计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200200029

基于 OBJ 模型的三维彩色切片技术研究

邢敬普[1]　李凤岐[1]　王胜法[2]　王　祎[2]　宗贵升[3]　范永刚[1]
1 大连理工大学软件学院　辽宁 大连 116620
2 大连理工大学国际信息与软件学院　辽宁 大连 116620
3 深圳七号科技有限公司　广东 深圳 518000
(solvc@mail.dlut.edu.cn)

摘　要　近年来,随着 3D 打印技术的不断进步,彩色 3D 打印正成为行业的普遍需求。然而作为目前 3D 打印领域模型描述的标准文件格式,STL 文件并不存储三维模型的颜色信息,无法满足彩色 3D 打印对模型信息提取和切片处理过程的新需求。在此背景下,选取 OBJ 彩色模型作为彩色切片技术的研究对象,解析其文件结构,提取其中保存的模型几何和颜色相关信息,并进行切片过程优化处理;结合 OBJ 模型的具体特点,在传统拓扑切片算法的基础上,提出基于模型连续性的彩色切片算法,给出了算法的整体流程步骤。应用该算法的切片处理提高了分层处理的效率,得到了模型的分层轮廓信息,完成了彩色切片的处理过程。实验结果证明该技术可对 OBJ 模型进行彩色切片处理,其效果良好,稳定可靠。

关键词:3D 打印;彩色切片;OBJ 模型
中图法分类号　TP311

Study on 3D Color Slicing Technology Based on OBJ Model

XING Jing-pu[1],LI Feng-qi[1],WANG Sheng-fa[2],WANG Yi[2],ZONG Gui-sheng[3] and FAN Yong-gang[1]
1 School of Software Technology,Dalian University of Technology,Dalian,Liaoning 116620,China
2 International School of Information Science & Engineering,Dalian University of Technology,Dalian,Liaoning 116620,China
3 Shenzhen 7th Technology Co.,Ltd.,Shenzhen,Guangdong 518000,China

Abstract　In recent years,with the continuous progress of 3D printing technology,color 3D printing is becoming the general demand of the industry. However,as the standard file format of 3D printing field model description,STL file does not retain the color information of 3D model,which cannot meet the new requirements of color 3D printing for model information extraction and slice processing. In this context,the OBJ color model is selected as the research object of color slicing technology. Its file structure is analyzed,and the model geometry and color related information stored therein are extracted and optimized for slicing. Combined with the specific characteristics of OBJ model,the color slicing algorithm based on model continuity is proposed on the basis of traditional topological slicing algorithm,and the whole flow of the algorithm is given. The slice processing of the algorithm improves the efficiency of the layered processing,obtains the layered contour information of the model,and completes the processing of the color slice. The experimental results prove that the technology can perform color slice processing on the OBJ model with good effect,stability and reliability.

Keywords　3D printing,Color slicing,OBJ model

1　引言

3D 打印,作为一种快速成型技术,把金属粉末、流体材质、工程塑料等各种类别的可粘合材料,在设计好的数字模型文件的基础上通过计算机软件程序控制逐层堆叠粘接来构建实体模型[1-2]。它基于“离散/堆积”的根本原理,是传统制造工艺的全新升级,有着广阔的应用场景。然而,目前许多 3D 打印工艺系统都没有考虑到数字模型中颜色信息的处理,只是简单地进行几何方面的打印与堆叠,最终的产品模型观赏性通常较差,模型表达也不够清晰,给产品修正带来不小的困难[3]。

彩色 3D 打印是在传统 3D 打印的基础上增加了模型上色的环节,在实体打印的同时为模型的表面添加相应的颜色[4],使打印出来的实体模型呈现出多种色彩。彩色外观使得成形产品有了更直观的表现力,极大地发掘了 3D 打印技术的应用潜能。彩色 3D 打印不仅能够更好地表达出模型的各部分特征,便于产品评估,还能达到更好的视觉效果,满足用户需求。其中彩色切片处理是彩色 3D 打印的关键环节,在这一环节中可以由完整彩色模型得到分层的彩色截面图,

