文章编号：2095-4980(2015)01-0135-07

一种基于视点的大规模三维地形实时渲染算法

郭雪峰，孙红胜，岳春生

(信息工程大学 信息系统工程学院，河南 郑州 450002)

摘　要：针对大规模三维地形的实时渲染问题，提出一种基于顶点法向量的模型简化算法。渲染过程中利用视图体的投影方式快速裁剪数据块，丢弃与视图体相离的数据块，将部分可见和完全可见的数据块全部加载到内存，根据视点到节点的距离和地形的粗糙程度构造了节点分辨率评价函数，最后采用加点的方法消除裂缝。仿真结果表明，该算法能有效减少渲染地形所需的三角形数目和时间，且对地形的逼真度影响较小。

关键词：地形渲染；模型简化；法矢量；数据调度；节点评价函数

中图分类号：TN124　　**文献标识码：**A　　**doi：**10.11805/TKYDA201501.0135

A real-time rendering algorithm for large scale three-dimensional terrain based on viewpoint

GUO Xuefeng, SUN Hongsheng, YUE Chunsheng

(Institute of Information Systems Engineering, Information Engineering University, Zhengzhou Henan 450002, China)

Abstract: A simplified model algorithm based on vertex normal vector is presented aiming for real-time rendering of large-scale three-dimensional terrain problem. In the rendering process, blocks are cut quickly by the projection of body views; the data blocks which are apart from body views are discarded; and the partially visible or fully visible blocks are loaded into the memory. This algorithm constructs node resolution evaluation function according to the distance(between the viewpoint and the node) and the roughness of the terrain as well. It eliminates cracks by adding points. Simulation results show that the proposed algorithm can reduce the number of triangles and the time required to render the terrain effectively.

Key words: terrain rendering; model simplification; normal vector; data scheduling; node evaluation function

大规模三维地形的实时渲染技术在地理信息系统、3D 虚拟游戏、战场环境仿真等领域有着重要的应用，其研究的核心问题是解决海量地形数据构成的复杂地形表面模型与计算机硬件有限绘制能力之间的矛盾。由于人们对地形的逼真程度要求越来越高，仿真的地形范围越来越大，建立三维场景需要的数据量越来越多，且这些数据无法一次性被装入内存进行调度，这就要求尽可能地降低地形模型的复杂度，选用合适的数据调度策略，以提高系统渲染地形的效率。目前对这些的研究工作主要集中在地形网格的快速生成和简化、基于外存(Out-of-Core)的海量数据存储、组织和调度、地形的渲染算法等方面。

细节层次模型[1](Level of Detail, LOD)技术是解决这一难题的核心技术，该技术分为视点无关 LOD 算法和视点相关 LOD 算法。前者原理简单，易于实现，但是需要存储多个相互独立的模型，对计算机硬件要求较高，且模型之间不能自然过渡，容易产生跳变现象。视点相关 LOD 算法通常基于块对地形模型进行二叉树或四叉树[2]分割，目前国内外都对其进行了大量的相关研究，典型的算法有基于视点的递进网格(View Dependent Progressive Meshing, VDPM)算法[3]和实时优化自适应网格(Real-time Optimal Adaptive Mesh, ROAM)算法[4]等。这类算法在采样点密集的地方，对基于三角形的视图体裁剪仍需大量计算，在一定程度上降低了渲染的效率。本文根据 LOD 算法思想，利用求顶点法向量的方法对规则格网进行简化，同时采用基于块的视区裁剪和基于块的数据调度策略，根据系统的评价函数判断渲染地形时所采取的细节程度，一次性整体构网，网格实时更新，有效地提高了地形实时渲染的效率。

