第 35 卷 第 2 期 计 算 机 工 程 2009 年 1 月
Vol.35 No.2 Computer Engineering January 2009
·开发研究与设计技术· 文章编号：1000—3428(2009)02—0268—03 文献标识码：A 中图分类号：TN911.73

基于三角形二叉树的实时大规模地形渲染算法

林继承，万旺根，周俊玮，谭小辉
(上海大学通信与信息工程学院，上海 200072)

摘 要：提出一种大规模地形渲染算法，对大规模地形进行分块，用三角形二叉树表示地形网格，在实时漫游中，通过强制分割和强制合并实时更新网格，充分利用帧与帧之间的连贯性并自动避免裂缝。实验结果表明，该算法可以有效提高 ROAM 算法性能，能以较高帧速实现大规模地形的实时漫游。

关键词：地形渲染；ROAM 算法；三角形二叉树；强制分割；强制合并

Real-time Algorithm for Large-scale Terrain Rendering Based on Triangle Bintree

LIN Ji-cheng, WAN Wang-gen, ZHOU Jun-wei, TAN Xiao-hui
(School of Communication and Information Engineering, Shanghai University, Shanghai 200072)

【Abstract】 This paper presents an algorithm for large-scale terrain rendering. It splits the terrain into several patches, and uses triangle bintrees to represent terrain meshes. At run time, it updates terrain meshes by forced split and forced merge operations, which can take the advantage of frame-to-frame coherency and automatically avoid cracks. Experimental result shows that this algorithm can efficiently improve the performance of ROAM algorithm and can be used in real-time walkthroughs in large-scale terrain at high frame rates.

【Key words】 terrain rendering; ROAM algorithm; triangle bintree; forced split; forced merge

1 概述

大规模地形的实时渲染在虚拟现实、地理信息系统、飞行模拟、城市规划和三维游戏等领域中被广泛应用，是计算机图形学的研究热点和难点。在大规模地形中，三角形数量极大，高性能图形加速卡不能满足大规模三维场景实时渲染的要求。因此，研究者提出各种方法来降低场景复杂度，其中较有效的是层次细节(Lever Of Detail, LOD)技术，它能在不影响画面视觉效果的前提下提高渲染效率。

自 1996 年以来，LOD技术在实时地形渲染中得到了广泛应用[1-4]。在各种地形LOD算法中，ROAM算法可以实时、动态、高效地简化模型。本文基于ROAM算法，提出一种基于视点的高效实时大规模地形渲染算法。针对此算法提出一种简单有效的误差度量标准，并通过视域剔除提高算法效率。

2 相关工作

近十年来，国内外学者提出了很多地形渲染算法。根据生成的网格，可以将地形算法分为 2 类，即基于不规则三角网(Triangulated Irregular Network, TIN)的算法和基于规则网格的算法。

基于不规则三角网的算法所生成网格的三角形数目较少，但数据结构复杂，难以生成和裁减。其中，具有代表性的是文献[1]提出的基于视点的渐进网格(View-Dependent Progressive Mesh, VDPM)算法。该算法能产生较好的视觉效果，但占用内存大且效率不高。

基于规则网格的算法所生成网格的三角形数目较多，但适合用四叉树或二叉树进行组织，易于裁剪和简化，便于对动态地形的处理。因此，多数地形渲染算法都基于规则网格。基于规则网格的算法可分为四叉树算法和二叉树算法 2 类。四叉树的实施简单、有效，较具代表性的是文献[2]提出的实时连续 LOD 算法。文献[3]提出一种基于限制四叉树的大规模地形渲染算法。二叉树算法中较具代表性的是文献[4]提出的 ROAM 算法。ROAM 算法采用三角形二叉树表示地形网格，通过强制分割保证每一步操作均不会产生裂缝。它用分割和合并队列控制三角网格分割的精度，能有效利用帧与帧之间的连贯性，但在实时渲染过程中，每次优先级队列的更新会消耗大量 CPU 时间。在 ROAM 的基础上，文献[5]、文献[6]提出了各自的改进算法。由于 ROAM 算法简单高效且易于扩展，因此在地形渲染中得到广泛应用。

