

基于微机环境的三维数据场 多等值面快速显示算法

宛 铭 唐泽圣

(清华大学计算机科学与技术系 北京 100084)

摘要 直接体绘制技术能够利用半透明效果显示三维数据场,提供了比等值面绘制方法更为丰富的信息,但是,由于数据场中所有体素都参与了图象生成过程,使得该技术的计算开销昂贵,远远无法达到交互式操作的要求。事实上,如果用边界表示法来表示三维数据场,就可以利用三维空间连续性来大幅度缩短绘制时间。边界表示法只关心有等值面穿过的边界体元,用 $O(kn^2)$ 内存单元来表示 $O(n^3)$ 大小的原始数据场,从而产生大规模数据压缩。本文在此基础上提出一种基于微机环境的显示算法,生成图象效果近似于直接体绘制技术,而速度至少提高一个数量级。文章还给出了用新算法绘制的三维医学图象,并与 ray casting 算法的结果在图象质量和绘制时间上做了比较。

关键词 体可视化,不透明度,等值面,数据压缩技术,医学图象。

三维数据场绘制方法分为2大类:等值面绘制方法^[1,2]和直接体绘制方法。^[3]前者需要构造物质面片,然后用传统的图形绘制技术来显示。后者直接绘制三维数据场,无需先拟合几何元素。直接体绘制方法又可分为 ray casting^[4,5]和 direct projection^[6~8],前者从屏幕的每个象素发出一根视线,并在其穿越数据场的过程中进行光亮度积分;后者则将数据场的每个体元投影到图象平面上。直接体绘制方法的关键性进步在于采用了半透明技术,从而能够显示出非常丰富的信息,甚至连数据场中细微的特征都不会丢失,其原因在于数据场的所有体元都对结果图象产生影响。但是它的致命缺陷也正源于此,大量的计算引起的等待时间令用户无法接受。尤其在微机环境下,虽然可以产生高质量的画面,但绘制一幅图所耗的漫长时间(常常是几十分钟)使之无法实用化。

本文提出一种基于微机环境的显示算法,其速度远超过已发表的直接体绘制方法,而图象质量的牺牲并不显著。该算法借助于一种称为边界表示法的数据压缩方法,利用三维空间连续性来大幅度缩短绘图时间。事实上,这种新算法几乎囊括了上述2大类绘制算法的主要优点。首先,它绘制的具有 shading 效果的图象能体现出数据场所包含的多种半透明或不透明的物质,因而同样能提供比传统的等值面绘制法^[2]更为丰富的信息。其次,它只利用了3

* 作者宛铭,1967年生,博士生,主要研究领域为计算机图形学,科学计算可视化。唐泽圣,1932年生,教授,博士生导师,主要研究领域为计算机图形学,科学计算可视化,计算机辅助设计技术,计算几何。

本文通讯联系人:宛铭,北京100084,清华大学计算机科学与技术系

本文1996-03-04收到修改稿

维数据场中与物质界面有关的体元信息,而忽略了其它部分,因而其计算开销大为降低.再者,它还利用了 early ray termination 策略以节省绘制时间,该策略原本是 ray casting 算法^[5]独有的优化方法.此外,新算法在预处理阶段将含有物质边界的体元存放起来,以便将来从不同视角绘图时直接调用,无需重新计算,这在直接体绘制方法中是无法做到的.虽然实际节省的绘制时间依赖于具体数据场,但实验结果表明,对于达到实用大小和复杂程度的 CT 数据场,新算法的绘制速度比传统的直接体绘制方法要快一个数量级以上.

本文处理的体数据是一组典型的定义在三维直线型网格点上的标量值,如医学图象领域中一系列连续的二维 CT 或 MR 切片.数据场中的每个体元是相邻 2 个切片上的 8 个顶点所围成的立方体中的区域,体元中标量值(也称密度值)函数是变化的.边界表示法只保留有物质边界面穿过的体元(称为边界体元)中心的位置坐标和梯度值.因此只需用 $O(kn^2)$ 个内存单元来表示 $O(n^3)$ 大小的数据场,从而形成高度数据压缩.由于三维数据场的分辨率已经很高,我们将采用点元作为中间过渡的几何元素,使算法的空间和时间效率进一步提高,以适应微机环境.

