

文章编号:1001-9081(2009)03-0729-03

## 一种基于近似曲率的边折叠简化算法

张 果,刘旭敏,关 永

(首都师范大学 信息工程学院,北京 100048)

(zg226@163.com)

**摘 要:**针对目前三角网格简化算法在低分辨率的状态下往往丢失模型重要几何特征,从而导致视觉上的失真问题,提出了一种改进的边折叠三角网格简化算法。在 Garland 算法基础上引入了近似曲率的概念,并将其加入到二次误差测度中,使得二次误差测度在能够度量距离偏差的情况下,能够反映模型局部表面几何变化。实验结果表明改进的算法有效保持了模型的细节特征,简化效果更好。

**关键词:**网格简化;边折叠;二次误差测度;近似曲率;形状特征

**中图分类号:** TP394.41 **文献标志码:** A

## Edge collapse simplification based on similar curvature

ZHANG Guo, LIU Xu-min, GUAN Yong

(College of Information Engineering, Capital Normal University, Beijing 100048, China)

**Abstract:** To reserve more important shape features in the low-level model, an improved edge-collapse triangle mesh simplification algorithm was presented. On the basis of Garland's simplification algorithm, the concept of similar curvature was introduced and embedded into the original Garland's quadric error metric, so that the metric can not only measure distance error but also reflect geometric variations of local surface. The experimental results show that the improved algorithm effectively retains the detail characteristics of the model, and obtains a better simplified effect.

**Key words:** mesh simplification; edge-collapse; quadric error metric; similar curvature; shape feature

### 0 引言

在计算机图形学和几何造型中,物体表面常用多边形网格模型尤其是三角网格模型描述,这已经是几何造型中最直接最简单的方法。但是由于测量得到的数据分布十分密集,由此建立三维模型所得的三角网格通常由几十万个、几百万个,甚至上亿个三角面片组成。这样大的数据量对于计算机分析、显示与存储来说,都是一个挑战。特别是在虚拟手术、虚拟内窥镜等虚拟现实应用中,大规模数据的处理与要求绘制、分析等的实时响应是矛盾的。解决这些问题的一个途径就是对复杂的三维模型进行简化和多分辨率建模,用比较简单的几何模型来代替复杂的原始模型,以减少数据量,加快处理速度,节约存储空间。

现有的网格简化策略<sup>[1]</sup>中,几何元素删除法应用最为广泛,它包括顶点删除法、三角形折叠法、边折叠法等。其中,前两种方法需要对产生的空洞重新三角化,处理比较费时。相比之下,边折叠法只需将相连的两个顶点合并到一个新顶点上,并延续原有的拓扑关系,具有简单、方便、折叠误差小的特点,效率大大提高,而且鲁棒性更好,所以它成为一种常用的几何模型简化方法。

边折叠法是 Hoppe 等人<sup>[2]</sup>在 1993 年提出来的,其关键是确定边折叠的次序和折叠后新顶点的位置,Hoppe 使用能量函数作为误差测度得到了很好的简化效果,但由于考虑了全局信息,计算量太大;Garland 等人<sup>[3]</sup>于 1997 年提出基于二次误差测度的边折叠简化算法(简称 QEM 算法),以顶点到相关三角平面距离

的平方和作为误差测度,算法速度快,而且能生成高质量的简化模型。但是 QEM 算法的误差测度标准过于单一,所产生的简化网格都是均匀的,在低分辨率的状态下仍然不能有效地表示模型的一些重要几何特征,未能满足更广泛的应用需求。

网格模型简化的目标是在尽量保持模型逼真度的前提下,大幅度地减少三角形面片的数目。所以,一个好的模型简化方法应具有保持原始模型特征的特点<sup>[4]</sup>。改进的算法作为一种边折叠网格简化算法,不仅继承了 Garland 简化算法计算速度快、简化质量好的优点,而且克服了网格分布过于均匀、不能突出模型重要特征的缺点。

