**基于三维模型体素化与重采样的三角形网格LOD模型生成算法**

**华南理工大学 练孙鸿**

**摘要** 在计算机图形学中，三维物体常常以三角形网格的形式表示。在不同的情景下渲染不同精细度的模型可以提高渲染效率。细节层次（LOD，Level of Detail）模型就是一系列不同精细度的模型。本文提出一种闭合网格模型的LOD模型生成算法，该算法先使用本文提出的方法对给定网格模型进行体素化（Voxelization），得到指定分辨率的体素（Voxel）模型。然后对体素模型进行一定分辨率的重采样之后用Marching Cubes算法来重构出不同细节程度的三角形模型。本文算法有如下特点：可以通过调节体素化分辨率与重采样分辨率，在LOD模型生成质量与生成速度之间进行权衡；生成的三角形网格比较均匀；能在其他场景应用，例如CT切层扫描数据、3D扫描数据的网格LOD模型生成。

关键词：LOD 体素化 重采样 网格重构 MarchingCube 网格简化

1 引言

在游戏、科学可视化等计算机图形学的应用中，渲染效率与渲染质量是都是要兼顾的。当前大部分的计算机性能还不不能以较快的速度渲染海量三角形。经过精心制作的模型，或者3d扫描仪扫描并重构得到的模型一般都有几十万、几百万甚至更多的三角形。

但不是所有时候都需要用最高精度的模型。例如，同样的模型处在远处时成像会比较小，从而很多细节就被忽略了，这时候高精度的模型就会浪费计算性能。这时候如果在远处的能用更低精度的模型渲染，那么在不太影响渲染质量的情况下可以提高效率。这就需要用网格简化算法来生成更低细节层次（Level Of Detail，LOD）的模型，于是可以在不同情况下选择不同细节层次的模型来渲染。所以网格简化算法与LOD模型在科学研究与工业界中均有很大用处。

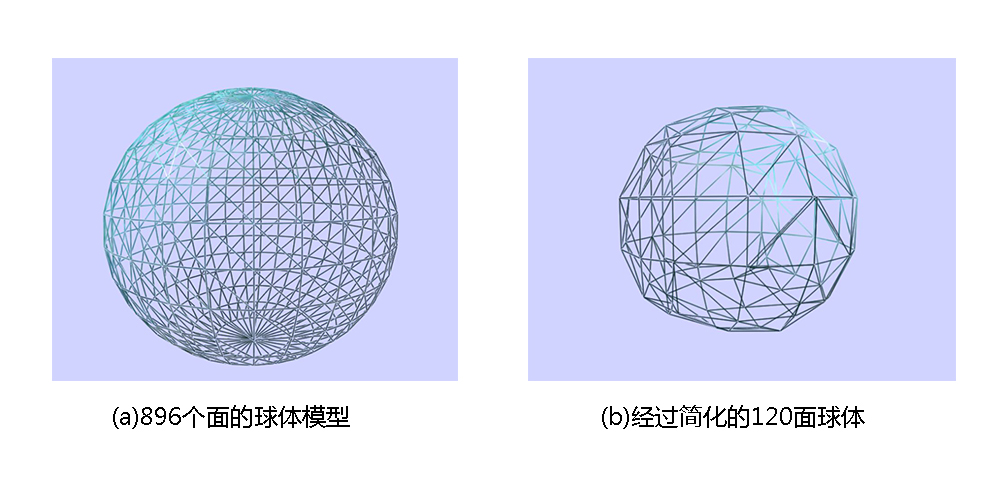


图1：同一个模型不同细节层次的网格，(b)由本文算法生成

网格简化算法在国内外已有不少研究成果。网格简化有很多种思路与方法。Rossignac[1]提出用顶点聚类消除多余顶点，这是一种比较简洁快速的算法，但效果不一定理想。周昆[2]用八叉树自适应划分对此算法进行改进，Kanaya[5]也给出了保持拓补(topology-preserving)的改进。Maria-Elena[3]提出在一种使用边坍缩来简化网格的方法。当然还有不少网格简化的改进与变体，但是大部分算法的思路在[4]中有了精炼的概括：顶点聚类；合并共面三角形；受控的顶点、边、三角形删除(decimation)；基于能量函数的优化。

