



(12)发明专利申请

(10)申请公布号 CN 109076240 A

(43)申请公布日 2018.12.21

(21)申请号 201780024589.0

(72)发明人 林建良 林鸿志 李佳盈 张胜凯

(22)申请日 2017.05.08

(74)专利代理机构 北京市万慧达律师事务所

(30)优先权数据

62/332,505 2016.05.06 US

62/356,571 2016.06.30 US

15/585,224 2017.05.03 US

11111

代理人 王蕊 白华胜

(85)PCT国际申请进入国家阶段日

2018.10.19

(51)Int.Cl.

H04N 19/597(2014.01)

H04N 19/88(2014.01)

H04N 13/139(2018.01)

G06T 3/00(2006.01)

(86)PCT国际申请的申请数据

PCT/CN2017/083476 2017.05.08

(87)PCT国际申请的公布数据

W02017/190710 EN 2017.11.09

(71)申请人 联发科技股份有限公司

地址 中国台湾新竹市新竹科学工业园区笃

行一路一号

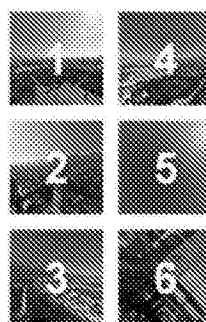
权利要求书2页 说明书25页 附图15页

(54)发明名称

将全向图像映射到布局输出格式的方法和装置

(57)摘要

公开了处理全向图像的方法和装置。根据一种方法,接收使用所选择的投影格式从360度全景视频序列中的每个球面图像转换的当前一组全向图像,其中所选择的投影格式属于包括立方体格式的投影格式组,以及具有该立方体格式的该当前全方向图像集合由六个立方体组成。如果该所选择的投影格式对应于该立方体格式,将该当前全方向图像集合映射到属于输出布局格式集合的当前立方体贴图图像的一个或多个映射语法元素被标识。提供比特流中的编解码数据,所述比特流包括用于该当前全方向图像集合的该一个或多个映射语法元素。



1. 一种全向图像处理方法,所述方法包括:

使用所选择的投影格式接收从360度全景视频序列中的每个球形图像转换的当前全向图像集合,其中所选择的投影格式属于包括立方面格式的投影格式组,以及具有该立方面格式的该当前全方向图像集合由六个立方面组成;以及

如果该所选择的投影格式对应于该立方面格式:

标识将该当前全方向图像集合映射到属于输出布局格式集合的当前立方体贴图图像的一个或多个映射语法元素,该输出布局格式集合包括从包括立方体贴图布局 and 等距长方投影格式的布局组中选择的至少两个输出布局格式;以及

提供比特流中的编解码数据,所述比特流包括用于该当前全方向图像集合的该一个或多个映射语法元素。

2. 根据权利要求1所述的方法,其特征在于,该投影格式组还包括180-3D格式、圆柱图格式、cubemap_32布局、cubemap_180布局、plane_poles布局、plane_poles_6布局、plane_poles_cubemap布局、plane_cubemap布局、plane_cubemap_32布局、flat_fixed布局、cubemap_1x6布局、cubemap_2x3布局、cubemap_3x2布局和cubemap_6x1布局。

3. 根据权利要求2所述的方法,其特征在于,如果该当前全方向图像集合是该等距长方投影格式,则将该当前全方向图像集合转换成该立方面格式,并且通过将转换的当前全方向图像集合视为具有该立方面格式,该一个或多个映射语法元素被标识用于该转换的当前全方向图像集合。

4. 根据权利要求1所述的方法,其特征在于,该一个或多个映射语法元素包括与该当前立方体贴图图像相关联的当前立方体类型,并且该当前立方体类型属于由1x6立方体贴图布局、2x3立方体贴图布局、3x2立方体贴图布局和6x1立方体贴图布局组成的当前输出布局格式集合。

5. 根据权利要求4所述的方法,其特征在于,该一个或多个映射语法元素还包括布局映射索引,其中每个布局映射索引将该当前全方向图像集合的一个立方面与该当前立方体贴图图像的一个位置相关联。

6. 根据权利要求5所述的方法,其特征在于,每个布局映射索引使用具有等于要映射的立方面数的条目数的代码表进行编解码。

7. 根据权利要求5所述的方法,其特征在于,对于该当前全方向图像集合的每个立方面,除了该当前全方向图像集合的最后一个立方面之外,标识一个布局映射索引。

8. 根据权利要求7所述的方法,其特征在于,每个布局映射索引使用具有等于要映射的剩余立方面数的条目数的代码表进行编解码。

9. 根据权利要求5所述的方法,其特征在于,该一个或多个映射语法元素还包括旋转索引,其中每个旋转索引指示该当前全方向图像集合的一个立方面在该当前立方体贴图图像的所述一个位置处的旋转角度。

10. 根据权利要求9所述的方法,其特征在于,针对该当前全方向图像集合的每个立方面,标识一个旋转索引。

11. 根据权利要求10所述的方法,其特征在于,每个旋转索引使用代码表进行编解码,以指示从对应于 $\{0^\circ$ 和 $90^\circ\}$, $\{0^\circ$, $+90^\circ$, -90° 和 $180^\circ\}$ 或 $\{0^\circ$, 90° , 180° 和 $270^\circ\}$ 选转角度集合中选择一个旋转角度。

12. 根据权利要求9所述的方法,其特征在於,该一个或多个映射语法元素还包括默认布局标志,用于指示具有该当前立方体类型的该当前全方向图像集合是否使用默认立方体贴图图像,并且其中仅当该默认布局标志指示该默认立方体贴图图像未被用于该当前全方向图像集合时,该布局映射索引和该旋转索引被标识用于该当前全方向图像集合。

13. 根据权利要求9所述的方法,其特征在於,该一个或多个映射语法元素还包括默认布局标志,用于指示具有该当前立方体类型的该当前全方向图像集合是否使用默认立方体贴图图像,并且其中若该默认布局标志指示该默认立方体贴图图像被用于该当前全方向图像集合时,默认布局映射索引和默认旋转索引被用于该当前全方向图像集合。

14. 根据权利要求1所述的方法,其特征在於,用于该当前全方向图像集合的该输出布局格式集合以序列级别、视图级别、图像级别、切片级别、序列参数集合、视频参数集合、或应用参数集合标识在360度全景视频序列的比特流中。

15. 根据权利要求1所述的方法,其特征在於,该一个或多个映射语法元素以序列级别、视图级别、图像级别、切片级别、序列参数集、视频参数集、或应用参数集标识在360度全景视频序列的比特流中。

16. 根据权利要求1所述的方法,其特征在於,该一个或多个映射语法元素基于一个或多个参考映射语法元素被预测地标识。

17. 根据权利要求16所述的方法,其特征在於,该一个或多个参考映射语法元素的多个集合以序列级别、视图级别、或图像级别标识在用于360度全景视频序列的比特流中,并且在切片级别或图像级别中标识标志,以从该当前全方向图像集合的一个或多个参考映射语法元素的该多个集合中选择所述一个或多个映射语法元素。

18. 根据权利要求16所述的方法,其特征在於,一个或多个参考映射语法元素由来自先前图片、切片或帧的一个或多个第一映射语法元素预测。

19. 一种全向图像处理装置,所述全向图像处理装置包括一个或多个电子电路或处理器,其被布置为:

使用所选择的投影格式接收从360度全景视频序列中的每个球形图像转换的当前全向图像集合,其中所选择的投影格式属于包括立方面格式的投影格式组,以及具有该立方面格式的该当前全方向图像集合由六个立方面组成;以及

如果该所选择的投影格式对应于该立方面格式:

标识一个或多个映射语法元素以将该当前全方向图像集合映射到属于输出布局格式集合的当前立方体贴图图像,该输出布局格式集合包括从包括立方体贴图布局、cubemap_32布局、cubemap_180布局、plane_poles布局、plane_poles_6布局、plane_poles_cubemap布局、plane_cubemap布局、plane_cubemap_32布局、flat_fixed布局、cubemap_1x6布局、cubemap_2x3布局、cubemap_3x2布局,和cubemap_6x1布局的布局集合中选择的至少两个输出布局格式;以及

提供比特流中的编解码数据,所述比特流包括用于该当前全方向图像集合的该一个或多个映射语法元素。

将全向图像映射到布局输出格式的方法和装置

[0001] 【交叉引用】

[0002] 本申请要求2016.5.6提交的美国临时申请号为62/332,505的美国临时申请案和2016.6.30提交的美国临时申请号为62/356,571的美国临时申请案的优先权,其全部内容也一并引用于此。

【技术领域】

[0003] 本发明涉及360度视频。更具体地说,本发明涉及使用所选择的投影格式将360度全景视频序列中的每个球面图像(spherical image)转换的一组全向图像(omnidirectional image)映射成输出格式。

【背景技术】

[0004] 数字视频编码标准的进步导致了过去十年来诸如智能电话、数字电视和数字照相机等多媒体系统的成功。在H.261、MPEG-1、MPEG-2、H.263、MPEG-4和H.264/AVC的标准化活动之后,由于需要更大的图像分辨率、更高的帧率(frame rate)、更好的视频品质,对视频压缩性能的提高的需求依然强劲。因此,开发新的、比H.264/AVC更好的编解码效率的视频编解码技术一直都没有结束。HEVC基于基于混合块的运动补偿变换编解码架构。

[0005] 三维(3D)电视机近年来一直是技术趋势,旨在提供给观众轰动的观看体验。已经开发出各种技术来实现3D观看,而多视角视频是3D电视应用的关键技术之一。例如,视频可以是二维(2D)媒介,其仅从摄像机的角度向观看者提供场景的单一视图。然而,多视图视频能够提供动态场景的任意观点(viewpoint),并为观众提供现实感的感觉。3D视频格式还可以包括与相应纹理图片(texture picture)相关联的深度图(depth map)。深度图也必须进行编解码,以呈现三维视图或多视图。由于编解码多视图数据,较大图像分辨率和更好质量的要求,提高了3D和多视图视频编码效率的强烈需求,提出了各种技术。

[0006] 作为对HEVC和下一代3D视频编解码标准的扩展,3D-HEVC视频编解码标准的标准化由3D视频编解码扩展开发(Joint Collaborative Team on 3D video coding Extension Development,简称为JCT-3V)联合协作小组于2012年7月份正式推出,并于2015年2月举行的第11届JCT-3V会议后确定。为了更实际地支持自动立体多视图显示,多视图视频加深度(multi-view video plus depth,简称为MVD)格式被引入为新的3D视频格式,用于3D-HEVC。MVD格式由纹理图片及其相关的深度图组成。与表示物体的亮度和色度信息的纹理图像不同,深度图是包含与摄像机拍摄平面的物体的距离相关的信息的图像,并且通常用于作为非视觉信息(non-visual information)的虚拟视图呈现。

[0007] 具有头戴式显示器(head-mounted display,简称为HMD)的虚拟现实(Virtual reality,简称为VR)与各种应用相关联。可以使用向用户显示宽视野内容的能力来提供沉浸式的视觉体验。必须在所有方向上捕获真实世界的环境,从而产生对应于观察球(viewing sphere)的全方位视频。随着相机托架(camera rigs)和HMD的进步,由于代表此类内容所需的高比特率,VR内容的传递很快将成为瓶颈。由于全向视频通常具有4K或更高

的分辨率,压缩对于降低比特率至关重要。提供的全向视频是等距长方投影(equirectangular projection)。图1示出了等距长方投影格式的全向视频(称为“Hangpai_2”)的图像的示例。当原始图像为全彩色时,黑白版本如图1所示(因为黑白图像足以说明本发明)。

[0008] 等距长方投影格式可以转换为例如如图2中所示的不同格式:(a)立方体贴图(cubemap)、(b)Cubemap_32、(c)Cubemap_180、(d)Plane_poles、(e)Plane_poles_6、(f)Plane_poles_cubemap、(g)Plane_cubemap、(h)Plane_cubemap_32、(i)Flat_fixed、(j)180degree 3D视频(即,180-3D)以及(k)圆柱图/圆柱形(Cylindermap/Cylindrical)。图2a到图2i中的图像基于图1的图像。

