步进立方体算法的 SOB 数据结构的改进

马仁安 张二华 杨静宇 赵春霞
(南京理工大学计算机系,南京 210094)

摘 要 为了对步进立方体算法进行加速,首先分析了在等值面生成时此算法出现蜕变点的原因 1,并提出了相应的解决方法;然后论述了一种加速步进立方体方法的数据结构——基于层和对象(SOB)的结构,并同八叉树和行程编码方法进行了比较.实验数据表明当数据体中存在一个对象时,SOB 方法和八叉树方法的绘制时间没有大的差别,而当数据体中存在多个对象或等值面时,虽然八叉树方法的绘制时间是 SOB 方法的几倍,但 SOB 方法的存储容量要比八叉树方法大一些.可见,在数据量不是很大的情况下,以差别不大的存储空间来换取较少的等值面生成时间是可行的,其比八叉树方法有较高的效率,尤其在用户需要选择感兴趣的对象进行绘制时,比八叉树和行程编码更具有灵活性.

关键词 计算机图形学(520·6030) 等值面绘制 步进立方体 三角形蜕化 SOB 数据结构
中图法分类号: TP391.41 **文献标识码**: A **文章编号**: 1006-8961(2003)11-1309-05

Improved Marching Cube Algorithm with SOB Data Structure

MA Ren-an, ZHANG Er-hua, YANG Jing-yu, ZHAO Chun-xia
(*Department of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing* 210094)

Abstract This paper discusses the reason of triangle metamorphosis in Marching Cube Algorithm. According to the reason we give a method to protect the metamorphosis. This paper also discusses a new data structure—— Slice-object-based(SOB) structure, which advances the algorithm. The experiments show that SOB method takes the same rendering time when there is only a object in data set. But if there is more than one object, octree algorithm takes few times rendering time than SOB algorithm, so if the data set does not contain so many voxels, it is meaningful that less iso-surface rendering time is taken by the lost of the approximate memory space. That is to say, SOB algorithm excels other methods when the data set contains more than one object, e. g. octree algorithm and run length encoding algorithm, in the executing efficiency. When the data set contains more than one object, octree algorithm only use one octree data structure for all objects, so when we want to render one or several of the objects, this algorithm need to create the octree structure again. SOB algorithm is not so, it creates the SOB structure for each of the objects from the beginning of rendering, so we can render any one of objects selectively according to user's interest, so SOB algorithm is adapt to interactive rendering.

Keywords Isosurface rendering, Marching cube, Triangle metamorphosis, Slice-object-based(SOB) data stucture

0 引 言

步进立方体(Marching Cube:MC)算法是三维数据场等值面生成的经典算法,其基本思路是首先通过逐个处理数据场中的体素立方体来分类出与等值面相交的立方体,然后采用三线性插值来计算等值面与体素立方体边的交点,再根据体素立方体每一顶点与等值面的相对位置,将等值面与体素立方体边的交点按一定的连接方式生成等值面,并将其作为在该体素立方体内的一个逼近表示.其作为一种标准的等值面生成算法,已有许多文献对其作了论述,文献[1~3]对其拓扑二义性问题分别作了论述,并提出了相应的解决方案,但在 MC 方法中还有一个问题,即在确定采样点状态时,将位于等值面外和等值面上的采样点认为是等同的,这就导致一些三角面片蜕化成点或线段,而这些点和线段不但对最终生成的图象没有贡献,却占用一定的计算资

收稿日期:2003-01-06;**改回日期**:2003-06-10

源，为此，本文分析了此种三角形的蜕变原因，并提出了相应的解决方法.

为了加速 MC 算法，不少研究人员从不同角度提出了对其进行加速的方法. 如文献[4]利用了数据体的空间相关性提出了数据体的八叉树表示方法，从而避免了对不含等值面的体素进行测试；文献[5]也提出了类似的行程编码方法，用来分别从不同角度对 MC 方法进行加速. 本文也依据数据体的空间相关性，论述了基于对象和数据层的数据体表示方法——SOB(Slice-object-based)方法，并与上述方法进行了比较. 实验证明，在具有多个对象或多个等值面的体数据中，SOB 方法比八叉树和行程编码方法具有更高的效率，这主要是由于八叉树方法需要通过搜索才能求得体素坐标值，行程编码方法又需要访问完此体素前所有行程才能得到坐标值，而 SOB 方法则能够直接从属性链表中得到体素坐标值.

