基于GPU的平滑地形可视化算法

张燕燕，姜洪洲，韩俊伟

(哈尔滨工业大学机电工程学院 哈尔滨 150001)

【摘要】提出了一种基于GPU的平滑地形可视化算法，侧重于解决地形可视化方法面临的时间连续性和空间连续性问题。算法采用了规则地形块的批LOD可视化方法。基于平滑过渡的思想，考虑了地形块相邻层次间的过渡和相邻的不同地形块间的边界匹配关系，以地形块的区域划分为基础，为每个顶点实时分配相应的过渡权值，在地形块的绘制过程中同时完成了不同LOD层次以及不同地形块间的平滑过渡，实现了整个地形的平滑可视化。面向GPU的算法设计与实现保证了其执行效率。针对典型数据集，该算法能够以较高的帧率完成大规模地形的实时平滑漫游，避免可视化过程中的裂缝和突跳等不连续现象。

关 键 词 GPU算法； Geomorph； 层次细节模型； 平滑可视化； 大地形实时渲染

中图分类号 TP391 **文献标识码** A doi:10.3969/j.issn.1001-0548.2009.06.024

Smooth Terrain Rendering Algorithm Based on GPU

ZHANG Yan-yan, JIANG Hong-zhou, and HAN Jun-wei

(School of Mechatronics Engineering, Harbin Institute of Technology Harbin 150001)

Abstract A smooth terrain rendering algorithm based on GPU is proposed to solve time and space discontinuity problems during terrain visualization. The algorithm is a coarse-grained aggregated LOD method. Based on the smooth geomorph idea, the terrain chunk is divided into regions and the morphing weight of every vertex in the region is evaluated in real time for current morphing operation. The morphing weight involves the transition between different detail levels and the level difference between neighboring chunks. Through this morphing operation, during the terrain rendering, the smooth transition between different LOD levels and different terrain chunks can be achieved at the same time, and a smooth terrain walkthrough can be gotten. The method is implemented in a GPU-oriented way and the main computation for morphing is accomplished on GPU. The tests show that the algorithm can generate the smooth terrain scene at a high frame rate, and the cracks and pops during rendering can be eliminated efficiently.

Key words GPU-based algorithms; Geomorph; level of details; smooth data visualization; terrain interactive walkthrough

近年来，地形可视化一直是计算机图形学及相关领域的重要研究内容，在游戏、飞行训练、军事演习模拟等领域有着广泛的应用。为保证大地形海量数据集的实时可视化，视点相关的平滑可视化(level of detail，LOD)技术是研究的热点，出现了面向CPU的顶点级LOD方法[1-7]和基于块的批LOD方法[8-12]。这些方法共同面临着可视化的连续性问题，包括时间连续性和空间连续性。其中时间连续性问题是指在场景漫游时不同LOD层次变换过程中的突跳问题；空间连续性问题主要是实时构网中出现的裂缝问题。

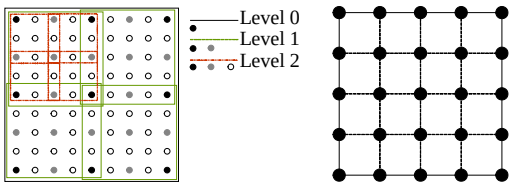
对于时间连续性问题，典型的解决方法是Geomorph方法[1-2]，通过两个LOD层次顶点位置的平滑插值，使模型由一个LOD层次逐渐过渡到另一个层次。对于空间连续性问题，在顶点级LOD算法中是保证产生一个一致性的三角形网。依靠顶点间的依赖关系等规则递归增加粗糙节点的顶点[3-5]；在限制性四叉树的条件下采用特定的三角化方法忽略精细节点的顶点[6]，以及改变精细节点的顶点高程值到与粗糙节点对应位置相同的高度[7]都是较为有效的解决方案。对基于块的批LOD方法，空间连续性除了体现在上述块内三角形网的一致性上，还体现在相邻地形块间的无缝拼接，平滑处理较难控制。一些方法从地形块的构造出发，通过避免块边界的简化[13]或在保持边界特定的状态下完成地形块的生成工作[8,14]，从而保证了块拼接时边界的一致性。该

收稿日期：2008－11－25；修回日期：2009－05－17
基金项目：教育部新世纪优秀人才支持计划(NCET_04_0325)
作者简介：张燕燕(1981－)，女，博士生，主要从事视景仿真和大地形可视化方面的研究.

