

文章编号: 1007- 2985(2010) 01- 0073- 05

# Marching Cubes 算法中等值面的优化分析\*

李 改

(顺德职业技术学院计算机技术系, 广东 顺德 528300)

**摘 要:** 三维重建中等值面的优化操作可以减少等值面中三角片的数目或改变三角片的组织、存储结构, 并尽可能地保持等值面原来的拓扑结构和几何形状. 运用中点选择法和三角带生成算法, 优化了等值面中三角片的组织、存储结构, 从而使得三维模型的绘制速度有很大提高.

**关键词:** Marching Cubes; 等值面; 中点选择; 三角带生成; 三维重建

中图分类号: TP751

文献标识码: A

在三维重建中, 传统的等值面的优化方法是直接提取出三角片数据, 然后用 OpenGL 对三角片数据直接进行绘制. 本文运用中点选择避免线性插值及三角带生成算法, 在很大程度上优化了三角片的组织、存储结构, 更利于运用 OpenGL 来进行快速三维显示, 从而加快了三维模型的绘制速度.

医学图像的三维重建方法主要有三维体绘制<sup>[1]</sup>和三维面绘制<sup>[2-3]</sup>. 三维体绘制将数据场中的体元看成一种半透明物质, 并赋予其一定的颜色和光度, 由光线穿过整个数据场, 进行颜色合成. 三维面绘制是表示三维物质形状最基本的方法, 它可以提供三维物体的全面信息. 在三维面绘制中, 基于等值面的表面曲面表示方法最为常用, 就是直接从三维体数据生成等值面, 目前已有多种方法, 其最具代表性的是 Marching Cubes (MC) 算法. 对于标准 MC 算法, 其本身还有很多需要改进和完善的地方. 比如在三维面绘制中, 一般使用三角片作为最基本的元素, 但随着应用需求的发展, 等值面变得越来越复杂, 三角片的数量迅速增长. 虽然利用提高硬件性能的办法可以在一定程度上使问题得到改善, 但不是最终的解决办法. 使用中低档工作站甚至是微机来进行三维成像处理成为近年来热门的研究课题, 解决问题的折衷办法就是进行等值面的优化. 等值面优化的目标是减少等值面中三角片的数目或改变三角片的组织、存储结构, 并尽可能保持等值面原来的拓扑结构和几何形状. 为了达到优化等值面的目的, 本文运用中点选择避免线性插值, 即等值边上的中点代替等值点, 从而简化线性插值计算量, 并且可以使同平面相邻体素的多边形合并成一个多边形, 减少多边形数量, 有利于后续的三角带生成. 然后运用三角带生成算法, 对等值面提取出来的三角片数据进行处理以生成三角带, 优化三角片的组织、存储结构, 从而有效地利用存储空间、提高等值面的显示速度.

## 1 MC 算法的基本原理

传统的 MC 算法是 W. Lorensen 等人<sup>[4-5]</sup>于 1987 年提出的一种三维表面重建方法, 其算法是面显示算法中的一种, 因为它的本质是从一个三维的数据场中抽取出一个等值面, 所以也被称为“等值面提取”(Isosurface Extraction) 算法.

\* 收稿日期: 2009- 11- 10

作者简介: 李 改 (1981- ), 男, 湖北荆州人, 顺德职业技术学院计算机技术系讲师, 硕士, 主要从事三维医学图像建模和计算机图形学研究.

传统的 MC 算法的基本思想是把三维图像相邻层上的各 4 个像素组成立方体的 8 个顶点, 逐个处理三维图像中的立方体, 分类出与等值面相交的立方体, 采用插值计算出等值面与立方体边的交点. 根据立方体每一顶点与等值面的相对位置, 将等值面与立方体的边的交点按一定方式连接生成等值面, 作为等值面在该立方体内的一个逼近表示. 立方体每个顶点有 2 种状态(大于或小于等于等值面的值), 因此共有 256 种组合状态. 根据互补对称性和旋转对称性可将其简化成 15 种构型, 如图 1 所示<sup>[5]</sup>.