1 三角面片蜕变的原因与解决方法

确定体素立方体中的等值面分布是该算法的基础，其顶点分类规则[1]为：

(1) 如果立方体顶点的数据值大于等于等值面的值，则定义该顶点位于等值面之外，记为“1”；

(2) 如果立方体顶点的数据值小于等值面的值，则定义该顶点位于等值面之内，记为“0”.

由上面的顶点分类规则知道，原始 MC 算法将位于等值面上的点和位于等值面外的点视为等价的，这就会使某些三角面片蜕变为点或线段，由于这些点或线段对最终的图象不仅没有贡献，反而占用了一定的计算资源，因而，避免点或线段的生成，将有助于提高算法的效率.

在 MC 算法中，一个立方体由 8 个相邻层上的采样点构成(如图 1 所示). 如果体素的两个相邻顶点的状态不同，则等值面肯定与此体素边相交，这些交点坐标可表示为

$$x=x_0+\Delta x\frac{s_{th}-s_0}{s_1-s_{th}},$$

$$y=y_0+\Delta y\frac{s_{th}-s_0}{s_1-s_{th}},$$

$$z=z_0+\Delta z\frac{s_{th}-s_0}{s_1-s_{th}}$$

$\Delta x,\Delta y,\Delta z$ 分别是沿 x,y,z 方向两采样点的距离.

图 2 所示是原始算法的 4 种等值面片的连接情

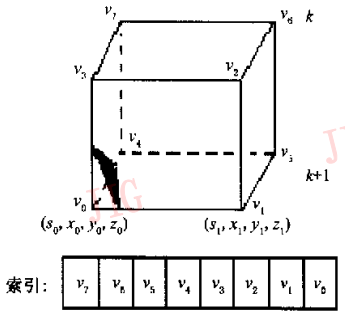


图 1 原始 MC 等值面生成

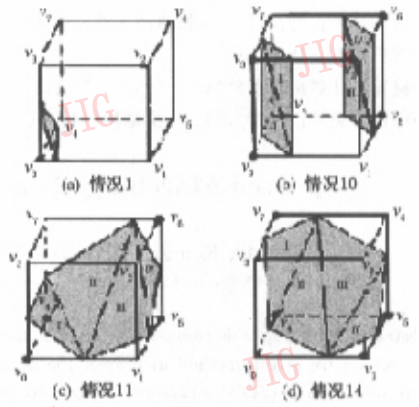


图 2 原始 MC 算法的 4 种情况

况. 对于第 1 种情况(图 2(a))，假定采样点 v_0 值等于等值面阈值 s_{th}，则另外 7 个采样点的值小于 s_{th}，虽然根据原始算法思想，等值面与体素边的交点会构成 1 个三角形，但经过插值之后，这个三角形实际上就蜕变为一个点，即 v_0；对于第 10 种情况(图 2(b))，假定 v_0 的值等于 s_{th}，则三角形 1 就蜕变成为一条线段，但并不是位于等值面上的采样点都会产生三角形的蜕变，如对于情况 11(图 2(c))中的采样点 v_4，其等值面上 4 个三角形都不会发生蜕变；情况 14(图 2(d))中的 v_5 也类似. 像原始算法中的二义性一样，尽管在实际应用中不会频繁出现，但预防这些无用的点和线段的产生是有必要的，在频繁出现这种蜕变点的情况下，如果对这些蜕变点进行适当的处理，那么就能够有效地节约计算时间和提高算法的效率.

只要相邻两个采样点的状态相异，则等值面一定与此条边相交，且一个采样点至多有 3 个相邻的采样点. 根据上面的分析可知，如果 3 个相邻点中有多于一个点的状态与之相异，则就有可能出现蜕变

现象，如果此采样点的值等于等值面阈值，则称这样点的为蜕变点.

如果采样点的值等于阈值 s_{th}，则其相邻点所在位的状态为“1”的个数大于 1，就会产生蜕变. 每个体素的状态索引如图 1 所示. 这种蜕变现象可以通过检测与蜕变点相邻的采样点的状态来加以避免. 并且这种检测只是改变当前体素索引中该蜕变点的状态，而其在相邻体素索引中的状态仍旧维持原来的值. 这样，情况 1 中的 v_0 和情况 10 中的 v_0 状态都将变为“0”，但情况 11 中的 v_4 与情况 14 中的 v_5 仍旧为“1”；而情况 1 在 v_0 为蜕变点时，就变为原始算法中的情况 0，即不与等值面相交；情况 10 在 v_0 为蜕变点的情况下就变为原始算法中的情况 6. 由于相邻体素索引中，此蜕变点所在位的状态未作改变，因而等值面仍经过此点. 据上分析，在生成体素的状态索引时，应加入如下的判定方法：

IF　v_i 的值等于 s_{th}

判断 v_i 相邻点的状态

IF 与 v_i 状态相异的邻点数大于 1

　　将 v_i 在状态索引中的所在位取反.

