第35卷 第20期 计 算 机 工 程 2009年10月
Vol.35 No.20 Computer Engineering October 2009
·图形图像处理· 文章编号：1000—3428(2009)20—0205—03 文献标识码：A 中图分类号：TN911.3

基于边优化的三角网格简化算法

王永皎[1]，郑春峰[2]
(1. 平顶山工学院计算机科学与工程系，平顶山 467000；2. 平顶山交通局信息办，平顶山 467000)

摘 要：在渐进网格算法的基础上，提出一种新的基于边优化的三角网格简化算法。在该方法重建出的多分辨率模型表面上，模型的细节层次呈连续分布，并且能跟随视点位置的变化发生动态变化。实验结果表明，该算法运算速度快，显示效果较好，能有效支持细节层次模型的表示。

关键词：网格简化；边折叠；渐进网格；细节层次模型

Triangular Mesh Simplification Algorithm Based on Edge Optimization

WANG Yong-jiao[1], ZHENG Chun-feng[2]
(1. Department of Computer Science and Engineering, Pingdingshan Institute of Technology, Pingdingshan 467000;
2. Information Office, Pingdingshan Bureau of Transportation, Pingdingshan 467000)

【Abstract】 Based on the simplification of progressive meshes, presents the triangular mesh model simplification based on edge optimization. The Level Of Detail(LOD) is continuously distributed in the mufti-resolution mesh model reconstructed by this approach, and dynamically changed with the movement of the view point. Experimental result demonstrates that the implementation of the algorithm is good effect and runs fast, it also supports the LOD model efficiently.

【Key words】 mesh simplification; edge collapse; progressive mesh; Level Of Detail(LOD) model

1 概述

随着三维图形和建模技术的不断发展，可以得到的网格模型数据量越来越大，因此，有必要对网格进行进行有效的简化，以缓解数据量给模型存储、传输、编辑以及绘制等带来的困难。

网格简化算法的实质就是减少模型的点和面，但又要逼近原始网格。近年来提出了很多网格简化算法。如基于顶点删除、边折叠、三角形删除、三角形折叠、顶点聚类和网格重新划分等简化算法。文献[1]提出边折叠算法，它采用能量函数最优化方法来简化模型，通过引入距离能量、表示能量和弹簧能量，使简化模型是原模型的整体优化逼近。由于优化过程是非线性的。计算量很大，因此该算法的缺点是效率低下。文献[2]用局部二次误差来衡量边折叠的代价，以新顶点到被折叠边的2个顶点相关联平面的距离平方和作为误差量，计算简单并且运行速度较快，简化效果也不错，被广泛应用于模型简化领域。文献[3]提出基于网格重新划分的多边形模型简化方法，该算法产生的网格三角形分布非常均匀，但是牺牲了对原网格的逼近度。

本文研究渐进网格算法，并对其进行改进和优化。

2 渐进网格算法

渐进网格(Progressive Mesh, PM)算法是Hoppe H于1996年在SIGGRAPH会议上提出的，在对其研究中发现，在网格优化过程中所定义的3种几何操作中，只有一种边删除操作对最后的网格简化有贡献，因此，他定义了一种新的表示结构—渐进网格来记录边删除过程。这种渐进网格把任意网格表示成一个简化网格和一组记录了网格细化信息的序列，渐进网格模型的原理是每次从原始网格 M 中删除一条边，逐步将分辨率降低，最后得到一个简化的粗糙网格 M^0 和一系列细节信息记录。根据这一系列的细节信息记录，重新向网格中插入节点和三角形，就可以恢复出具有原始分辨率的模型。同时，PM模型也达到了数据压缩的目的。

3 网格简化算法

3.1 实现步骤

本文提出的简化算法如下：

Step1 输入误差条件。

Step2 对网格中的每个顶点进行分类，区分边界点和内部点。

Step3 取点，是否为最终点。

Step4 非边界点，获取该点周围的拓扑结构。

Step5 对网格进行边折叠操作，折叠误差度量最小的边，含有边界点的边不进行边折叠操作。如果边的两端点的折叠误差大于给定的值，返回Step 3，否则运行Step 7。

Step6 点在同一群内，删除该点，并重新三角化该点周围的点集。返回Step 3。

Step7 算法终止。

本文的简化算法是基于边的误差度量进行的，将边按误

作者简介：王永皎(1977－)，女，讲师，主研方向：计算机图像处理，图形学；郑春峰，工程师

收稿日期：2009-02-04 **E-mail：**wangyongjiao@126.com

差度量从小到大的顺序进行排列，每次对误差度量最小的边进行边折叠操作，同时对含有边界点的边不进行边折叠操作。在边折叠时，新生成的顶点是取原来边的 2 个顶点的其中之一，不用另外产生新的顶点。如果 2 个网格的法向量的夹角在给定的误差范围内，则认为这 2 个网格在同一个群内或认为在一个平面上。如果一个非边界点周围的三角网格两两都在同一个群内，则删除该点并且重新三角化该点周围的点集。重复执行这个操作，直到不能找到满足这个要求的点为止。由于在简化操作之后还需要做网格优化操作，因此这样可以提高算法的效率。

