。

31. 如权利要求 23 所述的数字处理系统，其中所述处理器被配置成:

重新分布所述多个比特，所述重新分布将每个所述 SE 分量的所述比特返回到所述多个分区中与所述分布之前的一个位置对应的位置；和

重新组合来自所述多个分区的每个所述 SE 分量，其中所述重新组合将所述 SE 分量返回到与所述分布之前的一个位置对应的位置。

32. 如权利要求 31 所述的数字处理系统，其中所述处理器还被配置成通过按照所述 SE 分量分配顺序将每个所述 SE 分量的所述比特从所述比特流中的规则位置重新分配到所述多个分区而进行重新分布。

33. 如权利要求 31 所述的数字处理系统，其中对接收的比特流进行所述重新分布，从多个接收的分组中生成所述接收的比特流。

34. 如权利要求 31 所述的数字处理系统，其中所述处理器被配置成通过以规则顺序将所述数据组的 SE 分量从所述多个分区中分离出来而进行重新组合。

35. 一种系统，构造为使信号的传输差错定位于一组抽样值，所述系统包括:

用于产生多个 SE 分量的装置，其中每个 SE 分量包括抽样值分组，所述抽样值分组包括所述一组抽样值中的至少一个抽样值;

用于将每个 SE 分量分布到多个分区的装置，每个 SE 分量具有多个比特；和

用于将所述 SE 分量的所述多个比特从所述多个分区分布在生成的比特流上的装置。

36. 如权利要求 35 所述的系统，其中所述信号是视频信号。

37. 如权利要求 35 所述的系统，其中用于将所述所述 SE 分量分布到所述多个分区的所述装置还包括以规则顺序将 SE 分量分配到所述多个分区。

38. 如权利要求 35 所述的系统，其中用于产生所述多个 SE 分量的所述装置还包括链接信号码元（SE）的一个分组，其中 SE 包括已编码的抽样值。

39. 如权利要求 35 所述的系统，其中多个分区包括 N 个 SE 分量，并且多个分区包括 Y 个分区，用于分布所述 SE 分量的所述装置还包括：

用于将第（X+Y）个 SE 分量变换到所述第 Y 分区的装置，其中 X 是一个整数；

用于使 X 递增一设定值的装置；

用户将所述第（X + Y）个 SE 分量变换到所述第 Y 分区的装置；以及

重复步骤 1 - 3，直到所有 N 个 SE 分量被变换到所述 Y 个分区中的一个分区。

40. 如权利要求 37 所述的系统，其中用于分布所述多个比特的所述装置还包括用于根据所述 SE 分量分配顺序将所述多个比特分配到所述比特流中的规则位置的装置。

41. 如权利要求 35 所述的系统，其中用于分布所述多个比特的所述装置还包括用于根据所述 SE 分量重新分配顺序将所述多个比特重新分配到所述比特流中的模块化位置的装置。

42. 如权利要求 35 所述的系统，其中用于分布所述多个比特的装置对所有分区都重新执行操作。

43. 如权利要求 37 所述的系统，还包括：

用于重新分布所述多个比特的装置，所述用于分布的装置将每个所述 SE 分量的所述比特返回到与所述分布步骤之前的一个位置对应的位置；和

用于重新组合来自所述多个分区的每个所述 SE 分量的装置，其

中所述用于重新组合的装置将所述 SE 分量返回到与所述分布步骤之前的一个位置对应的位置。

44. 如权利要求 43 所述的系统，其中用于重新分布的所述装置还包括用于根据所述 SE 分量重新分配顺序将每个所述 SE 分量的所述比特从所述比特流中的规则位置重新分配到所述多个分区的装置。

45. 如权利要求 43 所述的系统，其中对接收的比特流进行所述重新分布，从多个接收的分组中生成所述接收的比特流。

46. 如权利要求 43 所述的系统，其中用于重新组合的所述装置还包括用于以规则顺序将 SE 分量从所述多个分区中分离出来的装置。

用于在有损耗传输环境中定位传输差错以提供强有力的差错恢复的设备与方法

本申请是 1998 年 1 月 30 日提交的题为“Source Coding to Provide for Robust Error Recovery During Transmission Losses”专利申请系列号为 09/016,083 的美国专利申请的继续；本申请是 1998 年 1 月 2 日提交的题为“Image-to-Block Mapping to Provide for Robust Error Recovery During Transmission Losses”的申请系列号为 09/002,547、1998 年 1 月 2 日提交的题为“Source Coding to Provide for Robust Error Recovery During Transmission Losses”的申请系列号为 09/002,470、1998 年 1 月 2 日提交的题为“Multiple Block Based Recovery Method to Provide for Robust Error Recovery During Transmission Losses”的申请系列号为 09/002,553 的部分继续申请；这些申请系列号的申请是 1997 年 10 月 23 日提交的题为“Image-to-Block Mapping to Provide for Robust Error Recovery During Transmission Losses”的申请系列号为 08/956,632、1997 年 10 月 23 日提交的题为“Source Coding to Provide for Robust Error Recovery During Transmission Losses”的申请系列号为 08/957,555 和 1997 年 10 月 23 日提交的题为“Multiple Block Based Recovery Method to Provide for Robust Error Recovery During Transmission Losses”的申请系列号为 08/956,870 的部分继续。现将 1998 年 1 月 30 日提交的申请系列号 09/016,083、1998 年 1 月 2 日提交的申请系列号 09/002,547、1998 年 1 月 2 日提交的申请系列号 09/002,470、1998 年 1 月 2 日提交的申请系列号 09/002,553、1997 年 10 月 23 日提交的申请系列号 08/956,632、1997 年 10 月 23 日提交的申请系列号 08/957,555 和 1997 年 10 月 23 日提交的申请系列号 08/956,870 引入在此作为参考。

本发明涉及为由于信号传输期间引起的数据损耗而提供强有力的差错恢复。

存在许多技术来再现由于信号传输期间引起的随机差错而丢失

的数据。然而，这些技术不能处理连续的数据分组的损耗。在本领域中将数据分组的连续损耗描述为突发差错。突发差错导致再现的信号具有终端用户容易察觉到的降级质量。另外，用于实施高速通信的压缩方法加重了由于突发差错引起的信号降级，因而增加了再现信号的降级。影响发射信号和/或存储信号的突发差错损耗的示例在高清晰度电视（“HDTV”）信号与其中压缩方法起重要作用的移动电信应用中可以发现。

HDTV 的出现导致具有比国家电视系统委员会（“NTSC”）建议的当前标准高得多的清晰度的电视系统。所建议的 HDTV 信号主要是数字的。因此，在变换彩色电视信号用于数字使用时，通常利用八个比特来数字化亮度与色度信号。彩色电视的数字传输要求每秒二百一十六兆比特的标称比特率。用于 HDTV 的传输速率更大，这标称上要求每秒约 1200 兆比特。如此高的传输速率远远超过当前无线标准所支持的带宽。因此，要求一种有效的压缩方法。

压缩方法也在移动电信应用中起着重要作用。一般地，在移动电信应用中在远程终端之间传送数据分组。移动通信中有限数量的传输信道要求数据分组传输之前有效的压缩方法。可用许多压缩技术实现高的传输速率。

自适应动态范围编码（“ADRC”）和离散余弦变换（“DCT”）编码提供本领域公知的图像压缩技术。这些技术利用图像内的位置相关性来实现高的压缩比。然而，因为在随后解码时编码信号中的差错更突出，所以有效的压缩算法导致复合的差错传播。此差错倍增导致用户容易察觉到的降级的视频图像。

现描述用于源编码信号的一种方法。特别是，处理包括多个信号码元的信号。编码每个信号码元，以形成比特流。给定比特流内的比特分布在不同的比特流上。因而，描述分段码元组成分量的参数分布在不同的比特流上。此分布步骤导致多个电平上的差错分布。因此，在解码器反向该分布步骤时，突发传输差错变成定位损耗的分布集。

也描述用于多级（multiple level）混洗处理的另一种方法。信号定义为多级，其中每个级包括多个帧、多个像素和多个比特。在一个实施例中，混洗出现在每个级上并出现在级之间。多级混洗使突发差错损耗分布在多级上，从而实现其中出现损耗的那些图像区域的图

像再现。

从下面具体的描述中本发明的目的、特征与优点对于本领域技术人员来说将是显而易见的，其中：

图 1 概括地表示信号编码、传输与解码的处理。

图 2 表示分组结构的一个实施例。

图 3 是表示根据本发明教导的编码处理的一个实施例的流程图。

图 4 是表示根据本发明教导的解码处理的一个实施例的流程图。

图 5 表示根据本发明教导的图像-块变换的一个实施例。

图 5a 表示图像-块变换中使用的混洗模式的一个实施例。

图 6 是示例性互补与互锁块结构的示意图。

图 7a、7b、7c、7d 表示用于一个帧集内 Y 个块的混洗模式的一个实施例。

图 8 是用于缓冲器 0 的累积 DR 分布的一个实施例的示意图。

图 8a 是根据本发明教导的部分缓冲处理的一个实施例的示意图。

图 9 表示根据本发明教导的缓冲器内 YUV 块混洗处理的一个实施例。

图 10 表示根据本发明教导的组内 VL 数据混洗处理的一个实施例。

图 11 表示根据本发明教导的 3 块组内 Q 码链接的一个实施例。

图 11a 表示根据本发明教导的用于包括运动块的帧对的 Q 码链接的一个实施例。

图 12 表示 1/6 突发差错损耗引起的像素数据差错的一个实施例。

图 12a 表示根据本发明教导的混洗 Q 码与分布 Q 码比特的一个实施例。

图 12b 表示重新分配的 Q 码的 1/6 突发差错损耗引起的像素数据差错的一个实施例。

图 12c 表示重新分配的 Q 码的 1/6 突发差错损耗引起的像素数据差错的一个实施例。

图 13 表示根据本发明教导的 MIN 混洗的一个实施例。

图 13a 表示一个帧对中运动标志混洗与固定长度数据损耗的一个

实施例。

图 14 表示模块化混洗的一个实施例。

图 14a 表示与模块化混洗相关的模块化混洗结果与固定长度数据损耗的一个实施例。

图 14b 表示与模块化混洗相关的模块化混洗结果与固定长度数据损耗的另一实施例。

图 14c 表示与模块化混洗相关的模块化混洗结果与固定长度数据损耗的又一实施例。

图 15 表示在一个帧集内缓存的可变长度数据的一个实施例。

图 16 表示根据本发明教导的分段之间 VL 数据混洗的一个实施例。

图 17 是概括表示本发明的数据恢复处理的一个实施例的流程图。

图 18 是本发明的 Q 比特（Qbit）与运动标志恢复处理的一个实施例的流程图。

图 19 是表示候选解码的一个实施例的表。

图 20a、20b、20c、20d 表示图 18 的 Q 比特与运动标志恢复处理中使用的测量的实施例。

图 21 表示用于确定图 18 的 Q 比特与运动标志恢复处理中使用的平方差错概率函数的表的一个实施例。

图 22 表示根据本发明教导的 Q 比特、运动标志与辅助信息恢复处理的一个实施例。

图 23 表示双向 Q 比特与运动标志恢复处理的一个实施例中后置码的使用。

图 24a、24b 与 24c 表示用于评估候选解码的另一实施例。

图 25 表示根据本发明一个实施例教导的平滑测量的使用。

图 26a、26b、26c、26d 与 26e 表示用于评估候选解码的处理的另一实施例。

图 27a 表示用于评估候选解码的另一处理，和图 27b 表示用于确定加权值的一个实施例。

本发明提供一种方法，用于编码与安排信号流，以提供强有力的差错恢复。在下面的描述中，为解释目的，陈述许多细节，以提供本

发明的完全理解。然而，对于本领域技术人员来说，显然实践本发明不要求这些特定细节。在其他示例中，公知的电子结构与电路以方框图来表示，以便不徒然地使本发明难于理解。

从其中信号为视频信号的一个实施例的观点出发来描述信号处理方法与结构。然而，本文所述的方法与设备打算可应用于各种类型的包括音频信号的信号和数据的其他数字比特流，其中每个信号由多个信号码元组成。而且，本文所述的处理的实施例使用自适应动态距离编码（“ADRC”）处理来压缩数据；然而可以使用许多编码技术与算法。至于有关 ADRC 的更具体的讨论，参见 1991 年 9 月 4-6 日在意大利都灵召开的 Fourth International Workshop on HDTV and Beyond 会议上 Kondo、Fujimori 与 Nakaya 的“Adaptive Dynamic Range Coding Scheme for Future HDTV Digital VTR”。

在上面文件中，解释三种不同类型的 ADRC。这些 ADRC 根据以下等式来实现：

非边缘匹配 ADRC:

$$DR=MAX-MIN+1$$

$$q=\left[\frac{(x-MIN+0.5)\cdot 2^{Q}}{DR}\right]$$

$$\tilde{x}=\left[\frac{(q+0.5)\cdot DR}{2^{Q}}+MIN\right]$$

边缘匹配 ADRC:

$$DR=MAX-MIN$$

$$q=\left[\frac{(x-MIN)\cdot(2^{Q}-1)}{DR}+0.5\right]$$

$$\tilde{x}=\left[\frac{q\cdot DR}{2^{Q}-1}+MIN+0.5\right]$$

多级 ADRC:

$$DR=MAX-MIN+1$$

$$q = \left[\frac{(x - MIN + 0.5)\cdot 2^{Q}}{DR}\right]$$

$$\tilde{x} = \left[\frac{(q + 0.5)\cdot DR}{2^{Q}} + MIN\right]$$

其中 MAX 表示块的最大电平，MIN 表示块的最小电平，X 表示每个抽样值的电平，Q 表示量化比特的数量，q 表示量化码（编码的数据），$\tilde{X}$表示每个抽样值的解码电平，和方括号[ ]表示对方括号内的值执行的舍位操作。

在图 1 中概括地表示信号编码、传输和随后的解码处理。信号 100 是输入到编码器 110 的数据流。编码器 110 遵循自适应动态距离编码（“ADRC”）压缩算法并生成用于通过传输介质 135 传输的数据分组 1，...N。解码器 120 从传输介质 135 中接收数据分组 1，...N 并生成信号 130。信号 130 是信号 100 的再现。

编码器 110 与解码器 120 能以各种方式来实施，以执行本文所述的功能。在一个实施例中，编码器 110 和/或解码器 120 实施为存储在介质上并利用通用或特殊配置的一般包括中央处理单元、存储器与一个或多个输入/输出装置和协同处理器的计算机系统来执行的软件。或者，编码器 110 和/或解码器 120 可以实施为逻辑电路来执行本文所述的功能。另外，编码器 110 和/或解码器 120 能实施为硬件、软件或固件的组合。

在本实施例中，信号 100 是包括视频帧序列的彩色视频图像，每帧包括表示隔行扫描视频系统中图像的信息。每帧由两场组成，其中一场包含此图像偶数行的数据，而另一场包含此图像奇数行的数据。此数据包括描述此图像中相应位置的彩色分量的像素值。例如，在本实施例中，彩色分量由亮度信号 Y 和色差信号 U 与 V 组成。显然：本发明的处理能应用于除隔行扫描视频信号之外的其他信号。而且，本发明显然不限于以 Y、U、V 彩色空间来实施，而能应用于以其他彩色空间表示的图像。

再参见图 1，编码器 110 根据 ADRC 算法划分 Y、U 与 V 信号并且独立地处理每一组信号。为简化讨论，下文描述 Y 信号的处理；然而，对于 U 与 V 信号，重复这些编码步骤。

在本实施例中，编码器 110 将信号 100 的两个后续帧（在此称为

帧对）上的 Y 信号组合为三维（“3D”）数据块。对于一个实施例而言，通过组合来自给定帧对上同一定位区域的两个 2D 块来生成 3D 块，其中通过组合一帧或一场内的定位像素来生成二维 2D 块。打算本文所述的处理能应用于不同的块结构。下面在图像-块变换部分中将进一步描述信号的组合。

继续本实施例，对于给定的 3D 块，编码器 110 来计算形成 3D 块的 2D 块之间的像素值是否有变化。如果这些值有显著变化，则集运动标志。如本领域所公知的，运动标志的使用允许编码器 110 在每个帧对内具有定位图像重复时减少量化码的数量。编码器 110 也检测 3D 块内最大像素强度值（“MAX”）与最小像素强度值（“MIN”）。利用值 MAX 与 MIN，编码器 110 计算给定的 3D 数据块的动态范围（“DR”）。在非边缘匹配 ADRC 的情况中，对于一个实施例，DR = MAX − MIN + 1。对于边缘匹配 ADRC，DR = MAX − MIN。在可选择的实施例中，编码器 110 逐帧编码代表视频帧序列的帧流的信号。在另一实施例中，编码器 110 逐场编码代表视频场序列的场流的信号。因此，不使用运动标志，而使用 2D 块来计算 MIN、MAX 和 DR 值。

在本实施例中，编码器 110 将计算的 DR 与门限表（未示出）进行比较来确定用于编码对应于此 DR 的块内的像素的量化比特（“Q 比特”）的数量。像素的编码得到量化码（“Q 码”），Q 码是用于存储或传输目的的相关压缩的图像数据。

在一个实施例中，从 3D 块的 DR 中导出 Q 比特选择。因此，给定 3D 块内所有的像素使用同一 Q 比特来编码，得到 3D 编码的块。Q 码、MIN、运动标志和 3D 编码块一起称为 3D ADRC 块。可选择地，编码 2D 块，并且给定 2D 块的 Q 码、MIN 与 DR 一起形成 2D ADRC 块。

能实施许多门限表。在一个实施例中，门限表由一行 DR 门限值组成。Q 比特对应于用于编码此门限表的一行内二个相邻的 DR 之间 DR 值范围的量化比特的数量。在可选择的一个实施例中，此门限表包括多个行并且行的选择取决于所需的传输速率。此门限表中的每一行利用门限索引来识别。下面在部分缓冲的讨论中描述门限选择的一个实施例的详细说明。ADRC 编码与缓冲的进一步描述公开在转让给本发明的受让人的题为“High efficiency Coding Apparatus” 的 US 专利号 4,722,003 和题目也为“High efficiency Coding

Apparatus”的US专利号4,845,560中。

本文在下面将Q码称为可变长度数据(“VL数据”)。另外，DR、MIN和运动标志称为块属性。块属性与门限索引一起构成固定长度数据(“FL数据”)。而且，鉴于上面的讨论，术语块属性描述与信号码元的分量相关的参数，其中信号码元包括多个分量。在一个可选择的实施例中，FL数据包括Q比特值。优点是在解码处理期间不必从DR中导出Q比特信息。因而，如果DR信息丢失或损坏，仍然能从Q比特值中确定Q比特信息。而且，如果Q比特值丢失或损坏，能从DR中导出Q比特信息。因而减少恢复DR与Q比特的要求。

包括Q比特值的缺点是要为每个ADRC块发送附加比特。然而，在一个实施例中，例如，根据诸如加法或链接的函数组合ADRC块组的Q比特码。例如，如果ADRC块三个成组并且如果每个ADRC块的Q比特值分别为3、4与4，则编码为FL数据的相加值为11。因而，表示此和所要求的比特数量少于表示每个单个值所要求的比特数量，并且此组未损坏的Q比特值能用于确定损坏块的Q比特值而无需执行诸如接着要描述的Q比特恢复处理。

再考虑其他的实施例。例如，也可以编码运动标志数据。具有Q比特与运动标志数据的标记能生成并用于标注码表。编码的结构与功能能根据应用而变化。

帧、块属性与VL数据描述视频信号内的各个分量。这些分量的边界、位置与数量取决于视频信号的传输与压缩特性。在目前的实施例中，视频信号的比特流内的这些分量进行变化与混洗，以保证传输损耗期间强有力的差错恢复。

为说明目的，下面的描述根据视频信号的ADRC编码与混洗提供1/6连续分组传输损耗容限。因此，下面的分量的定义与划分是对一个实施例而言。也考虑其他的实施。数据集包括视频数据或其他类型的数据信号的分区。因而，在一个实施例中，帧集是包括一个或多个连续帧的一种类型的数据集。分段包括具有存储包括在帧集中的Q码与块属性的六分之一部分的容量的存储器。而且，缓冲器包括具有存储包括在帧集中的Q码与块属性的六十分之一部分的容量的存储器。通过互换分段和/或缓冲器中的分量来执行数据的混洗。然后，存储在分段中的数据用于生成用于传输的数据分组。因而，在下面的描述

中，如果丢失分段，则在传输期间丢失从此分段中生成的所有分组。同样地，如果丢失分段的一部分，则在传输期间丢失从此分段中生成的相应数量的分组。

虽然下面的描述涉及使用 ADRC 编码来编码的数据的 1/6 连续分组损耗，但本文所述的方法与设备打算可应用于与各种编码/解码方案耦合的 1/n 连续分组损耗容限的设计。

图 2 表示用于点对点连接以及网络上的数据传输的分组结构 200 的一个实施例。分组结构 200 由编码器 110 生成并通过传输介质 135 进行发送。对于一个实施例而言，分组结构 200 包括 5 个字节的标题信息、8 个 DR 比特、8 个 MIN 比特、一个运动标志比特、5 比特门限索引和 354 个比特的 Q 码。本文所述的分组结构是示意性的并且一般可以实施为用于异步传送模式（“ATM”）网络中的传输。但是，本发明不限于所述的分组结构，而能利用各种网络中使用的各种分组结构。

如上所述，不假定传输介质 135 提供无差错传输并因此可能丢失或损坏分组。如上所述，存在常规的方法来检测这样的丢失或损坏，但一般将出现严重的图像质量恶化。本发明的系统与方法因此教导源编码来提供从这样的丢失或损坏中进行强有力的恢复。在整个下面的讨论中假定是几个连续分组丢失的突发损耗是最有可能的差错形式，但是也可能出现一些随机的分组损耗。

为了保证一个或多个连续的数据分组损耗的强有力恢复，本发明的系统与方法提供多级混洗。特别地，包括在发送分组中的 FL 数据与 VL 数据分组包括来自图像的空间与时间上不连续位置的数据。混洗数据保证分散任何一个突发差错和便于差错恢复。如下面将要描述的，混洗允许块属性与 Q 比特值的恢复。

数据编码/解码

图 3 是表示编码器 110 所执行的编码处理的一个实施例的流程图。图 3 还描述用于抵抗图像恶化和实施强有力的差错恢复的混洗处理的概要。

在图 3 的步骤 1 中，抽取也称为显示分量的输入帧集，以减少传输要求。水平抽取 Y 信号为其原始宽度的四分之三，并将 U 与 V 信号均抽取为其原始高度的一半与其原始宽度的一半，这得到每个帧对中

具有 3960 个 Y 块、660 个 U 块与 660 个 V 块的 3：1：0 视频格式。如上所述，此讨论将描述 Y 信号的处理，但是此处理可应用于 U 与 V 信号。在步骤 2，将两个 Y 帧图像变换为 3D 块。在步骤 3，混洗 3D 块。在步骤 4，使用 ADRC 缓冲与编码。在步骤 5，在缓冲器内混洗所编码的 Y、U 与 V 块。

在步骤 6，混洗一组编码的 3D 块的 VL 数据及其相应的块属性。在步骤 7，混洗不同分段上的 FL 数据。在步骤 8，执行后置码填充，其中在缓冲器末尾的可变空间利用预定的比特流来填充。在步骤 9，混洗不同分段上的 VL 数据。

为说明目的，下面的混洗描述提供在编码之前与之后像素数据处理的方法。对一个可选择的实施例而言，利用硬件混洗/去混洗独立的数据值。特别地，此硬件将块值的地址变换为不同的地址来实施混洗/去混洗处理。然而，因为混洗得在数据处理之后进行，所以地址变换对于数据依赖值是不可能的。下面所描述的 VC 数据组内混洗包括数据依赖值。还有，为说明目的，下面的混洗描述发生在离散的数据集上。然而，对可选择的实施例而言，根据从比特到像素和到帧的范围内的多个数据电平来定义信号。对于此信号中定义的和此信号不同的数据电平之中的每个电平，混洗是有可能的。

图 4 是表示解码器 120 所执行的解码处理的一个实施例的流程图。优选地，变换与去混洗处理是图 3 所示处理的逆处理。图 4 以 Q 比特、运动标志、DR、MIN 与像素数据的不同组合进一步描述用于差错恢复的创新处理。下面以不同实施例的 Q 比特 、运动标志、DR、MIN 与像素恢复的不同组合描述差错恢复处理。

图像-块变换

在本实施例中，单个帧一般包括 5280 个 2D 块，其中每个 2D 块包括 64 个像素。因此，由于来自第一帧的一个 2D 块与来自随后一帧的一个 2D 块一起形成一个 3D 块，所以一个帧对包括 5280 个 3D 块。

为了将数据帧或帧集分别划分为 2D 块与 3D 块而执行图像-块变换。而且，图像-块变换包括使用互补和/或互锁模式将像素划分为帧，以便在传输损耗期间实施强有力的差错恢复。但是，为了提高给定的 DR 值不太大的概率，从定位区域的像素中构造每个 2D 块。

图 5 表示示例性的图像的 16 像素部分的图像-块变换处理的一个

实施例。图像 500 包括形成单个帧的定位区域的 16 个像素。图像 500 中的每个像素利用强度值来表示。例如，此图像左上侧的像素具有等于 100 的强度值，而此图像右底部的像素具有为 10 的强度值。

在一个实施例中，来自图像 500 不同区域的像素用于生成 2D 块 510、520、530 与 540。编码、混洗（如下所示）并发送 2D 块 510、520、530 与 540。在发送之后，2D 块 510、520、530 与 540 进行重新组合并用于形成图像 550。图像 550 是图像 500 的再现。

尽管有可能的传输损耗，但为了保证图像 500 的准确表示，图 5 是一种互锁互补块结构，图 5 所示的一个实施例用于再现图像 500。特别地，用于生成 2D 块 510、520、530 与 540 的像素选择保证互补和/或互锁模式用于在再现图像 550 时组合这些块。因此，当在传输期间丢失特定 2D 块的属性时，图像 550 的相邻部分在再现期间不失真。例如，如图 5 所示，2D 块 540 的 DR 在数据传输期间丢失。然而，在图像 550 的再现期间，解码器利用相邻块的多个相邻像素，从中能恢复 2D 块 540 丢失的 DR。另外，如下面将要描述的，互补模式与移位的组合增加相邻像素的数量，更好地最大化源于其他块的相邻像素的数量，显著改善 DR 与 MIN 恢复。

图 5a 表示在图像-块变换处理的一个实施例中用于形成 2D 块的混洗模式的一个实施例。一个图像根据选择的像素分解为二个子图像，即子图像 560 与子图像 570。在子图像 560 中形成矩形来描述 2D 块边界。为讨论目的，2D 块编号为 0、2、4、7、9、11、12、14、16、19、21 与 23。瓷砖（tile）565 表示子图像 560 内 2D 块的像素分布。

在子图像 570 中，2D 块分配水平移位八个像素并且垂直移位四个像素。这在再现期间组合子图像 560 与 570 时得到环绕处理的 2D 块分配与重叠。2D 块编号为 1、3、5、6、8、10、13、15、17、18、20 与 22。瓷砖 575 表示子图像 570 内 2D 块的像素分布。瓷砖 575 是瓷砖 565 的互补结构。因此，当在传输期间丢失特定块的属性时，存在从中能恢复丢失 2D 块的块属性的相邻像素。另外，存在具有相似的块属性集的像素的重叠 2D 块。因此，在图像的再现期间，解码器具有来自从中能恢复丢失的块属性的相邻 2D 块的多个相邻像素。

图 6 表示其他的互补与互锁 2D 块结构。也可以利用其他的结构。与图 5 相似，尽管给定的 2D 块的传输有损耗，但图 6 中所示的 2D 块

结构保证周围的 2D 块出现。然而，模式 610a、610b 与 610d 在像素变换为后续的 2D 块期间使用水平和/或垂直移位。水平移位描述在开始新的 2D 块边界之前在水平方向上将瓷砖结构移位预定数量的像素。垂直移位描述在开始新的 2D 块边界之前在垂直方向上将瓷砖结构移位预定数量的像素。在应用中，可以只采用水平移位，可以只采用垂直移位，或者可以采用水平与垂直移位的组合。

模式 610a 表示用于图像-块变换的螺旋形模式，此螺旋形模式进行水平移位以便在图像-块变换处理期间生成后续的 2D 块。模式 610b 与 610d 表示互补模式，其中将像素选择进行水平或垂直移位以便在图像-块变换处理期间生成后续的 2D 块。还有，模式 610b 与 610d 表示 2D 块之间像素选择的选择偏移。模式 610c 表示利用不规则的像素抽样来生成用于图像-块变换的 2D 块。因此，图像-块变换遵循任意一种变换结构，假定像素只变换为 2D 块一次。

图 5、图 5a 与图 6 描述用于 2D 块生成的图像-块变换。显然这些处理可应用于 3D 块。如上所述，3D 块生成遵循与 2D 块相同的边界定义，但是此边界划分扩展到后一帧，得到 3D 块。特别地，通过收集用于定义第一帧中的 2D 块的像素与来自后一帧中 2D 块的像素来生成 3D 块。在一个实施例中，来自第一帧的 2D 块和来自后一帧的 2D 块中的像素都来自完全相同的位置。

帧集块内混洗

给定图像的像素值与定位区域紧密相关。然而，在相同图像的另一区域中，像素值可以具有明显不同的值。因而，在编码之后，空间相近的 2D 或 3D 块的 DR 与 MIN 值可以与另一部分图像中块的 DR 与 MIN 值完全不同。因此，在缓冲器顺序地利用来自空间相近的 2D 或 3D 块的编码数据进行填充时，出现缓冲器空间不成比例的使用。帧集块内混洗出现在 ADRC 编码之前并包括混洗在图像-块变换处理期间生成的 2D 或 3D 块。此混洗处理保证在随后的 ADRC 编码期间均衡的缓冲器使用。