周勇^[9]曾提出一种改进的 projection 方法,利用一种类似于 sticks 表示法^[10]来保留边界体元的索引.由于所用的中间几何元素是面元而非点元,该方法更适合于稀疏数据场.但是由于在绘制阶段,该方法仍离不开原始数据场,而且在数据表示中还用到了双向指针链,使数据访问时间增大,因而只比传统的体绘制速度提高 3~4 倍.此外,该方法没有给出各种特殊情况的处理方法,如出现不封闭区域的情形,而这些现象在实际数据中经常出现.绘制阶段的光亮度积分也未考虑图象的 shading 效果,否则绘图时间还会更长.所有这些问题在我们的新算法中都将妥善解决.

本文第 1 节给出边界表示法及其空间复杂度;第 2 节给出基于边界表示法的快速显示算法;实用 CT 数据的实验结果将在第 3 节给出;结束语和将来的工作在第 4 节.

1 体数据的边界表示法

为了高效率地表示三维数据场,减少绘图时间,本文提出一种边界表示法,只记录数据场中与边界体元有关的信息.该数据模型在算法的预处理阶段生成,需对已分类的数据场进行边界检测.

1.1 数据分类

在应用领域中,三维数据场往往看作多种物质的组合,如医学领域的骨骼、肌肉和皮肤.给不同的物质赋予不同的颜色和不透明度,数据场中的综合信息就能有效地显示出来.本文采用的数据分类方法是阈值值技术.给定 4 个独立的分类转换函数 $R(S(X,Y,Z))$, $G(S(X,Y,Z))$, $B(S(X,Y,Z))$ 和 $O(S(X,Y,Z))$,将不同范围的标量值映射到特定的颜色和不透明度,以确定不同的物质. $S(X,Y,Z)$ 是数据场中位于 (X,Y,Z) 处的现成的或插值得到的密度值.例如,对于三维医学 CT 数据,我们用 3 个不同的 CT 值 C_1 , C_2 和 C_3 作为阈值来区分皮肤、肌肉和骨骼(见图 1).若标量值 $S(X,Y,Z)$ 落在区间 $[C_1, C_2)$, 则该处有不透明度为 8% 的黄色皮肤;若属于区间 $[C_2, C_3)$, 则为不透明度为 12% 的红色肌肉;若属于 $[C_3, S_{max}]$, 则为完全不透明的白色骨骼. 本文将用 $i (1 \leq i \leq M)$ 来表示密度值下限为 C_i 的物质

类型及阈值为 C_i 的(物质)边界类型, 这里 M 是物质类数. 其他形式的转换函数参见文献[3].

1.2 边界检测

一旦给定了物质边界的阈值, 就可以通过边界检测将原始三维数据场转换为边界表示形式, 包括 1 张线性边界体元表 L 和 1 张二维统计表 T (见图 2). 表 L 包含所有边界体元的信息, 如物质类型 id ($1 \leq id \leq M$), 体元位置 (i, j, k) 和体元中心梯度向量 (nx, ny, nz) . 每个体元的 XYZ 坐标 (i, j, k) 分别为它在切片中的行号和列号以及该切片号. 表 T 记录了具有相同的 Y 和 Z 坐标的同一行体元中边界体元的个数.

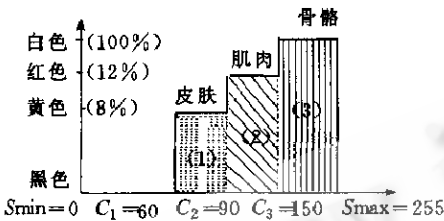


图1 物质分类转换函数示例

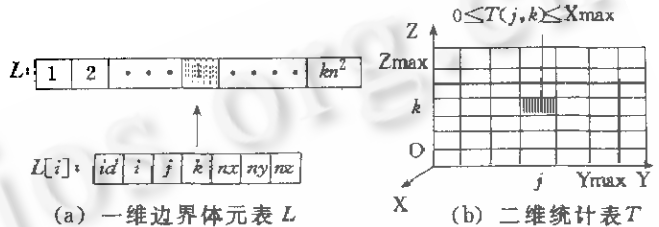


图2 边界表示法数据结构示意图

边界检测过程如下: 首先建立表 L 和表 T , 并将 T 中所有元素初使化为 0. 然后在三维数据场中逐片逐行地检测边界体元, 即找出某些但非全部 8 个顶点的密度值小于某个给定的阈值 C_{id} 的体元. 对于每个边界体元, 在表 L 中记下它的物质类型 id 和位置 (i, j, k) . 特殊情况下, 如果有多个物质边界同时穿过该体元, 则只记下物质类型最大的一个 (理由将在后文中给出). 同时计算 8 个顶点的梯度值, 并插值出体元中心处的梯度值, 作为边界面法向量的估计, 也记到表 L 中. 最后将统计表 T 中 (j, k) 处元素的值加 1.

