

(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 101877147 A

(43) 申请公布日 2010. 11. 03

(21) 申请号 201010215500.9

(22) 申请日 2010. 06. 29

(71) 申请人 浙江大学

地址 310027 浙江省杭州市浙大路 38 号

(72) 发明人 吴庆标 金勇

(74) 专利代理机构 杭州宇信知识产权代理事务所(普通合伙) 33231

代理人 张宇娟

(51) Int. Cl.

G06T 17/20 (2006. 01)

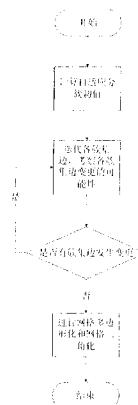
权利要求书 2 页 说明书 7 页 附图 2 页

(54) 发明名称

三维三角形网格模型的简化算法

(57) 摘要

本发明公开了一种三维三角形网格模型的简化算法,它包括如下步骤:1) 给定一个需要简化的三维三角型网格模型和目标简化模型多边形边数 N_c ,将模型表面三角形划分为 N_c 簇集;2) 根据定义的平面逼近误差度量,迭代更新各簇集的边界和各簇集所包含的三角形,以减少各簇集与最终的简化多边形的误差度量;3) 在得到最终的优化簇集划分后,将三个簇集以上的交点视为多边形网格顶点,得到简化后的多边形网格模型。本发明的算法直观有效,简化后的网格模型能够有效保持原始网格的细节,且该算法由于不涉及网格拓扑的修改,使得算法简洁并且有着较好的稳定性,可以有效应用于几何造型系统中。



1. 一种三维三角形网格模型简化算法, 其特征在于包括如下步骤 :

1) 给定一个需要简化的三维三角型网格模型和目标简化模型多边形边数 N_c , 将模型表面三角形划分为 N_c 簇集 ;

2) 根据定义的平面逼近误差度量 ϵ , 迭代更新各簇集的边界和各簇集所包含的三角形, 以减少各簇集与最终的简化多边形的误差度量, 最终得到优化的簇集划分 ;

3) 在得到最终的优化簇集划分后, 将三个簇集以上的交点视为多边形网格顶点, 得到简化后的多边形网格模型。

2. 如权利要求 1 所述的三维三角形网格模型的简化算法, 其特征在于 : 步骤 1) 中, 模型簇集的初始划分是基于模型曲率估计建立的, 其曲率估计公式为 :

$\kappa_G(v) = (2\pi - \sum_{\theta_j \in \text{Neighbour}(v)} \theta_j) / A_v$, 其中 θ_j 为顶点 v 周围的顶角, A_v 为顶点 v 的 Voronoi 区域的面积。

3. 如权利要求 1 所述的三维三角形网格模型的简化算法, 其特征在于 : 所述 N_c 个簇集是按照如下步骤建立 :

1) 选取一个三角形作为种子建立一个分片簇集, 归并与簇集相邻的三角片, 直至该簇集的高斯曲率总值达到目标设定值时, 停止并入三角片, 该簇集建立完成 ;

2) 循环上述建立簇集的方法直到建立 N_c 个簇集。

4. 如权利要求 3 所述的三维三角形网格模型的简化算法, 其特征在于 : 定义三角形法向与簇集法向的误差 $|n(T) - Unif\left(\sum_{T_j \in C_i} \rho_{T_j} n_{T_j}\right)|$, 当簇集停止并入三角片后, 继续将 $|n(T) - Unif\left(\sum_{T_j \in C_i} \rho_{T_j} n_{T_j}\right)| < dTh$ 的三角片并入当前簇集中, dTh 为预先定义的值。

5. 如权利要求 1 所述的三维三角形网格模型的简化算法, 其特征在于采用如下步骤来迭代更新各簇集的边界和各簇集所包含的三角形 :

1) 定义每一个簇集的特征法向 n_{C_i} , n_{C_i} 离散表示为 :

$$n_{C_i} = Unif\left(\int_{v \in C_i} n(v) dv\right) = Unif\left(\sum_{T_j \in C_i} \rho_j n_j\right) = \frac{\sum_{T_j \in C_i} \rho_j n_j}{\left\| \sum_{T_j \in C_i} \rho_j n_j \right\|}, \text{其中 } T_j \text{ 为属于簇集 } C_i \text{ 的所有三角片, } n_j \text{ 为该三角片的单位法向量, } \rho_j \text{ 为该三角片的面积 ;}$$

2) 定义平面逼近误差度量, 误差度量表示为 $\epsilon(F, C) = \int_{v \in F} \|n(v) - n_{C_i(v)}\|^2 dv$, 以表示各个簇集中的各个三角片的法向与其所在簇集的特征平面法向的误差 ;