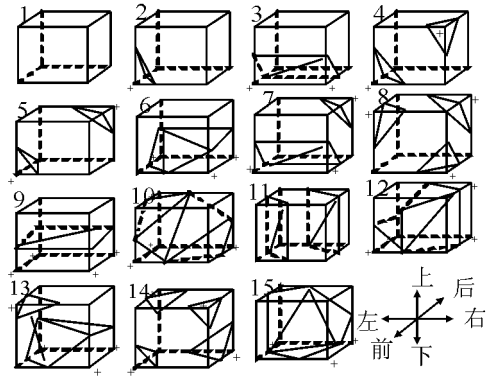


图 1 六面体体素的 15 种标准构形

传统的 MC 算法采用了隐式的等值面提取方法, 它不直接计算  $f(x, y, z)$ , 而是直接从体数据中获取等值面的信息. 算法需要用户提供一个域值, 也就是所希望提取出来的物质密度值, 比如要提取出骨骼, 域值就要相对大一些, 然后根据体数据的信息, 就可以提取出等值面的三角网格表达.

要想用真实感图形学技术将等值面显示出来, 除了要知道每个等值点的坐标外, 还必须知道每个等值点的法向量. 在传统的 MC 算法中, 线性插值公式如下:

$$\begin{cases} P = P_1 + (\text{isovalue} - V_1)(P_2 - P_1)/(V_2 - V_1), \\ N = N_1 + (\text{isovalue} - V_1)(N_2 - N_1)/(V_2 - V_1), \end{cases} \quad (1)$$

其中:  $P$  代表等值点坐标;  $P_1, P_2$  代表 2 个端点的坐标;  $V_1, V_2$  代表 2 个端点的灰度值;  $\text{isovalue}$  代表阈值;  $N$  代表等值点法向量;  $N_1, N_2$  代表 2 个端点的法向量.

传统的 MC 算法的实现相当直接, 并且得到的图像解析度很高. 但是传统的 MC 算法存在着下面几点缺陷: 拓扑不一致, 计算效率低, 绘制速度慢. 其中绘制速度慢的问题主要是因为 Marching Cubes 算法提取出的等值面中包含大量无序的三角片, 使得运用 OpenGL 来绘制等值面的速度比较慢. 本文主要就如何解决绘制速度慢的问题来进行研究.

## 2 用中点选择避免线性插值

在传统的 MC 算法中, 一个包含等值面的立方体最多会生成 4 个三角片. 每个三角片都很小, 有的甚至比重构图像中的像素点还小. 因此, 立方体内等值面三角片顶点位置的轻微变动对最后图像生成质量影响很小. 本文提出用中点选择避免线性插值, 也就是对于一个三角片的顶点不像标准 MC 算法中通过线性插值计算得到, 而是用该顶点所在边的中点代替. 采用中点为三角片顶点避免线性插值计算交点, 减少了计算量, 也更有利于后续的三角带生成. 在标准的 MC 算法中, 每次线性插值需要 5 次代数计算. 即使 (1) 式中系数  $(\text{isovalue} - V_1)/(V_2 - V_1)$  不变, 也需 9 次计算才能得到三角片顶点的位置. 而应用中点选择, 确定顶点位置只需 3 次计算, 由此可见, 生成一个三角片能减少 18 次计算. 对于大多数应用来说, 中点算法能节约大量的计算时间<sup>[6]</sup>. 中点选择方法如下:

$$P = (P_1 + P_2)/2, N = (N_1 + N_2)/2, \quad (2)$$

其中:  $P$  代表等值点坐标;  $P_1, P_2$  代表 2 个端点的坐标;  $N_1, N_2$  代表 2 个端点的法向量;  $N$  代表等值点法向量.

中点选择所具有的优点是<sup>[7]</sup>: (1) 引起的误差低于 1/2 立方体边长, 这在医学图像的解析度越来越高的情况下, 所重建出来的图像与线性插值得到的图像并没有明显的视觉上的差异<sup>[8]</sup>;

(2) 如果先放大 10 倍再进行运算, 就可以完全采用整数运算, 避免浮点运算; (3) 可以减少运算量, 生成一个三角片能减少 18 次计算; (4) 使得局部表面更平坦, 如图 2 所示. 可以使立方体中的三角片数量减少, 且使一个立方体内或者相邻立方体之间连接的三角片合并成一个多边形, 有利于在需要时对三角片的组织、存储进行优化, 如生成三角带.