为了产生更精细的显示结果, 我们可以把边界体元细分成更小的体元. 当数据场各维上的分辨率不一致时 (常见于三维医学数据^[11]), 这种细分则不可避免. 显然, 小体元中心的位置和梯度值可以从边界体元的 8 个顶点的值插值得到. 但是在上述边界检测过程中加入体元细分操作时一定要特别小心, 必须保持逐片逐行顺序增大的顺序, 以便绘图时用.

类似的边界检测和体元细分策略也曾在显示单个不透明物质的 dividing cubes 算法中使用过^[2], 但是, 对多种半透明物质的处理及二维统计表 T 的使用尚未发现.

1.3 空间复杂性

边界表示法空间复杂性 (C) 可以用三维数据场在 YZ 平面上平行投影的面积以及每行上边界体元个数的平均值 k 来估算:

$$C(\text{Boundary-Rep}) = O(kn^2) \cdot C(\text{List-Element}) + O(n^2) \cdot C(\text{Table-Element}) \quad (1)$$

从图 2 我们看到表 L 的每个元素由 7 个域组成, 且均为整型数. 事实上, 利用表 T , 我们能容易地推算出表 L 中各元素的位置三元组 (i, j, k) 中的 j 和 k 的值. 这样只有 5 个域是必须的. 另外, 表 T 的每个元素是一个整型数. 因此, 公式 (1) 可以写成:

$$C(\text{Boundary-Rep}) = [O(5kn^2) + O(n^2)] \cdot \text{sizeof}(\text{int}) \quad (2)$$

由此可见, 边界表示法可以用 $O(kn^2)$ 个边界元素来替代 $O(n^3)$ 大小的原始数据场. 显然, 只有当 k 值很小时, 该表示法效果才好. 可喜的是实际情况确实如此. 例如, 对于一个大小为 $128 \times 128 \times 197$ 的实际树獭 CT 数据, 每行的边界体元平均个数 k 为 3.44, 明显低于数据场在 X 轴向上的长度 128. 而且整个数据场也压缩至原先的 23.08%.

2 快速显示算法

这里给出的快速显示算法的处理对象是用边界体元集合表示的数据场,并在前文所述的预处理阶段由原始数据场生成,因而该算法只处理原始数据场中与物质边界有关的信息,并对包围在边界内的物质的光亮度作出估计.这里要用到2个buffer,即RGBA buffer和face buffer,二者初值均为零.RGBA buffer记录图象平面中每个象素上累积的光亮度RGB值和不透明度值;face buffer存放图象平面中每个象素上最新投影的边界体元的信息,包括有关的边界类型及其距图象平面的距离.

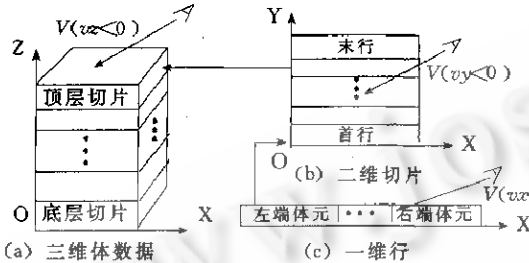


图3 三维数据场中体元排列方式示意图

该算法首先建立一个体元处理顺序,以保证表L中的每个边界体元以视线方向 $V(V_x, V_y, V_z)$ 为基准,按深度优先顺序处理.也就是说,如果体元A遮挡体元B,则B一定在A之后处理.利用表L中体元排列的特点,即逐片自底向上,片中逐行从前向后,行中逐个自左向右的排列顺序(见图3),我们给出一种简单而高效的访问表L中体元的顺序:如果轴Z与视线方向V的夹角大于 $\pi/2$ (即从顶向下看),则低层的切片将被高层的遮挡,因此各切片的访问顺序应该是从最上层切片至最底层切片,否则应保持表L中原先的自底向上的顺序.一片及一行内的体元的处理方式也类似(见图3(b)和图3(c)).表L中每个切片及片中每行的起始体元的位置可以根据统计表T中的信息直接计算出来.

