

## 基于视点的分块 LOD 大规模地形生成算法研究

原庆红, 韩 燮

(中北大学 电子与计算机科学技术学院, 山西 太原 030051)

**摘 要:** 针对虚拟战场环境仿真中构建大规模地形的速度问题, 介绍了一种基于视点和分块的 LOD 大规模地形生成算法, 能够快速地构建海量数据的地形模型。该算法首先对海量地形数据分块后以四叉树结构存储; 然后应用简单的圆锥视景体快速裁剪地形块, 丢弃与视景体相离的地形块, 对于与视景体相交地形块剪裁出位于视景体内的四叉树结点。根据视点 to 节点距离和地形本身的粗糙度, 选用合适的 LOD 渲染地形, 最后应用基于视点的方法消除裂缝。在数据相同情况下, 该算法比四叉树算法减少约 1/2 时间, 并且更逼真地生成了大规模数据地形。本算法也可用于逆向工程设计中重建物体等问题。

**关键词:** 四叉树; 视点; 视景体裁剪

**中图分类号:** TP39

**文献标识码:** A

**文章编号:** 1000-7180(2011)06-0041-05

## Block-based Viewpoint Generation Algorithm for Large Scale Terrain LOD

YUAN Qing-hong, HAN Xie

(College of Electronics and Computer Science and Technology, North University of China, Taiyuan 030051, China)

**Abstract:** Virtual battlefield environment simulation for construction of the speed of large-scale topography, have introduced a block based on viewpoint and the LOD terrain generation algorithm for large-scale, which can quickly build massive data terrain model. The algorithm first massive terrain data block after the quadtree storage. Then apply the simple cone visual terrain Rapid cutting block and apply the simple cone visual terrain rapid cutting block. Discarded and the visual bulk of the terrain from the block, the intersection of the terrain and visual body block cut out from the body in visual quadtree node. To the node depending on the view distance and the roughness of the terrain itself, choose the appropriate LOD terrain rendering, the final application of the method based on point of view to eliminate cracks. Experimental results show that the algorithm compared with Quadrees can be reduced by about 1/2 hours, and to generate large-scale high-fidelity terrain data can be used for reconstruction of objects in reverse engineering design problems.

**Key words:** quadtree; viewpoint; view frustum culling

### 1 引言

在虚拟战场环境仿真中, 地形绘制技术是其研究热点, 绘制的效率与逼真性在仿真技术中有着非常重要的影响。原始地形模型庞大的数据量已超过了系统的实时处理能力, 也增加了地形渲染的负担。为了解决上述问题, Duchaineau 提出 ROAM 算法, 即用等腰直角三角形来表示地形的每一小块, 把直

角顶点和斜边中点连接来拆分地形以达到需求的精度<sup>[1]</sup>。接着有不少学者在 ROAM 算法基础上利用四叉树进行研究和改进, 实现大规模地形的实时渲染。上述两种算法绘制地形时, 均直接对三角形或四叉树节点进行视景体裁剪, 虽然通过视景体裁剪机制很大程度上减少了绘制数量。但是该方法只局限于少量地形数据, 当在大规模地形数据建模的场合, 上述方法执行效率较低。

针对其缺点,本文提出了一种基于视点的分块 LOD 大规模地形构建方法.首先根据计算机内存页面大小,将大量地形数据尽量分成等大的地形块.然后在视景体裁剪阶段分为基于地形块剪裁和基于四叉树节点裁剪两个阶段,该算法中经过地形块裁剪,70%的地形块被剔除了.本算法和传统四叉树算法相比,最后送入图形设备绘制的三角形面片数是一样的,但该方法很大程度上减少了四叉树节点的计算量.

## 2 基于视点的 LOD 大规模地形生成算法

### 2.1 算法流程

大规模地形实时构建算法由数据预处理、模型构建和模型渲染三部分组成,算法流程如图 1 所示.数据预处理阶段的任务是将 DEM 高程数据按照四叉树结构进行有重叠的分块.模型构建阶段首先测试地形块是否位于视景体中,若地形块不在视景体内,则该块不需要再处理直接将其丢弃;若地形块与视景体相交,则用基于四叉树节点的裁剪进行局部细致裁剪,得到位于视景体内部的四叉树节点.将需要绘制的四叉树节点,根据视点到地形块的距离和地形粗糙度选择适合的 LOD.完全置于视景体内的块直接渲染地形.处理完所有地形块数据,把最终生成的视点相关的地形简化模型交由显像管道系统的硬件,完成地形简化模型的渲染.最后采用基于视点的方法消除裂缝.

