精细建筑物碎片化纹理优化的二维装箱方法

张琳琳[1],朱 庆[1],胡 翰[2],翁其强[1],丁雨淋[3],李 赟[1]
(1.西南交通大学地球科学与环境工程学院,四川 成都 611756;
2.香港理工大学土地测量与地理资讯学系,香港 999077;
3.香港中文大学太空与地球信息科学研究所,香港 999077)

摘 要:精细建筑物模型存在许多几何小面片,导致大量碎片化纹理,这不仅会降低模型加载效率,更会降低模型实时渲染效率,制约大规模城市三维模型集成应用。为此,本文提出一种精细建筑物碎片化纹理优化的二维装箱方法。首先计算共用纹理并集区域,优化冗余纹理内容,并同时顾及超出常规坐标[0,1]范围的异常纹理;然后根据指定纹理集尺寸,使用二维装箱算法简化纹理,降低纹理数量;最后根据变化信息,重映射纹理坐标。实验结果表明,使用本文方法,纹理数据量减少了17.27%,纹理文件个数减少了98.67%,GPU耗时减少了39.69%,并可有效避免因异常纹理坐标导致的纹理拉花问题,同时也提高了数据格式兼容性。

关键词:纹理集;纹理优化;精细建筑物模型;二维装箱算法

1 引 言

随着多源数据多细节层次建模技术的不断发展,三维城市模型的精细程度不断提高,几何、纹理数据量呈几何级数增长。结构复杂的精细建筑物模型,如LOD-3(level of detail)建筑物模型,一个窗户有多个平面,每个平面对应一个碎片化纹理。现有可视化引擎,如Microsoft XNA、Unity 3D等,通常不支持多纹理模式,需要将三维格网分割成多个简单的平面,每个平面对应一个单独的纹理,对不同材质纹理需通过多次指令逐一渲染。由于每个平面都会单独调用中央处理器(central processing unit,CPU)指令进行绘制,增加了模型加载压力,降低了渲染效率,无法充分利用图像处理器(graphics processing unit, GPU)批处理机制,不适合大规模城市三维模型应用。因此,如何将碎片化纹理进行封装,减轻CPU负担,提高GPU加载和绘制效率,减少磁盘读取操作数目,成为当前倾斜摄影测量三维城市模型集成应用的关键问题。

现有的纹理合并工具,依赖人工经验,自动化程度低,应用局限较大,如NVIDIA公司的纹理软件工具(texture tools)、Autodesk公司3DMax软件的UVW-UnWrap工具、使用ISOMAP的商业软件,以及OpenSceneGraph的NodeOptimizer。三维模型的纹理组织方法可分为连续表面模型的纹理组织方法和离散表面模型的纹理组织方法两类。连续表面模型的纹理组织方法已有成熟应用,如Google Earth、Virtual Earth,主要方法为硬件支持的切片图方法和瓦片影像金字塔方法。离散表面模型的纹理组织方法有Blockmap方法、纹理集(texture atlas)方法等。目前相关的大部分研究主要着眼于如何更优地进行纹理合并,虽然在一定程度上减少了纹理文件个数,但模型中存在UV坐标超出[0,1]范围的异常纹理,导致其结果存在大量冗余纹理,纹理数据量反而增加,通过牺牲数据量换取渲染效率的提高。虽有一些研究中涉及了异常纹理的处理方法,但是并未进行完整的讨论。

为此,本文设计了一种在保证可视化效果的前提下,能够有效减少纹理文件个数和数据量,提高模型加载和绘制效率的精细建筑物碎片化纹理优化的二维装箱方法。

2 算法流程

如图1所示,精细建筑物碎片化纹理优化的二维装箱方法包含以下关键技术环节:

(1)纹理去冗余。首先,解析原始建筑物模型,根据 UV 坐标是否超出[0,1]将所有纹理分为正常纹理和异常纹理两类;然后,分别对正常纹理和异常纹理进行去冗余处理,得到简化纹理和重复纹理。

(2)纹理合并。首先,根据得到的简化纹理,模拟二维装箱过程,若装箱结果适应给定纹理集尺寸,则将纹理进行降采样,重新模拟装箱过程,直至满足给定纹理集尺寸;然后,根据装箱结果对降采样后的简化纹理进行合并,得到纹理集。