基金项目:国家重点研究计划(2016YFB1101100);国家重点研发计划子课题(2017YFB1107704);中央高校基本科研费(DUT19ZD209)
This work was supported by the National Key R&D Program of China (2016YFB1101100),Subproject of National Key Research & Development Program(2017YFB1107704) and Fundamental Research Funds for the Central Universities(DUT19ZD209).
通信作者:李凤岐(lifengqi@dlut.edu.cn)

以指导实际打印中的上色过程。

目前 3D 行业内应用最广泛的文件格式 STL 只是由模型表面的三角面片定义组成[5]，并不存储任何相关的模型颜色信息，无法应用于彩色切片处理。因此，本文选取 OBJ 模型作为彩色切片研究的文件接口，提取其中包含的模型几何和颜色信息，应用于彩色切片处理。

本文第 2 节描述了 OBJ 文件的基本组成结构及表达方式；第 3 节阐述了对模型进行信息提取与处理的过程；第 4 节提出了基于模型连续性的彩色切片算法，阐明了算法的基本思想与处理方法并展示了该彩色切片技术对模型的处理效果；最后总结全文。

2 OBJ 文件介绍

OBJ 文件是图像业巨头 Alias|Wavefront 为其 3D 建模软件 Advanced Visualizer 开发的一种标准文件格式，文件中包含了 3D 模型的顶点、纹理坐标、法线向量和材质使用等相关信息。一般而言，OBJ 文件的每一行都由前缀字符加上若干相关参数组成[6]。其中，前缀字符表明了这一行所存储的信息种类，参数则是详细的信息表达内容。OBJ 文件的常见前缀如表 1 所列。

表 1 OBJ 文件基本前缀

Table 1 Basic prefix of OBJ file

前缀	含义
v	表示该行指定一个顶点。此前缀后有 3 个单精度浮点数，分别表示该顶点的 X、Y、Z 坐标值
vn	表示该行指定一个法线向量。此前缀后有 3 个单精度浮点数，分别表示其 X、Y、Z 坐标值
vt	表示该行指定一个纹理坐标。此前缀后有两个单精度浮点数，分别表示此纹理坐标的 U、V 值
f	表示该行指定一个表面。一个表面本质上就是一个三角面元。此前缀后的参数格式将在下面具体说明

在 STL 文件中，对于模型中每一个三角面片的定义，都要存储其完整的顶点信息，这样某一顶点的表达就会在多个面片定义中重复出现，造成大量的数据冗余[5]。而 OBJ 文件采用了顶点索引的形式，对于在面片表示中将会使用到的所有空间顶点，先完成这些顶点的定义，并根据其定义的先后顺序，默认赋予每个顶点唯一的索引值（从 1 开始，依次递增）。这样在之后的面片定义中，就可以通过索引值来简单直接地进行顶点表达，其中顶点的引用顺序遵循右手定则[7]。这样的处理方式不仅消除了重复顶点的冗余信息，而且在一定程度上可以反映出模型顶点与面片以及面片之间的几何关系。

对于纹理坐标和法线向量，OBJ 文件也进行相同的处理。而在进行三角面片定义时，将多个顶点、纹理坐标和法线向量的索引依次排列，之间用符号“/”进行隔开，即若干个“v/vt/vn”就可以完成面元的表达。其中后两者在面片的表示中并非是必不可少的，在不需要时可以进行省略。

在 OBJ 文件中，前缀“mtllib”后跟随一个参数，用以指明此 OBJ 文件所引用的材质库的文件路径，材质库的信息被存储为一个独立文件[8]。在 MTL 文件中，对材质进行具体描述，包括该材质的环境色（Ka）、散射色（Kd）、镜面反射色（Ks）等，它们的颜色均以 RGB 形式定义，参数取值范围从 0～1.0。其中散射色、环境色不受视点位置的影响，物体看起来是什么颜色，在很大程度上受到散射色的影响。而对于真实世界中的物体，其散射色与环境色通常是一样的。因此，散射色 Kd 是模型对象直观反映的颜色，即我们通常所指的物体颜色，也称为模型的固有色，这将是彩色切片中要关注的重点信息。

此外，MTL 文件在其中的每一个“newmtl”中都支持对纹理贴图的指定，纹理贴图与 OBJ 文件中定义的纹理坐标共同构建了模型的纹理映射[9-10]。纹理映射可以对映射的相应材质参数进行修改，在物体表面的表现上有很好的灵活性。