本文提出一种新的思路：对网格进行**体素化(Voxelization)**，然后进行重采样（一般是降采样），然后用**等值面提取算法（特指Marching Cube[8]算法）重构出三角形网格**。考虑到Marching Cube算法的特点，该算法需要实心的、经过填充的体素模型，所以这种思路会更适合闭合的三维网格模型，因为本文提出的体素化算法只能用在**闭合网格模型**上。

本文先提出一种基于切层与扫描线填充的闭合模型的**表面与内部体素化**方法，由这个体素化方法得到指定分辨率的、经过内部填充的体素模型。关于体素化算法的工作，[7]中给出了基于八叉树体素化（相当于光栅化的三维推广）每个三角形方法，进而实现体素化整个网格模型的功能。[6]先从体素化三角形开始，再用种子填充算法填充模型内部体素，从而得到实心的体素模型。[10]中提出与本文体素化算法类似的思路，先用一系列切片去切割模型，而且用了GPU加速与硬件自带的光栅化功能去实现体素化，但是这样的原理并不能填充模型内部。

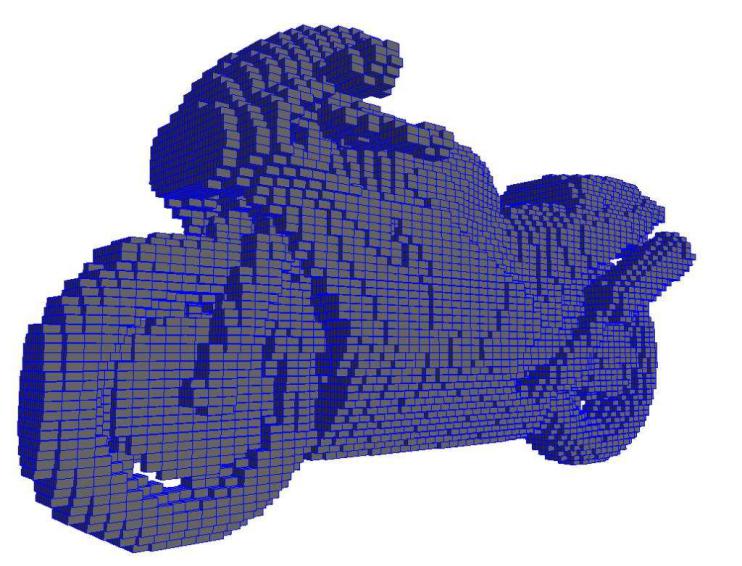


图2：体素模型，用大量的体素(voxel)去逼近物体

2 算法描述

2.1 坐标系约定

本文的所有几何描述都在三维笛卡尔坐标系（左手系）下，其中y轴垂直于水平面，xz平面水平。

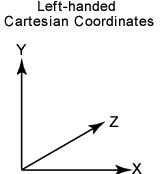


图3：本文约定使用的坐标系

2.2算法流程与应用场景

本文算法的大致步骤如图4所示，大体上有三个阶段：

* 原始的闭合网格模型
* 经体素化后的体素化模型
* 经过体素模型重采样、网格重构后的简化/细分网格模型（LOD）

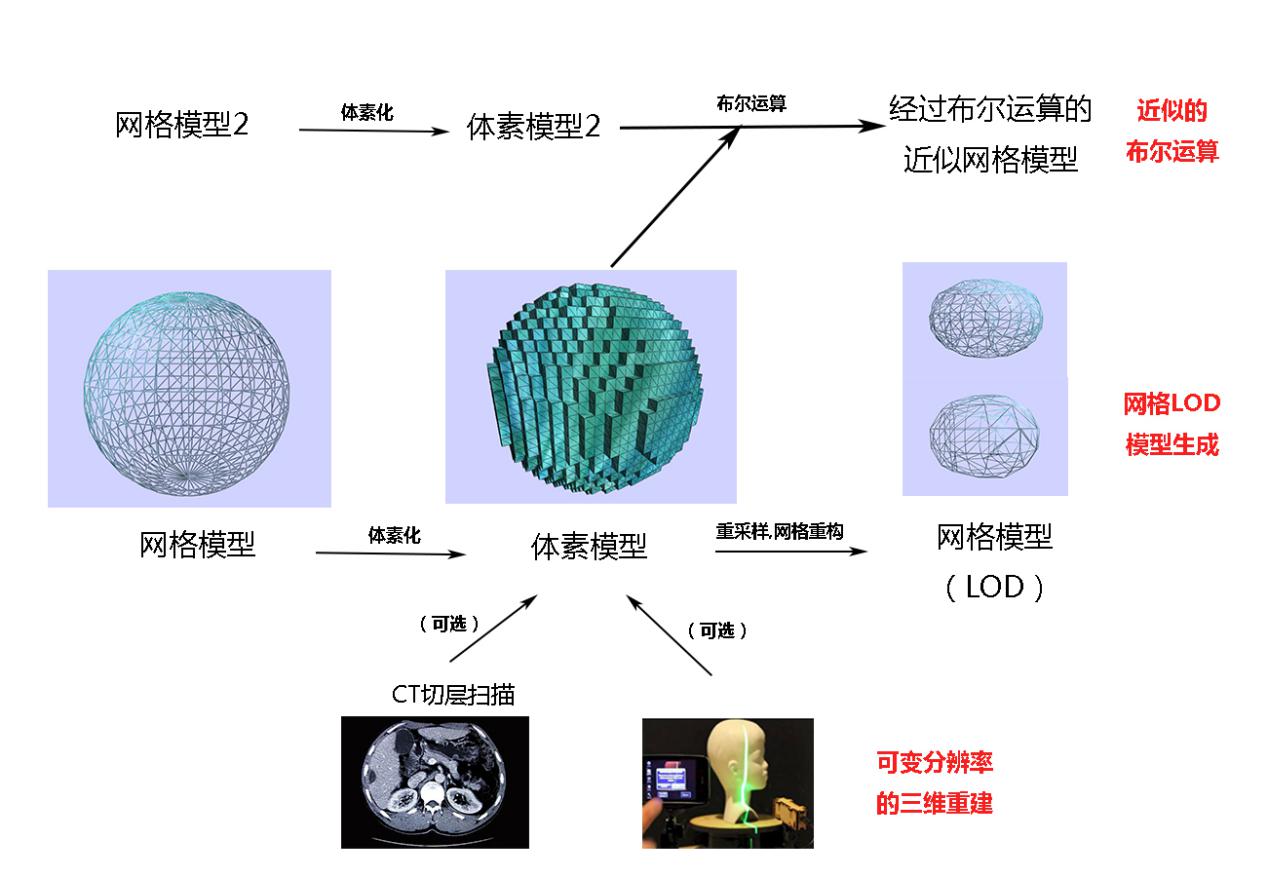


图4：本算法的大致流程以及其他应用场景