3) 离散误差度量 $\epsilon(F, C)$, $\epsilon(F, C)$ 离散表示为 :

$$\begin{aligned} \epsilon(F, C) &= \int_{v \in F} \|n(v) - n_{C_i(v)}\|^2 dv \\ &= \epsilon_d(F, C) = \sum_{i=0}^{N_c-1} \left(\sum_{T_j \in C_i} \rho_j \left\| n_j - \frac{\sum_{T_k \in C_i} \rho_k n_k}{\left\| \sum_{T_k \in C_i} \rho_k n_k \right\|} \right\|^2 \right), \text{简化离散形式得到 :} \end{aligned}$$

$$\varepsilon_d(F, C) = \sum_{i=0}^{N_C-1} \left(2 \sum_{T_j \in C_i} \rho_j - 2 \left\| \sum_{T_j \in C_i} \rho_j n_j \right\| \right).$$

6. 如权利要求 1 或 5 所述的三维三角形网格模型的简化算法, 其特征在于 : 对基于迭代更新各个簇集边界和各簇集所包含的三角形采用贪心算法来优化, 所述贪心算法优化步骤如下 : 每一步迭代中, 如果簇集边界 e 相邻的两三角片之一还未归于任何簇集, 那么将其归于另一三角片所属于的簇集 ; 如果两三角片归于不同的簇集, 那么执行簇集边界变更判断, 更新簇集边界 E_c , 直到所有簇集边界的变更判断结果为 C^{init} 停止, C^{init} 表示当前情况并没有发生簇集边界变更。

7. 如权利要求 6 所述的三维三角形网格模型的简化算法, 其特征在于 : 所述变更判断步骤如下 :

- 1) 考虑在簇集边界处的如下三种变更情况, 得到三个不同的误差度量 :
 - a) 原始情况 : 簇集边界两边的三角形仍然属于各自的簇集 ;
 - b) 簇集边界一边的三角形并入另一边簇集 ;
 - c) 簇集边界一边的三角形并入另一边簇集 ;
- 2) 取得到最小误差度量的情况作为最后的变更结果, 然后更新簇集边集合 E_c 。

三维三角形网格模型的简化算法

技术领域

[0001] 本发明涉及一种三维三角形网格模型的简化算法。

背景技术

[0002] 三维三角形网格在计算机辅助几何设计、计算机动画、虚拟现实、计算机游戏和医学影像等领域有着大量的应用。随着三维扫描技术的发展，顶点数为数万的模型已经非常常见，有的模型顶点数达到数十万甚至更多，所以在网格的多分辨率显示，加速网格着色，网格压缩等过程中需要快速的并且能够保持网格模型细节的网格简化算法。已有的几种主要多边形网格简化技术如下：

[0003] 1、删减 (Decimation) :Garland 等^[1] 提出一种基于边缩减的网格简化算法，该方法采用网格的顶点到其相关平面的二次距离作为度量，进行边缩减的迭代直到达到目标网格边数为止。类似地，Hoppe 等^[2]、Klein 等^[3]、Garland 等^[4] 对网格元素提出一种误差度量，其误差度量基于顶点坐标、颜色或是纹理坐标来建立，然后对网格元素进行迭代缩减以简化网格。删减的方法虽然可以得到有效的网格简化效果，但是该类方法得到网格往往存在边度数非常高的顶点，并且由于操作网格的拓扑结构而比较耗时。

[0004] 2、网格精炼 (Mesh Refinement) :Eck 等^[5]、Delingette 等^[6]、Lee 等^[7] 都提出以一个粗网格来逼近表示原始网格模型，然后以各自的策略迭代地在局部进行精炼加密粗网格以达到能够准确表示原始网格模型的精度，但这类方法适用的模型类型非常有限。

[0005] 3、网格重构 (Remeshing) :Alliez 等^[8,9]、Gu 等^[10] 提出能够控制目标网格顶点数的网格重构方法，但是这些方法都受限制于网格的参数化，包括大量的计算和数值不稳定性。Valette 等^[11,12]，通过局部的的贪心算法构造近似的 Centroidal Voronoi Diagrams，然后构造 CVD 的对偶图作为目标简化网格，但是该方法受制于模型简化的程度，当模型简化比例非常大的时候效果不佳。

[0006] 4、全局优化 (Global optimization) :Hoppe 等^[13] 提出将网格简化问题视作全局优化问题。以一个能量函数来度量原始网格，同时通过控制网格的顶点数，顶点坐标，和拓扑连接关系以优化定义的能量函数，可以得到保持原始网格模型细节和曲率简化效果。Cohen-Steiner 等^[14] 提出变分网格逼近方法，该方法将原始网格分为目标数量的近似平面簇集，使用与法向有关的能量来度量平面簇集，以 Lloyd 算法来得到优化的网格分簇，最后每一片分簇以一个多边形表示以得到最终的简化网格。该方法直观有效，简化后的网格模型能够有效保持原始网格的细节。然而该方法速度局限于 Lloyd 算法的迭代次数，Lloyd 算法也无法保证得到一个全局最优的结果，并且需要后期处理（合并法向几乎一致的平面簇集）以达到理想的效果。

参考文献

[0008] [1]Garland M, Heckbert PS. Surface Simplification Using Quadric Error Metrics. In:Whitted T, ed. Proceedings of ACM SIGGRAPH. Los Angeles :ACM Press, 1997. 209–216.

