第34卷 第13期 计 算 机 工 程 2008年7月
Vol.34 No.13 Computer Engineering July 2008
·开发研究与设计技术· 文章编号：1000—3428(2008)13—0270—03 文献标识码：A 中图分类号：TP391

大规模地形实时可视化算法

韩 敏，汤松涛，李 洋
(大连理工大学电子与信息工程学院，大连 116023)

摘 要：提出一种实现大规模三维地形可视化的算法。该算法主要通过地形数据分块和多线程数据动态调度技术实现海量地形数据的实时动态调度，利用四叉树结构来简化地形网格，通过引入Y型基元的方法消除边界裂缝。实验结果表明，采用该方法可以实时生成大规模地形，提高地形漫游的效率与可视化效果。

关键词：地形可视化；动态调度；四叉树；裂缝消除

Real-time Visualization Algorithm of Large-scale Terrain

HAN Min, TANG Song-tao, LI Yang
(School of Electronic and Information Engineering, Dalian University of Technology, Dalian 116023)

【Abstract】 In this paper, an algorithm is proposed to implement the large-scale 3D terrain visualization. Dynamic schedule massive data in real time is mainly based on the terrain data block partition and multithreading dynamic scheduling technique. Quad-tree data structure is used to simplify terrain meshes and cracks are eliminated by importing Y model primitive. The experimental results demonstrate that the proposed method can render large-scale terrain in real time, improve the efficiency and visual effects of terrain walkthrough.

【Key words】 terrain visualization; dynamic scheduling; quad-tree; crack elimination

地形可视化是以数字高程模型(DEM)或数字地面模型(DTM)为研究对象，采用一定的工具和算法对其简化、显示和仿真等的一门技术。目前，地形可视化已广泛应用于地理信息系统(GIS)、战场环境仿真和土地管理与利用等领域。大规模地形场景可视化是三维GIS研究中的一个重要问题。由于大规模地形的数据远远超出了当前计算机硬件的数据处理能力，因此在建模和实时显示2个方面都有较高的要求，这样就需要相应地解决地形数据庞大的方法。

1 相关研究

针对以上提出的问题近年来国内外许多专家从不同方面进行过研究。目前主要可以通过以下2个方面来解决大规模地形的可视化：

(1)通过地形数据简化算法减少绘制的三角形数量。文献[1]首先提出一种基于四叉树的规则格网数据LOD模型绘制算法；文献[2]提出实时优化自适应性网格ROAM算法，该算法基于二叉树结构并应用于规则格网数据的简化；文献[3]将其视点相关递进网格(VDPM)应用到地形绘制中，实时生成不规则三角形网格。

(2)可以通过对地形数据进行分块处理减少系统一次性处理的数据量，为了充分利用系统资源提高地形绘制速度，在数据处理的同时进行数据的动态预调度。文献[4-5]对海量数据的管理和调度进行了研究，使用了数据分块和动态调用方法，其中有的学者还综合考虑了数据动态调度和地形简化2个方面技术。

在地形数据简化算法中，对规则地形的简化主要是构建顶点的四叉树或二叉树结构来实现模型的多细节层次提取，其中四叉树的应用更为广泛。在应用树结构对地形进行多分辨率表示时，会出现裂缝现象。解决四叉树层次结构模型中的裂缝问题是利用四叉树进行动态构网的关键。

消除裂缝的方法主要可以分为2类：(1)将造成裂缝现象的低分辨率节点作适当分裂，使相邻节点具有相同的边界。最常用的方法是构建限制四叉树，但是限制四叉树块间的分辨率差别最多不能超过一层，网络结构不紧凑；在出现裂缝的边界处，通过填补三角面来消除裂缝的方法[6]，会导致模型表面的不连续，并可能出现狭长三角形。(2)对高分辨率节点处理。将较高分辨率节点的边界顶点移到相邻较低分辨率节点的方法[7]会产生T型连接，在T型点处产生光照的不连续；垂直边缘法(vertical skirt)破坏了网格的规则特征，容易造成接缝处陡峭。本文主要通过引入Y型基元的方法消除边界出现的裂缝现象。这种方法不但可以快速消除裂缝，还能保持模型表面的连续性。