有一点是值得注意的，本文算法的应用场景不只是能用于闭合网格模型的简化，还可以有其他方面的应用，例如：

* CT/MRI切层扫描数据的曲面重建：CT/MRI等医学扫描数据可以很直观地转化为体素模型（处在扫描切层上的像素可以直接转换为体素），继而可以用本算法的“重采样、网格重构”直接生成网格LOD模型。
* 3D扫描数据：一般的3D扫描得到的原始数据就是点云。点云经过处理（例如用体素化的三维栅格来滤波）以后，也是比较容易转化成体素模型。因为扫描数据一般比较密集，要可视化扫描到的物体一般都要求进行简化。所以用本算法的“重采样-网格重构”可以一步到位生成简化的网格，而不是先重构再简化。
* 曲面细分：虽然体素模型重采样阶段一般做得都是降采样，但是重构之后的三角形却有可能比原来的多，所以此算法有潜力实现真正的曲面细分。但是问题在于，原网格模型是原始数据中最精确的表示了，不断的降采样只会减少信息。本算法生成的新网格即使面数更多，其表面曲率信息也是不经过插值的，不比原模型的表面信息多。所以算法虽然可以生成更密集更均匀的网格模型，但还不能做到平滑的曲面细分。
* 近似的网格布尔运算：三维网格模型的布尔运算也是一个很复杂的问题，网格模型直接进行并、交、差等问题必定涉及大量的几何分析与特殊情况的处理。文献[11]中就直接用精确的几何求交然后进行三角化相交边缘来得到进行了布尔运算的模型。而本文算法经稍加修改，即可实现一个网格的近似布尔运算的方法。体素模型的布尔运算（并、交、差）非常的直观，而且这些运算都可以用c++的计算位运算符实现，快速且方便。但是这种方法原理上比直接用几何求交的算法更粗糙一点，只能得到近似的结果，但是体素模型布尔运算再重构网格直观上讲就比较的健壮。

限于主题，本文接下来的部分只介绍基于体素化与重采样的**网格模型简化**算法。

2.3体素化

