第27卷 第2期
2001年3月

自 动 化 学 报
ACTA AUTOMATICA SINICA

Vol. 27, No. 2
Mar., 2001

拓扑结构可变的动态多细节层次模型[1)]

陶志良 成迟薏 潘志庚 石教英

(浙江大学 CAD&CG 国家重点实验室 杭州 310027)

(E-mail: {zltao, chengcy, pzg, jyshi}@cad.zju.edu.cn)

摘 要 研究多细节层次技术在虚拟环境中的广泛应用. 提出了一种新的动态多细节层次结构,这个结构不仅可以适用于多种不同的拓扑结构保持的网格简化方法,还适用于拓扑可变的网格简化方法,可应用于任意网格模型. 在此基础上还设计了一种基于拓扑结构可变的网格简化算法:顶点合并. 通过顶点合并和顶点分裂两个对偶操作,实现不同细节层次模型之间的平滑转换,并能够自适应地改变模型的拓扑结构.

关键词 多细节层次,网格简化,实时绘制,虚拟现实.

DYNAMIC LEVEL OF DETAIL FRAMEWORK WITH TOPOLOGICAL STRUCTURE MODIFICATION

TAO Zhi-Liang CHENG Chi-Yi PAN Zhi-Geng ShI Jiao-Ying

(*State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou* 310027)

(E-mail: {zltao, chengcy, pzg, jyshi}@cad.zju.edu.cn)

Abstract The computation and storage requirement for complex polygonal scenes far exceeds the capacity of modern graphics hardware. One approach to speed up rendering is to exploit level of detail (LoD) techniques in virtual environments. This paper presents a new dynamic LoD hierarchy. Arbitrary meshes and operations changing model's topological types are valid in our framework. In addition, we describe a mesh simplification method: vunify, which supports topological type modification to support this new framework.

Key words Multiple level of detail, mesh simplification, real-time rendering, virtual reality.

1 引言

解决复杂场景实时绘制的一种技术是多细节层次模型(Level of Detail,简称 LoD),即先定义一组不同细节层次的网格模型,再根据视点的不同来选择不同的 LoD 模型进行

1) 国家自然科学基金资助项目.

收稿日期 1999-04-09 收修改稿日期 2000-05-24

绘制.

文献[1]中介绍了一种递进网格表示方式,简化网格通过一系列顶点分裂操作就可以动态实时地恢复成原始网格.文献[2～4]分别提出了基于层次结构的动态网格简化方法,它们具有通用、全自动和自适应等特点.这些层次结构能够无缝地结合各种不同的网格简化算法,在复杂几何场景中随着视点的动态转变实时构造不同细节层次模型.

然而这些方法都有其局限性,对递进网格[1]来说,首先,它只能处理有方向的二维网格表面,即与圆盘或半圆同构的表面模型.而且在复杂场景中,也只能针对单个物体模型进行处理,不能合并两个或两个以上的物体;其次,递进网格保证每个层次的网格具有相同的拓扑结构,这样对于复杂模型就存在一个下限,物体的最简网格数目不可能超过这个下限.

同样,文献[2,3]提供的方法都具有这种局限性,不能随视点变化对模型做最大程度的简化.文献[4]提供了一种基于空间分割的网格动态简化框架,能够根据需要改变模型的拓扑结构,但是这种方法基于特定的空间分割网格简化方法,不能与其他网格简化算法很好地结合.文献[5]将三维模型表面组织成有层次结构的多分辨率模型,所采用的结构便于实时进行与视点相关的自适应简化.

本文提出一种拓扑结构可变的动态 LoD 结构,这个框架不仅适用于多种网格简化方法,同时允许简化过程中拓扑结构可变.另外,拓扑结构可变的网格动态简化方法具有较大适用范围、自适应性强和更加灵活等特点.本文还将提出一种新的网格简化方法,以实现可变拓扑结构的动态多细节层次.