### 2.2 DEM 数据分块

目前计算机硬件性能的限制,对于大规模地形数据不能一次性地装入内存,因此海量地形实时绘制算法通常对地形进行分块后,数据块动态装入内存,以此降低算法对计算机硬件性能的要求<sup>[2]</sup>.另外本文采用数据分块的方法,主要目的实现基于块的快速视景体裁剪,对于不在视景体中的数据块直接丢弃.从而大大降低了不必要的地形数据对内存的消耗,减少了计算量,提高了绘制速度.

本文将规则格网 DEM 作为地形的数据模型,根据四叉树节点的性质,将整个地形以相邻块边界数据重复地分成  $(2^n + 1) \times (2^n + 1)$  块大小.所分块大小尽量与内存页面大小一致,以降低数据块调度时的缺页中断次数.相邻块边界数据重复是为了最后合并小块地形时,各分块间连接处能自动保持二阶连续性.分块后的地形数据以文件形式存储在硬盘中,每块地形可根据需要建立金字塔数据结构.

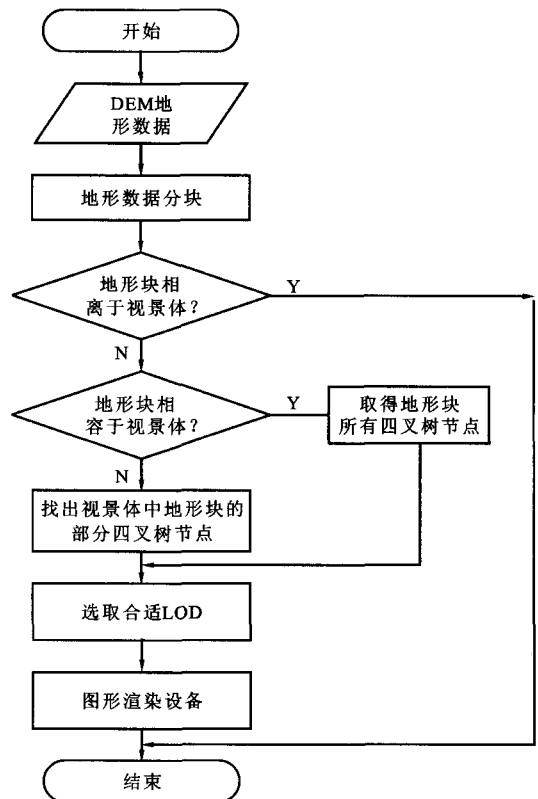


图 1 基于视点的分块 LOD 大规模地形生成算法流程图

### 2.3 四叉树结构的存储方式

将上节分块后的地形,采用基于四叉树的 LOD 简化算法对地形进行实时连续绘制.采用自顶向下的方式逐一对地形进行分割,如图 2 所示.首先将整个地形看作一个正方形称为根节点(如图 2 中  $C_0$ ),按四叉树结构对其进行递归分割,每次分割产生 4 个新的更高层次的节点(如图 2 中  $C_1 C_2 C_3 C_4$ ).这样一直分层次地递归分割地形,直至最高分辨率为止,就建立起了整个地形的四叉树结构,以高效地构建动态 LOD 模型.创建 LOD 时,只需根据 2.5 节中的评价因子来动态地从树中选取具有合适细节层次的节点来表达地形,从而建立起多分辨率地形<sup>[3]</sup>.

### 2.4 视景体裁剪

目前地形简化的裁剪运算,在提交绘制命令之前对三角形面片进行可见性判断,由于其节点数量巨大,裁剪运算约占总时间的 1/3,在大程度上影响了算法的效率,如 ROAM 算法.当节点不在当前视野范围内时,如果把这些节点全部参与绘制就会耗费资源<sup>[3]</sup>.另外传统视景体裁剪过程,是将顶点坐标代入视景体六个面的方程中计算,观察计算结果的正负号来决定顶点是否在视景体内.该方法涉及大量重复的计算,效率较低.