类方法能够完全避免块间裂缝的发生，但增加了构造地形块的复杂程度。相应地，使用垂直条带、突缘以及三角形填充的方式弥补块间裂缝[9,15]能保证地形块的生成不依赖于特定的地形块生成方法，是一种较简单的解决方案。但是该类方法容易在块接缝处形成陡峭和不连续的特征。另外，针对特定的规则地形块结构，通过实时选择考虑块拼接关系的层次索引模板或衔接模板修改边界顶点的连接方式，实现块间拼接的方法[10-11]简单易行。但是若考虑可视化的时间连续性，基于地形块的Geomorph实现破坏了该方法的有效性。

为此，本文提出一种简单高效的平滑地形可视化算法，采用基于规则块的粗粒度视相关LOD地形可视化方法渲染大地形。针对可视化过程的连续性问题，该算法基于规则地形块结构，给出一种空间连续性和时间连续性一体化处理的平滑解决方案。即采用平滑过渡思想，将整个地形块分区，考虑不同LOD层次间的过渡和相邻块边界的匹配关系，为区内的每个顶点实时分配相应的过渡权值，在地形块的绘制过程中同时完成不同LOD层次以及不同地形块间的平滑过渡，实现整个地形的平滑可视化。在算法实现上，采用面向GPU的设计，将平滑处理过程放在GPU中进行，保证了整个算法的执行效率。

1　算法模型结构



a. 规则地形块层次　　b. 单个地形块拓扑结构

图1　模型结构示意图

本文的算法采用一种规则的层次地形块作为其实现的基础，即在预处理阶段使用一种二倍率过滤的简化方法和基于四叉树的地形块组织方式，将原始的地形采样数据组织成由固定大小的规则地形数据块构成的金字塔层次模型结构。图1a为该结构的俯视图(以3层数据为例)，不同形状的线框表明该层地形块数据覆盖的区域，框内与层次对应的顶点表明该地形块的采样数据集，其中顶层的数据采用二倍率隔行采样的方式由相邻底层获得。单个地形块示意图如图1b所示，块内采样点均匀分布，能够产生一个一致性的三角形网，即块内空间连续；并且所有块具有相同的顶点数和拓扑结构，数据块的数据集为：

$$P_l=\{V_{i,j}\,|\,V_{i,j}=H(x,z),0\leqslant i,j\leqslant p_{\text{size}},\ x=p_{\text{xoff}}+2^l i,z=p_{\text{zoff}}+2^l j\} \tag{1}$$

式中　$H(x,z)$ 为高程采样函数；p_{xoff}、p_{zoff} 为地形块的层内偏移量；i、j 为块内的网格坐标；x、z 为物体空间坐标；2^l 为采样点空间间隔。

2　一体化平滑处理

本文采用平滑过渡的思想，通过为顶点实时分配时间和空间相关的过渡权值保证在地形块的绘制过程中同时完成不同LOD层次以及不同地形块间的平滑过渡，实现地形的平滑可视化。

2.1　平滑过渡原理

具体的过渡操作是指根据顶点的过渡权值连续地改变一个LOD层次模型中顶点的位置，使该顶点平滑地过渡到其在相邻另一层次模型中的对应位置。如图2所示，顶点B在位置B和M间平滑过渡。过渡过程中顶点位置为：

$$y'=y-\omega\Delta y \tag{2}$$

式中　y' 为过渡过程中顶点的高程值；y 为当前模型中顶点的高程值；Δy 为顶点与其相邻粗糙的模型层次中对应位置顶点的高程差；ω 为顶点的过渡权重，在过渡过程中是一个变化量，与顶点的空间位置和时间状态相关。

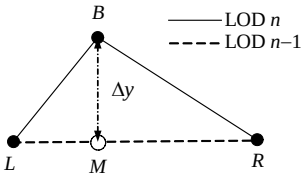


图2　平滑过渡原理示意图

2.2　过渡权值的计算与分配

过渡权值控制着平滑过渡的过程。在本文的一体化平滑处理算法中，过渡权值 ω 有两个作用。对于地形块内的顶点，ω 用来实现两个LOD层次间的平滑切换；对于边界处的顶点，ω 还用来保证地形块边界的无缝拼接。因此，将 ω 分解为时间过渡权值 ω_t 和空间过渡权值 ω_s。