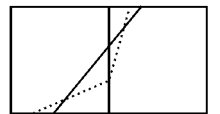


图 2 局部平坦化

### 3 三角带的生成

通过 MC 算法可以得到等值面的三角网格表达. 因为有 3 个顶点才能表达一个三角面片, 所以要想表达具有  $n$  个三角面片的网格, 就必须有  $3 \times n$  个顶点. 而如果使用三角带的话, 因为三角面片之间共享了顶点, 所以可以仅使用  $n+2$  个顶点就能表达  $n$  个三角面片. 这样可以大量地节省内存和显示卡内存之间传输的带宽, 极大地提高显示速度<sup>[7]</sup>.

为了更快地绘制生成的网格, 这里采用了一种快速的三角带生成算法, 对提取出来的等值面生成更紧凑的三角带表达. 在等值面的提取中, 本文采用了传统的顶点表和面表的表达方式. 顶点表 VertexTable 记录了所有的顶点坐标和法向量, 它的表达形式如下所示:  $\{x, y, z; nx, ny, nz\}$ . 其中:  $x, y, z$  是顶点的 3 个坐标;  $nx, ny, nz$  是顶点在 3 个方向上的法向量. 而面表 FacetTable 与传统的表达方式有所不同, 它只记录了一层中的等值面的连接方式, 它的表达形式如下所示:  $\{\text{index1}, \text{index2}, \text{index3}\}$ . 其中:  $\text{index1}, \text{index2}, \text{index3}$  分别代表一个三角片的 3 个顶点在表 VertexTable 中的位置索引. 在这里顶点表 VertexTable 中没有重复的顶点, 并且在面表 FacetTable 中记录的是三角片的 3 个顶点在顶点表 VertexTable 中的索引值, 这样算法可以仅仅在面表中进行操作. 算法的描述如下:

(1) 算法由面表创建一个邻接图. 邻接图实际上就是一个大的哈希表, 它记录了每个三角片在每条边上的邻居信息. 要创建邻接图, 在面表 FacetTable 中查找所有共享一条边的三角面片, 并将它们插入到邻接图中.

(2) 从邻接图中找到具有最小度数(邻居数)的三角片. 如果有好几个三角片都具有同样的度数, 那么再查找它们邻居的度数, 选中邻居度数最小的那个三角片. 如果它们的邻居的度数仍然一样的话, 可以随便选中一个.

(3) 从选中的这个三角片开始, 循环生成三角带. 将被选中的三角片加入到三角带中, 并且在它的邻居中查找具有最小度数的三角片, 将其加到三角带中, 一直选择循环下去, 直到所有的邻居都被访问过为止. 如果碰到一个三角片, 它没有任何邻居, 那么此时应该创建一个新的三角带. 如果碰上有几个相同度数的三角片时, 处理方法与上面所介绍的相同. 为了避免无限循环, 在将一个三角片加入到三角带之后, 需要在邻接图中将它和所有指向它的指针全部去掉.

(4) 检查生成的三角带是否有效地组织了所有的顶点, 是否是有效的三角带, 这时要对一个三角带中的所有三角片进行检测, 看它的最后 2 个顶点是否是下一个三角片的前 2 个顶点. 实现三角带生成所需的数据结构如图 3 所示.

图中  $T_1 \dots T_n$  是经过表面重建后提取出的三角片在三角片数组  $T_A$  中的序号,  $V_1, V_2, V_3$  是三角片中 3 个顶点在顶点表 VertexTable 中的序号. 图 3 中的  $T_m, T_n$  是与三角片  $T_1$  共享由  $V_1, V_2$  2 顶点构成的边的三角片在三角片数组  $T_A$  中的序号,  $T_m', T_n'$  是与三角片  $T_n$  共享由  $V_1', V_3'$  2 顶点构成的边的三角片在三角片数组  $T_A$  中的序号. 采用该数据结构可以实现三角带生成算法中所需的邻接图.