体素（Voxel）[9]，体积元素（Volume Pixel）,是数字数据在三维空间上分割最小单位。在本文里体素含义类似，特指逻辑上不可以分割的正方体单元，但其尺寸是我们不关心的问题。如何把一个三角形网格模型转化为由体素逼近表示的模型，就是体素化（Voxelization）要讨论的问题。为了节省内存，**本文规定体素是二值化的，即一个体素只能取0或者1两个值**。

本文的体素化算法分成两个步骤：

* 切层，即用平行于XZ平面的平面移动切割原模型
* 光栅化，基于上一步得到的结果进行光栅化与内部填充。

把一系列都经过光栅化与填充之后的切层组合起来就得到了体素模型。

下面是一些符号的规定：

**规定1** 顶点表示为实数的三元组

**规定2** 线段表示为实数的三元组

**规定3** 三角形表示为顶点的三元组

**规定4** 三角形表示为顶点集与三角形集的二元组

，



其中VertexCount为网格的顶点个数，TriangleCount为网格的三角形个数。

**规定5** 为几何对象obj（线段、三角形、三角形网格）可以取到的最小x坐标。

**规定6** 为几何对象obj（线段、三角形、三角形网格）可以取到的最大x坐标。

**规定7** 与规定5，6类似，同理定义，，，

**规定8** 在规定5，6，7的基础上，定义:







**规定9** 设经过网格体素化得到的体素模型的在x,y,z方向上的分辨率分别为，即生成的体素模型会有个体素。

**规定10** 线性插值函数：。一般情况下a,b指的是实数。若a,b指的是同样维度的向量时，两个向量的按分量逐个进行线性插值。

2.3.1切层

本文提出的体素化算法借鉴了“3D打印”的思想。3D打印利用很多层有一定厚度的材料来逼近目标模型。在这里，有一定厚度的3D打印的材料线就可以抽象为一系列体素的集合。