### 1 相关知识

边折叠简化算法首先计算模型中所有边在折叠时的代价,然后按折叠代价从小到大进行排序,每次从队头取出一条边进行折叠,当对该边折叠之后,相关联的边都从优先级队列中删除,重新计算其折叠的代价,并且将满足折叠条件的边按优先级顺序插入队列,继续进行边折叠,直到没有边满足折叠条件为止。图 1 为边折叠前后的比较图。

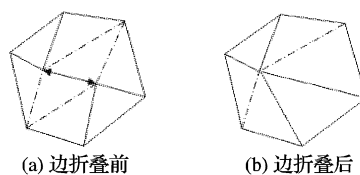


图 1 边折叠前后比较

Garland 算法是基于二次误差测度的边折叠算法,以顶点

收稿日期:2008-09-18;修回日期:2008-11-11。 基金项目:国家自然科学基金资助项目(0873006);北京市教育委员会科技发展计划重点项目(KZ200710028014);北京市自然科学基金资助项目(4082009)。

作者简介:张果(1985-),女,河南洛阳人,硕士研究生,主要研究方向:计算机图形学;刘旭敏(1956-),女,辽宁锦州人,教授,主要研究方向:计算机图形学、数据挖掘;关永(1966-),男(满族),辽宁凤城人,教授,博士,主要研究方向:智能信息处理、嵌入式系统设计。

到相关三角平面距离的平方和作为误差测度,算法速度快,简化质量较高,成为一种经典的边折叠算法。

该方法对每个顶点  $\mathbf{v} = [v_x \ v_y \ v_z]^T$  定义其误差为  $\mathbf{v}$  与其相关联平面集合  $planes(\mathbf{v})$  的距离平方和。这个误差测度可以写成二次型形式:

$$\Delta(\mathbf{v}) = \sum_{p \in plane(\mathbf{v})} (\mathbf{p}^T \mathbf{v})^2 = \sum_{p \in plane(\mathbf{v})} \mathbf{v}^T (\mathbf{p} \mathbf{p}^T) \mathbf{v} = \mathbf{v}^T \left( \sum_{p \in plane(\mathbf{v})} \mathbf{K}_p \right) \mathbf{v} \quad (1)$$

其中  $p$  是方程  $ax + by + cz + d = 0$  ( $a^2 + b^2 + c^2 = 1$ ) 定义的与  $\mathbf{v}$  相关联三角形所在平面,  $\mathbf{K}_p$  是平面  $p$  的基本误差二次型,即:

$$\mathbf{K}_p = \mathbf{p} \mathbf{p}^T = \begin{bmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & c^2 & cd \\ ad & bd & cd & d^2 \end{bmatrix} \quad (2)$$

把  $\mathbf{Q}_v = \sum_{p \in plane(\mathbf{v})} \mathbf{K}_p$  称为顶点  $\mathbf{v}$  的二次误差测度矩阵。当进行边折叠  $(\mathbf{v}_i, \mathbf{v}_j) \rightarrow \mathbf{v}'$  时,边折叠代价为:

$$\Delta(\mathbf{v}') = \mathbf{v}'^T (\mathbf{Q}_i + \mathbf{Q}_j) \mathbf{v}' \quad (3)$$

折叠后  $\mathbf{v}'$  的二次误差测度矩阵为:

$$\mathbf{Q}_{v'} = \mathbf{Q}_i + \mathbf{Q}_j = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ q_{14} & q_{24} & q_{34} & q_{44} \end{bmatrix} \quad (4)$$

由式(3)可见,确定新顶点位置的方法的不同将直接导致边折叠误差的不同,从而影响模型简化过程中边折叠的顺序,最终影响简化模型的精度。简单的方法是选择  $\mathbf{v}_i, \mathbf{v}_j$  或  $\frac{\mathbf{v}_i + \mathbf{v}_j}{2}$ 。但是  $\mathbf{v}'$  的选择显然存在一个最佳位置。将式(3)转换为式(5),得到折叠误差的二次多项式表示:

$$\Delta(\mathbf{v}') = \mathbf{v}'^T \mathbf{Q} \mathbf{v}' = q_{11}x^2 + 2q_{12}xy + 2q_{13}xz + 2q_{14}x + q_{22}y^2 + 2q_{23}yz + 2q_{24}y + q_{33}z^2 + 2q_{34}z + q_{44} \quad (5)$$