图 7a-7d 表示混洗 3D Y 块的一个实施例。图 7a-7d 中的 3D Y 块通过对只包含 Y 信号的帧对应用上述的图像-块变换处理来生成。混洗 3D Y 块，以保证用于存储编码帧对的缓冲器包含来自此帧对的不同部分的 3D Y 块，这导致 ADRC 编码期间相似的 DR 分布。每个缓冲

器内相似的 DR 分布导致一致的缓冲器利用。

图 7a-7d 还表示利用物理上不相连的 3D 块的 3D 块混洗以保证连续分组的传输损耗导致损坏的块属性分散在此图像上，这与此图像的定位区域相反。

块混洗设计为在出现小、中等或大的突发分组损耗时广泛分布块属性。在本实施例中，小的突发损耗认为是丢失几个分组的损耗；中等损耗是丢失能保持在一个缓冲器中的数据量的损耗；而大的损耗是丢失能保持在一个分段中的数据量的损耗。在 3D 块混洗期间，从此图像相对遥远的部分中选择每一组的三个邻近块。因此，在随后的（下文将详细描述的）组 VL 数据内混洗期间，从具有不同统计特性的 3D 块中形成每个组。因为未损坏的 3D 块包围着损坏的 3D 块并且未损坏的 3D 块能用于恢复丢失的数据，所以分布的块属性损耗允许强有力的差错恢复。

图 7a 表示包含水平方向上的 66 个 3D Y 块与垂直方向上的 60 个 3D Y 块的帧对，这些 3D Y 块部署在分段 0－5 中。如所示的，3D Y 块分配遵循二乘三列部分，以致来自每个部分的一个 3D 块与一个分段相关。因而，如果不执行另外的混洗并且出现第一 880 分组的突发损耗，则丢失与分段 0 相关的所有块属性。然而，如下文所述的，执行 FL 数据混洗以便进一步分散块属性损耗。

图 7b 表示用于输入分段 0 的编号为“0”的 3D Y 块的扫描顺序。图 7a 的每个“0”3D Y 块编号为 0，1，2，3，...，659 来表示其在输入到分段 0 中的数据流中的位置。利用此块编号来进行分段分配，将剩余的 3D Y 块输入到分段 1－5 中，从而得到在多个分段上混洗的帧对。

图 7c 表示包括一个分段的 660 个 3D Y 块。编号为 0－65 的 3D Y 块输入到缓冲器 0 中。同样，与编号的 3D Y 块相邻的 3D Y 块输入到缓冲器 1 中。重复此处理来填充缓冲器 2－9。因此，数据传输期间缓冲器的损坏导致丢失来自此图像不同部分的 3D Y 块。

图 7 d 表示缓冲器上“0”3D Y 块的最后顺序。3D Y 块 0、1 与 2 占用缓冲器中前面三个位置。对于此缓冲器的其余部分，重复此处理。因此，数据传输期间 3 个 3D Y 块的丢失导致丢失来自此图像内远端位置的 3D Y 块。

图 7a-d 表示用于帧集的 3D Y 块的 3D 块分布的一个实施例。然而，在可选择的实施例中，用于 3D U 块与 3D V 块的 3D 块分布是可利用的。通过对仅包含 U 信号的帧集采用上述的图像-块变换处理来生成 3D U 块。同样地，通过对仅包含 V 信号的帧集采用图像-块变换处理来生成 3D V 块。3D U 块与 3D V 块都遵循上述的 3D Y 块分布。然而，如先前所述的，3D U 块与 3D V 块的数量相对 3D Y 块均具有 1:6 比例。

图 7a-d 用于表示 Y 信号的帧集块内混洗以致容许传输期间多达 1/6 的分组丢失的突发差错并且还保证均衡的缓冲器使用的一个实施例。本领域技术人员将认识到：分段、缓冲器与 ADRC 块分配能进行变化以保证抵抗 1/n 突发差错损耗或修改缓冲器利用。

部分缓冲

如图 3 所示，ADRC 编码与缓冲处理发生在步骤 4。根据编码技术，编码在图像-块变换处理期间生成的 2D 或 3D 块，得到 2D 或 3D ADRC 块。一个 3D ADRC 块包含 Q 码、MIN 值、运动标志和 DR。同样地，一个 2D ADRC 块包含 Q 码、MIN 和 DR。但是，由于对单个帧或单个场执行编码，所以 2D ADRC 块不包括运动标志。

在现有技术找到许多缓冲技术（例如，参见 Kondo 等人的美国专利号 4,845,560“ High Efficiency Coding Apparatus ”和 Kondo 的美国专利号 4,722,003“ High Efficiency Coding Apparatus ”）。这两个“High Efficiency Coding Apparatus”专利引入在此作为参考。

下述的部分缓冲处理描述用于确定在 ADRC 编码中利用的编码比特的一种创新方法。特别地，部分缓冲描述在限制差错传播的同时从设计为在远程终端之间提供恒定传输速率的门限表中选择门限值的一种方法。在可选择的一个实施例中，门限表还设计为提供最大的缓冲器利用。在一个实施例中，缓冲器是存储来自给定帧集的编码数据的六十分之一部分的存储器。门限值用于确定编码从上述的图像-块变换处理中生成的 2D 或 3D 块中的像素所使用的 Q 比特的数量。

此门限表包括也称为门限集的门限值行，并且门限表中的每一行利用门限索引来标引。在一个实施例中，门限表利用生成位于此门限表的上面行中更高数量的 Q 码比特的门限集来构造。因此，对于具有

预定数量的可利用比特的给定缓冲器，编码器 110 下移此门限表，直至遇到生成小于预定数量的比特的门限集。合适的门限值用于编码此缓冲器中的像素数据。

在一个实施例中，需要不大于 30Mb/s 的传输速率。所需要的传输速率得到可用于任意一个给定缓冲器中 VL 数据存储的 31152 个比特。因此，对于每个缓冲器，计算累积的 DR 分布并从此门限表中选择门限集来将 3D 或 2D 块中的像素编码为 VL 数据。

图 8 表示所选择的门限值和用于缓冲器 0 的 DR 分布的一个实施例。图 8 的垂直轴包括累积的 DR 分布。例如，值“b”等于其 DR 大于或等于 L_3 的 3D 或 2D 块的数量。水平轴包括可能的 DR 值。在一个实施例中，DR 值的范围从 0 到 255。门限值 L_4、L_3、L_2 与 L_1 描述用于确定缓冲器的编码的门限集。

在一个实施例中，利用门限值 L_4、L_3、L_2 与 L_1 编码存储在缓冲器 0 中的所有块。因此，具有大于 L_4 的 DR 值的块使其像素值利用 4 个比特来编码。同样地，利用 3 个比特来编码属于具有 L_3 与 L_4 之间的 DR 值的块的所有像素。利用 2 个比特来编码属于具有 L_2 与 L_3 之间的 DR 值的块的所有像素，利用 1 个比特来编码属于具有 L_1 与 L_2 之间的 DR 值的块的所有像素。最后，利用 0 个比特来编码属于具有小于 L_1 的 DR 值的块的所有像素。选择 L_4、L_3、L_2 与 L_1，以使用于编码缓冲器 0 中所有块的比特总数尽可能接近 31152 个比特的极限值而不超出 31152 的极限值。

图 8a 表示一个实施例中部分缓冲的使用。帧 800 进行编码并存储在缓冲器 0－59 中。假定传输差错禁止数据恢复，对帧 800 进行解码处理，直至对丢失的数据执行差错恢复。然而，部分缓冲限制缓冲器内的差错传播，从而允许其余缓冲器的解码。在一个实施例中，传输差错禁止缓冲器 0 中块 80 的 Q 比特与运动标志恢复。部分缓冲限制差错传播到缓冲器 0 中的剩余块。因为由于固定的缓冲器长度而知道缓冲器 0 的末尾和缓冲器 1 的开头，所以差错传播限制到缓冲器 0。因此，解码器 120 能立即开始缓冲器 1 内的块的处理。另外，使用不同的门限集来编码不同的缓冲器允许编码器 110 最大化/控制包括在给定缓冲器中的 Q 码的数量，从而允许更高的压缩比率。而且，因为缓冲器 0－59 由固定长度组成，所以部分缓冲处理允许恒定的传输速

率。

在一个实施例中，因为存在有限数量的门限集，所以缓冲器的可变空间不完全利用Q码比特来填充。因此，固定长度缓冲器中的其余比特利用称为后置码的预定比特流模式来填充。如随后将要描述的，因为后置码在缓冲器的末尾之前描述VL数据的末尾，所以后置码能进行双向数据恢复。

缓冲器内YUV块混洗

Y、U与V均具有独特的统计特性。为改善（下述的）Q比特与运动标志恢复处理，在缓冲器内多路复用Y、U与V信号。因此，传输损耗对特定信号没有显著的影响。

图9表示缓冲器内YUV块混洗处理的一个实施例，其中分别从Y、U与V信号中导出YUV ADRC块。缓冲器900表示帧集块内混洗之后ADRC块分配。缓冲器900包括在11个U－ADRC块之前的66个Y－ADRC块，在这11个U－ADRC块之后又为11个V-ADRC块。缓冲器910表示在缓冲器内YUV块混洗之后YUV ADRC块结构。如所示的，三个Y－ADRC块之后为一个U－ADRC块或三个Y－ADRC块之后为一个V－ADRC块。缓冲器内YUV块混洗减少缓冲器内相邻块的比特流之间的相似性。利用不同信号（即，YUV比率或其他彩色空间）的缓冲器内YUV块混洗的可选择实施例根据初始图像格式是有可能的。

组内VL数据混洗

组内VL数据混洗包括三个处理步骤。这三个处理步骤包括Q码链接、Q码重新分配和随机化链接的Q码。图10表示组内VL数据混洗的一个实施例，其中对存储在缓冲器中的Q码连续采用三个处理步骤。在可选择的一个实施例中，在组内VL数据混洗中采用处理步骤的子集。每个处理步骤独立地辅助在传输期间丢失的数据的差错恢复。因此，分别描述每个处理步骤。下面在数据恢复的讨论中提供差错恢复的详细描述。

1．Q码链接

Q码链接保证一起解码ADRC块组。因为在下面详细描述的数据恢复处理期间可从相邻块中获得附加信息，所以组解码便于差错恢复。对于一个实施例而言，对缓冲器中存储的三个ADRC块的每个组独立地进行Q码链接。在另一可选择实施例中，组包括来自不同缓冲器的

ADRC 块。三个 ADRC 块上的 Q 码链接描述为生成一个链接的 ADRC 瓷砖。图 11 与图 11a 表示生成链接的 ADRC 瓷砖的一个实施例。

图 11 表示从 2D ADRC 块中生成链接的 ADRC 瓷砖的一个实施例。特别地，对包括在 2D ADRC 块 0、1 与 2 中的每个 Q 码（q_0-q_{63}）执行链接，得到链接的 ADRC 瓷砖 A 的 64 个 Q 码。例如，2D ADRC 块 0 的第一 Q 码 $q_{0,0}$（第 0 量化值）链接到 2D ADRC 块 1 的第一 Q 码 $q_{0,1}$。这两个链接的 Q 码又链接到 2D ADRC 块 2 的第一 Q 码 $q_{0,2}$，从而得到链接的 ADRC 瓷砖 A 的 Q_0。重复这些处理，直至生成 Q_{63}。或者，利用下面等式来描述链接的 ADRC 瓷砖 A 中 Q_i 的生成：

$$Q_i=[q_{i,0},\ q_{i,1},\ q_{i,2}] \quad i=0,1,2,\ldots 63$$

另外，与链接的 ADRC 瓷砖 A 中每个 Q_i 相关具有表示链接生成单个 Q_i 的比特总数的相应数量的 N 比特。

图 11a 表示从包括运动块的帧对中生成链接的 ADRC 瓷砖的一个实施例。运动块是具有设置运动标志的 3D ADRC 块。当在利用上述的图像-块变换处理生成的两个 2D 块结构内像素的预定数量在第一帧与后一帧之间的值中变化时，设置运动标志。在另一可选择实施例中，当在第一帧与后一帧的 2D 块之间的每个像素变化的最大值超过预定值时，设置运动标志。相反地，无运动（即，静止）块包括具有未设置的运动标志的 3D ADRC 块。当在第一帧与后一帧的两个 2D 块内像素的预定数量的值未变化时，运动标志保持不设置。在另一可选择实施例中，当在第一帧与后一帧之间的每个像素变化的最大值没有超过预定值时，运动标志保持不设置。

运动块包括来自第一帧中编码的 2D 块与后一帧中编码的 2D 块的 Q 码。相应于单个编码的 2D 块的 Q 码的集合称为 ADRC 瓷砖。因此，一个运动块生成两个 ADRC 瓷砖。然而，由于缺乏运动，静止块只需包括运动块一半数量的 Q 码，从而只生成一个 ADRC 瓷砖。在本实施例中，通过平均第一帧中的 2D 块与后一帧中相应的 2D 块之间相应的像素值来生成静止块的 Q 码。随后编码每个平均的像素值，得到形成单个 ADRC 瓷砖的 Q 码的集合。因此，运动块 1110 与 1130 生成 ADRC 瓷砖 0、1、3 与 4。静止块 1120 生成 ADRC 瓷砖 2。

图 11a 的链接的 ADRC 瓷砖生成将 ADRC 瓷砖 0－4 的 Q 码链接为链接的 ADRC 瓷砖 B。特别地，对包括在 ADRC 瓷砖 0、1、2、3 与 4

中的每个 Q 码（q_0-q_{63}）执行链接，得到链接的 ADRC 瓷砖 B 的 64 个 Q 码。或者，利用下面的数学等式来描述链接的 ADRC 瓷砖 B 中的每个 Q 码（Q_i）的生成：

$$Q_i=[q_{i,0},\ q_{i,1},\ q_{i,2},\ q_{i,3},\ q_{i,4}] \qquad i=0,1,2,...63$$

2. Q 码重新分配

Q 码重新分配保证在空间上不连续的像素内定位由于传输损耗而引起的比特差错。特别地，在 Q 码重新分配期间，Q 码进行重新分布并混洗重新分布的 Q 码的比特。因此，因为未损坏的像素包围着每个损坏的像素，所以 Q 码的重新分配便于差错恢复。而且，因为像素损坏均匀分布在整个 ADRC 块上，所以有助于 DR 与 MIN 恢复，下面在数据恢复讨论中具体描述 DR 与 MIN 恢复。

图 12 表示在 1/6 突发差错损耗的传输损耗期间像素恶化的一个实施例。特别地，2D ADRC 块 1210、1220 与 1230 均包括利用 3 个比特编码的 64 个像素。因此，2D ADRC 块的每个像素（从 P0 到 P63）利用三个比特来表示。2D ADRC 块 1210 表示在丢失每六个比特之中的第一比特时比特的比特损耗模式，这利用暗方框来表示。同样地，分别以 2D ADRC 块 1220 与 1230 来表示每六个比特之中第二比特或第四比特丢失时的比特损耗模式。图 12 表示在 Q 码不重新分配的情况下 2D ADRC 块 1210、1220 与 1230 的所有像素的一半由于 1/6 突发差错损耗而被破化。

就一个实施例而言，Q 码重新分配独立应用于存储在缓冲器中的每个链接的 ADRC 瓷砖，因而保证在去混洗时在空间上不相交的像素内定位比特差错。在另一可选择实施例中，Q 码的重新分配应用于存储在缓冲器中的每个 ADRC 块。

图 12a 表示从链接的 ADRC 瓷砖中生成混洗的 Q 码比特的比特流的 Q 码重新分配的一个实施例。表 122 与表 132 表示 Q 码重新分布。比特流 130 与 140 表示 Q 码比特的混洗。

表 122 表示用于链接的 ADRC 瓷砖 A 的链接的 Q 码。Q_0 是第一链接的 Q 码，而 Q63 是最后一个链接的 Q 码。表 132 表示 Q 码重新分布。对一个实施例而言，Q_0、Q_6、Q_{12}、Q_{18}、Q_{24}、Q_{30}、Q_{36}、Q_{42}、Q_{48}、Q_{54} 与 Q_{60} 包括在第一集、分区 0 中。根据表 132，后面的 11 个链接的 Q 码包括在分区 1 中。这些步骤对分区 2－5 重复。利用表 132 中的垂直

线来描述分区的边界。链接Q码的此不相交的空间分配到六个分区保证1/6突发差错损耗导致分布在一组连续像素上的比特损耗模式。

图12b表示由于重新分布的Q码的1/6突发差错损耗而引起的比特模式损耗的一个实施例。特别地，2D ADRC块1215、1225与1235均包括利用三个比特编码的64个像素。因此，每个2D ADRC块的每个像素P_0-P_{63}利用三个比特来表示。在2D ADRC块1215、1225与1235中，利用暗方框表示的比特损耗模式定位在一组连续的像素上。因此，对于给定的分段损耗来说，每个2D ADRC块1215、1225与1235内只有11个连续的像素被破化。在另一可选择实施例中，Q码分配分区包括来自不同运动块的Q码，从而给六个分段提供Q码的不相连的空间与时间分配。这在1/6突发差错损耗期间得到另外的未损坏的空间-时间像素并且进一步有助于强有力的差错恢复。

参见图12a，表132中重新分布的Q码的比特在生成的比特流上进行混洗，于是比特流中相邻的比特来自相邻的分区。表132中所有分区的Q码比特链接到比特流130中。对于给定的分区，比特流130中相邻比特分散到所生成的比特流140中的每一个第六比特位置上。因此，比特流140的比特号0-5包括每个分区中第一Q码的第一比特。同样地，比特流140的比特号6-11包括每个分区中第一Q码的第二比特。对所有的Q码比特重复此处理。因此，1/6差错损耗将导致空间上不相连的像素损耗。

图12c表示重新分配（即，重新分布与混洗）的Q码的1/6突发差错损耗所引起的比特模式损耗的一个实施例。特别地，2D ADRC块1217、1227与1237均包括利用三个比特编码的64个像素。因此，每个2D ADRC块的每个像素P0-P63利用三个比特来表示。在2D ADRC块1217、1227与1237中，利用暗方框表示的比特损耗模式分布在空间上不相连的像素上，从而有助于像素差错恢复。

3. Q码比特的随机化

Q码比特利用掩蔽密钥进行随机化，以辅助解码器恢复丢失与损坏的数据。特别地，在编码期间利用KEY表示的密钥用于掩蔽Q码的比特流。因此，解码器必须识别KEY的正确值来去混洗（unmask）Q码的比特流。

在一个实施例中，KEY用于掩蔽利用三个ADRC块的Q码重新分配

生成的Q码的比特流。如前文所述，一个ADRC块包括FL数据与Q码。利用FL数据值和与相应的ADRC块相关的量化比特（“q_i”）数量的组合来生成掩蔽密钥的每个密钥元素（“d_i”）。在一个实施例中，运动标志与Q比特用于定义一个密钥。因此，在此实施例中，从下面的数学等式中生成密钥元素的值：

$$d_i=5 \cdot m_i+q_i \qquad \text{其中 } i = 0, 1, 2 \text{ 和 } q_i=0, 1, 2, 3, 4$$

变量m_i等于运动标志。因此，在相应的ADRC块为静止块时，m_i等于0，而在相应的ADRC块为运动块时，m_i等于1。而且，变量q_i表示用于编码相应的ADRC块的量化比特。因此，q_i具有用于四比特ADRC编码技术的0、1、2、3或4的值。在一个实施例中，根据下面的等式利用三个密钥元素（“d_i”）定义用于三个ADRC块的一个组的KEY：

$$KEY = d_0+10.d_1+100.d_2$$

因而，在运动标志或Q比特数据的恢复期间，根据用于生成掩蔽密钥的值来再生可能的密钥值。所再生的密钥值用于去混洗接收的Q码的比特流，得到候选的解码。下面在数据恢复的讨论中提供再生密钥值与特定候选解码选择的具体描述。

在另一可选择实施例中，从各种密钥元素中生成掩蔽密钥。因而，给解码器提供涉及一个密钥元素的特定信息而不必通过传输介质发送此元素。在一个实施例中，与ADRC块相对应的DR或MIN值用于生成掩蔽密钥来掩蔽表示此ADRC块的比特流。

图10-12表示容许传输期间高达1/6分组数据损耗的组内VL数据混洗。本领域技术人员将意识到，能改变总的分区数量与比特间隔，以保证抵抗1/n突发差错损耗。

分段之间FL数据混洗

分段之间FL数据混洗描述在不同分段之间重新调整块属性。重新调整块属性提供分布的数据损耗。特别地，当在传输期间丢失来自一个分段的FL数据时，丢失的DR值、MIN值与运动标志值不属于同一块。图13与14表示分段之间FL数据混洗的一个实施例。

图13表示分段0-5的内容。对一个实施例而言，每个分段包括880个DR，880个MIN、880个运动标志和与660个Y块、110个U块及110个V块相对应的VL数据。如图MIN混洗1300所示的，用于

分段 0 的 MIN 值移到分段 2，用于分段 2 的 MIN 值移到分段 4，而用于分段 4 的 MIN 值移到分段 0。另外，用于分段 1 的 MIN 值移到分段 3，用于分段 3 的 MIN 值移到分段 5，而用于分段 5 的运动标志移到分段 1。

图 13a 表示运动标志混洗。如所示的，在图运动标志混洗 1305 中，用于分段 0 的运动标志值移到分段 4，用于分段 2 的运动标志值移到分段 0，而用于分段 4 的运动标志值移动到分段 2。另外，用于分段 1 的运动标志值移到分段 5，用于分段 3 的运动标志值移到分段 1，而用于分段 5 的运动标志值移到分段 3。损耗模式 1310 表示传输期间分段 0 丢失之后 FL 数据损耗。

对于特定的块属性，图 13 与图 13a 都表示在分段之间混洗特定块属性的所有示例。例如，在图 13 中，来自分段 0 的 880 个 MIN 值与分段 2 中的 880 个 MIN 值一起进行交换。同样地，在图 13a 中，用于分段 0 的 880 个运动标志与分段 4 中的 880 个运动标志一起进行交换。在连续分组的传输损耗期间，此块属性的集中混洗导致块组的不成比例的特定块属性的损耗。在一个实施例中，一个块组包括三个 ADRC 块。

图 14 表示用于 DR、MIN 与运动标志值的模块化三混洗处理的一个实施例。模块化三混洗描述三个不同分段中的三个块（即，一个块组）上共享的混洗模式。对于三个不同分段内的所有块组，重复此混洗模式。然而，对于不同的组属性，使用不同的混洗模式。因此，模块化三混洗处理在所有三个分段上分布块属性。特别地，对于给定的一个块组，模块化三混洗保证在分段传输损耗期间只丢失一种特定块属性示例。因此，在下述的数据恢复处理期间，生成减少数量的候选解码来恢复块内的数据损耗。

如 DR 模块化混洗 1410 中所示，一个分段存储 880 个 DR 值。因此，根据从中导出给定 DR 值的块给这些 DR 值编号 0－879。在模块化三混洗中，混洗三个分段的 FL 数据的内容。0－2 的计数用于识别用于模块化混洗的三个分段中的每个 DR 值。因此，属于编号为 0，3，6，9...的块的 DR 值属于计数 0。同样，属于编号为 1，4，7，10...的块的 DR 值属于计数 1，而属于编号为 2，5，8，11...的块的 DR 值属于计数 2。因而，对于给定的计数，与那个计数相关的 DR 值在分段 0、

2 与 4 上进行混洗。同样，与同一计数相关的 DR 值在分段 1、3 与 5 上进行混洗。

在 DR 模块化混洗 1410 中，属于计数 0 的 DR 值不进行混洗。属于计数 1 的 DR 值进行混洗。特别地，分段 A 中的计数 1 DR 值移到分段 B，分段 B 中的计数 1 DR 值移到分段 C，而分段 C 中的计数 1 DR 值移到分段 A。

属于计数 2 的 DR 值也进行混洗。特别地，分段 A 中的计数 2 DR 值移到分段 C，分段 B 中的计数 2 DR 值移到分段 A，而分段 C 中的计数 2 DR 值移到分段 B。

MIN 模块化混洗 1420 表示 MIN 值的模块化三块属性混洗处理的一个实施例。一个分段包括 880 个 MIN 值。在 MIN 模块化混洗 1420 中，用于 DR 模块化混洗 1410 中的计数 1 与计数 2 的混洗模式移到计数 0 与计数 1。特别地，用于 DR 模块化混洗 1410 中的计数 1 的混洗模式应用于计数 0。用于 DR 模块化混洗 1410 中的计数 2 的混洗模式应用于计数 1，而属于计数 2 的 MIN 值不进行混洗。

运动标志模块化混洗 1430 表示运动标志值的模块化三块属性混洗处理的一个实施例。一个分段包括 880 个运动标志值。在运动标志模块化混洗 1430 中，用于 DR 模块化混洗 1410 中的计数 1 与计数 2 的混洗模式分别移到计数 2 与计数 0。特别地，用于 DR 模块化混洗 1410 中的计数 2 的混洗模式应用于计数 0。用于 DR 模块化混洗 1410 中的计数 1 的混洗模式应用于计数 2，并且属于计数 1 的运动标志值不进行混洗。

图 14a 表示模块化混洗 1410、1420 与 1430 的模块化混洗结果。模块化混洗结果 1416 表示属于分段 0 的块的每个属性目的地。在此示例中，分段 0 对应于图 14 的分段 A。根据图 14 的模块化混洗 1410、1420 与 1430 来定义此目的地。图 14a 也表示在传输期间丢失分段 0 之后块属性的分布损耗。特别地，损耗模式 1415 表示在对初始利用模块化混洗 1410、1420 与 1430 进行混洗的接收数据进行随后的去混洗之后 6 个分段上的 DR、运动标志与 MIN 值损耗。如图 14a 所示，块属性损耗周期性地分布在分段 0、2 与 4 上，而分段 1、3 与 5 没有块属性损耗。因此，空间损耗模式 1417 表示在传输期间丢失分段 0 之后损坏的 FL 数据的去混洗的空间分布。空间损耗模式 1417 表示在对

接收的数据进行随后的去混洗之后 DR、运动标志与 MIN 值损耗。在空间损耗模式 1417 中，损坏的块四周是未被损坏的块，并且能利用四周未被损坏的块来恢复损坏的块属性。

图 14 与 14a 表示模块化三混洗模式与在传输期间丢失分段之后块属性的分布损耗。在可选择的一个实施例中，改变计数变量或分段数量来改变丢失的块属性的分布。图 14 b 表示模块化混洗结果 1421 与损耗模式 1420。同样地，图 14c 表示模块化混洗结果 1426 与损耗模式 1425。损耗模式 1420 与损耗模式 1425 都表示与前述的 3 个分段相对的 6 个分段上的块属性的分布损耗。

打算在可选择的实施例中将分布块属性的各种组合来执行混洗处理。

分段之间 VL 数据混洗

在分段之间 VL 数据混洗处理期间，安排例如 6 个分段的预定数量的分段之间的比特来保证在高达 1/6 分组传输损耗期间空间隔开与周期性的 VL 数据损耗。图 15 与 16 表示分段之间 VL 数据混洗处理的一个实施例。

在本实施例中，希望达到 30Mb/s 的传输速率。因此，此所需的传输速率导致可用于 60 个缓冲器之中每一个缓冲器中 VL 数据的 31152 个比特。其余空间由包括在一个缓冲器中的 88 个块的 FL 数据使用。图 15 包括达到 30Mb/s 传输速率的帧集内的 VL 数据缓冲器结构。如上所述，部分缓冲用于最大化每个缓冲器内可利用的 VL 数据空间的使用，并且未使用的 VL 空间利用后置码来填充。

图 16 表示保证空间上隔开与周期性 VL 数据损耗的混洗处理的一个实施例。第一行表示来自图 15 的 60 个缓冲器的 VL 数据重新安排在链接的 1869120 比特流中。第二行表示新的比特流中每个第六比特的集合。因而，在解码器随后反向此处理时，高达 1/6 的发送数据的突发损耗变换为周期性的损耗，其中至少 5 个未被损坏的比特隔开每个集合的两个损坏的比特。

第三行表示将流 2 的每 10 个比特组合为新的比特流，即流 3。组合的边界也利用一个分段中的比特数量来定义。每个第十比特的流 2 的组合保证 1/60 数据损耗得到每个集合的两个损坏的比特之间 59 个未被损坏的比特，这在丢失 88 个连续的数据分组的情况中提供空间

上隔开与周期性的 VL 数据损耗。

第四行表示将流 3 的每 11 个比特组合为流 4。组合边界也利用一个分段中的比特数量来定义。每个第 11 比特的流 3 的组合保证 1/660 数据损耗得到两个损坏的比特之间 659 个未被损坏的比特，导致 8 个连续分组的传输损耗期间空间上隔开与周期性的 VL 数据损耗。

流 4 内的每组 31152 个比特连续存储在缓冲器 0-59 中，其中第一组的比特存储在缓冲器 0 中，而最后一组的比特存储在缓冲器 59 中。

本领域技术人员将意识到，图 16 的组合要求可用于保证高达 1/n 传输损耗的空间上隔开与周期性的 VL 数据损耗容限。

传输

上述的混洗处理生成具有混合的 FL 数据与 VL 数据的缓冲器。对于一个实施例来说，分组根据分组结构 200 从每个缓冲器中生成并通过传输介质 135 进行发送。

数据恢复

如上所述，用于编码数据比特流的创新方法能进行一般由于丢失的数据分组而出现的数据的强有力恢复，已经在图 4 中示出此解码处理的概要。

参见图 4，利用多级去混洗处理（步骤 425、430、435 与 440）来处理分组中接收的数据，其中利用分组接收的比特流的不同级或部分进行去混洗来检索数据。然后根据本领域（例如，1991 年 9 月 4-6 日在意大利都灵召开的 Fourth International Workshop on HDTV and Beyond 会议上 Kondo、Fujimori、Nakaya 的文章“Adaptive Dynamic Coding Scheme for Future HDTV Digital VTR”）公知的教导对此数据进行 ADRC 解码。

