

基于网格优化的隐式曲面自适应多边形化

王秀友^{1,2}, 汪继文¹, 王浩^{1,2}, 王峰²

(1. 安徽大学 计算机科学与技术学院, 安徽 合肥 230039;

2. 阜阳师范学院 计算机系, 安徽 阜阳 236041)

摘要: 隐式曲面多边形化是隐式曲面绘制的一种常用算法。基于网格优化的隐式曲面快速自适应多边形化算法, 首先用多边形化算法生成一个粗糙的初始网格, 再利用网格优化方法从网格顶点位置、规则性和网格法向三个方面对粗糙网格进行调整, 最后根据网格的局部曲率用多边形细分策略细分优化后的网格。实验结果表明, 该算法在网格生成速度和网格规则性上都胜于 Marching Cubes 的多边形化算法, 恢复的隐式曲面能较好地反映形状特征。

关键词: 隐式曲面; 网格优化; 多边形化; 适应性细分

中图分类号: TP391

文献标识码: A

文章编号: 1673-629X(2008)03-0078-03

Adaptive Polygonization of Implicit Surfaces Based on Mesh Optimization

WANG Xiu-you^{1,2}, WANG Ji-wen¹, WANG Hao^{1,2}, WANG Feng²

(1. School of Computer Science and Technology, Anhui University, Hefei 230039, China;

2. Dept. of Computer, Fuyang Normal College, Fuyang 236041, China)

Abstract: To convert implicit surfaces into polygons is a popular approach in rendering implicit surfaces. Presents a fast adaptive polygonization of implicit surfaces. First get a coarse initial mesh, the mesh is then optimized to improve the quality of the triangles, the refined mesh is finally subdivided using a polyhedral subdivision scheme and a mapping procedure. Experiments show the method produces better results than Marching Cubes polygonizer both in speed and mesh regularity. The presented algorithm is convenient for objects with large smooth and complex surfaces. The method produces a triangular mesh that consists of well-shaped triangles.

Key words: implicit surface; mesh optimization; polygonization; adaptive subdivision

0 引言

近年来, 隐式曲面在计算机图形学领域日益发挥重要作用。隐式曲面由于其光滑拼接的特性, 可以很方便地应用于光滑物体建模和动画领域。当前, 隐式曲面主要有两类绘制方法: 光线跟踪方法和多边形化方法。光线跟踪方法利用定义曲面的隐式方程直接实现隐式曲面的可视化, 因此可以得到高质量的图像, 即使将图形放大若干倍, 也不会产生多边形化所产生的不连续感。但由于在光线跟踪过程中需要大量计算光线与曲面的交点, 加之隐式方程通常较为复杂, 故计算过程非常耗时, 速度缓慢。多边形化方法通过将隐式曲面转化为拓扑一致的网格逼近形式, 来实现曲面

的快速绘制。大多数的图形加速卡和商业动画软件包都支持高性能的、基于多边形的绘制, 因而多边形化方法成为当前隐式曲面快速绘制的主流。

隐式曲面的多边形化需要解决两个相互依存的问题^[1]:

(1) 采样问题, 即如何合理地在隐式曲面上分布采样点;

(2) 连接问题, 即以何种方式将采样点连接成为多边形。

Marching-cubes^[2]是最常用的隐式曲面多边形化技术, 但该技术抽取等值面时会盲目计算大量无关节点的距离值, 算法效率低。Bloomenthal^[3,4]提出使用自适应空间八叉剖分的数据结构来提高多边形化的速度, 其缺点是三角片规则性不好。Hartmann^[5]的扩展三角化方法能生成规则性较好的三角形, 但在扩展区域交界处的规则性不是很好, 且不能处理动态曲面。Karkanis^[6]基于曲率的三角化方法, 既能提供近似等边三角形的网格, 又能根据曲率的大小调整三角片大小,

收稿日期: 2007-06-24

基金项目: 安徽省高校青年教师资助计划项目(2007jql145); 阜阳师范学院自然科学研究项目(2005LQ10)