选择的原则是使  $\Delta(\mathbf{v}')$  达到局部极小值。这里通过解决线性优化的方法,可通过对式(5)中的  $x, y, z$  分别求偏导数,并令其为零,即:

$$\frac{\partial \Delta(\mathbf{v}')}{\partial x} = \frac{\partial \Delta(\mathbf{v}')}{\partial y} = \frac{\partial \Delta(\mathbf{v}')}{\partial z} = 0 \quad (6)$$

这样就得到一个线性方程组(7):

$$\begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{v}' = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (7)$$

如果方程组(7)有唯一解,则  $\mathbf{v}'$  由式(8)给出;否则,取  $\mathbf{v}_i, \mathbf{v}_j, \frac{\mathbf{v}_i + \mathbf{v}_j}{2}$ , 这三个点中使式(3)值最小的点作为折叠后新的顶点。

$$\mathbf{v}' = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (8)$$

## 2 改进的算法

在基于二次误差测度的边折叠简化算法中,边折叠的顺序和新顶点的位置是由二次误差测度决定的,因此如果在每个二次

误差测度矩阵上乘一个自适应的权值,将改变边折叠的顺序,同时并不改变二次型的基本性质以及初始误差值<sup>[5]</sup>。

在二次误差测度中,Garland 用三角形面积作为权值,对保持模型整体质量取得一定效果,但三角形面积并不能反映局部表面变化程度,因此简化后模型网格分布仍然较为均匀。

为了使这个权值能够连续反映局部表面变化,同时能够自适应地选取,引入了近似曲率这个概念,用它来反映三角网格模型中顶点处的弯曲程度,并将其作为权值嵌入到二次误差测度中。

在三角形网格模型中,由于大片的平滑区域中三角形的顶点处的弯曲程度一般都较小,而丰富细节的边缘处一般来说三角形的密度较大,其顶点处的弯曲程度一般都较大,所以用顶点的近似曲率作为这个自适应权值的主要依据,能更好地保持模型的几何特征。

### 2.1 顶点的近似曲率的计算

由于被简化的多边形表面是由一系列的三角形构成的,它不是二阶可微的曲面图形,从理论上来说,它不存在曲率,但它可以看作是光滑表面的分段近似<sup>[6]</sup>。

改进的算法用式(9)来代表顶点处的近似曲率,它表示模型中顶点处的弯曲程度:

$$K_s = \sum_{v_i \in neiverts(v_s)} c_{si} / m_i \quad (9)$$

其中,  $c_{si}$  是顶点  $\mathbf{v}_s$  和  $\mathbf{v}_i$  的法向量夹角的余弦值,  $\mathbf{v}_i$  是与  $\mathbf{v}_s$  相关的顶点即  $\mathbf{v}_i \in neiverts(\mathbf{v}_s)$ ,  $m_i$  是与顶点  $\mathbf{v}_s$  相关的顶点的个数,  $K_s$  表示了  $\mathbf{v}_s$  与其相关顶点的法向量之间的夹角余弦值的平均值。其中,

$$c_{si} = \cos(\mathbf{n}_s, \mathbf{n}_i) = \frac{\mathbf{n}_s \cdot \mathbf{n}_i}{\|\mathbf{n}_s\| \|\mathbf{n}_i\|} \quad (10)$$