(3)纹理重映射。根据纹理去冗余和纹理合并过程中的变化信息,重映射纹理坐标,得到纹理优化后的建筑物模型。

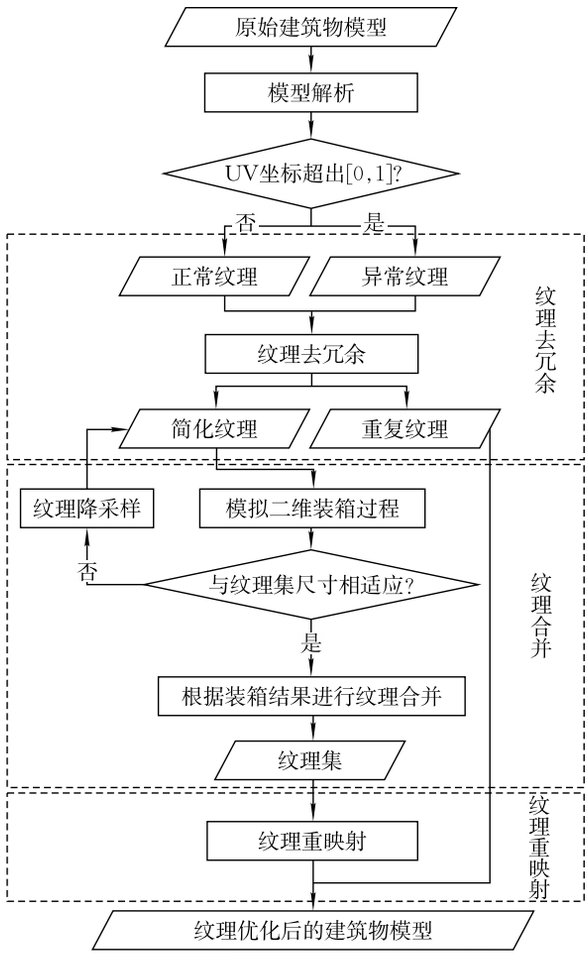


图 1 算法流程

3 精细建筑物碎片化纹理优化的二维装箱方法

3.1 纹理去冗余

通过三维坐标和纹理坐标(UV 坐标)可以确定纹理影像坐标区域的外接矩形,即有效纹理区域,有效纹理区域以外的图像部分被称为冗余纹理。冗余纹理不仅占据了存储资源,根据纹理映射机制,还可能使慢速硬盘与高速内存、显存之间出现传输瓶颈。因此,在纹理打包前,需要剔除冗余纹理。如图 2 所示,根据 UV 坐标裁剪原纹理图像,得到新纹理图像并更新纹理坐标,本文将该过程称为纹理更新。

本文定义纹理坐标超出[0,1]范围的纹理为异常纹理,异常纹理在逐基元渲染中可被有效处理,但难以进行纹理封装,常导致可视化中的纹理拉花问题。因此,将模型中的纹理分为正常纹理和异常纹理两类,分别进行去冗余,同时重置异常纹理的 UV 坐标到[0,1]范围内,以便进行后续的纹理合并。

1. 正常纹理去冗余

根据单张纹理影像是否被多个面片使用，将模型中的正常纹理分为共用纹理和单独纹理两类。共用纹理首先计算该纹理影像对应的所有面片纹理坐标并集，然后进行纹理更新；单独纹理直接进行纹理更新，得到简化纹理。正常纹理去冗余流程如图 3 所示。

2. 异常纹理去冗余

纹理复用量超过阈值的异常纹理为重复纹理。根据纹理是否被多个面共用，将重复纹理以外的异常纹理再细分为共用纹理和单独纹理。重复纹理直接输出，不进行纹理更新，也不参与后续纹理打包；共用纹理先计算与该纹理影像对应所有面的纹理坐标并集，然后进行纹理更新；单独纹理进行纹理更新。异常纹理去冗余流程如图 4 所示。