2　拓扑结构可变的网格简化实现

2.1　网格简化元操作

根据不同的网格简化元操作,可以把网格简化算法分成顶点移去、边折叠和三角形合并等不同类型[6].图1所示就是基于边折叠简化元操作的网格简化过程.许多基于边折叠的网格简化方法都通过设置边折叠操作进行的约束条件,或者通过检查简化后是否出现自相交来保证网格模型的拓扑结构保持不变,目的是为了确保使用简化网格绘制场景的质量.因此,经过基于边折叠简化元操作迭代产生的简化网格有着与原始网格相同的拓扑结构,不可能随意减少三角形数目[7].而实际上,当模型距离观察者视点很远,或者模型位于视区外和被遮挡的情况下,物体拓扑结构的细小变化对用户最终观察到的绘制质量影响不大[8].

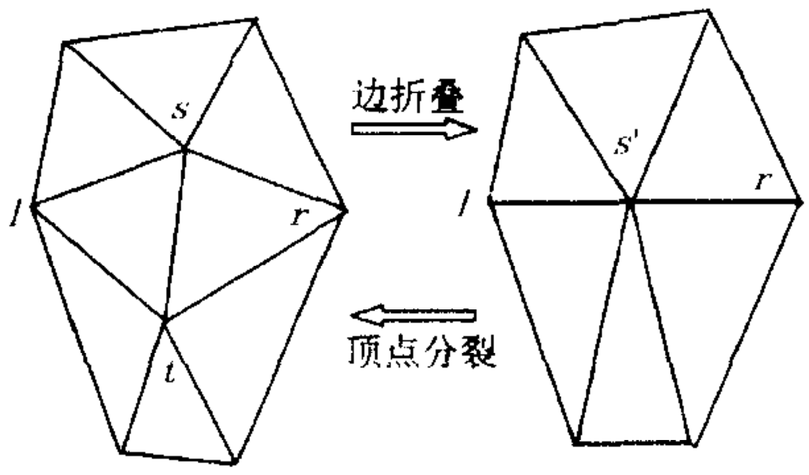


图1　边折叠操作示意

不同的网格简化方法基于不同的网格简化元操作,为了描述可变拓扑结构的网格动态简化方法,引入顶点合并操作 vunify 代替边折叠 ecol 来生成递进网格

$$M_k \underset{\text{vsplit}_{k-1}}{\overset{\text{vunify}_{k-1}}{\rightleftharpoons}} M_{k-1} \underset{\text{vsplit}_{k-2}}{\overset{\text{vunify}_{k-2}}{\rightleftharpoons}} \cdots M_1 \underset{\text{vsplit}_0}{\overset{\text{vunify}_0}{\rightleftharpoons}} M_0, \tag{1}$$

顶点合并操作比边折叠操作具有更大的普遍性,不仅允许多个顶点之间的合并,而且允许

对网格中任意顶点对(可以不组成边)之间进行合并操作.

2.2 网格顶点合并转换

顶点合并操作 vunify($\{s\},\{t\},\cdots$)可以实现任意多个顶点合并成一个新的顶点.理论上基于顶点合并操作的网格简化算法可以处理任意类型的网格模型,网格中可以包括单个顶点、单个线段或高维多面体.为了简化起见,在这里只讨论任意两个顶点之间的合并,支持的网格类型不包括单个顶点和单个线段.实际上多个顶点之间的合并类似两个顶点之间的合并.

图2显示一个顶点合并操作的例子.图中原始网格是一个3D 流形(manifold)网格,包含了一个四面体和一个没有邻接面的三角形.经顶点合并后网格的拓扑结构已经改变.原始网格中的四面体被合并为一个三角形;与图1中每条边最多与两个三角形相邻接不同,图2中边$\{s,t\}$与三个三角形相邻接.顶点合并 vunify($\{s\},\{t\}$)一共删除了1个顶点,4条边和4个三角形.