- [0009] [2] Hoppe H. Progressive Meshes. In : Rushmeier H, ed. Proceedings of ACM SIGGRAPH. New Orleans : Addison-Wesley Professional, 1996. 99–108.
- [0010] [3] Klein R, Liebich G, Straß er W. Mesh Reduction with Error Control. In : Yagel R, Nielson GM, eds. Proceedings of IEEE Visualization. San Francisco : IEEE Computer Society Press, 1996. 311–318.
- [0011] [4] Garland M, Heckbert PS. Simplifying Surfaces with Color and Texture using Quadric Error Metrics. In : Ebert DS, Rushmeier H, Hagen H, eds. Proceedings of IEEE Visualization. Washington : IEEE Computer Society Press, 1998. 263–269.
- [0012] [5] Eck M, DeRose T, Duchamp T, Hoppe H, Lounsbery M, Stuetzle W. Multiresolution analysis of arbitrary meshes. In : Mair SG, Cook R, eds. Proceedings of ACM SIGGRAPH. Los Angeles : ACM Press, 1995. 173–182.
- [0013] [6] Delingette H, Herbert M, Ikeuchi K. Shape representation and image segmentation using deformable surfaces. Image and Vision Computing, 1992, 10(3) : 132–144.
- [0014] [7] Lee AWF, Sweldens W, Schröder P, Cowsar L, Dobkin D. Maps : Multiresolution adaptive parameterization of surfaces. In : Machover C, ed. Proceedings of ACM SIGGRAPH. Orlando : ACM Press, 1998. 95–104.
- [0015] [8] Alliez P, Meyer M, Desbrun M. Interactive Geometry Remeshing. In : Appolloni T, ed. Proceedings of ACM SIGGRAPH. San Antonio : ACM Press, 2002. 355–361.
- [0016] [9] Alliez P, Cohen-Steiner D, Devillers O, Levy B, Desbrun M. Anisotropic Polygonal Remeshing. In : Rockwood AP, ed. Proceedings of ACM SIGGRAPH. San Diego : ACM Press, 2003. 485–493.
- [0017] [10] Gu X., Gortler S, Hoppe H. Geometry Images. In : Appolloni T, ed. Proceedings of ACM SIGGRAPH. San Antonio : ACM Press, 2002. 363–374.
- [0018] [11] Valette S, Chassery J-M. Approximated Centroidal Voronoi Diagrams for Uniform Polygonal Mesh Coarsening. Computer Graphics Forum (Proc. Eurographics), 2004, 23(3) : 381–389.
- [0019] [12] Valette S, Kompatsiaris I, Chassery J-M. Adaptive Polygonal Mesh Simplification With Discrete Centroidal Voronoi Diagrams. In : Lazzari G, Pianesi F, Crowley JL, Mase Kenji, Oviatt SL, eds. Proceedings of ICMI. Trento : ACM Press, 2005. 655–662.
- [0020] [13] Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W. Mesh Optimization. In : James TK, ed. Proceedings of ACM SIGGRAPH. Anaheim : ACM Press, 1993. 19–26.
- [0021] [14] Cohen-Steiner D, Alliez P, Desbrun M. Variational Shape Approximation. In : Marks J, ed. Proceedings of ACM SIGGRAPH. Los Angeles : ACM Press, 2004. 905–914.

发明内容

[0022] 本发明目的在于提出一种快速、有效并且鲁棒的三维三角形网格模型的简化算法，它能够比现有技术更有效、更高效地解决三维三角形模型的简化和多分辨率显示的问题。

[0023] 本发明的三维三角形网格模型的简化算法是先对网格模型进行类平面分簇，然后根据分簇结果建立简化后的多边形网格。其具体步骤是：

[0024] 1) 给定一个需要简化的三维三角型网格模型和目标简化模型多边形边数 N_c ，将模型表面三角形划分为 N_c 簇集；

[0025] 2) 根据定义的平面逼近误差度量 ϵ ，迭代更新各簇集的边界和各簇集所包含的三角形，以减少各簇集与最终的简化多边形的误差度量，最终得到优化的簇集划分；

[0026] 3) 在得到最终的优化簇集划分后，将三个簇集以上的交点视为多边形网格顶点，便可以得到一个多边形网格模型，注意此处多边形不一定是平面多边形。可以进一步使用经典的多边形三角化方法将简化的多边形网格转换为简化的三角形网格。