随后执行帧集块内去混洗并接着执行块-图像变换，步骤 450、455。步骤 425、430、435、440、445、450 与 455 是为编码数据而执行的上述处理步骤的逆处理并且在此不再进行详细描述。然而，应注意：在一个实施例中，利用步骤 425、430 与 440 表示的去混洗级是数据独立的。例如，所执行的去混洗处理利用地址变换或表查找来预定或规定。由于去混洗步骤 425、430 与 440 和数据内容无关，所以例如由于分组损耗而引起的数据损耗不阻止执行去混洗步骤。同样

地，步骤 450 与 455 是数据独立的。然而，组内 VL 数据去混洗处理取决于数据的内容。更特别地，组内 VL 数据去混洗处理用于确定用于这些组的块的量化码。因此，在步骤 435，如果丢失分组，则不能处理受影响的组。

在执行去混洗、解码与变换（步骤 425、430、435、440、445、450 与 455）之后，执行恢复处理，以恢复位于丢失的分组中的 Q 比特与运动标志值。一般由于 DR 损耗（由于丢失的分组）而丢失 Q 比特值。在不知道 Q 比特或运动标志值时，不能从数据比特流中确定像素的 Q 码比特。如果未正确确定 Q 比特或运动标志值，则此差错将传播为其中将不正确识别缓冲器中数据的后续块的开始点。

图 17 描述用于恢复 Q 比特与运动标志值的一般处理。此特别实施例描述利用多个数据块来恢复 Q 比特与运动标志值的处理，然而，打算特定数量的块不受本文讨论的限制并能是一个或多个块。参见图 17，根据比特流中差错的检测，步骤 1705，为所检查的 3 个块生成基于特定参数的候选解码。在步骤 1715，每个候选解码根据其是正确解码的可能性进行评分，并在步骤 1720，使用具有最佳评分的候选解码，此解码识别能进行受影响的块的像素的后续解码的 Q 比特与运动标志值。

再参见图 4 的解码处理，一旦选择最佳的解码，恢复由于丢失的分组而丢失的任意 DR 或 MIN 值，步骤 465。能采用本领域技术人员公知的各种恢复处理来恢复 DR 与 MIN，包括相邻块的值的最小平方或平均值。例如，参见 1991 年 9 月 4-6 日在意大利都灵召开的 Fourth International Workshop on HDTV and Beyond 会议上 Kondo、Fujimori、Nakaya 的文章 “Adaptive Dynamic Coding Scheme for Future HDTV Digital VTR”。在本实施例中，创新的图像-块变换处理与从中生成的数据结构增加相邻块的数量，从而提供附加的数据并有助于更准确的 DR 或 MIN 恢复。特别地，在一个实施例中，根据下式恢复 DR 与 MIN:

$$DR = \frac{m \cdot \sum_i (y_i - MIN) \cdot q_i}{\sum_i q_i^2}$$

其中 DR’对应于恢复的 DR，q_i 是 ADRC 块中的第 i 个值并且

$q_i \in \{0, 1, \ldots 2^Q-1\}$；对于边缘匹配 ADRC，$m=2^Q-1$，而对于非边缘匹配 ADRC，$m=2^Q$；$Y_i$ 是相邻块像素的解码值；Q 是 Q 比特值；和

$$MIN' = \frac{\sum_i \left(y_i - \frac{DR}{m} \cdot q_i \right)}{N}$$

其中 MIN'对应于恢复的 MIN 和 N 是相加中使用的项的数量（例如，当 i = 0 - 31 时，N = 32）。在另一实施例中，如果同一块的 DR 与 MIN 同时被损坏，根据下面等式来恢复 DR 与 MIN:

$$DR' = \frac{m \cdot \left[N \cdot \sum_i q_i \cdot y_i - \sum_i q_i \cdot \sum_i y_i \right]}{N \cdot \sum_i q_i^2 - \left[\sum_i q_i \right]^2}$$

$$MIN' = \frac{\sum_i \left(y_i - \frac{DR'}{m} \cdot q_i \right)}{N}$$

在步骤 470，在步骤 475 对先前未在 Q 比特与运动标志恢复之前解码的那些块进行 ADRC 解码并执行像素恢复处理来恢复由于丢失的分组或随机差错而可能出现的任何有差错的像素数据。另外，在步骤 480 执行 3: 1: 0→4: 2: 2 后变换，以便以显示所需的格式来设置图像。

图 18 表示本发明的解码处理的 Q 比特与运动标志恢复处理的一个特殊实施例。在此特殊实施例中，此处理的输入是相邻块信息，和三个块的块属性与像素数据将进行处理。也输入表示丢失数据位置的差错标志。这些差错标志能以本领域技术人员公知的各种方法来生成并且除了假定这些标志表示哪些比特是利用损坏的或丢失的分组发送的之外在本文将不再进一步进行讨论。

在步骤 1805，生成候选解码。能以各种方式来生成候选解码。例如，虽然处理负担相当重要，但候选解码能包括所有可能的解码。或者，能根据预先规定的参数生成候选解码来减小要进行评估的候选解码的数量。

在本实施例中，根据用于随机化上述的组内 VL 数据混洗处理的

比特流的可能的密钥值确定候选解码。另外，应注意：候选解码还受仍要进行解码的比特的长度和多少块保留的知识的限制。例如，如下文将要讨论的，如果处理最后的一块，一般知道那个块的解码长度。

继续本实施例，图 19 表示本实施例的可能情况，其中值 X 表示（可能由于分组损耗而）未知的值。这利用示例进一步进行解释，m_i 定义为第 i 块的运动标志，q_i 是第 i 块的量化比特的数量，n_i 是第 i 块可能候选的数量，而 d_i 是组内 VL 数据混洗中上述的第 i 块的密钥元素的值。在每个组内定义第 i 块。在此示例中，每组内的块数量为 3，用于这 3 块组的密钥生成为 $d_0+10d_1+100d_2$。假设不知道第一块中的运动标志并且量化比特数量为 2，m_0 等于 X，而 q_0 等于 2。根据上述的等式来生成密钥元素，$d_i=15 \cdot m_i+q_i$，用于 d_0 的可能的数字集合由{2 与 7}构成。因而，可能值（n_0）的数量为 2。假设第二块具有值为 1 的运动标志值和一个量化比特，并且 d_1 的值为 5*1 + 1 = 6 和 $n_1=1$。第三块具有为 1 的运动标志值和未知数量的量化比特。因而，数字比特 d_2 包括由{6，7，8，9}组成的集合和 $n_2=4$。因而，此组可能的候选数量 M 为 $2 \cdot 1 \cdot 4 = 8$，并且用于生成候选解码的密钥为 662、667、762、767、862、867、962、976 的变量。此处理最好用于受数据损耗影响的每一组。

再参见图 17，在步骤 1715，一旦根据密钥数据解码此数据，所生成的候选解码进行评估或根据其是此数据的正确解码的可能性进行评分。在步骤 1720，选择使用具有最佳评分的候选解码。

各种技术能用来对候选编码进行评分。例如，此评分能从特定候选解码的块的像素如何与此图像的其他像素相符的分析中导出。此评分最好根据诸如平方差错与相关性的表示差错的准则来导出。例如，对于相关性，假设相邻像素将多少有点密切相关是相当安全的。因而，缺乏相关表示候选解码是否是正确解码。

如图 18 所示，分析四种不同的准则来选择最佳的候选解码。但是，打算能分析一、二、三或更多的不同准则来选择最佳的候选解码。

参见图 18，本实施例利用随后组合为一个最后评分的四个子评分准则。特别地，在步骤 1815，生成平方差错测量，在步骤 1820，确定水平相关性，在步骤 1825，确定垂直相关性，并在步骤 1830，测量时间活动（根据 M 个候选、N 个块和数据的 2 个帧/块的每个 M 乘

2*N 矩阵）。虽然使用水平与垂直相关性，但应认识到能检查包括对角相关性的许多相关性测量。在步骤 1835、1840、1845、1850，为每个准则生成置信测量来标准化所生成的测量，并在步骤 1855、1860、1865 与 1870，生成每个不同准则的概率函数。这些概率函数然后例如通过乘以一个概率值来进行组合以生成一个评分，例如图 18 中所示的似然函数。用于候选解码的评分随后与所有候选解码评分进行比较来确定可能的候选。

应认识到：各种技术能用来评估候选解码并为每个候选生成“评分”。例如，置信测量是标准化准则的一种方式。而且，能使用除上述之外的各种置信测量。同样地，乘以基于每个准则的概率值来生成总的似然函数仅仅是组合检查的各种准则的一种方式。

编码处理有助于最佳候选解码的确定，这是因为不是可能候选的候选解码一般将具有相对差的评分，而是非常可能的候选的解码将具有明显较好的评分。特别地，组内 VL 数据混洗处理中前述的 Q 码随机化处理有助于这方面。

图 20a、20b、20C 与 20d 提供为生成特定候选解码的评分与总评分而在图 18 的步骤 1815、1820、1825 与 1830 上执行的不同测量的说明。图 20a 表示在与其解码的相邻 $Y_{i,j}$ 相比较时评估候选解码像素 X_i 的平方差错，其中下标“i，j”与“i”的相邻地址相对应。最好除去一些最大的项以消除由于尖峰值（这是由于图像中的合法边缘而引起的项）引起的任何影响。最好地，抛弃 $(X_i - Y_{i,j})^2$ 的三个最大项以消除可能出现的尖峰值。图 20b 表示时间活动准则。这只有在其是或假定是运动块时才可应用。此时间活动准则假定：候选解码越好，块之间的差异越小。因此，候选解码越坏，块之间的差异就越大。空间相关性假定：最有可能的候选解码将导致大的相关性，这是因为真实图像趋于以缓慢不变的方式变化。图 20c 所示的水平相关处理与图 20d 所示的垂直相关处理利用那个假定。

图 18 的步骤 1835、1840、1845 与 1850 上的置信测量提供用于标准化以前步骤（步骤 1815、1820、1825 与 1830）中确定的准则的处理。在一个实施例中，例如，平方差错的置信测量从间隔[0,1]中取值，其中，例如，置信度在差错相等时等于 0（例如，低置信度）并在一个差错为 0 时等于 1（例如，高置信度）。也考虑标准化的其他

测量或方法。

同样地，空间相关性的置信测量是:

最大（Y，0）－最大（X，0）

其中 Y 是最佳相关值，而 X 是当前候选解码的相关值。根据下式确定时间活动置信测量:

conf=（a-b）/（a+b）

其中 a＝最大（X，M-TH）和 b＝最大（Y，M-TH），其中 M-TH 是候选块的运动门限，而 Y 是最佳测量，即最小的时间活动，并且 X 等于时间活动的当前候选测量。

在图 18 的步骤 1855、1860、1865 与 1870 上，为每个不同准则生成概率函数。能利用许多方法来生成概率测量。例如，评分能规定为置信测量。如果置信测量大于预定的值，例如 0.8，则基本评分减 10；如果置信测量在 0.5 与 0.8 之间，则基本评分减 5。图 21 表示其中一张表用来生成用于平方差错测量准则的概率函数的一个实施例。此表包括根据经验确定的包含置信与平方差错测量的任意存储的数据和已知的候选解码。更特别地，此表可以利用未损坏的数据并假定 DR 被破坏或丢失来生成，然后生成正确或不正确解码的密钥与置信测量。此表反映正确与不正确解码的概率比率。利用此表，对于特定平方差错值（列）与置信值（行），能确定概率。例如，因此能明白：对于零的置信测量时的各种平方差错测量，具有近似 40%—50% 的候选是正确的概率。如果置信不是 0，但是小，概率显著地下降。根据相应经验确定的准则测量与置信测量为相关性与时间测量生成相似的概率表。

在本实施例中所生成的概率认为是生成“评分”的数据，并如前所述，可以使用其他的技术来对候选解码进行评分。在步骤 1875，不同的概率组合为似然函数 $L_i=\pi j*P_{i,j}$，其中 πj 是概率函数 $P_{i,j}$ 的放大函数，而 $P_{i,j}$ 是候选 I、块 j 的概率函数，因此此候选选择为最大化函数 L_i 的一个候选。

再参见图 18，可能需要恢复在丢失的分组中发送的某些块属性。因此，在步骤 1810，在需要时恢复 DR 与 MIN 值。可以利用从缺省值、平均、平方差错函数到更复杂技术的包括在 1993 年 9 月 20-22 日在澳大利亚墨尔本召开的 IEEE Visuol Signal Proccssing and

Communications 会议上 Kondo、Fujimori、Nakaya 的文章“Adaptive Dynamic Range Coding Scheme for Future HDTV Digital VTR”以及 Kondo、Fujimori、Nakaya 与 Uchida 的文章“A New Concealment Method for Digital VCR”中讨论的各种技术来如上所述生成候选解码。

或者，在 Q 比特确定处理期间确定 DR 与 MIN 值，这在图 22 示出。特别地，如前所述，在本实施例中，运动标志与量化比特的数量用于编码处理中并在以后在恢复处理期间进行使用来减少可能的候选解码的数量。如前所述，也能使用其他的信息。因而，DR 的值和/或 MIN 的值也可以用于编码此数据。或者，DR 比特的一部分用于编码（例如，DR 的二个最低有效比特）。虽然编码 DR 数据，但在增加变量时可能的候选解码的数量明显地增加。参见图 22，因此生成 K*.M 候选解码，其中 K 是未知数据的候选值的数量，例如，如果编码 DR_1、DR_2 与 DR_3 之和的二个比特（DR_1、DR_2 与 DR_3 代表此组的块的 DR 值），则 K = 4。因此，利用提供的辅助信息，例如 DR_1、DR_2 与 DR_3 之和的编码的二个比特来恢复 DR 与 MIN。这以附加开销为代价来改善候选选择的处理，以检查更大数量的候选解码。

应注意：一般地，解码的相邻块越多，Q 比特与运动标志恢复处理就越好。而且，在一些实施例中，此处理应用于缓冲器的每个后续块；如果所有或一些 FL 数据可利用，则能减少候选解码的数量，有可能减少到一个候选解码，假定一个块的所有 FL 数据可利用的话。但是，由于 Q 比特与运动标志恢复处理是相对耗时的一个处理，所以希望一起避免此 Q 比特与运动标志恢复处理。而且，希望使用尽可能多的信息来执行 Q 比特与运动标志恢复处理。在一个实施例中，从缓冲器的开头开始处理块，直至到达具有丢失的 Q 比特/运动标志信息的块，这定义为前向 Q 比特与运动标志恢复。在另一个实施例中，参照缓冲器的末尾来确定缓冲器的最后一块的末尾位置并从此缓冲器的末尾中恢复数据，直至到达具有丢失的 Q 比特/运动标志数据的块。这定义为后向 Q 比特与运动标志恢复。

如前所述，块的长度由于 VL 数据的长度而是变化的，因此需要确定形成块的 VL 数据的比特的数量，以便能准确定位此缓冲器中后续块的位置。在编码处理期间，预定的与最好易于识别的模式的后置

码放置在缓冲器中以填充未用的比特位置。在解码处理期间，后置码将放置在此块与缓冲器的末尾之间。由于此模式是易于识别的，所以比特模式的检查能使系统定位后置码的开头并因此定位此缓冲器中最后一块的末尾。此信息能以两种方式来使用。如果最后一块包含损坏的 Q 比特/运动标志数据并已知最后一块的开头（例如，已成功解码前面的块），紧接在前面的块的末尾与后置码的开头之间的差与此块的长度相对应。此信息能用于计算块的 Q 比特和/或运动标志。后置码的开始位置也能用于执行在此缓冲器的最后一块上开始并朝向此缓冲器的开头继续的 Q 比特与运动标志恢复。因而，能双向实施 Q 比特与运动标志恢复处理。

图 23 表示在双向 Q 比特与运动标志恢复处理中后置码的使用。参见图 23，缓冲器 2300 包括 VL 数据块的 N 个组的 FL 数据 2303。每组由多个块（例如，3 个块）构成。在本实施例中，解码前面两个组 2305、2310，并由于损坏的 DR/运动标志数据而不能立即解码第三组 215。此时，为了恢复损坏的数据而要求 Q 比特/运动标志恢复处理。不以前向方向继续处理这些组，此处理反而参照通过查找后置码模式 220 确定的缓冲器的末尾。确定后置码的开头并因此确定块的最后一组的末尾。因为 DR/运动标志数据表示 VL 数据的长度，所以确定最后一块的 VL 数据的开头并因此确定紧接在前面的一块的末尾。因此，能解码这些块，例如块 225、230、235，直至到达具有损坏数据的块 240。然后最好利用上述的 Q 比特/运动标志恢复处理来恢复损坏的 215、240 和阻塞的块 250。

应注意：双向处理不限于前向与反向处理的顺序；处理能以任一方向或双向进行。而且，在一些实施例中，可以要求并行执行这样的处理来提高效率。最后，打算通过直接存取 Q 比特/运动标志信息可以恢复未损坏的阻塞的块而不执行上述的 Q 比特/运动标志恢复处理。

如上所述，各种评分技术可以用来确定选择为解码的最佳候选解码。在另一可选择实施例中，评估使用每个候选解码的图像的平滑度。在一个实施例中，完成 Laplacian（拉氏）测量。Laplacian 测量测量第二阶图像表面特性，例如表面曲率。对于线性图像表面，即光滑平面，Laplacian 测量将得到近似为 0 的值。

将参考图 24a、24b 与 24c 来说明此处理。图 24a 表示 Laplacian 内核的一个实施例。打算也可以使用其他的实施例。内核“L”表示 3×3 区域。为测量图像区域的平滑度，图像（图 24b）的 3×3 子区域利用内核进行卷积并平均卷积值。能根据应用改变区域与子区域的尺寸（并因此改变内核尺寸）。

参考图 24c 描述此处理的一个实施例。此实施例利用内核和 3×3 的子区域尺寸以及 8×8 的区域尺寸，利用下标 i，j 来识别各个元素。在步骤 2460，标准化候选解码的值 X[i][j]。例如，可根据下式标准化这些值：

$$\bar{x}[i][j]=\frac{x[i][j]}{\sqrt{\sum_{i,j}\left(x[i][j]-X_{mean}\right)^2}},\quad 0\le i,j<8$$

$$\text{其中，}X_{mean}=\frac{\sum_{i,j}x[i][j]}{64},\quad 0\le i,j<8$$

在步骤 2465，这些标准化的值用于根据下式计算表示平滑度的块 Laplacian 值 Lx。

$$l[i][j]=\sum_{m=-1}^{1}\sum_{n=-1}^{1}L[m][n]\cdot x'[i+m][j+n],\quad 0\le i,j<8$$

$$L_x=\frac{\sum_{i,j}|l[i][j]|}{64}$$

块 Laplacian 值越接近 0，图像部分就越平滑。因而，能根据块 Laplacian 值测量评分，并且具有最小的 Laplacian 值的解码为正确的解码。

也能利用候选编码的值 q[i][j]来实现 Laplacian 评估。基本处理与图 24c 的候选解码值情况相同。此实施例使用内核与 3×3 的区域尺寸以及区域尺寸 8×8，利用下标 i，j 来识别各个元素。在步骤 2460，标准化候选编码的值 q[i][j]。例如，能根据下式标准化这些值：

$$q'[i][j]=\frac{q[i][j]}{\sqrt{\sum_{i,j}\left(q[i][j]-Q_{mean}\right)^2}}, \qquad 0\le i,j<8$$

$$\text{其中,}Q_{mean}=\frac{\sum_{i,j}q[i][j]}{64}$$

在步骤 2465，标准化的值用于根据下列等式计算表示平滑度的块 Laplacian 值 Lg:

$$l[i][j]=\sum_{m=-1}^{1}\sum_{n=-1}^{1}L[m][n]\cdot q'[i+m][j+n], \qquad 1\le i,j<7$$

$$L_q=\frac{\sum_{i,j}|l[i][j]|}{36}$$

块 Laplacian 值越接近 0，图像部分就越平滑。因而能根据块 Laplacian 值测量评分，并且具有最小的 Laplacian 值的候选是正确的。

再考虑其他的变化。在可选择实施例中，更高阶图像表面特性能用作平滑测量。在那些情况中，使用更高阶内核。例如，可以利用第四阶内核执行第四阶块 Laplacian 测量。利用级联的两个第二阶 Laplacian 计算能实现这样的第四阶内核。

还认为评估处理取决于图像是否具有大于预定级别的活动或运动。如果图像部分评估为具有大于预定级的运动，则最好在场的基础上而不在帧的基础上执行测量，这结合图 25 来解释。图 25 解释利用平滑度测量的处理；然而，认为能利用各种类型的测量来实施此处理。

图像区域的帧 2505 由场 0 与场 1 构成。如果未检测到运动，步骤 2510，通过计算每个帧内块的块 Laplacian 值来计算平滑度测量，步骤 2515。如果检测到大于预定级的运动，则对每个场执行块 Laplacian 测量，步骤 2520、2525，并组合这两个测量，步骤 2530，例如，进行平均来生成平滑度测量。

能以各种方法来检测/测量运动。在一个实施例中，评估场之间变化的程度并在此变化程度超过预定门限时检测到运动。

运动检测和帧信息及场信息的使用来生成恢复的值（一般替换丢

失或损坏的值）能应用于要求生成恢复值的处理的任何一个部分。例如，运动检测和帧信息及场信息的选择使用来生成恢复的值能应用于DR/MINF恢复、像素恢复以及Q比特与运动标志恢复处理。因而，根据所检测的运动的级别，此恢复处理将利用场基础上或帧基础上已有的信息。而且，此处理能与特定方向（例如，水平或垂直）上根据相关级别选择的加权值的应用进行组合。

在Q比特与运动标志恢复处理的另一实施例中，根据块内与块之间测量来评估候选解码。在下面的讨论中，术语“块”指帧或场的一部分。块内测量评估候选解码的图像部分，例如，图像部分的平滑度。块之间测量测量候选解码怎样与相邻图像部分相符。图26a与图26b表示组合的块之间与块内评估。特别地，图26a表示可作为块之间与块内测量接受的候选解码是好的，而在图26b中即使块内测量相当好，但块之间测量是差的。

块内测量的示例包括上述的平滑度测量。块之间测量的示例包括上述的平方差错测量。另一可选择的块之间测量是相容的边界像素与候选ADRC块上边界像素总数的比值。

将参见图26c、26d和26e解释进行ADRC编码的8×8块的块之间与块内评估的示例。图26d表示由从中生成候选解码值X的q个值构成的编码值2650与由Y个值构成的相邻解码数据2655的数据的图像部分（块）。如图26C的流程图所示的，在步骤2605，计算块内测量来生成测量，例如块Laplacian L_x。在步骤2610，计算块之间测量Sx，以便生成相邻块之间的相容性的测量。在步骤2615，生成组合的测量Mx，此组合的测量提供用于选择候选解码的信息。

在当前实施例中，S_x计算为位于候选解码（参见图26e）的每个边界像素的有效范围内的相邻数据的数量。图26e是表示一个实施例的有效范围的表，表示每个观察的量化值 q_i 的有效范围。因而，L_Q <DR<U_Q，其中 L_Q、U_Q 分别表示与量化比特＝Q的数量相对应的DR的下与上边界。最好根据下式标准化 S_x：$S_x=S_x$/边界像素的数量

在本实施例中，根据下列等式计算组合的测量：$M_x=S_x+(1-L_x)$。或者，可以加权组合的测量，以致将使用下式：$M_x=W\cdot S_x+(1-w)\cdot(1-L_x)$，其中W是加权值，一般是根据经验确定的加权值，该值例如可以为0和1之间的值。

再考虑用于确定已丢失/损坏的 DR 与 MIN 值的其他实施例。例如，上述等式能修改为利用较高的准确性来恢复 DR 与 MIN 值。在另一可选择实施例中，应用中值技术。在中值技术的一个实施例中，MIN 的值恢复为如下计算的所有 MINi 值的中值:

$$MIN_i = Y_i - q_i \cdot s$$

其中 q_i 表示编码的像素值，而 Y_i 表示与 q_i 相邻的解码像素。对于边缘匹配 ADRC，$s = DR/(2^Q-1)$。对于非边缘匹配 ADRC，$s = DR/2^Q$，其中 Q 表示每个像素的量化比特的数量(Q 比特值)。

所使用的值可以是时间上近似或空间上近似。Y_i 的值可以是相邻帧/场或同一场中相邻像素的解码的值。Y_i 的值可以是来自与相邻帧/场中或同一场中的 q_i 相同的位置的像素的解码值。

另外，任意一个 DR 和/或 MIN 恢复技术可以与剪切处理进行组合来提高恢复准确度并在恢复处理期间阻止数据溢出。此剪切处理将恢复的数据限制到值的预定范围；因而，在此范围之外的值剪切到最接近的范围界限。在一个实施例中，剪切处理将值限制在 $[L_Q, U_Q]$ 的范围内，其中 L_Q、U_Q 分别表示量化比特 Q 的数量表示的像素值范围的下与上边界。该值还可限制为：$MIN + DR \leq Num$，其中 Num 表示最大的像素值；在本实施例中，Num 是 255。在本实施例中，可应用 $U_Q+1=L_Q+1$。

将此准则组合在单个等式中得到 DR 的无界的恢复值（val'），从下式中获得最后剪切的恢复值（val）:

$$Val = \max(\min(val, \min(U_Q, 255 - MIN)), L_Q)$$

其中 min 与 max 分别表示最小与最大函数。

在另一可选择实施例中，能过滤用于生成恢复的 DR 和/或 MIN 的边界像素 y_i 为只使用显示最佳相关的那些边界像素，从而更好地恢复 DR 与 MIN。不使用不满足此准则的那些边界像素。在一个实施例中，如果存在使 $L_Q \leq DR < U_Q$ 的 DR 的值并且原始像素 y_i 已编码为 q_i，则认为边界像素 y_i 对于 DR 计算是有效的。因而，如果满足下列等式，则像素是有效的:

$$\frac{(y_i - MIN)m}{\max(q_i - 0.5, 0)} \geq L_Q$$

$$\frac{(y_i - MIN)m}{\min(q_i - 0.5, m)} < U_Q$$

其中 m 表示最大的量化级 = $2^Q - 1$。然后能根据下式计算 DR 恢复值（val'）：

$$val' = \frac{m \cdot \sum_i (y_i - MIN)q_i}{\sum_i q_i^2}$$

随后能将此值剪切到有效范围内。因而，此处理强迫 DR 恢复值到利用门限表定义的有效区域内，减少其真的 DR 位于门限表边界附近的点的准确性。

已经说明：由于量化噪音，静止 ADRC 块的 DR 在帧与帧之间轻微地变化。如果此变化经过 ADRC 编码边界，和如果 DR 在几个连续帧上进行恢复，那么具有有效像素选择的 DR 恢复值在每个交叉点趋于过冲，导致显示时明显的闪烁效果。在为减少此效果出现的尝试中，在一个实施例中，修改有效像素选择处理以便松弛（relax）上与下边界，允许边界像素侵入相邻有效区域。通过包括正好在边界外部的点，恢复值将更有可能取靠近上或下边界的值。利用松弛常数 r 计算松弛边界 L'_Q 与 U'_Q。在一个实施例中，r 设置为 5，能使用其他值：

$L'_Q = r\ L_{Q-1} + (1 - r)L_Q$

$U'_Q = (1-r)U_Q + rU_{Q+1}$

上述讨论提出在 DR 与 MIN 值被损坏或丢失时恢复这些值的许多方法。通过检查时间和/或空间上数据之间的相关性并因此根据计算的恢复值进行加权能实现进一步的增强。更特别地，如果在特定方向或时间上具有大的相关性，例如水平相关性，则很有可能在具有大相关性的那个方向上图像特性继续平滑，并因此使用高相关数据恢复的值一般生成较好的估算。为利用此，边界数据分散在相应的方向（例如，垂直、水平、场-场）中并根据相关测量加权来生成最终恢复的值。

参考图 27a 描述此处理的一个实施例。在步骤 2710，在一个方向

上生成将要恢复的 DR 或 MIN 值的恢复值，并在步骤 2715，在另一方向上生成恢复的值。例如，如果此处理在空间上是自适应的，那么沿水平边界的边界像素用于生成第一恢复值，即“hest”，而沿垂直边界的边界像素用于生成第二恢复值，即“vest”。或者，如果此处理在时间上是自适应的，则相邻场之间的边界像素用于生成第一恢复值，而相邻帧之间的边界像素用于生成第二恢复值。

在步骤 2720，根据表示每个方向上相关级的相关计算加权所恢复的值。组合加权的第一与第二恢复值，以生成组合的恢复值，步骤 2725。应注意：此处理不限于只在两个方向上生成加权的恢复值；很显然：加权与组合的恢复值的数量能根据应用而改变。各种公知技术可以用来生成表示特定方向上相关级的相关值。而且，各种准则可以用来根据相关级选择加权因子。一般地，如果一个相关性大于另一个，则组合的恢复值应主要基于相应的恢复值。在一个实施例中，组合的恢复值如下进行计算：

$$val' = \begin{cases} \alpha hest + (1-\alpha) vest & : \quad hc \geq vc \\ (1-\alpha) hest + \alpha vest & : \quad hc < vc \end{cases}$$

其中 hc 表示水平相关性，vc 表示垂直相关性，hest 表示只基于左与右边界信息的 DR 恢复值，和 vest 表示只基于上与下边界信息的 DR 恢复值，而α表示加权值，此加权值能以许多方法来确定。图 27b 表示用于确定为水平相关性与垂直相关性之间差函数的加权值的一个实施例。更特别地，α选择为：

$$\alpha(|hc - vc|) = \begin{cases} 0.5 + 0.25 \cdot e^{-8(0.35 - |hc - vc|)} & : \quad |hc - vc| < 0.35 \\ 1 - 0.25 \cdot e^{-8(|hc - vc| - 0.35)} & : \quad |hc - vc| \geq 0.35 \end{cases}$$

