(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 103688548 A
(43)申请公布日 2014.03.26

(21)申请号 201280033913.2
(22)申请日 2012.07.06
(66)本国优先权数据
PCT/CN2011/076991 2011.07.08 CN
(85)PCT国际申请进入国家阶段日
2014.01.08
(86)PCT国际申请的申请数据
PCT/CN2012/078301 2012.07.06
(87)PCT国际申请的公布数据
WO2013/007171 EN 2013.01.17
(71)申请人 汤姆逊许可公司
地址 法国伊西莱穆利诺
(72)发明人 蔡康颖 江文斐 田疆
(74)专利代理机构 北京市柳沈律师事务所 11105
代理人 史新宏
(51)Int.Cl.
H04N 19/70(2014.01)
H04N 19/20(2014.01)
G06T 17/20(2006.01)

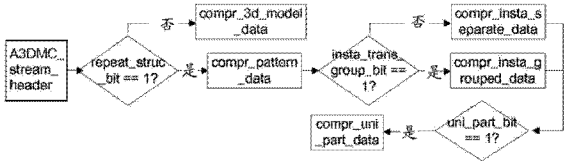
权利要求书2页 说明书14页 附图2页

(54)发明名称

为具有重复结构的三维模型编码和解码位流的系统及方法

(57)摘要

通常，3D网格通过三种类型的数据来表示：连接数据、几何数据和性质数据。可以像位流那样表示、发送和/或存储编码3D网格模型。虽然位流嵌入了所有变换数据，但这是有效的，并且可以解决有时位流大小、解码效率或抗错性最重要的几种应用。因此，公开了如何将一个实例的变换数据，即，它的位置、取向和比例因子放入位流中的两种选择。在第一模式中，在位流中一起打包一个实例的位置、取向和可能比例因子。在第二模式中，在位流中按照数据类型一起打包所有实例的变换数据模型，例如，位置、取向或可能比例因子。



CN 103688548 A

1. 一种编码或解码代表 3D 模型的位流的方法，包含如下步骤：

存取与 3D 模型相联系的多个实例信息，每个实例与相应模式和变换数据相联系；以及

以第一和第二模式之一编码或解码在位流中传输的多个模式信息，其中在第一模式下，逐个实例地以各自实例在位流中分组与每个各自实例相联系的变换信息，在第二模式下，逐种变换信息类型地在位流中分组变换信息。

2. 如权利要求 1 所述的方法，进一步包括使用指示符位来确定在第一模式下还是在第二模式下分组实例信息的步骤。

3. 如权利要求 2 所述的方法，其中变换数据的类型包括位置、取向和比例因子数据的一种或多种。

4. 如权利要求 3 所述的方法，进一步包含编码或解码与接在实例信息后面的 3D 模型的独特部分有关的数据的步骤。

5. 如权利要求 3 所述的方法，其中使用第一和第二模式的自适应组合编码或解码位流，其中使用第一模式编码或解码实例信息的所选几个，以及使用第二模式编码或解码实例信息的所选其它几个。

6. 一种编码代表 3D 模型的位流的编码器，包含：

在包括模式、和与各自模式和变换数据相联系的实例的 3D 模型中寻找重复结构的模块；以及

以第一和第二模式之一编码位流中的实例信息的模块，其中在第一模式下，逐个实例地以各自实例在位流中分组与每个各自实例相联系的变换信息，在第二模式下，逐种变换信息类型地在位流中分组变换信息。

7. 如权利要求 6 所述的方法，其中用于编码的模块包括确定在第一模式下还是在第二模式下分组实例信息的指示符位。

8. 如权利要求 7 所述的方法，其中变换数据的类型包括位置、取向和比例因子数据的一种或多种。

9. 如权利要求 8 所述的方法，其中用于编码的模块进一步包括与接在实例信息后面的 3D 模型的独特部分有关的数据。

10. 如权利要求 8 所述的方法，其中用于编码的模块使用第一和第二模式的自适应组合编码位流，其中使用第一模式编码或解码实例信息的所选几个，以及使用第二模式编码或解码实例信息的所选其它几个。

11. 一种解码代表 3D 模型的位流的解码器，包含：

访问包括与相应模式和变换数据相联系的实例信息的位流的模块；以及

以第一和第二模式之一解码位流中的实例信息的模块，其中在第一模式下，逐个实例地以各自实例在位流中分组与每个各自实例相联系的变换信息，在第二模式下，逐种变换信息类型地在位流中分组变换信息。

12. 如权利要求 11 所述的方法，其中用于解码的模块根据指示符位确定在第一模式下还是在第二模式下分组实例信息。

13. 如权利要求 12 所述的方法，其中变换数据的类型包括位置、取向和比例因子数据。

14. 如权利要求 13 所述的方法，其中用于解码的模块进一步解码与接在实例信息后面的 3D 模型的独特部分有关的数据。

15. 如权利要求 13 所述的方法，其中用于解码的模块使用第一和第二模式的自适应组合解码位流，其中使用第一模式编码或解码实例信息的所选几个，以及使用第二模式编码或解码实例信息的所选其它几个。

