·信号与信息处理·

基于视觉效果的网格简化方法研究与QEM算法实现

吴婧文,全吉成,赵秀影,刘　宇

(空军航空大学,吉林　长春　130022)

摘　要:由于计算机在处理大规模三维网格的过程中会出现严重的延迟,影响场景显示的视觉效果,主要研究了以视觉效果为首要条件的三维网格模型简化方法及QEM算法。首先,对网格简化方法进行了分类;其次,给出了网格简化的误差评判方法;最后,基于OpenGL,采用顶点对收缩方法,并结合QEM算法进行了实现。模型简化了93.7%时,仅用时111.78 ms,同时保证了输出网格的视觉效果。

关键词:三维模型;视觉效果;网格简化;二次误差测度

中图分类号:TP391.41　　**文献标识码**:A　　**文章编号**:1673-1255(2012)03-0059-06

Study on Mesh Simplification Methods Based on Visual Effect and Implement of QEM Method

WU Jing-wen, QUAN Ji-cheng, ZHAO Xiu-ying, LIU Yu

(*Aviation University of Air Force, Changchun* 130022*, China*)

Abstract: The computer may cause the serious delay when it processes the large 3D mesh, which influences the visual effect of the scene. Taking the visual effect as the primary condition, the simplification method of 3D mesh model and QEM method are studied. Firstly the simplification methods are classified. Then the error evaluation methods of the simplification mesh are studied. Finally the QEM method is implemented based on OpenGL. It only takes 111.78 ms to accomplish 93.7% of the simplification, meanwhile the visual effect is also remained.

Key words: 3D models; visual effect; mesh simplification; quadric error metrics

在三维场景的动态交互过程中,常常遇到三维模型网格数目过大而使场景的渲染速度过慢的情况。为实现流畅而逼真的三维效果,通常需要将场景中的三维模型进行网格简化,以达到计算机渲染速度和场景展现效果的最佳平衡。目前已经有很多国内外学者对这一领域进行了研究,概括来讲,网格简化方法主要有两大类:一类是以逼真程度为首要条件的简化方法。这种方法要求经过简化的模型外观要达到特定的视觉逼真效果,在保证视觉效果的前提下对网格中的面片数进行尽可能多的简化。这一"视觉逼真效果"通常由简化前后模型之间的误差来进行定量评价。另一类是以简化误差代价为首要条件的简化方法。这种方法给定简化后的网格数目,要求以最小的误差代价来实现三维网格的简化。但是实际应用中这一方法无法保证视觉效果的逼真程度,并且达到最优化的误差代价相对而言也比较困难。

文中主要研究以视觉效果为首要条件的网格简化方法及其误差评价方法。首先对网格简化方法进行了分类研究,然后研究定量评价误差的方法,最后采用顶点对收缩方法,并结合QEM算法对模型进行简化。

收稿日期:2011-03-23

作者简介:吴婧文(1987-),女(回族),河北香河人,硕士研究生,研究方向为信息管理与信息系统;全吉成(1960-),男(朝鲜族),吉林和龙人,教授,博士,博士研究生导师,研究方向为信息管理与信息系统.

1 网格简化方法

1.1 边折叠方法

Hoppe[1]首先提出用边折叠的方法来对三维网格进行简化。这种方法将一条边折叠为一个顶点，整个网格模型因此减少一条边和其所连接的所有三角形。Hoppe的边折叠方法是可逆的，其可逆过程称为顶点分裂，整个网格模型会增加一条边和相应的三角形。

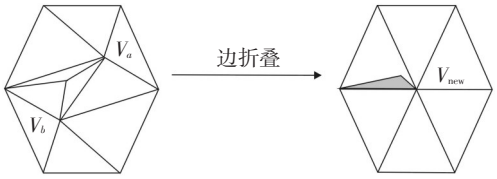
边折叠方法提出以来已经得到广泛应用，主要可以将这些应用分为两类，即半边折叠和全折叠。半边折叠方法是指在边折叠的过程中，新的顶点位置选在边的两个顶点上，即v_a或者v_b。而全折叠方法中的折叠顶点可以是通过计算得出的新的顶点位置。通常所说的边折叠方法是指全折叠。