2.3 算法描述

根据图2描述的顶点合并操作,作者提供了一种可以改变原始网格拓扑结构的网格简化方法.下面列出算法的基本步骤:

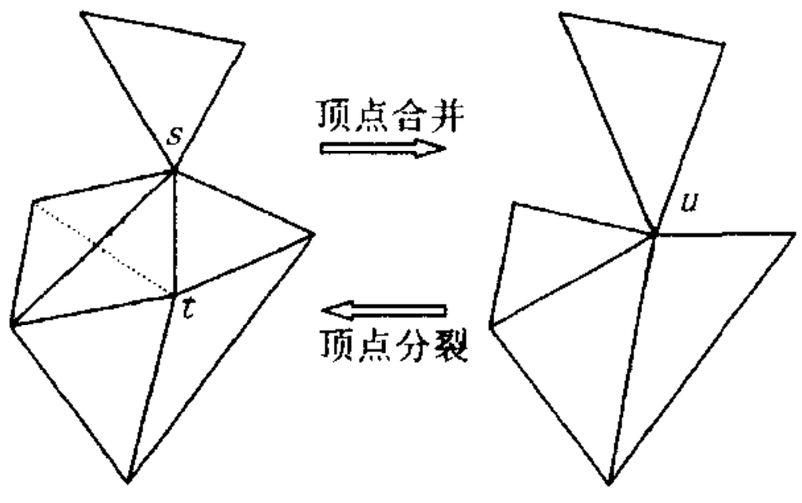


图2 网格顶点合并操作示意

步骤1.读入原始网格 M_n,网格数据结构初始化,建立待合并的后备顶点对队列;

步骤2.选择符合条件的顶点对($\{j\},\{k\}$),如果找不到则转步骤8;

步骤3.合并该顶点对,生成一个新顶点$\{l\}$;

步骤4.删除顶点$\{j\},\{k\}$,如果存在边$\{j,k\}$则删除$\{j,k\}$;

步骤5.检查由于顶点合并引起的三角形重合,删除相应的三角形;

步骤6.修改由于顶点合并引起的拓扑结构变化;

步骤7.修改顶点对队列,重复步骤2;

步骤8.输出网格 M_0.

为了尽量保证输出简化网格的质量,作者在上述算法框架中加入各种误差判断标准,计算每次选择的顶点对合并以后所引起的网格畸变与原始网格之间的误差,通过建立顶点优先级队列使每次顶点合并操作引起的误差较小.本文使用了一种基于特征和顶点距离的混合误差方法.

$$E(\{j\},\{k\}) = w_1 E_{\text{Dist}}(\{j\},\{k\}) + w_2 E_{\text{Feature}}(\{j\},\{k\}), \tag{2}$$

其中 $E_{\text{Dist}}(\{j\},\{k\})$表示顶点对$(\{j\},\{k\})$之间的距离平方,$E_{\text{Feature}}(\{j\},\{k\})$表示其几何特征误差;$w_1,w_2$分别表示上述两种误差的权值.在给出特征误差的计算方式之前,首先定义一些网格表面基本特征.给定任一初始网格 $M_n=(K,V)$,K 为网格拓扑构造,V 为几何构造.定义

$$\text{Sharp}(\{j,k\}) = \begin{cases} \text{TURE} & \text{Dihedral}(\{j,k\}) < \theta, \\ \text{FALSE} & \text{Dihedral}(\{j,k\}) \geqslant \theta, \end{cases} \tag{3}$$

其中$\{j,k\}$为 K 中的边,并且边$\{j,k\}$有且仅有两个邻接面;θ 为一个二面角的阈值,当关于边$\{j,k\}$的二面角小于阈值 θ 时,认为此边是棱边,也即特征边.那么网格 M_n 中任意边

$\{j,k\}$为一条特征边的条件是:a)边$\{j,k\}$的邻接三角形数目不为2;b)边$\{j,k\}$有且仅有两个邻接三角形,并且 Sharp($\{j,k\}$)为真.

找到网格 M_n 中所有特征边后,接着要计算网格 M_n 中所有的特征顶点.对任意顶点$\{j\}$,根据其所有邻接特征边的数目定义其特征值.参见表1.

网格顶点对($\{j\},\{k\}$)的特征误差函数 $E_{\text{Feature}}(\{j\},\{k\})$由两个顶点各自的特征误差与顶点对特征误差组合而成.公式(2)中的 w_1,w_2分别可以根据实际情况设置,主要依据为顶点对之间距离对网格简化效果的影响大小(见表2).这样,算法步骤1可以使用上面给出的误差标准对网格中所有顶点对进行排序,步骤2每次都选择误差产生最小的顶点对合并.