16. 一种存储在有形介质中的代表 3D 模型的位流，包含：

与相应模式和变换数据相联系的实例信息；

该实例信息以第一和第二模式之一包括在该位流中，其中在第一模式下，逐个实例地以各自实例在位流中分组与每个各自实例相联系的变换信息，在第二模式下，逐种变换信息类型地在位流中分组变换信息。

17. 如权利要求 16 所述的方法，其中该位流包括指示在第一模式下还是在第二模式下分组实例信息的指示符位。

18. 如权利要求 17 所述的方法，其中变换数据的类型包括位置、取向和比例因子数据。

19. 如权利要求 18 所述的方法，其中该位流进一步包括与接在实例信息后面的 3D 模型的独特部分有关的数据。

20. 如权利要求 18 所述的方法，其中该位流包括第一和第二模式的自适应组合，其中使用第一模式编码或解码实例信息的所选几个，以及使用第二模式编码或解码实例信息的所选其它几个。

为具有重复结构的三维模型编码和解码位流的系统及方法

技术领域

[0001] 本发明涉及基于重复结构寻找的 3D（三维）模型压缩算法的位流语法和语义、编码它们的方法和装置以及解码它们的方法和装置。

背景技术

[0002] 在实际应用中，许多 3D 模型由大量相连部件组成。并且，如图 1 所示，这些多连 3D 模型通常包含在各种变换下的许多重复结构。这种类型的 3D 模型的有效压缩方法应该能够提取存在于重复结构之中的冗余。

[0003] 在 2010 年 6 月 9 日提交、发明名称为“Efficient Compression Scheme for Large3D Engineering Models（大型 3D 工程模型的有效压缩方案）”、和转让给 Thomson Licensing（汤姆森特许公司）的 WO2010149492 中提出了利用在输入模型中寻找重复结构的多连 3D 模型的有效压缩算法，在此通过引用并入其教导。它寻找在各种位置中、在各种取向上以及以各种比例因子重复的结构。然后，将 3D 模型组织成“模式 - 实例”表示。模式是相应重复结构的代表性几何形状。属于重复结构的相连部件被叫做相应模式的实例，并通过它们与模式有关的变换，即，位置、取向和可能比例因子来表示。实例的取向通过在笛卡尔坐标系中用（x0，y0，z0）和（x1，y1，z1）表示，或在球形坐标系中用（α，β，γ）表示的两根正交轴表示。

发明内容

[0004] 这里公开的是涉及基于重复结构寻找的压缩算法的压缩位流语法和语义，基于重复结构寻找的压缩算法已经被证明比 MPEG-3DGC（运动图像专家组 - 三维图形压缩标准）提供的静态 3D 模型压缩算法更有效。我们基于重复结构寻找的压缩算法的所公开压缩位流语法和语义可应用于，例如，MPEG（运动图像专家组标准）。

[0005] 本发明是有关压缩位流语法和语义的。

[0006] 本发明还提供了为具有重复结构的 3D 模型编码和解码位流的系统及方法。

[0007] 利用该方法的装置公开在下面。

[0008] 本发明还提供了含有可执行指令，以便使计算机可以执行包含为具有重复结构的 3D 模型编码和解码位流的相应步骤的方法的计算机可读介质。

附图说明

[0009] 图 1 示出了具有大量相连部件和重复结构的示范性 3D 模型；

[0010] 图 2 示出了按照本发明处理 3DMC 压缩位流的结构的方法；

[0011] 图 3 示出了按照本原理的 3D 模型的示范性编码器；以及

[0012] 图 4 示出了按照本原理的 3D 模型的示范性解码器。

具体实施方式

[0013] 三维(3D)网格被广泛用在像视频游戏、工程设计、电子商务、虚拟现实、以及建筑和科学可视化那样表示3D对象的各种应用中。通常,它们的原始表示需要巨量数据。但是,为了存储或传输,大多数应用偏爱简洁的3D网格表示。通常,3D网格通过三种类型的数据来表示:连接数据、几何数据和性质数据。连接数据描述顶点之间的相邻关系,几何数据指定顶点地点,以及性质数据指定像法线向量、材质反射率和纹理坐标那样的属性。大多数3D压缩算法分开压缩连接数据和几何数据。几何数据的编码次序由底下连接编码决定。几何数据通常通过三个主要步骤来压缩:量化、预测和统计编码。3D网格性质数据通常以类似方式压缩。

[0014] 本发明涉及大型3D工程模型的有效压缩方法。这样的模型往往由几个分区,即所谓的"相连部件"组成。重复几何特征模式的表示中的冗余可以通过将等效(例如,在位置、尺寸标准化之后)的所有相连部件当作一种几何模式的实例来减小。可以聚集等效部件。一个群集可以只指3D模型的一些或所有部件。然后,可以通过相应几何模式(或聚类)的像字母数字标识符那样的标识符和可以从几何模式中重构部件的变换信息表示每个相连部件。这种变换信息可以示范性地包含比例因子、平均(或中心)值、取向轴(或/和分别地,旋转信息)、或移动信息的一种或多种。原则上,还有其他也是可能的。