相对于半边折叠而言，全边折叠方法在确定新顶点位置上有着更灵活的可选择性，因此其所实现的网格简化效果比半边折叠更逼真。然而半边折叠也有着全边折叠无法替代的优点：半边折叠所产生的网格是原模型的一个子集，因此简化过程中没有新的顶点产生，比全边折叠方法更节省空间资源；同时半边折叠所需处理的三角形数目也少于全边折叠，这使半边折叠在简化过程中效率更高。

边折叠方法的优点是应用简单，但是在实现的过程中需要注意：(1)网格的重叠：网格重叠是指在边折叠的过程中产生的缝隙或网格的叠加。可以在编程实现中对每次折叠都计算相应三角形法向量的变化，如果出现网格重叠，其法向量通常会有大于90°的变化，视觉上会出现纹理错乱的现象。(2)拓扑的不连续：通常一条边(v_a, v_b)由两个三角形所共用，因此与其端点v_a, v_b同属同一三角形的邻接顶点有1~2个。但当网格中有如图1所示的拓扑结构时(一条边的邻接三角形多于两个)，边折叠操作之后就会产生网格拓扑结构的错乱。图1a为原网格结构，进行边折叠后形成图1b中的非流线形结构。多数算法是无法对非流线形结构进行网格简化的，因此在边折叠的过程中要避免产生拓扑结构的变化。

1.2 顶点对收缩方法

顶点对收缩方法[2-4]将两个无边连接的顶点进行折叠，也称为虚拟边折叠方法(如图2)。这一方法可以将未连接的部分连接起来，并粘合网格中的空洞。



(a)原网格结构 (b)非流线结构

图1 边折叠操作引起的拓扑错乱

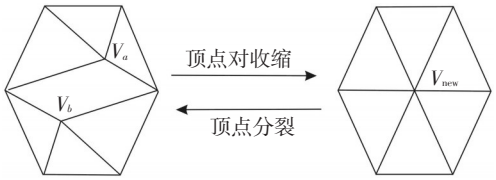


图2 顶点对收缩操作

对于一个有N个顶点的网格，顶点对收缩方法的时间复杂度将为$O(n^2)$。为降低时间复杂度，Garland做了卓有成效的工作。他所提出的算法中首先设定了两个顶点的距离阈值δ，距离小于δ的两个顶点是有效顶点，在算法中仅对有效顶点进行收缩操作。当δ=0时，顶点对收缩方法成为边折叠方法。

1.3 三角形折叠方法

三角形折叠方法将一个三角形(v_a,v_b,v_c)折叠到一个顶点v_{new}[5,6]。v_{new}可以是v_a ,v_b ,v_c中的一个顶点，或者是通过计算得到的一个新的顶点位置。三角形折叠方法可以看做是两次边折叠操作，其优点在于，将网格以三角形为单元进行存储，其结构没有边存储那样复杂，因此所需要的存储空间较小；但实际应用中，三角形折叠所得到的网格粒度相对边折叠要粗糙。

1.4 细胞折叠方法

细胞折叠方法将整个网格分为小的细胞单元，并将每个细胞中的顶点都进行折叠操作，最终的结果则是：每个细胞中都只有一个顶点或一条边。这一过程无法保持原网格的拓扑结构，简化后网格的分辨率层级则由细胞分割的细致程度决定。

细胞折叠方法的优势在于其同时简化了网格的几何结构和拓扑结构，有着良好的鲁棒性且较容易

编程实现。然而,当网格的位置和角度发生变化时,即视点位置发生变化时,由于网格细胞的分割也随着变化,因此网格简化的输出结果也会不同;同时若网格中存在空洞,若空洞位于细胞边缘,这一空洞将会被放大,而若出现于细胞内部,简化后则会消失。细胞折叠方法因此有着很大的不确定性。

1.5 顶点删除方法

顶点删除方法中一次操作删除一个顶点及其所连接的边以及三角形,然后将产生的空洞进行三角化(如图3)[7,8]。这一方法中三角化空洞的过程有很多方法,其中一种即是半边折叠法。因此顶点删除方法

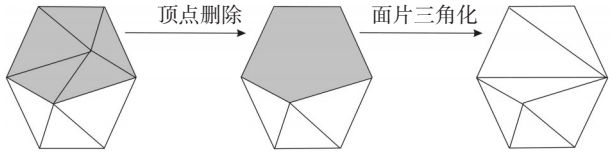


图3 顶点删除与三角化操作

可以看做是半边折叠方法的一个拓展,只是多了一个三角化网格的过程。

对一个凸面的三角化,其可能的方法有几种[9],在网格的离散优化中可以从这些方法中选择一种。

$$c(i)=\frac{1}{i+1}\times\binom{2i}{i}=\frac{1}{i+1}\times\frac{(2i)!}{i!(2i-i)!}=\frac{1}{i+1}\times\frac{(2i)!}{i!i!}=\frac{(2i)!}{(i+1)!i!}$$