如上所述，自适应相关处理可应用于 DR 与 MIN 恢复。然而，最好剪切 MIN 恢复以保证 MIN + DR ≤ 255，因此能使用函数 val=max(min(val′,255-MIN),0)。而且，如上所述，时间相关性能进行评估并用于加权恢复的值。另外，能执行时间与空间相关性的组合。例如，在场之间生成一个恢复值作为时间恢复值。在一个场内生成另一个恢复值作为空间恢复值。最终恢复的值计算为具有时间与空间相关性组

合的组合值。可以利用运动数量代替相关性组合。也考虑其他的不同情况。此方法也能应用于音频数据。

在另一可选择实施例中，使用低复杂性修改为最小平方技术。利用此实施例，减少由于恢复的 DR 值而经历的闪烁。为了下面的讨论，QV 表示来自其 DR 正在进行恢复的具有一组点 q_i 的图像部分或 ADRC 块的编码值的表，并且 Y 是从 QV 中的垂直或水平相邻点中取出的解码值的表，其中 Y_i 表示 q_i 垂直或水平相邻点。在每个点 q_i 上可以具有多达 4 个的解码的相邻点，一个像素或点可以增加到四（q_i, y_i）对。DR（DR_{uls}）的无限的最少平方估算因而为：

$$(DR)_{uls}=\frac{2^{Q}\cdot\sum_{i}(y_i-MIN)\cdot(0.5+q_i)}{\sum_{i}(0.5+q_i)^2}$$

其中 Q 是量化比特的数量，MIN 是作为块属性发送的最小值。上式假定非边缘匹配 ADRC；对于边缘匹配 ADRC 而言，利用 2^Q-1 代替 2^Q 并利用 q_i 代替 $(0.5+q_i)$。

无限的最少平方估算最好进行剪切，以保证与门限表和在编码期间（一般地，对于非边缘匹配 ADRC，允许的 DR 值在 1－256 的范围内）执行的等式 MIN＋DR≤ 255 一致。因而，通过下式剪切此最少平方估算（DR_{1sc}）：

$$(DR)_{1sc}=\max(\min(UB, DR_{uls}), LB)$$

其中 UB 表示上边界，而 LB 表示下边界，并且 min 与 max 分别表示最小与最大函数。

在另一可选择实施例中，通过选择更适于 DR 估算的像素来计算 DR 的估算值能改善估算。例如，图像中平坦区域提供比其中高活动性出现的那些区域更适于 DR 估算的像素。特别地，边缘中的锐边缘可能减少估算的准确性。下面的实施例提供用于选择用来计算 DR 的估算值的像素的计算量少的方法。

在一个实施例中，计算最少平方估算（DR_{1sc}），例如，DRu1s 或 DR1sc。利用此估算，编码值 QV 的表变换为候选解码值 X，其中 X_i 是从 q_i 中导出的 X 的成员。此 X_i 值是利用 DR 的第一估算值形成的恢复的解码值。根据下式定义此 X_i 值：

边缘匹配 ADRC: $x_i = MIN + \left(0.5 + \frac{q_i \cdot DR_{lsc}}{2^Q - 1}\right)$

非边缘匹配 ADRC: $x_i = MIN + \left(\frac{(q_i + 0.5) \cdot DR_{lsc}}{2^Q}\right)$

假设 DR1sc 是真的 DR 的合理估算值，则 X_i 相对靠近 y_i 的地方可以判断为低活动区域并因而是希望的匹配。新的 X 与 Y 表随后可以通过仅考虑其中 X_i 与 Y_i 接近的匹配来形成，并重新计算最少平方估算来生成更新的估算值。

能以许多方法来确定用于确定认为什么是“接近（close）”的准则。在一个实施例中，使用差错函数的 ADRC 编码。此方法由于计算上便宜而是希望的。对于此处理而言，定义由点 $e_i=|y_i-x_i|$ 构成的表 E。分别定义 emin 与 emax 为此表中的最小与最大值，则 eDR=emax-emin。编码的差错值则能定义为：

$g_i=(e_i-emin)\ n1/eDR$

其中 n1 表示用于以类似于上述 ADRC 处理的方法重新量化 e_i 的量化级的数量。

因而，通过仅选择其中 g_i 小于某一门限的匹配来生成新表 X 与 Y。如果新表足够长，则这些表可以用来生成精细的最小平方估算 DRr1s。最好根据经验确定在精细化最少平方估算之前所需的 g_i 的门限和匹配的数量。例如，在一个实施例中，对于涉及 8×8×2 水平二次取样块并且 n1 为 10 的处理，只使用与 $g_i=0$ 相对应的匹配，并且只在这些新表至少包含 30 种匹配时才精细化此估算。

在另一可选择实施例中，通过剪切可能的 DR 值和重新计算 DR 估算值能改善 DR 估算。特别地，在一个实施例中，表 D 由包含将使 x_i 等于 y_i 的 DR 值的成员 d_i 组成。更精确地：

$d_i=2^Q(y_i-MIN)/(0.5+q_i)$

通过剪切每个 d_i 发现改善。即：

$d_i'=max(min(UB, d_i), LB)$

其中 DRc1s 随后计算为 d_i' 的平均值。剪切的方法（DRc1s）可以与其他的 DR 估算值进行组合，例如，与加权的平均值中的 DR1st 组合来生成最终的 DR 值。例如，根据下式确定加权的平均值 DRest：

$$DRest=W_1(DRc1s)+W_2(DR1se)$$

最好通过检查得到的估算值和从特定加权中生成的图像根据经验确定权值 W_1 与 W_2。在一个实施例中，$W_1 = 0.22513$，而 $W_2 = 0.80739$。

已结合优选实施例解释本发明。很显然根据前文的描述对于本领域技术人员来说许多替换、修改、变化和使用将是显而易见的。

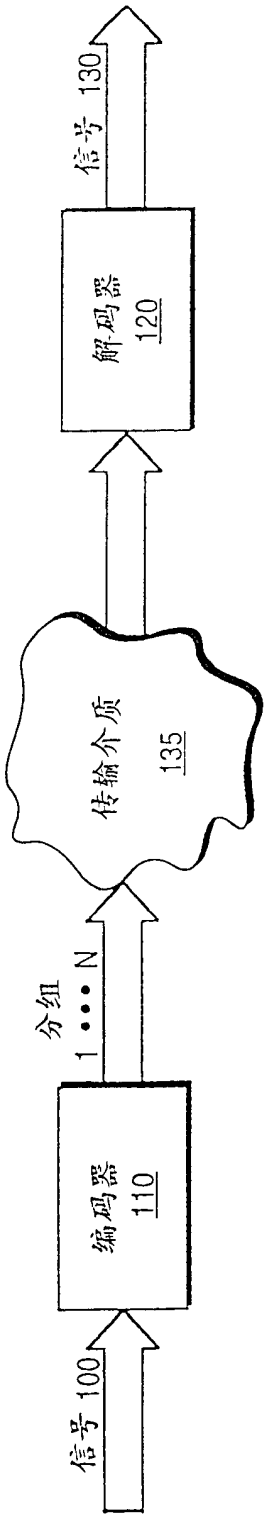


图 1

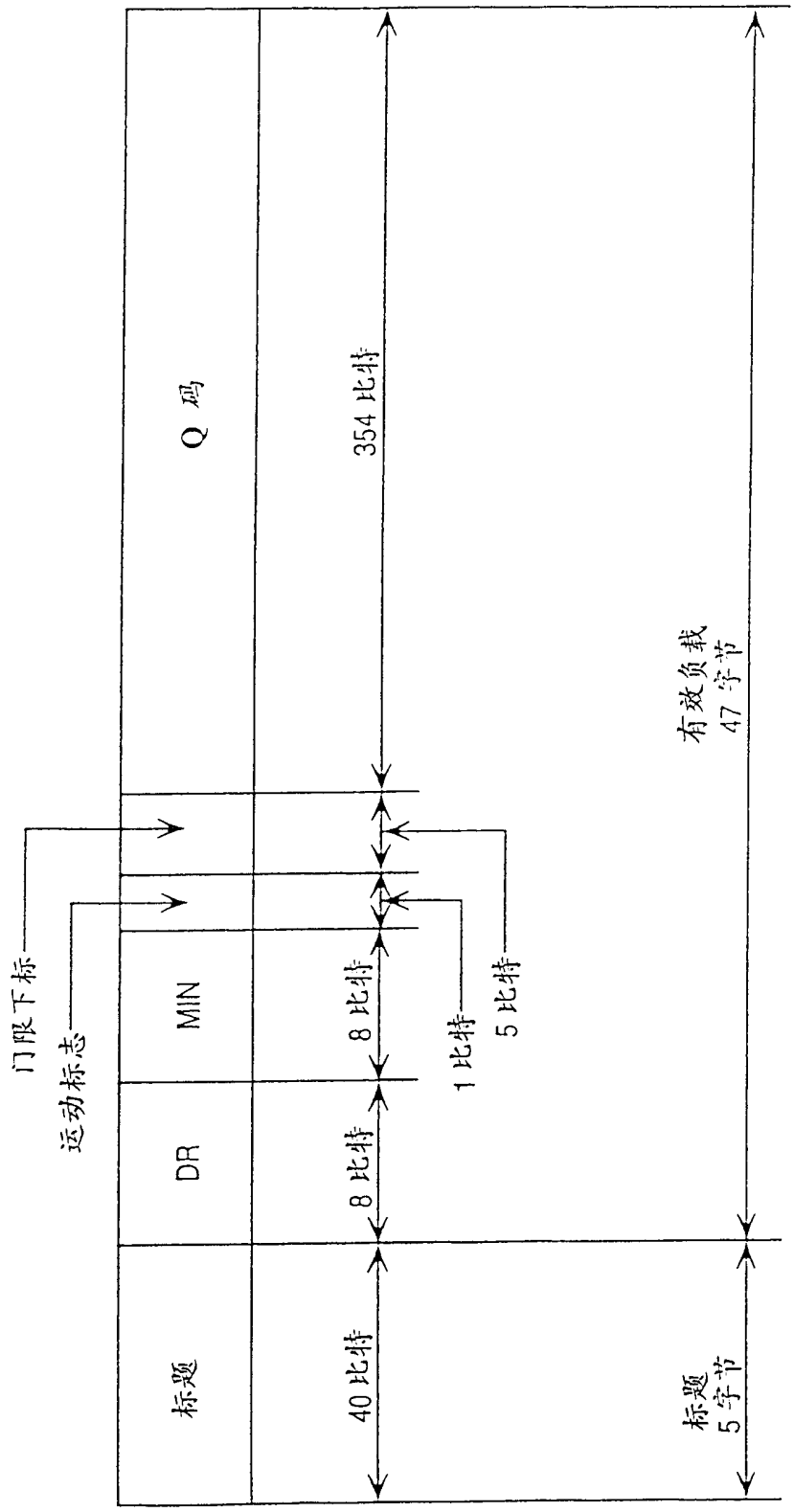
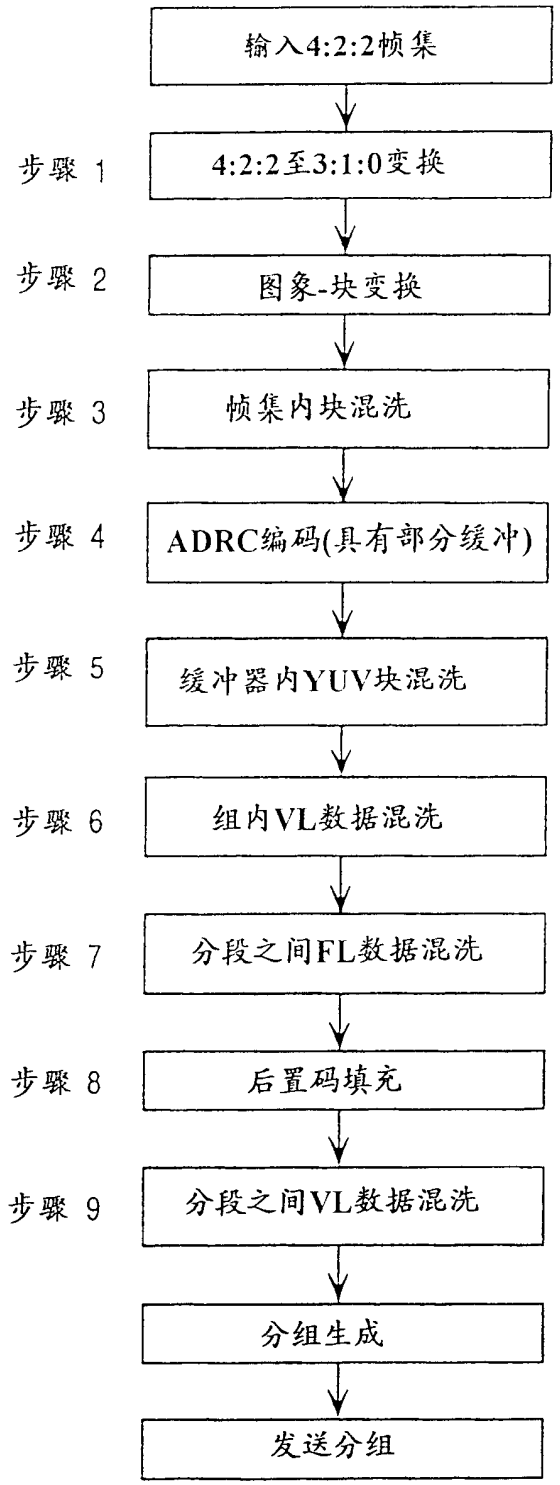

标题
DR
MIN
Q 码
门限下标
运动标志
40 比特
8 比特
8 比特
1 比特
5 比特
354 比特
标题
5 字节
有效负载
47 字节
分组结构 200

图 2



编码器中的
信号流程图

图 3

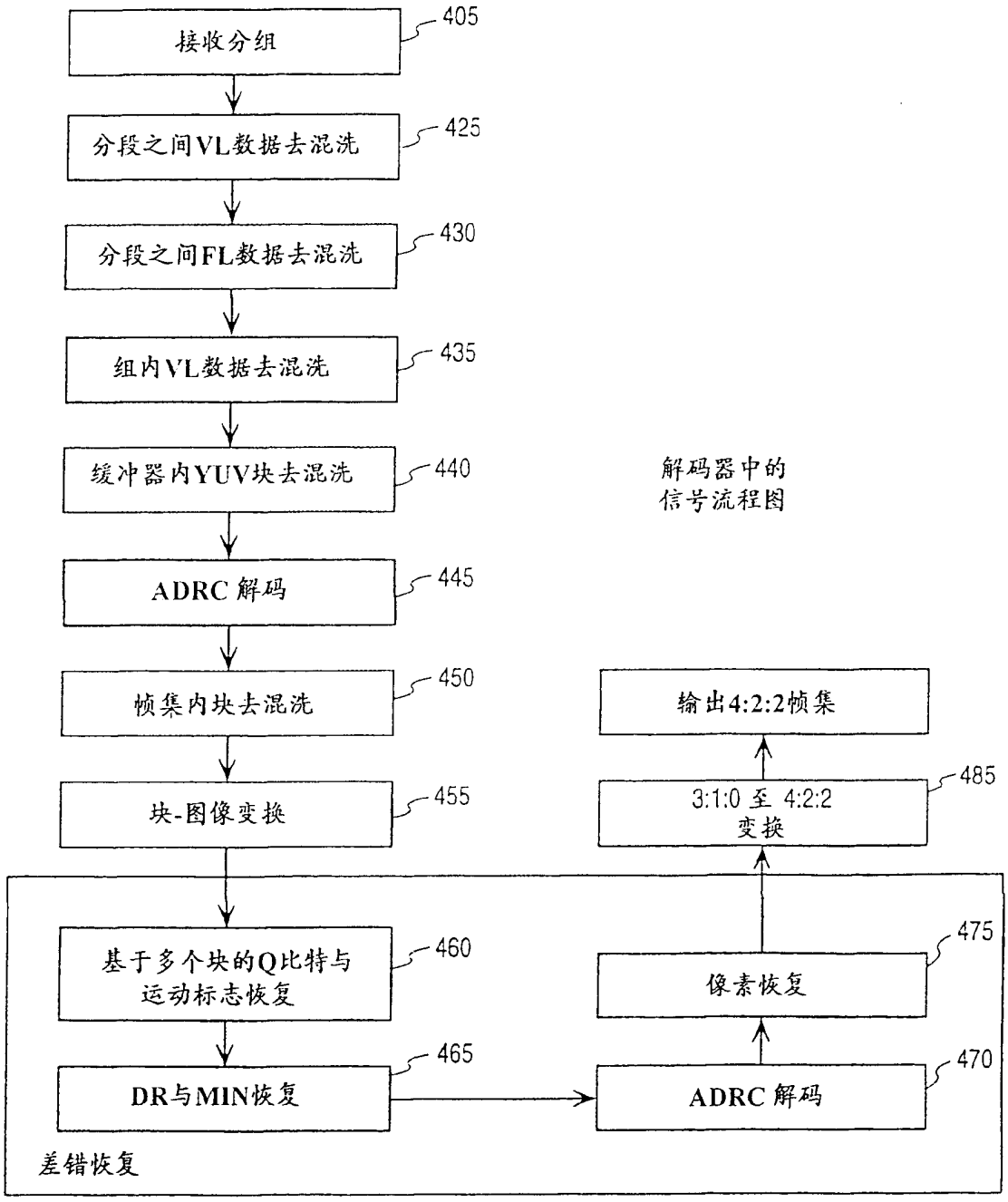

接收分组
405
分段之间VL数据去混洗
425
分段之间FL数据去混洗
430
组内VL数据去混洗
435
缓冲器内YUV块去混洗
440
ADRC 解码
445
帧集内块去混洗
450
块-图像变换
455
基于多个块的Q比特与运动标志恢复
460
DR与MIN恢复
465
差错恢复
解码器中的信号流程图
输出4:2:2帧集
3:1:0 至 4:2:2 变换
485
像素恢复
475
ADRC 解码
470

图 4

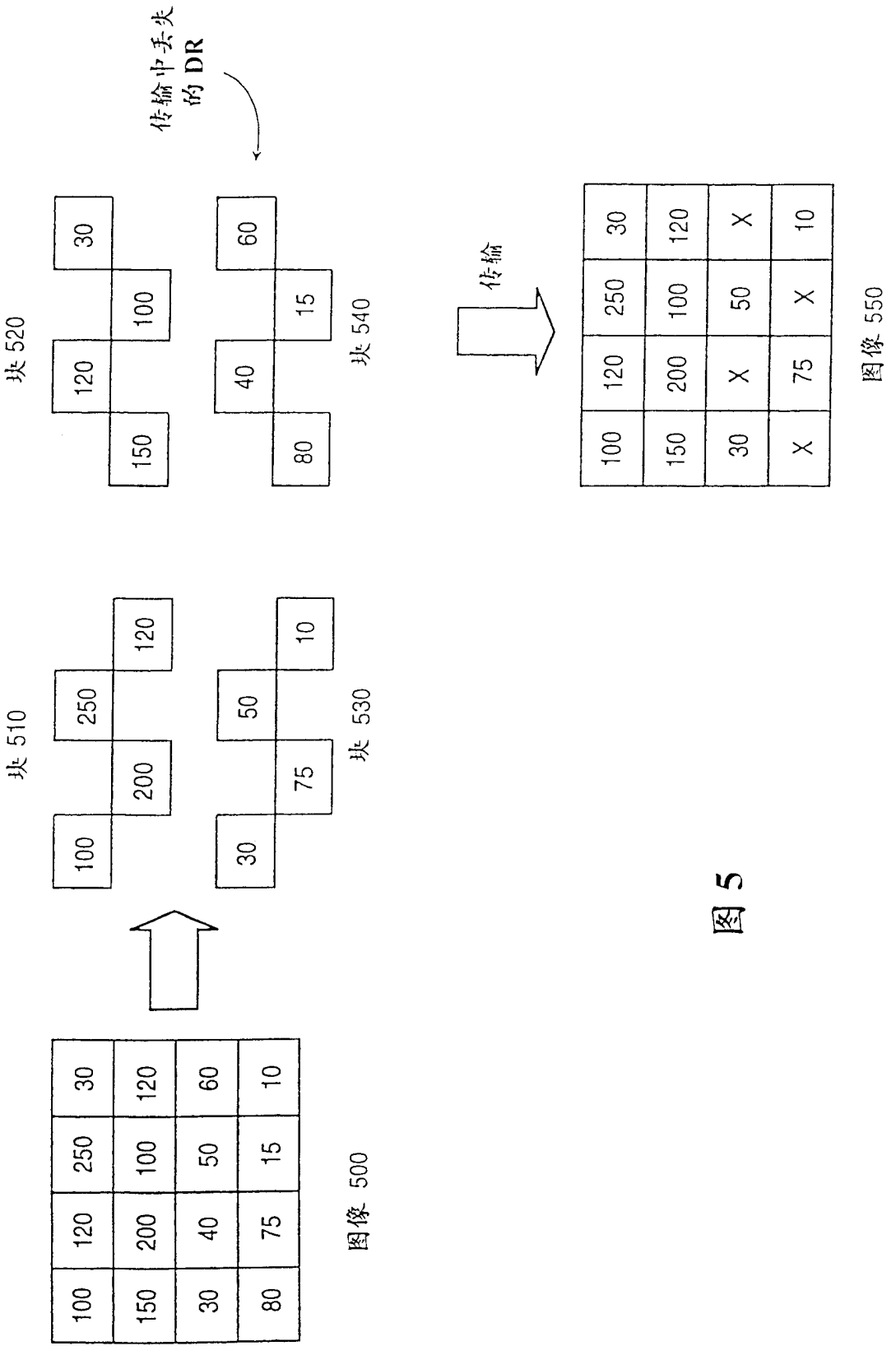
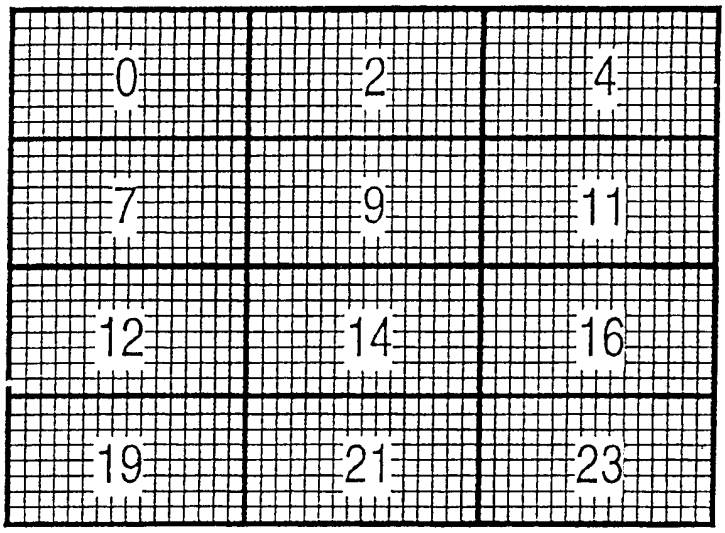
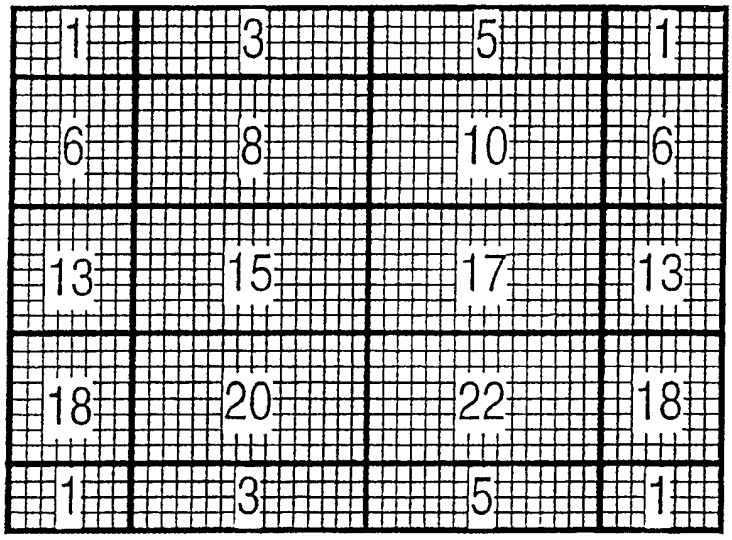