收稿日期：2014-01-23；修回日期：2014-02-24

1 基于块和视点的地形渲染算法

1.1 算法流程

大规模三维地形的实时渲染算法由数据预处理、模型构建[5]与简化、模型渲染 3 大部分组成。算法流程如图 1 所示。在数据预处理阶段对数字高程模型(Digital Elevation Model，DEM)高程数据进行四叉树划分；模型构建与简化阶段首先对构建的规则格网进行简化，然后测试地形块与视图体的关系，丢弃与视图体相离的地形块，根据节点评价函数选择合适的 LOD 模型，最后采用基于视点的方法消除裂缝，把最终的地形简化模型交由图形渲染设备进行处理。

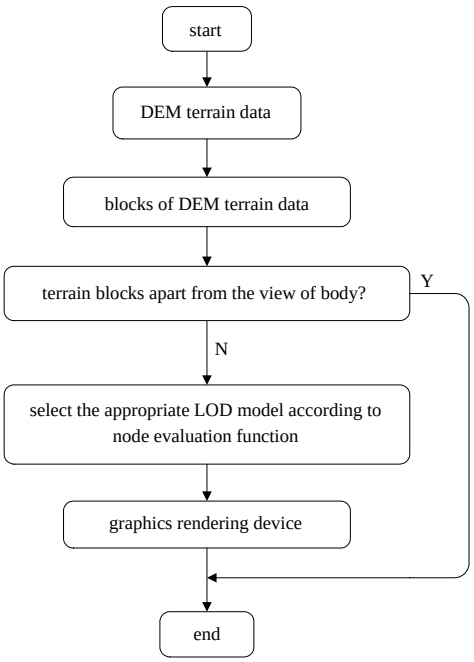

Fig.1 Process of massive terrain rendering algorithm based on viewpoint
图 1 基于视点的大规模地形渲染算法流程

1.2 模型简化算法

目前数字高程模型主要有 3 种表示模型：等高线模型、不规则三角网和规则格网模型。不管是哪种模型，都是采用网格来逼近地形，同一幅地形绘制的网格数越多越逼真，理想情况下(只考虑地形的真实感，不考虑渲染地形的时间)可以使用 B 样条或 Bezier 样条曲面进行逼真绘制。曲面的视觉特征主要表现在地形的起伏程度及视觉反映，如果是在大片的平坦地区或变化平缓的区域则存在着大量的数据冗余，此时用大量的数据表示地形是不必要的，可以对这些地区的模型进行简化[6]。曲面的一维数据模型如图 2 所示。

曲面上点 a 的梯度为 $\nabla f(a)$，则有：

$$\nabla f(a)=\left[\frac{\mathrm{d}f(t)}{\mathrm{d}t}\right]\boldsymbol{i}\Bigg|_{t=a} \tag{1}$$

将上式扩展到欧式空间为：

$$\nabla f(a)=\left[\frac{\partial f(x,y,z)}{\partial x}\boldsymbol{i}+\frac{\partial f(x,y,z)}{\partial y}\boldsymbol{j}+\frac{\partial f(x,y,z)}{\partial z}\boldsymbol{k}\right]\Bigg|_{(x,y,z)=\boldsymbol{a}} \tag{2}$$

式中 $\boldsymbol{a}$ 表示点 a 在世界坐标系中的坐标，同理可求得其他点的梯度值。由图中曲线起伏变化情况可以看出，a,b,c 3 点的梯度值近似相等，即 $\nabla f(a)\approx\nabla f(b)\approx\nabla f(c)$，可以认为 a,b,c 的起伏变化程度基本相同，由此可认为点 b 是冗余点，可以用矢量 $\boldsymbol{ac}$ 代替 $\boldsymbol{ab}$ 和 $\boldsymbol{bc}$；同理考虑到 $\nabla f(d)\approx\nabla f(e)\approx\nabla f(f)$，$\nabla f(g)\approx\nabla f(h)\approx\nabla f(i)$，判定点 e 和点 h 是冗余点，可以用矢量 $\boldsymbol{df}$ 代替矢量 $\boldsymbol{de}$，$\boldsymbol{ef}$，以 $\boldsymbol{gi}$ 代替 $\boldsymbol{gh}$，$\boldsymbol{hi}$，所以经过简化后只剩下点 a,c,d,f,g,i,j。