3 彩色模型的信息提取与数据结构建立

3.1 基本信息提取

OBJ 模型的颜色信息由附属的 MTL 文件进行存储和表达，MTL 文件中定义了彩色模型相关的许多材质信息，其中包括模型中相应面片的颜色。我们构建材质类 Material 对其有效信息进行提取，Material 类结构为：

```
class Material
{
  public:
    string mtlName;
    float kd[3];
    string map_pic;
    int start,end;
    //其他相关材质颜色结构
};
```

根据 OBJ 模型的组织特点，建立一个顶点向量区 vector〈Vertex〉points 来存储顶点的几何信息。在 OBJ 文件中，三角面片顶点以浮点数形式存储，单位为毫米。而计算机对浮点型数据的计算误差相对较大，所以对切片而言，顶点的几何坐标要以 32 位整型数据变量存储，并结合实际的打印机精度要求，保留到原浮点数的小数点后三位，也就是精确到微米级。

在信息提取的过程中，当判断出某一行为顶点信息（即前缀字段为“v”）后，读取该行的后续 3 个浮点型数据类型，分别乘以 1 000 后强制转换为 32 位整型，存入 Point3 点类的对象中，并将其作为 Vertex 类对象的成员。

对于可能存在的纹理等索引信息，进行相似的处理，在此不再赘述。

由于不同建模软件生成 OBJ 文件的精度存在差异，以及后续切片过程的要求，并不是 OBJ 中三角面片所有的空间顶点信息都要存入系统内存中的顶点向量区中，还需要进行一定的顶点信息优化与模型缺陷检测。模型优化以三维空间的区域划分为基础，使得每一个小的空间区域内只有一个顶点存在，以简化模型信息。OBJ 文件以索引形式来表达三角面片信息，其中的索引是指顶点在文件中被定义的索引位置。在读取到面片信息时，应当将原始索引转换为其在顶点向量区的实际索引，并进行重复性检测，再加入面片信息区 vector〈Face〉faces 中。

面片的重复性检测是为了保证模型优化后面片信息的有效性，具体处理方法为：对于提取到的面片信息，根据其更新后的索引判断有无重合顶点，若没有，则通过顶点对象中的 faceIndexList 查找其每个顶点所在的面，查找其中是否有包含该面的所有 3 个顶点的面存在，如果没有，就将该面加入到

vector〈Face〉 faces，并更新其 3 个顶点类对象中的 faceIndex-List 信息。

3.2 颜色信息处理

对于通过了重复性检测，要加入 vector〈Face〉 faces 进行后续切片处理的面片，我们需要获得其相关的颜色信息，颜色信息的处理与上一节中对面片的重复性检测与优化同时进行。OBJ 模型中的颜色信息主要有两种表达方式：1)通过散射色 Kd 来指定，并应用到面片上，作为面片的颜色；2)以纹理贴图的形式来进行面片渲染。图 1(a)、图 1(b)分别为无色的三维模型与以面片颜色表示的彩色保时捷模型，图 2(a)、图 2(b)则为无色模型与添加纹理渲染后的彩色面具模型。

为了进行彩色切片相关的颜色提取与表达，可以建立一个颜色索引区。针对 OBJ 模型中颜色表达方式的不同，颜色索引区也有两种存在形式，分为面片索引与顶点索引，由一个全局变量所标识。颜色索引区中存储颜色类 Color，其中以整型变量表达 0～255 形式的 RGB 颜色值。

(a)无色模型 (b)彩色 OBJ 模型

图 1 保时捷模型展示

Fig. 1 Car model display

(a)无色模型 (b)彩色 OBJ 模型

图 2 面具模型展示

Fig. 2 Mask model display

在没有应用纹理贴图的情况下，模型中的颜色信息通过固有色 Kd 来指定，并应用到面片上，作为面片的颜色。因此只需建立与面片索引一一对应的颜色索引区，将面片对应的 Material 材质对象的“Kd”以 0.0～1.0 的浮点数形式存储的颜色转换为 0～255 的 RGB 形式，加入颜色索引即可。