[0015] 可以像位流那样表示、发送和/或存储编码模型。

[0016] 虽然我们希望位流嵌入所有变换数据,但我们也希望是有效的,并解决有时位流大小、解码效率或抗错性最重要的几种应用。

[0017] 因此,公开了如何将一个实例的变换数据,即,它的位置、取向和比例因子放入位流中的两种选择。它们两者具有它们自己的优势。两者的自适应组合尤其有利。

[0018] 选择(A)被叫做分组实例变换模式:当使用这种模式时,在位流中一起打包一个实例的位置、取向和可能比例因子。

[0019] 选择(A)的优势是:

[0020] • 它是抗错的。可以防止解码器丢失一些实例的变换。

[0021] • 在线解码。这意味着可以在读取压缩位流期间逐个解码实例。无需等待到整个压缩位流读取的完成。

[0022] • 较高编解码速度。

[0023] • 编解码器无需(或只需很小)缓冲区。

[0024] 选择(B)被叫做单独实例变换模式:在位流中一起打包所有实例的位置、取向和可能比例因子。换句话说,在位流中分开打包一个实例的位置、取向和可能比例因子。

[0025] 选择(B)的优势是:

[0026] • 相对于选择(A)压缩3D模型规模较小。

[0027] 使用选择(B)的解码器还具有如下特征:

[0028] • 解码器是非抗错的。

[0029] • 离线解码。这意味着解码器只能在读取了整个压缩位流之后才开始解码。

[0030] • 较低编解码速度。

[0031] • 缓冲区是必需的。

[0032] 我们的位流定义包括上面的两种选择(A)和(B)两者。然后,用户或自动控制器可以选取较好地适合它们的一种或多种应用的那一种。

[0033] 我们基于重复结构寻找的压缩算法 A3DMC 的压缩位流的一般结构如图 2 所示。

[0034] 该位流从首标缓冲区(A3DMC_stream_header)开始,该首标缓冲区(A3DMC_stream_header)包含解码压缩流的所有必要信息:在原始模型中是否存在重复结构、用于压缩模式和其它部分(如果有必要)的 3D 模型压缩方法、在这种位流中使用“分组实例变换模式”还是“单独实例变换模式”、是否存在原始模型未包括在任何重复结构中的一些部分(独特部分)等。

[0035] 如果在原始模型中不存在重复结构(repeat_struc_bit!=1),则位流的左部(例如,开头)是使用在 A3DMC_stream_header 中所指的 3D 模型压缩方法的压缩输入 3D 模型。否则,位流中的下一个部分是所有模式的压缩结果。取决于在这种位流中选取了哪种实例变换打包模式,在位流中下一个部分是 compr_insta_grouped_data 或 compr_insta_separate_data。如果在原始 3D 模型中存在独特部分,则附加 compr_uni_part_data。否则,该位流结束。

[0036] A3DMC 的压缩位流语法和语义将详细说明如下。

[0037] 位流语法和语义

[0038] 语法功能、类别和描述符的规定

[0039] f(n):使用写入(从左到右)的 n 个位的固定模式位串。n 取决于每个符号的代码长度。

[0040] ec(v):熵编码(例如,算术编码)的语法元素,其包括可能配置符号。

[0041] A3DMC_stream 类

[0042]

class A3DMC_stream{	位数	描述符
A3DMC_stream_header		
A3DMC_steam_data		
}		

[0043] 表 1

[0044] A3DMC_stream_header:包含首标缓冲区。

[0045] A3DMC_steam_data:包含数据缓冲区。

[0046] A3DMC_stream_header 类

[0047]

class A3DMC_stream_header{	位数	描述符
repeat_struc_bit	1	
3d_model_compr_mode	2	
QP	5	

If(repeat_struc_bit==1){		
pattern_num	8	
If(pattern_num==255){		
pattern_num_2	16	
}		
instance_num	16	
If(instance_num==65535){		

[0048]

instance_num_2	32	
}		
insta_trans_group_bit	1	
insta_orient_mode_bit	1	笛卡儿 / 球形坐标
usescalingbit	1	
__ uni_part_bit	1	
reserved_bits	4	
}		
}		

[0049] 表 2

[0050] repeat_struc_bit :指示在 3D 模型中是否存在重复结构的 1 位无符号整数。0 意味着无重复结构,1 意味着有重复结构。

[0051] 3d_model_compr_mode :指示用于压缩模式、独特部分、和如果未包括重复结构,原始 3D 模型本身的 3D 模型压缩方法的 2 位无符号整数。

[0052]

3d_model_compr_mode	含义
00	SC3DMC
01	3DMC 扩展

10&11	ISO 保留

[0053] 表 3

[0054] QP :指示量化参数的 5 位无符号整数。例如，QP 的最小值是 3，最大值是 31。

[0055] pattern_num :如果小于 255 则指示所有模式的数量的 8 位无符号整数。pattern_num 的最小值是 1。

[0056] pattern_num_2 :如果不小于 255 则指示所有模式的数量的 16 位无符号整数。在这种情况下，总模式数是(pattern_num_2+255)。

[0057] instance_num :如果小于 65535 则指示所有实例的数量的 16 位无符号整数。instance_num 的最小值是 1。

[0058] instance_num_2 :如果不小于 65535 则指示所有实例的数量的 32 位无符号整数。在这种情况下，总实例数是(instance_num_2+65535)。