$\mathbf{n}_s$  是顶点  $\mathbf{v}_s$  的法向量,  $\mathbf{n}_i$  是顶点  $\mathbf{v}_i$  的法向量,由公式可以看出,  $c_{si}$  的值越大,这两个顶点法向量之间的夹角就越小,顶点处的弯曲程度就越大,近似曲率就越大;  $c_{si}$  的值越小,这两个顶点法向量之间的夹角就越大,顶点处的弯曲程度就越小,近似曲率就越小。这里,顶点的法向量用与其相关的三角平面的法向量的和来表示。

因此,  $K$  值的大小反映了顶点  $\mathbf{v}_s$  的邻域内的近似曲率的大小。  $K$  值越大,顶点  $\mathbf{v}_s$  的邻域内的近似曲率越大,模型表面的几何变化越明显;  $K$  值越小,顶点  $\mathbf{v}_s$  的邻域内的近似曲率越小,模型表面的几何变化就不明显。

### 2.2 算法的改进

为了使得二次误差测度在能够度量距离偏差的情况下,而且能够反映模型局部表面几何变化,用顶点的近似曲率作为自适应权值嵌入到二次误差测度中,由此在式(3)的基础上得到式(11):

$$\Delta(\mathbf{v}') = \mathbf{v}'^T (\mathbf{Q}_s + \mathbf{Q}_t) \mathbf{v}' - \frac{1}{2} (K_s + K_t) = \mathbf{v}'^T (\mathbf{Q}_s + \mathbf{Q}_t) \mathbf{v}' - \frac{1}{2} \left( \sum_{v_i \in v_s, neiverts} \frac{c_{si}}{m_i} + \sum_{v_j \in v_t, neiverts} \frac{c_{tj}}{m_j} \right) \quad (11)$$

式(11)中,前半部分  $\mathbf{v}'^T (\mathbf{Q}_s + \mathbf{Q}_t) \mathbf{v}'$  表示顶点  $\mathbf{v}'$  到与其相关三角平面的距离平方和,而括号中的两项分别给出了顶点  $\mathbf{v}_s$  和  $\mathbf{v}_t$  与其相关顶点的法向量之间的夹角余弦值的平均值,这个值的大小反映了顶点  $\mathbf{v}_s, \mathbf{v}_t$  的邻域内的近似曲率的大小,因此这两项的平均值越大,说明在边  $(\mathbf{v}_s, \mathbf{v}_t)$  的邻域内网格表面的弯曲程度越大,几何形状变化也越明显,于是这条边

的删除就更应该后置。

另外,由于这两项的取值总是在 $[-1, 1]$ 中,为防止它们在边长很大的情况下失去作用,对上式进一步修改如下:

$$\Delta(\mathbf{v}') = \mathbf{v}'^T (\mathbf{Q}_s + \mathbf{Q}_t) \left[ 1 - \frac{1}{2} (K_s + K_T) \right] \mathbf{v}' = \mathbf{v}'^T (\mathbf{Q}_s + \mathbf{Q}_t) \left[ 1 - \frac{1}{2} \left( \frac{\sum_{v_i \in v_s, \text{neiverts}} c_{si}}{m_i} + \frac{\sum_{v_j \in v_t, \text{neiverts}} c_{tj}}{m_j} \right) \right] \mathbf{v}' \quad (12)$$

有了以上的边权定义,简化时网格的几何特征就能够得到较好的保持。

### 3 边界问题

对于有边界的模型,在简化时应该尽可能保持其边界。为此,为每个边界边生成一个虚拟平面,这个平面通过该边界边,并垂直于边界边所在的三角平面。给该虚拟平面规定一个重要度作为该平面基本误差二次型的权值,然后累加到该边界边端点初始的二次误差测度矩阵中。由于虚拟平面重要度比较大,因此边界边的折叠代价相对较高,边界可以得到很好的保持。其中,虚拟平面基本误差二次型权值的大小应选得适当,太小起不到保持边界的作用,太大边界处的三角片过密,影响效果。