而对于使用了纹理贴图的 OBJ 模型，模型实际体现出来的表面颜色由贴图所确定。在这种情况下，OBJ 文件中会有以“vt”前缀来定义的纹理坐标。与顶点几何坐标相同，纹理坐标同样以索引形式被应用于面片定义，其纹理坐标与顶点坐标一一对应。几何坐标表明了顶点在模型空间中的位置，而纹理坐标则决定纹理对象的哪一个纹素被赋予该顶点[11]。

因此我们需要载入面片对应材质的纹理图片，通过纹理坐标找到图片中相应的像素点，该点的颜色信息即为 OBJ 模型中对应三角面片顶点的颜色信息。但此时还存在一个问题：纹理中的纹素位置是以整数形式指定的，而我们的纹理坐标是 0～1 之间的浮点数，因此可能会存在(0.79,0.54)这样的纹理坐标，映射到 512 * 512 的纹理对象上，就得到了(404.48,276.48)的纹素位置，无法直接对其进行颜色获取。在这里我们选择线性滤波的方法，根据纹素位置(404.48,276.48)周围的多个纹素值来确定最终的目标纹素值，如使用((404,276),(404,277),(405,276)和(405,277))这 4 个位置纹素值的线性插值，将纹理坐标映射到非整数点的纹素位置，此线性插值通过对目标点周围纹素值的加权平均来实现。

由此，我们就可以将该三角面片的三个顶点对应到其确定的 RGB 值，并记录在相应的颜色索引位置。综合上述的信息提取处理过程，该处理的基本流程如图 3 所示。

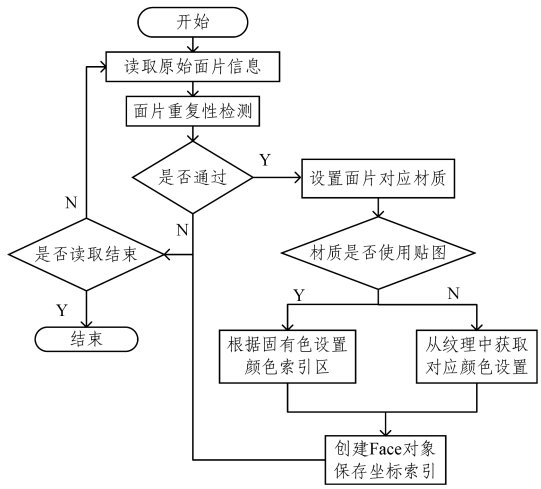

图 3 颜色信息处理流程

Fig. 3 Flow chart of information extraction

4 彩色切片处理与应用

STL 文件格式在 3D 打印领域被普遍应用，由于传统的切片算法大多是基于对 STL 模型的处理提出的，实际上也就是针对三角面片边界模型的，因此对于 OBJ 模型而言同样适用，这也是选用 OBJ 模型作为彩色切片文件接口的优势之一。

目前的等厚切片算法大致可分为两大类：基于模型几何特征的切片算法[12-13]和基于模型拓扑信息的切片算法[14-15]，二者都是通过对原始模型的预处理来提高切片效率。本文结合彩色切片的应用场景与具体需求，在基于模型拓扑信息切片算法的基础上，充分利用 OBJ 文件的组织特点，提出了基于模型连续性的彩色切片算法，来对 OBJ 彩色模型的切片进行处理。

4.1 基于模型连续性的切片算法

4.1.1 拓扑结构的建立

模型的拓扑信息结构主要包括两个方面：1)各个顶点所在面的拓扑关系；2)各个面的邻接面之间的拓扑关系。通过这两个拓扑关系，可以由当前相交面片快速确定邻接面片，计算相关交点。对应用于 STL 模型的拓扑切片算法而言，建立模型拓扑信息后，分层的交点计算和轮廓建立过程都得到了大幅简化，其主要开销在于整体拓扑结构的建立。而对于 OBJ 模型，这一开销可以被大大降低。

OBJ 文件中，顶点以索引形式在面片定义中被应用，这实际上已经隐含了顶点与面片之间的拓扑关系。在 3.1 节中对模型的基本信息进行提取时，我们通过顶点类对象的 vector〈uint32_t〉faceIndexList 变量直接记录了该顶点的所在面信息。