[0009] 图3示出了使用投影转换(projection conversion)310将等距长方投影格式转换为立方格式(cubic format)的示例,其中标记为1至6的图像对应于用于表示360度视频的立方体的六个面上的图像。图4中示出了四个常用布局(即1x6-布局410,2x3-布局420,3x2-布局430和6x1-布局440)。在每个布局中,来自6个面的图像被组装成一个单个矩形图像(rectangular image)。图5示出了等距长方格式和立方格式之间的几何比较。图5中示出了等距长方几何510和立方几何520。图像512是等距长方格式的示例,图像522是立方格式的示例。

[0010] 在用于将立方方面转换成输出格式的现有方法中,总是使用相同的选择的输出布局格式,并且以固定的方式将六个面分配给输出布局格式。虽然固定的映射很简单,但它阻止用户使用其他布局格式来满足用户的需求。此外,在将立方方面转换为输出布局格式之后,经转换的输出图像经常被压缩以减少所需的空间。选择的输出布局格式和固定映射对于压缩可能不是有效的。

【发明内容】

[0011] 公开了处理全向图像的方法和装置。根据一种方法,接收使用所选择的投影格式从360度全景视频序列中的每个球面图像转换的当前一组全向图像,其中所选择的投影格式属于包括立方方面格式的投影格式组,以及具有该立方方面格式的该当前全方向图像集合由六个立方方面组成。如果该所选择的投影格式对应于该立方方面格式,将该当前全方向图像集合映射到属于输出布局格式集合的当前立方体贴图图像的一个或多个映射语法元素被标识。提供比特流中的编解码数据,所述比特流包括用于该当前全方向图像集合的该一个或多个映射语法元素。

[0012] 投影格式组可还包括等距长方格式、180-3D格式和圆柱图格式。如果该当前全方向图像集合是该等距长方投影格式,则将该当前全方向图像集合转换成该立方方面格式,并且通过将转换的当前全方向图像集合视为具有该立方方面格式,该一个或多个映射语法元素被标识用于该转换的当前全方向图像集合。

[0013] 映射语法元素可包括与该当前立方体贴图图像相关联的当前立方体类型,并且该当前立方体类型属于由1x6立方体贴图布局、2x3立方体贴图布局、3x2立方体贴图布局和6x1立方体贴图布局组成的当前输出布局格式集合。映射语法元素可还包括布局映射索引,其中每个布局映射索引将该当前全方向图像集合的一个立方方面与该当前立方体贴图图像的一个位置相关联。对于该当前全方向图像集合的每个立方方面,除了该当前全方向图像集

合的最后一个立方面之外,标识一个布局映射索引。每个布局映射索引使用具有等于要映射的剩余立方面数的条目数的代码表进行编解码。在另一实施例中,每个布局映射索引使用具有等于要映射的立方面数的条目数的代码表进行编解码。

[0014] 在另一实施例中,映射语法元素还包括旋转索引,其中每个旋转索引指示该当前全方向图像集合的一个立方面在该当前立方体贴图图像的所述一个位置处的旋转角度。针对该当前全方向图像集合的每个立方面,标识一个旋转索引。每个旋转索引使用代码表进行编解码,以指示从对应于{0和90}、{0,+90,-90和180}或{0,90,180和270}旋转角度集合中选择一个旋转角度。

[0015] 映射语法元素还包括默认布局标志,用于指示具有该当前立方体类型的该当前全方向图像集合是否使用默认立方体贴图图像,并且其中仅当该默认布局标志指示该默认立方体贴图图像未被用于该当前全方向图像集合时,布局映射索引和旋转索引被标识用于该当前全方向图像集合。若该默认布局标志指示该默认立方体贴图图像被用于该当前全方向图像集合时,默认布局映射索引和默认旋转索引被用于该当前全方向图像集合。

[0016] 用于该当前全方向图像集合的该输出布局格式集合可以以序列级别、视图级别、图像级别、切片级别、序列参数集合、视频参数集合、或应用参数集合标识在360度全景视频序列的比特流中。映射语法元素以序列级别、视图级别、图像级别、切片级别、序列参数集、视频参数集、或应用参数集标识在360度全景视频序列的比特流中。

[0017] 映射语法元素可基于一个或多个参考映射语法元素被预测地标识。在一个实施例中,一个或多个参考映射语法元素的多个集合以序列级别、视图级别、或图像级别标识在用于360度全景视频序列的比特流中,并且在切片级别或图像级别中标识标志,以从该当前全方向图像集合的一个或多个参考映射语法元素的该多个集合中选择所述一个或多个映射语法元素。在另一个实施例中,参考映射语法元素由来自先前图片、切片或帧的一个或多个第一映射语法元素预测。

【附图说明】

[0018] 图1示出了等距长方投影格式中的全向视频(称为“Hangpai_2”)的图像的示例。

[0019] 图2展示了包含a)立方体贴图、(b) Cubemap_32、(c) Cubemap_180、(d) Plane_poles、(e) Plane_poles_6、(f) Plane_poles_cubemap、(g) Plane_cubemap、(h) Plane_cubemap_32、(i) Flat_fixed、(j) 180degree 3D视频(即,180-3D)以及(k)圆柱图/圆柱形的不同格式的输出布局格式。

[0020] 图3示出了使用投影转换将等距长方投影格式转换为立方格式的示例,其中标记为1至6的图像对应于用于表示360度视频的立方体的六个面上的图像。

[0021] 图4中示出了四个常用布局:1x6-布局410,2x3-布局420,3x2-布局430和6x1-布局440。

[0022] 图5中示出了等距长方格式和立方体格式的几何比较。

[0023] 图6示出了将六个立方面映射到1x6布局的示例。

[0024] 图7示出了将六个立方面映射到2x3布局的示例。

[0025] 图8示出了将六个立方面分配到1x6布局的示例,其中(a)步骤1中面#5分配到布局的第一位置,(b)步骤2中面#4分配到布局的第二位置,(c)步骤3中面#6分配到布局的第三

位置。

[0026] 图9示出了将六个立方面分配到1x6布局的示例,其中将具有(-90度旋转)的立方面#1分配给布局中的第四位置的示例。

[0027] 图10示出了旋转预测的示例,其中参考布局用于预测目标输出布局。

[0028] 图11示出了立方面索引的六个立方体的表示和相应的等距长方图像的示例。

[0029] 图12中示出了用于1x6布局、2x3布局、3x2布局和6x1布局中的每个立方体布局中的位置顺序。

[0030] 图13示出了六个立方面和目标3x2立方体布局的默认布局的示例。

[0031] 图14示出了立方体的默认相对位置及立方面索引的索引和对应的等距长方。

[0032] 图15中示出了1x6布局、2x3布局、3x2布局和6x1布局中的每个立方体布局中的位置的示例性顺序。

[0033] 图16示出了(a)不同的6x1立方体布局,(b)不同的3x2立方体布局,(c)不同的2x3立方体布局,和(d)不同的1x6立方体布局的其他预定义布局。

[0034] 图17示出了根据本发明的实施例的系统处理全向图像的示例性流程图。

【具体实施方式】

[0035] 以下描述是实现本发明的最佳设想模式。这一描述是为了说明本发明的一般原理,而不应被认为是限制性的。本发明的范围最好通过所附权利要求书来确定。

[0036] 在本发明的一个方面,预定义了一组不同的输出布局,并且在比特流中以序列级、视图级、画面级、切片(slice)级别、序列参数集(sequence parameter set,简称为SPS)、视频参数集(video parameter set,简称为VPS)或应用参数集(application parameter set,简称为APS)发送显式标志(explicit flag),以从不同的输出布局集中选择输出布局。例如,不同输出布局的集合可以包括从包括立体图布局、cubemap_32布局、cubemap_180布局、plane_poles布局、plane_poles_6布局、plane_poles_cubemap布局、plane_cubemap布局、plane_cubemap_32布局、flat_fixed布局、cubemap_1x6布局、cubemap_2x3布局、cubemap_3x2布局、和cubemap_6x1布局的组中选择的至少两种输出布局格式。

[0037] 在另一示例中,输出布局格式的集合仅包括cubemap_1x6布局、cubemap_2x3布局、cubemap_3x2布局、和cubemap_6x1布局。传输标志以选择cubemap_1x6布局、cubemap_2x3布局、cubemap_3x2布局、和cubemap_6x1布局中的输出布局。

[0038] 图6和图7示出了将六个立方面映射到两个可能的布局的示例,其中六个立方面映射到图6中的1x6布局,且六个立方面映射到图7中的2x3布局。

[0039] 根据本发明的一种方法,可以根据以下步骤将六个立方面分配给可能的布局:

[0040] 步骤1:发送(在编码器侧)/接收(在解码器侧)标志,以将六个立方面中的一个分配给布局中的第一位置。图8a示出了将面#5分配到1x6布局的第一位置的示例。

[0041] 步骤2:发送/接收标志以将剩余的五个立方面中的一个分配给布局中的第二位置。图8b示出了将面#4分配给1x6布局的第二位置的示例。

[0042] 步骤3:发送/接收标志以将剩余的四个立方面中的一个分配给布局中的第三位置。图8c示出了将面#6分配给1x6布局的第三位置的示例。

[0043] 步骤4:发送/接收标志以将剩余的三个立方面中的一个分配给布局中的第四位

置。

[0044] 步骤5:发送/接收标志以将剩余的两个立方面中的一个分配给布局中的第五位置。

[0045] 步骤6:将最后剩余的面分配给布局中的最终位置。由于只有一个剩余的立方面,所以不需要用标识最后一个标志。

[0046] 在上述提出的方法中,可以使用如表1所示的截断的一元代码(Truncated Unary code)来发送标志。在步骤1中,使用尺寸为6的截断的一元代码对第一标记进行编码,以选择六个立方面中的一个。在步骤2中,使用具有大小为5的截断的一元代码对第二标记进行编码,以选择剩下的五个立方体中的一个,等等。

[0047] 表1.

n (非 负)	n (严 格为正)	截断的 一元代码	可选 地
0	1	0	1
1	2	10	01
2	3	110	001
3	4	111 0	000 1
4	5	111 10	000 01
5	6	111 11	000 00

[0048] [0049] 另一个实施例中,面的旋转也被定义用于将立方面映射到一个输出布

[0050] 局。例如,根据以下步骤将六个立方面分配给可能的布局:

[0051] 步骤1:发送(在编码器侧)/接收(在解码器侧)标志,以将六个立方

[0052] 面中的一个分配给布局中的第一位置。发送/接收另一个标志来定义所选面的旋转。

[0053] 步骤n (n从2到5):发送/接收标志以将剩余的(6-n+1)个立方面中的一个分配给布局中的第n个位置。发送/接收另一个标志以定义所选面的旋转。

[0054] 步骤6:将最后剩余的面分配给布局中的最终位置。发送/接收另一个标志来定义所选面的旋转。

[0055] 这六个面的旋转也可以在六个面被分配之后被传送,如下面的步骤所示:

[0056] 步骤1-a:发送(在编码器侧)/接收(在解码器侧)标志,以将六个立方面中的一个分配给布局中的第一位置。

[0057] 步骤n-a (n从2到5):发送/接收标志以将剩余的(6-n+1)个立方面中的一个分配给布局中的第n个位置。

[0058] 步骤6-a:将最后剩余的面分配给布局中的最终位置。

[0059] 步骤n-b (n从1到6):发送/接收标志以定义第n个面的旋转。

[0060] 另一方面,面的分配和该面的旋转也可以组合为一个标志,如以下步骤所示:

