软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2012,23(Suppl.(2)):69−76

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于 K-D 树快速精确排序的四面体可视化*

李 昕[1,2], 吴福理[3], 童琪杰[3], 陈伟锋[2], 华 炜[2], 陈 为[2+]

[1](中国石油大学(华东) 计算机与通信工程学院,山东 青岛 266555)

[2](CAD & CG 国家重点实验室(浙江大学),浙江 杭州 310058)

[3](浙江工业大学 计算机学院,浙江 杭州 310032)

Quick and Accurate Sorting for Visualization of Tetrahedral Volume Datasets Based on K-D Tree

LI Xin[1,2], WU Fu-Li[3], TONG Qi-Jie[3], CHEN Wei-Feng[2], HUA Wei[2], CHEN Wei[2+]

[1](College of Computer and Communication Engineering, China University of Petroleum (East China), Qingdao 266555, China)

[2](State Key Laboratory of CAD & CG (Zhejiang University), Hangzhou 310058, China)

[3](College of Information Engineering, Zhejiang University of Technology, Hangzhou 310032, China)

+ Corresponding author: E-mail: chenwei@cad.zju.edu.cn, http://www.cad.zju.edu.cn/home/chenwei

Li X, Wu FL, Tong QJ, Chen WF, Hua W, Chen W. Quick and accurate sorting for visualization of tetrahedral volume datasets based on K-D tree. *Journal of Software*, 2012,23(Suppl.(2)):69−76 (in Chinese). http://www.jos.org.cn/1000-9825/12026.htm

Abstract: The Projected Tetrahedra is a popular method in the field of tetrahedra database visualization. Tretrahedra must be sorted according to obstruction between them to achieve an accurate rendering image, but strong dependency among tetrahedra results in not only inefficient sorting, but also poor parallel execution. This paper proposes a tetrahedra sorting algorithm which is based on K-D tree spatial partitioning. The database in one leaf node are peeled into layers in natural order, and the tetrahedra in the same layer are unobstructed . The peeling of different leaf node is independent, and their sorted tetrahedra are organized together according to the obstruction between leaf nodes. The sorting efficiency has improved greatly through two-level parallelism and guarantees accurate sorting. The data structure can be implemented easily in a graphics processing unit (GPU). The experimental results show that the quick and accurate sorting based on K-D tree processed in GPU shortens the sorting time greatly.

Key words: volume rendering; tetrahedra projection; K-D tree spatial partitioning; GPU (graphics processing unit); exact sorting

摘 要: 投影四面体法是四面体体数据可视化的一种重要方法.为了保证绘制结果准确,每一帧都需要对所有四面体按照遮挡关系进行排序,然而四面体之间强烈的依赖性不仅导致排序效率很低,而且很难并行实现.提出了一种基于 K-D 树空间划分的快速精确的四面体排序策略,在每个叶节点内逐层并行提取互不遮挡的四面体,层与层之间

* 基金项目: 国家自然科学基金(61003193, 81172124); 国家高技术研究发展计划(863)(2012AA120903); 浙江省科技厅公益项目(2011C21058)

收稿时间: 2012-05-30; 定稿时间: 2012-09-29

自然有序,且各叶节点的操作彼此独立进行.最后将结果按照叶节点之间的空间遮挡顺序组织在一起.通过两个级别的并行,在保证精确排序的同时极大地提高了效率,且数据结构易于图形处理单元 GPU 实现.实验结果表明,基于 K-D 树快速精确排序策略的 GPU 实现极大地缩短了排序时间.

关键词: 体绘制;四面体投影;K-D 树空间划分;图形处理单元;精确排序

科学计算可视化是计算机应用科学的一个重要研究方向,其中体绘制因为能够有效展示三维数据场的内部细节信息,被广泛地应用于地质数据分析[1]、流体计算仿真[2]、气象分析[3]等许多领域.可视化的结果可以让相关人员从繁杂的数据中摆脱出来,迅速提取有意义的特征和结果.与规则体数据在三维空间中等间距剖分不同,大部分物理模拟计算(如有限元分析)生成的不规则体数据在空间上分布密度变化较大,能够展现变化剧烈区域的细节,并同时减少变化平缓区域的数据存储冗余.四面体是一种最常见的不规则体数据,作为三维空间中的单体,它具有表示简单、插值容易、易于表达其他类型体数据的特点,因而成为近些年来科学可视化领域的热点研究问题[3−9].