而对于构成某条边的一对顶点来说，在建立了顶点与所在面的拓扑关系之后，其中每个顶点单独所在的所有三角面片集合中，必定有两个是共有的，其中一个是当前面，另一个就是所求的邻接面[16]，由此便建立了面的邻接面的拓扑结构。

可以看出，对于 OBJ 三维模型来说，其整体拓扑结构建立的过程非常简单迅速，而这也直接影响到切片处理的效率。

4.1.2 算法思想方案

对最初的拓扑切片算法而言，每层的切片过程都需要对模型的所有三角面片进行遍历，以确定其与切平面是否有交线产生，并通过逐层的完全遍历来实现切片处理。在这个过程中，对于每一个切平面高度的切片处理，都可以得到一个与当前切平面相交的三角面片集合。通常来说，这样的面片集合只占据模型全部三角面片的集合的很小一部分，因此在对面片进行完全遍历时就产生了大量无效判断，从而影响了算法的整体效率。可以想象，如果在切片求交处理之前就找到了这个面片集合，无疑会有效地提升算法的性能。

模型的连续性，在切片过程中实际表现为与切片面相交的三角面片集合的连续性以及切层轮廓的连续性[17]。由于 3D 打印技术的精度要求，在对三维模型进行切片时，一般而言，分层的层数要足够多，相邻分层的高度差也相对较小，因此相邻的切平面之间存在一定的联系；而由于已经建立了模型的拓扑关系，交点计算中可得到确定的交线段，每个切平面对模型切片得到的切层轮廓也一定是有向连续的。

我们将与某一高度为 $Z=z_i$ 的切平面相交的三角面片集合记作 F_i，与紧接着下一层高度为 $Z=z_{i+1}$ 的切平面相交的三角面片集合记作 F_{i+1}，由于模型的连续性，除少数切平面外，与相邻切平面相交的三角面片集合都相差不大。也就是说，F_i 中的大部分元素都会在 F_{i+1} 中出现。

在进行高度为 $Z=z_{i+1}$ 的切平面切片时，我们称集合 $F_c=F_i$ 为候选集，该候选集可通过上一层进行的拓扑切片来确定。在理想状态下，得到了该候选集之后，从其中便可以快速地查找到和当前切平面有相交关系的三角面片，作为该层轮廓的起始面；由于切层轮廓的连续性，根据拓扑关系从该起始面出发，即可得到一个分层轮廓。而在这一拓扑查找的过程中，我们自然而然可以确定与下一个切平面相交的面片，进行了集合 F_{i+1} 的构建。构建出完整的 $F_c=F_{i+1}$ 便是一个新的候选集，可以用于后一层的起始面查找。

相比模型全部三角面片的集合 F，候选集 F_c 只是从属的非常小的子集。根据这样的思想，我们可以通过对三角面片进行一次初始的完全遍历，对设置好的各切平面高度的三角面片数目做一个简单统计，同时根据拓扑切片得到 F 的子集 $F_c(F_0)$ 为候选集；对下一层切片的起始面进行查找，只需遍历候选集 F_0 即可，根据遍历结果可得到新的候选集 $F_c(F_1)$…依次向下进行，便可以逐层完成切片处理工作。

实际上，在统计得到了各层相交三角面片数目之后，甚至可以不完全遍历候选集，只要当前层已加入新候选集的面片数目满足统计数目，即可结束本层的切片处理，这说明与当前层相关的所有面片都得到了处理。通过这种方法，可以缩小与切平面相交三角面片的查找范围，达到与基于模型几何特征切片算法相似的效果，综合了两类切片算法的优点。

当然，在某些极端情况下，也有可能出现两层面片集合 F_i 与 F_{i+1} 完全无关的情况，即 F_i 和 F_{i+1} 不存在交集，无法从候选集 $F_c(F_i)$ 中找到与下一层切平面相交的三角面片来进行拓扑切片并构建出新的 $F_c(F_{i+1})$。我们称从候选集 $F_c(F_i)$ 中找到相关面片并由此成功构建出 $F_c(F_{i+1})$ 的情况为“命中”，上述情况为“未完全命中”。除此之外，对一些特殊模型而言，还可能出现“部分未命中”的情况，也就是根据 F_i 得到了 F_{i+1} 的一个子集，而 F_{i+1} 中存在一部分与 F_i 无关的三角面片集合，与切平面相交构成了一个新的独立轮廓。我们将“完全未命中”与“部分未命中”的情况统称为“未命中”，在未命中的情况下，仍需要遍历所有三角面片，以确保切片处理的有效性与完整性。