2 研究方法

通过对地形数据分块和多线程数据动态调度技术，减少系统一次性处理的数据量。然后应用基于四叉树结构的简化算法对调入内存中的地形数据进行合理简化。

2.1 地形数据分块和动态调度

2.1.1 地形数据分块

针对大规模地形场景中数据庞大的问题，本文采用分块策略，对数据分而治之来减少系统一次性处理的数据量。

使用平均分块法对地形数据进行处理。这样，各个子块的细分过程只要用相同的程序模块就能完成。另外，在采用

基金项目：国家“973”计划基金资助项目(2006CB403405)；国家科技支撑计划基金资助项目(2006BAB14B05)
作者简介：韩 敏(1959－)，女，教授，主研方向：3S系统，神经网络及混沌序列分析；汤松涛、李 洋，硕士研究生
收稿日期：2007-07-17 **E-mail：**minhan@dlut.edu.cn

四叉树来构建地形网格时，相邻的块间边缘需有重复数据，因此每块的大小都必须是$(2^n+1)\times(2^n+1)$。大规模地形对应的纹理数据也很庞大，同样采用平均分块法，将纹理分割为与地形具有相同的子块数。最后将地形子块与相对应的纹理子块保存在同一文件中便于管理和调度。

2.1.2　简化视景体投影剪裁方法

视景体是一个满足透视投影原理的四棱锥体，用来模拟人眼的视觉系统。视景体内的地形即是计算机三维场景的可见区域。一般的视景体剪裁需要检测视景体的 6 个平面，考虑到在地形漫游中的实际情况[7]：视点往往比较低，地形面大部分都在视景体的上、下平面之间，并且视景体的近平面往往离视点很近，所以可以不考虑上、下平面和近平面，从而简化视景体。将简化的视景体投影到*X*-*Y*平面上，根据投影区域计算判断地形块的可见性。

2.1.3　数据动态调度策略

在大规模三维地形场景漫游时，只有在视景体内的才可见。由于数据量太大，因此不可能将全部数据调入内存，为了保证不影响漫游效果就需要在数据缓存中有下步可能要显示的数据。解决这个问题可以采用数据动态调度的方法，在显示视域中场景的同时将下步要显示的数据调入内存。根据这种策略，本文将场景中的数据划分为可见、预可见和不可见数据。为了避免在地形漫游时实时的数据调度影响漫游的效果，预可见区域的计算和预可见数据的调度成了关键所在。在三维场景漫游时，视点的运动趋势是随机的，因此预可见区域的选择要包括不同运动趋势可能显示的范围。预可见区域如果太大或太小都会影响到漫游的连贯性和实时性。本文采用简单扩展可见区域范围的方法，将可见区域中数据块的相邻不可见块设置为预可见区域，这样既包含了不同运动趋势的预可见区域，又避免了复杂的计算。

采用多线程的方式，使数据的动态调度和地形实时绘制同时进行。本文设计了 2 个线程：主线程主要是绘制渲染三维地形，并在地形漫游时启动数据调度线程；数据调度线程主要是完成预可见区域的计算和预可见数据的调度，保证内存中总是有下步可能显示的数据。

主线程方法描述如下：

(1)初始化三维地形场景，读入初始可见地形数据。

(2)开始地形漫游，同时启动数据调度线程。

(3)判断可见数据是否全部在内存中，如果没有则停止当前数据调度线程，将立即调用标志置 1，重新启动数据调度线程。否则直接执行下一步。

(4)刷新地形场景，若可见区域改变则启动数据调度线程。

(5)判断是否停止漫游，如果没有则返回到第(3)步。

(6)结束。

数据调度线程方法描述如下：

(1)判断立即调用标志是否为 1，如果不是则转至第(3)步。

(2)调入没在内存中的可见数据。

(3)计算当前预可见数据。

(4)调入预可见数据到数据缓存。

(5)删除数据缓存中不可见数据。

(6)结束。

2.2　基于四叉树结构的地形网格生成方法

2.2.1　建立地形数据块对应状态位表

四叉树的快速访问机制是由于在四叉树遍历过程中大量使用了递归的算法，因此，四叉树的存储形式将直接影响构模的速度。

本文采用一种有效的方法就是建立一个对应地形数据块的状态位表。用字节中的一位来表示对应顶点是否参与了构网，减少了系统内存的开销。比如一个 17×17 的地形块，只需要(17/8+1)×17 B 大小的二维数组。在位表中用 1 表示对应的顶点被激活，即该点参与了构建地形网格；0 表示对应顶点未激活，也就是没有参与构建地形网格。

在对地形块进行从上至下的四叉树分割中，首先将最顶层节点的 4 个角点对应在位表中的位激活，然后如果该节点满足分割要求，就将该节点的 4 个边中点和中心点位激活，再以同样的方法递归地判断节点的 4 个子节点。这样整个地形四叉树分割完成后，就构建出了一个地形四叉树对应状态位表。