[0061] 步骤1:发送(在编码器侧)/接收(在解码器侧)标志以分配六个立方面中的一个,及其旋转(方向)到布局中的第一位置。

[0062] 步骤n (n从2到5):发送/接收标志以将余下的(6-n+1)个立方面中的一个及其旋转(方向)分配给它的布局中的第n个位置。

[0063] 步骤6:将最后剩余的面分配给布局中的最终位置。发送/接收一个标志来定义第六个面的旋转。

[0064] 面的旋转可以从集合{0度,90度}中选择。在另一个实施例中,面的旋转也可以从集合{0度,90度,180度和270度}或{0度,90度,-90度,180度}中选择。例如,在步骤4中,发送或接收标志,以将旋转过的立方面#1分配给布局中的第四位置。在该示例中,在步骤1至3之后,剩余的面是{面#1,面#2和面#3}。基于以下截断的一元表,使用代码“0”发送索引0(非负)/索引1(严格为正),以将面#1分配给表2所示的第四位置,以及将表3所示的旋转代码“10”(或表4所示的旋转代码“1”)分配给定义的-90度旋转。

[0065] 表2.

n(非负)	n(严格为正)	截断的一元代码	可选地
0	1	0	1
1	2	10	01
2	3	11	00

[0067] 表3.

	n (非 负)	旋转角 度类型 1	截断 的一元代 码	可选 地
[0068]	0	0 度	0	1
	1	-90 度	10	01
	2	+90 度	110	001
	3	+180 度	111	000

[0069] 表4.

	n (非 负)	旋转角 度类型 1	截断 的一元代 码	可选 地
[0070]	0	0 度	0	1
	1	-90 度	1	0

[0071] 图9示出了将具有(-90度旋转)的立方面#1分配给布局中的第四位置的示例。

[0072] 可以在比特流中以序列级别、视图级别、图像级别、切片(slice)级别、SPS(序列参数集)、VPS(视频参数集)或者APS(适应参数集)级别发送上述提出的将等距长方格式标志到布局的方法。在另一个实施例中,如下所示,在序列级别、视图级别、图像级别、SPS、VPS或APS中标志N组对输出布局的等距长方的映射格式:

[0073] 映射格式:

[0074] 索引1: {面#1(旋转角度),面#2(旋转角度),面#3(旋转角度),面#4(旋转角度),面#5(旋转角度),面#6(旋转角度)}

[0075] 索引2: {面#1(旋转角度),面#2(旋转角度),面#3(旋转角度),面#5(旋转角度),面#4(旋转角度),面#6(旋转角度)}

[0076]

[0077] 索引N: {面#2(旋转角度),面#3(旋转角度),面#1(旋转角度),面#5(旋转角度),面#4(旋转角度),面#6(旋转角度)}

[0078] 然后在切片或图像级别中进一步发送标志,以从{索引1,索引2,...,索引N}中选择一种映射格式。

[0079] 在另一个实施例中,可以从另一映射格式预测映射格式。例如,可以从在序列级别、视图级别或图像级别中发送的N组映射格式中选择随后的参考映射格式。在又一个实施

例中,来自先前图片/切片/帧的映射格式可以用作参考映射格式以预测当前的映射格式。参考映射格式{面#1,面#2,面#3,面#4,面#5,面#6}是已知的,用于预测当前的映射格式。

[0080] 假设目标映射格式为{面#1,面#2,面#4,面#3,面#5,面#6},为从{面#1,面#2,面#3,面#4,面#5,面#6}预测目标映射格式,预测演算法可以说明如下:

[0081] 步骤1:按顺序的剩余面是:{面#1,面#2,面#3,面#4,面#5,面#6}。索引0被传送来决定布局{面#1,---,---,---,---,---}中的第一位置中的选定面#1。

[0082] 步骤2:按顺序的剩余面是:{面#2,面#3,面#4,面#5,面#6}。索引0被传送来决定布局{面#1,面#2,---,---,---,---}中的第二位置中的选定面#2。

[0083] 步骤3:按顺序的剩余面是:{面#3,面#4,面#5,面#6}。索引1被传送来决定布局{面#1,面#2,面#4,---,---,---}中的第三位置中的选定面#4。

[0084] 步骤4:按顺序的剩余面是:{面#3,面#5,面#6}。索引0被传送来决定布局{面#1,面#2,面#4,面#3,---,---}中的第四位置中的选定面#3。

[0085] 步骤5:按顺序的剩余面是:{面#5,面#6}。索引0被传送来决定布局{面#1,面#2,面#4,面#3,面#5,---}中的第五位置中的选定面#5。

[0086] 步骤6:剩余面为{面#6},其被选定为布局{面#1,面#2,面#4,面#3,面#5,面#6}的最终的面。

[0087] 也可以从参考布局来预测旋转。图10示出了旋转预测的示例,其中图像1010对应于六个立方面,图像1020对应于参考布局,图像1030对应于目标布局。在图10中,旋转目标布局中的前三个位置。

[0088] 在上述示例中,以下步骤可被用于从参考布局预测目标布局。在这个示例中,参考布局是{面#1(0),面#2(0),面#3(0),面#4(0),面#5(0),面#6(0)},目标布局为{面#1(-90),面#2(-90),面#3(-90),面#5(0),面#4(0),面#6(0)},其中(0)表示0度,(-90)表示-90度。

[0089] 步骤1:按顺序的剩余面是:{面#1(0),面#2(0),面#3(0),面#4(0),面#5(0),面#6(0)}。索引0被传送到布局{面#1,---,---,---,---,---}中的第一个位置中的选定面#1。如表5所示,大小为6的截断的一元代码可用于编码索引0,其中所选代码的行以粗斜体字体(bold italic font)表示。进一步发送角度差(-90),最终角度重构为:参考角度+角度差=(0)+(-90)=(-90)。可以使用表6中的旋转编码类型1或表7中的旋转编码类型2(表4)对角度差(-90)进行编码,其中所选代码行以粗斜体字体表示。

[0090] 表5.

	n (非 负)	n (严 格为正)	截断 的一元代 码	可选 地
	0	1	0	1
[0091]	1	2	10	01
	2	3	110	001
	3	4	1110	0001
	4	5	11110	00001
	5	6	11111	00000

[0092] 表6.

	n (非 负)	旋转角 度类型 1	截断 的一元代 码	可选 地
	0	0 度	0	1
[0093]	1	-90 度	10	01
	2	+90 度	110	001
	3	+180 度	111	000

[0094] 表7.

	n (非 负)	旋转角 度类型 1	截断 的一元代 码	可选 地
	0	0 度	0	1
[0095]	1	-90 度	1	0

[0096] 步骤2:按照顺序的剩余面的集合是: {面#2 (0), 面#3 (0), 面#4 (0), 面#5 (0), 面#6 (0)}。索引0被传送到布局 {面#1, 面#2, ---, ---, ---, ---} 中的第二位置中的选定面#2。角

度差(-90)进一步传递。在这种情况下,可以使用如表8所示的大小为5的截断的一元代码来编码索引0,其中所选代码的行以粗斜体字体表示。

[0097] 表8.

	n (非 负)	n (严 格为正)	截断 的一元代 码	可选 地
[0098]	<i>0</i>	<i>1</i>	<i>0</i>	<i>1</i>
	1	2	10	01
	2	3	110	001
	3	4	1110	0001
	4	5	1111	0000

[0099] 步骤3:按顺序排列的剩余面的集合是:{面#3(0),面#4(0),面#5(0),面#6(0)}。索引0被传送到布局{面#1,面#2,面#3,---,---,---}中的第3位置的选定面#3。进一步传递角度差(-90)。在这种情况下,可以使用如表9所示的大小为4的截断的一元代码来编码索引0,其中所选代码行以粗斜体字体表示。

[0100] 表9.

	n (非 负)	n (严 格为正)	截断 的一元代 码	可选 地
[0101]	<i>0</i>	<i>1</i>	<i>0</i>	<i>1</i>
	1	2	10	01
	2	3	110	001
	3	4	111	000

[0102] 步骤4:按照顺序的剩余面的集合是:{面#4(0),面#5(0),面#6(0)}。索引1被发送到布局{面#1,面#2,面#3,面#5,---,---}中的第四位置的选定面#5。进一步发送角度差(0)。在这种情况下,可以使用如表10所示的尺寸为3的截断的一元代码来编码索引1,其中所选代码行以粗斜体字体表示。

[0103] 表10.

	n (非 负)	n (严 格为正)	截断 的一元代 码	可选 地
[0104]	0	1	0	1
	<i>1</i>	<i>2</i>	<i>10</i>	<i>01</i>
	2	3	11	00

[0105] 步骤5:按顺序的剩余面的集合是: {面#4 (0), 面#6 (0)}。索引0被发送到布局 {面#1, 面#2, 面#3, 面#5, 面#4, ...} 中的第5位置的选定面#4。进一步发送角度差 (0)。在这种情况下,可以使用如表11所示的大小为2的截断的一元代码来编码索引0,其中所选代码的行以粗斜体字体表示。

[0106] 表11.

	n (非 负)	n (严 格为正)	截断 的一元代 码	可选 地
[0107]	0	1	0	1
[0108]	<i>1</i>	<i>2</i>	<i>1</i>	<i>0</i>

[0109] 步骤6:剩下的面是 {面#6} 并被选择为布局 {面#1, 面#2, 面#4, 面#3, 面#5, 面#6} 中的最后位置。进一步发送角度差 (0)。

[0110] 例如,可以首先在图片/切片/帧级别中发送索引,以选择一个映射布局作为参考布局。然后,预测方法进一步用于从参考布局预测当前图片/切片/帧的目标布局。在另一示例中,参考布局可以从先前编码的图像/切片/帧继承,并且然后应用预测方法来预测当前图像/切片/帧的目标布局。

[0111] 在上述提出的方法中,截断的一元代码也可以由其他熵编解码方法代替。例如,以下熵编解码方法之一可以应用于本发明。

[0112] -ae (v) :上下文自适应算术熵编解码语法元素。

[0113] -b (8) :具有任何比特串 (8比特) 模式的比特组 (bit string)。该描述符的解析过程由函数read_bits (8) 的返回值指定。

[0114] -f (n) :使用具有最先写入左边比特 (从左到右) 的n比特写入的固定模式比特串。该描述符的解析过程由函数read_bits (n) 的返回值指定。

[0115] -se (v) :带符号整数的0阶指数哥伦布编解码语法元素,左边比特是第一个。

[0116] -u (n) :使用n比特的无符号整数。当n在语法表中为“v”时,比特数以取决于其他语

法元素的值的方式变化。该描述符的解析过程由函数read_bits(n)的返回值指定,解释为无符号整数的二进制表示,最高有效比特首先写入。

[0117] -ue(v):无符号整数0阶指数哥伦布编解码语法元素,左边比特是第一个。

[0118] 根据本发明的实施例的用于布局信令的示例性语法设计如下所示:

[0119] 表12.

	layout_mapping_extension() {	描述符
	...	
[0120]	for (i=0; i<num_of_layout_faces-1; i++) {	
	layout_face[i]	ue(v)
	}	
	...	
	}	

[0121] 在上述语法表中,语法num_of_layout_faces指定布局中面的总数。语法layout_face[i]将剩余(num_of_layout_faces-i)面的索引指定为布局中第i个位置的对应面,layout_face[i]的值应在0到(num_of_layout_faces-i-1)的范围内,包括0和(num_of_layout_faces-i-1)。语法layout_face[num_of_layout_faces-1]被推断为等于最后剩余面。

[0122] 根据本发明的实施例的用于具有旋转的布局信令的示例性语法设计如下所示:

[0123] 表13.

	layout_mapping_extension() {	描述符
	...	
[0124]	for (i=0; i<num_of_layout_faces-1; i++) {	
	layout_face[i]	ue(v)
	layout_rotation[i]	ue(v)

	}	
[0125]	layout_rotation[num_of_layout_faces-1]	ue(v)
	}	

[0126] 在上述语法表中,语法num_of_layout_faces指定剩余num_of_layout_faces-i个面中的索引作为布局中第i个位置的对应面,并且layout_face[i]的值应在0到(num_of_layout_faces-i-1)的范围内,包括0和(num_of_layout_faces-i-1)。语法layout_face[num_of_layout_faces-1]被推断为等于最后剩余面。语法layout_face[i]指定表14或表15中规定的布局中第i个位置的相应面旋转,并且layout_face[i]的值应在0到3(或1),包括0和3(或1)。

[0127] 表14.

layout_rotatio n[i]	旋转角度
0	0
1	-90
2	+90
3	+180

[0129] 表15.

layout_rotatio n[i]	旋转角度
0	0
1	-90

[0131] 根据另一个实施例,用于布局信令的示例性语法设计在表16中示出,其中为这些六立方面的置换和旋转的信息标志附加语法。默认立方面的相对位置。初始立方面阵列等于{Left,Front,Right,Rear,Top,Bottom}。立方面索引和相应的等距长方的标示如图11所示。

[0132] 表15.