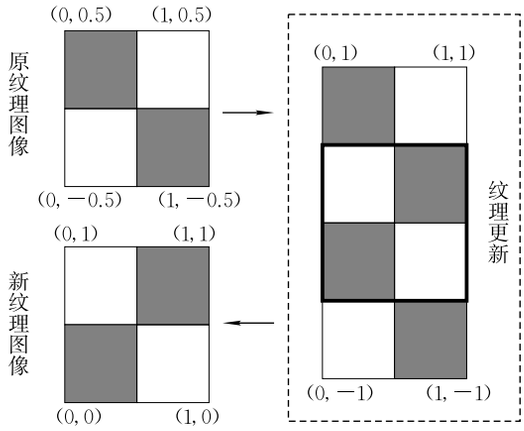


图 2　纹理更新

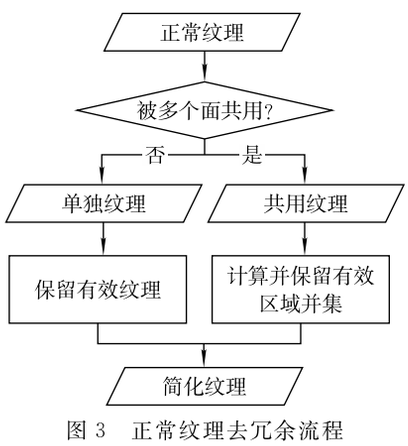


图 3　正常纹理去冗余流程

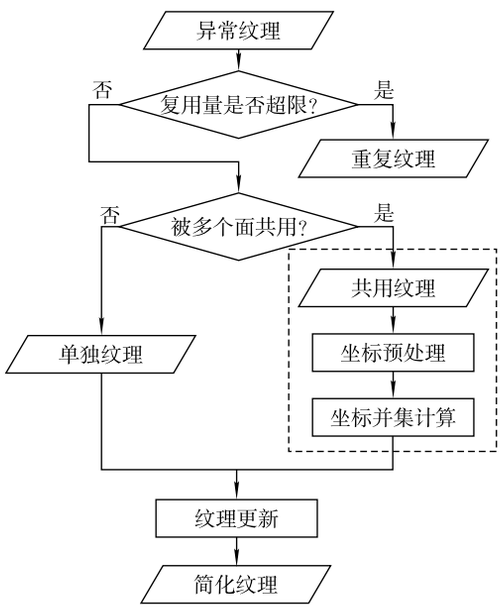


图 4　异常纹理去冗余流程

如图 5 所示，异常纹理之间的纹理坐标可能相差较大，直接使用原纹理坐标进行并集计算可能会使得到的并集范围很大，造成纹理更新上的困难。

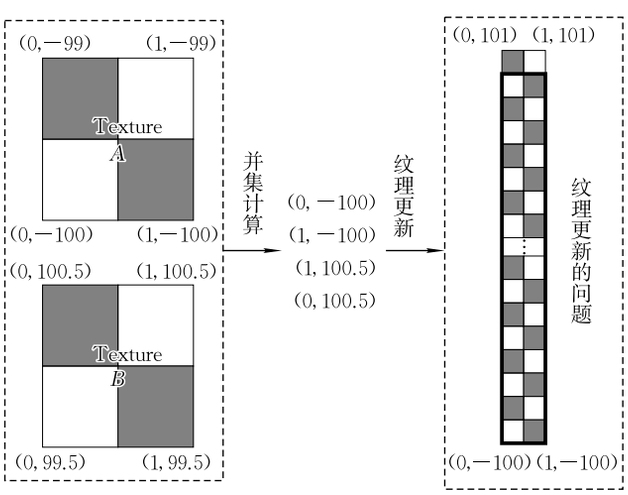


图 5　纹理更新问题

由于将纹理坐标进行整数量偏移不会改变纹理贴图的效果,因此在进行并集计算前,对纹理坐标进行预处理,在 U、V 方向上各加一个整数偏移值,使纹理左下角 UV 坐标值均在[0,1]范围内。如图 6 所示,通过坐标预处理,可有效解决纹理坐标差异大带来的纹理更新问题。

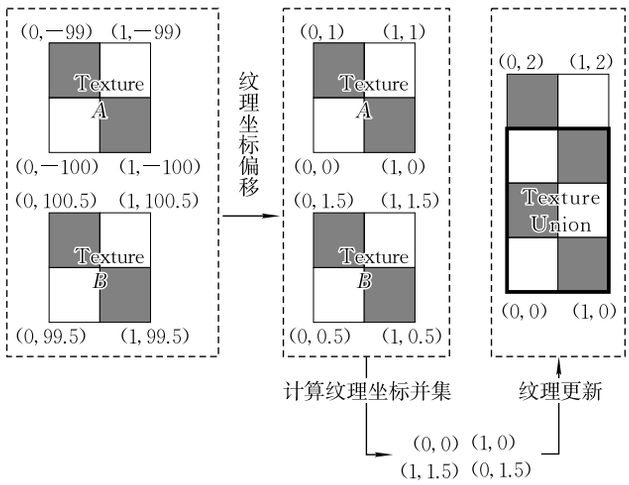


图 6 坐标预处理

3.2 纹理合并

纹理文件的特点是个体小、数量多,在模型渲染时引起大量纹理状态切换,导致批次数目急剧增加,严重影响场景渲染效率。将单个建筑物模型中的大量碎片化纹理合并为一张大纹理,能够有效提高场景渲染效率。