2　体数据的 SOB 数据结构表示

2.1　SOB 数据结构

在 MC 方法中，文献[4]中作者提出了一种八叉树数据结构，从而跳过其中没有等值面存在的体素，后来，文献[5]又提出了行程编码数据结构来表示与等值面相关的体素，以避免在数据体的不存在等值面区域进行相应的计算，虽然这些数据结构对于表示单一的对象是高效的，在绘制期间，也比较容易得到相应体素的坐标值，但当数据体中有多个物体或多个等值面情况出现时，坐标计算就会占用相当一部分的等值面生成时间，而且对于交互式绘制来说，要想绘制数据体中的某一对象，却无能为力. 为此，本文提出了一种基于层和对象的数据结构(Slice-object-based:SOB)，用以克服八叉树和行程编码数据结构存在的缺点. 此种表示体数据的思想在文献[6～8]中虽已作了论述，但其都是用于改进体数据的直接体绘制.

设数据体为 $\boldsymbol{V}$，$p(x,y,z)$ 是该数据体中的体素，则

$$\boldsymbol{V}=\{p(x,y,z)\mid 1\le x\le X_{max},1\le y\le Y_{max},1\le z\le Z_{max}\} \quad (1)$$

令 f,g 为从体空间到特定数值集的映射，则 $f(p)$ 表示体素 p 的密度值，$g(p)$ 表示体素 p 的显示信息，如颜色和不透明度等. 这样整个体数据可记为 $\boldsymbol{D}=(\boldsymbol{V},f,g)$，对于给定的体数据 $\boldsymbol{D}$，经过二值化后，可得到一个二值化的体数据 $\boldsymbol{D}_b=(\boldsymbol{V},f_b,g_b)$，其中，b 是二值化函数，$f_b$ 定义如下

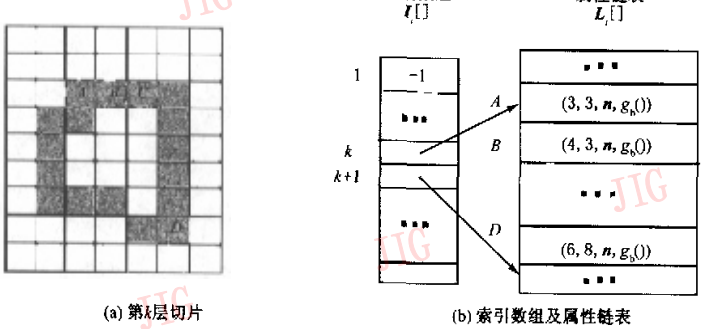
$$f_b(p)=\begin{cases}1 & t_1\le b(p)\le t_2\\ 0 & \text{其他}\end{cases} \quad (2)$$

二值化后，体空间 $\boldsymbol{V}$ 分成了两个体素集：密度值为 1 的体素集和密度值为 0 的体素集. 前者包含了目标体素，后者在绘制时，可以忽略. 在密度值为 1 的体素集中，一个离散的等值面[8]可定义为

$$\boldsymbol{S}=\{p\in\boldsymbol{V}\mid f_b(p)=1\wedge(\exists\, q\in\boldsymbol{N}_{26}((p),f_b(q)=0)\} \quad (3)$$

这里，$\boldsymbol{N}_{26}(p)$ 表示与体素 p 二十六连通的相邻体素集，而 $\boldsymbol{S}$ 是构成某一对象表面的体素的最小集合.