考虑到本文的算法以规则地形块作为LOD模型选择单位，为简单起见，时间相关的 ω_t 以整个地形块为单位进行计算，ω_t 值可以直接选取为受时间或视点运动速度控制的参数。为了避免视点固定而顶点位置自动变化的现象，需要保证 ω_t 是与视点运动位置相关的，所以间接选择视点与地形块的距离作为 ω_t 的计算标准，从而保证地形块随视点的运动逐渐由当前层次过渡到其相邻的粗糙层次，即有：

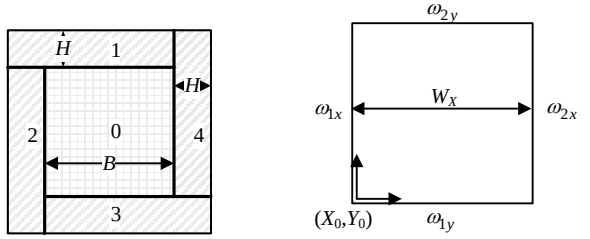
$$\omega_t = \min(\max((D - D_{\min})/(D_{\max} - D_{\min}),0),1) \quad (3)$$

式中 D 为地形块包围体上距视点最近的点同视点间的距离；$D_{\min}$ 和 $D_{\max}$ 为选择该地形块的两个极限距离。

另一方面，为了实现空间连续性，地形块边界处相应的空间过渡权值 ω_s 根据地形块与其相邻块间的层次关系确定，即：

$$\omega_s(l,l_n)=\begin{cases}1 & l-l_n>0\\ \\ 0 & l-l_n\leqslant 0\end{cases} \quad (4)$$

若相邻块比当前块粗糙，则边界 ω_s 设定为1，图2中顶点*B*位于低层模型的*M*位置，保证边界处的顶点位置和相邻粗糙层次的完全相同。若相邻块与当前块的层次相同或更精细，则边界 ω_s 设定为0，图2中顶点*B*保持在该层模型相应的位置不变，即以最精细的状态呈现。使用 ω_s 设定值，则无论两个相邻块的层次关系如何，都可以保证在边界处的顶点完全匹配，从而能够有效地消除裂缝。



a. 地形块区域划分示意图 b. 区域内过渡权值的分配示意图

图3 过渡权值分配方法

为地形块内顶点分配相应的 ω 值可实现整个地形的一体化平滑处理。本文为了便于过渡权值分配，将每个地形块分为5个区域，如图3a所示。区域0代表地形块的内部，区域1～4代表地形块边界向内部过渡的4个边 界区域。对每个区域可以根据时间和空间连续性的相关要求规定4个边界的过渡权值 ω，如图3b所示。区域内部每个顶点的 ω 可由边界过渡权值平滑插值获得，即 $\omega=\max(\omega_x,\omega_y)$。其中 ω_x 和 ω_y 分别由沿*x*方向和沿*y*方向对边界 ω 线性插值得到：

$$\omega_x=\frac{\omega_{2x}-\omega_{1x}}{W_x}(X-X_0)+\omega_{1x} \quad (5)$$

式中 W_x 为*x*方向上区域的宽度；ω_y 可以采用类似的方法获得。因此，通过为每个区域实时分配4个边界的过渡权值可达到不同的过渡目的。对于0区域，其负责实现地形块在不同层次模型间时间相关的过渡，因此本文的算法将区域0的4个边界过渡权值都分配为式(3)计算的 ω_t。边界区域1～4主要负责实现相邻的不同层次地形块间的无缝拼接，同时还须保证地形块边界到内部的平滑变化。从而对于边界区域，其位于地形块边界的两条边分配为式(4)计算的 ω_s，而边界区域中位于地形块内部的另外两条边的权值指定为地形块的 ω_t 值，与相邻的区域0一致，保证从地形块边界向内部的平滑过渡。