	Extension_data(i) {	描述符
	While (next_bits(32) == extension_start_code) {	
	extension_start_code	f(32)
	if (i == 0) {	
	if (next_bits(4) == '0010')	
	sequence_display_extension()	
	else if (next_bits(4) == '0011')	
	temporal_scalability_extension()	
	else if (next_bits(4) == '0100')	
[0133]	copyright_extension()	
	else if (next_bits(4) == '1010')	
	mastering_display_and_content_metadata_extension()	
	else if (next_bits(4) == '1011')	
	camera_parameters_extension()	
	else if (next_bits(4) == '1100')	
	layout_mapping_extension()	
	else	
	...	

[0134] 表16.

[0135]	layout_mapping_extension() {	描述符

	num_of_layout_face_minus1	ue(v)
	for (i=0;	
	i<num_of_layout_face_minus1; i++) {	
	layout_face[i]	ue(v)
[0136]	layout_rotation[i]	ue(v)
	}	
	layout_rotation[num_of_layout_face_min	ue(v)
	us1]	
	}	

[0137] 在上述语法表中,num_of_layout_face_minus1指定布局中面的总数。语法num_of_layout_faces推断为num_of_layout_face_minus1+1。语法layout_face[i]指定剩余(num_of_layout_faces-i)面中的索引作为在布局中的第i个位置的对应face,layout_face[i]的值应在0到(num_of_layout_face_minus1-i) (含0和num_of_layout_face_minus1-i)的范围内。语法layout_face[num_of_layout_face_minus1]被推定为最后的剩余面。语法layout_rotation[i]指定表17中规定的布局中第i个位置的相应面旋转,并且layout_rotation[i]的值应在0到3的范围内,包括0和3。

[0138] 表17.

layout_rot ation[i]	旋转角度 (逆时针)
0	0
1	-90
2	+90
3	+180

[0140] 图12中示出了用于1x6布局(1210)、2x3布局(1220)、3x2布局(1230)和6x1布局(1240)中的每个立方体布局中的位置顺序(即layout_face[i])。图13示出了六个立方体的默认布局(1310)和目标3x2立方体布局(1320)的示例。

[0141] 根据上述实施例的标志立方体布局的示例在下表中示出。对于i=0、1和2,前三个布局面选择对应于列表中的第一个位置(即0 {Left}, 0 {Front} 和0 {Right})。因此,对于i=

0、1和2,选择索引0。对于*i* = 3,选择列表中的第二位置(即,1 {Top})。因此,标志索引1。对于*i* = 4,选择列表中的第二位置(即0 {Rear})。因此,标志索引1。由于只剩下一个立方面,所以不需要为最后一个立方面布局标志索引。但是,最后一个立方面(即*i* = 5)仍然需要layout_rotation[*i*]。

[0142] 表18.

[0143]

步骤#0: num_of_layout_face_minus1 = 5			
步骤#1	立方面阵列中的剩余面	layout_face[i]	layout_rotation[i]
<i>i</i> = 0	{Left(0), Front(1), Right(2), Rear(3), Top(4), Bottom(5)}	0 {Left}	0
<i>i</i> = 1	{Front(0), Right(1), Rear(2), Top(3), Bottom(4)}	0 {Front}	0
<i>i</i> = 2	{Right(0), Rear(1), Top(2), Bottom(3)}	0 {Right}	0
<i>i</i> = 3	{Rear(0), Top(1), Bottom(2)}	1 {Top}	1
<i>i</i> = 4	{Rear(0), Bottom(1)}	0 {Rear}	2
步骤 #2: 布局的最后一个位置是立方面 { Bottom } 和			

[0144]

layout_rotation[num_of_layout_face_minus1] = 1

[0145] 在另一个实施例中,用于布局信令的语法设计包括用于发送这六个立方体的置换和旋转的信息的附加语法。立方体的默认相对位置及其索引如图14所示,其中示出了立方体索引(1410)和对应的等距长方(1420)。

[0146] 根据上述实施例的示例性语法设计在表19a和表19b中示出。在表19a中, mapping_extension() 由斜体文本定义。mapping_extension() 的详细信息如表19b所示。

[0147] 表19a.

		描述符
	Extension_data(i) {	
	While (next_bits(32) == extension_start_code) {	
	extension_start_code	f(32)
	if (i == 0) {	
	if (next_bits(4) == '0010')	
[0148]	sequence_display_extension()	
	else if (next_bits(4) == '0011')	
	temporal_scalability_extension()	
	else if (next_bits(4) == '0100')	
	copyright_extension()	
	else if (next_bits(4) == '1010')	
	mastering_display_and_content_metadata_extension()	
	else if (next_bits(4) == '1011')	
	camera_parameters_extension()	
	else if (next_bits(4) == 'xxxx')	
[0149]	<i>mapping_extension()</i>	
	else	
	...	

[0150] 表19b.

	描述符
mapping_extension() {	
num_of_layout_face_minus1	ue(v)
for (i=0; i<num_of_layout_face_minus1; i++) {	
<i>layout_face[i]</i>	ue(v)
<i>layout_rotation[i]</i>	ue(v)
}	
layout_rotation[num_of_layout_face_minus1]	ue(v)
}	

[0152] 在上表中,语法num_of_layout_face_minus1指定布局中面的总数。语法num_of_layout_face被推断为num_of_layout_face_minus1+1。语法layout_face[i]指定立方面索引作为布局中第i个位置的对应面,layout_face的值[i]的范围为0到(num_of_layout_face_minus1) (含0和(num_of_layout_face_minus1))。语法layout_face[num_of_layout_face_minus1]被推定为最后的剩余面。语法layout_rotation[i]指定表20中规定的布局中第i个位置的相应面旋转,并且layout_rotation[i]的值应在0到3的范围内,包括0和3。

[0153] 表20.

layout_rotation[i]	旋转角度 (逆时针)
0	0
1	-90
2	+90
3	+180

[0155] 图15中示出了1x6布局(1510)、2x3布局(1520)、3x2布局(1530)和6x1布局(1540)中的每个立方体布局中的位置(即layout_面[i])的另一示例性顺序。

[0156] 根据上述实施例的另一示例性语法设计在表21中示出。

[0157] 表21.

	mapping_extension() {	描述符
	mapping_format	ue(v)
	if (mapping_format == 'CUBICface') {	
[0158]	cubic_type	ue(v)
	default_layout_flag	u(1)
	if(!default_layout_flag) {	
	for (i=0; i<5; i++) {	
	layout_face[i]	ue(v)
	layout_rotation[i]	ue(v)

	}	
	layout_rotation[5]	ue(v)
	}	
	}	
	else if (mapping_format == 'EQUIRECTANGULAR')	
	...	
[0159]	else if (mapping_format == '180-3D')	
	...	
	else if (mapping_format == 'CYLINDERMAP')	
	...	
	}	

[0160] 在上表中,mapping_format指定如下表所示的360度视频的输入映射格式,其中mapping_format的值应在0到3的范围内,包括0和3。

[0161] 表22.

值	描述符
0	CUBICFACE
1	EQUIRECTANGULAR
2	180-3D
3	CYLINDERMAP

[0163] 在表21中,cubic_type指定如表23所示的立方体格式的立方体类型,并且cubic_

type的值应在0至3的范围内,包括0和3。

[0164] 表23.

值	描述符
0	1x6 立方体 格式
1	2x3 立方体 格式
2	3x2 立方体 格式
3	6x1 立方体 格式

[0166] 语法default_layout_flag等于1表示立方体布局遵循表24中指定的默认布局。语法default_layout_flag等于0表示默认立方体布局不存在,并且显式地标志每个位置的相应面和旋转。表24指定了默认布局。根据图12所示的顺序,立体布局中每个位置的相应面和旋转指定如下。图11中定义了从等距长方到立方体的映射。

[0167] 表24.

[0168]

Cubic_type	默认布局
1x6 立方体 格式	{Left(-90 ⁰), Front(-90 ⁰), Right(-90 ⁰), Top(+180 ⁰), Rear(0 ⁰), Bottom(+180 ⁰)}
2x3 立方体格 式	{Left(-90 ⁰), Top(+180 ⁰), Front(-90 ⁰), Rear(0 ⁰), Right(-90 ⁰), Bottom(+180 ⁰)}
3x2 立方体格 式	{Left(0 ⁰), Front(0 ⁰), Right(0 ⁰), Top(-90 ⁰), Rear(+90 ⁰), Bottom(-90 ⁰)}
6x1 立方体格 式	{Left(0 ⁰), Front(0 ⁰), Right(0 ⁰), Top(-90 ⁰), Rear(+90 ⁰), Bottom(-90 ⁰)}

[0169] 在表21中,layout_face[i]将剩余面中的索引指定为布局中第i个位置的对应面。根据图12所示的顺序,每个位置的面和旋转(即,layout_face[i]和layout_rotation[i])

如下迭代地标志 (signaled iteratively)。初始 (面-选择) 阵列包括 {Left, Front, Right, Rear, Top, Bottom} 的顺序的所有六个面。对于第一个位置, 标志 layout_face[0], 以指示阵列中所选面的索引 (即 {Left, Front, Right, Rear, Top, Bottom}) 作为位置0的对应面。layout_face[0] 的值应在0到5的范围内, 包括0和5。所选择的位置0的面随后从剩余候选者的阵列中移除。对于第i个位置, 标志 layout_face[i], 以指示由剩余的未选择面组成的更新阵列中的索引作为位置i的相应面。layout_face[i] 的值应在0到5-i的范围内, 包括0和5-i。位置i的对应的选定面随后从余下的候补阵列中移除。对于最后的位置, 它被推断为最后剩下的未选择的面。

[0170] 在表21中, layout_rotation[i] 指定了表20中规定的布局中第i个位置的相应面旋转, 并且 layout_rotation[i] 的值应在0到3的范围内, 包括0和3。

[0171] 根据另一个实施例, 表25中示出了用于布局信令的示例性语法设计, 其中针对这六个立方体的置换和旋转的信息标志附加语法。

[0172] 表25.