为了有效地存储和绘制由体素组成的等值面片，本文提出了 SOB 数据结构. SOB 由两类结构组成：一个是大小为 Z_{max} 的索引数据集 $\boldsymbol{I}_i$，一个是属性链表 $\boldsymbol{L}_i$[8]，其中保存的属性向量为 $(x,y,\boldsymbol{n},g_b(b))$，$i$ 是对象代码(如图 3 所示). 这样，在进行绘制时，等



(a) 第k层切片　　(b) 索引数组及属性链表

图 3　SOB 结构示意图

值面体素是以切片顺序存储在 L_i 中，在切片中，体素是以 y 值的大小依次存储的，I_i 中的每个索引表示在这个切片中，对象 i 的等值面体素在 L_i 中存储的起始位置，正如图 3(a)所示，切片 k 中等值面体素从 A 到 D 按 y 值大小依次存储，I_i 中的第 k 个值指示体素 A 的属性值在 L_i 中的存储位置，如果一个切片不包括任何对象体素，则将其相应的索引值置为 -1，因而在生成三角面片时，遇到 -1 时就跳过该切片. 链表 L_i 中存储的是体素的投影和光照属性参数，$(x,y,\boldsymbol{n},g_b(b))$中，$x,y$ 是体素在切片中坐标值，$\boldsymbol{n}$ 是体素的法向量，$g_b(p)$是相应体素的可视属性值.

2.2 SOB 与其他方法的比较

由于八叉树和行程编码都是利用体素间的空间相关性来定义的数据结构，且只是仅对等值面上的体素进行编码，因而需要进行必要的计算才能求出相应的等值面体素的三维坐标值，例如，对于行程编码方法，由于需要遍历位于当前体素前的所有行程，才能得到此体素的坐标，而在八叉树情况下，又由于需要从根节点遍历到此体素的相应叶节点才能得到坐标值，因而增加了等值面的生成时间，而 SOB 则可以从 I_i 和 L_i 中直接得到体素的坐标值.

另外，对于交互式绘制来说，如果用户只想绘制数据体中的某一对象，那么对八叉树和行程编码方法来说，只能在生成此结构时，加入相应的对象类属标识，但在生成等值面时，却要搜索整个结构，以得到所选对象的所有等值面体素. 对于这种情况，SOB 方法就灵活得多，因为每一对象对应一 SOB 结构，这样，对象的添加或删除，就无需对整个数据结构进行更新，而只要加入一个 SOB 结构就可以了，而八叉树和行程编码则要对整个数据结构进行更新，所以在多个对象或等值面时，SOB 方法可节约计算时间.

SOB 方法是以层和对象为基本的结构单位来对数据体进行编码的. 对于那些没有等值面体素的层来说，则需要在索引数组中给予标识，这就在一定程度上浪费了存储空间. 另外，随着对象数的增加，SOB 方法需要从整个数据体出发，以层为单位来对新增对象进行编码，这样相应的存储量自然就会增加，而八叉树和行程编码方法则只需对体素所属对象加上一标识即可，其存储空间没有明显的增加，实验结果证明，运用 SOB 方法，对于非海量数据或对象数较少的体数据，存储量的增加是不大的，基本上不影响整个算法的效率. 目前，由于微机内存容量的增大是比较廉价的，因而 SOB 方法以存储空间来换取等值面生成时间是可行的.

3 实验结果分析与结论

为了说明 SOB 数据结构的有效性，在 MC 方法中，采用了 SOB 与八叉树方法进行了实验. 实验比较结果如表 1 所示. 实验是在一台 P Ⅲ 667，256MB 的微机上进行的.

表 1 SOB 方法与八叉树方法的比较结果

数据集	执行时间(s)		存储量(MB)	
	SOB	Octree	SOB	Octree
1 个球	33	34	1.54	1.23
4 个球	34	52	1.56	1.22
16 个球	36	173	1.62	1.30
64 个球	41	405	1.83	1.47

实验中使用 4 个具有不同半径的球体数据，即半径为 80pixels 的球，半径为 40pixels 的 4 个球，半径为 20pixels 的 16 个球和半径为 10pixels 的 64 个球. 实验用数据体的大小均为 128 × 128 × 128pixels. 其等值面生成图如图 4 所示. 若球的半径减半，则将导致表面积减小四分之一，但由于球数增大了 4 倍，因而这 4 个数据体的等值面体素数基本相同，另外，由于等值面的表面积基本相同，因而等值面生成时间也应当基本相同，故表 1 中的执行时间差别只是反应了不同数据结构遍历时间的差异，这也就反应了各种数据结构的有效性.

对于 SOB 方法来说，由表 1 可以看出，对不同数据体的处理时间差别不大，由上面的分析知道，由

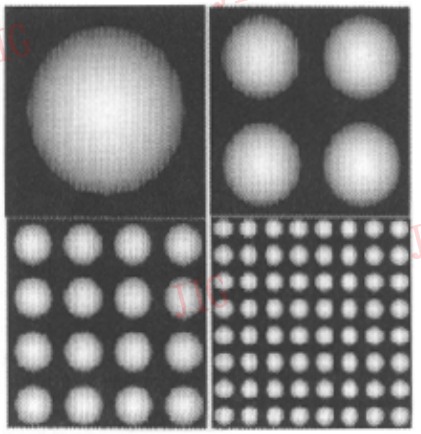


图 4 不同半径的球体数据绘制结果

于这些差别只是用于遍历索引数组，因而引入索引数组对整个等值面的生成时间影响不大.

