第 42 卷 第 12 期 西 安 交 通 大 学 学 报 Vol. 42 №12
2008 年 12 月 JOURNAL OF XI′AN JIAOTONG UNIVERSITY Dec. 2008

一种八叉树编码加速的 3D 纹理体绘制算法

邹华，高新波，吕新荣
（西安电子科技大学电子工程学院，710071，西安）

摘要：针对后分类 3D 纹理体绘制算法在数据量较大或片元着色程序较复杂时绘制速度较慢的问题，提出了一种基于八叉树编码的加速算法. 首先设置八叉树的高度，并根据空间位置对体数据逐层剖分，然后构建八叉树，用八叉树结点来保存生成子块的相关信息，最后遍历八叉树来实现空间跳跃，减少了在体数据内采样生成的片元数量，从而缩短了 GPU 对片元进行着色、测试、混合等处理的时间. 实验结果表明，该算法可应用于不同片元程序的纹理体绘制中，绘制速度均在 10 帧/s 以上，与自适应分块加速算法相比，该算法能获得平均 3.7 的高加速比.

关键词：3D 纹理体绘制；八叉树；空间跳跃；后分类
中图分类号：TP391.41 **文献标志码**：A **文章编号**：0253-987X(2008)12-1490-05

An Accelerating Algorithm for 3D Texture Volume Rendering with Octree Encoding

ZOU Hua， GAO Xinbo， LÜ Xinrong
（School of Electronic Engineering，Xidian University，Xi′an 710071，China）

Abstract：In order to accelerate the 3D texture volume rendering algorithm with post-classification，whose speed is somewhat slow when volume dataset is large or fragment shaders are complex，an octree encoding algorithm is proposed. The octree encoding algorithm is implemented in following steps. The octree level is set at first，and the volume data is subdivided level by level based on spatial positions. Then，the octree is constructed by storing the information of sub-blocks in octree nodes. Finally，the information in nodes is loaded for judging whether the corresponding sub-blocks should be rendered or skipped when the volume data is rendered by traversing the octree. The experimental results show that the method can be applied to 3D texture volume rendering by using different fragment programs with the rendering speed of 10 frames per second or higher and that the average speed ratio is 3.8 which is higher than that of the adaptive partitioning algorithm.

Keywords：3D texture volume rendering；octree；empty space skipping；post-classification

随着 GPU(图形处理单元)的出现，利用 GPU 提高绘制速度已成为体绘制算法[1-5]的发展趋势，但当数据量较大或片元着色程序较复杂时，会影响 GPU 的处理，导致速度下降. 八叉树编码[6]、虚拟八叉树模型[7]、层次细节[8]、自适应分块[9]等空间跳跃算法能减少 GPU 的运算量以提高绘制速度，但目前这些算法都需要进行较复杂的预处理. 对此，本文提出一种新的八叉树编码算法以加速后分类 3D 纹理体绘制. 首先设置八叉树高度，按空间位置对体数据剖分，然后构建八叉树来保存子块坐标、最大最小值等信息，最后通过遍历八叉树实现空间跳跃，缩短 GPU 的处理时间，达到提高绘制速度的效果.

收稿日期：2008-01-04. 作者简介：邹华(1980—)，男，博士生；高新波(联系人)，男，教授，博士生导师. 基金项目：国家重点基础研究发展规划资助项目(2006CB705700)；国家自然科学基金资助项目(60771068)；教育部长江学者和创新团队支持计划资助项目(IRT0645).

1　后分类 3D 纹理体绘制算法

后分类 3D 纹理体绘制算法[10]直接在原始体数据中采样获取体素值,再根据传输函数计算颜色和透明度值. 与先分类算法[2]相比,它需要的 Nyquist 采样频率较低[11],而且当传输函数改变时不需要重新加载 3D 纹理,但受当时硬件限制,该算法难以实现复杂光照模型.

本文用 Cg 语言编写片元着色程序来实现后分类 3D 纹理体绘制,体数据保存在 3D 纹理的 Alpha 通道中,为减少片元着色程序的复杂度,预先计算梯度保存于 RGB 通道中,再利用 Cg 语言实现 RGBA 绘制、最大密度投影、表面、Phong 光照等多种绘制效果.