在 2.3 节中所得的地形块四叉树存储结构基础

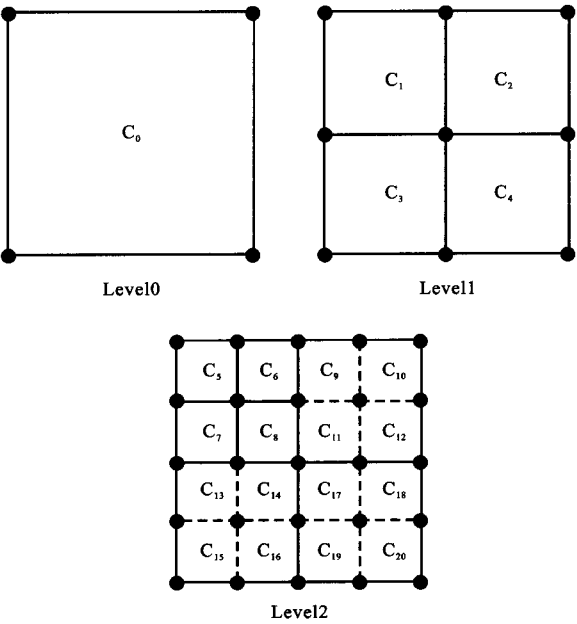


图 2 四叉树结构

上,本文考虑视点的影响,提出了基于视点的块整体裁剪和四叉树节点部分裁剪相结合的方法,以此提高裁剪效率.模型简化时进行两次视景体裁剪:首先是基于块的整体裁剪,去掉不在视景体内的地形块;其次对于和视景体相交的块,用基于四叉树节点的裁剪进行部分细致裁剪,得到位于视景体内部的四叉树节点.本节引进一种简单的视景体,将圆锥体加上远近裁剪面作为视景体,其侧面为等腰梯形,如图 3 所示.

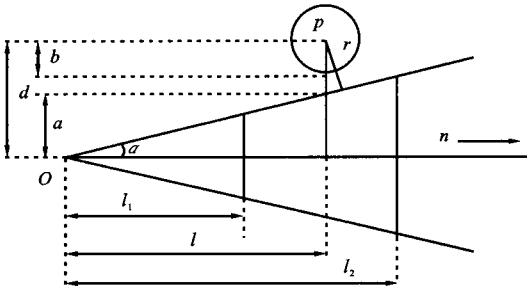


图 3 视景体裁剪

图 3 中  $P$  为子块包围球中心点,  $O$  为视点,  $\alpha$  为视角的一半,  $n$  表示视线方向的单位向量且  $\|n\| = 1$ ,  $l_1$  是近截面到视点的距离,  $l_2$  是远截面到视点的距离. 令  $l = (P - O) \cdot n$ .

若满足  $l < l_1 - r$  或  $l > l_2 + r$ , 则该子块不在视景体内, 否则计算:

$$d = \|(P - O) - l \cdot n\|$$

$a = l \cdot \tan \alpha$ ,  $b = r / \cos \alpha$  若满足  $d > a + b$  则认为该块不在视景体内;

若满足后面任一条件, 即

$\begin{cases} l > l_1 - r \text{ 且 } l < l_1 + r \\ l > l_2 - r \text{ 且 } l < l_2 + r \end{cases}$ , 则表明该子块部分在视景体内、部分在视景体外, 此时对该块进行基于四叉树节点的部分细致裁剪, 即针对每节点判断其是否在视景体内, 判断方法同地形块裁剪.

本算法中基于块的裁剪保证了裁剪的速度, 基于四叉树节点的裁剪保证了裁剪的精度. 相比传统视景体裁剪中每次计算视景体的六个平面方程, 很大程度上简少了计算量, 提高了算法效率.