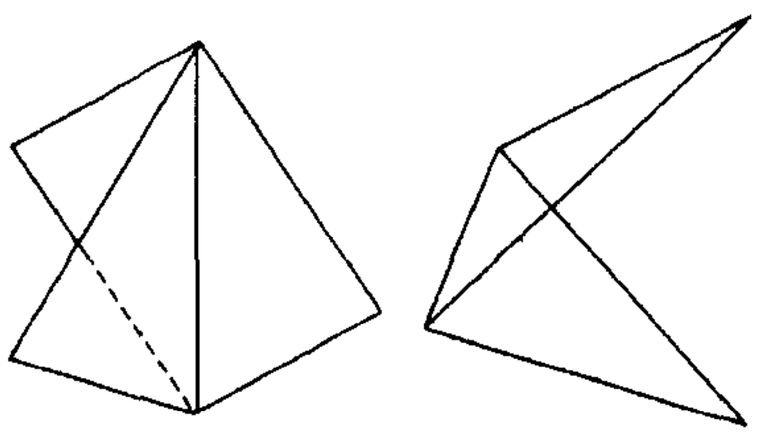
顶点合并完成后,需要修改相应的网格,包括删除一些顶点、边和三角形.与拓扑不变的网格简化算法不同,这里要注意两种特殊的网格三角形分布情况,如图3所示,算法在步骤5中,检查这两种情况并删除相应的三角形.

表1　特征顶点定义

	平滑点	投点(dart)	折点	交点
邻接特征边数目	0	1	2	>2
权重	0	0.5	1	>1

表2　顶点对($\{j\},\{k\}$)误差定义

($\{j\},\{k\}$)的连接关系	平滑边	特征边	没有连接边
权重	0	0.5	10



(a)顶点合并引起3个三角形删除　(b)顶点合并引起三角形重合的情况

图3　顶点合并操作中出现的特殊情况

3　可变拓扑结构的动态网格简化

3.1　顶点树及其生成

本算法采用顶点树作为基本数据结构,顶点树在网格简化过程中产生.网格简化过程由一系列网格简化元操作 vunify 组成.场景网格顶点树以由底向上的方式构造,原始网格中的所有顶点都以顶点树叶节点的形式初始化.网格简化中顶点树可以根据 vunify 所决定的顶点之间关系随迭代过程的进行逐步建立起来.由于假定网格简化只处理两个顶点之间的合并操作,所以最后生成树的类型为二叉顶点树.见图4.