3 地形的三角形网格表示

图 1 给出了一个经过 3 次分割得到的三角形二叉树。

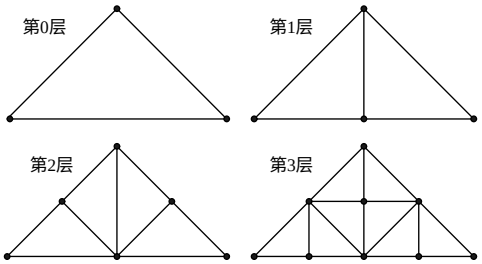

图 1 3 层三角形二叉树

基于 ROAM 算法思想，本文用三角形二叉树表示地形网

基金项目：国家“863”计划基金资助项目(2007AA01Z319)；信息产业部电子信息产业发展基金资助项目(2005688)
作者简介：林继承(1983－)，男，硕士研究生，主研方向：互动数字娱乐；万旺根，教授、博士生导师；周俊玮、谭小辉，硕士研究生
收稿日期：2008-07-23 **E-mail：**jachenangel@yahoo.com.cn

格。等腰直角三角形是三角形二叉树的基本数据单元。从直角顶点作垂线将三角形分为 2 个等腰直角三角形的过程即分割，此操作的逆过程即合并。ROAM 算法主要是对三角形进行分割和合并的操作，通过不同递归分割和合并操作，可以构成具有不同细节程度的地形网格。

三角形二叉树中每个三角形 T 都有 3 个邻接三角形，如图 2 所示。TB 是 T 的底邻居三角形，共享 T 的底边。TL 是 T 的左邻居三角形，共享 T 的左边。TR 是 T 的右邻居三角形，共享 T 的右边。三角形 T 的左、右邻居三角形只能与 T 处于同一个二叉树层级 l 或者处于 T 的下一层级 $l+1$。底邻居三角形只能与 T 处于同一个二叉树层级 l 或处于 T 的上一层级 $l-1$。

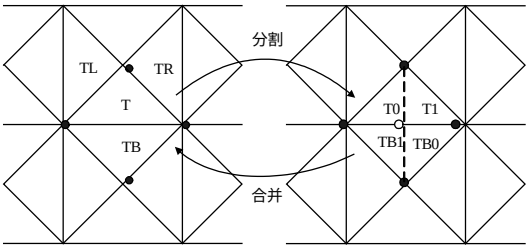

图 2　三角形 T 的邻接关系以及分割与合并操作

图 2 描述了三角形二叉树的分割和合并操作。当三角形 T 与其底邻居三角形 TB 处于同一层级时，T 和 TB 组成一个菱形。当 T 分割成 T0 和 T1 时，为了避免裂缝，TB 需要分割成 TB0 和 TB1。反之，当 T0 和 T1 及 TB0 和 TB1 都没有子节点时，可以将这 2 对三角形合并，得到 T 和 TB，称 T 和 TB 构成一个可合并的菱形。

当三角形 T 的斜边处于边界时，可以直接进行分割。当三角形 T 的底邻居三角形 TB 处于 T 的上一层级 $l-1$ 时，如果要分割三角形 T，则先要对三角形 TB 进行强制分割。对 TB 进行分割可能引起一系列递归分割。图 3 描述了一个强制分割过程，左边为分割前的三角形网格，对 TB 进行强制分割，得到右边的三角形网格。使用强制分割可以避免裂缝。

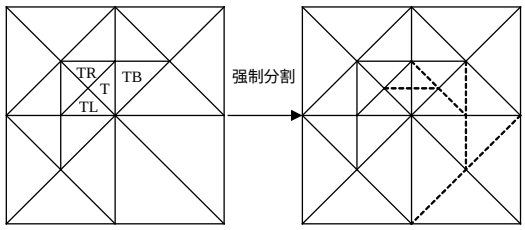

图 3　三角形 T 的强制分割

用三角形二叉树构造网格可以容易地实现地形网格模型层次细节的动态变化，且有利于动态地构造地形的连续层次细节模型。

4　误差度量的确定

在地形漫游时，要根据一定误差度量进行分割和合并。文献[4]提出基于嵌套的世界空间范围的误差，虽然较精确，但计算速度很低。本文在文献[5]的基础上提出一种简单有效的误差度量方法。用几何空间误差 *Variance* 描述地形粗糙程度，其值通过该点的插值高度与其在高度图中实际高度的差来计算，过程较简单。每个节点的几何空间误差 *Variance* 不变，可以事先计算。本文用直角三角形的斜边中点与视点的距离表示视点与节点的距离 *Distance*。上述方法计算方便，极大提高了渲染效率。得出的误差度量公式为