[0059] insta_trans_group_bit :指示在这种位流中使用“分组实例变换模式”还是“单独实例变换模式”的 1 位无符号整数。对于“单独实例变换模式”为 0，对于“分组实例变换模式”为 1。

[0060] insta_orient_mode_bit :指示实例取向的编码模式的 1 位无符号整数。0 意味着球形模式，1 意味着笛卡儿模式。

[0061] use_scaling_bit :指示实例变换是否包括比例因子的 1 位无符号整数。对于比例因子包括在实例变换中为 1，对于比例因子未包括在实例变换中为 0。

[0062] uni_part_bit :指示在原始 3D 模型中是否存在独特部分的 1 位无符号整数。0 意味着没有独特部分，1 意味着有独特部分。

[0063] reserved_bits :总是 0000 和用于字节对准的 4 位无符号整数。

[0064] A3DMC_stream_data 类

[0065]

class A3DMC_stream_data{	位数	描述符
if(repeat_struc_bit==1){		
compr_repeat_struc_data		
}		
else{		
compr_3d_model_data		
}		
}		

[0066] 表 4

[0067] compr_repeat_struc_data :包含压缩 3D 模型，其包括重复结构。

[0068] compr_3d_model_data：包含压缩3D模型，其没有重复结构，通过3d_model_compr_mode所指的压缩方法编码。

[0069] compr_repeat_struc_data类

[0070]

class compr_repeat_struc_data{	位数	描述符
compr_pattern_data		
if(insta_trans_group_bit==1){		
compr_insta_grouped_data		
}		
else{		
compr_insta_separate_data		
}		
if(uni_part_bit==1){		
compr_uni_part_data		
}		
}		

[0071] 表5

[0072] compr_pattern_data：包含所有模式的压缩模式数据，其通过3d_model_compr_mode所指的压缩方法编码。

[0073] compr_insta_grouped_data：包含使用“分组实例变换模式”的压缩实例变换数据。

[0074] compr_insta_separate_data：包含使用“单独实例变换模式”的压缩实例变换数据。

[0075] compr_uni_part_data：包含压缩独特部分数据，其通过3d_model_compr_mode所指的压缩方法编码。

[0076] compr_insta_grouped_data类

[0077]

class compr_insta_grouped_data{	位数	描述符
for (i = 0; i < numofInstance; i ++) {		
compr_ith_insta_patternID		ec(v)
compr_ith_insta_position	bit_num_insta_position()	f(bit_num_ insta_position())
If (insta_orient_mode_bit == 1){		
compr_ith_insta_orient_cartesian		
}		
else{		
compr_ith_insta_orient_spherical		
}		
if (use_scaling_bit){		
compr_ith_insta_scaling		ec(v)
}		
}		
}		

[0078] 表 6

[0079] compr_ith_insta_patternID :包含第 i 实例的压缩模式 ID。

[0080] compr_ith_insta_position :包含第 i 实例的压缩位置。

[0081] compr_ith_insta_orient_cartesian :包含笛卡尔模式下第 i 实例的压缩取向。

[0082] compr_ith_insta_orient_spherical :包含球形模式下第 i 实例的压缩取向。

[0083] compr_ith_insta_scaling :包含第 i 实例的压缩比例因子。

[0084] bit_num_insta_position() :根据 QP 计算每个实例位置值的位数。

[0085] compr_ith_insta_orient_cartesian 类

[0086]

class compr_ith_insta_orient_cartesian {	位数	描述符
compr_ith_insta_orient_x0	bit_num_orient_cartesian()	f(bit_num_orient_ cartesian())
compr_ith_insta_orient_y0	bit_num_orient_cartesian()	f(bit_num_orient_ cartesian())
ith_insta_orient_z0_sgn	1	
if (compute_z0() < *threshold*) {		
compr_ith_insta_orient_z0_res	bit_num_orient_ res_cartesian()	f(bit_num_orient_ res_cartesian())
}		
if (compute_z0() ≈ 0) {		
compr_ith_insta_orient_z1	bit_num_orient_cartesian()	f(bit_num_orient_ cartesian())
if (abs(ith_x0) < abs(ith_y0)){		
ith_insta_orient_x1_sgn	1	

[0087]

}else{		
ith_insta_orient_y1_sgn	1	
}		
}else {		
if (abs(ith_x0) < abs(ith_y0)){		
compr_ith_insta_orient_x1	bit_num_orient_cartesian()	f(bit_num_orient_cartesian())
}else {		
compr_ith_insta_orient_y1	bit_num_orient_cartesian()	f(bit_num_orient_cartesian())
}		
ith_insta_orient_delta_sgn	1	
if (compute_z1() < *threshold*) {		
compr_ith_insta_orient_z1_res	bit_num_orient_res_cartesian()	f(bit_num_orient_res_cartesian())
}		
}		
}		

[0088] 表 7

[0089] 通过 2 根正交轴(x0,y0,z0)和(x1,y1,z1)表示笛卡尔模式下第 i 实例的取向。

[0090] compr_ith_insta_orient_x0 :包含第 i 实例的取向的压缩 x0。

[0091] compr_ith_insta_orient_y0 :包含第 i 实例的取向的压缩 y0。

[0092] compr_ith_insta_orient_z0_sgn :指示使用 x0 和 y0 计算 z0 所需的 z0 的符号的 1 位无符号整数。对于“-”为 0,对于“+”为 1。