装箱算法起源于物流运输,其研究目标是求解小物品装入大容器中的总体布局方案,并使之在特定约束条件下达到特定优化目标,如消耗的容器数量最少等,与纹理合并目标具有一致性。因此,利用二维装箱算法进行纹理合并,能够最大限度提高纹理集的空间利用率,节省存储空间。

装箱算法中的物品矩形在插入前,需要在空间集合中按照一定规则搜索最适合容纳自身的自由矩形,然后进行放置。自由矩形是指容器中最大的可用矩形区域,也常被定义为矩形洞。自由矩形满足:①不与任意一个已插入物品矩形相交;②不存在任何其他可用矩形区域能容纳自由矩形,即自由矩形是最大的。二维装箱常用的插入规则有 5 种,分别是 BSSF(best short side fit)、BLSF(best long side fit)、BAF(best area fit)、BL(bottom left rule)和 CP(contact point rule)。BSSF 要求将新矩形放置在最适合的自由矩形短边上,BLSF 则将新矩形放置在最适合的自由矩形长边上,BAF 将矩形放入适合的最小自由矩形中,BL 将矩形按照俄罗斯方块规则进行放置,CP 尽可能选择矩形与其他矩形接触的位置。根据 5 种定义分别设计了纹理合并算法,对同一建筑物模型进行测试,指定纹理集尺寸大小均为 1 024×1 024,得到纹理集数据大小为:CP>BAF>BLSF>BSSF>BL。这表明在打包建筑物模型纹理方面,CP 算法比其他 4 种算法更优,因此本文采用 CP 规则的二维装箱算法来进行建筑物模型的纹理打包。

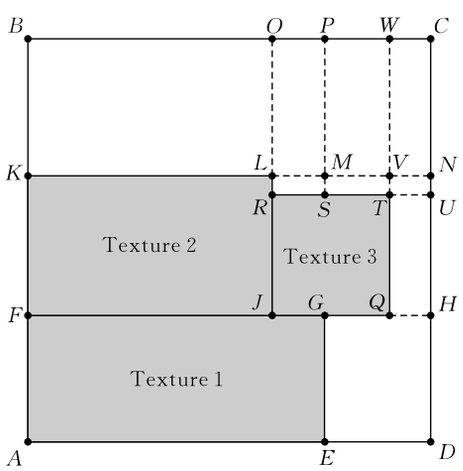


图 7 二维装箱

图 7 为采用 CP 规则的二维装箱过程。纹理集 $ABCD$ 为容器矩形,待插入的 3 个纹理为物品矩形。初始状态下仅有自由矩形 $ABCD$,纹理 1 直接进行插入,插入后产生自由矩形 $FBCH$、$JOCH$。按照 CP 规则,纹理 2 插入 $FBCH$ 中,与已插入物品矩形 $AFGE$ 的接触面最大,插入后产生自由矩形 $KBCN$、$JOCH$、$EPCD$。纹理 3 插入矩形 $JOCH$,获得与已插入物品矩形 $AFGE$、$FKLJ$ 的最大接触面,插入后产生自由矩形 $KBCN$、$ROCU$、

$EGHD$、$QWCH$。

经过纹理去冗余，建筑物模型中的纹理被分为简化纹理和重复纹理两类。重复纹理由于复用量较大，对其打包会较大程度上增加数据量，因此保留重复纹理，只合并简化纹理。

图 8 是纹理合并的算法流程，包含以下 3 个步骤：①对简化纹理进行预处理。首先，根据预设的纹理集尺寸，将简化纹理中尺寸大于该值的纹理缩放到纹理集尺寸；然后，将所有纹理按其最长边进行降序排列，得到排序后的待打包纹理。②根据预设纹理集尺寸及待打包纹理尺寸，生成相应的容器矩形和物品矩形，模拟二维装箱过程。若装箱失败，则对待打包纹理进行降采样，重新模拟装箱，直到将全部物品成功装入容器中为止。③根据装箱结果，合并纹理，得到纹理集。