作者简介: 王秀友(1975-), 男, 安徽宿州人, 讲师, 硕士研究生, 研究方向为算法设计与分析、计算机辅助设计与图形学; 汪继文, 博士后, 教授, 研究方向为计算数学和计算流体力学。

但速度很慢。

文中提出的基于网格优化的隐式曲面快速自适应多边形化算法,首先用多边形化算法对隐式曲面(或三维扫描数据点)生成一个粗糙的初始网格,再利用网格优化方法从网格顶点位置、网格规则性和网格法向三个方面对粗糙网格进行调整,最后根据网格的局部曲率用多边形细分策略细分优化后的网格。

1 隐式曲面快速多边形化算法

1.1 网格调整

在多边形网格的每一个顶点上需要施加三个力的合力:调整顶点位置的力、调整网格规则性的力和调整网格法向的力。该算法可归于 Gauss-Seidel 迭代方法,直到满足终止条件时,才停止迭代。算法伪代码如下:

```

Let  $P_i$  be the vertices of the mesh
Let  $n$  be the number of vertices
While( termination condition is not satisfied ) {
  For( all vertices  $P_i (i = 1, \dots, n)$  ) {
    Step 1: Calculate pseudo-normal  $N_i$ 
  }
  For( all nodes  $P_i (i = 1, \dots, n)$  ) {
    Step 2: Calculate displacement  $\Delta P_i$  caused by the force toward the implicit surface
    Step 3: Calculate displacement  $\Delta P_{u,i}$  caused by the force for regularizing mesh
    Step 4: Calculate displacement  $\Delta P_{c,i}$  caused by the force close to implicit surface normal
    Step 5:  $P_i = P_i + (\Delta P_i + \Delta P_{u,i} + \Delta P_{c,i})$ 
  }
}

```

每次迭代过程中,分别计算调整顶点位置的作用力引起的偏移、调整网格规则性的作用力引起的偏移和调整网格法向的作用力引起的偏移,再更新顶点位置。

1.1.1 调整网格顶点位置

首先讨论调整顶点位置 P 的力 ΔP_i ,该作用力把顶点移动到给定的隐式曲面上。最简单的方法是采用 Othake^[7,8]等沿着梯度方向引入的力:

$$\Delta P_i = -\tau A(P) \nabla f(P_i)^2 = -2\tau A(P) f(P_i) \nabla f(P_i) \quad (1)$$

其中 τ 是一个小的正参数, ∇f 代表梯度向量, $A(P)$ 表示包含顶点 P 的所有三角形的面积之和。由于 $f(x, y, z) = 0$ 是 $w = f(x, y, z)^2$ 的最小水平集合(Minimal Level Set),因此上式定义的力 ΔP_i 能把顶点移动到隐式曲面上。

为了保证迭代过程的数值稳定性,根据对一阶线性偏微分方程稳定性的分析而做的推论, Othake 定义 τ 为:

$$\tau = \frac{1}{100 \max(A \mid f \nabla f)}$$

梯度 ∇f 只有在函数 $w = f(x, y, z)$ 简单的情况下才能解析计算,因此,为了处理更常规的隐式曲面,采用中心差分的方法来估算梯度:

$$\nabla f(p) = (f(P + \Delta x) - f(P - \Delta x), f(P + \Delta y) - f(P - \Delta y), f(P + \Delta z) - f(P - \Delta z)) / 2\Delta$$

其中 Δ 是一个小的偏移量。

1.1.2 调整网格分布

下面讨论调整网格分布的力 $\Delta P_{u,j}$ 。文中使用拉普拉斯光滑算子^[9]的切向分量来提高网格规则性。拉普拉斯光滑算子,也称为伞向量 $U(P)$,迭代地把顶点拉到它的邻顶点的重心。设顶点 Q_i 是顶点 P 的邻顶点,伞向量 $U(P)$ 定义为:

$$U(P) = \frac{1}{n} \sum_{i=1}^n Q_i - P \quad (2)$$

其中 n 是顶点 P 的邻顶点个数。图 1 是伞向量示意图。

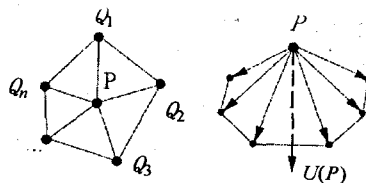


图1 伞向量示意图

如果直接使用拉普拉斯光滑算子,公式(1)里定义的 ΔP_i 力会受到干扰,为此,只使伞向量 $U(P_i)$ 垂直于顶点法向的一个分量,即 $\Delta P_{u,j}$ 力:

$$\Delta P_{u,j} = C[U(P_i) - (U(P_i) \cdot n(P_i))n(P_i)] \quad (3)$$

其中 C 是正常量,实验中取值为 0.1, $n(P)$ 是顶点 P 的单位法向。

现在, ΔP_i 和 $\Delta P_{u,i}$ 互相垂直,两者不会相互干扰,达到了调整顶点位置的同时调整网格规则性的目的。而且,实验结果并不存在标准拉普拉斯光滑算子引起的体积收缩问题。

1.1.3 调整网格顶点法向

调整每个三角网格 T 的顶点位置,使得网格法向 $n(T)$ 和隐式曲面的法向 $m(T) = \nabla f(C) / \|\nabla f(C)\|$ 一致, C 是 T 的重心。调整网格法向的力 $\Delta P_{c,i}$ 定义为^[6,9]:

$$\Delta P_{c,i} = \frac{1}{\sum A(T)} \sum A(T) v(T) \quad (4)$$

这里 $v(T) = [PC \cdot m(T)]$, $m(T)$ 是向量 PC 在 $m(T)$ 方向上的投影。 $A(T)$ 表示三角形 T 的面积。

$\sum A(T)$ 表示顶点 P 附属的所有三角形的面积之和。图 2 给出了几何意义。

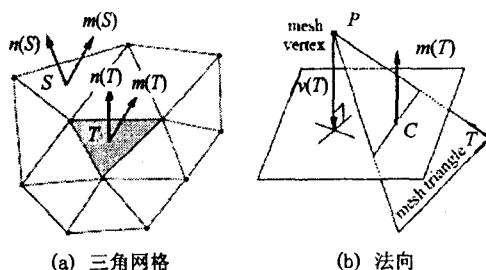


图 2 三角网格和法向通过 $\Delta P_{c,i}$ 调整 P 的位置

1.2 网格细分

隐式曲面快速多边形化算法的第二个步骤是细分优化后的网格。经过网格优化,三角片网格已经比较规则,但为了快速的要求,使用的初始网格比较粗糙,并不足以精确描述隐式曲面,可以利用细分来提高精度。采用三角形边细分策略,能够保持较好的规则性,即在三条边的中点增加新顶点,重新连接新、老顶点,将一个三角形细分为 4 个子三角形。图 3 给出了网格细分方法示意图和细分过程终止条件。

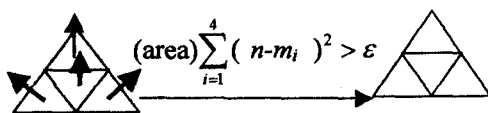


图 3 细分终止条件

由于新增加的顶点不一定位于隐式曲面上,需要把这些点投影到曲面上去。假设一个新增加的顶点 r_0 ,沿梯度方向移动直到位于曲面上,把曲面上这个点作为 r_0 的投影点。这个过程可以用牛顿迭代^[10]的方法实现。 r_0 在隐式表面上的投影点可迭代计算:

$$r_{k+1} = r_k - f(r_k) \frac{\nabla f(r_k)}{\|\nabla f(r_k)\|^2} \quad (5)$$

迭代过程在 $\|r_{k+1} - r_k\|$ 小于给定阈值 ϵ 时终止。图 4 是 2D 情况下迭代过程示意图,2D 情况下,细分策略是在每条边的中点增加一个新的顶点,将该边一分为二,图中 c_1 和 c_2 是曲线上两个顶点, r_0 是边 c_1c_2 的中点,投影过程将 r_0 投影到曲线上点 c 。经过一次细分和迭代投影,折线 c_1c 、 cc_2 比 c_1c_2 能更好地表现曲线形态。