2.5 地形简化的评价因子

上节完成视景体裁剪后, 需要对四叉树自底向上地进行网格划分, 根据视点 to 节点距离和地形本身的粗糙度, 确定该节点是否需要细分. 因为远处的景物看上去显得小且模糊, 可以采用较低细节层次表达地形; 近处的物体看上去清楚, 需要采用较高的细节层次来表达地形. 另外, 地形起伏不变的区域有较高的细节层次, 而平坦的地方不需要浪费过多的图元<sup>[4-5]</sup>.

将视点 to 地形块中心的距离当作视点 to 地形块的距离, 如图 4 所示, 采用空间距离计算公式得到  $D$ :  $D^2 = (e_x - c_x)^2 + (e_y - c_y)^2 + (e_z - c_z)^2$ . 其中地形块的中心点是  $(c_x, c_y, c_z)$ , 视点位置  $(e_x, e_y, e_z)$ .

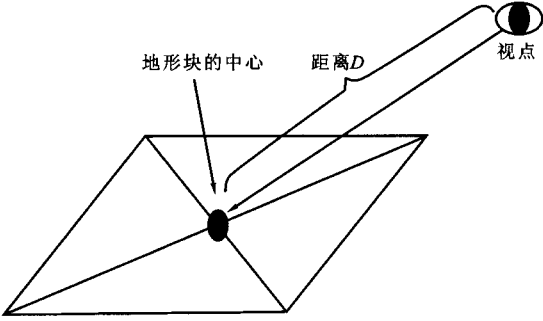


图 4 视点 to 地形块的距离

考虑一个节点包含的 9 个顶点, 其中心点和 4 个边点在节点被分割和不被分割时候会引起一定的误差, 这 6 个误差值为  $dh_1$  到  $dh_6$ , 如图 5 所示. 它们的数值越大表示这个节点代表的地形越粗糙. 记粗糙度评估值为  $d_2$ ,  $d_2 = \text{Max}(dh_1, \dots, dh_6) / d$ . 地形表面粗糙度计算:

$$f = \frac{l}{d \cdot C \cdot \max(k \cdot d_2, l)}$$

当  $f < 1$  的时候, 需要对节点进一步的细分.  $C$  是最小全局精度, 通过设置  $k$  的大小可达到一个恒定的帧率. 综合以上两个评价因子, 可以随着视点的移动实时调节地形各处的细节层次, 以保证图像交互性帧率<sup>[6]</sup>.

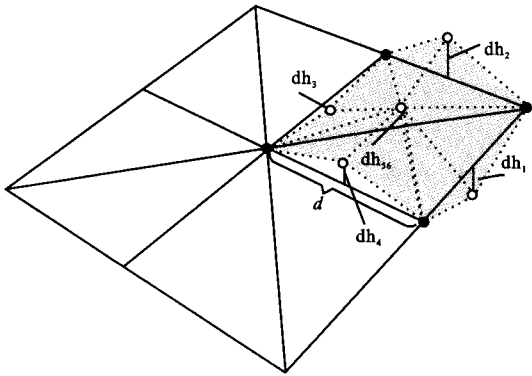


图 5 地形表面粗糙度计算

2.6 基于视点方法的消除裂缝

针对块内“裂缝”问题,不少学者研究采用限制相邻节点层次差不超过 1 的方法消除裂缝,且只适用于少量静态地形数据场合.但处理海量地形数据时,要不断地进行判断节点层次差计算量大耗费 CPU 时间长<sup>[7-8]</sup>.本节基于视点进行裂缝消除,从而避免了地形生成过程中出现 T 型裂缝.当视点远离节点中心时看到的细节较少,用低分辨率绘制图形,此时可以通过缩减边的方式实现裂缝消除,即删除因分割不合理而导致裂缝出现多余的边,并重新设置分割标志,如图 6 所示.在图 6(a) 中节点 A 与节点 O 的交界 E 和 F 处,会产生裂缝.将 A 及子节点 D 的分割标志改为 0,则可消除 E、F 处裂缝.至此裂缝完全被消除,如图 6(b) 所示.