四面体的体绘制方法主要分为光线投射法[4]和投影法[5].光线投射法在规则体数据的可视化中应用非常广泛,能够获得高质量的结果,但逐光线遍历不规则体数据效率较低;与其相比,投影法将所有四面体按照一定的顺序投影到屏幕上,累积光学属性,有利于存储空间分配和计算机的并行实现.Shirley 等人提出的四面体投影法(projected tetrahedra,简称 PT)[5]因为将投影后的四面体分解为三角形,利用了现代 GPU 的硬件加速,因而极大地促进了投影法的发展,成为一种基本四面体体绘制方法,被许多研究者所采用[6−8].

PT 法需要在投影前将四面体进行排序,这是制约 PT 法效率的关键因素.为了提高排序效率,Maximo 等人[6,7]以牺牲一定绘制质量为代价,利用四面体重心的 z 坐标进行近似排序.而早在 1992 年提出的精确排序 MPVO(meshed polyhedra visibility ordering)方法[10]利用遮挡关系进行排序,因为四面体之间的空间邻接关系依赖性很强,只能按照从远到近或从近到远的顺序进行排序,所以像快速排序等许多 CPU 中的优秀排序方法不能应用于其中,难以并行实现.

针对上述问题,本文提出了一种基于 K-D 树的快速精确排序方法.首先沿着坐标轴将数据集进行空间切割,将切割结果保存到 K-D 树中.然后在每个叶节点内逐层并行提取四面体,同层四面体之间互不遮挡,层与层之间自然有序.每个叶节点内的四面体排序是独立进行的,彼此并行.最后按照叶节点之间的空间遮挡关系将排序结果组织在一起.通过两个级别的并行,极大地提高了排序的效率.

本文第 1 节讨论相关工作.第 2 节介绍基于 K-D 树的空间切割.第 3 节介绍排序和绘制.第 4 节展示实验结果.第 5 节总结本文工作.

1 相关工作

投影法是基于物体空间的直接体绘制方法.Shirley 和 Tuchman 提出的 PT 法[5]因为用到了现代计算机的光栅化硬件加速而将投影法推到了新的高度,在此基础上,产生了一系列新方法[6−8].Wylie 等人[8]提出了第一种采用 GPU 加速的 PT 方法.方法用统一的模型表示投影分解的类型.这种简单而快速的方法在提高效率的同时带来了大量的数据冗余.Marroquim 等人的方法[6]在一次绘制中两次利用了 GPU.先在 GPU 中计算投影类型、四面体重心 z 坐标等,然后在 CPU 中排序,最后用 GPU 绘制.数据传输效率对该方法影响很大.硬件加速的四面体投影方法[7]利用了现代硬件的发展,将产生顶点数量变化的投影分解操作在几何渲染器中完成,建立了统一的绘制模型,效率较高.

从后向前:

$$C'_i=C_i+(1-A_i)C'_{i-1} \tag{1}$$

从前向后:

$$C'_i=C'_{i+1}+(1-A'_{i+1})C_i,\ A'_i=A'_{i+1}+(1-A'_{i+1})A_i \tag{2}$$

投影法在累积所有四面体对屏幕的光学属性贡献时,按照公式(1)或公式(2)[11]进行累加.其中,C 表示颜色,A

表示不透明度,C'和A'表示累加后的结果.无论从前向后还是从后向前,都必须依次进行,这就要求四面体必须是有序的,且顺序随着视点的变化而有所不同.现已提出许多四面体排序方法.基于 GPU 的快速排序[12]和双调排序网络**这两种方法按照四面体重心的 z 坐标进行不精确排序.快速自适应多路排序方法[13]考虑了四面体之间的空间连贯性,加快了不精确排序的效率.硬件加速的可见性排序方法[9]先后在物体空间和图像空间进行了两次排序,排序精度较高.MPVO[10]是一种精确排序方法,四面体间强烈的依赖性导致效率很低,且难以并行实现.

