PKU Visualization Blog 北京大学可视化与可视分析博客

SparseLeap：一种大规模体绘制中高效的空体素跳跃方法（SparseLeap: Efficient Empty Space Skipping for Large-Scale Volume Rendering）

作者: Li, Yanda 日期: 2018年8月27日

体渲染是体数据可视化中的一项重要的任务，体渲染主要分为等值面体渲染(Iso-surface Volume Rendering)及直接体渲染(Direct Volume Rendering), 而在直接体渲染中，最为广泛使用的是光线投射算法(Ray-casting)。对于大规模的体数据，在使用光线投射算法进行体渲染时，若不跳过空白区域，即进行空体素跳跃，则会产生极大的运算量。然而大规模体数据，如神经元、皮肤数据，往往具有精细复杂的结构，这使得空体素跳跃变得极为困难。

在已有的空体素跳跃方法中，最为朴素的方法为对于每一条光线，跳过其第一个非空样本之前以及最后一个非空样本之后的空间，并对两者之间的所有内容进行采样，如图1(b)所示。但此方法对于较为稀疏的数据会采样到其间许多空体素。另一种方法为基于八叉树结构的空体素跳跃方法，即递归地将每块体数据分为八个部分，并对光线经过的每块非空数据记录其入点及出点进行采样，如图1(c)所示。但基于八叉树的方法会额外记录许多空节点，增加采样的运算复杂度。

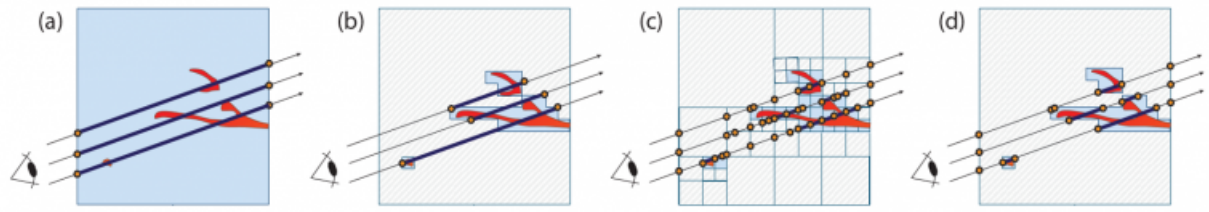

图1. 已有的空体素跳跃方法

该工作提出一种大规模体绘制中的高效空体素跳跃方法，流程主要分为四步，如图2所示。（a）递归地将每块体数据分为八部分，构成八叉树。对于每个叶结点的体数据块，标记属性为空、非空、未知的一种；对于每个非叶结点，统计其子结点的属性占比，并将其属性赋为占比最高的属性。（b）对体数据的八叉树进行宽度优先搜索，对于属性与其父节点不同的体数据块，构造边界框。（c）对于每条光线，在其穿过边界框时，根据规则进行合并与删除，确定采样范围。（d）使用延迟加载的方法，进行光线投射算法的体渲染。