(1)RGBA 绘制. 根据传输函数生成 1D 纹理查找表. 片元着色器拾取纹理值后,在查找表中查找颜色和阻光度对片元赋值,再从后向前混合片元,生成最终图像.

(2)最大密度投影绘制(MIP). 根据在体数据中采样遇到最大体素值确定最终图像颜色.

(3)Phong 光照绘制. Phong 光照综合考虑了表面方向 $\boldsymbol{N}$、光线方向 $\boldsymbol{L}$、视线方向 $\boldsymbol{V}$、光线与视线的半角向量 $\boldsymbol{H}$ 及环境系数 k_a、漫反射系数 k_d 和镜面反射系数 k_s 等对体素颜色 I 的影响,能取得较真实的效果,加入 Phong 光照渲染后体素的最终颜色为

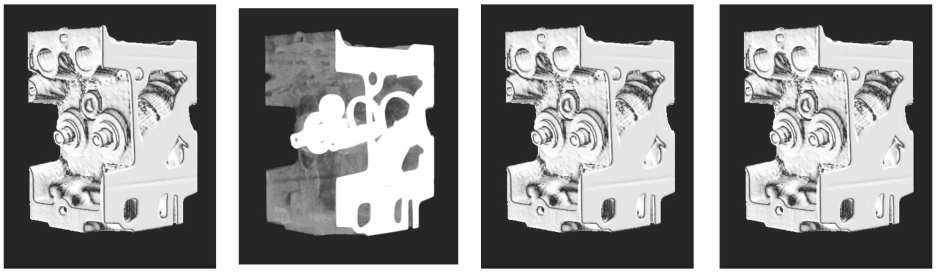
$$I_P = k_a + k_d(\boldsymbol{N} \cdot \boldsymbol{L})I + k_s(\boldsymbol{N} \cdot \boldsymbol{H})^g \quad (1)$$

式中:g 为高光控制系数.

(4)表面绘制. 把采样体数据时最早大于表面阈值的体素作为表面. 启用 Alpha 测试来指定阈值,再根据表面方向 $\boldsymbol{N}$、光线方向 $\boldsymbol{L}$ 及视线方向 $\boldsymbol{V}$ 的半角向量 $\boldsymbol{H}$ 等来确定表面体素的最终颜色为

$$I_s = k_d(\boldsymbol{N} \cdot \boldsymbol{L}) + k_s(\boldsymbol{N} \cdot \boldsymbol{H})^g \quad (2)$$

上述 4 种绘制效果如图 1 所示.



(a)表面绘制　(b)MIP 绘制　(c)Phong 绘制　(d)RGBA 绘制

图 1　引擎数据的多种绘制效果

2　空间跳跃算法

与 RGBA 绘制及最大密度投影绘制相比较,Phong 光照绘制和表面绘制速度明显下降. 对此,本文提出了一种新的跳跃空体素算法来加快 GPU 的处理过程.

2.1　八叉树编码

首先设定八叉树高度,根据高度对 3D 体数据逐层剖分,对生成的 8 个子块按空间位置进行编号,如图 2 所示. 构建与遍历八叉树需保存如下信息:包含子块空间与高度信息的方向编码项 C_X、C_Y、C_Z,子块内所有体素的最大最小值 V_{max}、V_{min},分别指向 8 个子结点的指针 $N_{children}[8]$,因此定义一种简单而高效的数据结构来保存剖分生成的子块信息. 本文数据结构描述如下:

Struct OctreeNode
{
unsigned int C_X, C_Y, C_Z;
int V_{max}, V_{max};
OctreeNode * $N_{children}[8]$;
}

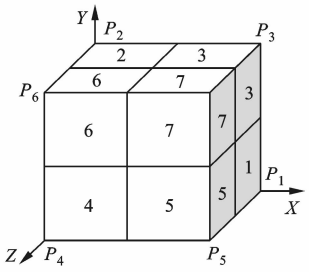


图 2　八叉树分块示意图

当八叉树结点的某方向编码为 $c_1c_2c_3\cdots c_h$(二进制形式),八叉树高度为 h,体数据在该方向上的宽度为 D_{width},则子块在体数据中该方向上的起始点坐标和宽度分别为

$$\left.\begin{aligned} i_{coord} &= (2^{-1}c_1 + 2^{-2}c_2 + 2^{-3}c_3 + \cdots + 2^{-h}c_h)D_{width} \\ i_{width} &= 2^{-h}D_{width} = 2^{-h}D_{width} \end{aligned}\right\} \quad (3)$$

为了在方向编码转换为十进制后仍能保存长度信息,在编码前加入一个标识位 1,这时编码就变为

$1c_1c_2c_3\cdots c_n$，转换为十进制数后就可以正确地保存子块信息.