[0027] 在上述步骤 1) 中，所述模型簇集的初始划分是基于模型离散曲率估计建立的，其曲率估计公式为：

[0028] $\kappa_g(v) = (2\pi - \sum_{\theta_j \in \text{Neighbour}(v)} \theta_j) / A_v$ ，其中 θ_j 为顶点 v 周围的顶角， A_v 为顶点 v 的 Voronoi 区域的面积。以基于模型离散曲率作为估计建立 N_c 个簇集。

[0029] 在步骤 (2) 中，平面逼近误差度量定义为 $\epsilon(F, C) = \int_{v \in F} \|n(v) - n_{C_i(v)}\|^2 dv$ ，其中 n_{C_i} 表示簇集 C_i 的特征平面法向： $n_{C_i} = \text{Unif}\left(\int_{v \in C_i} n(v) dv\right) = \text{Unif}\left(\sum_{T_j \in C_i} \rho_j n_j\right) = \frac{\sum_{T_j \in C_i} \rho_j n_j}{\|\sum_{T_j \in C_i} \rho_j n_j\|}$ ，其中 T_j 为属于簇集 C_i 的所有三角片， n_j 为该三角片的单位法向量， ρ_j 为该三角片的面积。

[0030] 实际计算中，将误差度量 $\epsilon(F, C)$ 离散为：

$$[0031] \epsilon(F, C) = \int_{v \in F} \|n(v) - n_{C_i(v)}\|^2 dv$$

$$[0032] = \epsilon_d(F, C) = \sum_{i=0}^{N_c-1} \left(\sum_{T_j \in C_i} \rho_j \left\| n_j - \frac{\sum_{T_j \in C_i} \rho_j n_j}{\|\sum_{T_j \in C_i} \rho_j n_j\|} \right\|^2 \right), \text{简化离散形式得到：}$$

$$[0033] \epsilon_d(F, C) = \sum_{i=0}^{N_c-1} \left(2 \sum_{T_j \in C_i} \rho_j - 2 \left\| \sum_{T_j \in C_i} \rho_j n_j \right\| \right).$$

[0034] 在步骤 (2) 中，我们设计了一种基于迭代更新各个簇集边界以优化误差度量的算法，在每一步的迭代过程中，我们需要考虑簇控制集边界变化以减小误差度量 $\epsilon_d(F, C)$ 的可能性。为此，我们采用贪心算法来优化误差度量：迭代地更新各簇集边界，直至没有簇集边发生变更停止。

[0035] 以上方法相比传统的方法，其优势在于本方法直观有效，简化后的网格模型能够有效保持原始网格的细节，且该算法由于不涉及网格拓扑的修改，使得算法简洁并且有着较好的稳定性。通过大量试验表明，本方法易于实现，并且具有直观的几何意义，可以有效应用于几何造型系统中。

附图说明

- [0036] 图 1 为本发明简化算法实施例的完整流程图；
- [0037] 图 2 为本发明简化算法实施例的模型在简化中三个状态的示意图；
- [0038] 图 3 为本发明简化算法实施例的局部簇集边界变更分析示意图。

具体实施方式

- [0039] 下面，结合附图和实施例来对本发明的具体实施方式做详细说明。
- [0040] 1. 问题简述
 - [0041] 以下为本发明实施例方法的具体描述，本方法基于三角形网格表述算法，三角形网格 M 可以表示为 $\{V, E, F\}$ ，其中 V 为网格顶点的集合， E 为网格边的集合， F 为网格面的集合：
 - [0042] $V = \{v_i = (x_i, y_i, z_i) \in \mathbb{R}^3 | 1 \leq i \leq N_v\}$
 - [0043] $E = \{e_i = \{v_{i_1}, v_{i_2}\}, i = 1, \dots, N_E\}$ (1)
 - [0044] $F = \{f_i = \{v_{i_1}, v_{i_2}, v_{i_3}\}, i = 1, \dots, N_F\}$
 - [0045] 本方法目标为将原始网格 M 的网格面片 F 划分为边连通的近似平面簇集 $C = \{C_i \subset F, i = 1, \dots, N_C\}$ ， $N_c < < N_F$ 的问题，其中 $\bigcup_{i=1, \dots, N_C} C_i = F$ ， $C_i \cap C_j = \emptyset, \forall i, j, N_c$ 用以控制网格的简化比例，每一簇 C_i 中的原始网格面 $f \in C_i$ 相互关于边连通。当分簇完成后，每一个簇集可以用一个四边形表示，进而可以得到一个逼近原始网格的四边形简化网格。
 - [0046] 图 1 为本实施例简化算法的流程图，得到目标模型后，对模型进行自适应的分片初值划分；然后迭代各个簇集边界，观察各个簇集边界发生变更的可能性，若有簇集边界变更发生，则继续这一步；若没有任何簇集边界发生变更，则得到了最终的分簇结果；对分簇结果进行多边形化或三角化得到最终的简化网格模型。
 - [0047] 图 2 以 bunny 模型表明了模型在简化过程中的三个不同的状态：如图 2a 所示，该原始 bunny 模型有 69451 个三角片；对原始模型进行分簇，得到 100 个簇集，如图 2b 所示；由最终分簇结果得到简化后的多边形网格模型，如图 2c 所示。
 - [0048] 2. 分簇算法
 - [0049] 2.1 初始化
 - [0050] 在本实施例中，为设计一个完整的分簇算法，需要一个初始分簇划分，然后实施贪心算法优化误差度量直到收敛为止。如果使用随机的簇集初值划分，不利于算法收敛的速度以及最终的效果，为此提出自适应的种子选取和簇集增长方式以获得初始分簇。
 - [0051] 使用二维流形三角网格的高斯曲率估计，对于网格顶点，其高斯曲率为：
 - [0052]