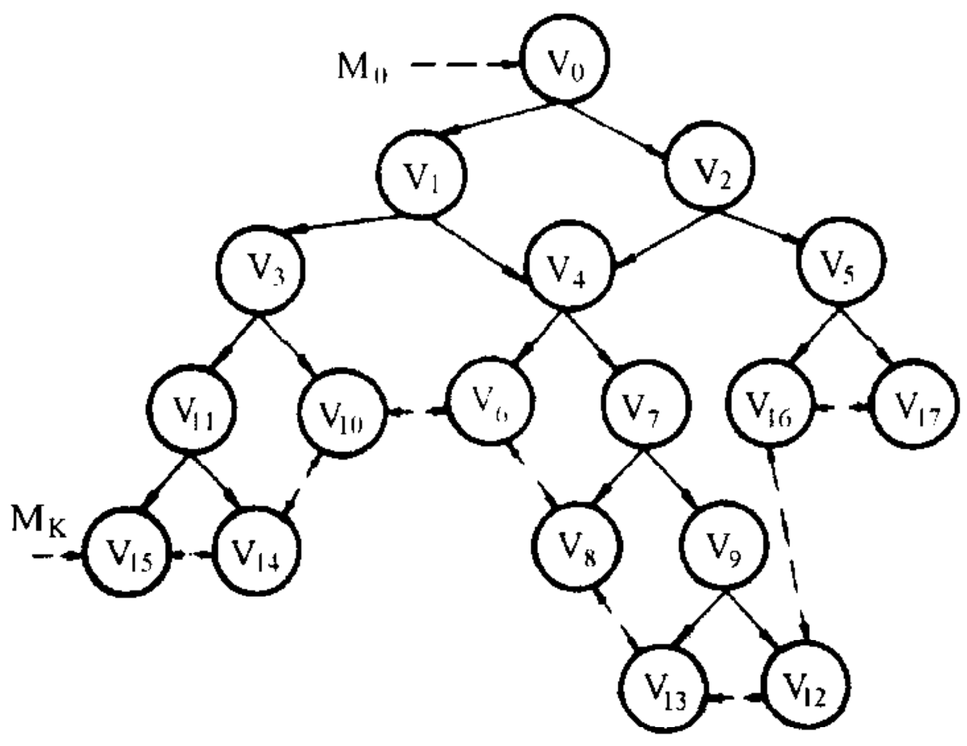


图4　LoD 网格模型的顶点层次结构示意(其中网格的顶点集合由顶点链表示)

从上面介绍可知该方法可以合并场景中的任意两个顶点,引起拓扑结构的改变.也就是说能够合并两个不同的物体模型,或者是更多物体之间的合并.事实上,如果允许线段合并或者孤立点合并操作,最终可以得到且仅得到一棵二叉顶点树.最简网格仅

包含一个顶点.

3.2 顶点合并表及/分裂操作

顶点树通过网格简化算法生成,虽然某些网格操作之间存在约束关系,但是有许多网格操作是相互独立的.因此选择某个合适的细节层次模型时,我们希望不按照网格简化算法确定的操作顺序进行网格操作,而是从顶点树中有目的地选择顶点节点对进行操作.

由于顶点树的每个节点保存了足够的操作信息,使得顶点合并操作 vunify 和顶点分裂操作 vsplit 成为一组对偶操作.

1)顶点合并操作 vunify. 顶点树中的顶点合并操作与网格简化过程中的顶点合并操作十分类似,顶点树利用一条双向顶点链表示当前网格的顶点集合,例如连接所有叶节点的顶点链表示了原始网格,而连接所有顶点树根节点的顶点链则代表了最简网格.顶点合并操作过程如下:

a)首先选取两个不同的顶点$\{s\}$,$\{t\}$;$\{s\}$,$\{t\}$必须都在当前顶点链中,并且$\{s\}$,$\{t\}$是同一个父节点$\{u\}$下的两个子节点;

b)删除父节点$\{u\}$中枚举的三角形集合,将其从当前网格三角形链中移去;

c)将顶点$\{s\}$,$\{t\}$从当前顶点链删去,同时把父节点$\{u\}$插入顶点链;

d)修改顶点映射表.

2)顶点分裂操作 vsplit. 顶点分裂操作 vsplit 是顶点合并操作 vunify 的逆过程:

a)选取当前顶点链上的任意顶点$\{u\}$,$\{u\}$包含两个不是当前节点的子节点$\{s\}$,$\{t\}$;

b)产生父节点$\{u\}$中枚举的三角形集合,将其中所有三角形插入当前的三角形链中;

c)将顶点$\{u\}$从当前顶点链删去,同时把子节点$\{s\}$,$\{t\}$插入当前顶点双向链中;

d)修改顶点映射表.

为了加快速度,顶点合并和顶点分裂都使用了顶点映射表完成三角形的结构转换.通过一张顶点索引表 VI 和基于 VI 的映射关系 $g:V\rightarrow V$,以便快速搜索与顶点邻接的三角形等几何元素

$$g(v)=\begin{cases} v, & \mathrm{VI}[v]=-1, \\ g(\mathrm{VI}[v]), & \mathrm{VI}[v]\neq -1. \end{cases} \tag{4}$$