图 1(a)给出了节点中各顶点的定义，其中有节点的角点(顶点 *a*, *b*, *c*, *d*)、中心点(顶点 *e*)和边中点(顶点 *f*, *g*, *h*, *k*)。单独由 4 个角点构成一个叶节点，边中点和中心点属于同层的顶点。

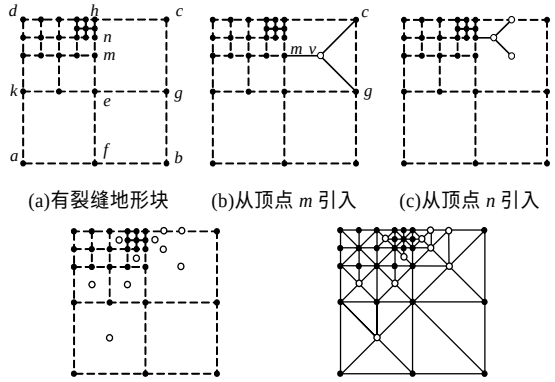


图 1　裂缝的消除

2.2.2　引入 Y 型基元的裂缝消除

裂缝是由于相邻节点的分辨率不一致所引起的。归根到底，裂缝是因为在节点的接边处，相邻节点参与构网的顶点数不同而造成的。因此，使相邻节点采用相同的边界顶点，就能消除裂缝。本文通过引入 Y 型基元的方法使相邻节点具有相同的边界顶点，从而消除边界裂缝。

Y 型基元的构成如图 1(b)所示。在出现裂缝的边界上，找到参与不同节点构网的顶点 *m*，从顶点 *m* 开始连接低分辨率节点中其同层的中心顶点 *v*，然后从顶点 *v* 分别连接该节点的 2 个角点 *c*, *g*。这样如图 1(b)中 4 个顶点连接的实线就构成了 Y 型基元。

在形成边界裂缝的节点处，对低分辨率的节点进行处理。从有裂缝边界上的最低层次顶点判断开始，如果该顶点没有参与该节点构网，就在该顶点引入 Y 型基元，只需要把构成该 Y 型基元的 4 个顶点中未激活的顶点激活。然后依次判断该边界上更高层次的顶点情况，直到该边界上没有更高一层的顶点为止。按照这个方法同样判断处理其他出现裂缝的边界。图 1(a)~图 1(d)表示了为消除裂缝，通过引入 Y 型基元的方法增加的新顶点。

2.2.3　实时构建地形网格

利用四叉树分割和引入 Y 型基元消除裂缝过程中建立的状态位表信息，使用递归算法实时构建出无裂缝地形网格，

如图 1(e)所示。

下面给出了绘制地形网格的主要伪代码，本文节点的绘制均采用 GL_TRIANGLE_FAN 模式。

地形网格绘制：

```
RenderSence( )
{
If 该节点 4 个边中点已激活
RenderSence 递归绘制其 4 个子节点；
Else     RenderNode 绘制该节点；
}
```

节点绘制：

```
RenderNode( )
{
If 该节点中有高于其两层顶点被激活
{ RenderNode 递归绘制激活顶点所属子节点；
剖去被激活顶点所属子节点部分，从中心点开始式绘制该节点剩余部分；}
Else
If 该节点中有高于其一层顶点被激活；
增加被激活顶点，从中心点开始绘制该节点
Else     直接绘制该节点；
}
```

经过地形四叉树分割和裂缝修补后，四叉树中节点的绘制只能为图 2 中所列的 8 种基本方式之一，其他方式均为基本方式的 *n* 倍 90 ° 旋转。

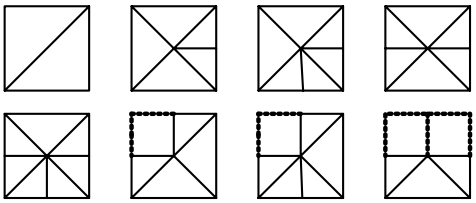
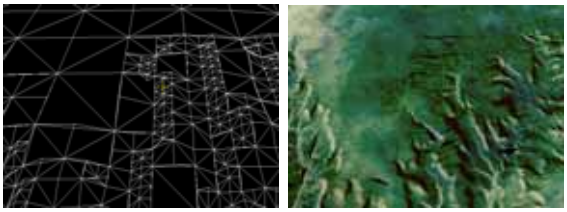