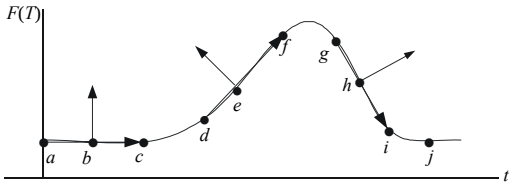

Fig.2 One-dimensional mathematical model of surface
图 2 曲面的一维数学模型

由于梯度(法向量)表示的是曲面起伏程度变化最大方向，因此用法向量来判决冗余点可以保证对所渲染的地形影响最小。而在地形的实时渲染中，通常采用网格来逼近地形，所以本文提出一种基于顶点法向量的模型简化算法，该算法以视觉特征最优化为简化准则来判决冗余点，顶点法向量的求法如图 3~图 4 所示。

如图 3 所示，与顶点 O 相交的有 4 个平面，这 4 个平面的单位法向量分别为 $\boldsymbol{n}_1,\boldsymbol{n}_2,\boldsymbol{n}_3,\boldsymbol{n}_4$，顶点 O 的法向量为 $\boldsymbol{n}$，令 $\boldsymbol{n}$ 为点 O 周围 4 个平面法向量之和的平均值，则有：

$$\boldsymbol{n}=\frac{1}{4}(\boldsymbol{n}_1+\boldsymbol{n}_2+\boldsymbol{n}_3+\boldsymbol{n}_4) \tag{3}$$

对其进行单位化，记单位法向量为 $\boldsymbol{n}_o$，则有：

$$\boldsymbol{n}_o=\frac{\boldsymbol{n}}{|\boldsymbol{n}|}=\frac{n_x\boldsymbol{i}+n_y\boldsymbol{j}+n_z\boldsymbol{k}}{\sqrt{(n_x)^2+(n_y)^2+(n_z)^2}} \tag{4}$$

现令：

$$\varepsilon(n_{io})=\left|1-\boldsymbol{n}_i\bullet\boldsymbol{n}_o\right|,\ (i=1,2,3,4) \tag{5}$$

式中 $\varepsilon(n_{io})\in[0,1]$，当 $\boldsymbol{n}_i$ 和 $\boldsymbol{n}_o$ 的夹角为零时，$\varepsilon(n_{io})=0$，由于是在地形变化缓慢的区域，所以 4 个平面的法向量与顶点的法向量夹角小于 90°，对 $\varepsilon(n_{io})$ 设定一个上限值 λ，当：

$$0\leqslant\varepsilon(n_{io})\leqslant\lambda \tag{6}$$

此时该区域地形变化缓慢或基本平坦，可认为点 O 是冗余点，删除该点对视觉效果的影响不大；同理对点 O 周围的顶点也做同样的处理，由此可得网格模型简化示意图如图 4 所示，该方法不但实现了对地形模型的简化，还可以通过设置 λ 的值来设定简化的力度，而且避免了简化时删除地形特征点，对地形的真实感影响较小。