在地形渲染过程中，该算法以地形块区域为单位实时计算地形块内顶点的过渡权值，并按照式(2)计算相应的顶点即时位置，实现了时间和空间的一体化平滑处理。同时，由于该平滑解决方案是基于规则地形块结构建立的，因而可扩展到纹理和法向量图的平滑处理上，保证渲染效果的一致性。

3 面向GPU的平滑算法实现

实时视相关LOD方法生成的地形以地形块区域为基本的渲染单元；并且为进一步减轻CPU的负载，该算法将平滑处理过程放到GPU中进行，即使用GPU的顶点着色器完成上述的平滑处理。

为增加顶点的重用率，提高地形块的渲染效率，该算法使用顶点索引机制构造三角形条带来完成整个地形区域的渲染，即以整个块的顶点排列为基础，按照图3a所示的区域划分方式(*H*、*B*依照地形块的尺寸成比例变化)以*Z*型顺序遍历其中每个三角形，为每个区域预生成一个三角形条带索引序列。由于所有地形块的顶点分布均相同，这种表示块内顶点连接方式的索引序列适用于所有地形块，所以只需为所有地形块的相应区域预生成5个抽象的索引序列，以vertex buffer object(VBO)的形式缓存在GPU中。

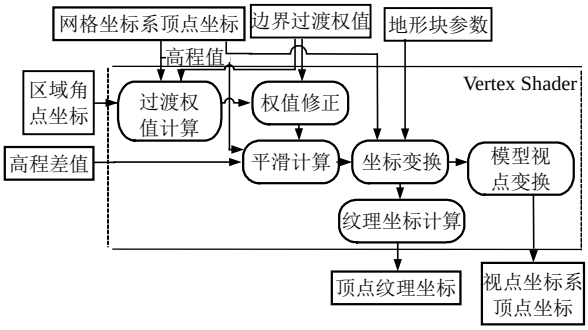


图4 顶点程序结构框图

由索引序列组织的地形顶点数据被输入到顶点着色器中，以区域为单位实现地形的平滑处理，顶点程序的结构如图4所示。为发挥上述抽象索引序列的作用，输入的顶点坐标在网格坐标系下给出，通过地形块参数(块的缩放比例和偏移位置)可计算出物体坐标系下相应的坐标值。顶点程序使用由式(3)

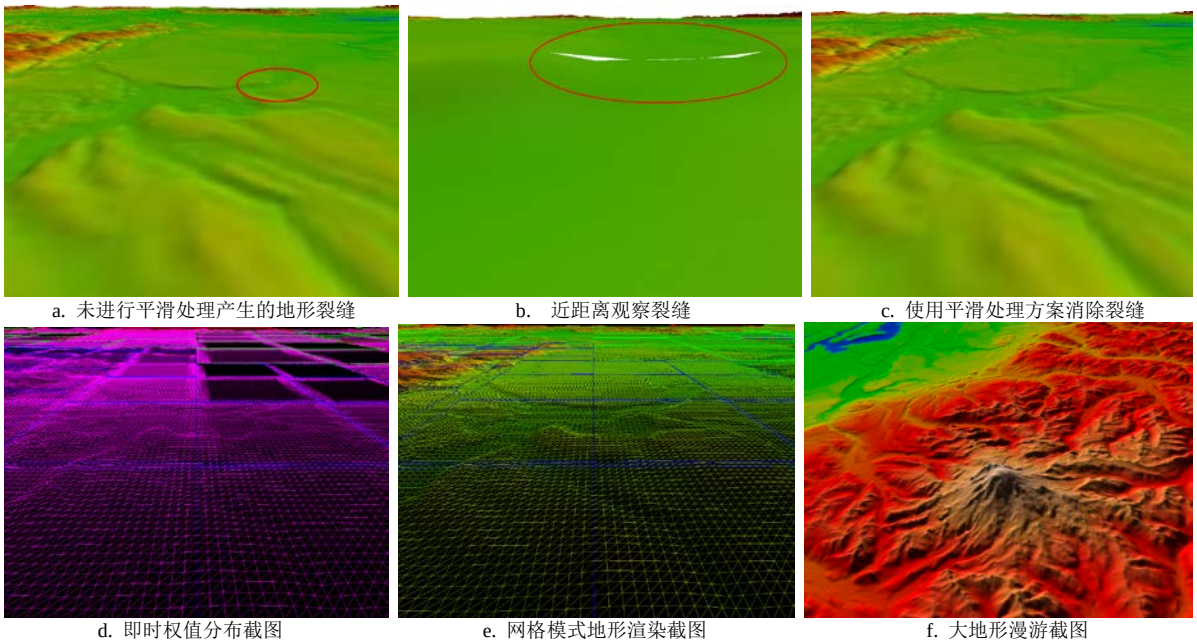
和式(4)指定的边界过渡权值，按照式(5)计算区域中每个顶点的过渡权值，并对该权值进行修正(保证权值的正确分布)。修正后的权值和高程差值结合使用式2对顶点的原始高程值进行平滑计算，生成当前的顶点高程值，最终输出视点坐标系下的顶点坐标，由GPU进行下一步的渲染处理。对于顶点程序，除网格坐标系顶点坐标和高程差值外，其余参数值均以常量的形式传入。