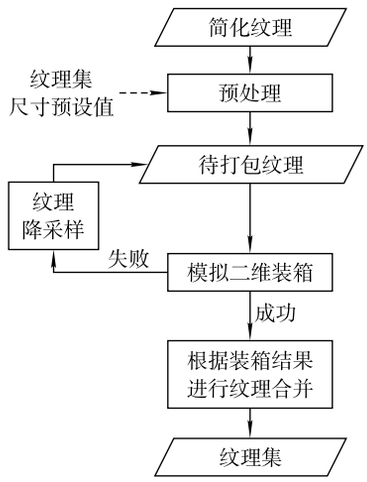


图 8　纹理打包

为了满足不同的应用需求，本文还设计了纹理无损打包算法，与上述纹理打包算法的区别在于允许生成多个纹理集，不对纹理进行降采样处理。

3.3　纹理重映射

纹理去冗余将纹理影像拼接、裁剪；纹理合并将纹理影像缩放、平移及旋转。纹理坐标随之发生了变化，原始的映射关系已经不再适用，因此需要进行纹理坐标重映射。

纹理去冗余后，新的纹理坐标计算方法为

$$\left.\begin{aligned} U' &= \frac{U - U_{\min}}{D_U} \\ V' &= \frac{V - V_{\min}}{D_V} \end{aligned}\right\} \tag{1}$$

式中，U、V 为原纹理坐标值，U'、V' 为计算得到的新纹理坐标值，$U_{\min}$、$V_{\min}$ 分别表示原纹理在 U、V 方向上的最小值，D_U、D_V 分别表示原纹理在 U、V 方向上的长度。

纹理合并后，新的纹理坐标计算方法为

$$\left.\begin{aligned} U' &= U \times T_U \\ V' &= V \times T_V \end{aligned}\right\} \tag{2}$$

式中，T_U、T_V 中分别记录了在插入纹理集的过程中，纹理在 U、V 方向上所进行的平移、缩放及旋转信息。

4　实验与分析

本文采用 C++作为开发语言，在 Visual Studio 2015 开发平台上进行实验。实验系统微机硬件配置如下：CPU 为 Intel(R) Core(TM) i5-3230M CPU @ 2.60 GHz；内存为 4.00 GB；显卡为 NVIDIA GeForce GT630M。实验数据是某教学楼模型，数据格式为 skp。

首先，对 skp 格式模型进行解析，直接输出原始模型。然后，使用本文方法，分别输出纹理打包模型和纹理无损打包模型。最后，使用同类方法，即不进行纹理去冗余和异常纹理处理的二维装箱方法，输出纹理直接打包模型和纹理直接无损打包模型。输出模型格式均为 Wavefront obj，模型顶点数量为 1 950，纹理集尺寸为 2 048×2 048，没有纹理的面由一个 64×64 白色影像替代。

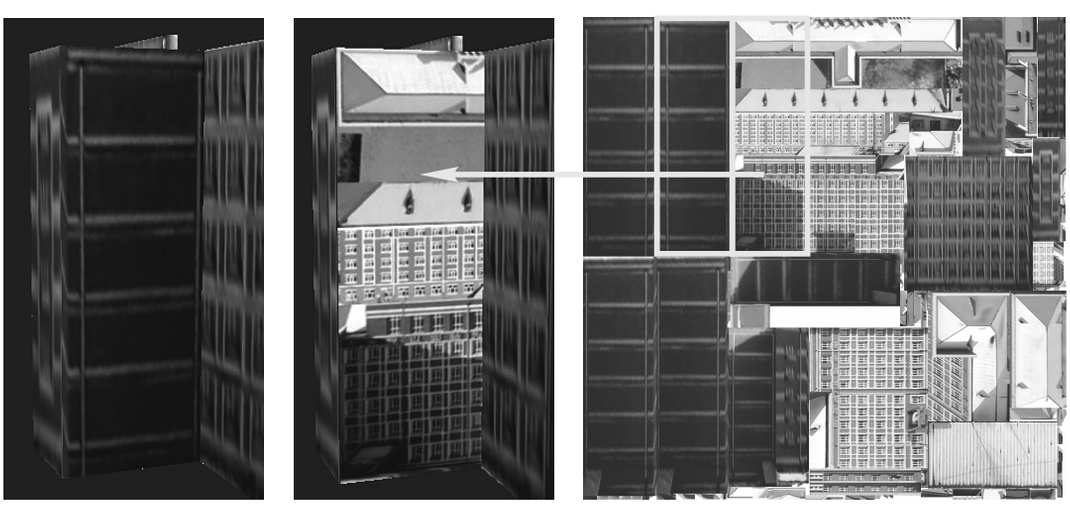
表 1 中记录了输出 5 种模型的纹理数据大小、纹理缩放系数、纹理文件个数以及在 OpenSceneGraph 平台上进行可视化实验得到的对应渲染帧率、GPU 耗时和载入耗时的对比结果。其中,渲染帧率、GPU 耗时及载入耗时均为 10 次实验结果的平均值。由于模型中存在 1 张尺寸超出 2 048×2 048 的大纹理,预处理后,该纹理数据量由 4.8 MB 变为 1.14 MB,因此从原始模型中减去该差值,以便进行后续的实验分析。