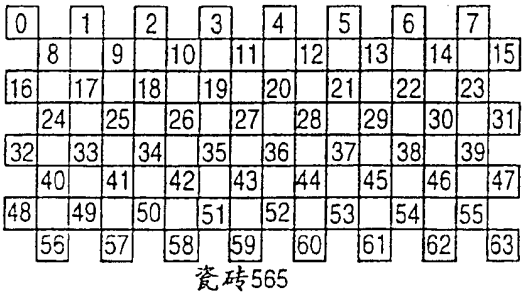
图 5



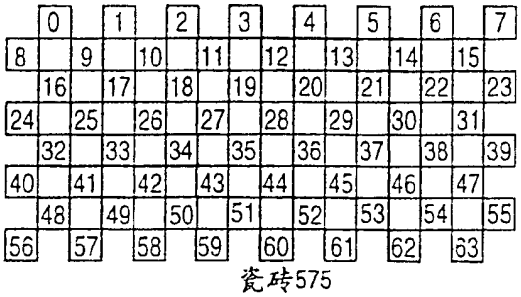
图像 560



图像 570



瓷砖565



瓷砖575

图 5a

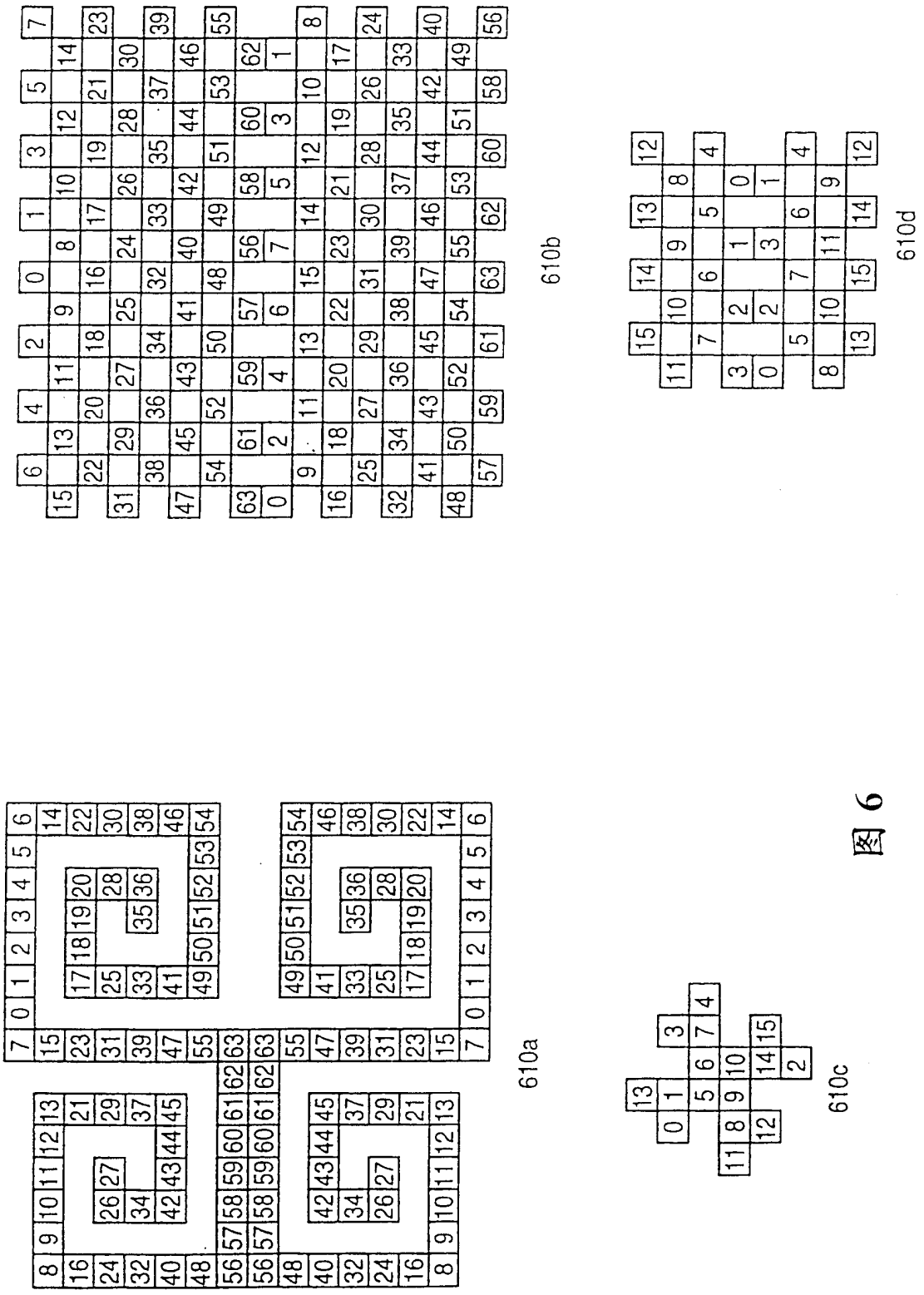


图 6

帧集内块混洗
分段定义:Y-块

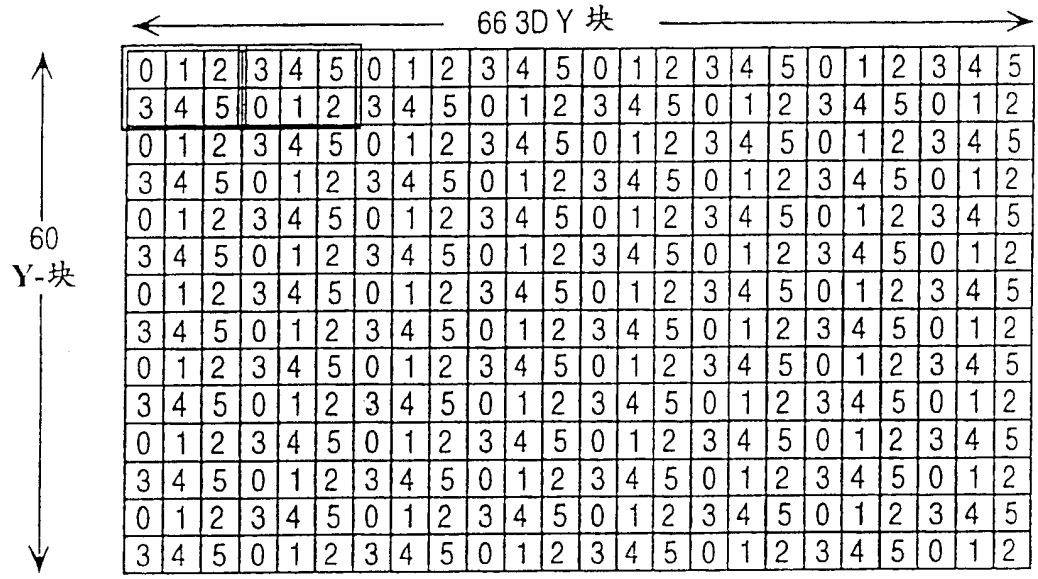


图 7a

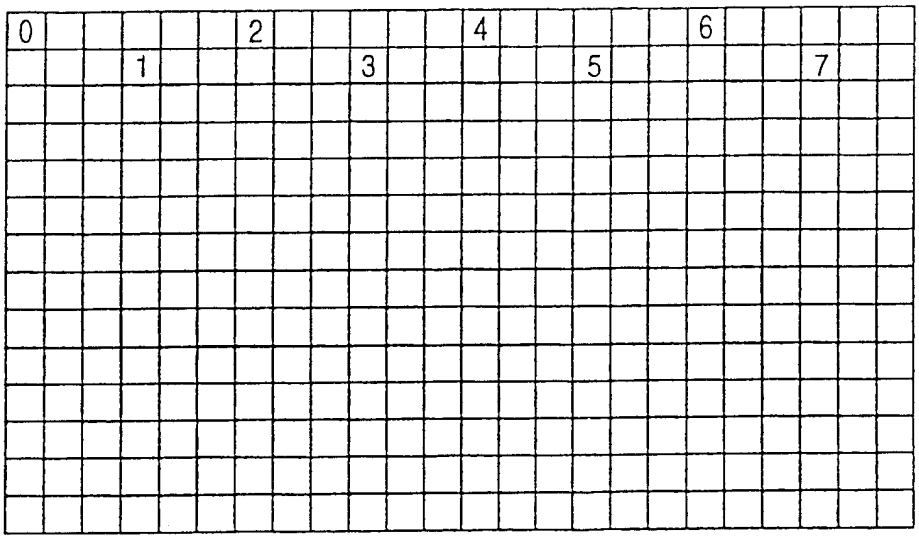


图 7b

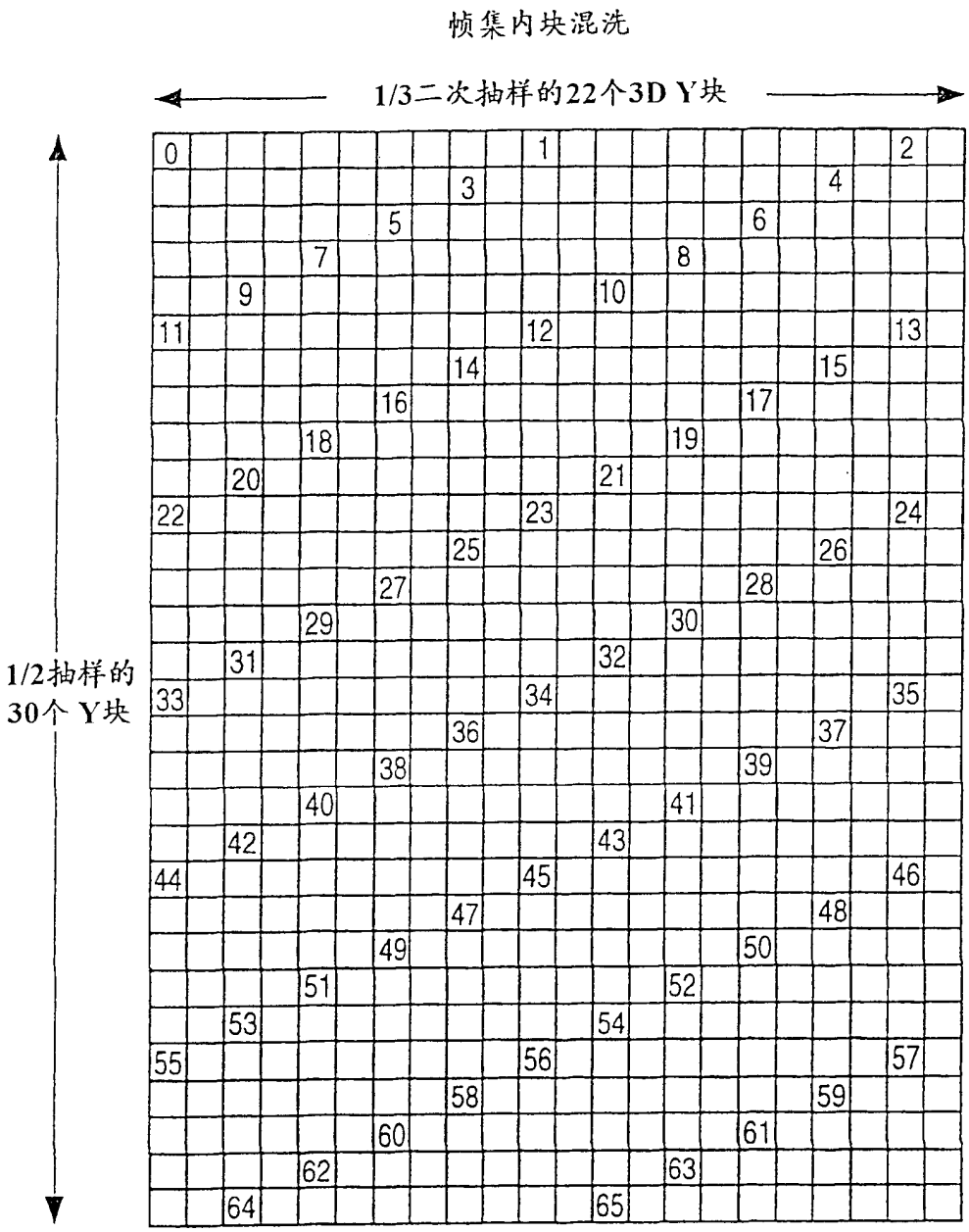


图 7c

帧集内块混洗

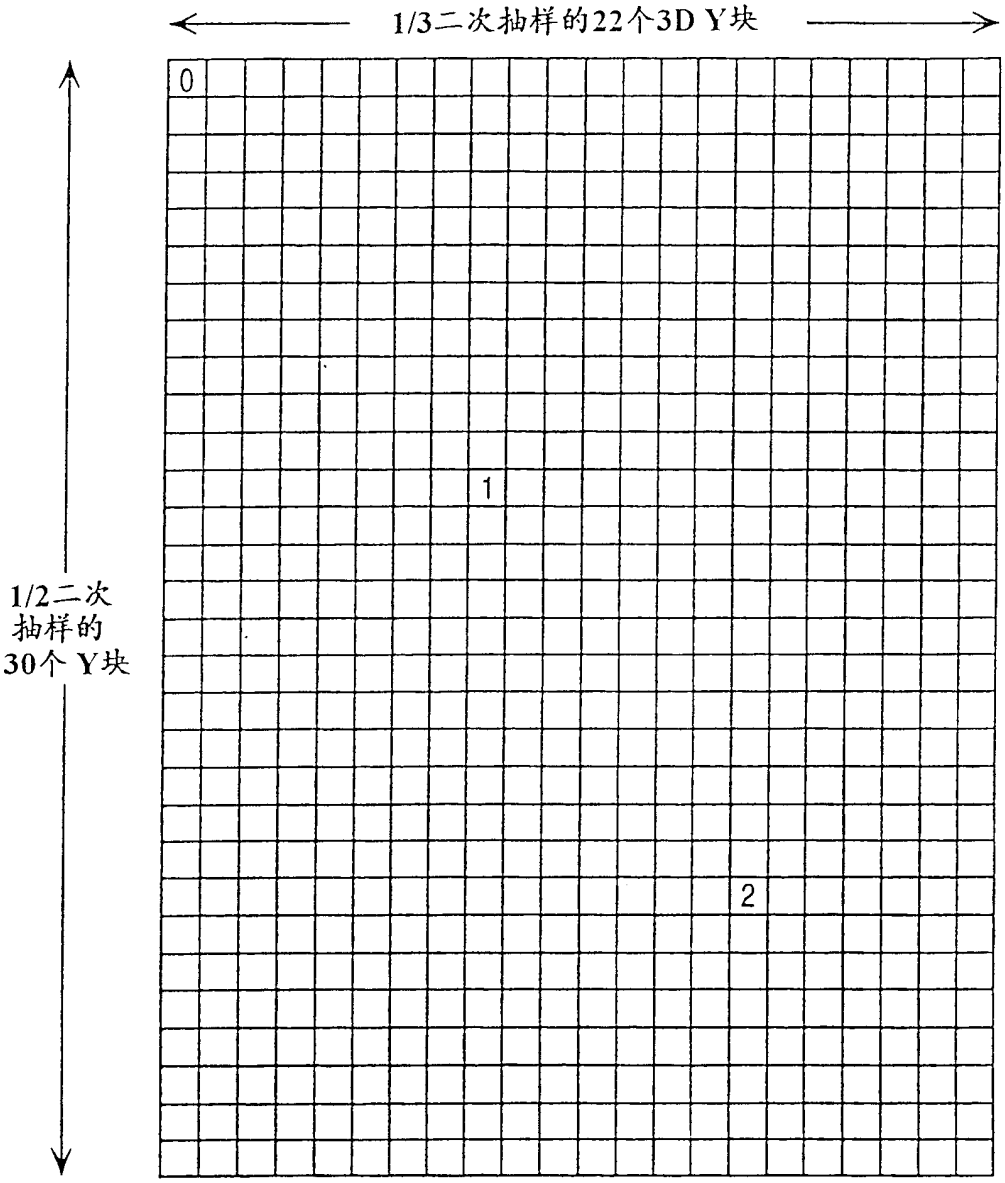


图 7d

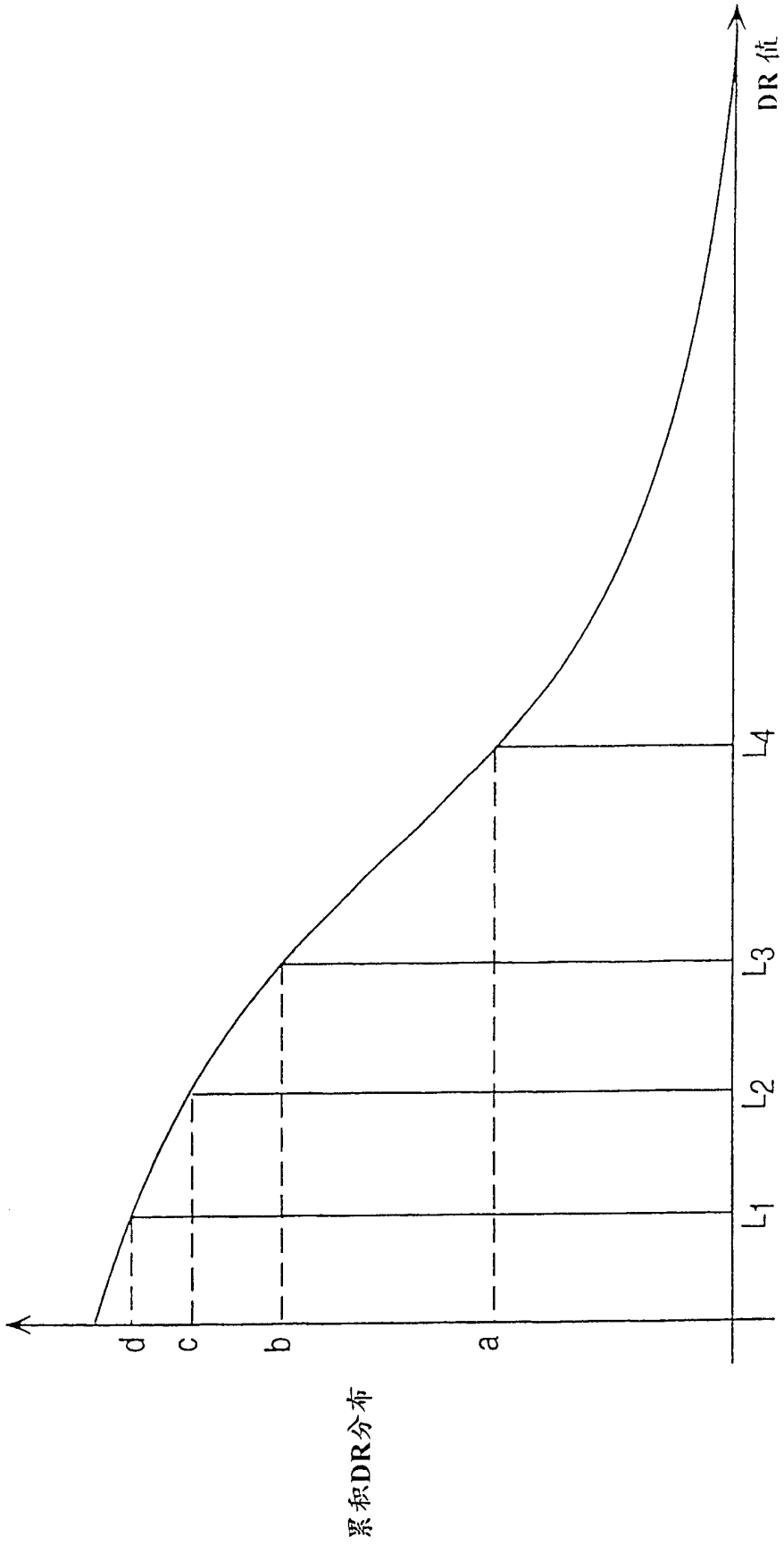


图 8

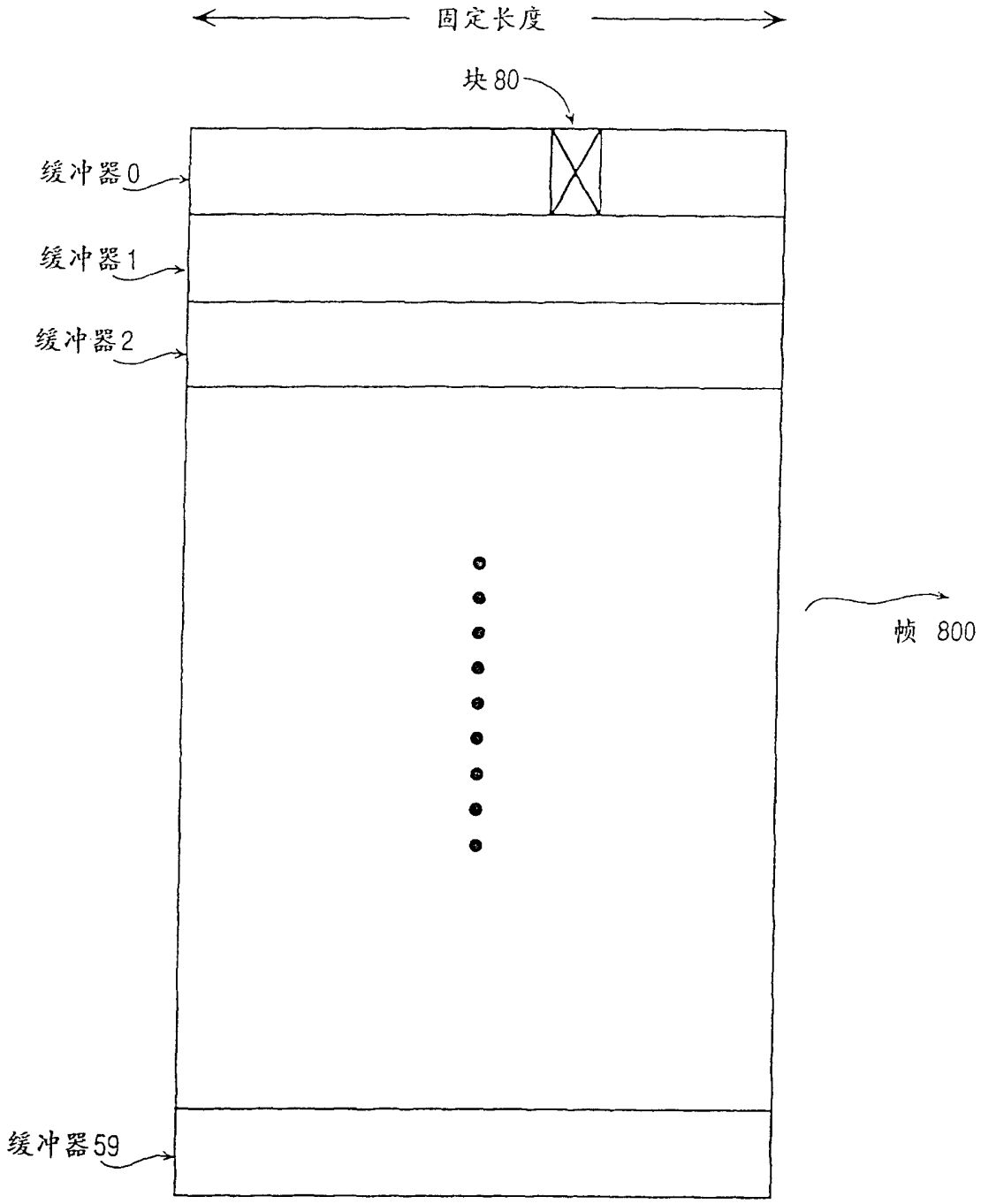


图 8a

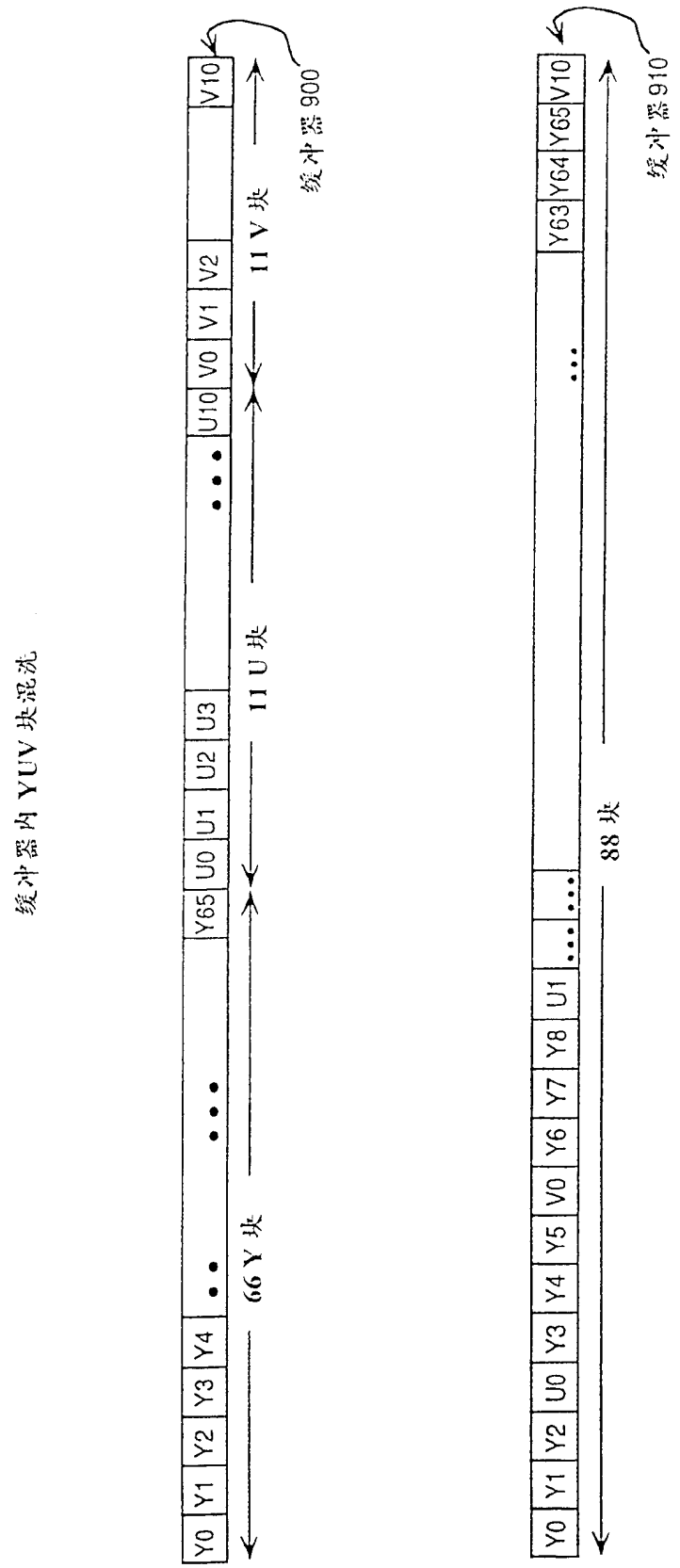

缓冲器内 YUV 块混洗
Y0 Y1 Y2 Y3 Y4 ... Y65 U0 U1 U2 U3 ... U10 V0 V1 V2 ... V10
66 Y 块
11 U 块
11 V 块
缓冲器 900
Y0 Y1 Y2 U0 Y3 Y4 Y5 V0 Y6 Y7 Y8 U1 ... ... Y63 Y64 Y65 V10
88 块
缓冲器 910

图 9

组内 VL 数据混洗

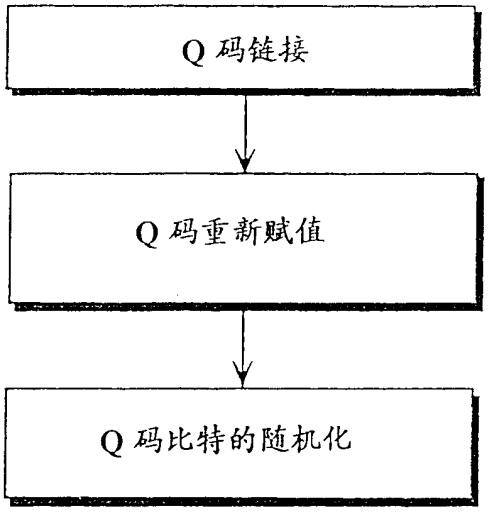


图 10

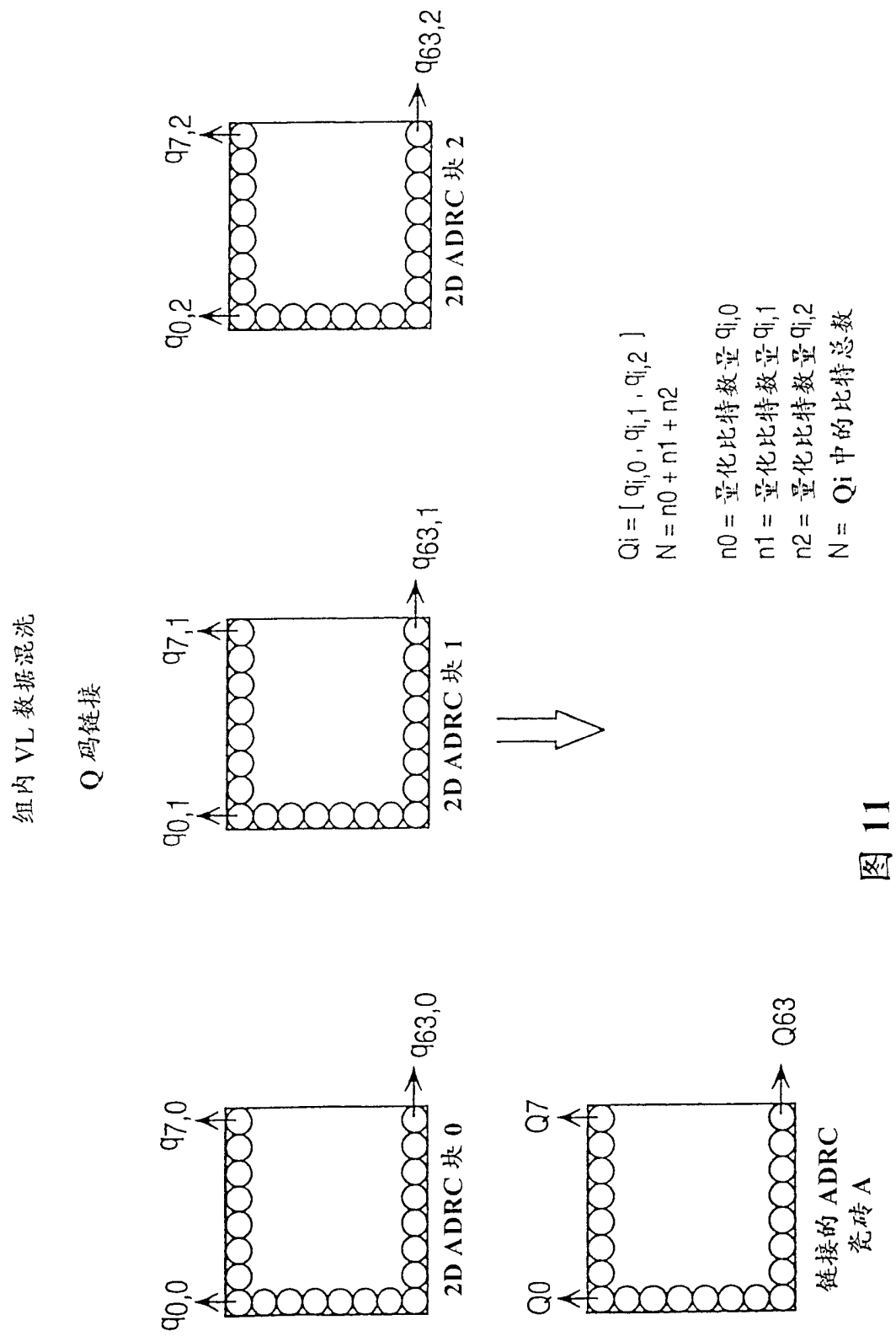

组内 VL 数据混洗
Q 码链接
q0,0
q7,0
q63,0
2D ADRC 块 0
q0,1
q7,1
q63,1
2D ADRC 块 1
q0,2
q7,2
q63,2
2D ADRC 块 2
Qi = [qi,0 , qi,1 , qi,2]
N = n0 + n1 + n2
n0 = 量化比特数量 qi,0
n1 = 量化比特数量 qi,1
n2 = 量化比特数量 qi,2
N = Qi 中的比特总数
Q0
Q7
Q63
链接的 ADRC
类块 A

图 11

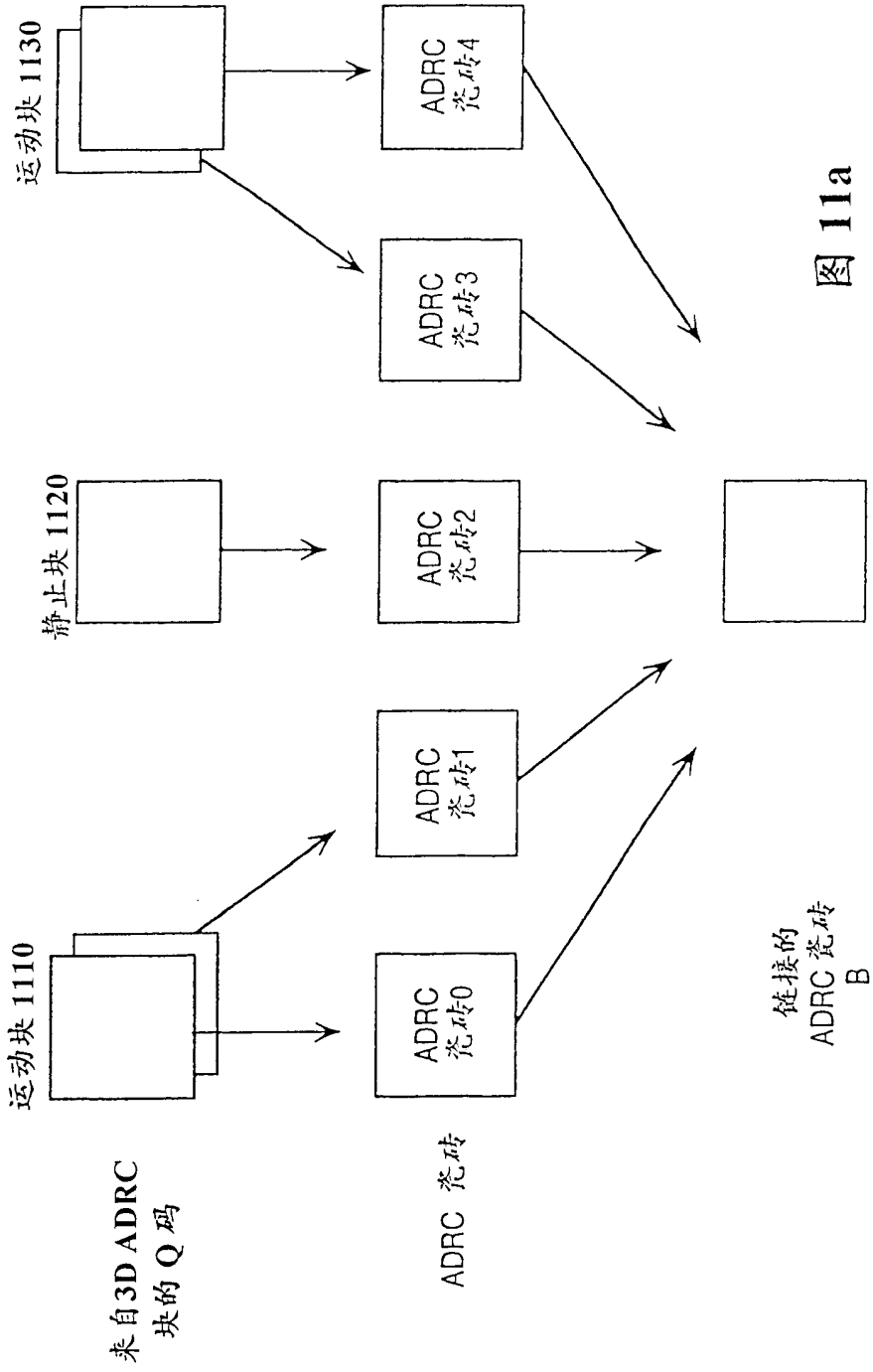
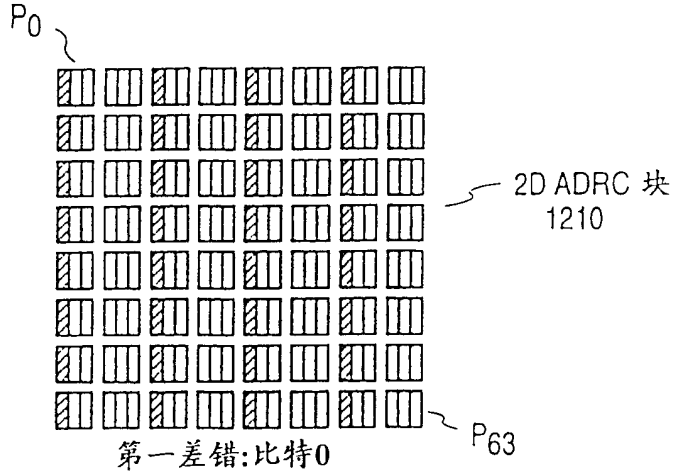
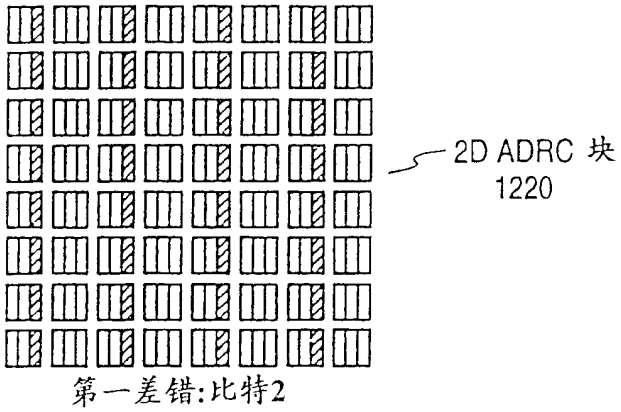