一旦表L中体元的扫描顺序确定下来,就可以对体元逐个处理.为了便于把它们投影到图象平面上,首先将其在原始三维数据场中的坐标 (i, j, k) 转换为图象空间中的坐标 (x, y, z) ,图象空间坐标系的XZ平面与图象平面平行,而Y轴与图象平面垂直.然后将体元投影到图象平面的 (x, z) 处.假设B是当前待处理的体元,且将投影到图象平面的 (x, z) 处:A是face buffer中记录的在B之前最新投影到图象平面 (x, z) 处的体元.算法将分3步执行.第1步,如果A存在,即face buffer的 (x, z) 处不为空,并且A与B恰巧分别被某种物质的前后边界穿过,则根据后文给出的公式(4)计算A,B之间所含物质的累积光亮度和不透明度,然后把计算结果按从前向后的顺序迭加到RGBA buffer中.^[12]第2步,如果穿过B的边界面是可见的,则根据其法向 (n_x, n_y, n_z) 用光照模型进行shading计算,并将得到的光亮度及相应物质的不透明度迭加到RGBA buffer中.第3步,把B的物质类型id及其深度y存放到face buffer中.执行上述3步操作的基本前提是RGBA buffer中 (x, z) 处的不透明度低于100%,或用户给定的某个门槛值.请注意,A~B这段物质的累积光亮度和不透明度的计算在上述操作中被分为2部分,首先对A中包含的前边界面作光亮度的shading计算,然后从A之后向B在该物质内部作积分运算(不含A).这样做的好处主要是节省了存放每个边界体元的法向量所需的内存空间.有关积分和shading操作的细节将在下文给出.

2.1 深度积分及shading操作

沿视线方向计算一对前后边界面之间的某段物质的累积光亮度 C_{now} 及不透明度 α_{now} 的过程称为深度积分.本文采用离散方式将该物质段的光亮度按深度顺序一步步累加起来.

假设 A 和 B 是一对边界体元, 分别包含同一种物质的前后边界面。 α_{mat} 和 C_{mat} 分别为该物质本身的不透明度和光亮度; 可以从前面的物质分类转换函数得到。 d 是 A 和 B 的深度差(见图 4)。从 A 到 B 对该物质段光亮度 C_{now} 和不透明度 α_{now} 进行计算的积分公式如下:

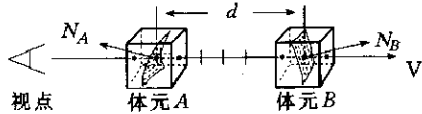


图 4 同一视线方向上 2 个相邻边界体元配对图

$$C_{now} \cdot \alpha_{now} = C_{mat} \cdot \alpha_{mat} \cdot \sum_{i=0}^d (1 - \alpha_{mat})^i \quad \alpha_{now} = \alpha_{mat} \cdot \sum_{i=0}^d (1 - \alpha_{mat})^i \quad (3)$$

为了提高显示效果并增强物质边界, 我们对每个可见的体元用 Phong 模型作 shading 计算, 要用到边界面法向量和光源参数。这样, 从 A 到 B 计算 C_{now} 和 α_{now} 的积分过程自然地分为 2 步: 第 1 步, 把体元 A 中心处的梯度向量作为 A 中所含的物质前边界面的法向量, 利用 Phong 模型计算出 A 中前边界面的光亮度, 相应的不透明度就是 α_{mat} ; 第 2 步, 计算从体元 A 后方直到体元 B 之间这段物质累积的光亮度 C'_{now} 和不透明度 α'_{now} , 计算公式可由公式(3)推出, 这里只需考虑环境光成分, 并假设环境光系数为 k_a :

$$C'_{now} \cdot \alpha'_{now} = k_a \cdot C_{mat} \cdot \alpha_{mat} \cdot \sum_{i=0}^{d-1} (1 - \alpha_{mat})^i \quad \alpha'_{now} = \alpha_{mat} \cdot \sum_{i=0}^{d-1} (1 - \alpha_{mat})^i \quad (4)$$

上述 2 步运算的结果应该按运算的先后顺序迭加到 RGBA buffer 中, 参看文献[4]。

实验结果表明, 用 shading 操作来增强物质边界的效果非常明显。Levoy 通过降低物质内部的不透明度, 同时加强物质边界面的不透明度来达到同样目的。与之相比, 我们的策略更加方便, 计算量也小。