为了提高精确排序效率,本文采用 K-D 树方式将原始体数据进行切割,每个叶节点内的四面体在 GPU 中独立排序,各叶节点之间的操作并行进行,从而提高整体的效率.很多方法[14,15]在光线追踪中使用 K-D 树,并证明了由 K-D 树建立的场景结构比统一模型效率高很多.本文将 K-D 树引入到四面体排序中,同样带来了大幅度的效率提升.

2 基于 K-D 树的空间切割

PT 法必须首先沿着视线方向对所有四面体进行排序,然后才能投影和绘制.对于大规模体数据,这种排序开销很大,而且随着视点的改变,必须重新排序.将原始数据分成多个子集,在每个子集中随着四面体数量的减少,排序的负担也会下降.各子集彼此独立,并行排序,就可以提高整体的效率.本文在这个策略的指导下,采用 K-D 树重新组织体数据集,与切割面相交的四面体将被切割成多个小四面体.属于不同叶节点的四面体集并行排序.

2.1 切割的原因

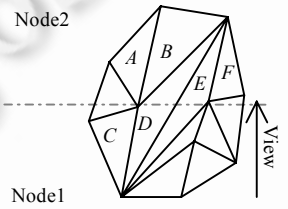


Fig.1 Diagram of partition (dotted line is partition plane)

图 1 数据集切割示意图(虚线表示切割面)

在采用多个子集表示原始数据时,最主要的问题是如何处理与切割面相交的四面体.如果将这些四面体完全丢弃,则会在绘制结果中产生裂缝.如果将四面体归属到离视点较近的叶节点中,则在某些情况下会出现不正确的遮挡关系,导致绘制效果中出现明显的错误.如图 1 所示,四面体 E 与切割面相交,如果将 E 归属到叶节点 1 中,此时 F 属于叶节点 2,按照叶节点之间的空间遮挡关系,E 应该遮挡 F,但事实正好相反.如果将四面体归属到离视点较远的那个叶节点中,会产生同样的问题,参见图 1 中的四面体 C 和 D 的关系.如果将切割面上的四面体同时归属到与其相交的两个叶节点中,则会使问题进一步恶化.出现以上现象的根源在于在空间上跨越多个叶节点的四面体,因此本文将与切割面相交的四面体切割成多个小四面体,使每个四面体在空间上属于且只属于 1 个叶节点,保证不同叶节点内的四面体数据集相互独立,从而避免上述问题的出现.

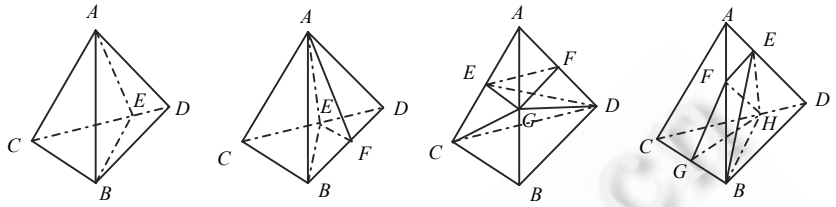
本文采用的切割面是与坐标轴垂直的平面,这样保证切割后每个叶节点在空间上都是长方体,一方面可以防止相互遮挡的现象,另一方面也容易确定两个叶节点在给定视点下的先后顺序.平面与坐标轴垂直可以使计算进一步简化.

2.2 切割模型

一个四面体被某个平面切割,会产生 4 种模型,如图 2 所示.在图 2(a)中,切割面通过原四面体的一条边,并与

** http://developer.download.nvidia.com/compute/cuda/sdk

另一条边相交,切割后形成 2 个新四面体;在图 2(b)中,切割面通过原四面体的一个顶点,而与该顶点相对面上的两条边相交,新生成的四棱锥 *ABCEF* 进一步被切割成两个四面体;在图 2(c)中,切割面与原四面体的 3 条边相交,三棱台 *EFGCDB* 被进一步划分为 3 个四面体.共产生 4 个新四面体;在图 2(d)中,切割面与原四面体的 4 条边相交,每一侧都被进一步划分为 3 个四面体.共生成 6 个新四面体.因为两侧的情况完全相同,为了视觉上的清晰,图 2(d)中只画了其中一侧的切割情况.