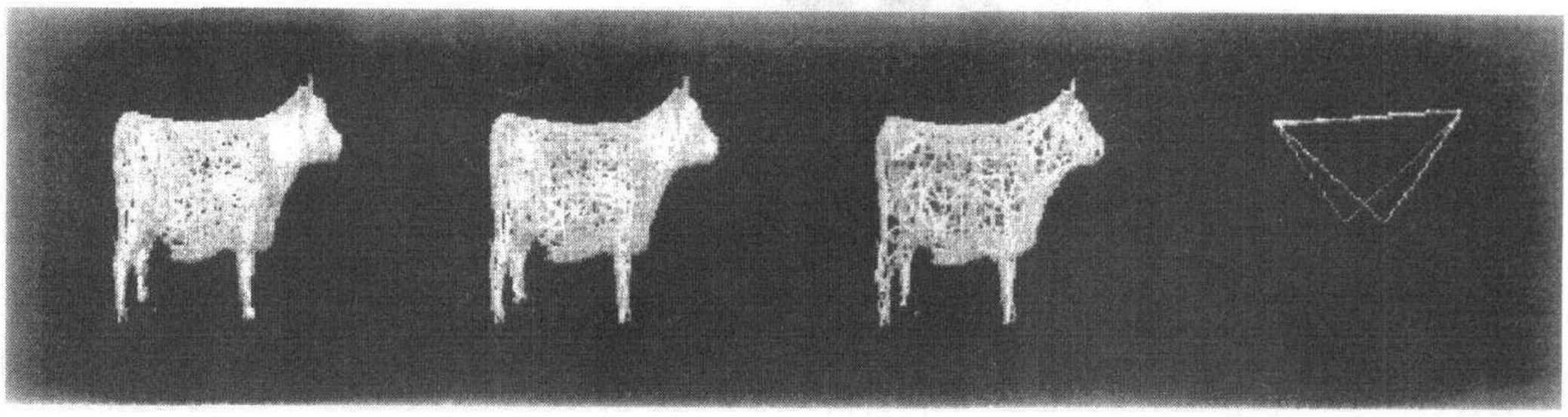
3.3 细节模型的绘制

为了绘制的方便,采用一条三角形链保存当前网格中所有的三角形,绘制时程序只需计算三角形链上的三角形.因此每次网格操作不仅需要修改顶点链,还要修改三角形链.由于视点之间存在相关性,每次随视点变换而引起的场景变化不大,所以可以直接在当前两条链的基础上进行网格操作迭代,这样,只要做一些局部网格操作就能得到新的 LoD 模型.

在场景绘制中,选择合适的细节模型不仅取决于对象的远近,还应当包括观察视区、可见性以及其他众多视觉因素.动态网格简化操作根据观察者视点位置的变化进行,本文使用三类判断,即对象视区内外判断、对象表面法向判断、和对象屏幕投影判断.