运动块 1130
静止块 1120
运动块 1110
ADRC 类砖4
ADRC 类砖3
ADRC 类砖2
ADRC 类砖1
ADRC 类砖0
来自3D ADRC 块的Q码
ADRC 类砖
链接的 ADRC 类砖 B

图 11a


P0
2D ADRC 块
1210
P63
第一差错:比特0


2D ADRC 块
1220
第一差错:比特2

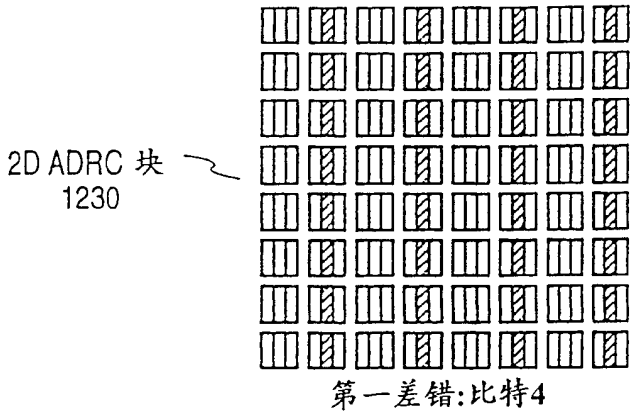

2D ADRC 块
1230
第一差错:比特4

图 12

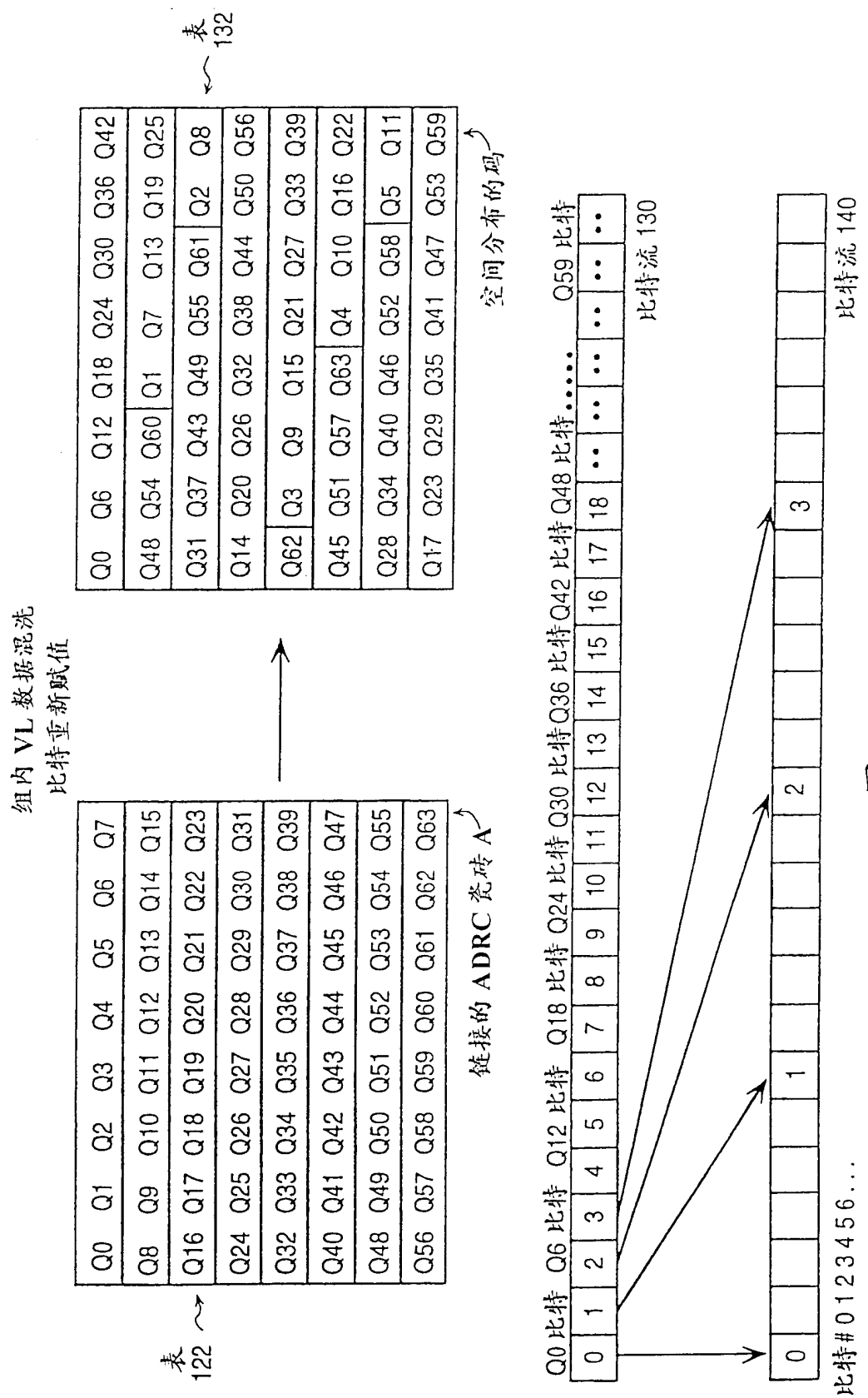


表 122
Q0 Q1 Q2 Q3 Q4 Q5 Q6 Q7
Q8 Q9 Q10 Q11 Q12 Q13 Q14 Q15
Q16 Q17 Q18 Q19 Q20 Q21 Q22 Q23
Q24 Q25 Q26 Q27 Q28 Q29 Q30 Q31
Q32 Q33 Q34 Q35 Q36 Q37 Q38 Q39
Q40 Q41 Q42 Q43 Q44 Q45 Q46 Q47
Q48 Q49 Q50 Q51 Q52 Q53 Q54 Q55
Q56 Q57 Q58 Q59 Q60 Q61 Q62 Q63
组内 VL 数据混洗
比特重新赋值
链接的 ADRC 类别 A
表 132
Q0 Q6 Q12 Q18 Q24 Q30 Q36 Q42
Q48 Q54 Q60 Q1 Q7 Q13 Q19 Q25
Q31 Q37 Q43 Q49 Q55 Q61 Q2 Q8
Q14 Q20 Q26 Q32 Q38 Q44 Q50 Q56
Q62 Q3 Q9 Q15 Q21 Q27 Q33 Q39
Q45 Q51 Q57 Q63 Q4 Q10 Q16 Q22
Q28 Q34 Q40 Q46 Q52 Q58 Q5 Q11
Q17 Q23 Q29 Q35 Q41 Q47 Q53 Q59
空间分布的码
Q0 比特 Q6 比特 Q12 比特 Q18 比特 Q24 比特 Q30 比特 Q36 比特 Q42 比特 Q48 比特..... Q59 比特
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
比特流 130
比特# 0 1 2 3 4 5 6 ...
0 1 2 3
比特流 140

图 12a

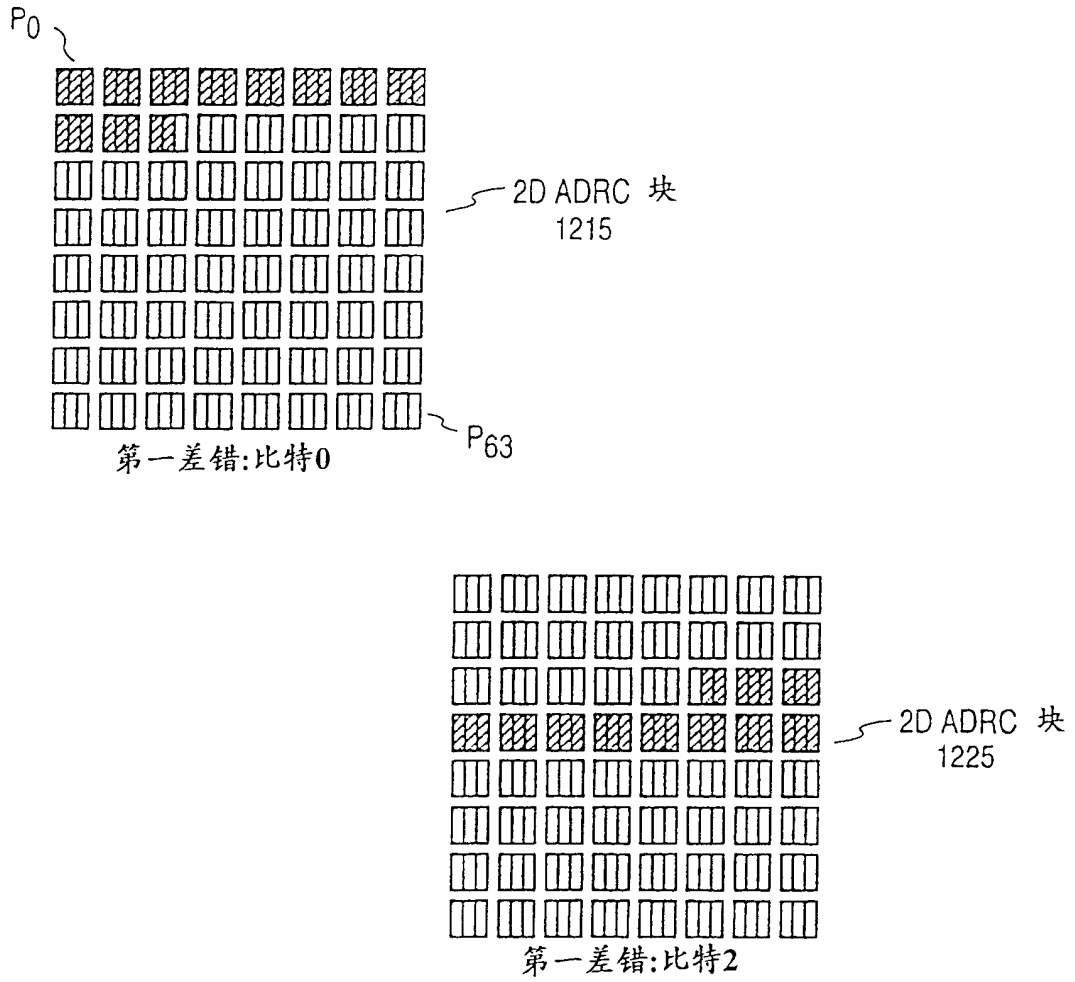
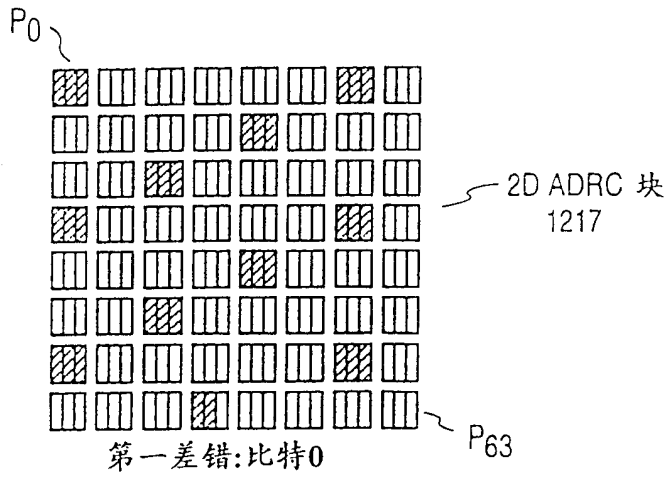
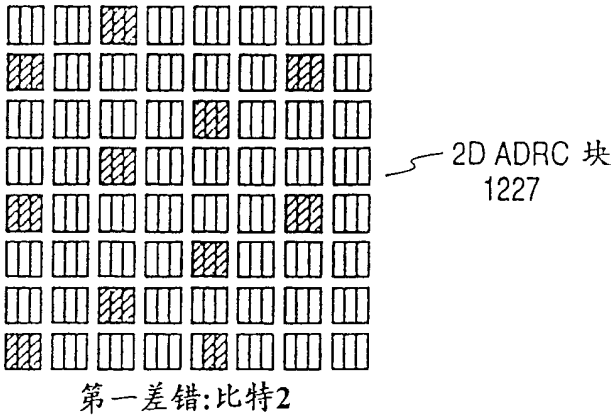

P0
2D ADRC 块
1215
P63
第一差错:比特0
2D ADRC 块
1225
第一差错:比特2
2D ADRC 块
1235
第一差错:比特4

图 12b


P0
2D ADRC 块
1217
P63
第一差错:比特0


2D ADRC 块
1227
第一差错:比特2

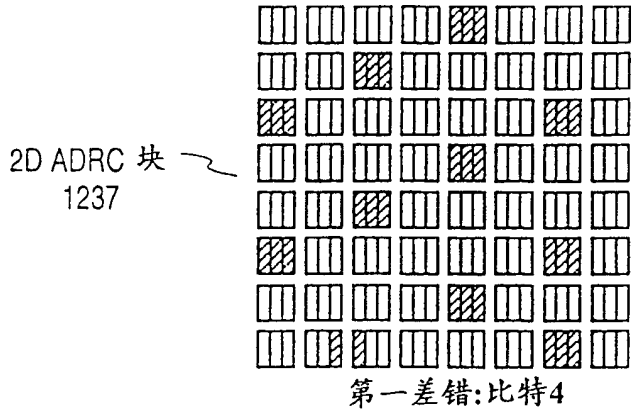
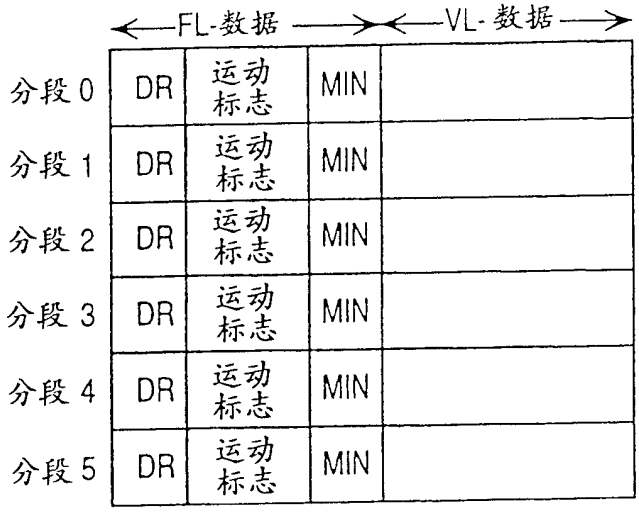

2D ADRC 块
1237
第一差错:比特4

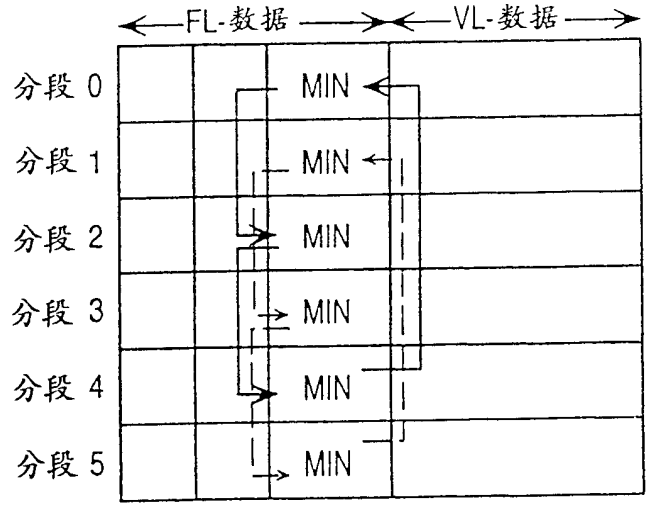
图 12c

分段之间 FL 数据混洗

原始



MIN 混洗 1300

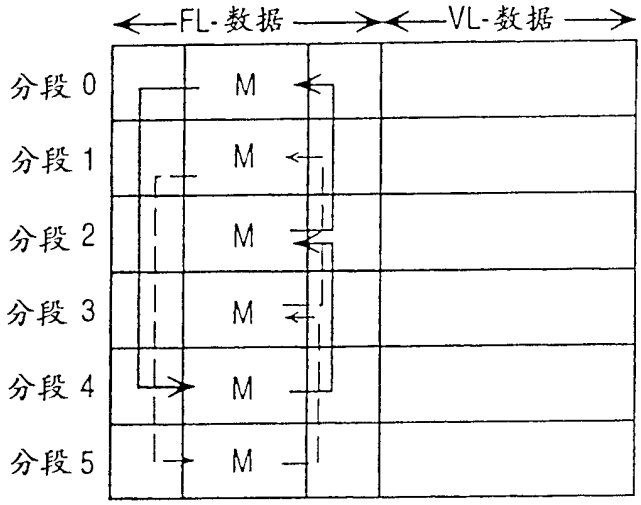


原始 混洗

分段 0 --> 分段 2
分段 2 --> 分段 4
分段 4 --> 分段 0
分段 1 --> 分段 3
分段 3 --> 分段 5
分段 5 --> 分段 1

图 13

运动标志混洗



原始 混洗
分段 0 --> 分段 4
分段 2 --> 分段 0
分段 4 --> 分段 2
分段 1 --> 分段 5
分段 3 --> 分段 1
分段 5 --> 分段 3

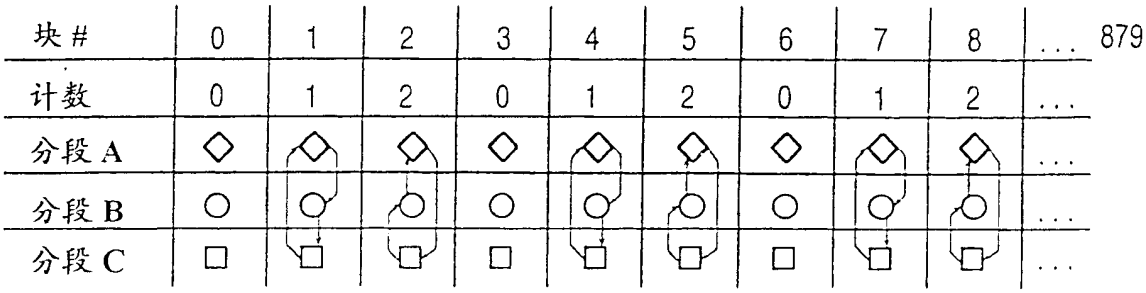
FL 数据混洗之后
分段0的FL数据损耗模式

分段 \ 块 #	0	1	2	3	4	5	• • • 879
0	DR	DR	DR	DR	DR	DR	• • •
1							• • •
2	M	M	M	M	M	M	• • •
3							• • •
4	MIN	MIN	MIN	MIN	MIN	MIN	• • •
5							

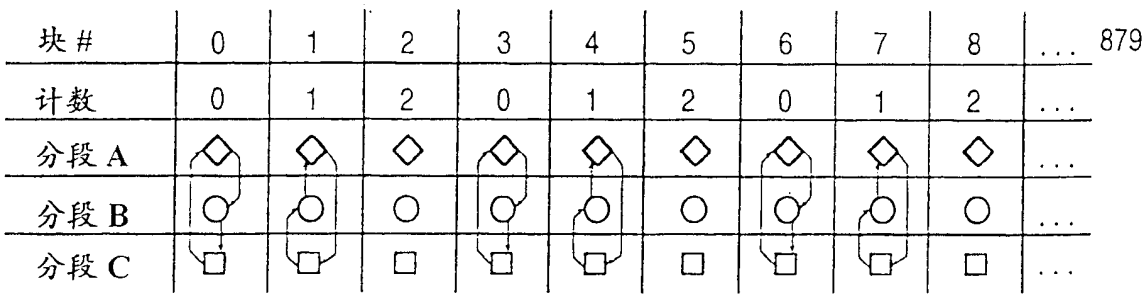
损耗模式 1310

M:运动标志

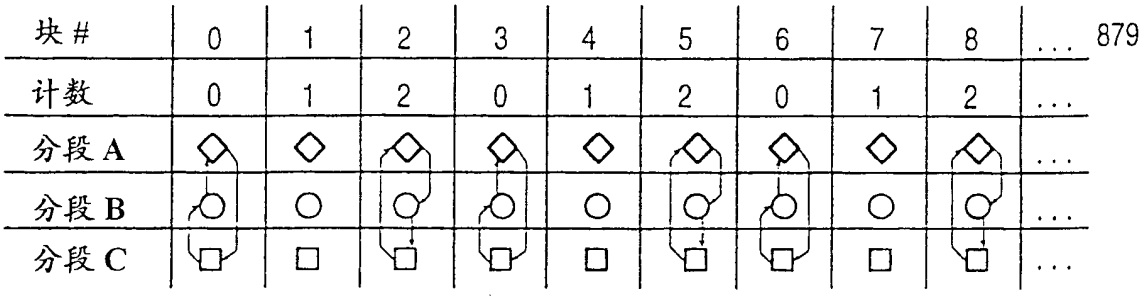
图 13a



模块化混洗 1410

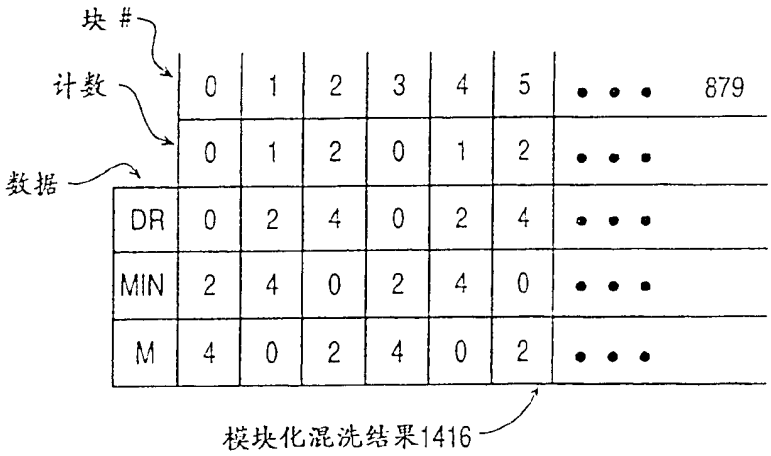


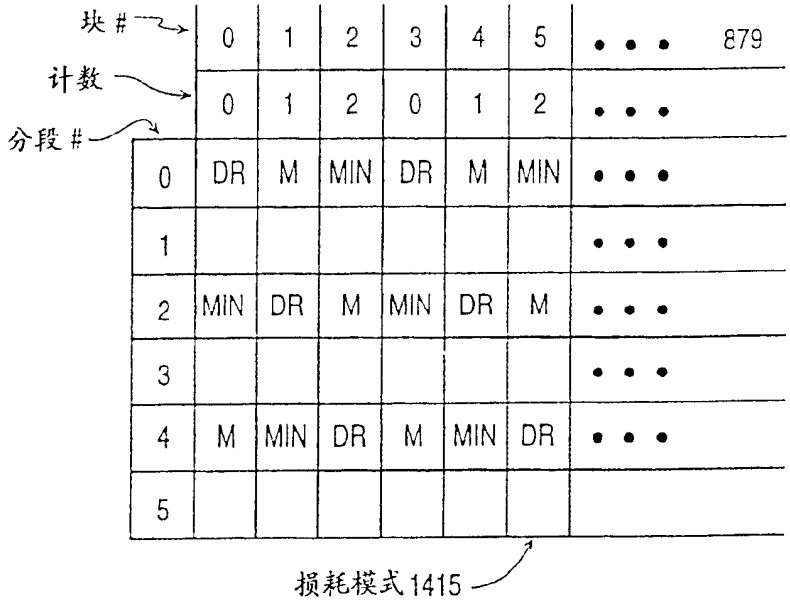
模块化混洗 1420



运动标志模块化混洗1430

图 14

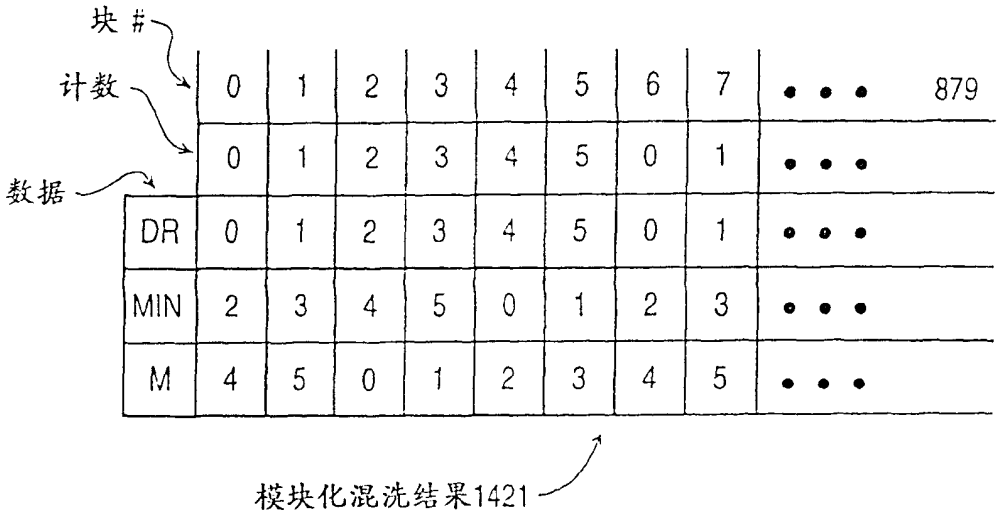




DR		MIN		M		DR		MIN		M	
	M		DR		MIN		M		DR		MIN
DR		MIN		M		DR		MIN		M	
	M		DR		MIN		M		DR		MIN
DR		MIN		M		DR		MIN		M	
	M		DR		MIN		M		DR		MIN
DR		MIN		M		DR		MIN		M	
	M		DR		MIN		M		DR		MIN
DR		MIN		M		DR		MIN		M	
	M		DR		MIN		M		DR		MIN

空间损耗
模式1417

图 14a


块 #
计数
数据
0 1 2 3 4 5 6 7 • • • 879
0 1 2 3 4 5 0 1 • • •
DR 0 1 2 3 4 5 0 1 • • •
MIN 2 3 4 5 0 1 2 3 • • •
M 4 5 0 1 2 3 4 5 • • •
模块化混洗结果1421