表 1 实验结果

模型类型	纹理数据大小/MB	纹理缩放系数	纹理文件个数	渲染帧率/fps	GPU 耗时/ms	载入耗时/s
原始模型	3.88(7.54)	1	376	59.85	1.31	5.73
无损打包模型	3.21	1	5	59.85	0.79	1.44
直接无损打包模型	4.21	1	4	59.87	0.81	1.41
打包模型	1.23	0.55	2	59.85	0.81	1.04
直接打包模型	1.40	0.41	1	59.88	0.83	0.99

从表 1 中可以看到,在纹理数据量方面,无损打包模型比原始模型减少了 17.27%,直接无损打包模型比原始模型增加了 8.5%,无损打包模型比直接无损打包模型减少的量是原始模型的 25.77%。考虑到纹理集中的空白区域及生成的白色纹理,在纹理打包算法中增加了去冗余步骤后,纹理数据量不但没有增加,反而有了较为可观程度的减少。在纹理文件个数方面,经过纹理无损打包后,纹理文件由 376 锐减为 5 个,减少了 98.67%,在允许进行纹理重采样的情况下,纹理文件可以直接减少为 1 张纹理集和 1 张对应没有纹理面的白色纹理。在 GPU 效率方面,各类打包模型的 GPU 耗时相近,但与原始模型相比有较大程度的减少,如无损打包模型减少了 39.69%。渲染帧率打包前后变化不明显。在模型载入耗时方面,两种无损打包模型相近,无损打包模型比原始模型减少了 74.87%;两种打包模型相近,打包模型比原始模型减少了 81.85%,比无损打包模型减少了 27.78%。

在可视化效果方面,观察 1 张 UV 坐标为(1,0)、(2,0)、(2,1)、(2,0)的纹理在无损打包模型和直接无损打包模型中的情况。该纹理在无损打包模型中贴图正确,而在直接无损打包模型中该位置贴上了对应纹理集中正确纹理旁边的部分。图 9(a)为异常纹理在无损打包模型中的可视化效果;图 9(b)为异常纹理在直接无损打包模型中的可视化效果;图 9(c)为该纹理在直接无损打包模型中对应的纹理集,靠左矩形线框住的部分是正确纹理,靠右矩形线框住的是由于直接对异常纹理进行打包导致被错误映射的纹理,箭头指示表明靠右矩形框内的纹理被错误映射到模型表面。图 10 是纹理无损打包模型在 OpenSceneGraph 中的可视化效果。



(a) 本文方法 (b) 同类方法 (c) 异常纹理在直接无损打包模型中对应的纹理集

图 9 异常纹理在无损打包模型与直接无损打包模型中的表现

图 10　无损打包模型可视化效果

5　结束语

通过对本文纹理优化方法的实验分析可以得出以下结论：

(1)该方法能够有效解决传统纹理合并方法在打包 UV 坐标超出[0,1]范围纹理时引起纹理错误映射的问题。

(2)该方法能够在不牺牲纹理分辨率的前提下，显著减少纹理数据量，解决了纹理合并中存储空间与绘制效率无法两全的难题。

参考文献：(略)

第一作者简介：张琳琳，女，1993 年生，硕士研究生，主要研究方向为三维模型重建的理论与方法。E-mail：ZhangLinlin36@163.com