2.2 视点排序

3D纹理体绘制算法按从后向前的顺序混合切片，因此遍历某结点的子结点也需按此顺序. 根据式(4)计算结点的子结点与视平面之间的距离

$$\boldsymbol{V}\cdot\boldsymbol{p}-d_0=d \tag{4}$$

式中：$\boldsymbol{V}$ 为视线方向；d_0 为视平面与原点的距离；d 为空间一点到平面的距离；$\boldsymbol{p}$ 为子结点的中心点坐标. 把 $\boldsymbol{p}$ 带入式(4)中求出各个子结点到平面的距离，对求出的距离取绝对值后排序.

2.3 遍历八叉树

从根结点出发，按2.2节的排序逐层遍历八叉树. 遍历某结点时，判断结点最大最小值范围 A($A=[V_{\min}, V_{\max}]$)和预设的显示范围 B($B=[T_{\min}, T_{\max}]$)之间的关系.

$A\cap B=\varnothing$时，表明该结点对应所有体素都不在显示范围内，跳过并遍历下一个结点.

$A\cap B\neq\varnothing$，且 $A\not\subset B$ 时，表明该结点对应有部分体素在显示范围内，如果该结点是叶子结点，则为有效结点，按排序遍历下一个结点；如果不是叶子结点，则遍历它的子结点.

$A\subseteq B$ 时，表明该结点对应子块内的所有体数据全部在显示范围内，该结点为有效结点.

2.4 交点计算

当体数据被剖分时，所有有效子块的边界都和采样平面相交形成采样多边形. 对此，本文采用一种快速交点计算的算法：首先根据采样平面的坐标方程计算采样平面与子块每条边的第一个交点 ρ_s

$$\boldsymbol{n}_s\cdot\rho_s-d_0=0 \tag{5}$$

式中：$\boldsymbol{n}_s$ 为采样平面的法线方向；d_0 为该平面与原点的距离. 子块2个顶点 $\boldsymbol{p}_i$ 和 $\boldsymbol{p}_j$ 之间边上的点可以表示为

$$\boldsymbol{p}_m=\boldsymbol{p}_i+m\boldsymbol{e}_{ij}\quad m\in[0,1] \tag{6}$$

式中：$\boldsymbol{e}_{ij}$ 为从顶点 $\boldsymbol{p}_i$ 到 $\boldsymbol{p}_j$ 的向量. 把式(6)带入式(5)中，计算出的 $\boldsymbol{p}_s$ 即是边与平面的交点. $\boldsymbol{e}_{ij}$ 上表示相邻交点间的矩离向量 $\boldsymbol{e}_{\text{dist}}$ 可以通过下式求出

$$\boldsymbol{n}_s\cdot\boldsymbol{e}_{\text{dist}}=\boldsymbol{n}_s\cdot\lambda\boldsymbol{e}_{ij}=d_{\text{slic}}\qquad\lambda\in[0,1] \tag{7}$$

式中：d_{slic} 为相邻采样平面的间距. 计算出 $\boldsymbol{e}$ 后，同一条边上后续的交点坐标就可通过第一个交点与 $\boldsymbol{e}_{\text{slic}}$ 相加求得. 交点计算完后，对它们排序生成采样多边形.

经过空间跳跃，采样平面只与有效结点的子块相交，生成采样多边形的总面积大大减小，在图形流水线对采样多边形进行图元装配和光栅化后，生成的片元数量也大大减少，从而缩短了GPU的处理时间，达到提高体绘制速度的效果.

3 实验结果与分析

实验平台如下：CPU为Pentium D 2.8 GHz、内存为1.0 GB、GPU为NVIDIA GeForce 7900GS. 2种实验数据都是unsigned char类型，设定的显示范围是[127,255].