$$Error = \frac{k \times Variance \times LandWidth}{Distance}$$

其中，k 是影响因子；*LandWidth* 是整个地形的宽度。该误差度量考虑了地表粗糙程度和视点与节点的距离，且便于计算。

5　算法的具体实现

5.1　数据加载和地形分块

进行地形渲染时，先要将地形数据载入内存。地形数据的加载很简单，在内存中开辟一定空间后，从高度图中读取相应高度数据并存入内存即可。

大规模地形的数据量较大，如果将整个地形作为一块来处理，将导致三角形二叉树层数较深，影响 LOD 模型的构建效率。因此，本文改进 ROAM 算法，对地形进行分块。加载地形数据后，将规模较大的地形分成较小的块，并为每个子块分别建立相应三角形二叉树。相当于将一棵深度很深的三角形二叉树分成若干深度较浅的三角形二叉树，从而提高 LOD 模型的构建效率。对地形进行分块还有利于纹理映射和视域裁剪，便于将本算法扩展到对超大规模地形的渲染。

地形分块方法如图 4 所示。其中，Terrain 代表整个地形，由多个大小相同、按空间位置排列的 Patch 组成。每个 Patch 由左右 2 个等腰直角三角形构成。这 2 个三角形分别对应一棵三角形二叉树的根节点。对于 ROAM 算法，Patch 的大小必须是 $(2^n+1)\times(2^n+1)$，且相邻 Patch 之间的边缘有重复数据。

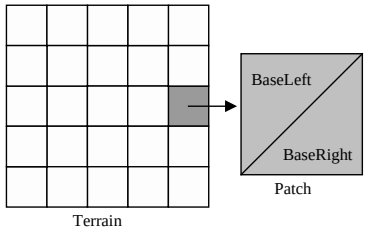

图 4　地形分块

5.2　视域剔除

对大规模地形进行视域剔除可以明显提高地形渲染速度。针对地形漫游，本文简化视锥体，只保留视锥体的远裁剪平面和左、右裁剪平面。只要将视锥体投影到 *x*-*z* 平面，就可以判断 Patch 的可见性，如图 5 所示，其中，三角形表示视锥体在 *x*-*z* 平面上的投影，灰色方格表示可见的数据块。

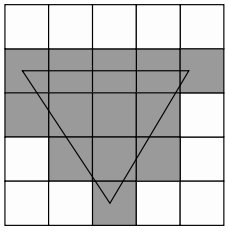

图 5　视域剔除

5.3　三角形二叉树的实时构网

ROAM算法[4]采用 2 个优先级队列，即分割队列和合并队列来控制三角形的分割和合并操作。它能利用帧与帧之间的连贯性，但由于优先级队列的更新将消耗大量CPU时间，因此会严重影响渲染效率。本文借鉴文献[6]提出的三角形合并算法思想，对分割和合并操作进行改进，采用强制分割和强制合并对网格进行实时更新。改进后的算法不需要优先级队列，能利用帧与帧之间的连贯性，且极大节省了CPU时间。

5.3.1　分割算法

分割算法能分割任意三角形，它通过强制分割解决裂缝

问题，其实现过程如下：

输入：三角形二叉树中一个没有子节点的三角形 T

(1)如果 T 的底邻居存在，且 T 底邻居的底邻居不是 T，则对 T 的底邻居执行分割操作。

(2)生成 T 的 2 个子节点。

(3)设置 T 的 2 个子节点的邻居节点和父节点。

(4)调整 T 的邻居节点和 T 邻居节点父节点的邻居节点。

5.3.2　合并算法

在 ROAM 算法中，只有当 2 个三角形构成一个可合并的菱形时，才能将其合并，该算法灵活性不高。本文采用强制合并方法对 ROAM 中的合并算法进行改进。为了保持网格的连续性并避免裂缝，所有邻居三角形必须进行合并。图 6 描述了一个强制合并过程。需要对左图中的三角形 T 进行合并，经过递归操作得到右图中的结果。可以看到，三角形 T 的所有子孙节点都被合并，且三角形 T 的所有邻居节点也被强制合并。