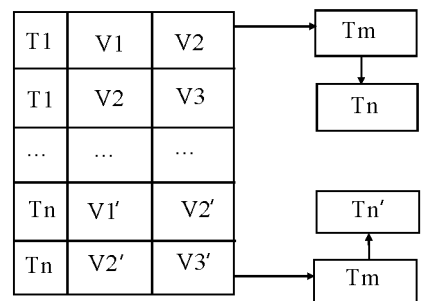


图 3 邻接图数据结构示意图

三角带生成算法的伪代码如下:

输入: 三角面片数组  $T_A$

输出: 三角带

Triangle ( )

Begin:

    遍历三角面片数组  $T_A$  得到共享公共边的三角片邻接

    图 Adj;

    While( 邻接图 Adj 非空)

Begin:

    创建一个三角带;

    查找邻接图中具有最小度数的三角片序号  $T_{\min}$ , 记其

度数为  $n$ ;

    if(具有度数为  $n$  的三角片只有一个

        将三角片  $T_{\min}$  加入到当前三角带中, 并将其从邻接图中删除;

    else

        Begin:

            在所有度数为  $n$  的三角片中找出邻居度数( $m$ ) 最小的三角片;

            if(邻居度数为  $m$  的三角片只有一个)

                将该三角片加入当前三角带, 并将其从邻接图中

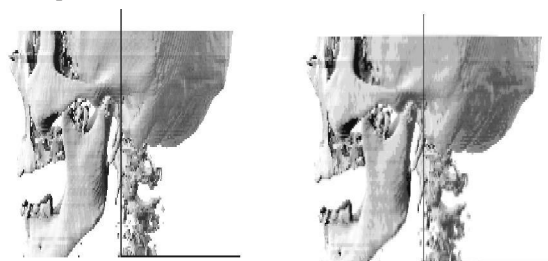
```

删除;
else
  随便取一个邻居度数为 m 的三角片加入当前三角
  带,并将其从邻接图中删除;
End;
While(当前加入三角带的三角片有邻居)
  Begin:
  从当前加入三角带的三角片的邻居中查找具有最小度数的
  三角片,序号为 T min1,记其度数为 n1;
  if(具有度数为 n1 的三角片只有一个)
    将三角片 T min1 加入到当前三角带中,并将其从
    邻接图中删除;
  else
    Begin:
    在所有度数为 n1 的三角片中找出邻居度数(m1)
    最小的三角片;
    if(邻居度数为 m1 的三角片只有一个)
      将该三角片加入当前三角带,并将其从邻接图中
      删除;
    else
      随便取一个邻居度数为 m1 的三角片加入当前三
      角带,并将其从邻接图中删除;
    End;
  End;
End;
End;
检查所有生成的三角带的有效性;
End;

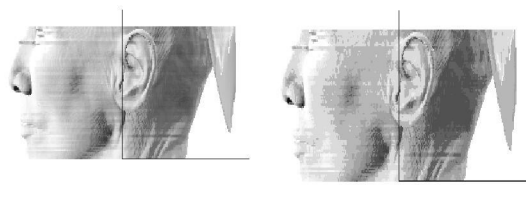
```

## 4 实验结果比较

图 4, 5 是重建出来的骨骼, 皮肤模型进行等值面优化前后的三维显示效果图. 在这里使用的三维显示工具是 OpenGL 开发包<sup>[9-10]</sup>.



a 优化前 b 优化后  
图 4 骨骼模型效果图



a 优化前 b 优化后  
图 5 皮肤模型效果图

运用传统的 MC 算法重建出的等值面优化前后的实验结果比较如表 1 所示. 实现本算法的微机的配置: CPU(Intel Celeron(TM) 1 000 MHz); 内存(256 MB); 显卡(NVIDIA GeForce2 MX/MX 400); 操作系统(Windows XP).

表 2 MC 算法等值面优化前后的实验结果比较(512×512×67 DICOM 格式的头部 CT 数据)

组织	实验	三角片个数	绘制时间/s
骨头	优化前	549 593	1.2
	优化后	549 593	0.5
皮肤	优化前	983 392	2.1
	优化后	983 392	0.9

从以上实验结果可以看出: 优化前后 MC 算法所产生的三角片数量相同, 优化没有改变三角片的数量, 但改变了三角片的组织和排列方式, 使之更利于运用 OpenGL 来进行三维绘制, 绘制速度显著提高. 因为采用了中点选择避免线性插值, 所以在某些细节上可能会造成一些图像损失, 但是由于大多数三角片会小于像素, 所以误差不会太大, 图像视觉上不会有太大影响.