	描述符
mapping_extension() {	
mapping_format	u(8)
if (mapping_format == 'CUBICFACE') {	
cubic_type	u(4)
default_layout_flag	u(1)
if(!default_layout_flag) {	
for (i=0; i<6; i++) {	
layout_face[i]	u(3)
layout_rotation[i]	u(2)

[0173]

	}	
	}	
	}	
	else if (mapping_format == 'EQUIRECTANGULAR')	
[0174]	...	
	else if(mapping_format == '180-3D')	
	...	
	else if (mapping_format == 'CYLINDERMAP')	
	...	
	}	

[0175] 在本发明的另一个实施例中,如果选择立方体作为布局格式,则在比特流的序列级别、视图级别、图像级别、切片级别、瓦片 (tile) 级别、SPS、VPS或APS中发送第二标志从一组立方体类型集合中选择一种立方体类型,该立方体类型集合包括1×6立方体格式、2×3立方体格式、3×2立方体格式和6×1立方体格式中的至少两种。

[0176] 在另一个实施例中,在比特流的序列级别,视图级别、图片级别、切片级别、瓦片级别、SPS、VPS或APS中发送标志,以指示是否使用了立方体或其他格式的预定义默认布局。如果不使用立方体或其他格式的预定义默认布局,则布局随后被显示地传输到比特流中。立方体的预定义布局可以定义为表24。

[0177] 也可以使用其他预定义的布局。例如,也可以如图16a所示定义不同的6×1立方体布局。图16b中显示了不同的3x2立方体布局。图16c中显示了两个不同的2x3立方体布局(1610和1611)。图16d中显示了六个不同的1x6立方体布局(1620到1625)。

[0178] 为了明确地标志布局,在比特流中发送与每个相应面相关联的对应面和/或旋转。在一个示例中,初始(面-选择)阵列包括具有给定预定义顺序的 {Left,Front,Right,Rear,Top,Bottom} 中的至少两个。

[0179] 对于第一位置,标志语法以将给定阵列中的选定面的索引指示为第一位置的对应面,并且语法的值应在0至(N-1),其中N表示总的面的数量。

[0180] 对于第i个位置,标志另一个语法来指示由剩余的未选择面组成的更新的阵列中的索引作为位置i的对应面,并且该语法的值应在0至(N-1)的范围内,包括0和(N-1)。对于最后的位置,它被推断为最后剩下的未选择的面。

[0181] 在另一示例中,初始(面-选择)阵列包括具有给定预定义顺序的 {Left,Front,

Right,Rear,Top,Bottom}中的至少两个。

[0182] 对于第一位置,标志语法,以将给定阵列中的选定面的索引指示为第一位置的对应面,并且语法的值应在0至(N-1)的范围内,其中N表示总的面的数量。之后,第一个位置的选定面将从阵列中删除。

[0183] 对于第i个位置,标志另一个语法来指示由剩余未选择面组成的更新阵列中的索引作为位置i的对应面,并且该语法的值应在0到(N-1-i)(含0和(N-1-i))的范围内。随后位置i的对应选定面随后被从阵列中移除。对于最后的位置,它被推断为最后剩下的未选择的面。

[0184] 为了指定布局中的每个位置的对应的面旋转,发送每个位置的语法以指示从一组旋转候选集合(其包括(-90°, +90°, +180°, 0°)中的至少两个)中指示相应的面旋转。旋转索引的规格可以定义如表20所示。

[0185] 图17示出了根据本发明的实施例的系统处理全向图像的示例性流程图。流程图所示的步骤以及本公开的其他流程图可以被实现为在编码器侧和/或解码器侧的一个或多个处理器(例如,一个或多个CPU)上可执行的程式代码。流程图所示的步骤也可以基于诸如被配置为执行流程图中的步骤的一个或多个电子设备或处理器的硬体来实现。根据该方法,在步骤1710中接收使用选择的投影格式的360度全景视频序列中的每个球面图像转换的当前全向图像集合,其中所选择的投影格式属于包括立方格式组的投影格式组,而立方格式的当前组全方位图像由六个立方方面组成。在步骤1720中检查所选择的投影格式是否对应于立方格式。如果所选择的投影格式对应于立方格式,则执行步骤1730和1740。否则(即非立方格式的“否”路径),步骤1730和1740被跳过。在步骤1730中,标志将当前的全方向图像集合映射到属于输出布局格式集合的当前立方体贴图(cubemap)图像中的一个或多个映射语法元素,其中输出布局格式集合包括从包括立方体贴图布局组和等距长方(equi-rectangular)格式的布局组中选择的至少两个输出布局格式。在步骤1740中,为当前的全向图像组提供包含该一个或多个映射语法元素的比特流中的编解码数据。

[0186] 以上所示的流程图旨在作为示例来说明本发明的实施例。本领域技术人员可以通过修改各个步骤、分离或组合步骤来实践本发明,而不脱离本发明的精神。

[0187] 呈现上述描述以使得本领域技术人员能够在特定应用及其要求的上下文中实施本发明。对所描述的实施例的各种修改对于本领域技术人员将是显而易见的,并且本文定义的一般原理可以应用于其他实施例。因此,本发明并不限于所示出和描述的特定实施例,而是符合与本文所公开的原理和颖特征相一致的最宽范围。在上述详细描述中,示出了各种具体细节以便提供对本发明的透彻理解。然而,本领域技术人员将理解,可以实施本发明。

[0188] 如上所述的本发明的实施例可以以各种硬件、软件代码或两者的组合来实现。例如,本发明的实施例可以是集成到视频压缩芯片中的一个或多个电子电路或集成到视频压缩软件中的程序代码,以执行本文所述的处理。本发明的实施例也可以是要在数字信号处理器(DSP)上执行以执行本文所述的处理的程序代码。本发明还可涉及由计算机处理器、数字信号处理器、微处理器或现场可程式化门阵列(FPGA)执行的多个功能。可以通过执行定义本发明所体现的特定方法的机器可读软件代码或固件代码来将这些处理器配置成执行根据本发明的特定任务。软件代码或固件代码可以以不同的程序语言和不同的格式或风格

开发。也可以为不同的目标平台编译软件代码。然而,根据本发明的不同的代码格式,软件代码的样式和语言以及配置代码的其他方式将不会脱离本发明的精神和范围。

[0189] 在不脱离本发明的精神或基本特征的情况下,本发明可以以其他具体形式实施。所描述的例子仅在所有方面被认为是说明性的而不是限制性的。因此,本发明的范围由权利要求而不是前面的描述来指示。属于权利要求范围的等同物的含义和范围内的所有变化将被包括在其范围内。