$$\kappa_G(v) = (2\pi - \sum_{\theta_j \in \text{Neighbour}(v)} \theta_j) / A_v, \quad (2)$$
 - [0053] 其中 θ_j 为顶点 v 周围的顶角， A_v 为顶点 v 的 Voronoi 区域的面积。三角片的高斯曲率 $K_G(T_i)$ 近似取为其三顶点的高斯曲率绝对值和的平均值，用以反应该三角片曲率的高低。
 - [0054] 考虑网格分簇问题：网格 $M = \{V, E, F\}$ ，目标分簇数 N_c ，为了拥有高曲率的三角片的簇集将归并较少的三角片，拥有低曲率的三角片的簇集将归并较多的三角片。我们预测

理想的分簇情况下,每一个簇集的高斯曲率总值应该接近于网格的平均值:

[0055]

$$D = \frac{1}{N_C} \sum_{T_i \in F} \rho_i \kappa_G(T_i), \quad (3)$$

[0056] 其中 T_i 为所有三角片, ρ_i 为 T_i 的面积。

[0057] 建立每一个簇集 C_i 时:随机取一个还未归属于任何一个簇集的三角片 T_{seed} 作为当前簇集 C_i 的种子三角片,循环地更新该簇集边界将还未归属于任何簇集并且与该簇集边相连的三角片以

$|n(T) - Unif\left(\sum_{T_j \in C_i} \rho_{T_j} n_{T_j}\right)|$ 为优先级并入到此簇集中,直到该簇集的高斯曲率总值达到目标设定值 D ,即 $\sum_{T_j \in C_i} \rho_j \kappa_G(T_j) > D$ 时,停止并入三角片。为了使几乎属于一个平面的三角片只归入一个簇集当中,当簇集停止并入三角形后,仍然可以将

$|n(T) - Unif\left(\sum_{T_j \in C_i} \rho_{T_j} n_{T_j}\right)| < dTh$ 的三角片并入当前簇集中,其中, dTh 为可以容忍的三角片法向与簇集法向误差。循环上述建立簇集的方法直到建立 N_c 个簇集。至此,得到一个初始簇集划分,最终实际簇集数 N_{rc} 可能小于 N_c ,那么修改用户指定的分簇数 N_c 为 N_{rc} 。

[0058] 2.2 平面逼近簇集的误差度量

[0059] 定义每一片簇集 C_i 的特征法向量 n_{C_i} 为该簇集最近似的平面的法向,可如下计算:

[0060]

$$n_{C_i} = Unif\left(\int_{v \in C_i} n(v) dv\right), \quad (4)$$

[0061] 其中 $n(v)$ 为 v 点的单位法向量, $Unif()$ 指对向量的单位化操作运算。

[0062] 希望给出一种分簇策略,使得每一个簇集中的所有点与其所对应的簇集特征向量最接近,这样各个簇集都各自最接近于平面。

[0063] 给定网格 $M = \{V, E, F\}$, 目标分簇数 N_c , 在对网格的分簇过程中,希望最小化如下度量:

$$\epsilon(F, C) = \int_{v \in F} \|n(v) - n_{C_i(v)}\|^2 dv, \quad (5)$$

[0065] 其中 $n(v)$ 为 v 点的单位法向量, $C_i(v)$ 为 v 所属于的簇集, $n_{C_i(v)}$ 为(2)式定义的特征法向量。

[0066] 观察(5)式,特征法向量 n_{C_i} 代表与该簇集最近似的平面的法向,将每一点的单位法向量与其所属于的簇集的特征法向量的方差的积分作为误差量,误差量越小,说明每一个簇集中的所有点与其所对应的簇集特征向量越接近,亦即每一个簇集越将法向相近的网格三角片归在一起。从图形的绘制上来分析,同一个簇集中的网格片拥有相近法向则拥有着相近的光照效果,有利于用一个多边形来表示一个簇集。

[0067] 给定网格 $M = \{V, E, F\}$, 目标简化多边形模型数 N_c , 误差度量 $\epsilon(F, C)$, 希望找到一个最优的划分 C_{opt} , 使得 $\epsilon(F, C_{opt}) = \min\{\epsilon(F, C)\}$, 其中 C 为该目标分簇数所对应的所有的可能的划分。