## 5 结语

在三维表面重建中, 由于 OpenGL 强大的功能及三角片在 OpenGL 中的高效性, 通常采用三角网格来表示三维表面. 运用传统的 MC 算法可以得到一个拓扑关系正确、几何正确的三角形网格, 但这时等值面中的三角片是无序存储的, 不利于运用 OpenGL 来对其进行三维显示. 通过本文的方法可以对这些等值面中的三角片数组进行优化, 生成一条条共享公共边的三角带, 这在很大程度上优化了三角片的组织、存储结构, 更利于运用 OpenGL 来进行快速三维显示, 从而加快了三维模型的绘制速度.

## 参考文献:

- [1] MAX N. Optical Models for Direct Volume Rendering [J]. IEEE Transaction on Visualization and Computer Graphics, 1995, 1(2): 99- 108.
- [2] 管伟光. 体视化技术及其应用 [M]. 北京: 电子工业出版社, 1998.
- [3] DOI A, KOIDE A. An Efficient Method of Triangulating Equi-Valued Surface by Using Tetrahedral Cells [J]. IEICE Transactions, 1991, E74(1): 214- 222.
- [4] 何曙光, 田 捷. 基于分割的三维医学图像表面重建算法 [J]. 软件学报, 2002, 13: 219- 226.
- [5] LOPES A, BRODLIE K. Improving the Robustness and Accuracy of the Marching Cubes Algorithm for Isosurfacing [J]. Visualization and Computer Graphics, 2003, 9: 16- 29.
- [6] 徐 毅, 李晓梅. 对体可视化 Marching Cube 算法的改进 [J]. 计算机工程, 1999, 11, 25(11): 52- 54.
- [7] 田 捷. 医学影像处理与分析 [M]. 北京: 电子工业出版社, 2003.
- [8] MONTANI C, SCATENI R, SCOPIGNO R. Discretized Marching Cubes [C]// BERGERON R D, KAUFMAN A E, eds. Proceedings of the Visualization' 94, 1994: 281- 287.
- [9] 白建烟, 朱亚军, 梁 辉, 等. OpenGL 三维图形设计与制作 [M]. 北京: 人民邮电出版社, 1999.
- [10] 杨淑莹, 卢刚等. 用 OpenGL 实现 CT 图像三维重建系统的设计 [J]. 天津师范大学学报: 自然科学版, 2001, 9: 46 - 49.

## Optimization Research of Isosurface in Marching Cubes Algorithm

LI Gai

(Department Computer Technology, Shunde Polytechnic, Shunde 528300, Guangdong China)

**Abstract:** The optimization operation of isosurface is very common in the three-dimension reconstruction. In this paper, the author uses middle point choice to replace lineal insert-value and the method of making a strip of triangle to optimize the form and the storage structure of the triangle in the isosurface, thus improving drawing speed of three-dimension models greatly.

**Key words:** marching cubes; isosurface; middle point choice; strip making of triangle; three-dimension reconstruction

(责任编辑 陈炳权)

(上接第 69 页)

## A New Ultimate Strength Calculating Method for Arbitrary Cross Section Under Bi-eccentric Loading with Meshless Method

YANG Yu-dong<sup>1,2</sup>, ZHANG Ke<sup>2</sup>, WU Min-zhe<sup>2</sup>

(1. School of Divil Engineering and Architecture, Anyang Normal University, Anyang 455002, Henan China;

2. School of Civil Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China)

**Abstract:** In order to analyze the cross-section bearing capacity of bi-eccentric compressive short-limbed shear wall, the authors designed the analyzing program of bi-eccentric cross-section bearing capacity. Based on the interpolation principle. The program calculates the inward pressure of concrete fully according to the accurate solution of integral region obtained by mathematical equation, so as to avoid the errors of mesh generation. The program results agree well with the experimental results. This method provides a new idea to develop similar program. By using that program, the interrelated curved surface N-M of bi-eccentric compression of short-limbed shear wall is obtained, and it is found that the equiaxed force line is similar to special-shaped column. The program can fully grasp the cross-section bearing capacity of bi-eccentric compressive short-limbed shear wall, and thus it can meet the requirements of engineering design.

**key words:** short-limbed shear wall; computer program; the cross-section bearing capacity

(责任编辑 陈炳权)