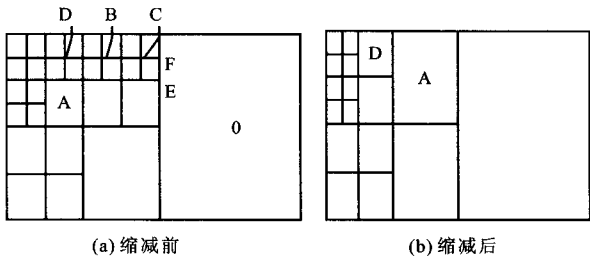


图 6 消除裂缝

当视点离节点中心距离较近时,能看到较多细节,通过分割边的方法以高分辨率绘制地形细节.在图 7(a) 中间菱形处出现了裂缝,只需在图 7(b) 中箭头所指裂缝处增加一条边即可消除裂缝,综上,从缩减和分割两个方向最终达到消除裂缝的效果,实现了地形不同细节层次间平滑无缝的过渡.

3 实验结果及分析

实验采用 VC++ 6.0、OpenGL 开发包,对 6 000×6 000 的某地形数据在 256×256 分辨率环境下进行实时构建渲染实验,截取部分地形效果如下图所示.图 8 运用四叉树裁剪算法生成的地形图.

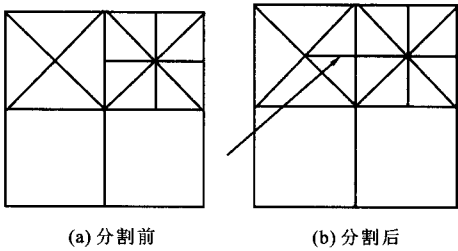


图 7 消除裂缝

图 9 则利用本文算法构建的地形图.

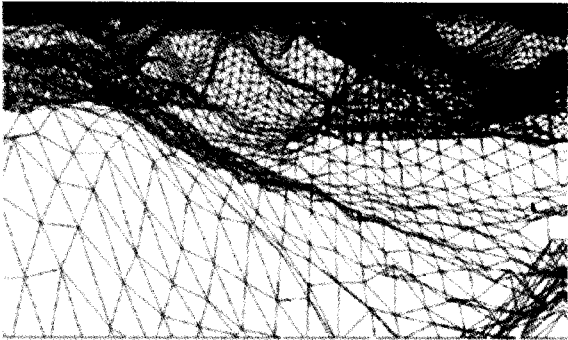


图 8 传统算法生成地形图

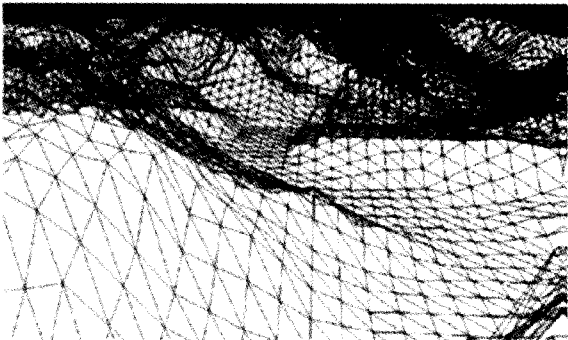


图 9 本文算法生成地形图

表 1 是四叉树算法和本文算法在相同分辨率情况下,运行时间的对比.可以看出,最后生成地形图中三角面片数是一样的,但是本文算法能节省将近 1/2 的时间.另外图 9 地形图在地形崎岖变化的地方,表现得清楚,比图 8 更能逼真表现地形的变化情况.

表 1 两种方法的情况对比

| 方法类别  | 运行时间/s | 三角面片数 |
|-------|--------|-------|
| 四叉树算法 | 0.91   | 3 150 |
| 本文方法  | 0.53   | 3 150 |

4 结束语

本文提出基于视点的分块四叉树 LOD 生成算法,首先将大规模规则格网 DEM 按照四叉树结构分块存储,然后采用本文提出的简化视景体对地形

块进行视景体裁剪,裁剪掉不在视景体内的地形块。对于部分在视景体内、部分在视景体外的地形块,裁剪出该块中位于视景体的四叉树节点。接着遵循观点到地形块中点的距离和地形块本身的粗糙度两个指标,选择合适的 LOD 进行地形渲染。在最后的的分块裂缝过程中,采用了基于视点的裂缝消除方法。本算法生成地形速度快,相比于传统的四叉树裁剪算法减少将近一半的时间。绘制效果比四叉树算法更逼真。