(a) 2 tetrahedra (b) 3 tetrahedra (c) 4 tetrahedra (d) 6 tetrahedra

Fig.2 Four types model of partitioned tetrahedra. Partition plane are: (a) ABE; (b) AEF; (c) EFG; (d) EFGH

图 2 四面体被平面切割的 4 种模型,切割面分别为:(a) ABE; (b) AEF; (c) EFG; (d) EFGH

2.3 切割策略

按照图 2 所示的切割模型,每个被切割的四面体将会产生 2~6 个新四面体.四面体的数量影响绘制的效率以及存储空间的分配,而切割后两侧四面体数量是否平衡影响排序的效率.因此分割面的确定必须考虑这两个因素,具体算法如下:

1) 在有效空间范围内,均匀建立 N 个与 x 轴垂直的切割平面 $X_i(i=1,2,\ldots,N)$,每个平面将原始的数据集切割成不相交的两部分,新生成四面体的数量与原数据集中四面体数量的比例为 β_i.

2) 找到最小的 β_i,如果 β_i 大于阈值 Φ,则转步骤 5);否则记录对应的 X_i.

3) 计算下式 α 的值,其中,l 和 r 表示 X_i 两侧所包含的四面体个数:

$$\alpha=(|l-r|)/((l+r)/2).$$

如果 α 小于设定的阈值 ε,则说明切割平面两侧负载平衡,记录 X_i,转步骤 6).

4) 查找 β_i 之外的最小值,重复步骤 2)和步骤 3),直到发现第 1 个符合条件的切割平面为止.如果所有的平面都不符合要求,则放弃沿 x 轴切割.

5) 沿 y,z 坐标轴做相同的处理,得到切割平面 Y_j 和 Z_k.X_i,Y_j 和 Z_k 所对应的 β 值最小的平面为切割平面.

6) 对新生成的节点,重复以上的步骤 1)~步骤 5),直到不能确定新的切割平面,或到达指定的划分深度为止.

在以上策略中,步骤 2 保证新生成的四面体数量尽量少,步骤 3 保证每次切割后两侧四面体数量的负载平衡.如果通过以上策略找不到合适的切割平面,则可以将原始的体数据根据自身的特点进行适当的旋转再切割.

采用 K-D 树记录切割之后的体数据.每个非叶节点记录切割平面的位置,每个叶节点记录所包含的四面体.任意节点都是空间上一个坐标轴对齐的长方体.切割后的任意一个四面体 T,存在且仅存在 1 个叶节点 N,T 在空间上不超出 N 的范围,记为 $T\in N$.

3 排序和绘制

为了保证绘制结果准确,本文将所有四面体进行精确排序.切割后形成的 K-D 树使排序工作能够在各个叶节点的空间范围里同时进行,然后按照叶节点之间的空间遮挡关系将各自的排序结果组织在一起,并依次投影到屏幕上,形成最终的图像.

3.1 叶节点内四面体的排序

对于叶节点内的所有四面体,按照以下 3 个步骤进行排序:

1) 建立四面体数据集的邻接关系.根据四面体之间的共享顶点和共享面,建立邻接关系.

2) 建立相邻四面体之间的遮挡关系.任意共享面都是平面,对于给定的视点,无论是平行投影还是透视投影,都可以确定共享该面的两个四面体之间的遮挡关系.

3) 拓扑排序.首先找到所有不被遮挡的四面体,作为第 0 层;然后查找与第 0 层相邻,且只被第 0 层遮挡的四面体,作为第 1 层;类似操作从近到远依次进行下去,直到所有四面体都被确定层次.同层的四面体彼此互不遮挡,因此顺序任意.拓扑排序过程也可以从远到近进行.