3.2 数据结构设计

模型简化主要有 2 个任务：(1)简化后的逼真度；(2)简化的效率。三维几何模型的数据量很大，有必要通过优化数据结构来提高数据存取效率，从而进一步提高简化速度。考虑到顺序结构具有最优的存取效率，在程序中使用顺序结构来存储顶点信息和三角面片信息。对于系统中大量的排序过程，选择堆排序。采用数组表示顶点信息列表。顶点列表除了记录每个顶点的坐标位置，还记录了每个顶点的顶点关联的面和顶点的度数。由于每个顶点关联面的个数不定，所以关联面采用动态链表来表示。

3.3 LOD 模型的构建

细节层次(Levels Of Details, LOD)模型是指对同一个场景或场景中的物体使用不同细节的描述方法得到的一组模型，可供绘制时选择使用。LOD 模型在复杂 3D 场景的快速绘制、飞行模拟器、3D 动画、交互式可视化等领域得到广泛的应用。本文算法在简化过程中，设初始网格模型为 M_n 每执行一次折叠操作，就把与该操作相关的信息 d_i 记录下来，这样当执行$(n\text{-}m)$次折叠操作得到简化模型 M_m 时，也得到 1 个信息序列$\{d_n, d_{n-1},\cdots, d_{m+1}\}$，由该序列就不难从 M_m 得到原始网格模型和最简网格模型之间层次的网格模型序列$\{M_n, M_{n-1},\cdots, M_{m+1}\}$，该构造方法与文献[4]提出的基于边折叠的累进网格构造方法相类似。

通过简化算法所产生的高质量的 LOD 模型的方法，不仅能节省存储空间，而且由算法对原始模型做一次性处理就可以生成。另外，随着网络的飞速发展，上述算法作为三维图形数据一种有效的增量传输手段适于 VRML 的应用。综上所述，本文算法可以用于构建网格的 LOD 模型。图 1 为牛模型的细节层次。

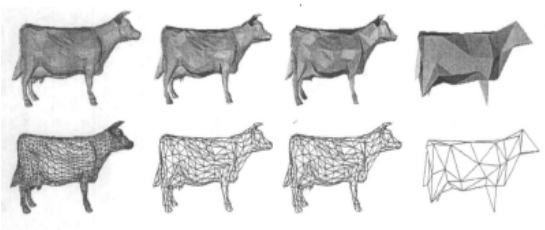


图 1　牛模型细节层次

4　网格优化算法

4.1 网格优化的相关概念

本文的主要目的是产生高质量的简化网格，不仅要保留网格上的特征细节，而且希望网格上的三角形分布均匀，本文在简化的基础上再做网格优化，在定义优化规则之前，需要一个参数来度量在给定网格 M 中的三角形的形状。因为本文的网格优化算法是基于边优化的，所以对在 M 中的所有边 e 定义一个形状因素函数 $h(e)$。

定义 1 形状因素函数 $h(e)$：

$$h(e)=\frac{6Le}{\sum Li}$$

其中，Le 是边 e 的长度；Li 是共享边(与边 e 有一个共同端点的边)的长度。

定义 2 网格 M 符合映射 $\eta 3$，只要满足以下条件：

$$e\in M,\ \frac{1}{\sqrt{2}}\le Le\le\sqrt{2}$$

其中，Le 是 $\eta 3$ 中的边 e 的长度。

定义 3 假设网格 M 有 n 个三角形。在 M 中的 $3n$ 个角度按从小到大的顺序排列。队列中的角度分别为：$\alpha_1, \alpha_2,\cdots, \alpha_{3n}$。于是对于所有的 $i<j$ 都有 $\alpha_i\le\alpha_j$。设 $\boldsymbol{A}(M)=(\alpha_1, \alpha_2, \cdots, \alpha_{3n})$为 M 中的角向量。于是相对应的有设 $\boldsymbol{A}(M')=(\alpha'_1, \alpha'_2, \cdots, \alpha'_{3n})$为 M'中的角向量。如果 $\boldsymbol{A}(M)$按字典编纂顺序都大于 $\boldsymbol{A}(M')$的话，则可认为 M 中的角度向量大于 M'中的角度向量。即表明如果存在一个索引 i, $1\le i\le 3n$ 并有 $\alpha_i=\alpha_j$，则对于所有的 $j<i$ 都有 $\alpha_i>\alpha'_i$，对于这样的情况，则可认为 $\boldsymbol{A}(M)>\boldsymbol{A}(M')$。对于任何一个拥有相同顶点的网格曲面 M 和 M'，如果都有 $\boldsymbol{A}(M)\ge\boldsymbol{A}(M')$，则认为 M 是角度优化的。