### 4 优先队列的存储问题

在边折叠简化算法中,每一次循环都要在网格中寻找下一条要删除的边,即折叠代价最小的边。这可以通过全局比较搜索的方法获得,但是全局搜索的过程非常耗时,对于一个有 $N$ 条边的网格模型,其简化的时间复杂度为 $O(N^2)$ 。为了降低搜索的复杂度,改进的算法使用了一个最小堆来对所有边的折叠代价进行排序,这样在算法的第一步中就可以直接从堆中读出折叠代价最小的边。利用建堆的方法来实现各元素的排序,所需要的时间为 $O(N \log N)$ 。

### 5 结果分析

为了验证算法的效果,用 Visual C++ 6.0 实现了该算法,并用 OpenGL 完成图形的绘制,并把分别用改进算法与 Garland 算法得到的简化模型进行了比较,如图 2 和 3 所示。

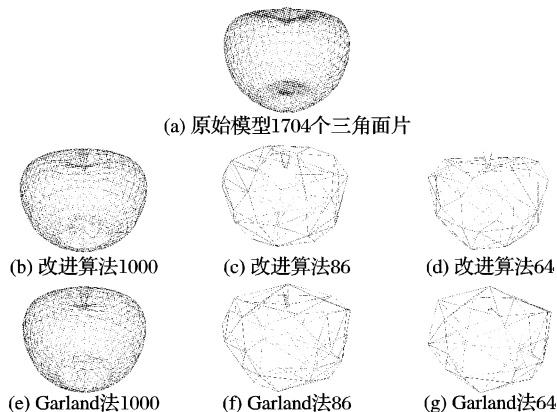


图2 苹果模型的特征对比

很显然,应用 Garland 算法得到的简化模型网格十分均匀,这样往往使得较为平坦的区域占用过密的网格,从而造成网格浪费,而且容易丢失模型的几何特征。改进的算法在一定程度上能够恢复模型的尖锐特征,能够按模型表面的近似

曲率变化合理地分配网格,使得平坦区域网格稀疏,而弯曲程度比较大的区域网格稠密。

在图 2 中,(b)~(g)为采用两种算法对苹果模型进行简化的对比图。(b),(e)为原始模型(a)采用两种简化算法简化到 1000 个三角面片时的简化结果,采用改进的简化算法,模型的简化已经集中在近似曲率比较小即较为平坦的区域,如苹果表面,而采用 Garland 算法,却仍然集中在具有尖锐特征的近似曲率比较大的区域,如苹果梗,这样很容易丢失模型的几何特征。(c),(f)为采用两种算法简化到 86 个三角面片时得到的简化结果,很容易看出,利用改进算法得到的简化模型能够更好地反映初始模型的大部分几何特征,而利用 Garland 算法得到的简化模型一些重要的几何特征已经表现得不明显。再看(d),(g),对比结果将会更明显。当模型简化到 64 个三角面片时,采用 Garland 算法得到的简化模型一些重要的几何特征已经完全退化,而采用改进的算法却得到了很好的保持。

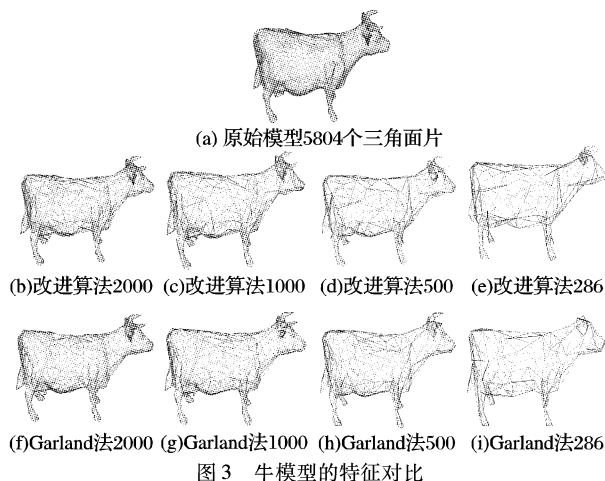


图3 牛模型的特征对比

在图 3 中,(b)~(i)为采用两种算法对牛模型进行简化的对比图。

很显然采用改进的算法能够更好地保持初始模型的大部分几何特征,随着三角面片数量的减少,简化效果会更明显。在同样简化到 286 个三角面片时,(i)中牛的角已经不明显,而(e)中牛的角却得到了很好的保持。