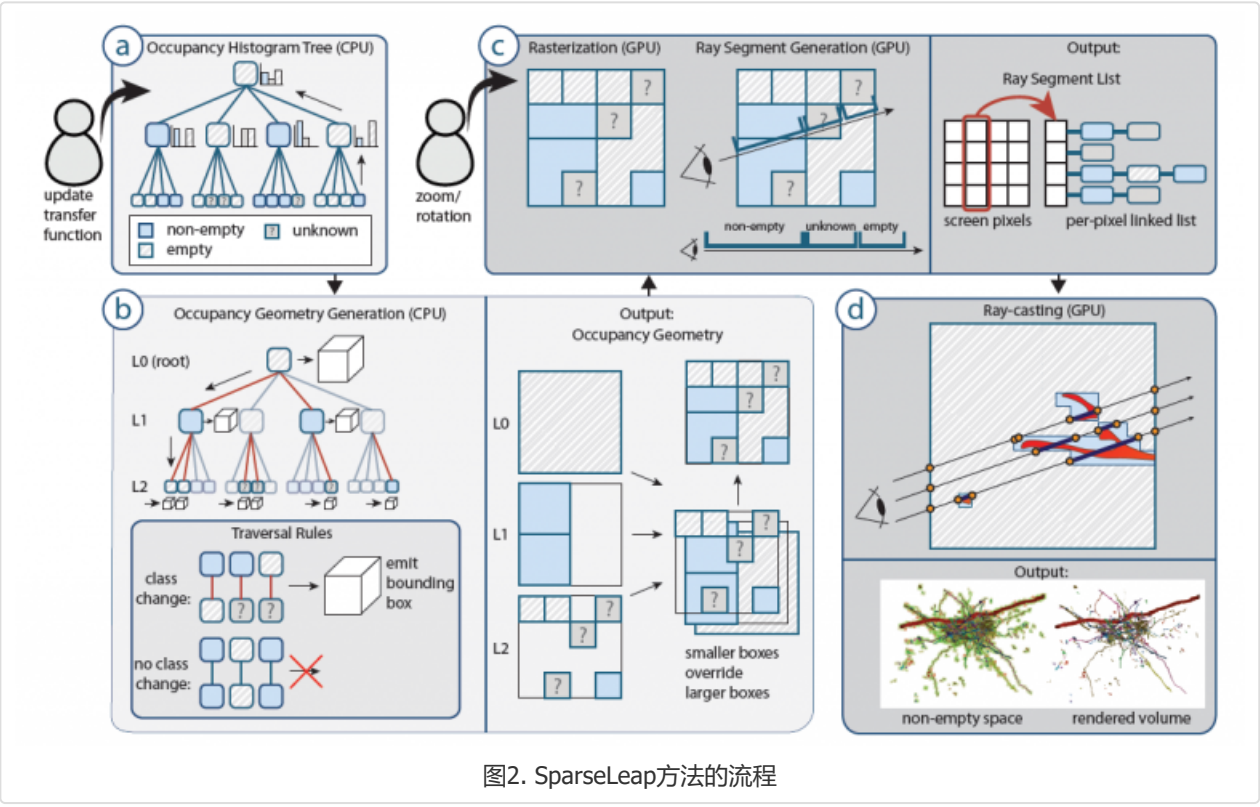

图2. SparseLeap方法的流程

对比此方法与基于八叉树结构的方法及无空体素跳跃方法，结果如图3所示。从实验结果可以看出，无论对于稀疏体数据或是分布较为密集的体数据，该方法均有较好的表现。另外，划分数据块的粒度对体渲染速度也有所影响，划分过细反而会降低渲染速度。

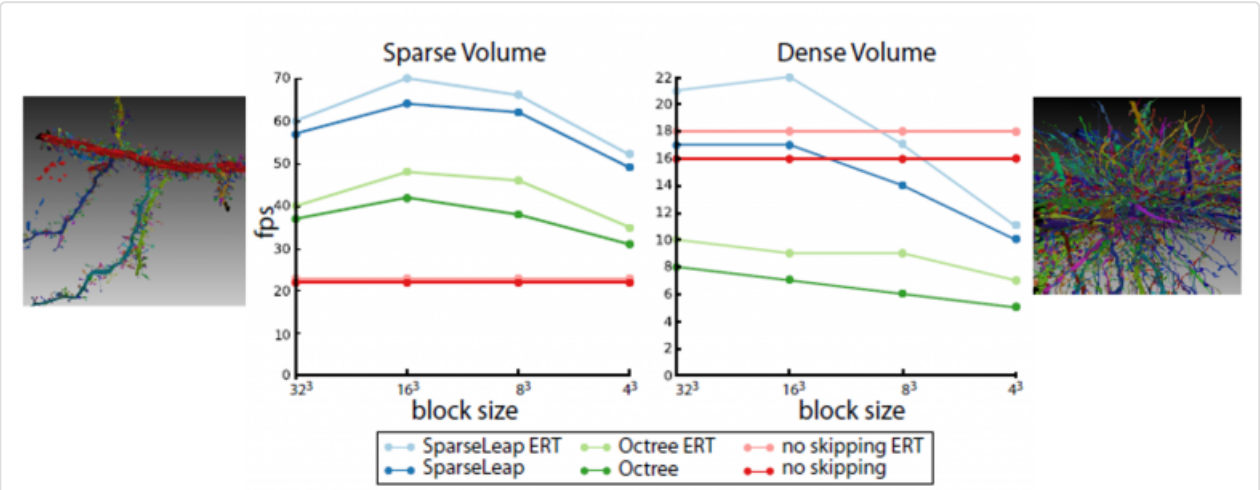

图3. 本文方法与基于八叉树结构的方法及无跳跃方法在稀疏（左）及稠密（右）体数据集上的渲染速度比较

总的来说，该工作提出并实现了一种光线投射算法中的空体素跳跃方法，该方法在渲染大规模、具有复杂结构的体数据时具有更加高效的表现。

参考文献

Hadwiger M, Alawami A K, Beyer J, et al. SparseLeap: Efficient Empty Space Skipping for Large-Scale Volume Rendering.[J]. IEEE Transactions on Visualization & Computer Graphics, 2017, PP(99):1-1.

论文报告 体数据可视化, 直接体渲染, 空体素跳跃