2.2 边界体元配对

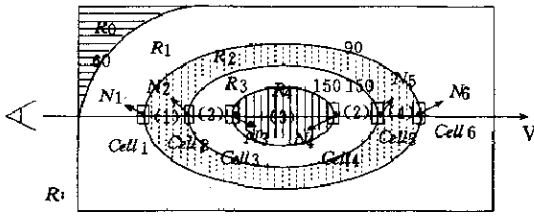
正确判断同一根视线前后 2 个相邻边界体元的关系是合理使用积分和 shading 操作的前提。仍然假设 A 和 B 分别为同一视线方向上前后 2 个边界体元, N_a 和 N_b 为各自的梯度向量, D_a 和 D_b 为它们所含边界面的类型(见图 4)。我们暂时假设每个边界体元只被一种物质边界穿过。相对于视线方向 V 来说, 梯度向量 N 有 3 种情况, 即 N 和 V 的夹角 θ 大于、等于或小于 $\pi/2$ 。第 1 种情况下, 体元被一段物质的前边界面穿过, 是可见的; 最后 1 种情况下, 体元包含物质段的后边界面, 故不可见; 中间情况下, 边界面与视线相切, 它对光亮度的贡献可以忽略不计。此外, 边界类型 D_a 和 D_b 的关系也有 3 种, 大于、小于或相等。这样我们共得到 12 种情况组合(见表 1), 其中 θ_a 是 N_a 与 V 的夹角, θ_b 是 N_b 与 V 的夹角。

表 1 相邻边界体元情况组合表

	$\theta_a > \pi/2, \theta_b > \pi/2$	$\theta_a > \pi/2, \theta_b < \pi/2$	$\theta_a < \pi/2, \theta_b > \pi/2$	$\theta_a < \pi/2, \theta_b < \pi/2$
$D_a = D_b$	—	②	③	—
$D_a > D_b$	—	—	—	④
$D_a < D_b$	①	—	—	—

事实上, 只有表中用数码标出的 4 种情况下, A, B 之间存在物质段需要用到积分运算。图 5 给出三维数据场沿视线方向 V 的一个横截面 R 。 R 被 3 种等值面分为 5 个子区, 3 种等值面的密度门槛值分别为 $C_1 = 60, C_2 = 90, C_3 = 150$ 。图中 6 个边界体元(用方框表示)中心梯度向量 $N_i (1 \leq i \leq 6)$ 是从高密度区指向低密度区。这些体元内的边界类型分别为 $D_1 = D_6 = 2, D_2 = D_3 = D_4 = D_5 = 3$ 。体元之间夹的数码与表 1 中 4 种需要积分运算的情况相对应。

沿视线方向从 A 到 B 作积分运算时, A, B 间所夹的物质的类型 D 应取 D_a 和 D_b 中较



$R_0 = \{P | 0 \leq S(P) < 60\}$, $R_1 = \{P | 60 \leq S(P) < 90\}$,
 $R_2 = \{P | 90 \leq S(P) < 150\}$, $R_3 = \{P | 150 \leq S(P) < 255\}$,
 $R_4 = \{P | S(P) < 150\}$, 其中 P 是截面 R 上任意一点, $S(P)$ 是点 P 的密度值

图5 边界体元配对示意图

说明 A, B 间不存在任何物质. 然而在实际情况下, 三维数据场中往往出现多种边界面同时穿过某一体元的情况. 这时情况③中 A, B 间物质类型 D 可以是任一小于 D_a 和 D_b 且大于 0 的值. 同时, 表 1 的第 3 列中的另外 2 种情况也要进行积分运算, 因为任何类型值小于 D_a 和 D_b 的物质都有可能在 A, B 之间出现. 解决多等值面同时出现这一特殊情况的最直接的方法是在表 L 中为每个体元增加 1 个域来记录穿过该体元的类型值最小的边界面. 但是通过分析实验数据, 我们发现同时被多个等值面穿过, 且每个等值面的阈值 D_i 均大于 1 (以保证 $D_i - 1 > 0$) 的体元的数目与整个数据场边界体元总数相比, 只占很小一部分 (从树獭 CT 数据测得结果是 0.66%). 因此为简便起见, 我们仍采用只在表 L 中记录每个边界体元所含边界面的最大阈值的策略.