[0068] 各簇集的特征法向量 n_{C_i} 可以离散表示为:

[0069]

$$n_{C_i} = Unif \left(\int_{v \in C_i} n(v) dv \right) = Unif \left(\sum_{T_j \in C_i} \rho_j n_j \right) = \frac{\sum_{T_j \in C_i} \rho_j n_j}{\left\| \sum_{T_j \in C_i} \rho_j n_j \right\|}, \quad (6)$$

[0070] 其中 T_j 为属于簇集 C_i 的所有三角片, n_j 为该三角片的单位法向量, ρ_j 为该三角片的面积。

[0071] 误差度量 $\epsilon(F, C)$ 可以离散为:

$$[0072] \epsilon(F, C) = \int_{v \in F} \|n(v) - n_{C_i(v)}\|^2 dv$$

$$[0073] = \epsilon_d(F, C) = \sum_{i=0}^{N_C-1} \left(\sum_{T_j \in C_i} \rho_j \left\| n_j - \frac{\sum_{T_j \in C_i} \rho_j n_j}{\left\| \sum_{T_j \in C_i} \rho_j n_j \right\|} \right\|^2 \right) \quad (7)$$

[0074] $\epsilon_d(F, C)$ 为 $\epsilon(F, C)$ 的离散形式, 设计一种基于迭代更新各个簇集边界以优化误差度量的算法, 在每一步的迭代过程中, 我们需要考虑簇控制集边界变化以减小误差度量 $\epsilon_d(F, C)$ 的可能性, 为此, 需要知道误差度量 $\epsilon_d(F, C)$ 在簇集边界变化前后的值, 如果以(5)式计算 $\epsilon_d(F, C)$, 其复杂度为 $O(M+N)$, 显然时间开销巨大。其中, M 和 N 为当前簇集边界相邻的两个簇集的三角片个数, 这两个簇集对于误差度量的贡献由于特征法向量的变化而会发生变化, 需要重新计算。为此,

$$[0075] \text{分析 (5) 式, 得到: } \epsilon_d(F, C) = \sum_{i=0}^{N_C-1} \left(\sum_{T_j \in C_i} \rho_j \frac{\left\| \sum_{T_j \in C_i} \rho_j n_j \right\| \cdot n_j - \sum_{T_j \in C_i} \rho_j n_j}{\left\| \sum_{T_j \in C_i} \rho_j n_j \right\|^2} \right)$$

$$[0076] = \sum_{i=0}^{N_C-1} \left(\frac{\sum_{T_j \in C_i} \rho_j \left\| \sum_{T_j \in C_i} \rho_j n_j \right\|^2 \|n_j\|^2 - 2 \left\| \sum_{T_j \in C_i} \rho_j n_j \right\| \sum_{T_j \in C_i} \rho_j n_j \cdot \sum_{T_j \in C_i} \rho_j n_j + \sum_{T_j \in C_i} \rho_j \left\| \sum_{T_j \in C_i} \rho_j n_j \right\|^2}{\left\| \sum_{T_j \in C_i} \rho_j n_j \right\|^2} \right)$$

[0077]

$$= \sum_{i=0}^{N_C-1} \left(\sum_{T_j \in C_i} \rho_j \|n_j\|^2 - 2 \left\| \sum_{T_j \in C_i} \rho_j n_j \right\| + \sum_{T_j \in C_i} \rho_j \right) \quad (8)$$

[0078] 注意到 $\forall T_j, \|n_j\|=1$, 则

[0079]

$$\epsilon_d(F, C) = \sum_{i=0}^{N_C-1} \left(2 \sum_{T_j \in C_i} \rho_j - 2 \left\| \sum_{T_j \in C_i} \rho_j n_j \right\| \right) \quad (9)$$

[0080] 对于每一个簇集 G_i , 只需记录一个向量 $N_i = \sum_{T_j \in C_i} \rho_j n_j$ 和一个标量 $S_i = \sum_{T_j \in C_i} \rho_j$, 便可以在迭代更新簇集边界时, 以 $O(1)$ 复杂度计算出误差能量 $\epsilon_d(F, C)$ 。

[0081] 根据(7)式, 我们提出一种迭代更新各个簇集 C_i 边界的贪心算法。假设网格边 $e \in E$, e 的两侧三角形为 T_m, T_n , 如果 T_m, T_n 属于不同的簇集, 那么称 e 是一条簇集边界, 所有簇集边界的集合为 E_c , 显然 $E_c \subset E$ 。

[0082] 每一步迭代中, 考虑每一条簇集边 $e \in E_c$ 变更以减小 $\epsilon(F, C)$ 的可能性: 假设 e

的两侧三角形为 T_m, T_n , 所属于的簇集分别为 C_k, C_l 。如图 3 可以考虑三种变更情况, 可以得到三个不同的误差度量:

[0083] a) 原始情况 $\epsilon_d(F, C^{init})$: 簇集边界两边的三角形 T_m, T_n 仍然属于各自的簇集 C_k, C_l , 即 $T_m \in C_k, T_n \in C_l$ 。

[0084] b) $\epsilon_d(F, C^l)$: 簇集边界一边的三角形 T_m 并入另一边簇集 C_k , 即 $T_m \in C_k, T_n \in C_k$;

(10)

[0085] c) $\epsilon_d(F, C^2)$: 簇集边界一边的三角形 T_n 并入另一边簇集 C_k , 即 $T_m \in C_l, T_n \in C_l$;

[0086] 取得到最小误差度量的情况作为最后的变更结果, 然后更新簇集边集合 E_c 。对于簇集边进行迭代更新, 可以迭代地减小误差度量 ϵ , 直至所有簇集边的变更判断结果为 C^{init} 停止。由于每一步变更都缩小误差度量 ϵ_d , 所以该算法的收敛性能够得到保证。为了保证簇集的连通性, 在实际迭代计算中, 如果簇集边的变更使得破坏簇集的连通性, 则禁止此步变更。

[0087] 在实际的计算中, 对每一个簇集 C_i , 只需记录 $N_i = \sum_{T_j \in C_i} \rho_j n_j$ 即可。

[0088] 事实上, 当考虑 $e \in E_c$ 的变更可能性时, 只有其相关的簇集 C_k, C_l 的所贡献的误差度量部分可能发生变化。由 (7) 式,

[0089]

$$E_{C_k \cup C_l} = 2 \left(\sum_{T_j \in C_k \cup C_l} \rho_j - \left\| \sum_{T_j \in C_k} \rho_j n_j \right\| - \left\| \sum_{T_j \in C_l} \rho_j n_j \right\| \right) \quad (11)$$

[0090] 在 (8) 中的三种簇集边 e 变更情况中, $\sum_{T_j \in C_k \cup C_l} \rho_j$ 为常量, 所以, 只需考虑

[0091]

$$L_{C_k \cup C_l} = \left\| \sum_{T_j \in C_k} \rho_j n_j \right\| + \left\| \sum_{T_j \in C_l} \rho_j n_j \right\| = \frac{1}{2} \left(E_{C_k \cup C_l} - \sum_{T_j \in C_k \cup C_l} \rho_j \right) \quad (12)$$

[0092] 分别计算 C^{init}, C^l, C^2 所对应的 L_{init}, L_1, L_2 的值, 取 L 最大的变更情况 (即对应误差度量最小) 作为最后的变更结果。 L 越大对应 (7) 式中所对应的误差度量 $\epsilon_d(F, C)$ 越小, 观察 (10) 式, L 为各簇集三角片的以面积为权的法向量和的模, 由向量和的性质: 一簇相同模长的向量, 方向越是相同其向量和越大; 所以也可以由此得到优化 (7) 式即 (10) 式的几何意义: 将具有相近法向量的三角片归于同一个簇集, 从而同一簇集的三角片近似属于某一特征平面, 这也正是分簇目的所在。

[0093] 得到初始的分簇划分后, 可以实现本节的局部贪心算法: 在每一步迭代中, 遍历所有的簇集边界 $e \in E_c$, 如果 e 相邻的两三角片之一还未归于任何簇集, 那么将其归于另一三角片所属于的簇集; 如果两三角片归于不同的簇集, 那么执行上一节中提出的变更判断, 更新簇集边界 E_c , 直到所有簇集边的变更判断结果为 C^{init} 停止。

[0094] 2.3 多边形化和三角化

[0095] 在得到最终的优化簇集划分后, 将三个簇集以上的交点视为多边形网格顶点, 便可以得到一个多边形网格模型, 注意此处多边形不一定是平面多边形。我们还可以将多边形网格分片为三角形网格, 可以使用 Cohen-Steiner 等^[15] 提出的多边形边校正方法优化和三角化该多边形网格, 或是以经典的多边形三角化方法将简化的多边形网格转换为简化的三角形网格。