[0093] compr_ith_insta_orient_z0_res :包含通过(z0-computer_z0())计算的 z0 的压缩残差。

[0094] compr_ith_insta_orient_z1 :包含第 i 实例的取向的压缩 z1。

[0095] ith_insta_orient_x1_sgn :指示使用 x0 和 y0 计算 x1 所需的 x1 的符号的 1 位无符号整数。对于“-”为 0,对于“+”为 1。

[0096] ith_insta_orient_y1_sgn :指示使用 x0 和 y0 计算 y1 所需的 y1 的符号的 1 位无符号整数。对于“-”为 0,对于“+”为 1。

[0097] compr_ith_insta_orient_x1 :包含第 i 实例的取向的压缩 x1。

[0098] compr_ith_insta_orient_y1 :包含第 i 实例的取向的压缩 y1。

[0099] ith_insta_orient_delta_sgn :指示使用 x0、y0、z0 和 y1 或 x1 计算 x1 或 y1 所需的符号的 1 位无符号整数。对于“-”为 0,对于“+”为 1。

[0100] compr_ith_insta_orient_z1_res :包含通过(z1-computer_z1())计算的 z1 的压缩残差。

[0101] threshold :在压缩领域中广泛接受的阈值。

[0102] computer_z0() :使用 x0、y0 和 z0 符号计算第 i 实例的 z0。

[0103] bit_num_orient_cartesian() :根据 QP 计算笛卡儿坐标系中每个取向值的位数。

[0104] bit_num_orient_res_cartesian() :根据 QP 计算笛卡儿坐标系中每个取向残差值的位数。

[0105] computer_z1() :使用 x0、y0、z0、x1 和 y1 计算第 i 实例的 z1。

[0106] compr_ith_insta_orient_spherical 类

[0107]

class compr_ith_insta_orient_spherical {	位数	描述符
compr_ith_insta_orient_alpha	bit_num_orient_alpha()	f(bit_num_orient_alpha())
compr_ith_insta_orient_beta	bit_num_orient_beta()	f(bit_num_orient_beta())
compr_ith_insta_orient_gamma	bit_num_orient_gamma()	f(bit_num_orient_gamma()
if (need_correction()) {		
compr_ith_insta_orient_res	6*bit_num_orient_ res_cartesian()	f(6*bit_num_orient_ res_cartesian())
}		
}		

[0108] 表 8

[0109] 通过 3 个角度,即,α、β 和 γ 表示球形模式下第 i 实例的取向。

[0110] compr_ith_insta_orient_alpha :包含第 i 实例的取向的压缩 α 。

[0111] compr_ith_insta_orient_beta :包含第 i 实例的取向的压缩 β 。

[0112] compr_ith_insta_orient_gamma :包含第 i 实例的取向的压缩 γ 。

[0113] compr_ith_insta_orient_res :包含第 i 实例的取向的在笛卡儿坐标系中的压缩残差。

[0114] bit_num_orient_alpha() :根据 QP 计算每个 α 值的位数。

[0115] bit_num_orient_beta() :根据 QP 计算每个 β 值的位数。

[0116] bit_num_orient_gamma() :根据 QP 计算每个 γ 值的位数。

[0117] need_correction() :检验取向,如果处在可能导致大错误的边缘条件下,则返回“真”;否则,则返回“假”。

[0118] compr_insta_separate_data 类

[0119]

class compr_insta_data_separate{	位数	描述符
compr_insta_patternID_length	32	
compr_insta_patternID_data		ec(v)
compr_insta_position_length	32	
compr_insta_position_data		
compr_insta_orient_length	32	
compr_insta_orient_data		
if(use_scaling_bit){		

[0120]

compr_insta_scaling_length	32	
compr_insta_scaling_data		ec(v)

}		
}		

[0121] 表 9

[0122] compr_insta_patternID_length :包含指示所有实例的压缩模式 ID 的长度的 32 位无符号整数。

[0123] compr_insta_patternID_data :包含所有实例的压缩模式 ID。

[0124] compr_insta_position_length :包含指示所有实例的压缩位置的长度的 32 位无符号整数。

[0125] compr_insta_position_data :包含所有实例的压缩位置。

[0126] compr_insta_orient_length :包含指示所有实例的压缩取向的长度的 32 位无符号整数。

[0127] compr_insta_orient_data :包含所有实例的压缩取向。

[0128] compr_insta_scaling_length :包含指示所有实例的压缩比例因子的长度的 32 位无符号整数。

[0129] compr_insta_scaling_data :包含所有实例的压缩比例因子。

[0130] compr_ins_position_data 类

[0131]

class compr_insta_position_data{	位数	描述符
insta_position_bbox	6*32	
config_n0_symbols		ec(v_n0)
config_n1_symbols		ec(v_n1)
config_n2_symbols		ec(v_n2)
}		