采用 Cignoni 等人在 1998 年提出的方法<sup>[7]</sup>对其进行误差估计,令两个模型的最大误差为 $E(S_1, S_2) = \max_{p \in S_1} e(p, S_2)$ ,平均误差为 $E_m(S_1, S_2) = \frac{1}{|S_1|} \int_{S_1} e(p, S_2) ds$ 。其中 $e(p, S)$ 为点 $p$ 到网格模型 $S$ 的距离, $e(p, S) = \min_{p' \in S} d(p, p')$ , $d(p, p')$ 为空间两点的欧氏距离。误差对比的结果见表 1 和 2。

表1 苹果简化模型误差对比

算法	1016 个三角面片		160 个三角面片	
	最大误差	平均误差	最大误差	平均误差
Garland	0.153 892	0.014 381	0.371 440	0.052 834
改进算法	0.214 998	0.016 610	0.378 906	0.055 051

对大部分模型来说,表面近似曲率变化大的区域往往只占整个模型的很小一部分,模型的大部分区域都比较平坦,因此应用改进的算法,简化模型的整体误差会有所增大,但从视觉效果上来看,这种增大的误差并不妨碍我们对物体的辨别,甚至还优于应用 Garland 算法得到的简化模型。

(下转第 816 页)

表 1 程序自动脱壳测试数据

加壳工具	版本	加壳文件 大小/KB	采集的数据 大小/B	构造基本 块个数	调用 API 个数	.text 节 相似度
无	N/A	65	N/A	21 800	1 008	N/A
UPX	2.03w	47	342 794	21 948	1 012	0.884 0
ASPack	2.12b	52	365 865	22 002	1 017	0.996 1
PECompact	2.70	49	389 133	22 166	1 024	0.972 0
WinUPack	0.39	44	371 657	21 902	1 011	0.997 9
PESpin	1.32	82	528 381	23 659	1 034	0.920 7
ASProtect SKE	2.3	388	1 933 237	29 716	1 071	0.963 0
Armadillo	4.40.0250	716	1 807 540	36 595	1 412	0.630 3
Themida	1.9.5.0	1 256	5 738 524	51 824	1 056	0.782 9

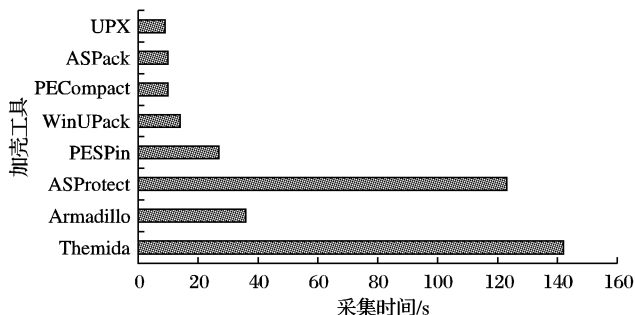


图 3 程序脱壳数据采集时间

上述时间测量样本是基于单机的采集环境得到的。对于实际的采集系统,可利用网络构造采集环境,以获得更理想的结果。

## 5 结语

本文研究基于系统仿真和动态监控的程序自动脱壳数据采集系统的设计思路,利用定制的 QEMU 仿真器可动态监控程序脱壳过程中改写的存储器内容,记录程序脱壳的执行行为。所采集的数据不仅可以用于恶意代码扫描和检测,进行分析整理后也可以用于原程序的重构。当然,完整地实现程序重构还有一些问题需要考虑,如动态行为监控的完备性、QEMU 仿真器反检测技术、程序入口点定位与还原等,这也是我们下一阶段研究的重点。

### 参考文献:

[1] MASHEVSKY Y. Watershed in malicious code evolution [EB/OL].