实验结果表明，在对一般三维模型做切片处理时，候选集“未命中”的情况实际上很少发生，命中率通常在 70%以上，处理效率相比传统算法有明显的提升。

算法的基本流程如图 4 所示，算法的具体描述如下。

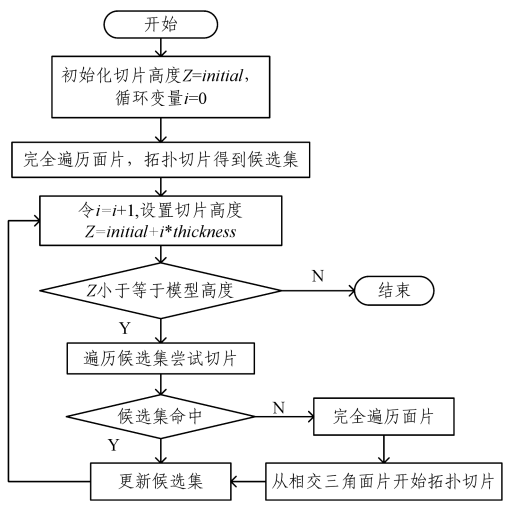

图 4 算法基本流程

Fig. 4 Basic flow of algorithm

输入：优化后的模型信息数据结构 OptimizedModel，初始切片高度 initial，分层厚度 thickness

输出：切片得到的轮廓信息 Polygons

Step1 初始化切平面高度为 Z=initial，循环变量 i=0；计算分层数目 total，设置 faceNum[total]数组元素为 0，完全遍历面片，统计各层相关面片数目记录在对应数组位置，并进行拓扑切片，得到候选集 $F_c=F_0$，并执行下一步。

Step2 令 i=i+1，提升切平面高度至 Z=initial+i * thickness。若 Z 小于等于模型高度，设置集合为空，执行下一步；否则结束处理。

Step3 从候选集 F_c 中按序查找与当前切平面相交的面片。若找到符合要求的面片，执行下一步；否则转 Step5。

Step4 以上一步中查找到的面片为起始面，根据拓扑关系进行切片，并在集合 F_i 中记录相切的三角面片索引。如果 F_i 中的面片数已满足统计数据 faceNum[i]，更新候选集 $F_c=F_i$，

转 Step2；否则转 Step3。

Step5 在当前切平面高度完全遍历三角面片，按序查找到的与当前切平面相交的面片，执行下一步。

Step6 以上一步中查找到面片为起始面，根据拓扑关系进行切片，并在集合 F_i 中记录相切的三角面片索引。如果 F_i 中的面片数已满足统计数据 faceNum[i]，更新候选集 $F_c = F_i$，转 Step2；否则转 Step5。

4.2 彩色切片

在上述处理中，模型颜色信息被统一转化存储在颜色索引区，有面片颜色索引与顶点颜色索引两种存在形式。OBJ 模型以面片固有色的形式定义颜色时，整个交线的颜色就是其所在面片的颜色；而在模型以纹理贴图的方式来定义颜色时，此时面片的三个顶点各有其从贴图上取到的颜色值，于是可以根据交点所在边的两个顶点的颜色，通过对其进行线性插值的方式来计算交点颜色。

按照直观的想法，对一个三角面片与相关切平面相交，得到一条交线段，此交线段有起点终点两个端点，附带各自的颜色信息[18]。而对以面片颜色索引形式表达的颜色信息，某条交线上的交点的两个相邻面片上可能有不同的表达；同时由于拓扑信息的存在，在对前后两个邻接的面片进行求交时，前一个交线段的终点与后一个交线段的起点实际上是在两个面片公共边上的同一个点，如果都进行计算存储，则对切片效率的影响较大。