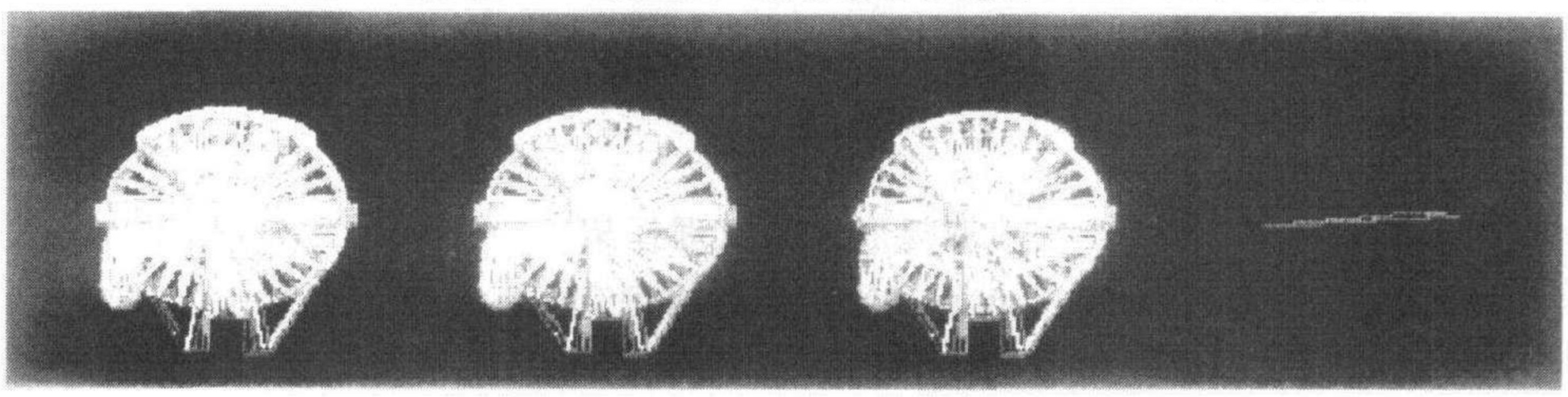
4　实验和实验结果

本文的简化方法可以处理任意网格,并且有较好的简化效果. 表3列出了相关实验数据,实验平台为联想微机,CPU 为 P Ⅱ-350,128M 内存. 模型例子参见图5.



a-1　　a-2　　a-3　　a-4

(a)牛模型(线框图,1～4模型的三角形数目分别为3535,2357,1179,2)



b-1　　b-2　　b-3　　b-4

(b)涡轮圈模型(平面绘制模型,1～4模型的三角形数目分别为3574,2384,1193,2)

图5　网格模型简化例子

注. a-1,b-1为原始网格;a-4,b-4为最简网格,仅包含两个三角形. 实验数据见表3.

表3　模型简化的时间及三角形数目统计

模型	原始三角形数目	简化模型三角形数目	简化时间(ms)	利用顶点树恢复时间(ms)	利用顶点树简化时间(ms)
牛	353 5	2	470	35	20
海豚	169 2	2	120	20	10
枪	821 0	2	163 1	31	50
涡轮圈	357 4	2	410	30	20
手	287 8	2	290	20	15

顶点树构造完成之后同时生成两个链表:当前顶点链和当前三角形链. 每当视点改变就搜索当前顶点链,根据视点的几个准则做顶点分裂或者顶点合并操作,并且对当前顶点链和当前三角形链做相应修改. 根据局部性理论,链表的修改通常发生在相邻区域,因此我们采用了一种分段处理方式,类似逐步求精. 每次变换视点后,并不对整个场景做简化或者细化处理,即并不搜索整个顶点链,相反每次只对其中一部分顶点做处理,下次视点转变时再接下去做剩下的. 实验表明这种方法可行,整个过程仍然能够收敛,而且能较大提高工作效率.

事实上,本文提出的顶点树结构不仅可以适用于基于两个顶点合并的网格简化算法,也适用于多于两个顶点之间合并的网格简化方法. 也就是说这个顶点树既可以为二叉树,

也可以是多叉树结构，而其他的结构基本不用改变. 根据这个方法在 Windows 系统上做了一个观察工具，并且测试了一组网格数据，取得了较好的结果，参见图6.