图3为脚数据加速前后的绘制效果，其中利用空间跳跃加速前后的绘制图像完全相同，证实了空间跳跃跳过的空体素对最终图像无贡献. 表1、表2比较了实验数据3D纹理体绘制与八叉树编码加速时子块大小分别为 64^3、32^3、16^3、8^3、4^3 像素的绘制速度.

由表1、表2可以看出，表面绘制与Phong光照绘制速度相对较慢. 经过分析，主要原因是表面绘制与phong光照绘制的片元着色程序较为复杂，降低了GPU的处理速度. 利用八叉树编码加速时，随着子块体积逐渐变小，绘制速度逐渐加快. 当子块体积小于某一规模后进一步变小时，速度开始下降. 通过统计被跳过的空体素，如表3所示，发现当子块体积较大时，子块最大最小值范围较大，跳过的空体素较少，而随着子块体积逐渐变小，子块最大最小值范围也逐渐缩小，跳过空体素数量大幅度增加，因此速度增加较快. 当子块体积进一步变小时，跳过的空体素数量趋近极限，而生成的子块包围盒的数量却以 2^{3h}(h 为八叉树高度)增长，极大地增加了CPU求解采样平面与子块交点坐标的计算量，因此导致了速度下降.

表1 不同剖分子块大小时引擎数据(256×256×256像素)多种效果的绘制速度

绘制方法	绘制速度/帧·s^{-1}					
	未分块	64×64×64像素	32×32×32像素	16×16×16像素	8×8×8像素	4×4×4像素
RGBA绘制	10.38	15.02	22.28	35.16	19.03	3.53
最大密度投影	10.84	15.08	22.17	35.73	19.07	3.46
Phong光照	5.37	8.84	12.79	22.03	18.55	3.32
表面绘制	4.69	7.39	10.87	18.99	19.03	3.02

图 3　脚数据绘制效果

表 2　不同剖分子块大小时脚数据(256×256×256 像素)多种效果的绘制速度

绘制方法	绘制速度/帧·s^{-1}					
	未分块	64×64×64 像素	32×32×32 像素	16×16×16 像素	8×8×8 像素	4×4×4 像素
RGBA 绘制	10.83	16.99	20.17	36.15	25.36	5.50
最大密度投影	10.84	17.32	20.11	36.15	25.36	5.50
Phong 光照	5.37	8.38	11.99	24.17	5.77	1.10
表面绘制	3.98	7.49	10.67	18.03	5.63	1.04

表 3　不同体数据在不同剖分子块大小绘制时跳过空体素占总体素的百分比

	跳过空体素占总体素的百分比/%					
	未分块	64×64×64 像素	32×32×32 像素	16×16×16 像素	8×8×8 像素	4×4×4 像素
引擎数据	0.00	32.81	77.15	80.84	85.46	88.97
脚数据	0.00	45.31	72.66	84.64	90.92	93.97

对本文算法与自适应分块算法的加速比进行比较,如表 4 所示,本文算法的平均加速比为 3.7,高于自适应分块加速算法平均加速比 3.3,而且本文算法仅根据空间信息对体数据进行规则划分,极大地简化了体数据剖分的过程.因此,本文算法具有更好的实用性.

表 4　不同体数据的平均加速比

参数	引擎数据	脚数据	总平均值
加速比	3.7	3.9	3.8

4　结　论

本文算法通过跳跃空体素,减少了经过图元装配及光栅化操作后生成片元的数量,从而减少了 GPU 对片元着色、测试、混合等处理的次数,提高了绘制速度.本文提出的算法和不同绘制效果的 3D 纹理体绘制算法相结合,均取得了较好的加速效果.由于目前受图形硬件存储空间的制约,当体数据量过大,不能全部放入显存时,本文算法无法对体数据进行处理,因此下一步研究的重点将放在解决图形硬件的存储瓶颈、实现海量体数据在图形硬件上的快速绘制上.

参考文献:

[1] AKELEY K. Reality engine graphics [C]// Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques. New York, USA: ACM Press, 1993: 109-116.