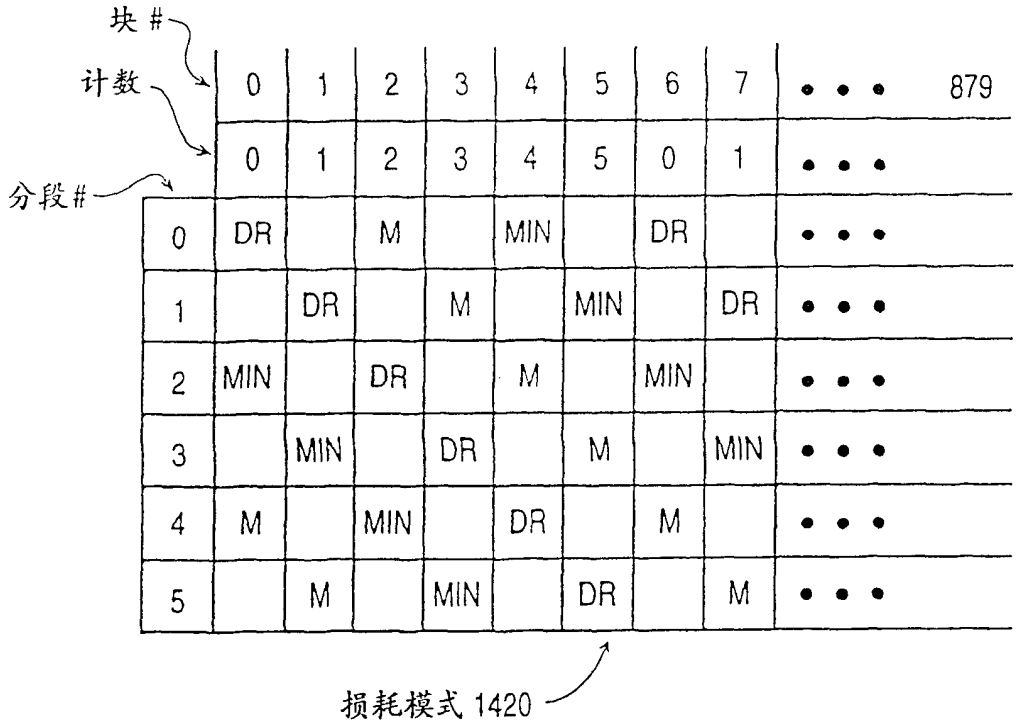
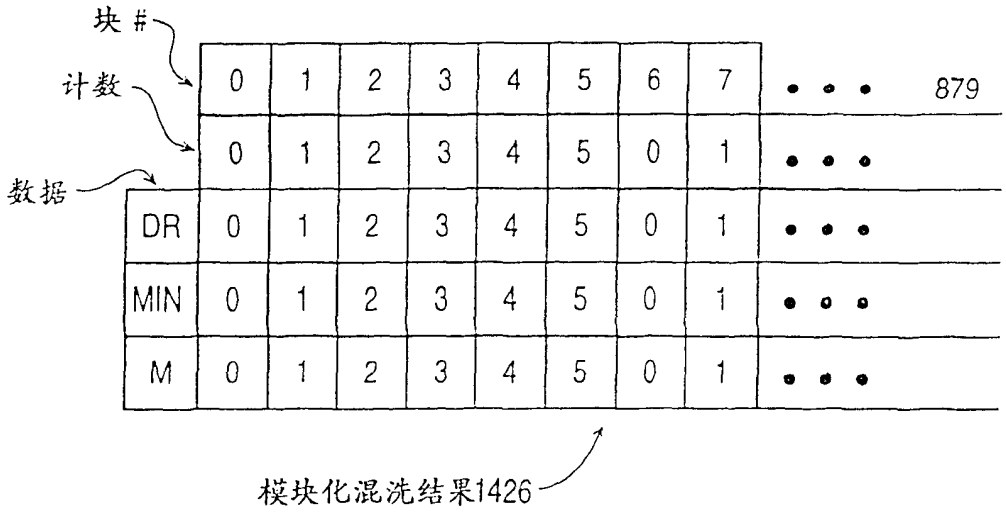

块 #
计数
分段#
0 1 2 3 4 5 6 7 • • • 879
0 1 2 3 4 5 0 1 • • •
0 DR M MIN DR • • •
1 DR M MIN DR • • •
2 MIN DR M MIN • • •
3 MIN DR M MIN • • •
4 M MIN DR M • • •
5 M MIN DR M • • •
损耗模式 1420

图 14b


块 #
计数
数据
0 1 2 3 4 5 6 7 • • • 879
0 1 2 3 4 5 0 1 • • •
DR 0 1 2 3 4 5 0 1 • • •
MIN 0 1 2 3 4 5 0 1 • • •
M 0 1 2 3 4 5 0 1 • • •
模块化混洗结果1426

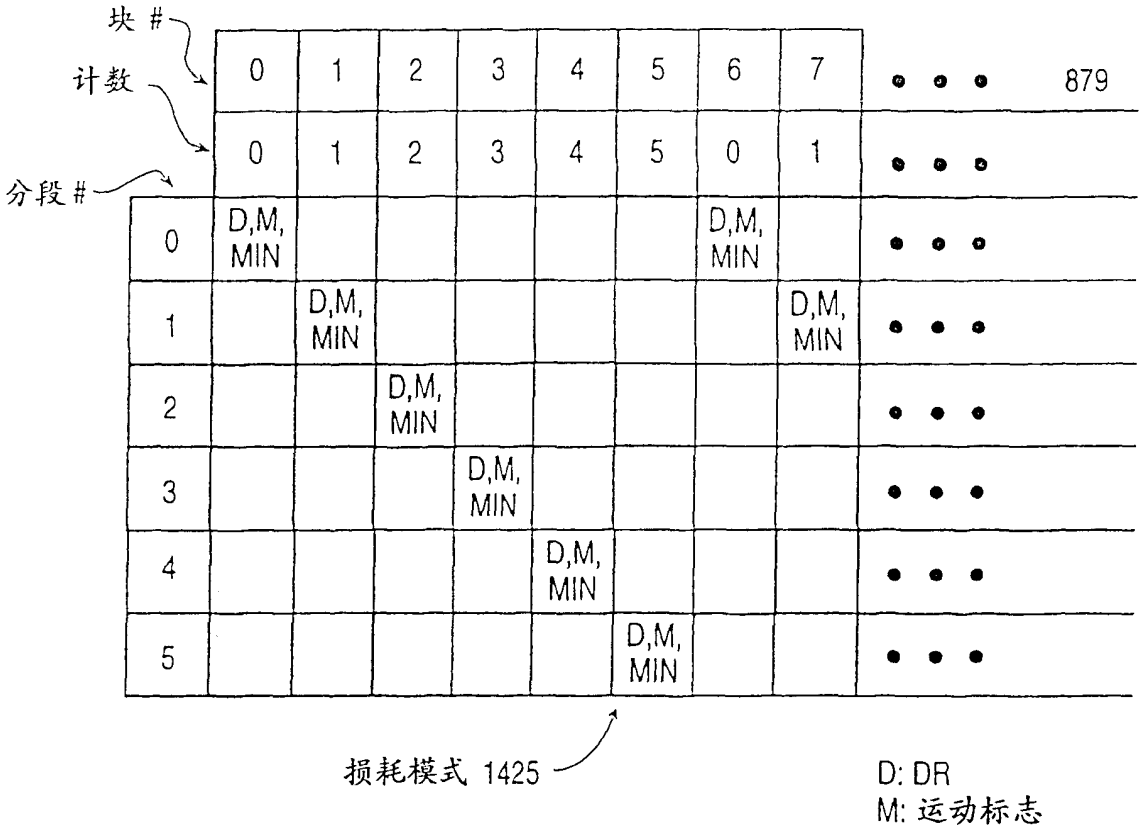

块 #
计数
分段 #
0 1 2 3 4 5 6 7 • • • 879
0 1 2 3 4 5 0 1 • • •
0 D,M, MIN D,M, MIN • • •
1 D,M, MIN D,M, MIN • • •
2 D,M, MIN • • •
3 D,M, MIN • • •
4 D,M, MIN • • •
5 D,M, MIN • • •
损耗模式 1425
D: DR
M: 运动标志

图 14c

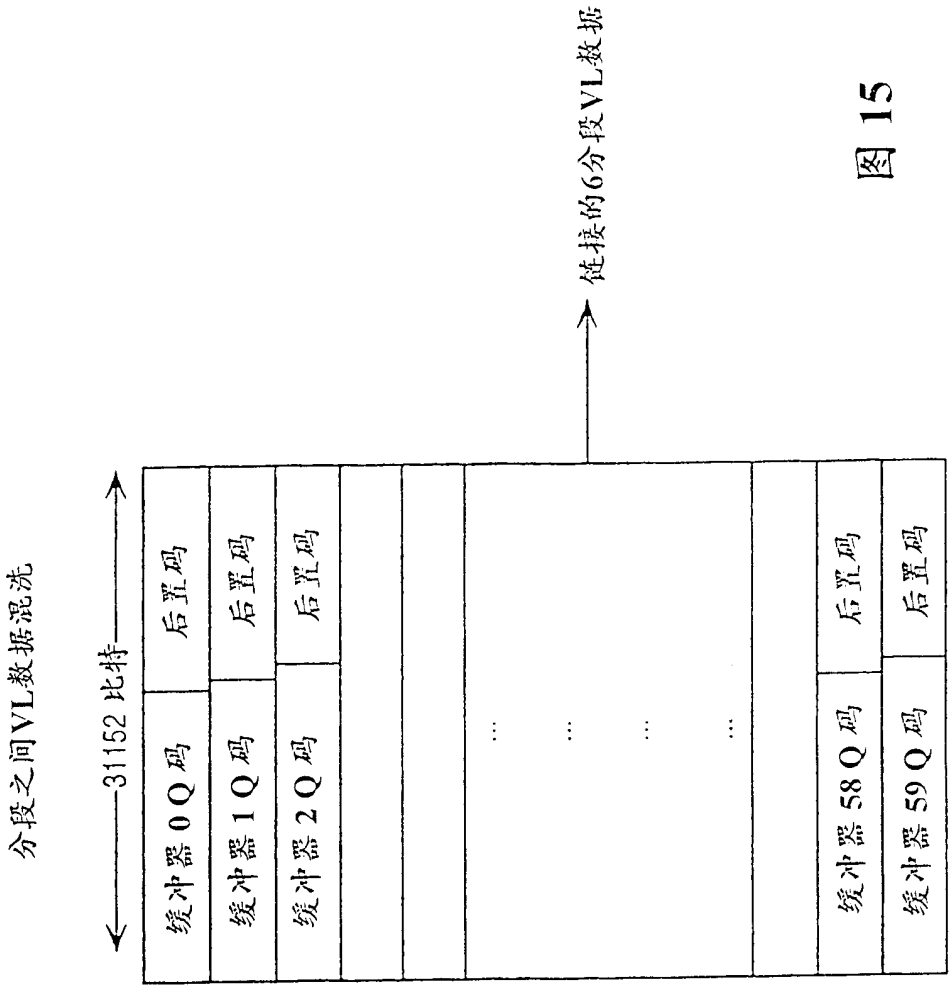

分段之间VL数据混洗
31152 比特
缓冲器 0 Q 码
后置码
缓冲器 1 Q 码
后置码
缓冲器 2 Q 码
后置码
缓冲器 58 Q 码
后置码
缓冲器 59 Q 码
后置码
链接的6分段VL数据

图 15

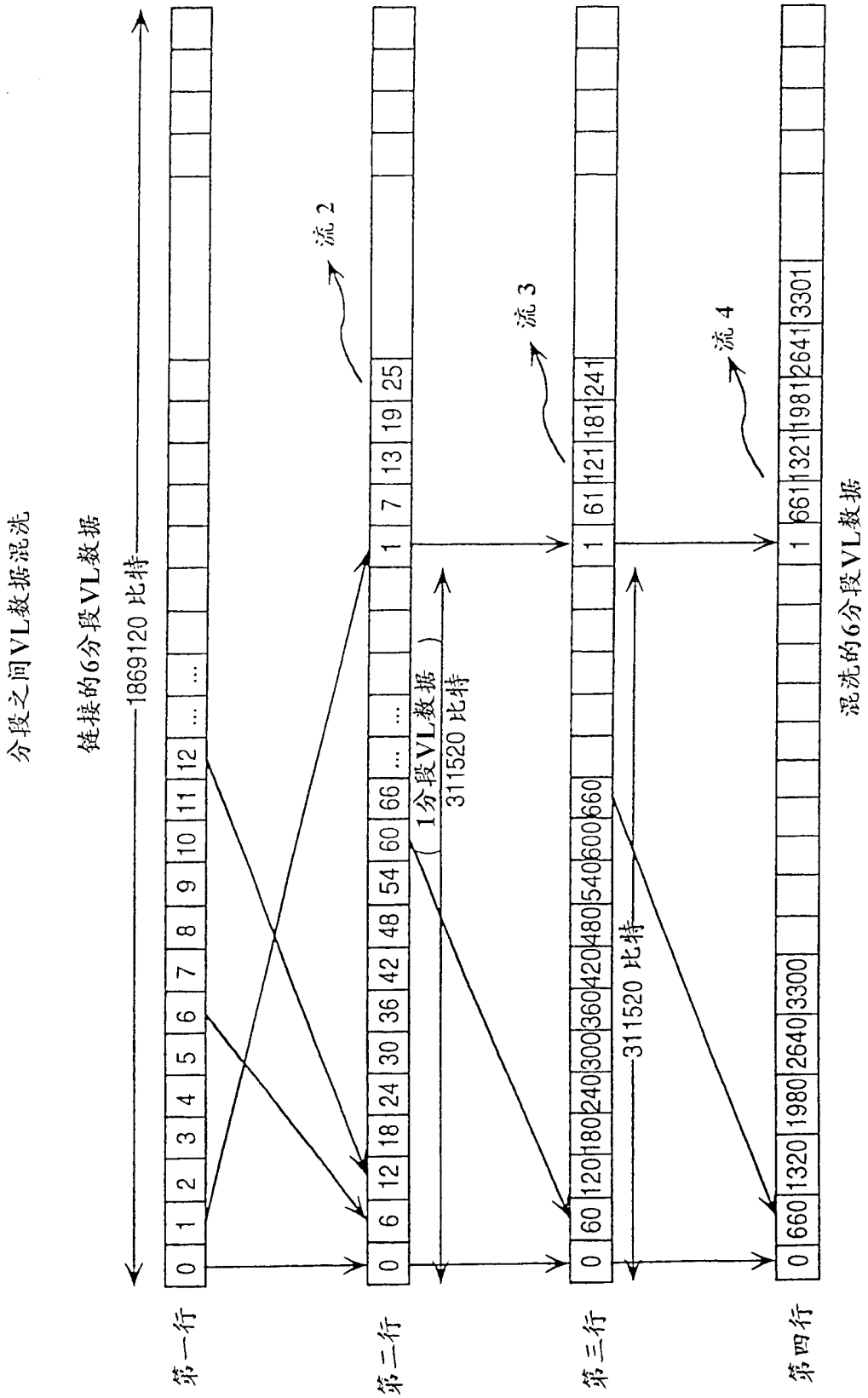

分段之间VL数据混洗
链接的6分段VL数据
1869120 比特
第一行
第二行
第三行
第四行
流 2
流 3
流 4
1分段VL数据
311520 比特
311520 比特
混洗的6分段VL数据

图 16

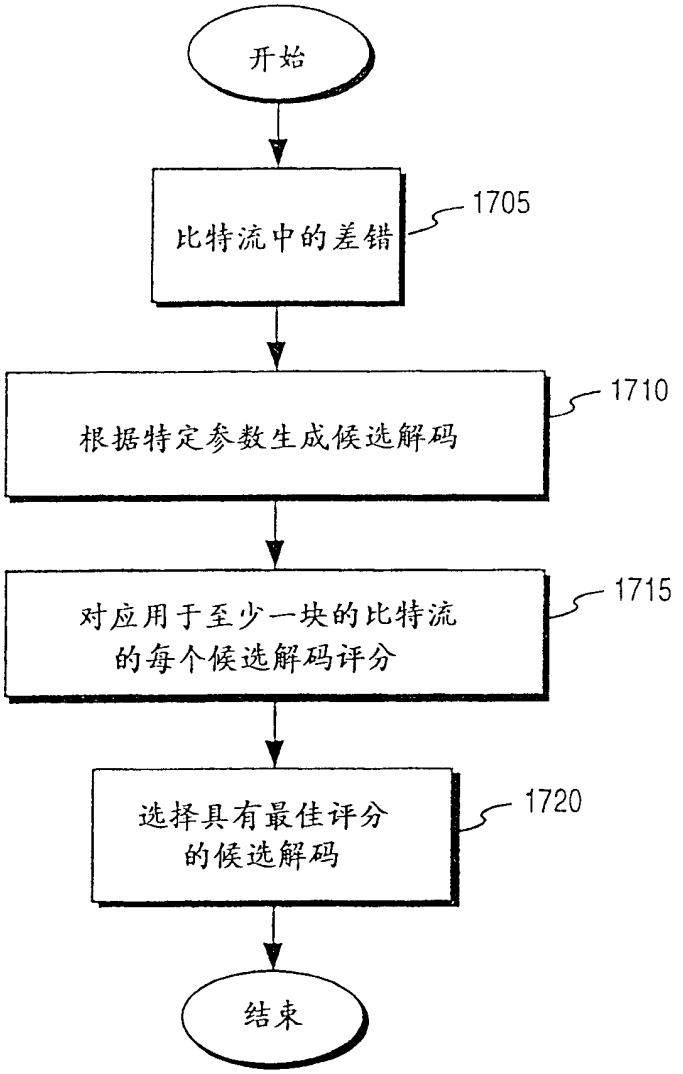

开始
比特流中的差错
1705
根据特定参数生成候选解码
1710
对应用于至少一块的比特流
的每个候选解码评分
1715
选择具有最佳评分
的候选解码
1720
结束

图 17

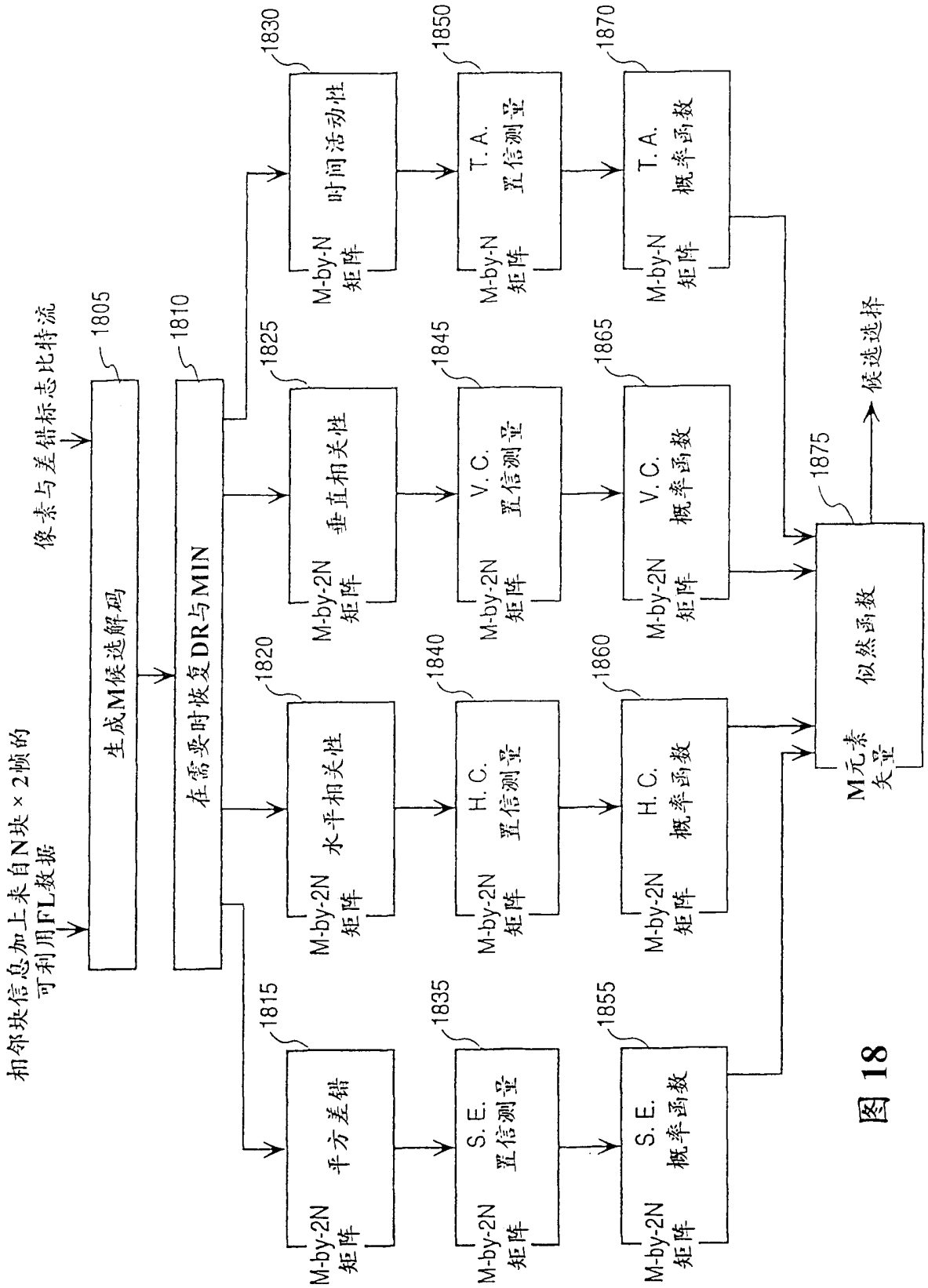

相邻块信息加上来自N块×2帧的可利用FL数据
像素与差错标志比特流
1805
生成M候选解码
1810
在需要时恢复DR与MIN
1815
M-by-2N 矩阵
平方差错
1820
M-by-2N 矩阵
水-平相关性
1825
M-by-2N 矩阵
垂直相关性
1830
M-by-N 矩阵
时间活动性
1835
M-by-2N 矩阵
S. E. 置信测量
1840
M-by-2N 矩阵
H. C. 置信测量
1845
M-by-2N 矩阵
V. C. 置信测量
1850
M-by-N 矩阵
T. A. 置信测量
1855
M-by-2N 矩阵
S. E. 概率函数
1860
M-by-2N 矩阵
H. C. 概率函数
1865
M-by-2N 矩阵
V. C. 概率函数
1870
M-by-N 矩阵
T. A. 概率函数
1875
M元素矢量
似然函数
候选选择

图 18

(mi, qi) 可能的情况 (x 表示未知数)

(mi, qi)	d_i 可能的值	可能性的数量 (n_i)
(mi = m, qi = q)	{5m + q}	1
(mi = x, qi = 0)	{0}	1
(mi = x, qi > 0)	{qi, 5 + qi}	2
(mi = 0, qi = x)	{0, 1, 2, 3, 4}	5
(mi = 1, qi = x)	{6, 7, 8, 9}	4
(mi = x, qi = x)	{0, 1, 2, 3, 4, 6, 7, 8, 9,}	9

● 生成所有 $M = n_2 \times n_1 \times n_0$ 可能的密钥

图 19

测量: 均方误差

$$S.E. = \sum_i \sum_j (x_i - y_{i,j})^2$$

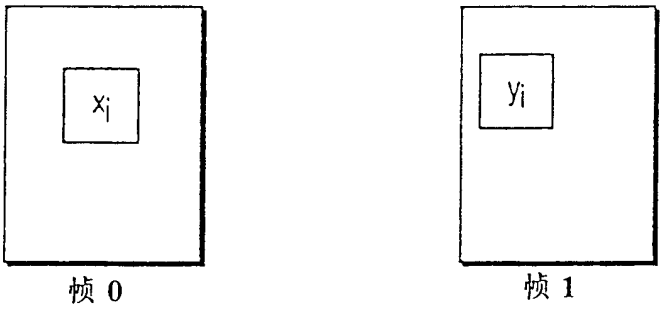
其中 x_i是像素i的(候选)解码值

y_{ij} 是 x_i 的水平或垂直相邻像素

*注意: 当前实施例抛弃3个最大项,即$(x_i - y_{i,j})^2$

图 20a

测量: 时间活动



x_i 与 y_i 是块 j 帧 0 与块 j 帧 1 中相应的像素

$$T.A. = \max | yi - xi |$$

块 j 中的 i

图 20b

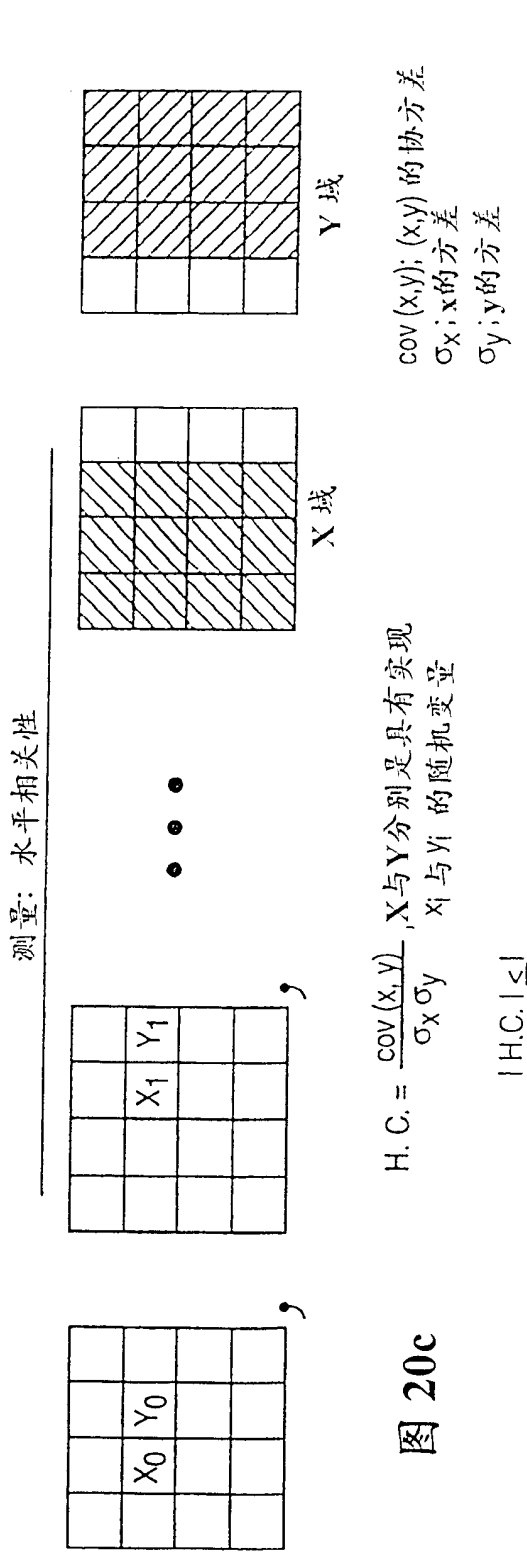

测量：水平相关性
X域
Y域
X_0 Y_0
X_1 Y_1
$H.C. = \frac{cov(x,y)}{\sigma_x \sigma_y}$,X与Y分别是具有实现
x_i 与 y_i 的随机变量
$|H.C.| \leq 1$
cov(x,y): (x,y) 的协方差
σ_x : x的方差
σ_y : y的方差

图 20c

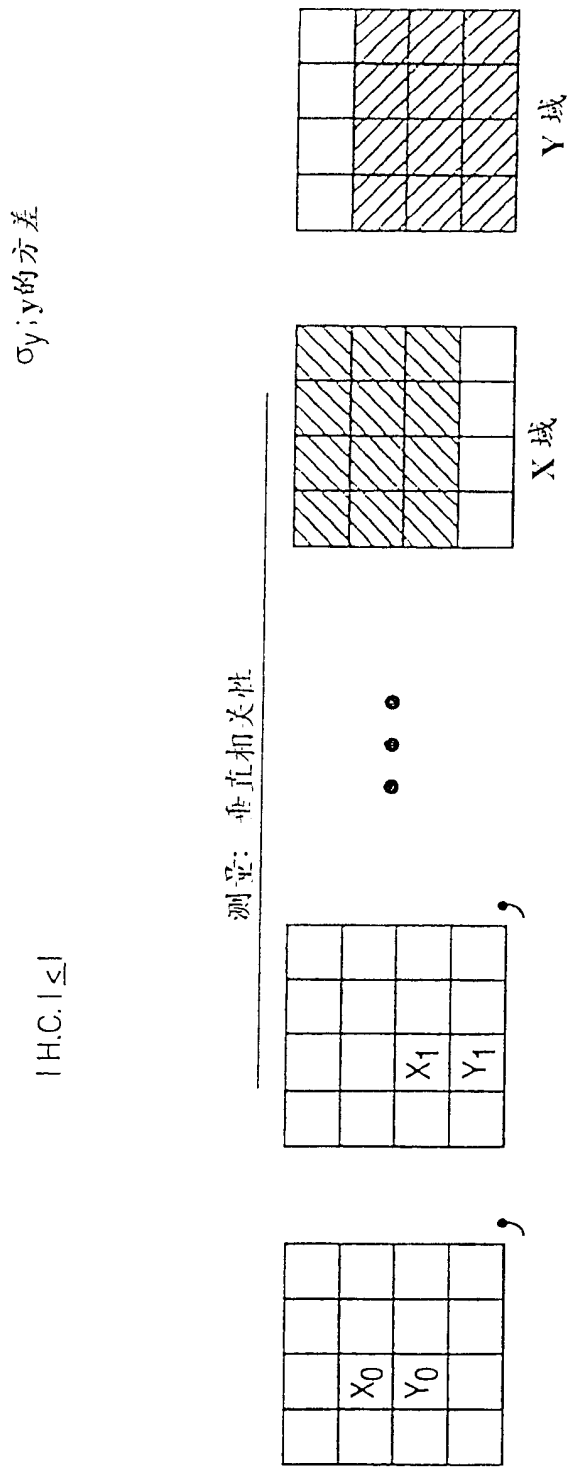

测量：垂直相关性
X域
Y域
X_0 Y_0
X_1 Y_1
$V.C. = \frac{cov(x,y)}{\sigma_x \sigma_y}$, $|V.C.| \leq 1$

图 20d

概率函数

S.E. 概率 (测量的 9/97)

S. E. 概率			S.E. <=2^(9)	2^(9) < S.E. <= 2^(11)	2^(11) < S.E. <= 2^(13)	2^(13) < S.E. <= 2^(15)
conf = 0			0.400607	0.486165	0.497679	0.499834
0	< conf <=	0.1	0.108873	0.097436	0.089769	0.102244
0.1	< conf <=	0.2	0.044734	0.014264	0.014457	0.004728
0.2	< conf <=	0.3	0.016635	0.001637	0.001385	0.001193
0.3	< conf <=	0.4	0.007419	0.000513	0.000001	0.000001
0.4	< conf <=	0.5	0.001317	0.000018	0.000022	0.000001
0.5	< conf <=	0.6	0.000264	0.000026	0.000001	0.000008
0.6	< conf <=	0.7	0.000141	0.000031	0.000001	0.000001
0.7	< conf <=	0.8	0.000078	0.000001	0.000001	0.000001
0.8	< conf <=	0.9	0.000001	0.000001	0.000001	0.000001
0.9	< conf <=	1	0.000001	0.000001	0.000001	0.000001