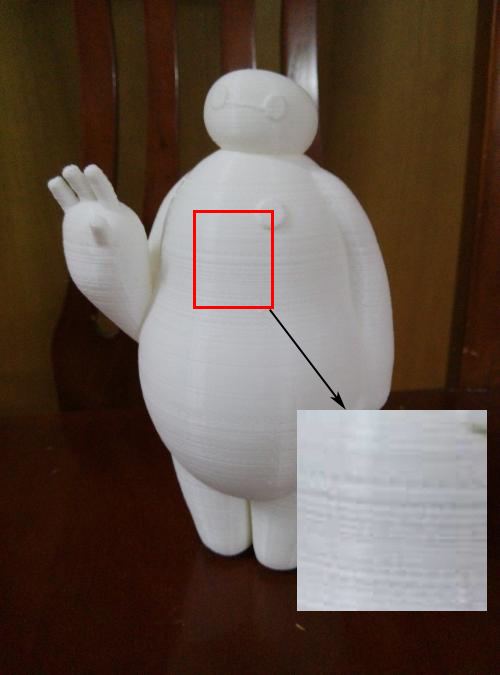


图5：3D打印模型，局部放大可以明显看到其分层叠加的树脂

所以体素化算法的第一步可以描述为：

**问题1** 给定闭合曲面，其y坐标取值范围为。用个平面来与曲面相交，求这j次相交得到的隐式曲线方程集合。

换句话讲，我们分别用个水平面去切割模型，每一次切割就是一次平面与闭合曲面求交的过程。显而易见的是（但证明起来有难度），第j次切割得到的结果会是**一条或多条封闭曲线**（由隐式曲线方程来表示）。

闭合的三角形网格是分片光滑的曲面，也可以用来表示，而且每个分片都是三角形，是一种用计算机去逼近描述曲面的不错的形式。所以问题1的求解可以采用分而治之的思路解决，即只需分别考虑每个三角形与平面的相交情况，然后再把结果综合起来。

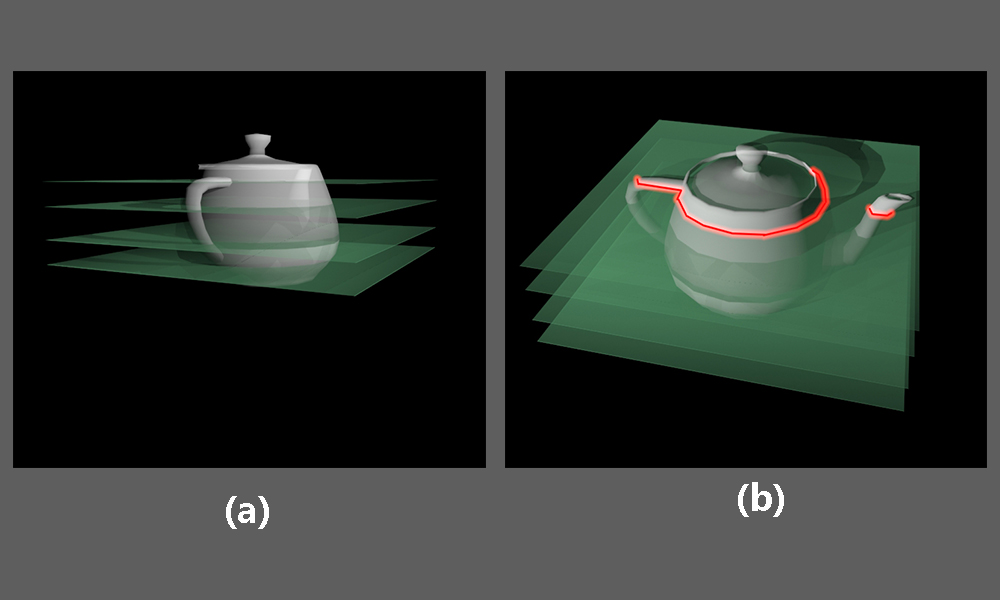


图6：用多个平面去切割三角形网格，(b)中红色标记的线为切割得到的线段集合

设单个三角形T与平面的一次求交过程为：



显然Segment(T,y)的返回值要么是只有一个元素的线段集合（水平面与三角形T相交线段），要么就是空集（水平面与三角形T不相交）。那么问题1中描述的曲面可以限制为三角形网格以后，问题可以转换为如下描述：

**问题2** 给定三角形网格，用多个相等间距的水平面与求交，求所有相交得到线段的集合，其中



问题2求解出来理论上会得到很多层的线段集合，线段集合在外观上应该也是一个或多个封闭图形的轮廓。

这样的问题描述就意味着求解时写的程序必须有两重循环了，而且要把所有三角形与切割面都配对求交，时间复杂度为。考虑到一般情况下大部分三角形T的y坐标范围都远小于网格模型的y坐标范围，所以可以减少遍历的切割面y坐标范围，交换循环的嵌套顺序，把不可能与三角形T相交的切割面预先排除掉。设比y坐标小的最近切割面标号（按y坐标从小到大给切割面标号）为，有