[2008-08-12]. <http://www.viruslist.com>.

- [2] McAfee Avert Labs. McAfee Avert Labs unveils predictions for top ten security threats in 2007 as hacking comes of age [EB/OL]. [2008-08-12]. <http://www.mcafee.com>.
- [3] Panda Research. Mal(ware) formation statistics [EB/OL]. [2008-08-12]. <http://research.pandasoftware.com>.
- [4] BROSCHE T, MORGENSTERN M. Runtime Packers: The Hidden Problem? [EB/OL]. [2008-08-26] <http://www.blackhat.com/presentations/bh-usa-06/BH-US-06-Morgenstern.pdf>.
- [5] ROYAL P, HALPIN M, DAGON D, *et al.* PolyUnpack: Automating the hidden-code extraction of unpack-executing malware [C]// Proceedings of 2006 Annual Computer Security Applications Conference (ACSAC). Washington, DC: IEEE Computer Society, 2006: 289-300.
- [6] KANG M G, POOSANKAM P, YIN H. Renovo: A hidden code extractor for packed executables [C]// Proceedings of the 2007 ACM Workshop on Recurring Malcode (WORM 2007). New York: ACM, 2007.
- [7] CHRISTODORESCU M, KINDER J, JHA S, *et al.* 1539, Malware normalization [R]. Madison, WI, USA: University of Wisconsin, 2005.
- [8] YASON M V. The art of unpacking [EB/OL]. [2008-08-25]. <http://www.us.ibm.com>.
- [9] BELLARD F. QEMU: A fast and portable dynamic translator [C]// Proceedings of the 2005 USENIX Conference. [S.l.]: USENIX Association, 2005: 41-46.
- [10] AHO A V, SETHI R, ULMAN J D. 编译原理 [M]. 李健中, 江守旭, 译. 北京: 机械工业出版社, 2007.

(上接第 731 页)

表 2 牛简化模型误差对比

算法	4 940 个三角面片		2 924 个三角面片	
	最大误差	平均误差	最大误差	平均误差
Garland	0.071 434	0.001 477	0.094 674	0.005 831
改进算法	0.081 810	0.001 622	0.116 255	0.006 412

通过实验可知,该算法得到了较好的实验效果。

## 6 结语

改进的算法引入了近似曲率的概念,并把它作为自适应权值嵌入到二次误差测度中,在不改变二次误差测度矩阵性质同时,有效地克服了 Garland 算法网格分布均匀、对表面变化反映不够精确的不足,合理地分配了网格密度,使得简化模型在较低分辨率下能够保持更多的重要几何特征,有效地降低了视觉失真。通过实验,验证了该算法并得到了较好的实验效果。

### 参考文献:

- [1] 田捷,包尚联,周明全. 医学影像处理与分析 [M]. 北京: 电子工业出版社, 2003.
- [2] HOPPE H, DEROSE T, DUCHAMP T, *et al.* Mesh optimization [C]// CUNNINGHAM S, ed. SIGGRAPH'93. New York: ACM Press, 1993: 9-26.
- [3] GARLAND M, HECKBERT P S. Surface simplification using quadric error metrics [C]// Proceedings of SIGGRAPH 97. Los Angeles: Addison Wesley, 1997: 209-216.
- [4] 刘晓利, 刘则毅, 高朋东, 等. 基于尖特征度的边折叠简化算法 [J]. 软件学报, 2005, 16(5): 669-675.
- [5] 杜晓晖, 尹宝才, 孔德慧. 一种边折叠三角网格简化算法 [J]. 计算机工程, 2007, 33(12): 12-15.
- [6] 郭震宇. 三角网格模型的简化技术及多细节层次模型 [D]. 大连: 大连理工大学, 2006.
- [7] CIGNONI P, ROCCHINI C, SCOPIGNO R. Metro: Measuring error on simplified surfaces [J]. Computer Graphics Forum, 1998, 17(2): 167-174.