3.2 整体顺序的建立

空间中任意两个凸体 A 和 B,如果针对某一视点位置 e,B 遮挡 A,记为 $A<_eB$;如果 B 遮挡 A 或互不遮挡,记为 $A\leqslant_eB$.根据 Williams 等人[10]对遮挡提出的定义,可以推导出对于任意两个叶节点 N_m 和 $N_k(m\neq k)$和任意两个四面体 $T_i\in N_m$ 和 $T_j\in N_k$,如果 $N_m\leqslant_e N_k$,则 $T_i\leqslant_eT_j$.因此,只要将每个叶节点完成排序的四面体按照叶节点之间的遮挡顺序组织在一起,就完成了所有四面体的排序.

如图 3 所示,体数据被划分为 4 个叶节点,按照从前向后 $B_1B_2B_3B_4$ 的顺序将各自独立排序的结果依次存储到四面体绘制队列中.B_2 中的四面体要比 B_3 中的四面体先执行投影操作.虽然 B_3 中的某些四面体可能离视点更近,但并不遮挡 B_2 中的四面体,不会影响最终的绘制结果.

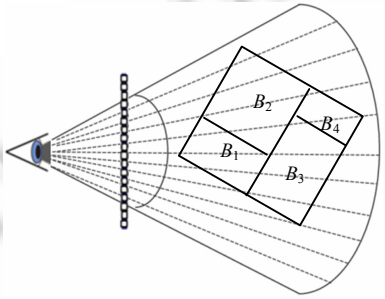


Fig.3 Diagram of 4 nodes of partition

图 3 体数据被划分为 4 个叶节点的示意图

3.3 投影分解和绘制

按照 PT 法的绘制流程,将绘制队列中的四面体逐一投影到屏幕上,然后按照图 4 所示的 4 种类型进行分解.分解后的每个三角形在顶点处记录了通过该点光线的入点值 S_f、出点值 S_b 和光线在该四面体内的有效距离 l,如图 5 所示.将三角形光栅化,通过查表可以得到该四面体对相应像素的贡献.按照投影顺序累积每个四面体的贡献,得到最终的绘制结果,如图 6 和图 7 所示.

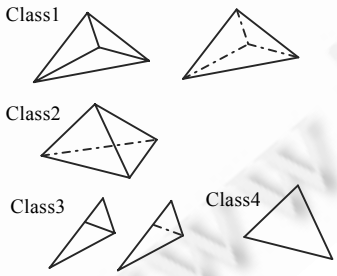


Fig.4 Decomposition type

图 4 分解类型

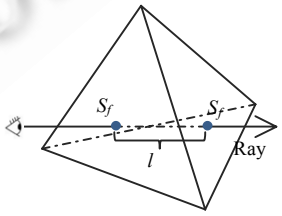


Fig.5 Ray and tetrahedra

图 5 光线与四面体

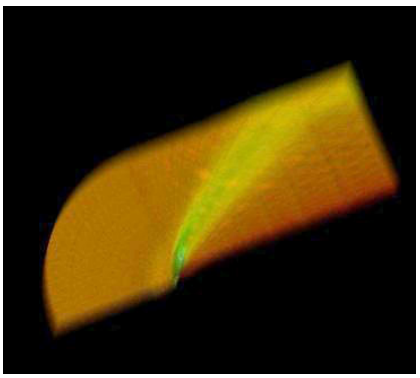


Fig.6 Rendering result of BluntFin
图 6 BluntFin 数据集的绘制结果

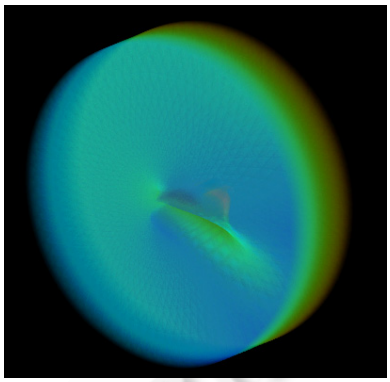


Fig.7 Rendering result of Post
图 7 Post 数据集的绘制结果

4 实验结果

实验配置如下:CPU:Intel E7400 2.8GHz,内存:4G,图形卡:NVIDIA GeForce GTX260(1 024MB 显存).本文采用的 CUDA 版本为 3.2,绘制图像分辨率均为 512×512.