则可以优化表示为：



实验证明改变循环嵌套顺序并减少相交次数的优化之后，算法运行时间明显减少。

现在只需要来讨论的具体实现，即求三角形与平面相交所得线段。

其实平面与平面相交实现起来还是一件比较繁琐的事情。幸运的是，这里的相交算法规定了无限大的切割面是平行于XZ平面的，多了一个这样的限制条件将会使得推导更加容易。下面我们进行分类讨论：

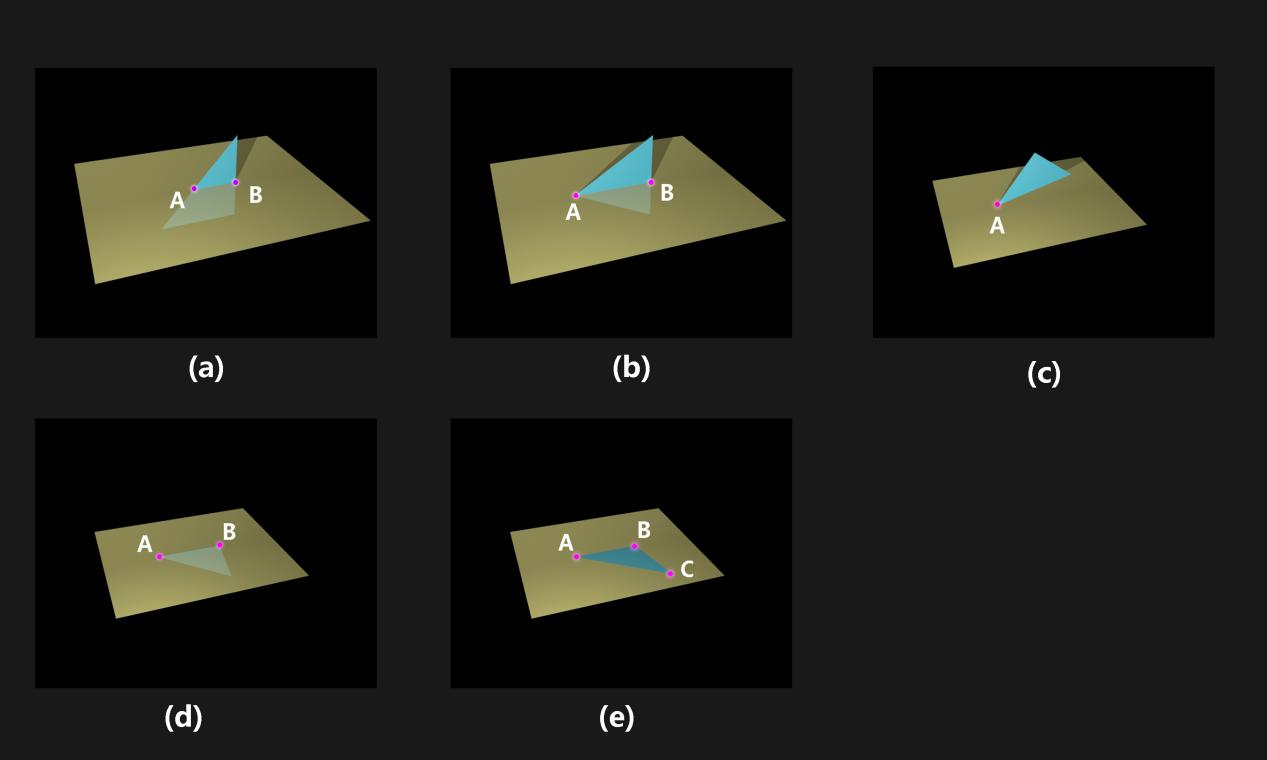


图7：单个三角形与水平面相交的几种情况，按三角形恰好在平面上的顶点数分类

看到图7分了五种相交情况，按照“恰好落在切层上的三角形顶点数”来分类，先排除了平凡的不相交情况。设在线段上的、给定y坐标的点的坐标可以由函数求得：



其中，，注意这是开区间，特意把线段的两个端点排除。这是给了下一小节扫描线光栅化的方便，统一三角形与切层相交得到的线段端点个数的奇偶性。

设切层的y坐标为，下面给出三角形与切层相交线段的求解方法：

**(a)**没有顶点落在切层上。设三角形与切割面相交的两条边为和，则相交线段为

**(b)**恰有一个顶点落在切层上，顶点所对的三角形边与切层相交。设三角形与切割面相交的那一条边为，落在切层上的三角形顶点为则相交线段为

**(c)**恰有一个顶点落在切层上，顶点所对的三角形边与切层不相交。则这种情况不生成相交线段。

**(d)**恰有两个顶点落在切层上。设落在切层上的两个顶点为，则相交线段为

**(e)**所有顶点都落在切层上。