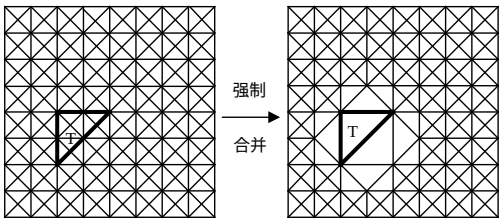

图 6　强制合并的过程

由于采用强制合并无须记录菱形和可合并菱形的信息，因此速度更快。采用强制合并与强制分割相结合，不用引入 2 个优先级队列，节省了优先级计算，提高了算法效率。

合并算法的实现过程如下：

输入：一个具有子节点的三角形 T

(1)将 T 的子节点置为空。

(2)如果 T 的左、右邻居 TL 和 TR 与 T 处于同一层次，或处于 T 的下一层次，则调整它们的邻居节点，否则对 TL 和 TR 执行合并操作，并调整 TL 和 TR 邻居节点的邻居节点和邻居节点父节点的邻居节点。

(3)如果 T 的底邻居有子节点，则对 T 的底邻居执行合并操作。

5.3.3　三角形网格的构造流程

在渲染时，相邻帧视点位置变化不大，帧和帧之间存在很大关联性。在第 1 帧渲染完成后，后继帧只要在上一帧所构造网格的基础上，通过合并和分割操作进行少量调整，从而避免重复构造网格，加快网格构造速度。

三角形网格构造流程的伪代码如下：

```
for 每一帧
   判断每个 patch 的可见性；
for 每一个可见的 patch
if patch 没有分割过 then
用 split 方法构造网格
      else 利用上一帧构造的网格，用 split 和 merge 进行调整
      end if
end for
end for
```

6　实验与结果分析

本文在 VC 6.0 环境下结合 OpenGL 实现本算法。硬件环境为 P4 2.4 GHz 的 CPU，内存大小为 512 MB，显卡为 ATI9550，地形数据为 1 025×1 025 个高程点的高度图。将地形分成 16×16 个 Patch，每个 Patch 的大小为 65×65 个高程点。在此硬件平台上，本文算法取得了良好的渲染效果，FPS 较高。渲染效果如图 7 所示。

图 7　渲染效果

比较本算法和传统算法，结果如表 1 所示。可以看出，本文算法性能明显高于经典 ROAM 算法。

表 1　2 种算法的比较

算法	地形大小/高程点数	三角形数目	帧速/fps
经典 ROAM 算法	1 025×1 025	8 925	58
本文算法	1 025×1 025	8 766	87

对只有分割操作的算法与分割和合并操作相结合的算法进行比较，结果如表 2 所示。可以看出，将合并操作与分割操作相结合后，帧速得到了很大提高。可见，本文算法效率比经典 ROAM 算法高。

表 2　合并操作性能

三角形数目	只有分割操作的算法帧速/fps	分割与合并相结合的算法帧速/fps
8 765	42	87
7 656	49	92
6 360	56	98

7　结束语

本文将 LOD、地形分块和视域剔除相结合，提出一种高效的大规模地形渲染算法。此算法具有以下优点：(1)将地形分块，便于数据处理以及将算法扩展到超大规模地形；(2)误差度量简单有效；(3)将强制分割和强制合并相结合，以充分利用帧与帧之间的连贯性并自动避免裂缝；(4)通过视域剔除有效提高渲染效率。实验结果表明，本文算法能实现大规模地形的实时漫游，且具有较高帧率，其性能较 ROAM 得到了很大提高。笔者将进一步研究如何通过动态载入地形数据实现超大规模地形的实时漫游。

参考文献

[1] Hoppe H. Smooth View-dependent Level-of-detail Control and Its Application to Terrain Rendering[C]//Proc. of IEEE Visualization Conference. [S. l.]: IEEE Computer Society Press, 1998: 35-42.

[2] Lindstrom P, Pascucci V. Visualization of Large Terrains Made Easy[C]//Proceedings of IEEE Conference on Visualization. San Diego, California: IEEE Computer Society, 2001: 363-370.

[3] Bao Xiaohong, Pajarola R, Shafae M. Smart: An Efficient Technique for Massive Terrain Visualization from Out-of-core[C]// Proceedings of Vision, Modeling and Visualization Conference. Stanford, USA: Aka GmbH, 2004: 413-420.

[4] Duchaineauy M, Wolinsky M. ROA Ming Terrain: Real-time Optimally Adapting Meshes[C]//Proc. of IEEE Visualization'97 Conference. [S. l.]: IEEE Computer Society Press, 1997: 81-88.

[5] Turner B. Real-time Dynamic Level of Detail Terrain Rendering with ROAM[EB/OL]. (2000-04-03). http://www.gamasutra.com/features/20000403/turner_01.htm.

[6] Ogren A. Continuous Level of Detail in Real-time Terrain Rendering[D]. Umea, Sweden: Sweden Umea University, 2000.