4　实验结果与分析

本文算法的硬件平台为P4 CPU 3.06 GHz，内存512 MB，显卡NVIDIA GeForce 7300GT。实验数据集选取Puget Sound 区域的两种不同分辨率的数据集Puget1K和Puget16K(http:// www.cc.gatech.edu / projects /large_models/)。该算法采用C++实现，顶点着色器部分使用CG语言完成，定义地形块过渡区域的宽度为整个地形块宽度的1/6。渲染程序在Windows平台下运行，定义其水平视场角为90°，窗口分辨率为1 024×768，规定视点沿预定义的路径按照固定的方式漫游。

该算法的平滑处理效果如图5a～图5c所示。图5a是没有进行平滑处理的地形场景，图中的圆圈标明了其中的一个裂缝，近距离观察该裂缝如图5b所示。比较而言，使用本文的平滑处理方案的渲染截图如图5c所示，原来的裂缝已经完全消除。因此，本文的平滑算法能够有效地解决地形渲染的空间连续性问题。图5f显示的是该算法针对Puget16K数据集的漫游截图。该算法能够保证以较高的帧率完成固定路径的场景漫游，实现在视觉外观上和原始地形接近的具有较为丰富细节的地形可视化效果。并且在漫游过程中没有明显的“突跳”现象，解决了地形渲染的时间连续性问题。图5d显示的是地形区域中过渡权值的分配情况，图中网格亮度由亮到暗对应过渡权值由0～1。图5e是和图5d相对应的地形网格图。

表1给出了算法对于特定数据集的平均帧率统计(τ=2)。从表中可看出，针对同样的数据集是否使用平滑处理方案对帧率的影响不大，因此基于GPU的平滑处理实现方法能够保证算法的整体性能。另外，针对大数据集Puget16K，本文的算法仍能以较高的帧率实现实时平滑漫游，表明该算法在海量地形数据的平滑渲染方面是有效的。



a. 未进行平滑处理产生的地形裂缝　b. 近距离观察裂缝　c. 使用平滑处理方案消除裂缝

d. 即时权值分布截图　e. 网格模式地形渲染截图　f. 大地形漫游截图

图5　渲染截图

表1　算法运行性能

渲染算法	数据集	高程分辨率	平均帧率(τ=2)
本文算法	Puget16K	16 385×16 385	98
本文算法	Puget1K	1 025×1 025	352
本文算法	Puget1K	1 025×1 025	375

(不使用平滑处理)

5　结　论

本文面向GPU实现了一种平滑地形可视化算法。该算法采用基于规则地形块的批LOD方法渲染

大地形，给出了一种同时处理空间连续性和时间连续性问题的一体化平滑解决方案。基于平滑过渡的思想，考虑了地形块相邻层次间的过渡和相邻的不同地形块间的拼接关系，以地形块的区域划分为基础，按照顶点在地形块平滑过渡过程中的作用为其实时分配相应的过渡权值，在地形块的绘制过程中同时完成不同LOD层次及不同地形块间的平滑过渡，实现了整个大地形的平滑可视化，有效地解决了基于地形块的地形渲染算法面临的时间连续性和空间连续性问题。在实现上该算法采用面向GPU的设计，使用基于模版的三角形条带可视化方法，并将平滑处理过程放到GPU中进行，保证了整个算法的高效执行。针对典型数据集，该算法能够以较高的帧率完成大规模地形的实时平滑漫游，避免了裂缝和突跳现象。

参 考 文 献

[1] HOPPE H. Progressive meshes[C]//Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. New York, NY, USA: ACM Press, 1996: 99-108.