1.6 面片融合方法

这一方法最早由Hinker和Hansen[10]提出。这一方法的思想是将距离较近的共面多边形和邻接多边形融合为一个大的多边形,再将这一大的多边形进行三角化。这一方法的过程与顶点删除方法类似,但比顶点删除方法更具操作性。因为面片融合方法并不局限于三角形网格结构,可以同时删除多个顶点并融合网格中的空洞。许多学者都对这一方法进行了研究,如文献[11]以及文献[12]的网格面融合等。

1.7 通用几何替代方法

这类方法将网格中的一个面片集以另一个较简单的面片集来代替,并保持其边界拓扑结构不变。文献[13]提出的多元三角化算法(multi-triangulation)就是这一思想的具体应用。将这一方法称为"通用"几何替代法,是因为面片集的替代过程包含了边折叠、顶点删除等多种算法,也包含了三角形中边的翻转。边折叠和边翻转操作足够适应所有的网格简化方法,因此这一方法也有着广阔的实际应用空间。

2 网格简化误差评判

三维网格简化之后与原网格有着一定的差别,可以接受的是使网格尽量简单但仍保持了模型外观的算法。误差评判主要用于对网格简化过程进行优化以及确定网格的简化质量。

网格简化误差的评价方法有很多,主要有以下几类:

2.1 几何误差

在计算机图形学中,三维网格中的顶点以三维坐标来表示,网格简化减少了顶点数目,也改变了网格中面片的形状。在简化过程中对三维几何误差加以度量可以将这一过程所产生的偏差最小,尽可能的保持与原网格一致的外观。

几何误差评判算法大概可以分为这样几类:计算点到点距离的误差[14,15],点到平面距离的误差[2,16],点到表面距离的误差[1],以及表面到表面的距离误差评判算法[17]等。

几何误差的实质是求简化前后网格中对应顶点之间的距离。两点之间的几何距离又称为欧氏距离,点$p_1=(x_1,y_1,z_1)$与点$p_2=(x_2,y_2,z_2)$之间的距离

$$Dis=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2} \quad (1)$$

但几何误差的度量过程中多需考虑两个平面之间的距离。平面可以看做是由无数个点构成的集合,然而逐点计算需要消耗大量的时间空间资源,因此在实际工程中可以将平面上的点划分为有限个子集,通过计算对应子集之间的最优距离来找到平面间距。常用的距离主要有:

(1) Hausdorff距离

Hausdorff距离在图像处理、表面建模和其他工程应用中有着广泛的应用。对于给定的两个点集A和B,Hausdorff距离是A和B中对应点的最小距离的最大值,即对于点集A中的每一个点都在B中找到距离最近的点,反之亦然。计算出这些点对之间的距离,其中的最大值就是点集A和B之间的Hausdorff距

离。Hausdorff距离公式如下

$$H(A, B) = \max(h(A, B), h(B, A)) \quad (2)$$

式中，$h(A,B)=\max_{a\in A}\min_{b\in B}\|a-b\|$

(2) 映射距离

Hausdorff距离有着一定的缺陷，即无法在两个点集之间建立逐点映射关系，点集中有些地区在另一点集中无法找到映射，有些地区又产生很多映射。因此用映射距离来解决这一问题。

对于给定的连续映射：$F:A\rightarrow B$ ，定义映射距离为 $D(F)=\max_{a\in A}\|a-F(a)\|$ (3)