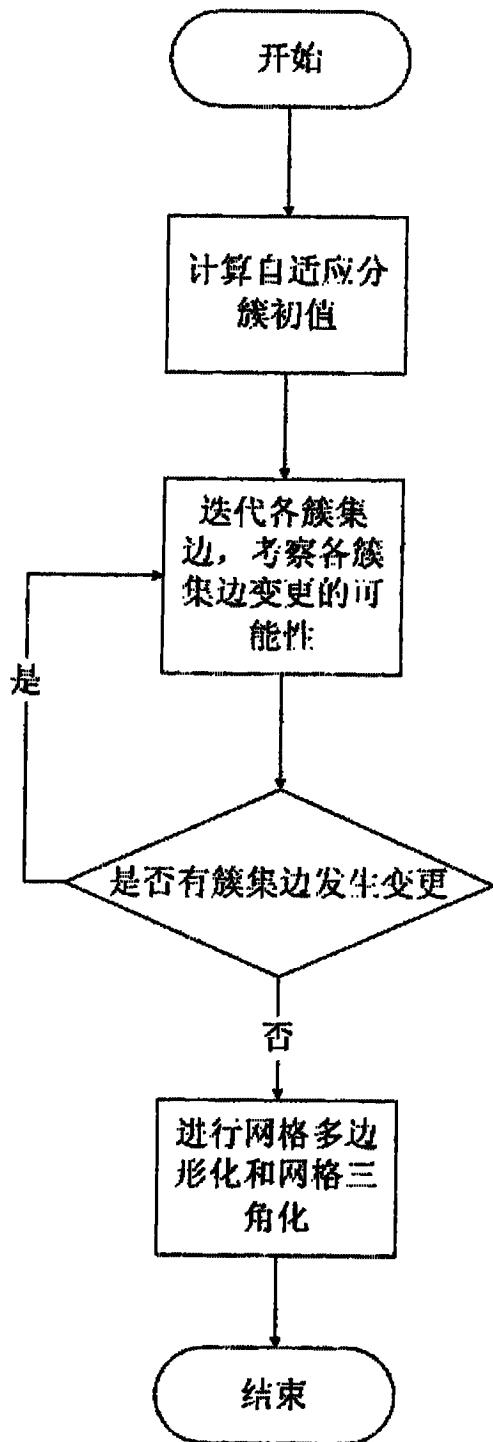


图 1

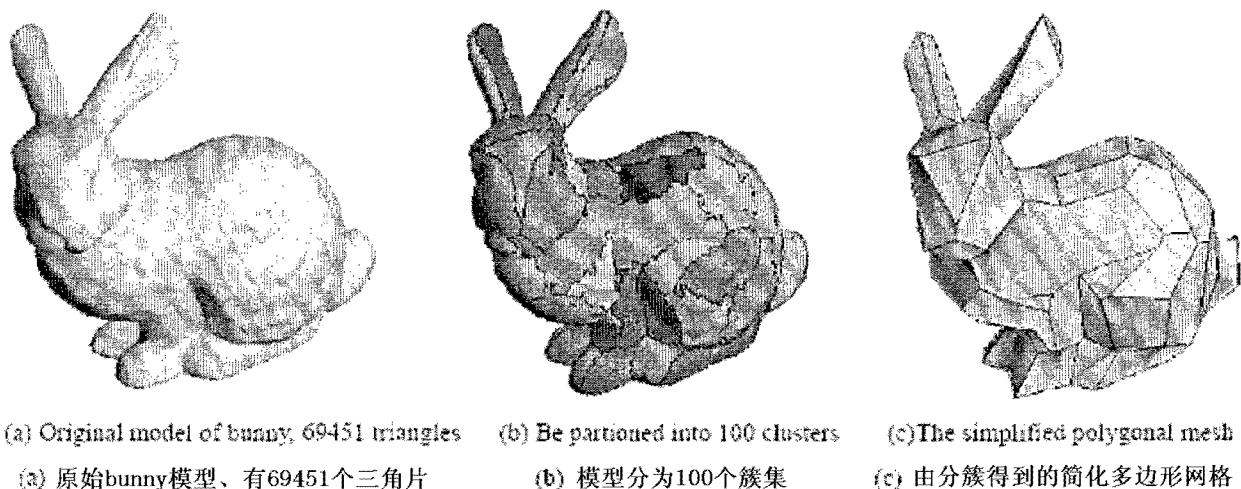


图 2

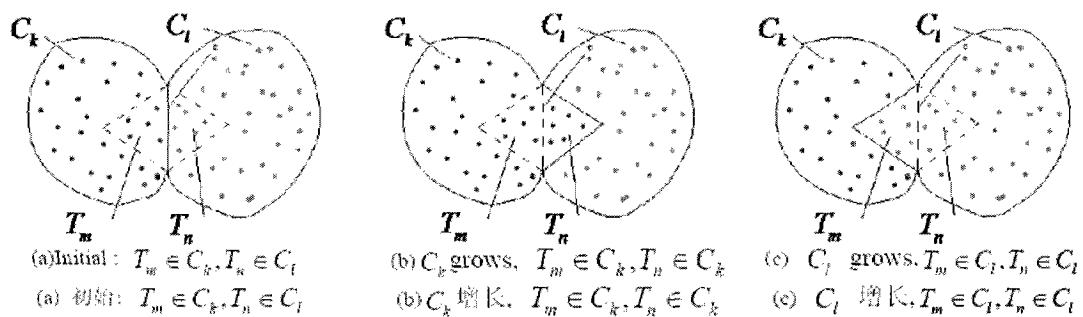


图 3