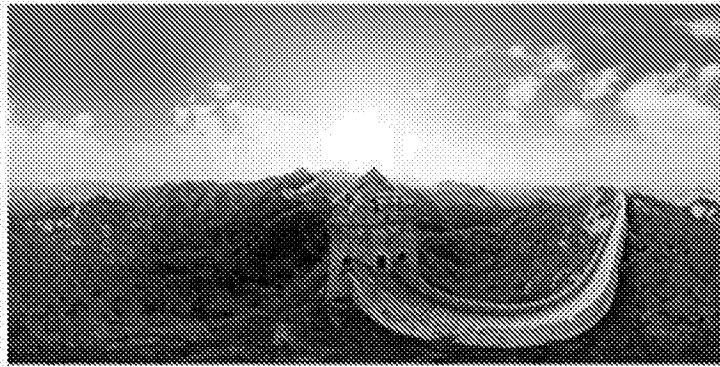


图1

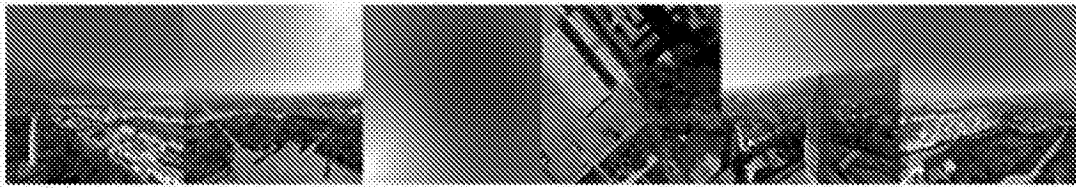


图2a



图2b

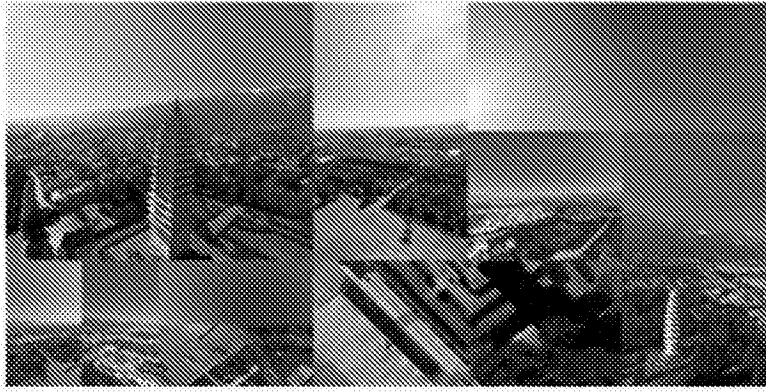


图2c

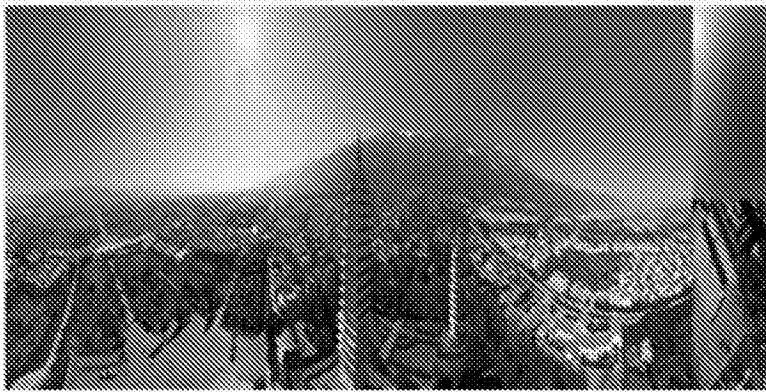


图2d



图2e

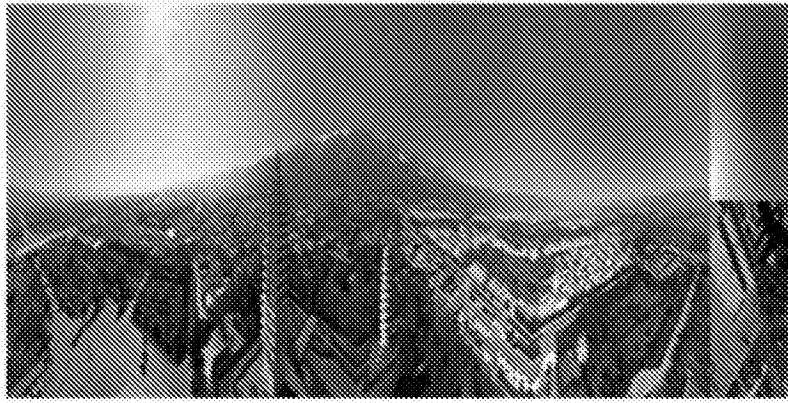


图2f

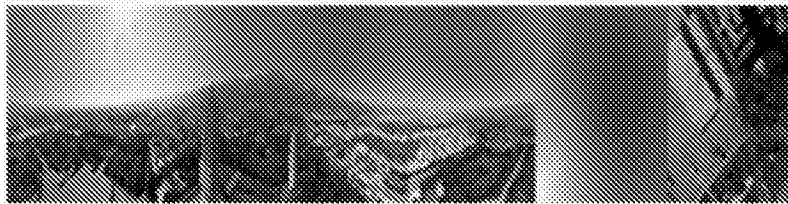


图2g

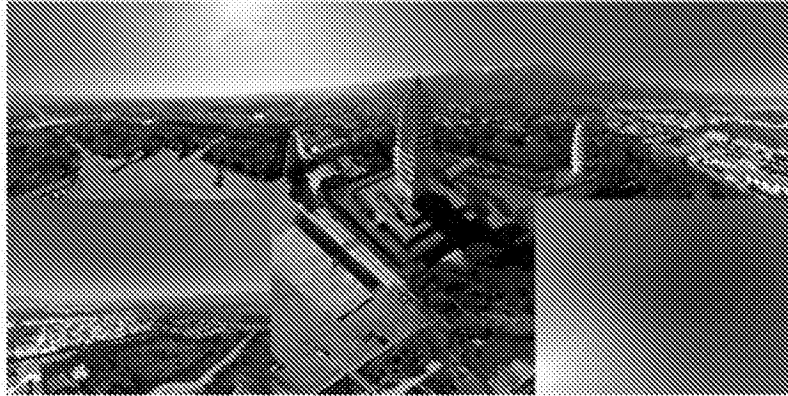


图2h



图2i



图2j

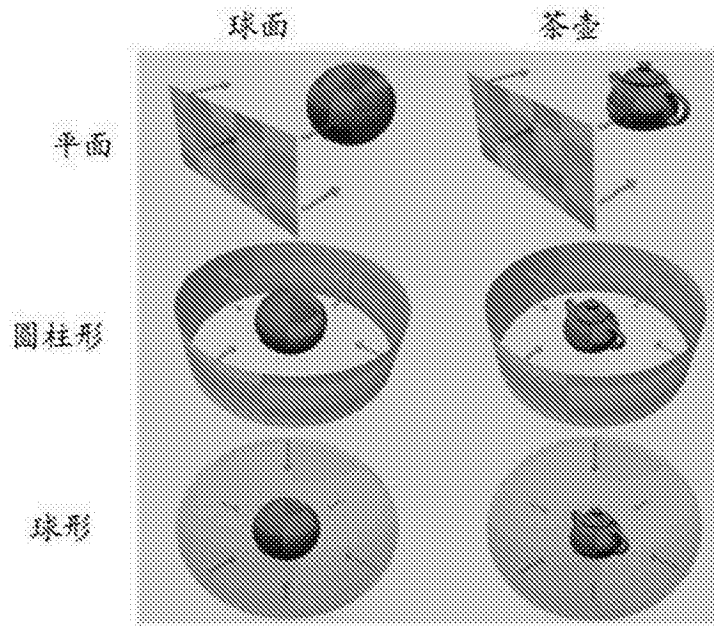


图2k

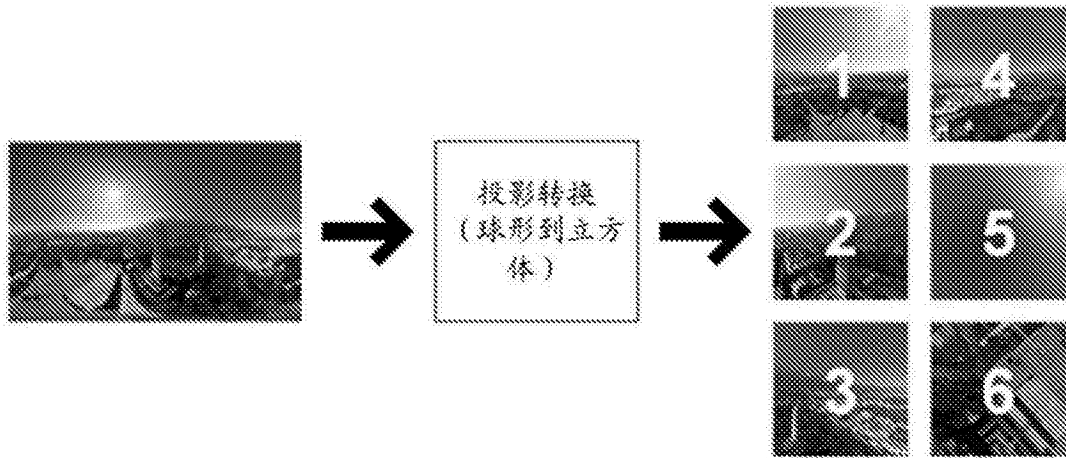