映射关系的确立可以有很多种，如，可以利用纹理坐标来建立二维和三维之间的映射关系，在三维网格中的点通过映射函数F映射到二维纹理坐标空间中，则F^{-1}将纹理坐标空间中的点x映射到三维网格中。这时简化前后网格中对应点的距离为

$$D(F)=\max_{a\in A}\|F_{i-1}^{-1}(X)-F_i^{-1}(X)\| \quad (4)$$

在映射距离中，两个点集之间的最小距离是所有可能的映射中$D(F)$的最小值。

除Hausdorff距离与映射距离之外，还可利用屏幕空间来对网格简化误差进行评价，鉴于篇幅，这里不做一一介绍。

2.2 属性误差

三维模型发展到今天，其网格文件不仅包含了拓扑几何坐标信息，也包含着诸如颜色、法向以及纹理坐标等属性信息。这些信息在一定程度上反映了几何拓扑无法表达的内容，例如网格中的裂缝和褶皱处，其顶点的法线会有两个方向的向量。网格简化的各种算法也开始将属性信息纳入考虑范围，这样计算出的结果会更平滑，更符合实际需要。

2.2.1 颜色

通常网格中的颜色属性由一个三维数组(r,g,b)来表示，每个值都在[0,1]之间。最简单的处理方法是将这一颜色数组看成是欧式空间中的一组坐标向量，对应顶点颜色之间的差值即为

$$D_{\text{color}}=\sqrt{(r_1-r_2)^2+(g_1-g_2)^2+(b_1-b_2)^2} \quad (5)$$

在颜色属性的优化过程中，可以将(r,g,b)看做是3个独立分量，因此可以对这3个分量分别进行误差计算。

然而以颜色属性为标准对网格简化误差进行评价和优化时，由于顶点的颜色域并非线性，因此相同色差数值所产生的视觉效果可能会不同，对于这一现象，可以将颜色域从RGB转换到CIE-L*u*v*空间[18]。有时通过计算，使整体颜色误差最小的新顶点颜色值会超过[0,1]这个范围，因此还要将计算所得到的结果进行变换，将该颜色值变换到[0,1]范围内。

2.2.2 法向

两个向量之间的角度值用下式计算

$$d=\arccos\left(\left(n_{1x},n_{1y},n_{1z}\right)\cdot\left(n_{2x},n_{2y},n_{2z}\right)\right) \quad (6)$$

法向的计算多用于检验网格简化之后是否有网格重叠的现象。如果简化前的三角面法向量与简化后的法向量之间的角度大于某一设定的阈值，则此次简化操作是一次产生了网格重叠的操作，对法向的误差进行这样的评判之后可以有效地减少因简化而产生的网格重叠。

2.2.3 纹理坐标

网格表面的纹理坐标以坐标对(u,v)来表示，定义了二维纹理空间中的点。与颜色属性一样，纹理坐标这一属性的值也在[0,1]范围之内，不同之处在于，纹理坐标中的(u,v)值不能看做是独立的分量。纹理坐标的误差评判方法则是在部分网格内部首先寻找一个“内核”（该部分网格边线的包络区域），如果找不到这样一个内核，则说明对于这一部分的网格，其所有顶点进行折叠之后的新顶点上无论如何设置纹理坐标(u,v)的值，都将产生一个网格重叠。因此此次操作判定为非法操作。

颜色、法向、纹理坐标等属性因子可以进行加权处理，通过经验或是否会引起网格的拓扑变化等来设定权重。

3 算法实现

由于三维场景的动态交互中，对时间的复杂度要求较高，为了达到最快的简化效率和最逼真的效果，并提高场景的渲染速度，采用1.2节顶点对收缩方法结合QEM评价算法，并基于OpenGL对其进行了工程实现。

QEM算法（quadric error metrics.，二次误差测度算法）利用了1.2节中顶点对折叠的思想对三维网格进行简化，并利用点到平面距离的思想进行误差测度。QEM算法可以在相对较少的存储开支条件下快

速实现网格简化,同时保持了三维模型外观的逼真性,是一种可以在算法速度、输出网格的逼真度以及算法鲁棒性之间达到最佳平衡的方法。

网格表面的纹理坐标以坐标对(u,v)来表示,定义了二维纹理空间中的点。与颜色属性一样,纹理坐标这一属性的值也在[0,1]范围之内,不同之处在于,纹理坐标中的(u,v)值不能看做是独立的分量。纹理坐标的误差评判方法则是在部分网格内部首先寻找一个"内核"(该部分网格边线的包络区域),如果找不到这样一个内核,则说明对于这一部分的网格,其所有顶点进行折叠之后的新顶点上无论如何设置纹理坐标(u,v)的值,都将产生一个网格重叠。因此此次操作判定为非法操作。