这种情况其实可以不产生任何线段，因为体素化算法的基础要求就是曲面闭合，这就意味着必定有其他三角形的某些边与这个三角形的三条边共边。

综上，三角形与切割平面的相交线段的求解函数由下表给出：

|  |  |
| --- | --- |
| **线段集合** | **条件** |
|  | 情况(a) |
|  | 情况(b) |
|  | 情况(c) |
|  | 情况(d) |
|  | 情况(e) |
|  |  |

表格1：Segment(T,y)的实现细节

至此，“切层”得以实现，我们现在就有了由线段组成的模型切片外轮廓了。

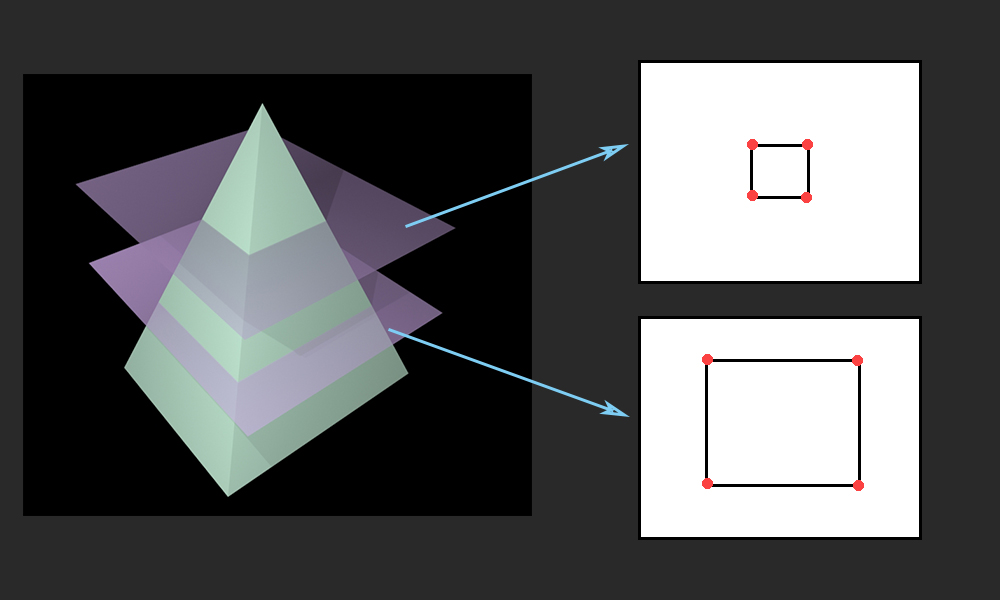


图8：切层算法的输出，每个切片都有一个切割轮廓，由一系列线段组成

2.3.2扫描线光栅化

经过2.3.1-切层的处理之后，原始网格模型的体素化问题就被降维成了“二维矢量多边形的光栅化及二值化填充”问题。

[12]光栅化(Rasterization)，其实就是把矢量图形转换为像素点的过程。并且如果要填充多边形的内部，那就要求使用多边形填充算法。一般的多边形填充算法有基于扫描线[13]的算法，有基于种子填充的算法[14]，还有两者结合的算法[15]。很多文献都介绍了基于这些基本填充方法的改进算法。本文的扫描线光栅化填充算法在原理上没有大的突破，但是**做了一点修改用于解决扫描线填充的一种二义性情况**。