图3

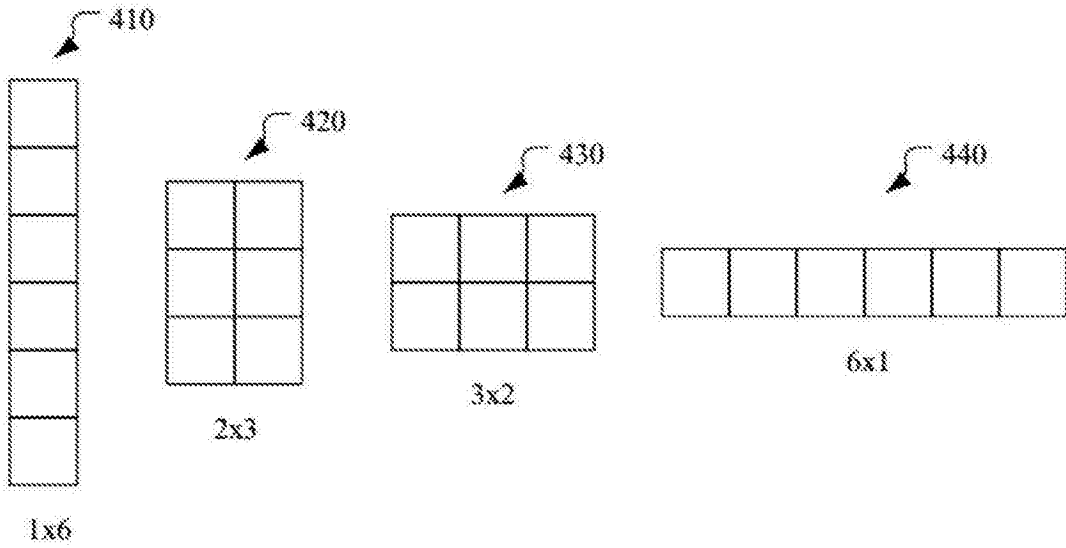


图4

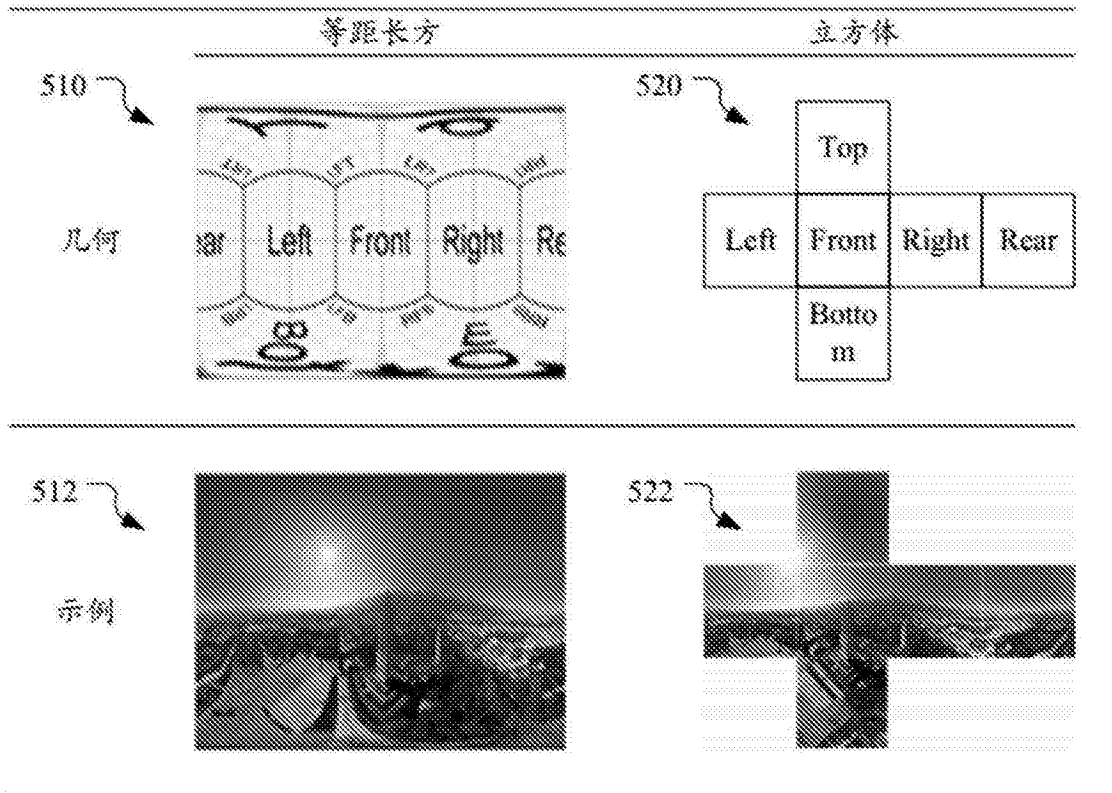


图5

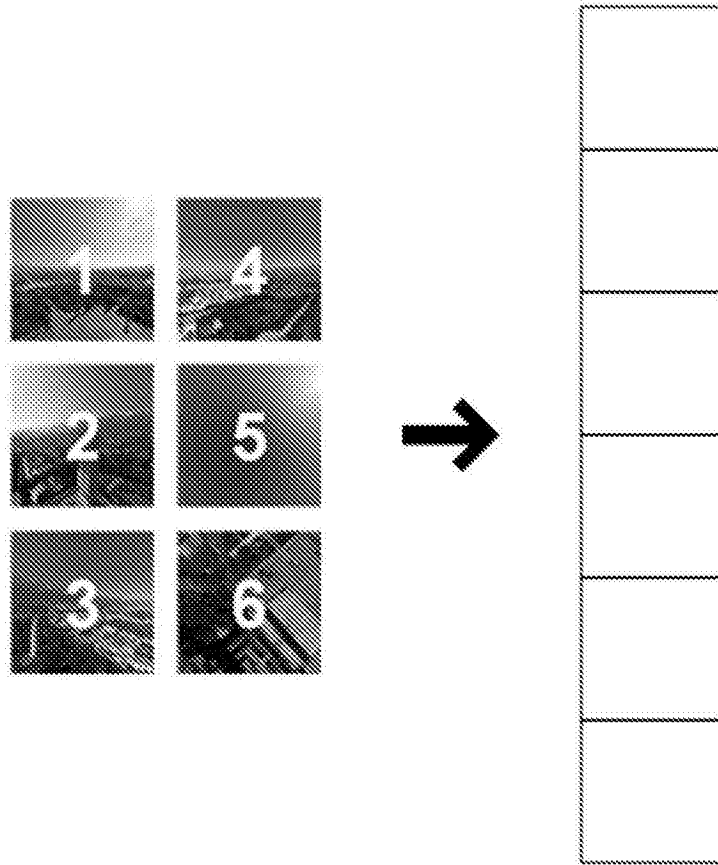


图6

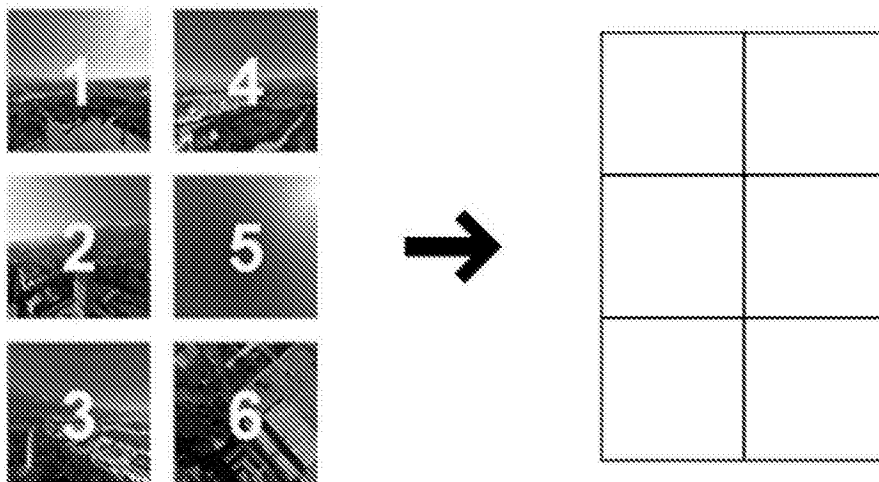


图7

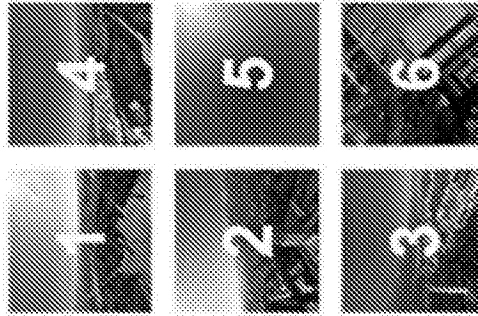


图8a

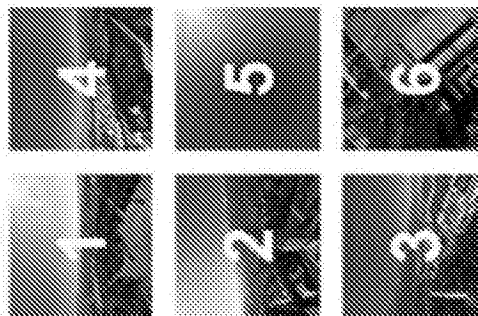


图8b

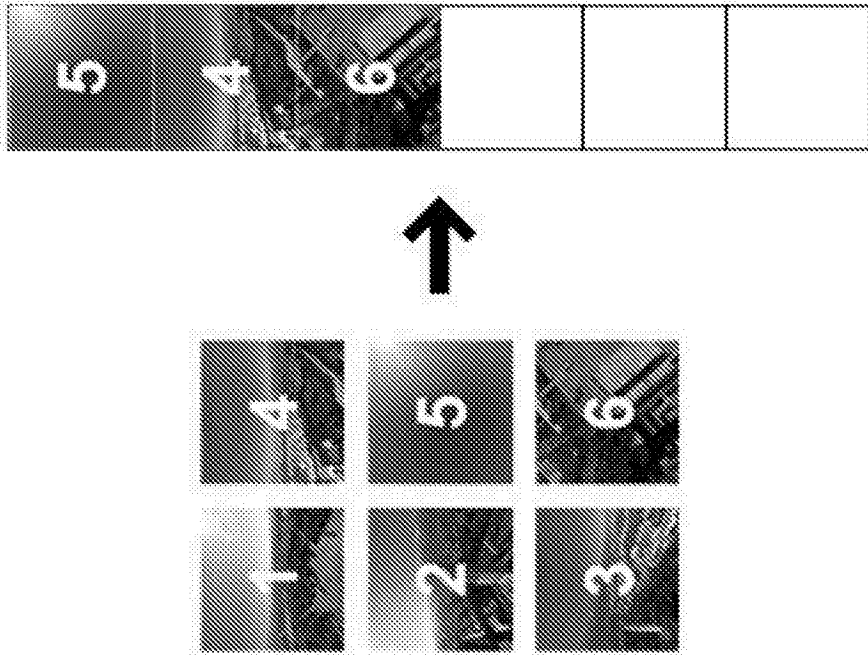


图8c

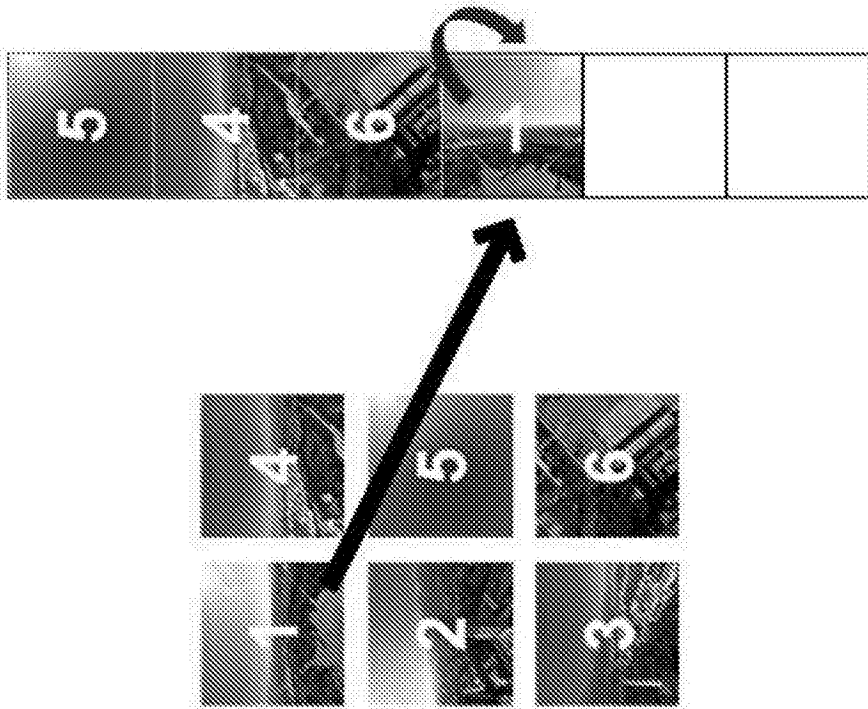


图9

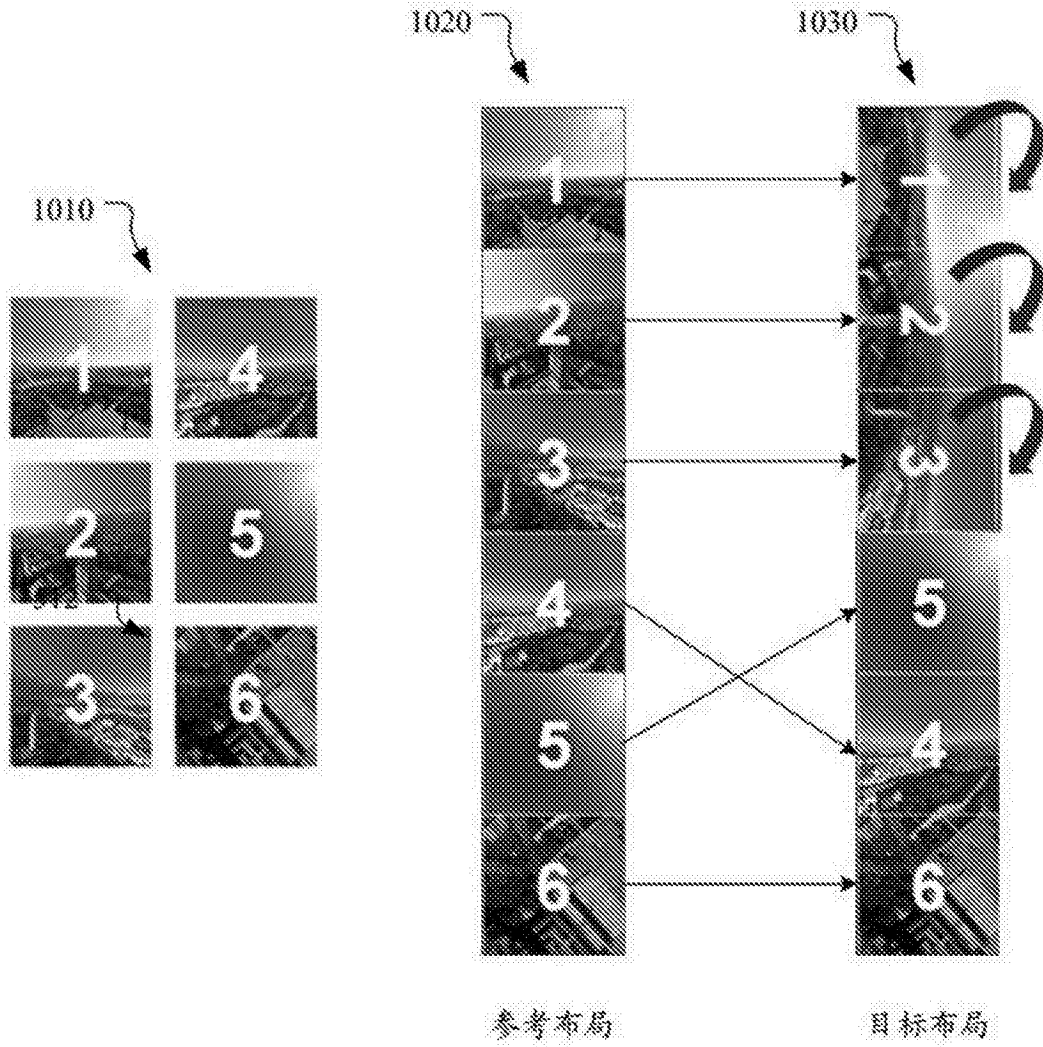


图10