[2] GELDER A, KIM K. Direct volume rendering with shading via 3D textures [C]// Proceedings of ACM Symposium on Volume Visualization. New York, USA: ACM Press, 1996: 23-30.

[3] STEGMAIER S, STRENGERT M, KLEIN T, et al. A simple and flexible volume rendering framework for graphics-hardware-based raycasting [C]// Proceedings of the International Workshop on Volume Graphics 2005. Los Alamitos, CA, USA: IEEE Computer Society, 2005: 187-195.

[4] 陈为. 硬件加速反走样体 Splatting 算法 [J]. 计算机辅助设计与图形学学报, 2005, 17(4): 677-682.

CHEN Wei. A hardware accelerated anti-aliasing volume splatting algorithm [J]. Journal of Computer Aided Design & Computer Graphics, 2005, 17(4): 677-682.

[5] JANSEN T, RYMON B V, HANSSEN N, et al. GPU-based frequency domain volume rendering [C]// Proceedings of the 20th Spring Conference on Computer Graphics. New York, USA: ACM Press, 2004: 55-64.

[6] 宋涛, 欧宗瑛, 王瑜, 等. 八叉树编码体数据的快速体绘制算法 [J]. 计算机辅助设计与图形学学报, 2005, 17(9): 1990-1996.
SONG Tao, OU Zongying, WANG Yu, et al. Fast volume rendering algorithm of octree encoded volume [J]. Journal of Computer-Aided Design & Computer Graphics, 2005, 17(9): 1990-1996.

[7] 吕广宪, 潘懋, 吴焕萍, 等. 面向真三维地学建模的海量虚拟八叉树模型研究 [J]. 北京大学学报(自然科学版), 2007, 43(4): 496-501.
LV Guanxian, PAN Mao, WU Huanping, et al. Research on large virtual octree model for true three dimensional geo-science modeling [J]. Acta Scientiarum Naturalium Universitatis Pekinensis, 2007, 43(4): 496-501.

[8] WEILER M, WESTERMANN R, HANSEN C, et al. Level-of-detail volume rendering via 3D textures [C]// Proceedings of the 2000 IEEE Symposium on Volume Visualization. New York, USA: ACM Press, 2000: 7-13.

[9] LI Wei, MUELLER K, KAUFMAN A. Empty space skipping and occlusion clipping for texture-based volume rendering [C]// Proceedings of the 14th IEEE Visualization 2003. Los Alamitos, CA, USA: IEEE Computer Society, 2003: 317-324.

[10] REZKSALAMA C, ENGEL K, BAUER M, et al. Interactive volume on standard PC graphics hardware using multi-textures and multi-stage rasterization [C] // Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware. New York, USA: ACM Press, 2000: 109-118.

[11] ENGEL K, KRAUS M, ERTL T. High-quality pre-integrated volume rendering using hardware-accelerated pixel shading [C]// Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware. New York, USA: ACM Press, 2001: 9-16.

(编辑 刘杨 苗凌)

2008 IEEE 国际电子商务工程学术会议在西安交通大学召开

2008年10月22日,由IEEE主办西安交通大学承办的第五届国际电子商务工程会议(2008 IEEE International conference on E-business engineering, ICEBE)在南洋酒店国际会议厅开幕. 郑南宁校长担任本届会议的荣誉主席,朱世华副校长代表学校在大会致欢迎词. 来自美国、英国、德国、意大利、日本、新加坡等17个国家以及国内清华、北大、浙大、上海交大等单位的130余名专家学者及研究人员参加了会议. 大会主席、西安交通大学电子与信息工程学院计算机系齐勇教授主持开幕式.

开幕式后,加拿大国家研究委员会 Shen Weiming 主任研究员为大会作了题为"Integration of Web Service and Software Agnets for Industrial Application"的特邀报告.

10月23日,IBM华盛顿研究中心 Juril Paraszczak 教授及香港科技大学的刘云浩教授分别在大会作了"The Greening of Innovation"和"Breaking the Privacy Barrier"的主题报告. 本届会议由18个session组成,会议主题涵盖了电子商务研究领域的各个方面,大会围绕电子商务研究领域中的关键技术问题进行了深入的交流、探讨.

(信息来源:西安交大科技在线)