颜色、法向、纹理坐标等属性因子可以进行加权处理,通过经验或是否会引起网格的拓扑变化等来设定权重。

3.1 算法简述

设三维网格中顶点v的坐标为$(v_x,v_y,v_z,1)^r$,平面p为$ax+by+cz+d=0$,其中

$a^2+b^2+c^2=1$,记$p=(a,b,c,d)^{\mathrm{T}}$。点v到平面p的距离平方为

$$d^2(v)=(p^{\mathrm{T}}v)^2=v^{\mathrm{T}}(pp^{\mathrm{T}})v=V^{\mathrm{T}}\boldsymbol{K}_pv \tag{7}$$

式中,$\boldsymbol{K}_p=pp^{\mathrm{T}}\begin{bmatrix}a^2 & ab & ac & ad\\ ab & b^2 & bc & bd\\ ac & bc & c^2 & cd\\ ad & bd & \mathrm{cd} & d^2\end{bmatrix}=\begin{bmatrix}q_{11} & q_{12} & q_{13} & q_{14}\\ q_{12} & q_{22} & q_{23} & q_{24}\\ q_{13} & q_{23} & q_{33} & q_{34}\\ q_{14} & q_{24} & q_{34} & q_{44}\end{bmatrix}$。

Garland定义网格中一个顶点的二次误差测度为该点到其相应三角面的距离平方和[19]

$$\begin{aligned}\triangle(V)&=\sum p\in planes(v)d_p^2(v)=\\&\sum p\in planes(v)v^T(\boldsymbol{K}_p)^v=\\&vT(\sum p\in planes(v)\boldsymbol{K}_p)v\end{aligned} \tag{8}$$

定义$\boldsymbol{Q}=\sum p\in planes(v)\boldsymbol{K}_p$为顶点$v$的二次误差矩阵,边收缩操作$v_1,v_2\rightarrow\overline{v}$的二次误差测度为顶点$v_1$和$v_1$的二次误差之和,即

$$\Delta'(\overline{v})=(\overline{v})T(Q_1+Q_2)\overline{v} \tag{9}$$

可以写成

$$Error(\overline{v})=\overline{v}T(\sum k_p)\overline{v}T=\overline{v}Q'\overline{v} \tag{10}$$

式中,$\boldsymbol{K}_p'$是指两个顶点在收缩操作中的二次误差矩阵之和。

3.2 算法实现步骤

算法的实现可分为初始化和迭代两部分。

3.2.1 初始化阶段

引入头文件#include <gl\gl.h>以及#include <gl\glu.h>,并分别在顶点、三角形的头文件中定义顶点和网格中三角形的数据结构如下:

Vertex{//顶点数据结构

_myVertex(x,y,z),//顶点的坐标值,并定义函数getXYZ()用于读取当前索引顶点的X,Y,Z坐标值。

_bActive(BOOL),//指示该顶点是否是有效顶点

_cost(INT),//存储该顶点收缩代价

_minCostNeighbor(INT),//代价最小边的另一个顶点

_index(INT)//顶点的索引值

}

Triangle{//三角形数据结构

_vert1(), _vert2(), _vert3() ,//三角形的三个顶点,并定义getVerts(int& v1, int& v2, int& v3)来获取三角形的三个顶点;

_mesh(),//存储当前网格的指针

bActive(BOOL),//当前三角形是否有效

_index()//三角形的索引值

}

①计算网格中每个顶点的二次误差矩阵$\boldsymbol{Q}$;

对于顶点来说,其二次误差标志着其折叠的权重大小,即决定着是否对该顶点进行收缩操作。根据前述QEM算法,程序伪代码如下:

For(遍历该顶点所在的所有三角形){

二次误差矩阵$\boldsymbol{Q}=\boldsymbol{Q}$+三角形法向矩阵$\boldsymbol{K}_P$

}

②根据预先设定的顶点对选择条件选择有效顶点对;

③对每一对有效顶点,计算其折叠后的二次误差,并将结果放入堆栈。伪代码如下:

For(当前指针所指顶点v){

计算v的二次误差矩阵$\boldsymbol{Q}$;

For(遍历该顶点的所有邻接顶点n){

计算n的二次误差矩阵$\boldsymbol{Q}$';

$\boldsymbol{Q}_{总}=\boldsymbol{Q}+\boldsymbol{Q}$';