[0132] 表 10

[0133] insta_position_bbox :包含所有实例位置的边界框。

[0134] config_n0_symbols :包含 n0 个普通八叉树配置符号。

[0135] config_n1_symbols :包含 n1 个普通八叉树配置符号。

[0136] config_n2_symbols :包含 n2 个普通八叉树配置符号。

[0137] compr_insta_orient_data 类

[0138]

class compr_insta_orient_data{	位数	描述符
If(insta_orient_mode_bit==1){		

for(i=0;i<numofInstance;i++){		
compr_ith_insta_orient_cartesian		
}		
}		

[0139]

else{		
for(i=0;i<numofInstance;i++){		
compr_ith_insta_orient_spherical		
}		
}		
}		

[0140] 表 11

[0141] 另外，也可以将上述的位流嵌入像由 MPEG-3DGC[w11455] 定义的 SC3DMC 位流那样的其它位流中。这里是那个解决方案的可能实施例。

[0142] 我们按如下使用 SC3DMCStreamHeader 的 encodingMode 的 ISO 保留值 3（参考 [w11455] 的原始表格 53）。

[0143] 表 1—SC3DMC 编码模式

[0144]

encodingMode	方法
0	QBCR
1	SVA
2	TFAN
→ 3	A3DMC
4-255	ISO 保留

[0145] 表 12

[0146] 我们按如下修订定义在 [w11455] 中的 SC3DMCStream。

[0147]

Class SC3DMCStream{	位数	描述符

SC3DMCStreamHeader;		
If(header.encodingMode==3){		
A3DMC_stream_header_SC3DMC;		
compr_repeat_struc_data		
}		
else{		
SC3DMCStreamData data;		
}		
}		

[0148] 表 13

[0149] A3DMC_stream_header_SC3DMC 类

[0150]

class A3DMC_stream_header{	位数	描述符
3d_model_compr_mode	2	
QP	5	
reserved_bits_first	1	
pattern_num	8	

[0151]

If(patternnum==255){		
_ pattern_num_2	16	
}		
instance_num	16	
If(instance_num==65535){		
instance_num_2	32	
}		

insta_trans_separate_bit	1	
insta_orient_mode_bit	1	笛卡尔 / 球面坐标
use_scaling_bit	1	
uni_part_bit	1	
reserved_bits	4	
}		

[0152] 表 14

[0153] reserved_bits_first :总是 0 的 1 位无符号整数。用于字节对准。

[0154] 尤其,本发明涉及

[0155] 1. 我们基于重复结构寻找的压缩算法的压缩位流语法和语义。

[0156] 2. 具有两者在位流中都有可能的单独实例或分组实例变换数据选择的解决方案。

[0157] 3. 在现有 SC3DMC 位流中我们当前位流建议的可能实施例。

[0158] 图 3 描绘了示范性 3D 模型编码器 300 的方块图。装置 300 的输入可以包括 3D 模型、编码 3D 模型的质量参数和其它元数据。3D 模型首先经过重复结构寻找模块 310,它以模式、实例和独特成分的形式输出 3D 模型。模式编码器 320 被应用于压缩模式,独特成分编码器 350 被应用于编码独特成分。例如,根据用户所选模式编码实例成分信息。如果选择实例信息组模式,则使用分组实例信息编码器 340 编码实例信息 ;否则,使用基本实例信息编码器 330 编码它。在重复结构核实器 360 中进一步核实编码成分。如果编码成分未满足它的质量要求,则使用独特成分编码器 350 编码它。在位流组装器 370 中组装模式、实例、和独特成分的位流。

[0159] 图 4 描绘了示范性 3D 模型解码器 400 的方块图。装置 400 的输入可以包括 3D 模型的位流,例如,编码器 300 生成的位流。压缩位流中与模式有关的信息由模式解码器 420 解码。与独特成分有关的信息由独特成分解码器 450 解码。实例信息的解码也取决于用户所选模式。如果选择实例信息组模式,则使用分组实例信息解码器 440 解码实例信息 ;否则,使用基本实例信息解码器 430 解码它。在模型重构模块 460 上重构解码的模式、实例信息和独特成分,以生成输出 3D 模型。

[0160] 本文所述的实现可以以,例如,方法或进程、装置、软件程序、数据流、或信号的形式实现。即使只在单种实现形式的背景下讨论(例如,只作为方法来讨论),所讨论的特征的实现也可以以其它形式(例如,装置或程序)实现。装置可以以,例如,适当硬件、软件、或固件的形式实现。方法可以在,例如,像例如处理器那样的装置中实现,处理器一般指处理设备,包括,例如,计算机、微处理器、集成电路、或可编程逻辑设备。处理器还包括像,例如,计算机、蜂窝式电话、便携式 / 个人数据助理(“PDA”)、和有助于在最终用户之间传送信息的其他设备那样的通信设备。

[0161] 提到本原理的“一个实施例”、或“实施例”、或“一种实现”或“实现”以及它们的其它变体意味着结合该实施例描述的具体特征、结构、特性等包括在本原理的至少一个实施