S. E. 概率			2^(15) < S.E. <= 2^(17)	2^(17) < S.E. <= 2^(19)	2^(19) < S.E.
conf = 0			0.499842	0.499689	0.498521
0	< conf <=	0.1	0.093066	0.091473	0.085688
0.1	< conf <=	0.2	0.003602	0.003743	0.003322
0.2	< conf <=	0.3	0.000001	0.000189	0.000192
0.3	< conf <=	0.4	0.000001	0.000075	0.000001
0.4	< conf <=	0.5	0.000001	0.000034	0.000001
0.5	< conf <=	0.6	0.000001	0.000001	0.000001
0.6	< conf <=	0.7	0.000001	0.000001	0.000001
0.7	< conf <=	0.8	0.000001	0.000001	0.000001
0.8	< conf <=	0.9	0.000001	0.000001	0.000001
0.9	< conf <=	1	0.000001	0.000001	0.000001

图 21

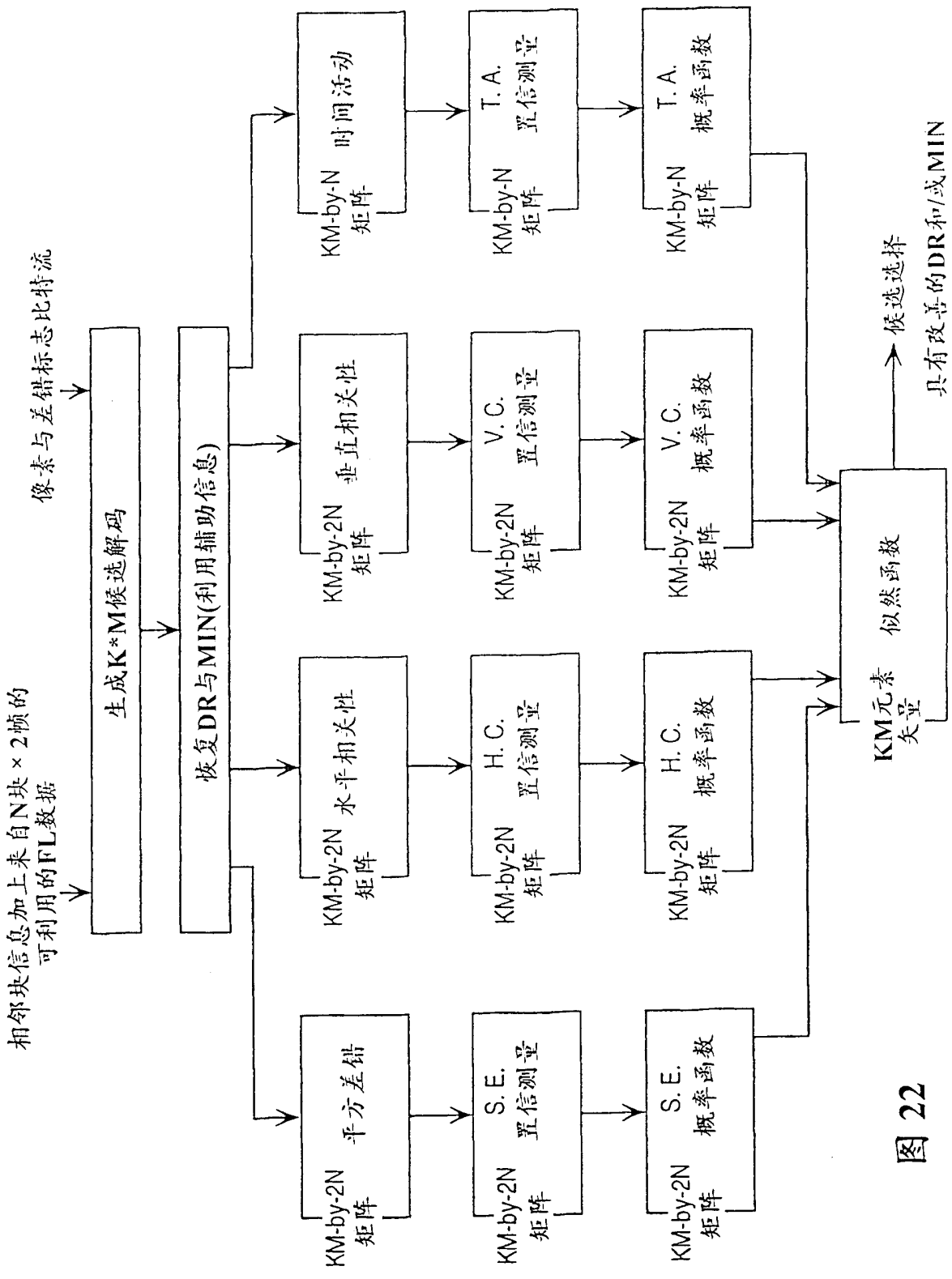

相邻块信息加上来自N块×2帧的
可利用的FL数据
像素与差错标志比特流
生成K*M候选解码
恢复DR与MIN(利用辅助信息)
KM-by-2N 矩阵
平方差错
S. E. 置信测量
S. E. 概率函数
水平相关性
H. C. 置信测量
H. C. 概率函数
垂直相关性
V. C. 置信测量
V. C. 概率函数
时间活动
T. A. 置信测量
T. A. 概率函数
KM-by-N 矩阵
KM元素矢量
似然函数
候选选择
具有改善的DR和/或MIN

图 22

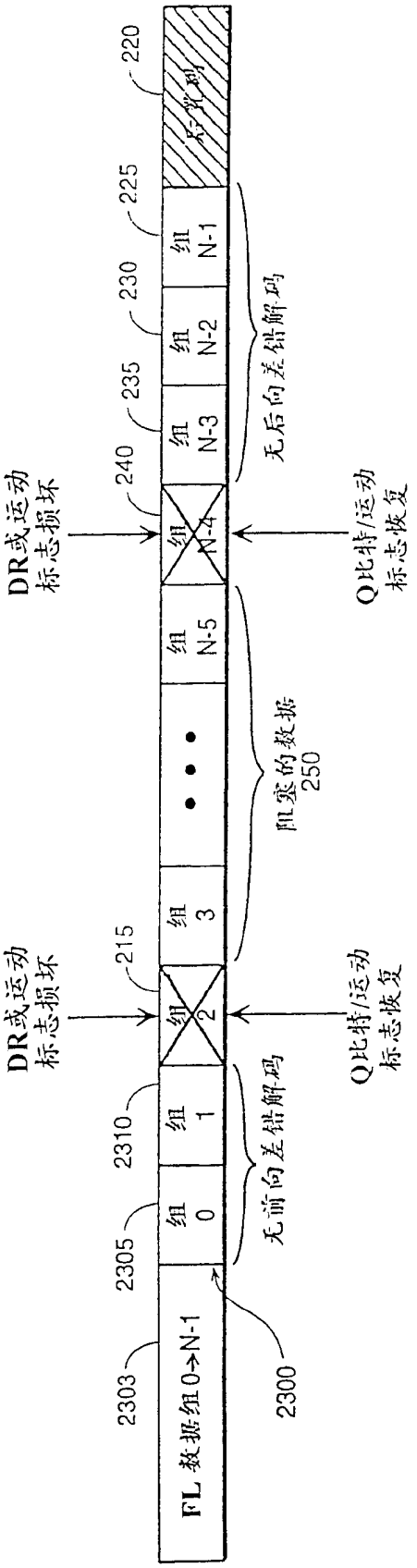

FL 数据组0→N-1
组 0
组 1
组 2
组 3
组 N-5
组 N-4
组 N-3
组 N-2
组 N-1
2303
2305
2310
215
240
235
230
225
220
2300
DR或运动标志损坏
DR或运动标志损坏
无前向差错解码
Q比特/运动标志恢复
Q比特/运动标志恢复
阻塞的数据 250
无后向差错解码
后导码

图 23

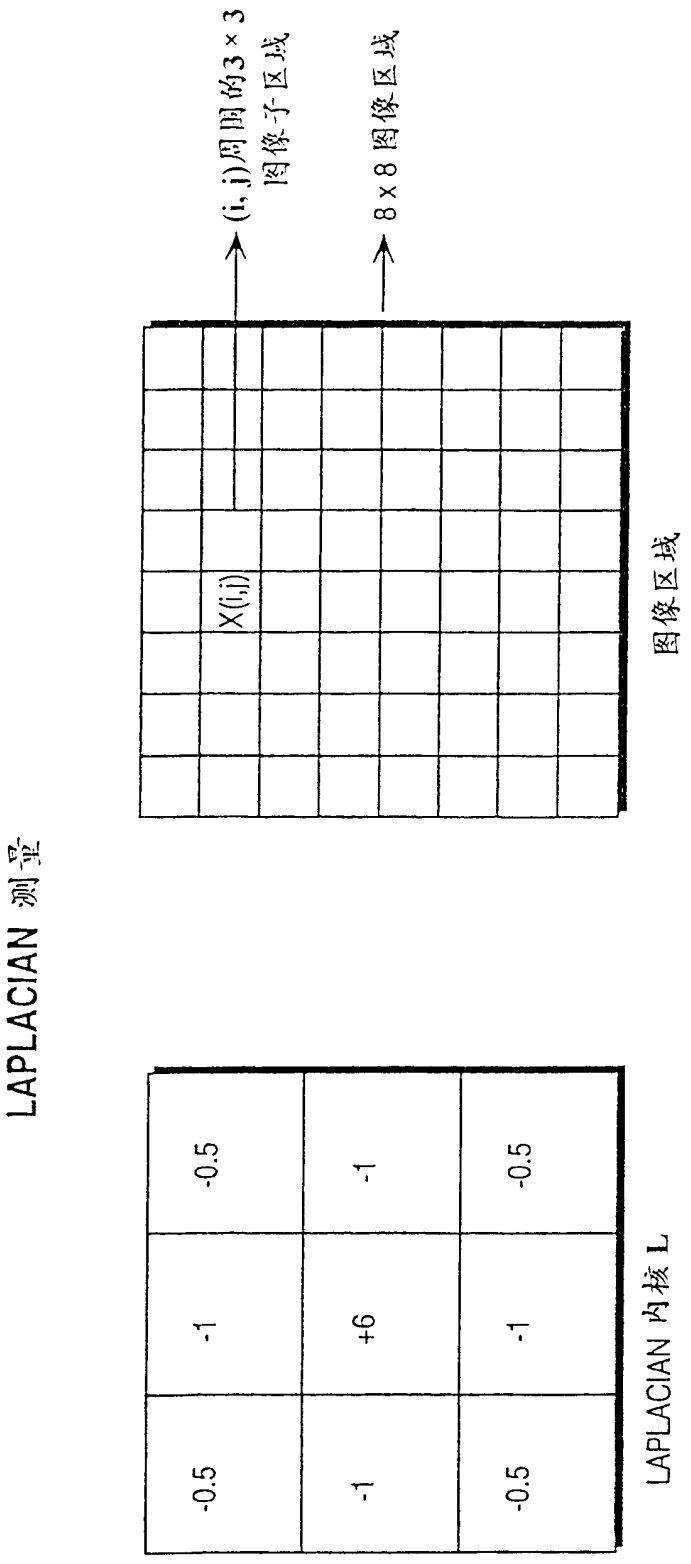

LAPLACIAN 测量
(i, j)周围的3 × 3
图像子区域
8 × 8 图像区域
X(i,j)
图像区域
-0.5
-1
-0.5
-1
+6
-1
-0.5
-1
-0.5
LAPLACIAN 内核 L

图 24b

图 24a

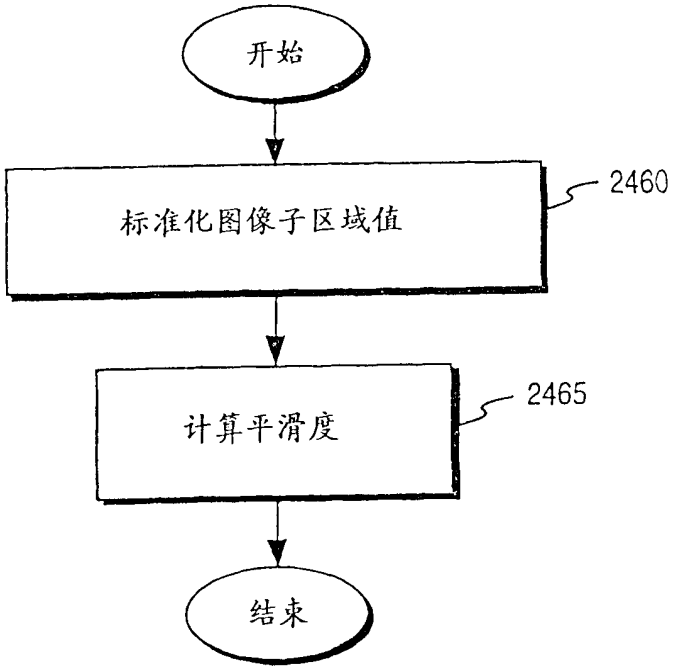


图 24c

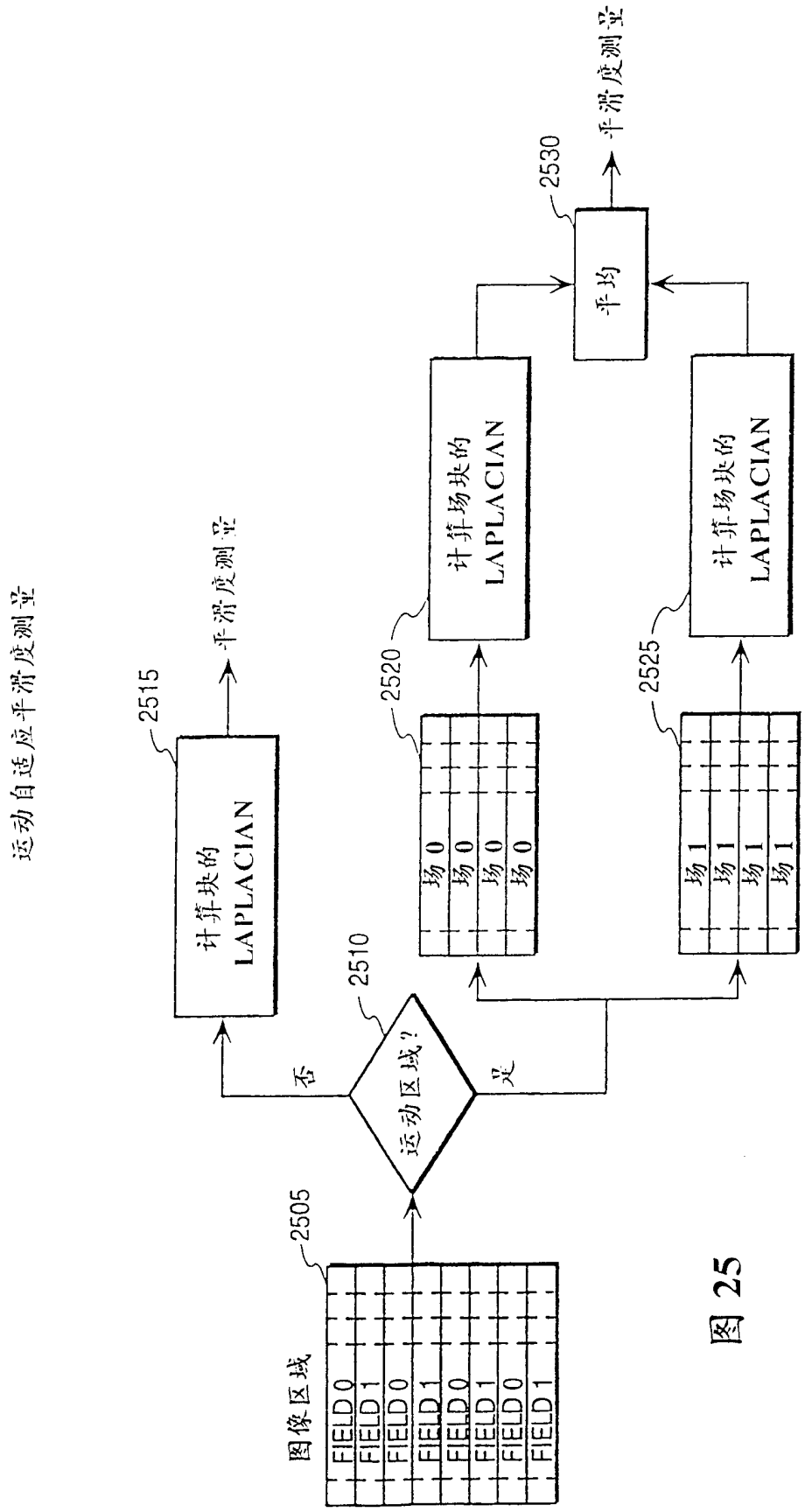

运动自适应平滑度测量
图像区域
2505
FIELD 0
FIELD 1
FIELD 0
FIELD 1
FIELD 0
FIELD 1
FIELD 0
FIELD 1
运动区域?
2510
否
是
计算块的
LAPLACIAN
2515
平滑度测量
场 0
场 0
场 0
场 0
2520
计算场块的
LAPLACIAN
场 1
场 1
场 1
场 1
2525
计算场块的
LAPLACIAN
平均
2530
平滑度测量

图 25

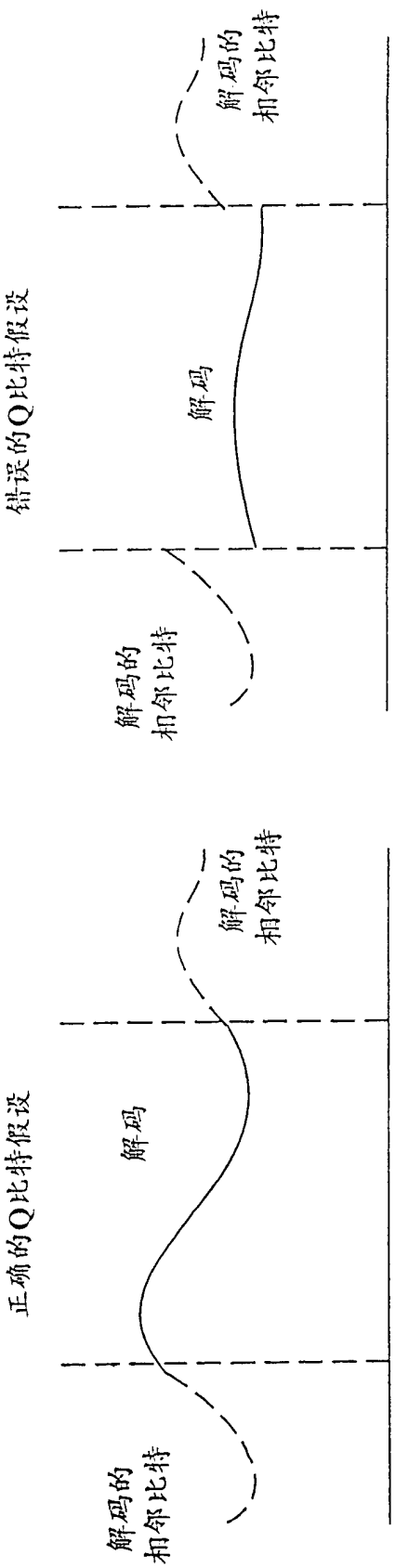


图 26a

图 26b

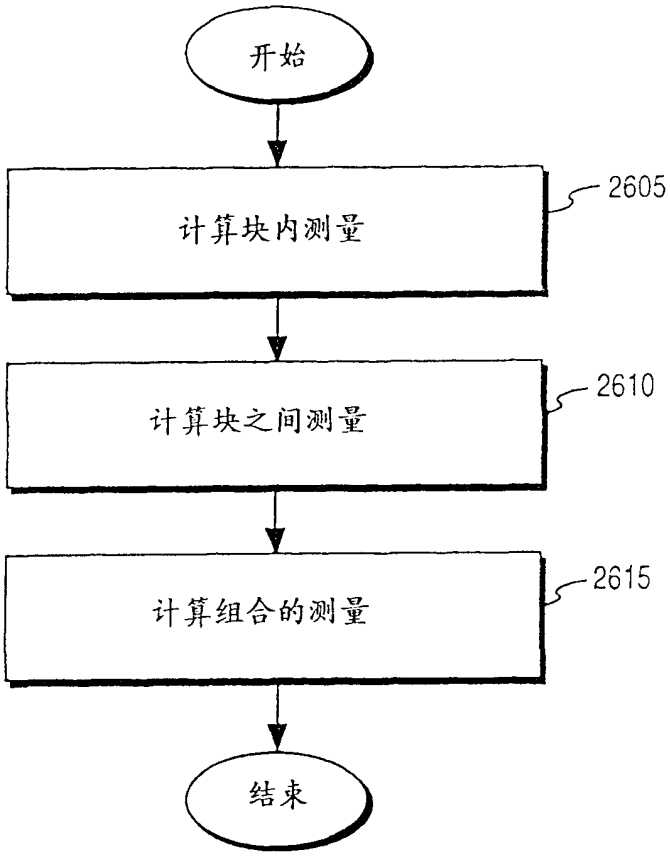


图 26c

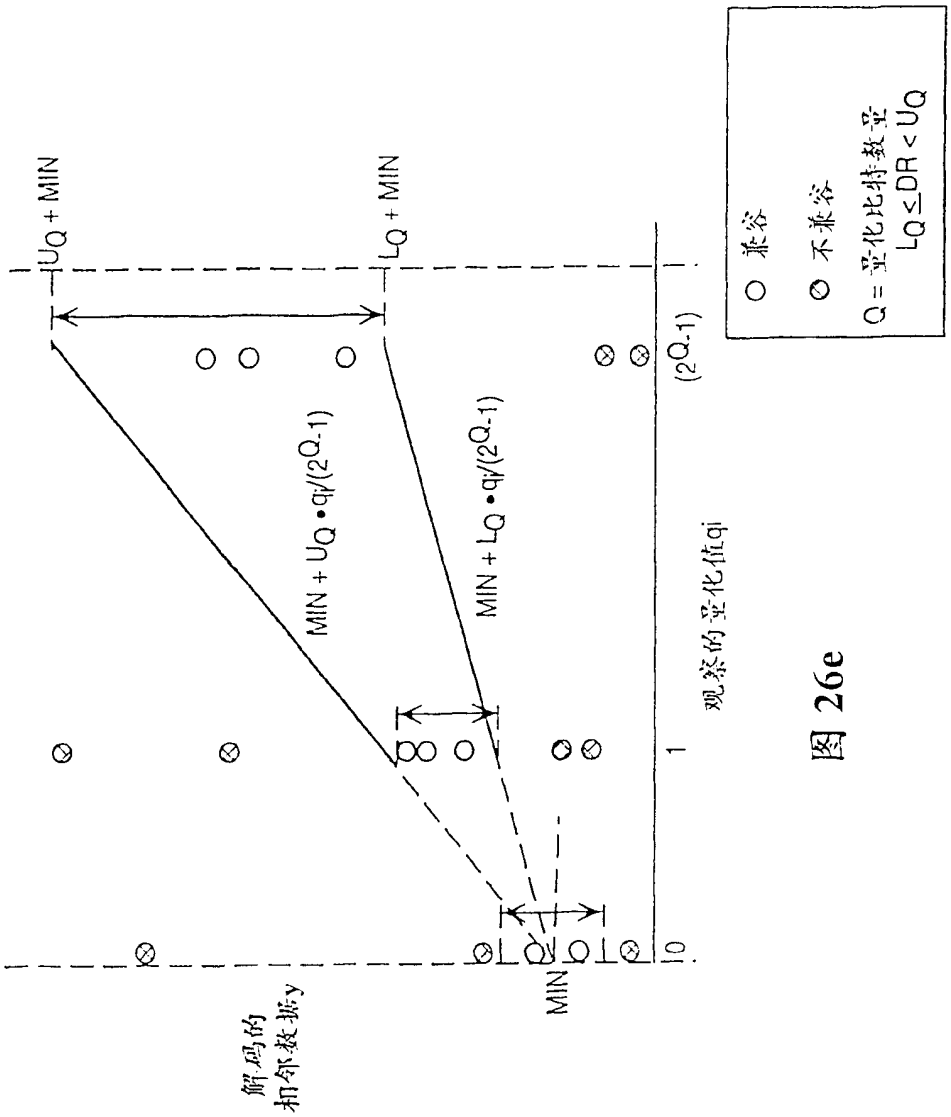

UQ + MIN
LQ + MIN
MIN + UQ • qi/(2Q-1)
MIN + LQ • qi/(2Q-1)
(2Q-1)
观察的变化值qi
MIN
解码的
扣尔数据y
○ 兼容
⊘ 不兼容
Q = 变化比特数量
LQ≤DR < UQ

图 26e

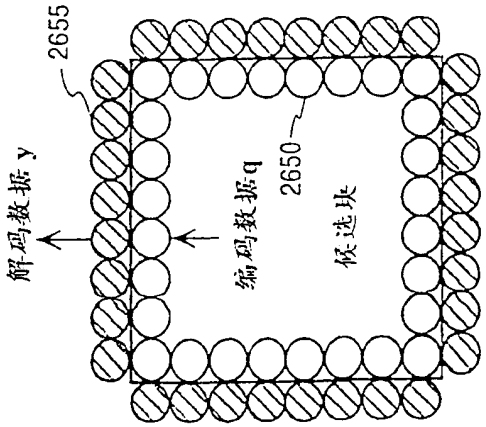

2655
解码数据 y
编码数据q
2650
候选块

图 26d

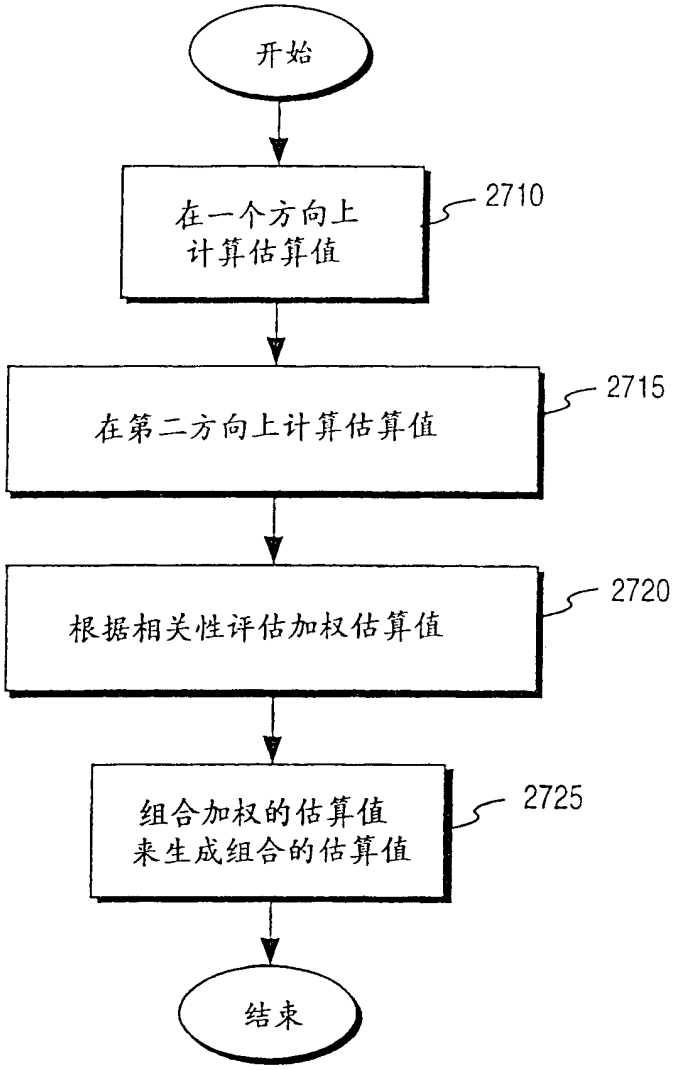

开始
在一个方向上
计算估算值
2710
在第二方向上计算估算值
2715
根据相关性评估加权估算值
2720
组合加权的估算值
来生成组合的估算值
2725
结束

图 27a

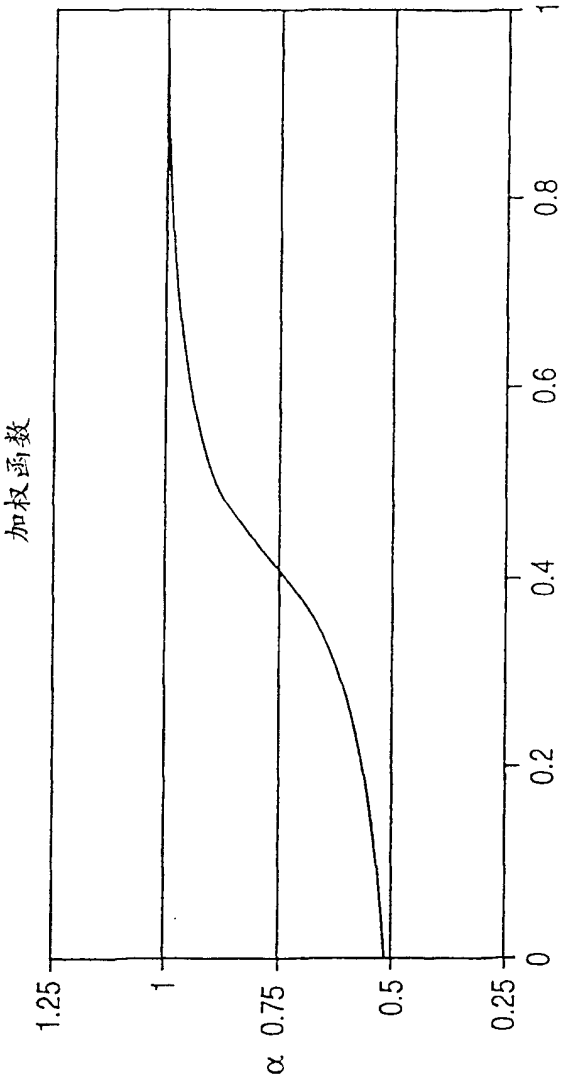


图 27b