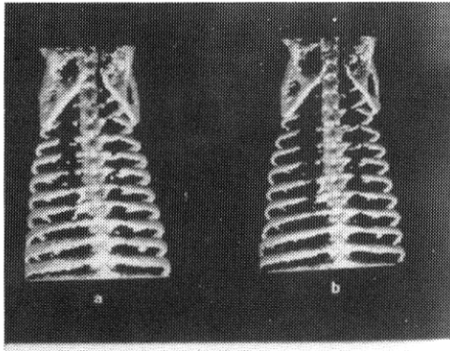
3 实验结果

新算法已在 PC486 上用 C 语言实现. 所用的 CT 数据是 197 片树獭切片, 每片分辨率为 128×128 . 为了便于定量比较, 我们在同样的机器上实现了 ray casting 算法^[4], 沿视线方向的重采样步距为 0.8 (假设体元边长为 1). 图 6(a) 中不透明的骨骼图象用新算法生成, 绘制时间小于 6s (见表 2). 这里测得的时间不包括生成表 L 和表 T 的预处理时间, 因为对于不同的视线方向, 它们无需重新计算. 光照模型使用 2 个白色光源, 对于提高图象视觉效果, 尤其是边界效果很有帮助. 图 6(b) 显示了 ray casting 图象, 绘制时间约为 256s. 2 幅图几乎一模一样, 可如果你细看一下, 会发现 ray casting 方法生成的骨骼要细一些, 原因在于受到重采样步距的限制, 该方法会遗漏骨骼边界附近的部分边界体元. 与 ray casting 方法相比, 我们的新算法速度要快得多, 很有希望达到交互显示速度.

表 2 数据场显示时间对照表

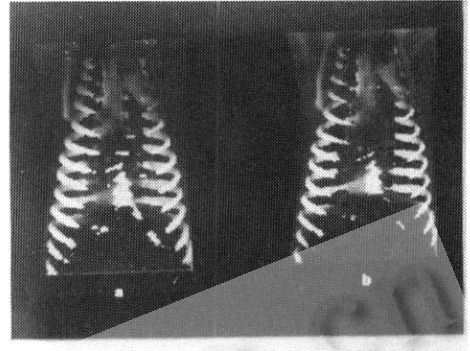
数据分类	新算法	ray casting	加速比
骨骼	5.934s	255.648s	43.082
骨骼+肌肉+皮肤	77.417s	833.021s	10.760

图 7(a) 显示了用新算法从相反的视线方向绘制的多物质综合图象, 根据图 1 中给出的物质分类函数, 我们将 3 个相邻的密度值范围分别映射到黄色的皮肤、红色的肌肉和内脏以及白色的骨骼, 并赋予相应的不透明度. 图 7(b) 中的图象是由 ray casting 方法采用相同的物质分类函数及视线方向绘制的. 2 幅图象选用的用户定义的不透明度阈值均为 95%. 通过比较, 我们认为新算法生成的图象效果是令人满意的. 然而其绘制时间也只用了不到 80s, 仍比 ray casting 方法快 10 倍以上, 后者耗时超过 800s.



(a) 新算法绘制,耗时不到 6s;
(b) 由 ray casting 方法生成,耗时近 256s

图 6 三维白色不透明骨骼图象



(a) 新算法绘制,耗时不到 80s;
(b) 由 ray casting 方法生成,耗时超过 800s

图 7 三维数据场内多种半透明物质综合图象

4 结论及今后的工作

本文提出了一种新的基于边界表示法的三维数据场显示算法,它能快速生成三维数据场所包含的多种半透明和不透明物质的综合图象.该算法利用物质分类转换函数将不同密度值范围与具有不同光亮度值和不透明度的物质对应起来.再根据由视线方向确定的体元处理顺序将在预处理阶段检测出的边界体元逐个投影到图象平面上.然后用简化的离散型积分模型沿视线方向对前后边界面之间的物质段(若存在的话)的累积的光亮度和不透明度进行计算.最后将计算结果按从前向后的顺序迭加到结果图象上.该算法还增加了可见面的 shading 计算,以进一步提高图象尤其是边界的视觉效果.

目前,新算法提高结果图象的分辨率的方法是在边界体元中进行密度值及梯度值的线性插值.但是,我们在实验中发现,这种灰度级插值方法在采样点较为稀疏的情况下将产生阶梯状混淆现象,而这种情况在临床的医学 CT 数据场中很常见,因为医学 CT 切片间距往往比切片内象素尺寸大得多.^[11,13]提出一种更精确的基于形状的插值方法,可以估算出丢失的切片上的物质边界.而边界信息正是我们的新算法所关心的,因此可以将二者结合起来,使新算法绘出更精致的图象,以适应医学临床应用的要求.