4.1 分割与四面体数量的变化对比

随着切割次数的增加,四面体的数量也会增长.因为本文方法每次切割都寻找新增四面体数量最小的切割面,因此增长比例得到较好控制.如图 8 所示,数量变化基本成线性增长,当切割为 8 份时,3 个数据集四面体数量都几乎为原数据集的 1.6 倍.但是当切割次数过多时,寻找最优面的耗时、需要被切割四面体的数量以及切割后重复顶点的剔除都会大量增加,使得切割的时间成倍增长,而且会对数据存储造成沉重负担,因此切割深度必须限制在一定范围内.

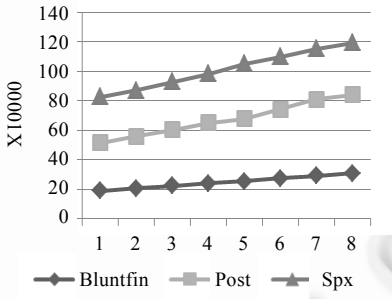


Fig.8 Partition and tetrahedra number
图 8 分割次数与四面体数量的变化

4.2 分割对排序层次和效率的影响

分割对于排序层数的影响非常大.每次分割将完全串行的层次遍历过程分成了并行的两部分.图 9 和图 10 展示了原始数据集,沿视线方向将数据集切割成 2 部分或 4 部分,这 3 种方案对排序层次和时间的影响.表 1 针对 Bluntfin,Post 和 Spx2 这 3 个数据集,数据集分成 2 部分和 4 部分平均下降为原来的 61.0%和 38.0%,而排序时间平均下降为分割前的 63.7%和 43.6%.在保证绘制结果不变的同时,极大地减少了排序的时间.分割对于投影分解和绘制的影响非常小,因此整体的帧率随之上升,表 1 中的 Spx2 数据集划分为 4 部分后帧率上升为原来的 1.8 倍.

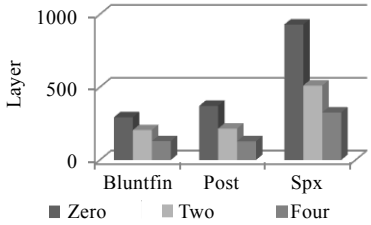


Fig.9 Partition and layer number

图 9 切割后层数的变化

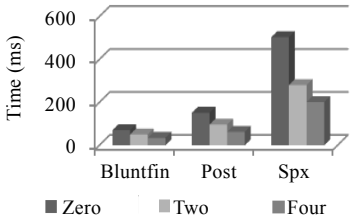


Fig.10 Partition and sorting time

图 10 切割后排序时间的变化

Table 1 Data comparison of datasets after partition

表 1 不同数据集切割后的数据对比

数据集	四面体数(k)	层次下降比例(%)		时间下降比例(%)		帧率		
		2 部分	4 部分	2 部分	4 部分	不分割	2 部分	4 部分
Bluntfin	187	70.0	44.2	71.5	49.3	9.4	11.5	14.1
Post	513	58.1	34.7	63.8	41.4	3.9	5.0	6.0
Spx2	828	54.9	35.1	55.7	40.1	1.5	2.2	2.7
平均值	–	61.0	38.0	63.7	43.6	–	–	–

5 总 结

本文提出了基于 K-D 树的快速精确排序方法,提高了投影法体绘制的效率.该方法的关键在于,通过切割使不同叶节点内的四面体具有了相对的独立性,从而达到了并行排序的目的.在叶节点排序时,属于同层的四面体是并行查找的,这两个级别的并行极大地提高了排序的效率.

References:

[1] Ma RN, Zhang EH, Yang JY, Zhao CX. Research on segmenting and volume rendering of irregular seismic events. Journal of Computer Research and Development, 2005,42(5):883−887 (in Chinese with English abstract).

[2] Wang FJ. Analysis of Computational Fluid Dynamics—The Principle and Application of CFD. Beijing: Tsinghua University Press, 2004. (in Chinese).