图 2　四叉树中节点的基本绘制方式

3　实验结果

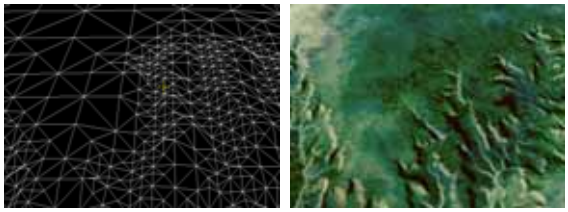
综合数据动态调用和实时生成多分辨地形网格两方面的技术，实现了一个大规模地形三维可视化系统。实验数据采用了某地形区域的 DEM 数据，4 097×2 049 个采样点。使用平均法将地形分为 32×16 块，每块为 129×129 个采样点。微机配置为：PIV1.6 GHz；512 MB 内存，NVIDIA Geforce2 显卡；软件环境为：WindowsXP, Visual C++ 6.0 和 OpenGL。

图 3 和图 4 给出了在某一视点时地形场景消除裂缝前后的对比效果图。图 5 给出了一段测试时间下渲染三角形数和对应绘制帧速率。可以看出生成的地形网格连续，消除了地形裂缝。在绘制 6 048～14 462 面三角形时，帧速率在 18 fps～39 fps 之间，获得较好的漫游效果，满足了实时显示的要求。



(a)有裂缝的地形网格　　(b)贴加纹理的效果图

图 3　有裂缝的地形



(a)消除裂缝的地形网格　　(b)贴加纹理的效果图

图 4　引入 Y 型基元消除裂缝后的地形

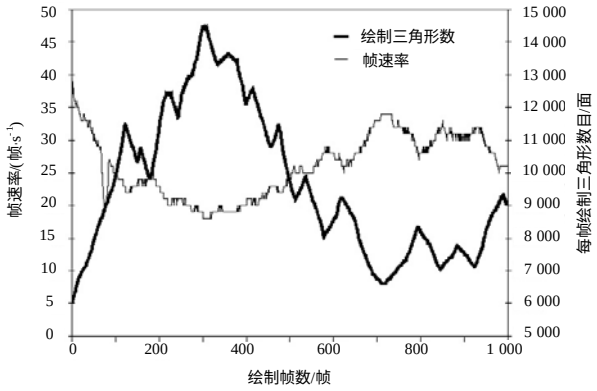


图 5　地形漫游中帧速率和三角形数变化曲线

4　结束语

本文综合了数据动态调用和地形网格简化两方面的技术，提出了一种解决大规模地形实时可视化的方案。通过实验验证该方法能取得很好的效果，在应用数据动态调用和引入 Y 型基元消除裂缝技术的基础上，保证了地形漫游时场景显示的实时、平滑和连续性。

在大规模地形可视化实际的应用系统中，不仅有地形和纹理数据，还有很多其他的数据，比如河流、湖泊和建筑等数据，如何对这些海量数据进行有效的综合管理和调度是今后研究的目标。

参考文献

[1] Lindstrom P, Koller D, Ribarsky R, et al. Real-time, Continuous Level of Detail Rendering of Height Fields[C]//Proc. of SIGGRAPH'96. New Orleans, Louisiana, USA: Computer Graphic, 1996: 109-118.

[2] Duchaineau M, Wolinsky M, Sigeti D, et al. Roaming Terrain: Real-time Optimally Adapting Meshes[C]//Proc. of IEEE Visualization. Phoenix, Arizona, USA: IEEE Visualization, 1997: 81-88.

[3] Hoppe H. Smooth View-dependant Level-of-detail Control and Its Application to Terrain Rendering[C]//Proc. of IEEE Visualization. Washington D. C., USA: IEEE Press, 1998: 35-42.

[4] 翟　巍, 迟忠先, 方　芳, 等. 大规模三维场景可视化的数据组织方法研究[J]. 计算机工程, 2003, 29(20): 26-27.

[5] 戴晨光, 张永生, 邓雪清. 一种用于可视化的海量地形数据组织与管理方法[J]. 系统仿真学报, 2005, 17(2): 406-409.

[6] 柯希林, 曾　军. 动态 LoD 四叉树虚拟地形绘制[J]. 测绘通报, 2005, 51(6): 10-13.

[7] 赵友兵, 石教英, 周　骥, 等. 一种大规模地形的快速漫游算法[J]. 计算机辅助设计与图形学学报, 2002, 14(7): 624-628.