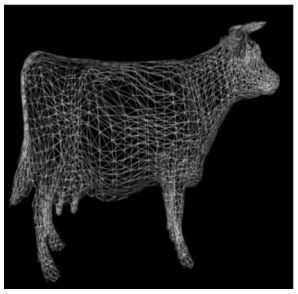
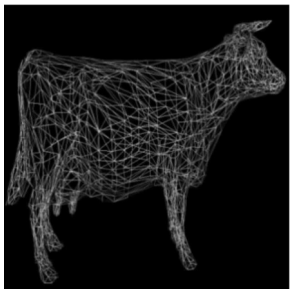
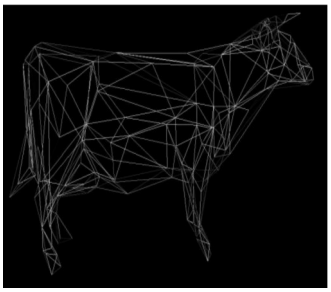
}

根据$\boldsymbol{Q}_{总}$计算顶点的二次误差,即顶点的收缩代价cost;

If(cost<mincost)cost=mincost;

}

表1 牛模型的QEM简化结果

模型网格			
三角形数	5 804个三角形 (100%)	2 636个三角形 (简化54%)	326个三角形 (简化93.7%)
简化用时	-(原始模型网格)	66.36 ms	111.78 ms

3.2.2 迭代阶段

①选择二次误差值最小的顶点对(V_1,V_2);

②收缩顶点对(V_1,V_2);

③更新网格中的所有顶点对,重新计算每一对的$\boldsymbol{Q}$值;

④迭代这一过程,直到输出的网格满足简化需求。

3.3 算法实验结果

文中实验环境为因特尔志强8核处理器,4 G内存, NVIDIA GeForce GTX 580显卡,基于OpenGL和c++进行程序设计。程序执行结果如表1所示。

4 结 论

从上述实验结果可以看出,QEM算法在保持了原网格的拓扑结构的同时对三维网格进行了大量的简化,模型简化到93.7%时,仅用时111.87 ms。然而从上述实验结果中也可以发现,QEM算法只考虑几何误差对网格进行简化,虽然保持了原始模型的总体视觉效果,但无法保持网格的尖锐和细节部分。下一步工作将对QEM算法进行改进,将网格的法向或纹理坐标等属性信息纳入误差考虑范围,对简化后的网格做进一步的细化处理。

参考文献

[1] Hoppe H. Progressive Meshes[C]// Proceedings of SIGGRAPH 96,1996:99-108.

[2] Garland M, P Heckbert. Surface Simplification Using Quadric Error Metrics[C]//Proceedings of SIGGRAPH 97, 1997: 209-216.

[3] Schroeder W. A Topology-Modifying Progressive Decimation Algorithm[C]// Proceedingsof IEEE Visualization '97, 1997: 205-212.

[4] Popovic J, H Hoppe. Progressive Simplicial Complexes[C]// Proceedings of SIGGRAPH97, 1997:217-224.

[5] Hamann B. A Data Reduction Scheme for Triangulated Surfaces[J]. Computer Aided Geometric Design, 1994, 11: 197-214.

[6] Gieng, et al. Constructing Hierarchies for Triangle Meshes [J]. IEEE Transactions on Visualization and Computer Graphics, 1998 4(2):145-161.

[7] Schroeder W. A Topology-Modifying Progressive Decimation Algorithm[C]// Proceedings of IEEE Visualization '97, 1997:205-212.

[8] KleinR, J Kraemer. Multiresolution Representations for SurfaceMeshes[C]// Proceedings of Spring Conference on Computer Graphics 1997, 1997:57-66.

[9] Doerrie H. Euler's Problem of Polygon Division[C]// 100 Great Problems of Elementary Mathematics: Their History and Solutions. NY: Dover, 1965:21-27.

[10] Hinker P, C Hansen. Geometric Optimization[C]//Proceedings of IEEE Visualization '93, 1993:189-195.