定义 4 定义边 e 的最小角度函数 $\alpha(e)$，$\alpha(e)=\min\{\angle\alpha\in f_{i,j}=1, 2\}$，其中，$f_{i,j}=1, 2$ 是共享边 e 的三角形，$\angle\alpha$ 是 $f_{i,j}=1, 2$ 中的一个角。

网格优化的目的是把初始的三角网格在一系列规则下转化成单位网格。它包括分裂最长边，缩并最短边(边的长度是由优化规则算出来的)和交换能达到角度优化的边。基本的网格变换过程如图 2 所示。

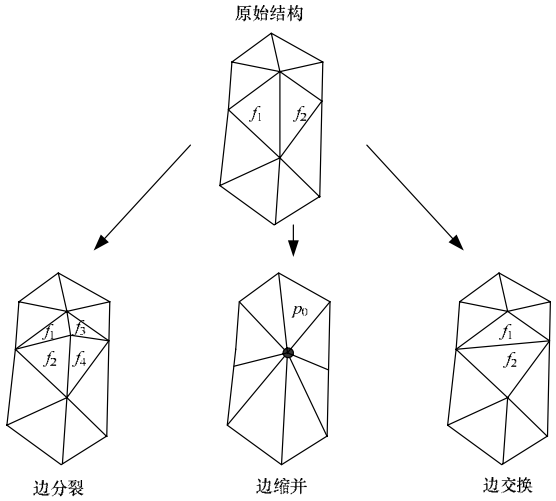


图 2　基本的网格变换

(1)边分裂。对于每一个网格边，边分裂操作包括找到边的中点 Pm 和创建一个由边 e 和对应边 e'的端点所决定的局部球表面。共享边 e 的 2 个三角形 f_1 和 f_2 被 4 个新的三角形 $f_{i,j}=1, 2, 3, 4$ 所代替。总共有一个新的向量和 4 条新的边被创建，并且共享边 e 的所有拓扑结构和它的端点都应被改变。

(2)边缩并。边缩并的操作是基于边的 2 个端点的，因此，就唯一决定了一个点 Pe，该点就是原始边的其中一个端点。它也改变了 Pe 和边 e 跟局部球面 Ss 的交点 P_m'的位置。这个操作删除了 2 个三角形 f_1 和 f_2，它们的共享边 e 和在 f_1 和 f_2 中 2 条边的交点，并将边的端点 Pd 移到 Pe。

(3)边交换。边交换操作的目的是决定目前的结构是否比交换后的结构好。2 个共享 e 的对应边 e'的新三角形(f_1'和 f_2')

代替了 2 个共享边 e 的三角形(f_1 和 f_2)。所有有关边 e 和 e'的终点的拓扑结构都要进行相应改变。

对边分裂操作，新创建的顶点的坐标必须要被确定；对边缩并操作边 e 的端点的新坐标也要被确定下来。因此，用一个局部球面 Ss 来确定这些点。

设边 e 的 2 个端点分别为 $P_1(x_1, y_1, z_1)$和 $P_2(x_2, y_2, z_2)$；对应边 e'的 2 个端点分别为 $P_3(x_3, y_3, z_3)$和 $P_4(x_4, y_4, z_4)$。则局部球面 Ss 即被确定，如图 3 所示。

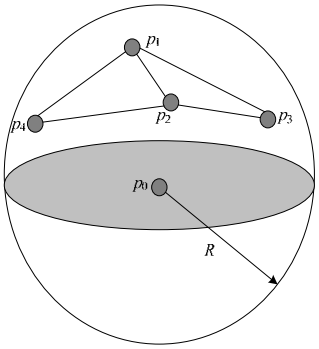


图 3　由 e 和 e'的端点所决定的局部球曲面 Ss

Ss 的方程如下：

$$\begin{vmatrix} x^2+y^2+z^2 & x & y & z & 1 \\ x_1^2+y_1^2+z_1^2 & x_1 & y_1 & z_1 & 1 \\ x_2^2+y_2^2+z_2^2 & x_2 & y_2 & z_2 & 1 \\ x_3^2+y_3^2+z_3^2 & x_3 & y_3 & z_3 & 1 \\ x_4^2+y_4^2+z_4^2 & x_4 & y_4 & z_4 & 1 \end{vmatrix} = 0 \quad (1)$$

边 e 的中点 $P_m=(P_1+P_2)/2$。设 n_1 是 P_1 的法线量；n_2 是 P_2 的法线量。近似认为 Pm 的法线量 $n_m=(n_1+n_2)/2$。于是，一条经过 P_m 以 n_m 作为方向的直线方程为：

$$P(t)=P_m+tn_m \quad (2)$$

由式(1)和式(2)可决定 2 个交点。其中，离 Pm 较近的交点称为与 Ss 的交点 Pm'。在边分裂操作中，将点 Pm'作为 Pm 进行分裂操作，而在边缩并操作中，将 Pe 的位置移到交点 Pm'上。