[2] LARSEN B D, CHRISTENSEN N J. Real-time terrain rendering using smooth hardware optimized level of detail[J]. Journal of WSCG, 2003, 11(2): 282-289.

[3] LINDSTROM P, KOLLER D, RIBARSKY W, et al. Real-time, continuous level of detail rendering of height fields[C]//Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. New York, NY, USA: ACM Press, 1996: 109-118.

[4] DUCHAINEAU M, WOLINSKY M, SIGETI D E, et al. Roaming terrain: real-time optimally adapting meshes[C]//Proceedings of the 8th Conference on Visualization'97. Los Alamitos, CA, USA: IEEE Computer Society Press, 1997: 81-88.

[5]韩 敏, 汤松涛, 李 洋. 大规模地形实时可视化算法[J]. 计算机工程, 2008, 34(13): 270-272.
HAN Min, TANG Song-tao, LI Yang. Real-time visualization algorithm of large-scale terrain[J]. Computer Engineering, 2008, 34(13): 270-272.

[6] ROTTGER S, HEIDRICH W, SLUSALLEK P. Real-time generation of continuous levels of detail for height fields[C]//Proceedings of WSCG'98. Plzen Czech Republic: Science Press, 1998: 315-322.

[7] ZHAO You-bing, ZHOU Ji, SHI Jiao-ying, et al. A fast algorithm for large-scale terrain walkthrough[C]// Proceedings of CAD/Graphics 2001. Kunming: International Academic Publishers, 2001: 602-606.

[8] LEVENBERG J. Fast view-dependent level-of-detail rendering using cached geometry[C]//Proceedings of the Conference on Visualization '02. Washington D C, USA: IEEE Computer Society, 2002: 259-266.

[9] ULRICH T. Rendering massive terrains using chunked level of detail control[C]//SIGGRAPH 2002 Course Notes. San Antonio, Texas: ACM, 2002.

[10] 程甜甜, 陈 阁, 陈 新, 等. 基于GPU的大规模地形场景实时渲染[J]. 苏州大学学报(工科版), 2008, 28(3): 1-5.
CHENG Tian-tian, CHEN Ge, CHEN Xin, et al. Real-time rendering large scale landscapes base on GPU[J]. Journal of Suzhou University(Engineering Science Edition), 2008, 28(3): 1-5.

[11] YOTAM L, ZVI K, JIHAN E S. Seamless patches for GPU-based terrain rendering[J]. The Visual Computer: International Journal of Computer Graphics, 2009, 25(3): 197-208.

[12] 廖昌阊, 李 辉, 潘宏伟, 等. 宏三角形的大规模地形漫游算法[J]. 电子科技大学学报, 2008, 37(1): 120-123.
LIAO Chang-chang, LI Hui, PAN Hong-wei, et al. A marco-triangle-based algorithm for large terrain rendering[J]. Journal of University of Electronic Science and Technology of China, 2008, 37(1): 120-123.

[13] HOPPE H. Smooth view-dependent level-of-detail control and its application to terrain rendering[C]//Proceedings of the Conference on Visualization'98. Los Alamitos, CA, USA: IEEE Computer Society Press, 1998: 35-42.

[14] 殷 媛, 陈国军, 吴 威. 地形分块绘制中的边界裂缝处理算法[J]. 计算机辅助设计与图形学学报, 2006, 18(10): 1557-1562.
YIN YUAN, CHEN Guo-jun, WU Wei. An algorithm of avoiding crack for rendering parting terrain[J]. Journal of Computer-Aided Design & Computer Graphics, 2006, 18(10): 1557-1562.

[15] LINDSTROM P, KOLLER D, HODGES L F, et al. Level-of-detail management for real-time rendering of phototextured terrain[R]. GVU Technical Reports. Georgia Institute of Technology, 1995.

[16] LOSASSO F, HOPPE H. Geometry clipmaps: Terrain rendering using nested regular grids[C]//Proceedings of the 2004 SIGGRAPH Conference. New York, NY, USA: ACM Press, 2004: 769-776.

编 辑 黄 莘