例中。因此，出现在整个说明书各处的短语“在一个实施例中”、或“在实施例中”、或“在一种实现中”或“在实现中”以及任何其它变体的出现不一定都指代相同的实施例。

[0162] 对于本领域的普通技术人员来说，显而易见，各种实现可以产生格式化成携带可以，例如，存储或发送的信息的多种信号。该信息可以包括，例如，执行方法的指令、或由所述实现之一产生的数据。例如，可以将信号格式化成携带所述实施例的位流。可以将这样的信号格式化成，例如，电磁波(例如，使用频谱的射频部分)或基带信号。格式化可以包括，例如，编码数据流和将编码数据流调制在载波上。信号携带的信息可以是，例如，模拟或数字信息。众所周知，可以在多种不同有线或无线链路上发送信号。可以将信号存储在处理器可读介质上。

[0163] 原则上，公开的发明也可以应用于其它数据压缩领域。本发明导致独特位流格式。

[0164] 虽然位流嵌入了所有变换数据，但这是有效的，并且可以解决有时位流大小、解码效率或抗错性最重要的几种应用。因此，公开了如何将一个实例的变换数据，即，它的位置、取向和比例因子放入位流中的两种模式选择。在第一模式(选择 A)中，在位流中一起打包一个实例的位置、取向和可能比例因子。在第二模式(选择 B)中，在位流中一起打包所有实例的位置、取向或可能比例因子。

[0165] 应当明白，仅仅通过举例的方式对本发明作了描述，可以不偏离本发明的范围地对细节加以修改。

[0166] 公开在本描述以及(在适当情况下)权利要求书和附图中的每种特征可以独立地或以任何适当组合方式提供。这些特征在适当情况下可以用硬件、软件或两者的组合来实现。这些连接在可应用的情况下可以实现成无线连接或不一定直接或专用的有线连接。出现在权利要求书中的标号只是为了例示，不会对权利要求书的范围起限制作用。

[0167] 引用的参考文献

[0168] [PA090039]Kangying Cai,J.Yu and Z.B.Chen,“EFFICIENT COMPRE-SSION SCHEME FOR LARGE3D ENGINEERING MODELS”,International Patent Application No.PCT/EP2010/058048,published as WO2010/149492A1(internal reference PA090039)；

[0169] [CAI2009]Kangying Cai,Wencheng Wang,Zhibo Chen,Quqing Chen and Jun Teng,“Exploiting Repeated Patterns for Efficient Compression of Massive Models”,VRCAI2009；

[0170] [w11455]Final text of ISO/IEC14496-164th Edition,MPEG-3DGC,93th MPEG meeting。

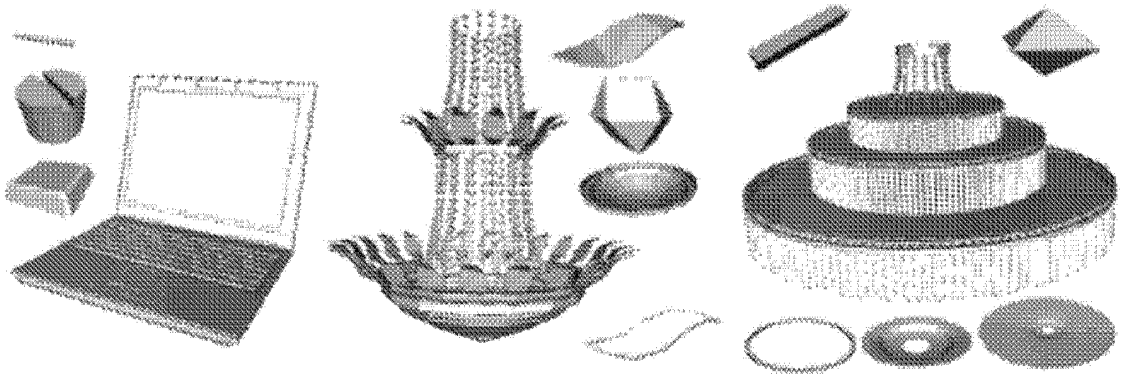


图 1

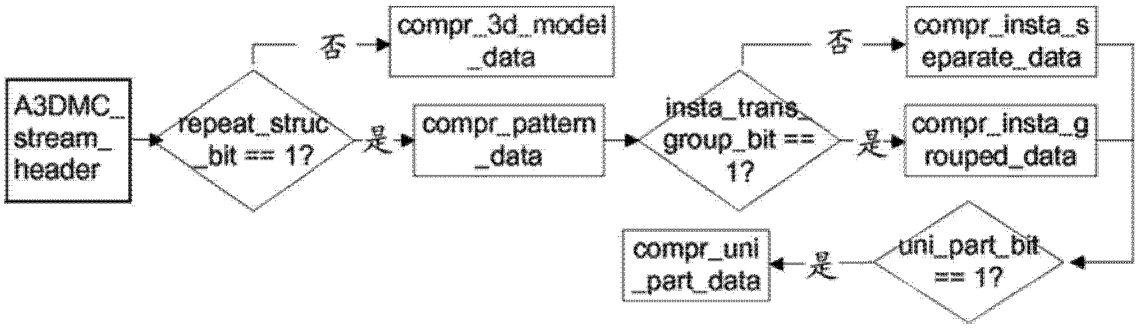

A3DMC_
stream_
header
repeat_struc
_bit == 1?
否
是
compr_3d_model
_data
compr_pattern
_data
insta_trans_
group_bit ==
1?
否
是
compr_insta_s
eparate_data
compr_insta_g
rouped_data
uni_part_bit
== 1?
是
compr_uni
_part_data