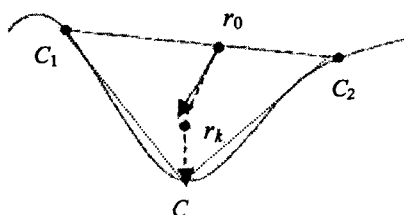


图 4 2D 情况下投影迭代过程示意图

在已构造的初始逼近三角化网格 T 中,其网格法

向 $n(T)$ 与隐式表面法向 $m = \frac{\nabla f}{\|\nabla f\|}$ 有大的偏差。其偏差由以下公式计算:

$$e_n(T) = A(T) \sum_{i=1}^4 (1 - |n(T) \cdot m(c_i)|) \quad (6)$$

其中 $A(T)$ 是初始逼近三角化网格 T 的面积, C_i 表示细分的三角形的面心, $i = 1, 2, \dots, k$ 。

计算出偏差后,再根据其计算值,补偿隐式表面的偏差 k ,从而使计算得到的隐式表面偏差值更接近其真实值。

2 实验结果及讨论

在 Windows 2000 下利用 VC6.0 实现了文中的算法,机器配置为 Pentium IV 2GHz, 256M 内存。图 5 是对球形使用 MC 算法和文中算法的实验结果,可以看出经过网格优化后,球体上的顶点分布较规则,对优化后的网格进行细分能得到更加规则的三角片。图 6 是输入的头模型的三维扫描点利用文中算法重构的实例。

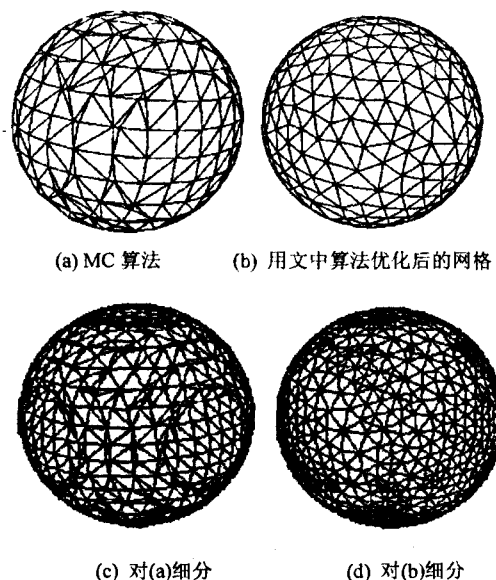


图 5 实验结果

3 结束语

提出了隐式曲面快速多边形算法,由输入的三维扫描数据点得到隐式曲面较为粗糙的三角形表面网格形状,再利用自适应性优化方法对粗糙网格自适应性调整。整个算法从思想上、实现上都很简单。实验结果表明,利用该方法,恢复的网格形状,无论在表面还是在隐式曲面的轮廓部分都能获得很好的视觉效果,并减少了冗余三角形的产生,所有算法的时间复杂度均为 $O(n)$ 。

(下转第 84 页)

中,DSR 协议优于 AODV 协议,但是在较苛刻的环境中则 AODV 协议优于 DSR 协议,并且随着环境变得越来越苛刻(分组速率越快,移动性变得越强),AODV 协议的优势越来越明显。这是因为在高速移动条件下,链路中断可能非常频繁发生,而 DSR 不断地采取路由存储,因此,在路由存储中存在许多过时失效路由,而且没有任何机制用于过时路由期满而删除,或用于确定各条路由的新鲜度以便在面对多条路由选择时做出正确的选择。而 AODV 协议只存储了最新的路由,所以会产生较 DSR 协议低的路由负载和时延,而当节点处于低的移动状态时,路由失效的概率会低一些,因此 DSR 协议可以在路由存储器中多条路由中选择到达目的节点的路由,而 AODV 协议中只存储了最新的一条路由,一旦路由失效,就必须发起路由寻找过程。

分组速率对两种协议的各项性能指标在高速移动条件下和节点较多的情况下影响比较大,而在节点移动速度慢和节点个数较少的情况下影响不明显。

因此,AODV 协议适合工作在移动性强、节点密度大的环境中,而 DSR 协议在这种环境中工作会产生大量的路由负载和较高的时延;DSR 协议适合在移动性较弱、节点密度小的环境中工作,而 AODV 协议在这种环境中会有较低的分组投递率。