由表 1 知，八叉树结构所需的存储空间只是 SOB 结构所需空间的 0.8 倍，这主要是由于八叉树结构无需记录体素的坐标所致. 不过八叉树方法却需要更多的等值面生成时间，实验证明，用八叉树方法和 SOB 方法处理一个对象时，处理时间没有大的区别，但随着对象数的增加，由于求取坐标时间不断累积，因而八叉树方法总体需要更多的时间，由表 1 可见，16 个球的数据体的处理时间，八叉树方法是 SOB 方法的 4 倍；64 个球时，数据体的处理时间；八叉树方法是 SOB 方法的 10 倍.

由此可以看出，在数据量不是很大的情况下，以差别不大的存储空间来换取较少的等值面生成时间是切实可行的，这也反映了 SOB 方法的有效性，因此，可加速 MC 方法的实现.

图 5 是一个人体头部图象，数据规模为：128×128×128pixels，其等值面三角面片生成时间和存储空间如表 2 所示，表 2 数据更进一步验证了 SOB 方法要优于八叉树方法. 由于蜕变点数比较少，对最终的执行时间和图象质量没多大影响，因而，没有将对蜕变点进行绘制和不作绘制的 MC 方法作比较.

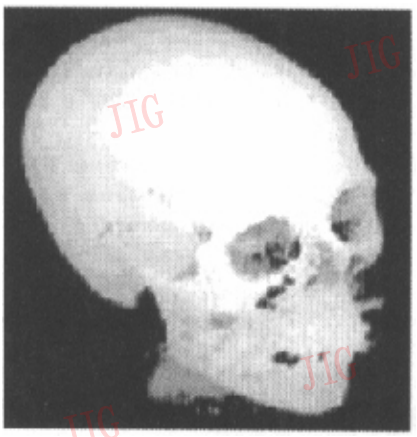


图 5　人体头部数据等值面绘制结果

表 2　人体头部数据实验结果

	蜕变点数	执行时间(s)		存储量(MB)	
		SOB	Octree	SOB	Octree
头部数据	145	347	556	2.44	1.83

参 考 文 献

1　Nielson G M, Hamann B. The asymptotic decider: resolving the ambiguity in marching cubes[A]. In: Proceedings of Conference on Visualization[C], San Diego, CA, USA, 1991, 83～91.

2　Wilhelms J, Gelder A V. Topological considerations in isosurface generation. Extended Abstract [J]. Computer Graphics, 1990, 24(4): 79～86.

3　周勇、唐泽圣. 用自适应的三线性逼近方法构造等值面[J]. 计算机学报，1994，17(增刊)：1～10.

4　Meagher D J. Geometric modeling using octree encoding [J]. Computer Graphics & Image Processing, 1982, 19(2): 129～147.

5　Philippe Lacroute, Marc Levoy Fast volume rendering using a shear-warp factorization of the viewing transformation [J]. Computer Graphics, 1994, 28(4): 451～458.

6　Udupa J K, Odhner D. Fast visualization, manipulation, and analysis of binary volumetric objects [J]. IEEE Computer Graphics and Application, 1991, 11(6): 53～62.

7　Udupa J K, Odhner D. Shell rendering [J]. IEEE Computer Graphics and Application, 1993, 13(6): 58～62.

8　Kim Bo Hyoung, Seo Jinwook, Shin Yeong Gil. Binary volume rendering using slice-based binary shell [J]. the Visual Computer, 2001, 17: 243～257.

马仁安　1974 年生，1997 年获解放军炮兵学院学士学位，2000 年获解放军炮兵学院硕士学位，现在南京理工大学计算机系攻读博士学位. 主要研究方向为计算机图形学、虚拟现实和三维数据场可视化.

张二华　1967 年生，南京理工大学计算机系副教授，硕士生导师. 主要研究方向为地震信号处理与可视化.

杨静宇　1941 年生，南京理工大学计算机系教授，博士生导师. 主要研究方向为机器人与模式识别.

赵春霞　1964 年生，南京理工大学计算机系教授，硕士生导师. 主要研究方向为计算机辅助设计与数字图象处理.