[3] Jing Ye, Svakhine, N, Lasher-Trapp S, Baldwin M, Ebert DS. An atmospheric visual analysis and exploration system. IEEE Trans. on Visualization and Computer Graphics, 2006,12(5):1157−1164.

[4] Weiler M, Kraus M, Merz M, Ertl T. Hardware-Based ray casting for tetrahedral meshes. In: Proc. of the IEEE Visualization. Los Alamitos: IEEE Computer Society Press, 2003. 333−340.

[5] Shirley P, Tuchman A. A polygonal approximation to direct scalar volume rendering. ACM Siggraph Computer Graphics, 1990, 24(5):63−70.

[6] Marroquim R, Maximo A, Farias R, Esperança C. Volume and isosurface rendering with GPU-accelerated cell projection. Computer Graphics Forum, 2008,27(1):24−35.

[7] Maximo A, Marroquim R, Farias R. Hardware-Assisted projected tetrahedra. Journal of Computer Graphics Forum, 2010,29(3): 903−912.

[8] Wylie B, Moreland K, Fisk LA, Crossno P. Tetrahedral projection using vertex shaders. In: Proc. of the IEEE Symp. on Volume Visualization and Graphics. Los Alamitos: IEEE Computer Society Press, 2002. 7−12.

[9] Callahan SP, Ikits M, Comba JLD, Silver CT. Hardware-Assisted visibility sorting for unstructured volume rendering. IEEE Trans. on Visualization and Computer Graphics, 2005,11(3):285−295.

[10] Williams PL. Visibility Ordering meshed polyhedra. ACM Trans. on Graphics, 1992,11(2):103−126.

[11] Engel K, Kraus M, Ertl T. High-Quality pre-integrated volume rendering using hardware-accelerated pixel shading. In: Proc. of the ACM SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware. 2001. 9−16.

[12] Cederman D, Tsigas P. A practical quicksort algorithm for graphics processors. In: Proc. of the 16th Annual European Symp. on Algorithms. Heidelberg: Springer-Verlag, 2008. 246−258.

[13] Li X, Wu FL, Chen WF, Hua W, Chen W. Visualizing tetrahedral volume datasets with quick adaptive multiway sorting. Journal of Computer-Aided Design & Computer Graphics, 2011,23(12):2025−2032 (in Chinese with English abstract).

[14] Foley T, Sugerman J. KD-Tree acceleration structures for a GPU raytracer. In: Proc. of the ACM SIGGRAPH/EUROGRAPHICS Conf. on Graphics Hardware. New York: ACM Press, 2005. 15−22.

[15] Horn DR, Sugerman J, Houston M, Hanrahan P. Interactive k-D tree GPU raytracing. In: Proc. of the Symp. on Interactive 3D Graphics and Games. 2007. 167−174.

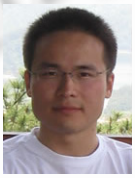
附中文参考文献:

[1] 马仁安,张二华,杨静宇,赵春霞.不规则地质体的分割与体绘制方法研究.计算机研究与发展,2005,42(5):883−887.

[2] 王福军.计算流体动力学分析:CFD 软件原理与应用.北京:清华大学出版社,2004.

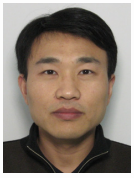
[13] 李昕,吴福理,陈伟锋,华炜,陈为.基于快速自适应多路排序的四面体可视化.计算机辅助设计与图形学学报,2011,23(12):2025−2032.

李昕(1978－),男,辽宁葫芦岛人,博士生,讲师,主要研究领域为科学计算可视化.



陈伟锋(1983－),男,博士生,主要研究领域为真实感绘制,科学计算可视化.

吴福理(1976－),男,博士,副教授,主要研究领域为计算机图形学,科学计算可视化,医学图像处理.



华炜(1972－),男,博士,副教授,主要研究领域为虚拟现实,实时图形绘制,视频处理.

童琪杰(1988－),男,硕士生,主要研究领域为科学计算可视化.

陈为(1976－),男,博士,教授,主要研究领域为科学计算可视化,信息可视化,可视分析.