今后的工作方向:需要对规则格网 DEM 地形数据存储方式、树结构组织方式等继续分析研究,选择出更适合的组织、效率更高的方式来实现可视化,最大限度地降低算法的时间和空间开销。

#### 参考文献:

- [1] Mark Duchaineau, Murray Wolinsky, David E Sigetiet al. ROAMing terrain: real-time optimally adapting meshes [C]// Proceedings of the IEEE Visualization'97, Phoenix, AZ, USA; IEEE, 1997; 81-88.
- [2] 吴有用, 万旺根, 金龙存, 等. 一种新的连续性 LOD 实现算法[J]. 微电子学与计算机, 2010, 27(6): 185-187.
- [3] 刘燕平, 余雪丽, 贾晓峰. 模型和视点相关性的多分辨率模型的渲染[J]. 微电子学与计算机, 2010, 27(1): 65

-69.

- [4] 许承东, 李强, 赵向领. 一种基于视点的大规模地形生成算法[J]. 北京交通大学学报, 2007(1): 69-72.
- [5] 张淑军, 陈芳, 周忠. 基于四叉树部分的 LOD 地形绘制算法[J]. 系统仿真学报, 2008, 9(20): 25-29.
- [6] Jin Hailianga, Lu Xiaopinga, Liu Huijiea. View-dependent fast real-time generating algorithm for large-scale terrain[J]. Procedia Earth and Planetary Science, 2009, 1(6): 1147-1151.
- [7] Rafael Rodriguez a, Eva Cerezo b, Sandra Baldassarri b, et al. New approaches to culling and LOD methods for scenes with multiple virtual actors [J]. Computers & Graphics, 2010, 7(10): 1-13
- [8] Liu a S Q, Ong b S K, Chen a Y P, et al. Real-time dynamic level-of-detail management for three-axis NC milling simulation [J]. Computer-Aided Design, 2006, 11(38): 378-391.

#### 作者简介:

原庆红 女, (1984-), 硕士研究生. 研究方向为虚拟现实、虚拟战场环境仿真中构建三维地形。

韩 雯 女, (1964-), 博士, 硕士生导师. 研究方向为图形图像处理, 虚拟现实。

(上接第 40 页)

#### 参考文献:

- [1] 郝莹, 钱丽萍, 田芳, 等. 基于特征检测的人眼定位方法[J]. 微电子学与计算机, 2008, 25(2): 50-53.
- [2] Kyrki V, Kamarainen J, Heikki K. Simple gabor feature space for invariant object recognition[J]. Pattern Recognition Letters, 2004, 25(3): 311-318.
- [3] Viola P, Jones M. Rapid object detection using a Boosted cascade of simple features [C]// Proceeding IEEE Conference on Computer Vision and Pattern Recognition. Kauai, HI, USA; IEEE Computer Society, 2001: 511-518.
- [4] Niu Zhiheng, Shan Shiguang, Yan Shengye, et al. 2D cascaded Adaboost for eye localization [C]// International Conference on Pattern Recognition. Hong Kong, CHINA; IEEE Computer Society, 2006: 1216-1219.
- [5] Ahonen T, Hadid A, Pietikainen M. Face description with local binary patterns: application to face recognition [J]. IEEE Transactions on Pattern analysis and

Machine Intelligence. 2006, 28(12): 2037-2041.

- [6] Kroon B, Boughorbel S, Hanjalic A. Accurate eye localization in low and standard definition content with application to face matching [J]. Computer Vision and Image Understanding, 2009, 113(8): 921-933.
- [7] Fukui K, Yamaguchi O. Facial feature point extraction method based on combination of shape extraction and pattern matching [J]. Transactions of the Institute of Electronics, Information and Communication Engineers, 1997(8): 2170-2177.
- [8] BioID faces database [EB/OL]. [2010-09-13]. <http://www.bioid.com/downloads/facedb/index.php/>.

#### 作者简介:

邓 亮 男, (1985-), 硕士研究生. 研究方向为生物特征识别。

张新曼 女, (1973-), 博士, 副教授. 研究方向为机器视觉、图像融合等。