[11] Kalvin AD, RH Taylor. Superfaces: PolygonalMesh Simplification with Bounded Error[J]. IEEE Computer Graphics

(下转第68页)

表 1 原图像与增强处理后图像细节评价参数

图像	算法	细节评价参数	归一化形式
cameraman	原图像	0.003 921	1
	HE算法	0.001 049	0.267 5
	文中算法	0.002 485	0.633 8
circuit	原图像	0.001 688	1
	HE算法	0.000 754 9	0.447 2
	文中算法	0.002 304	1.364 9

部分的灰度级过少时，可以大大提高图像的细节评价参数。如图2中应用文中方法处理后circuit图像的细节评价参数显著高于原图像。

4 结 论

针对传统HE后图像由于灰度级减少而丢失细节的问题，提出首先提取图像容易丢失的细节信息，然后再将其与HE处理后的图像相叠加的方法，来达到既增强图像的整体对比度又保留原图像更多细节信息的目的。实验结果表明，该方法达到了这一目的，具有复杂度低、计算量小的优点，并且可以通过调整权值 λ 来获得细节得到不同程度增强的图像，具有一定的适应性。然而文中只是采用了一种比较简单快速的图像细节信息提取方法，对图像细节的提取精度有待提高，因此如何既快速又准确提取图像细节，以及如何对图像增强算法的有效性进行更合理的定量评估是下一步工作的主要内容。

参考文献

[1] 刘刚，王立香，董延. MATLAB 数字图像处理[M]. 北京：机械工业出版社，2010.

[2] 田岩，彭复员. 数字图像处理与分析[M]. 武汉：华中科技大学出版社，2009.

[3] 王群，何永强，周云川. 基于中值滤波和生物仿生学的图像增强研究[J]. 光电技术应用，2011，26(5)：51–55.

[4] 张冰. 基于神经网络的图像增强模型算法[J]. 计算机仿真，2011，28(12)：253–255.

[5] 张静，李一兵，李骜. 基于双树小波变换的图像增强方法[J]. 计算机工程与科学，2011，33(11)：98–102.

[6] 程翔，唐力伟，汪伟. 烟幕干扰条件下目标区域的提取方法[J]. 光电技术应用，2008，23(4)：52–56.

[7] 天河，戴景民. 结合人眼视觉特性的红外图像增强新技术[J]. 红外与激光工程，2008，37(6)：951–954.

[8] Milan Sonka, Vaclav Hlavac, Roger Boyle . 图像处理、分析与机器视觉[M]. 3版.艾海舟，苏延超，兴军亮，等.北京：清华大学出版社，2011.

[9] 李开端，李树军. 基于直方图统计学的图像增强算法研究[J]. 科学技术与工程，2011，11(23)：5572–5575.

[10] 宋岩峰，邵晓鹏，徐军. 基于双平台直方图的红外图像增强算法[J]. 红外与激光工程，2008，37(2)：308–311.

[11] 武治国，王延杰. 一种基于直方图非线性变换的图像对比度增强方法[J]. 光子学报，2010，39(4)：755–758.

[12] 乔闹生. 一种改进的直方图均衡化[J]. 光学技术，2008，34(S)：141–142.

[13] 徐军，梁昌洪，张建奇. 一种红外图像增强的新方法[J]. 西安电子科技大学学报(自然科学版)，2000，27(5)：546–549.

(上接第64页)

and Applications，1996，16(3)：64–77.

[12] Garland M, A Willmott, P S Heckbert. Hierarchical Face Clustering on Polygonal Surfaces[C]//Proceedings of 2001 ACM Symposium on Interactive 3D Graphics，2001：49–58.

[13] Defloriani L, P Magillo, E Puppo. Building and Traversing a Surface at Variable Resolution[C]//Proceedings of IEEE Visualization '97，1997：103–110.

[14] Rossignac J, P Borrel. Multi–Resolution 3D Approximations for Rendering Complex Scenes[C]// Modeling in Computer Graphics: Methods and Applications. Berlin, New York：Springer–Verlag, 1993：455–465.

[15] Luebke D, C Erikson. View–Dependent Simplification of Arbitrary Polygonal Environments[C]//Proceedings of SIGGRAPH 97，1997：199–208.

[16] Ronfard R, J Rossignac. Full–Range Approximation of Triangulated Polyhedral[J]. Computer Graphics Forum，1996，15(3)：67–76, 462.

[17] Cohen J, A Varshney, D Manocha, et al. Simplification Envelopes[C]//Proceedings of SIGGRAPH 96, 1996:119–128.

[18] Rigiroli P, P Campadelli, A Pedotti, et al.Mesh Refinement with Color Attributes[J]. Computers & Graphics, 2001,25(3):449–461.

[19] 王芳.基于视景仿真模型简化算法的研究[D]. 哈尔滨:哈尔滨工程大学，2009.