3 结束语

文中使用 NS-2 仿真软件对两种常见的按需路由协议 AODV 和 DSR 进行了仿真,通过改变节点的个数、分组速率和节点暂停时间来分析比较这些参数的改变对这两个协议性能的影响,为在实际工作中选择哪种协议以及参数提供了很大的参考。

参考文献:

- [1] 李腊元,李春林. 计算机网络技术[M]. 第 2 版. 北京:国防工业出版社,2004:23-145.
- [2] 郑少仁,王海涛,赵志峰,等. Ad Hoc 网络技术[M]. 北京:人民邮电出版社,2005.
- [3] 吴继春. Ad Hoc 网络路由协议的研究与 NS2 仿真[D]. 武汉:武汉理工大学,2005.
- [4] 刘洛琨,张远,许家栋. AODV 和 DSDV 路由协议性能仿真与比较[J]. 计算机仿真,2006,23(2):118-119.
- [5] Royer E M, Toh C K. A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Network[J]. IEEE Personal Comm.,1999(4):46-55.
- [6] Ramanathan R, Redi J. A Brief Overview of Ad Hoc Networks: Challenges And Directions[J]. IEEE Comm. Magazine, 2002(5):20-22.

(上接第 80 页)

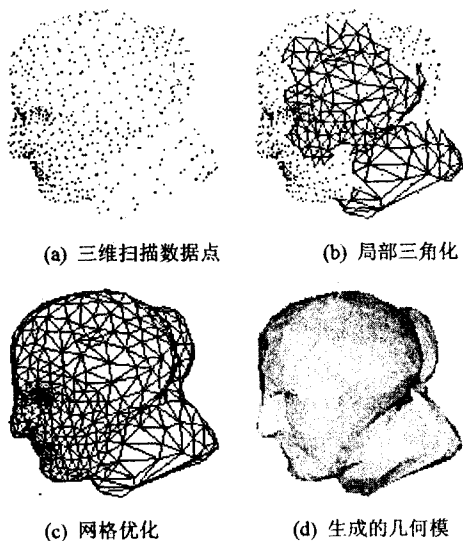


图 6 头模型的重构型

参考文献:

- [1] 庞明勇,卢章平,潘志庚. 隐式曲面的快速适应性多边形算法[J]. 计算机辅助设计与图形学学报,2004,16(11):1511-1516.
- [2] Hilton A, Illingworth J, Windeatt T. Marching Triangles: Range Image Fusion for Complex Object Modelling[C]//In:

Proceedings of IEEE International Conference on Image Processing. Laussane, USA:[s. n.],1996:381-384.

- [3] Bloomenthal J, Heckbert P. An Implicit Surface Polygonizer: Graphics Gems[M]. New York:Academic Press,1994.
- [4] Bloomenthal J. Polygonization of implicit surfaces[J]. Computer Aided Geometric Design,1988, 5(4):341-355.
- [5] Hartmann E. A marching method for the triangulation of surfaces[J]. The Visual Computer,1998,14(3):95-108.
- [6] Karkanis T, Stewart J. Curve dependent triangulation of implicit surfaces[J]. IEEE Computer Graphics and Applications, 2001,21(2):60-69.
- [7] Othake Y, Belyaev A G. Mesh optimization for polygonized isosurfaces[J]. Computer Graphics Forum(Eurographics 2001 issue),2001,20(3):368-376.
- [8] Othake Y, Belyaev A G, Pasko A. Dynamic Mesh Optimization for Polygonized Implicit Surfaces with Sharp Features[J]. The Visual Computer,2003,19(2):74-81.
- [9] Yamada A, Shimada K, Ruruhata T, et al. A discrete Spring Model to Generate Fair Curves and Surfaces[C]//In: Proc. Pacific Graphics'99(IEEE). Seoul, Korea:[s. n.],1999:270-279.
- [10] Jin X G, Sun H Q, Peng Q Sh. Subdivision Interpolating Implicit Surfaces[J]. Computers & Graphics,2003,27(5):763-772.