因此我们实际可以把一个面片与切片面相交得到的后向信息（即交线段终点信息）作为该面片所确定的切层轮廓信息，建立类 Intersection 记录其几何坐标与相应颜色。

由此得到了切片轮廓交点的颜色信息。由于模型的封闭性与拓扑关系的有效性，便可以得到完整的彩色轮廓，轮廓类由交点的集合来表示。

对于上节中展示的两个 OBJ 彩色模型，其切片效果分别如图 5、图 6 所示。

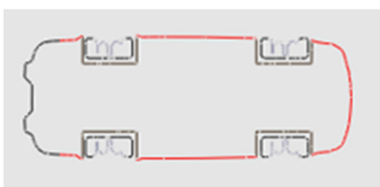

图 5 彩色轿车模型切片效果

Fig. 5 Slicing effect of color car model

图 6 贴图渲染的面具模型切片效果

Fig. 6 Slice effect of mask model in map rendering

结束语 本文针对 3D 打印行业发展对于彩色切片技术的新需求进行了相关的研究工作。文中选择 OBJ 模型作为彩色切片研究的数据接口，详细分析了其与彩色切片过程相关的文件组成；并提取优化了其中有效的几何和颜色信息，建立起了完整而简单的数据结构。之后，在分析传统 STL 模型切片算法的基础上，充分利用 OBJ 模型与 STL 模型的相似点与 OBJ 模型的自身特点，对模型拓扑切片算法进行了改进；提出的基于模型连续性的彩色切片算法，在继承拓扑切片算法优点的基础上，有效地提升了切层相关面片的查找效率，达到了良好的应用效果。实验结果证明，该切片技术处理准确、效果良好，具有相当的实用意义。

参 考 文 献

[1] HALEEM A, JAVAID M. Additive Manufacturing Applications in Industry 4.0: A Review[J]. Journal of Industrial Integration & Management, 2019, 4(4): 1930001.

[2] SINGH O P, AHMED S M, AHMED A M, et al. Modern 3D Printing Technologies: Future Trends and Developments[J]. Recent Patents on Engineering, 2015, 9(2): 129-135.

[3] WANG F Y. From social computing to social manufacturing: the coming industrial revolution and new frontier in cyber-physical-social space [J]. Bull Chin Acad Sci, 2012, 27: 658-669.

[4] CHENG Y L, CHANG C H, KUO C. Experimental study on leveling mechanism for material-jetting-type color 3D printing [J]. Rapid Prototyping Journal, 2020, 26(1): 11-20.

[5] YANG G, LIU W J, WANG W, et al. Research on the Rapid Slicing Algorithm Based on STL Topology Construction[J]. Advanced Materials Research, 2010, 97(101): 3397-3402.

[6] WANG Y G. 3D model file input and processing in OpenGL in 3ds and OBJ format[J]. Electronic World, 2013(6): 86-87.

[7] WANG J F, YAO G Q. Input and processing of OBJ format in OpenGL[J]. Computer Knowledge and Technology, 2011 (10): 187-190, 193.

[8] LU Z W. Importing and Processing of OBJ Model Files in OpenGL [J]. Scientific and Technological Information, 2013(7): 94-95, 124.

[9] WANG B, SUN W. Research on texture mapping method of new OBJ file based on OpenGL[J]. Computer and Digital Engineering, 2015(8): 135-138.

[10] REN S, WANG Z, XU Z C, et al. An information hiding algorithm based on texture mapping of OBJ 3D model [J]. Journal of Beijing University of Posts and Telecommunications, 2019(1): 127-129.

[11] OLIVEIRA M M, BISHOP G, MCALLISTER D. Relief Texture Mapping[J]. ACM SIGGRAPH Computer Graphics, 2000(6): 359-368.

[12] XU H W, JING W H, LI M J, et al. A slicing model algorithm based on STL model for additive manufacturing processes[C]// Proceedings of 2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC 2016). 2016: 1647-1650.

[13] LI Z L, LIANG D, LI D C, et al. Research on fast hierarchical processing algorithm based on information inheritance [J]. Journal of Xi'an Jiaotong University, 2002, 36(1): 43-46.

（下转第 280 页）