图 2

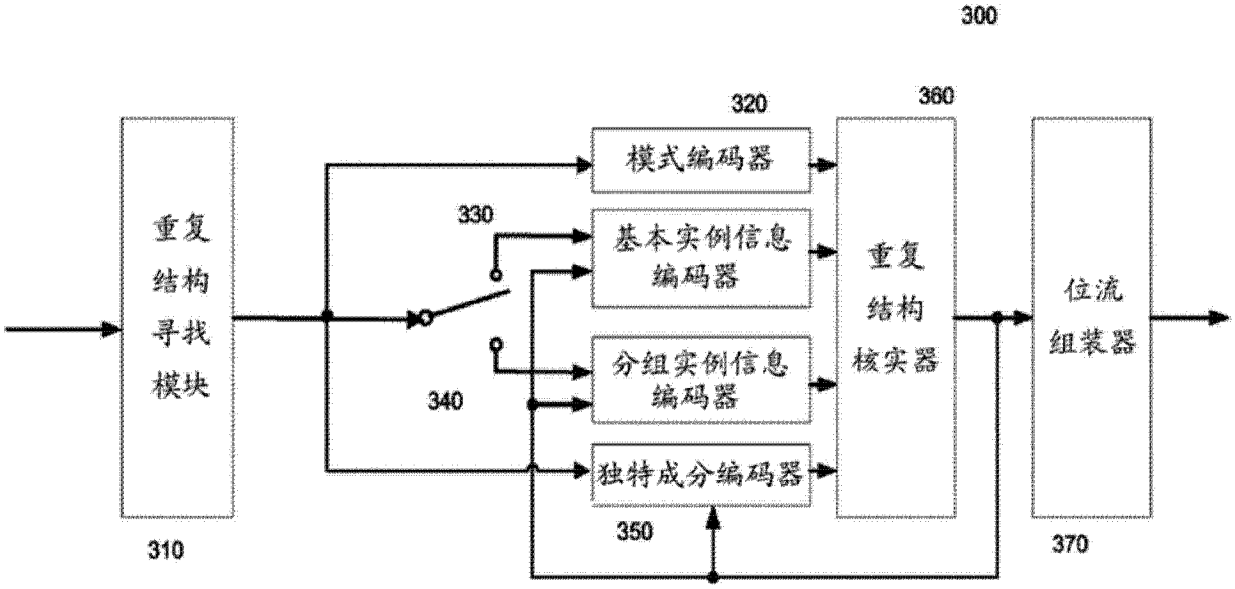

300
重复
结构
寻找
模块
310
320
模式编码器
330
基本实例信息
编码器
340
分组实例信息
编码器
独特成分编码器
350
360
重复
结构
核实器
位流
组装器
370

图 3

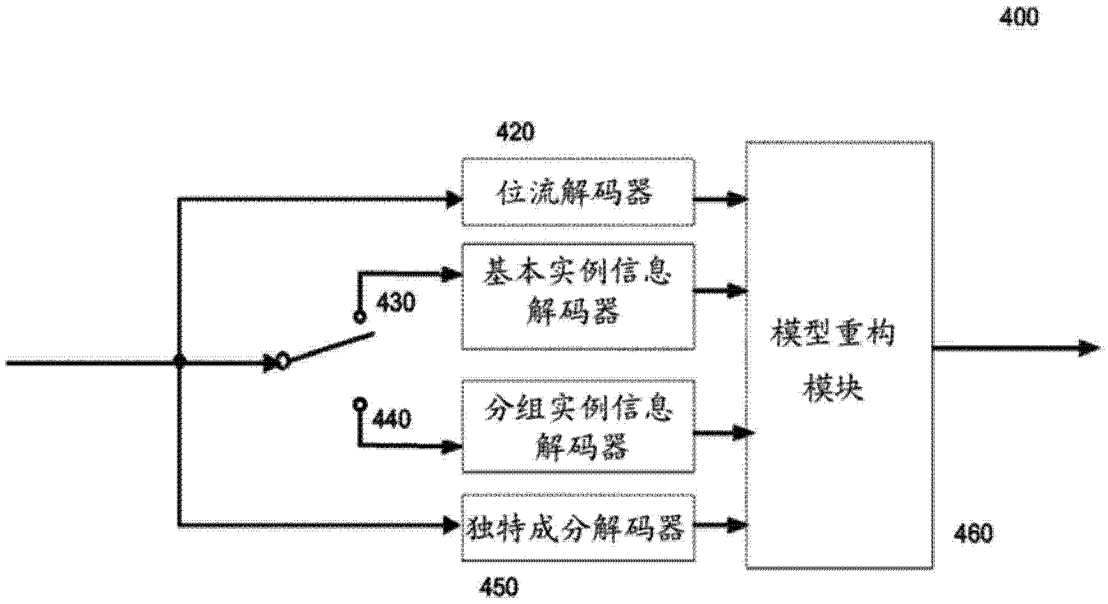

400
420
位流解码器
基本实例信息解码器
430
440
分组实例信息解码器
独特成分解码器
450
模型重构模块
460

图 4