4.2　算法实现

如果 2 个网格的法向量的夹角在给定的误差范围内，则认为这 2 个网格在同一个群内或认为在一个平面上。如果一个非边界点周围的三角网格两两都在同一个群内，则删除该点并且重新三角化该点周围的点集。重复执行这个操作，直到不能找到这个要求的点为止。算法的具体实现步骤如下：

(1)计算每条边的形状因素函数 $h(e)$和最小角函数 $\alpha(e)$。

(2)将所有边的 $h(e)$放在 1 个堆 Hs 上，并使最大的 $h(e)$在 Hs 的最上面。

(3)重复分裂放在 Hs 的最上面的边 e，删除边 e 并且插入边分裂操作产生的新边。当放在 Hs 的最上面的边的 $h(e)$符合 $h(e)\leq\sqrt{2}$ 时，则停止本操作。

(4)将所有边的 $h(e)$放在 1 个堆 Hc 上，并使最小的 $h(e)$在 Hc 的最上面。

(5)将所有边的 $\alpha(e)/\alpha'(e)$的值放在 1 个堆 Ha，并使最小的 $\alpha(e)/\alpha'(e)$的值在 Ha 的最上面。

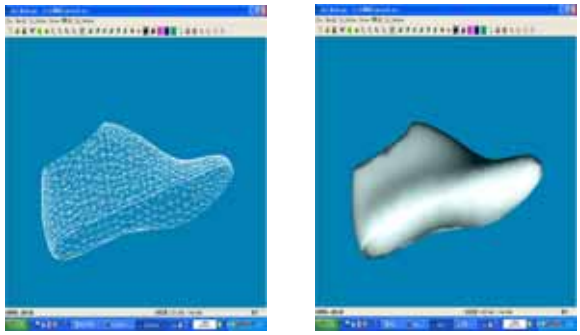
(6)删除放在 Hc 最上面的边，缩并边 e；删除该边在 Ha 上面的记录，并更新其他在 Hc 和 Ha 中的受影响的边的值。

(7)交换放在 Ha 最上面的边，并更新在 Hc 中受影响的边的值；更新在 Hc 和 Ha 中的所有边的值。

(8)如果 Hc 的最上面的边的 $h(e)$值不能满足 $h(e)\geq 1/2$ 或 Ha 的最上面的边的 $\alpha(e)/\alpha'(e)$值不能满足 $\alpha(e)/\alpha'(e)\geq 1$ 的话，回到步骤(6)。

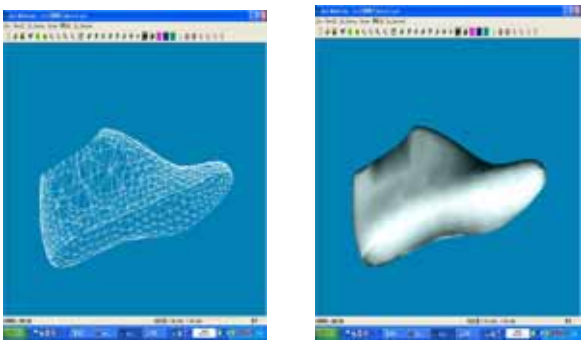
4.3　模型简化效果

原始模型网格图和通过该算法简化后的模型网格图见图 4、图 5。



(a)原始网格图　　(b)原始模型

图 4　原始模型网格图



(a)简化后的网格图　　(b)生成模型

图 5　简化后的模型网格图

5　结束语

以 Visual C++ 6.0 为开发平台，结合 OpenGL 图形库实现基于法向量夹角的二次误差度量的三角网格简化算法。评价一个网格简化算法的好坏的一个重要因素是处理之后的网格对于原始网格的保真度，即最后生成的网格同原始网格的误差。由实验效果可见，该算法不但提高了模型的简化质量，而且在一定程度上加速了模型的简化速度。

参考文献

[1] Hoppe H, Derose T, Duchamp T, et al. Mesh Optimization[J]. Proceedings of SIGGRAPH'93, 1993, 27(8): 19-26.

[2] Garland M, Heckbert P S. Surface Simplification Using Quadric error Metrics[J]. Proceedings of SIGGRAPH'97, 1997, 31(3): 209-216.

[3] Turk G. Retiling Polygonal Surfaces[J]. Proceedings of SIGGRAPH'92, 1992, 26(2): 55-64.

[4] Hoppe H. Progressive Meshes[J]. Proceedings of SIGGRAPH'96, 1996, 30(1): 99-108.

编辑　金胡考