虽然新算法已比体绘制算法快很多,但离交互式显示的要求还有一定距离,在绘制具有半透明效果的多物质图象时矛盾就更加突出.幸而,作为一种基于物体空间的显示算法,它能方便地利用物体空间分解策略在最常见的非共享内存的 MIMD 系统上实现并行.另外,根据预处理阶段生成的表 L 和表 T,我们可以预先了解整个计算量,并把它们精确地分配到每个处理器上.这样,作为目前并行处理的 3 大主要难点之一的负载平衡问题^[14],就很圆满地解决了.我们现在正在研究高效率的图象合成策略及如何保持较好的可扩展性,以实现高效率的并行算法,最终达到交互显示速度.

参考文献

- 1 Lorenson W E, Cline H E. Marching cubes: a high resolution 3D surface construction algorithm. *Computer Graphics*, 1987, **21**(4):163~169.
- 2 Cline H E, Lorenson W E, Ludke S *et al.* Two algorithms for the three-dimensional reconstruction of tomo-

- grams. *Medical Physics*, 1988, **15**(3): 320~327.
- 3 Upson C, Keeler M. V-BUFFER; visible volume rendering. *Computer Graphics*, 1988, **22**(4): 59~64.
 - 4 Levoy M. Display of surfaces from volume data. *IEEE CG&A*, 1988, **8**(3): 29~37.
 - 5 Levoy M. Efficient ray tracing of volume data. *ACM Transactions on Graphics*, 1990, **9**(3): 245~261.
 - 6 Drebin R A, Carpenter L, Hanrahan P. Volume rendering. *Computer Graphics*, 1988, **22**(4): 65~74.
 - 7 Westover L. Footprint evaluation for volume rendering. *Computer Graphics*, 1990, **24**(4): 367~376.
 - 8 Wilhelms J, Gelder A V. A coherent projection approach for direct volume rendering. *Computer Graphics*, 1991, **25**(4): 275~284.
 - 9 Zhou Y, Tang Z. Volume partitioning: a coherent subvolume projection approach for volume rendering. *Pacific Graphics'94, Beijing*, 1994. 325~340.
 - 10 Montani C, Scopigno R. Rendering volumetric data using the sticks representation scheme. *Computer Graphics*, 1990, **24**(5): 87~93.
 - 11 Rara S, Udupa J. Shaped-based interpolation of multidimensional objects. *IEEE Transactions on Medical Imaging*, 1990, **9**(1): 32~42.
 - 12 Porter T, Duff T. Compositing digital images. *Computer Graphics*, 1984, **18**(3): 253~260.
 - 13 Herman G T, Zheng J, Bucholtz C A. Shaped-based interpolation. *IEEE CG&A*, 1992, **12**(3): 69~79.
 - 14 Whitman S, Hansen C D, Crockett T W. Recent developments in parallel rendering. *IEEE CG&A*, 1994, **14**(4): 21~22.

MICRO-COMPUTER BASED MULTI-ISOSURFACES RENDERING ALGORITHM FOR 3D DATA SETS

Wan Ming Tang Zesheng

(Department of Computer Science and Technology Tsinghua University Beijing 100084)

Abstract Direct volume rendering techniques use semi-transparency to visualize much more important features in the three-dimensional scalar sample volumes than isosurface rendering methods. However, since all voxels participate in the generation of each image, these techniques are computationally expensive and far from interactive manipulation. In fact, on the base of the boundary representation scheme for the volumetric data, the 3D spatial coherence can be fully exploited to drastically reduce the rendering time for arbitrary viewing orientation. This data model only considers the bounding cells pierced by the isosurfaces of specified materials; therefore, it requires $O(kn^2)$ memory cells to represent $O(n^3)$ volumetric data, and produces a high degree of data compression. A PC-based rendering algorithm in use of the boundary scheme is proposed in this paper, which produces results visually similar to the volume rendering techniques, but speedup at least an order of magnitude. Examples from medical CT studies are given in contrast to the results from ray casting method both in image quality and in processing time.

Key words Volume visualization, opacity, isosurface, compression techniques, medical image.