2.3.3本文体素化算法部分实验结果

2.4重采样

（三次线性插值）

2.5基于Marching Cube算法的网格重构

（三角形数可能会变多，网格密度更大，但是曲率不会变得更平滑，原网格就是网格描述精细度的上限，所以理论上来讲不能实现更平滑曲面细分）

（边上线性插值系数求解方法，可能的改进（最小二乘求解））

3 实验结果

3.1 不同体素化三维分辨率的对比（速度、质量）

时间与分辨率的关系

3.2 不同的降采样三维分辨率的对比

时间与分辨率的关系

其实3.1 3.2可以合在一起搞，但是简化效率与简化质量两个指标都得讲，两者的权衡关系

3.3算法缺点，采样分辨率过低可能产生的后果（兔子耳，mc）

4 结语

算法大致概括。

Mc边的求解，以后能用个最小二乘什么的，求解完就专门存着“边信息”，免得每个cube生成的等值面不连续。/本算法的其他的应用

参考文献

[1]Rossignac J, Borrel P. Multi-resolution 3D approximations for rendering complex scenes[J]. Journal of Trauma & Dissociation the Official Journal of the International Society for the Study of Dissociation, 1993, 7(1):5-18.

[2]周昆, 潘志庚, 石教英. 一种新的基于顶点聚类的网格简化算法[J]. 自动化学报, 1999, 25(1): 1-8

[3]Maria‐Elena A, Francis S. Mesh Simplification[C]. Computer Graphics Forum. Blackwell Science Ltd, 1996:77-86.

[4]Cignoni P, Montani C, Scopigno R. A comparison of mesh simplification algorithm[J]. Computers & Graphics, 1997, 22(1):37-54.

[5]Kanaya T, Teshima Y, Kobori K I, et al. A topology-preserving polygonal simplification using vertex clustering[C]// International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia 2005, Dunedin, New Zealand, November 29 - December. DBLP, 2005:117-120.

[6]吴耕宇, 潘懋, 郭艳军. 利用几何求交实现三角网格模型快速体素化[J]. 计算机辅助设计与图形学学报, 2015, 27(11):2133-2141.

[7]吴晓军, 刘伟军, 王天然. 基于八叉树的三维网格模型体素化方法[J]. 图学学报, 2005, 26(4):1-7.

[8]Lorensen W E, Cline H E. Marching cubes: A high resolution 3D surface construction algorithm[J]. Acm Siggraph Computer Graphics, 1987, 21(4):163-169.

[9]百度百科-体素. <https://baike.baidu.com/item/%E4%BD%93%E7%B4%A0/8945761?fr=aladdin.>2017.8.27

[10]谭皓, 杨忠, 李玉峰,等. 基于硬件加速的真三维显示体素化算法[J]. 南京航空航天大学学报(英文版), 2006, 23(1):59-64.

[11]毕林, 王李管, 陈建宏,等. 三维网格模型的空间布尔运算[J]. 华中科技大学学报(自然科学版), 2008, 36(5):82-85.

[12]wikipedia-Rasterization. <https://en.wikipedia.org/wiki/Rasterisation> 2017.8.30

[13]甘泉. 通用扫描线多边形填充算法[J]. 计算机工程与应用, 2000, 36(2):57-59.

[14]陈元琰, 陈洪波. 一种基于链队列的种子填充法[J]. 广西师范大学学报(自然科学版), 2003, 21(3):30-33.

[15]王培珍, 许睿. 任意多边形填充新算法[J]. 安徽工业大学学报(自科版), 2009, 26(4):405-408.