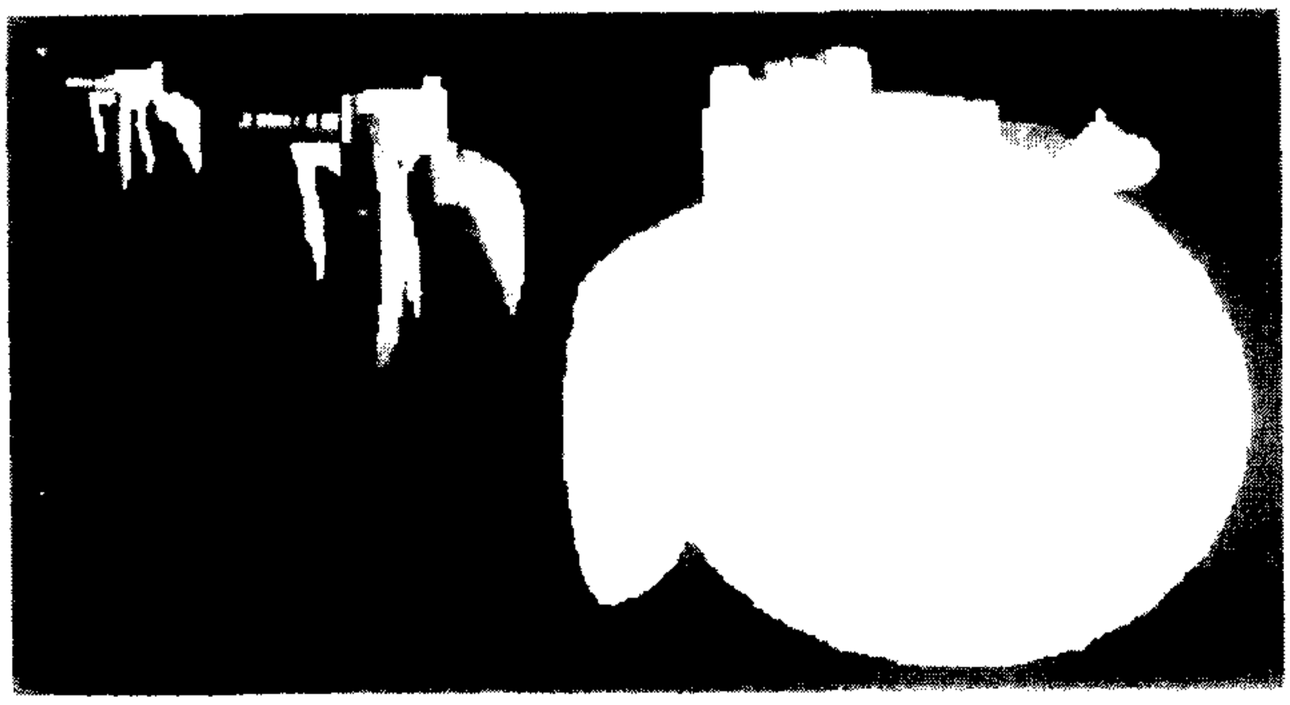


图6 视点相关的动态多细节层次

注. 图中是枪模型. 从左上角到右下角，网格模型分别含28，2258，3816，5814个三角形. 模型从远至近逐步精细.

5 结束语

作为一种有效的 LoD 模型的表现形式和管理框架，网格动态层次结构能够有效地配合复杂场景的实时绘制. 本文讨论了一种拓扑结构可变的网格动态简化方法. 在以后的研究中，我们将考虑更多的与视点相关的因素，研究更加适合于拓扑结构可变网格动态简化的网格简化方法，拓展网格动态层次框架的适用范围.

参 考 文 献

1 Hoppe H. Progress Meshes. In: proc. SIGGRAPH'96, New York: ACM press, 1996. 99～108

2 Hoppe H. View-dependent Refinement of Progressive Meshes. In: proc. SIGGRAPH'97, New York: ACM press, 1997. 189～198

3 Xia J, Varshney A. Dynamic View-dependent Simplification for Polygonal Models. In: proc. Visualization'96, Washington: IEEE, 1996. 327～334

4 Luebke D, Erikson C. View-dependent Simplification of Arbitrary Polygonal Environments. In: proc. SIGGRAPH'97, New York: ACM press, 1997

5 李捷，唐泽圣，郭红晖. 基于分形维数的层次多分辨率模型. 计算机学报，1998，**21**(9)：780～786

6 潘志庚，马小虎，石教英. 虚拟现实中多细节层次模型自动生成技术综述. 中国图像图形学报，1998，**3**(9)：754～759

7 Hoppe H, DeRose T, Duchamp T, Halstead M, Jin H, McDonald J, Schweitzer J, Stuetzle W. Piecewise Smooth Surface Reconstruction. In: proc. SIGGRAPH'94, New York: ACM press, 1994. 295～302

8 Popovic J, Hoppe H. Progressive Simplicial Complex. In: proc. SIGGRAPH'97, New York: ACM press, 1997

陶志良 1974年生，硕士. 现在华为公司上海研究所工作. 研究方向为虚拟现实和计算机图形学.

成迟慧 1976年生，浙江大学计算机系硕士研究生. 研究方向为虚拟现实和计算机图形学.

潘志庚 1965年生，博士、研究员、博士生导师. 研究方向为虚拟现实、分布式图形和多媒体计算.

石教英 1937年生，教授、博士生导师. 研究方向为科学计算可视化、虚拟现实和多媒体计算.