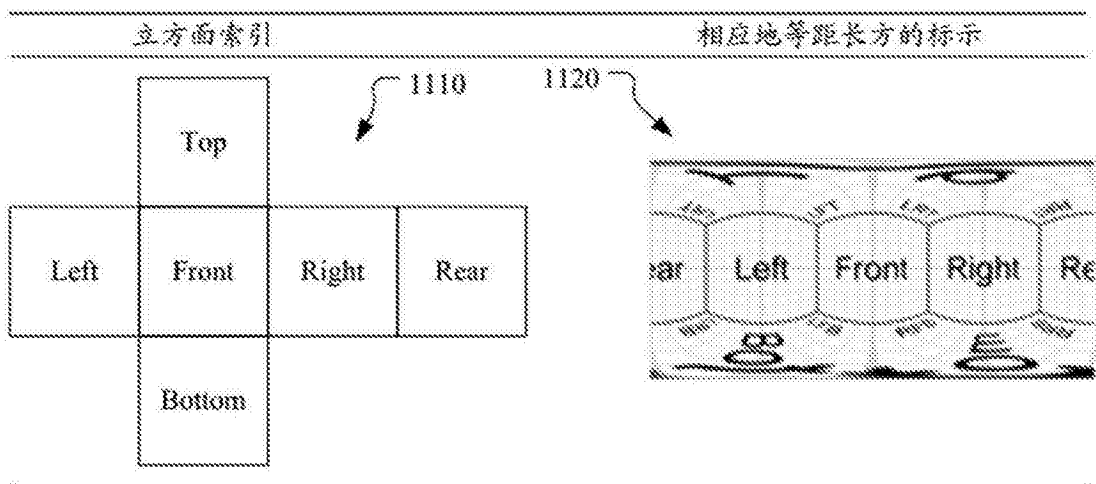


图11

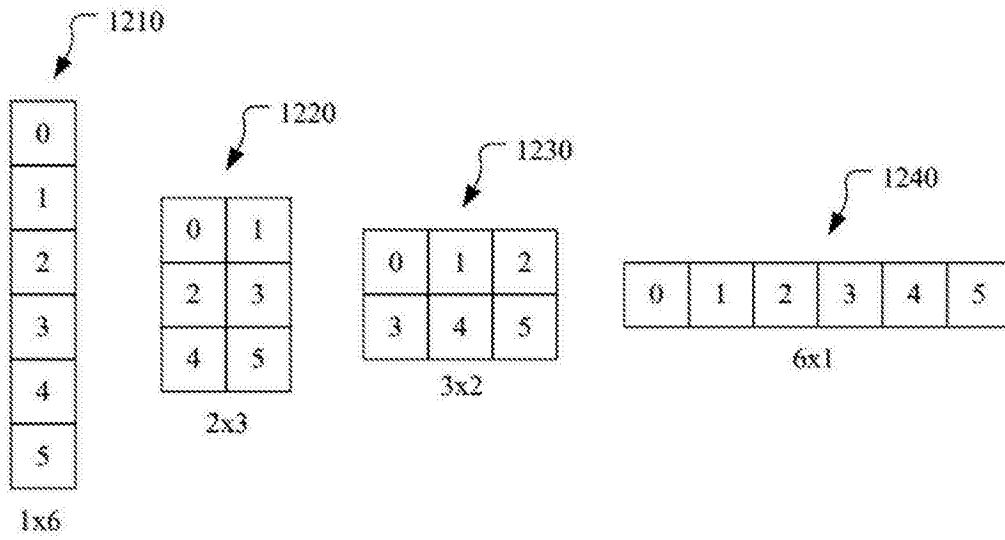


图12

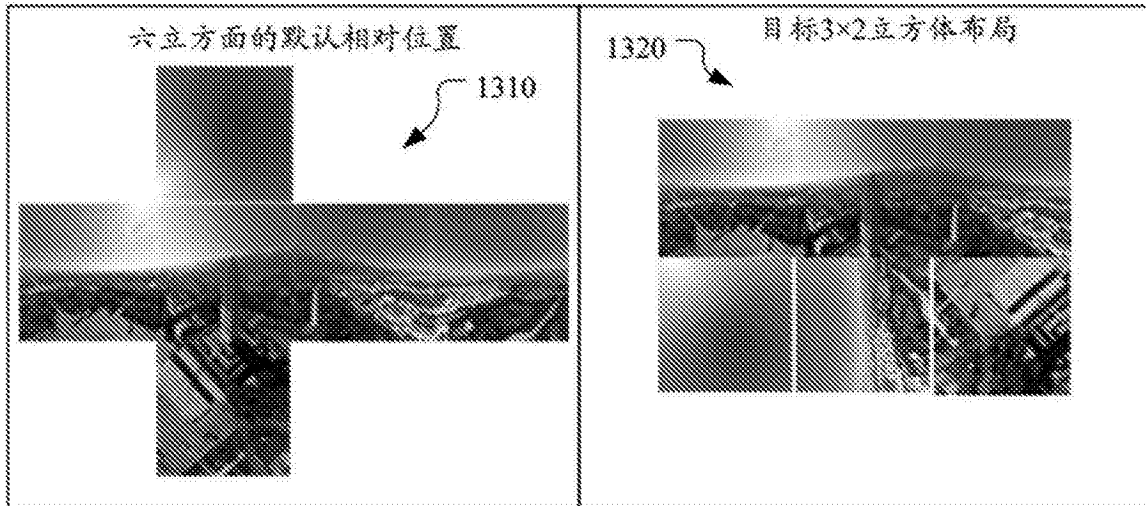


图13

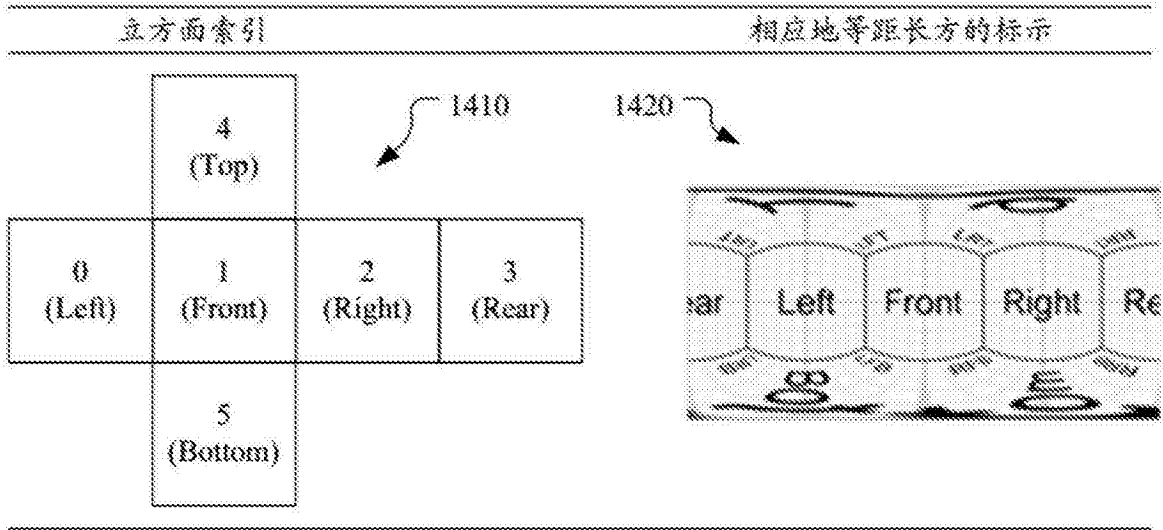


图14

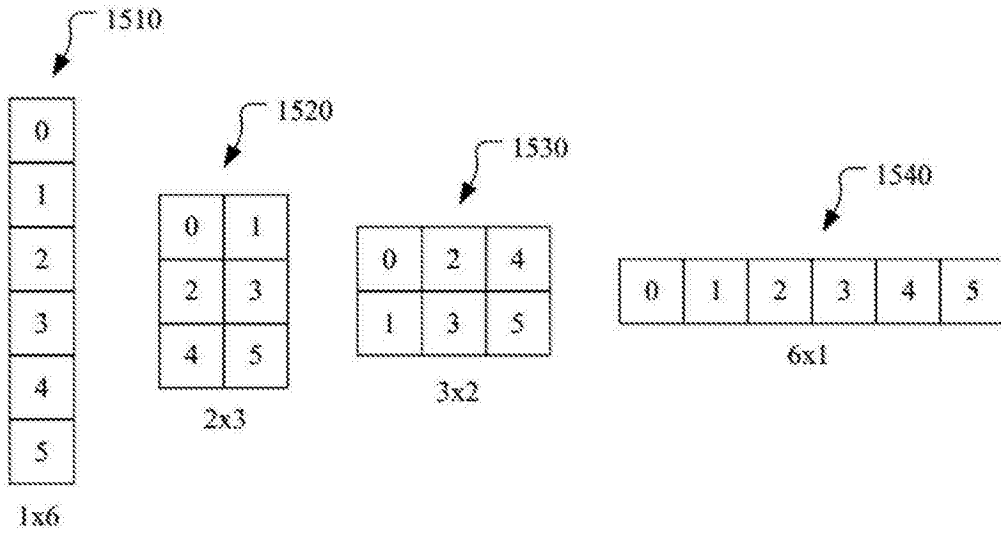


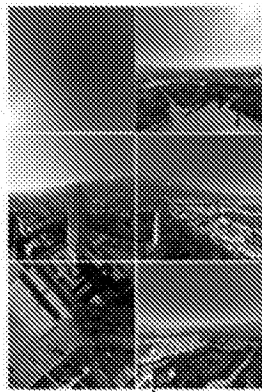
图15



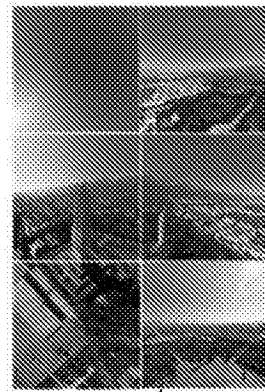
图16a



图16b



1610



1611

图16c

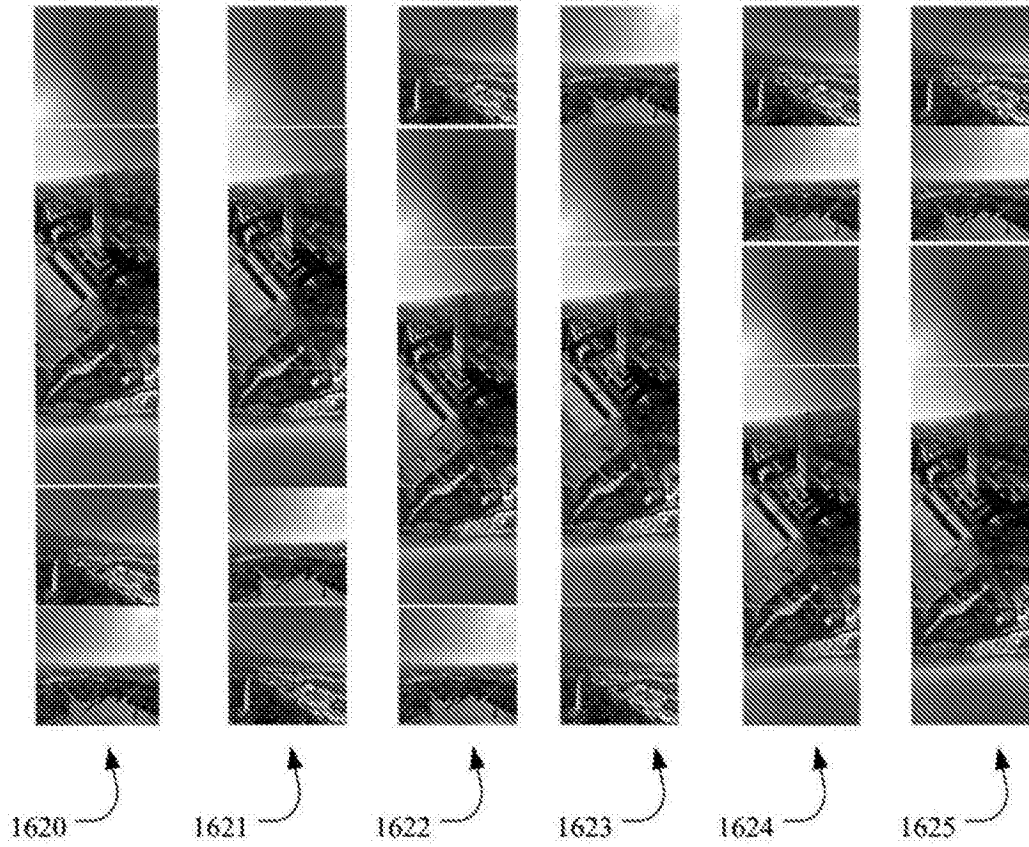


图16d

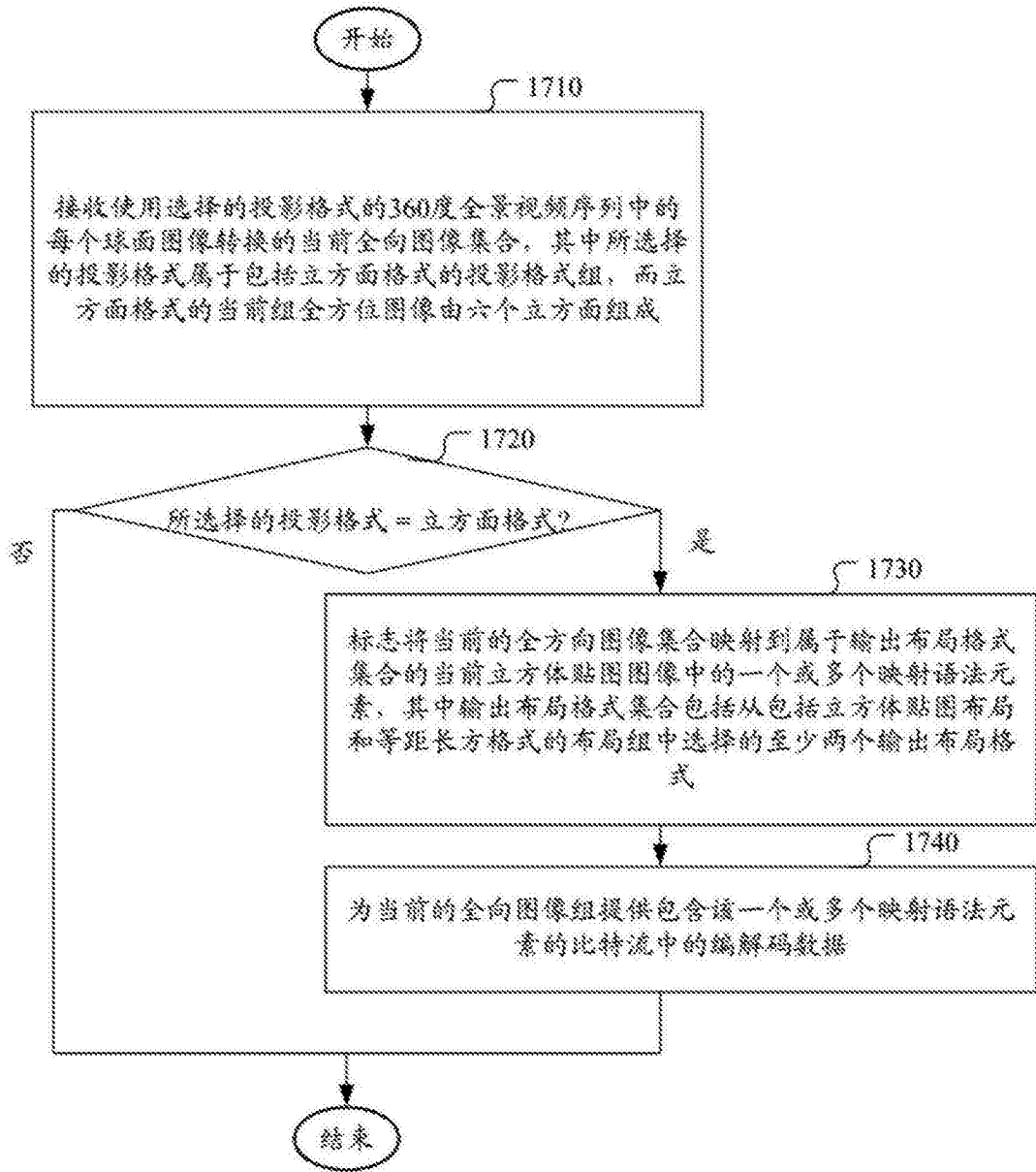


图17