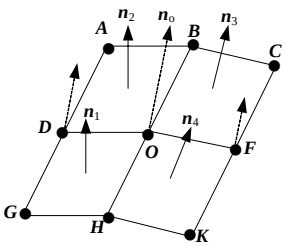

Fig.3 Schematic diagram of vertex normal
图 3　顶点法向量示意图

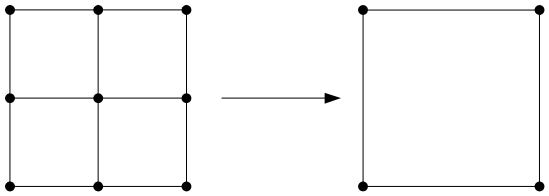

Fig.4 Schematic diagram of simplified grid model
图 4　网格模型简化示意图

1.3 视区裁剪和数据组织调度

在大范围的地形渲染中，大部分的数据都位于视区之外，如果在渲染地形时把这些数据全部加载渲染，会严重影响其实时性；当数据量很大时，不能完全读取这些数据，所以在地形渲染之前要先进行可视区域裁剪[7]，由于其节点数据量巨大，裁剪运算约占总时间的1/3，在很大程度上影响了算法的效率。本文考虑基于视点的影响，提出基于视点的数据块和四叉树节点相结合的裁剪算法。把视图体投影到 x–y 平面上，根据投影对分块地形数据进行可见性剔除和裁剪，如图 5~图 6 所示。

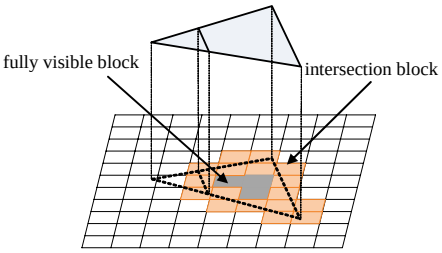

Fig.5 Schematic diagram of view of body projection
图 5　视图体投影示意图

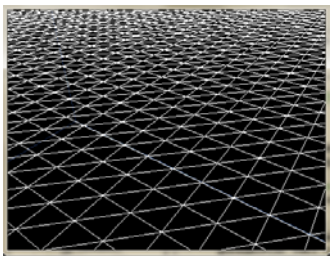

Fig.6 Schematic diagram of terrain planar projection
图 6　地形平面投影示意图

图 5 中白色区域表示位于视图体之外的数据快，灰色方格代表完全可见的数据块，黄色方格代表与视图体相交的数据块。对于位于视图体之内的数据块，渲染地形时需全部加载，而对于与视图体相交的数据块，目前大都对其节点进行分解，但是分解操作所带来的复杂运算对其效率影响也很大，所以本文对图 6 中与视图体相交的部分不进行分解操作，直接进行绘制，虽然多绘制了投影视域以外的部分，但是省去了裁剪处理的大量计算，性能并未降低。

三维地形可视化系统往往会涉及到较大的区域，相应的数据量也很大，由于计算机内存容量、计算速度和绘制能力有限，必须对输入地形数据量作一定限制[8]。场景数据的调度直接关系到场景显示的流畅性和交互的实时性，三维场景中 DEM 数据的调度是以视图体为基础的动态调度[9]，将视图体简化成水平面上的梯形，并将整个场景按照四叉树进行划分，以方便可见性判断。

根据上述投影，在视点确定的情况下，将位于视图体之内和相交部分的数据块都加载到内存，当视点漫游时[10]，如图 7 所示。视图体由实线移动到虚线位置时，同样加载

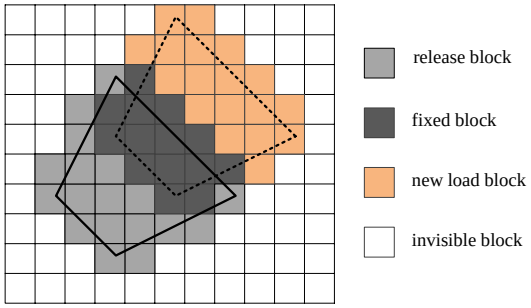

Fig.7 Scheduling policy
图 7　调度策略

位于视图体之内和视图体相交的数据块。由图 7 可以看出，移动前的部分可见块变为不可见块被释放出来，部分可见块原先就位于内存中，不需要再重新加载。当视点在场景中漫游时，视图体四周都有与视图体相交的数据块，本文在渲染地形时把完全可见数据块和部分可见数据块都读入内存，当视点漫游时，不管移动的方向如何，总有一部分可见块变为完全可见块，此时这些数据块就不需要重新加载，提高了地形渲染的效率，且能使画面具有连续感。

1.4 节点分辨率评价函数

节点分辨率评价函数[11]是实时地形表示的测度，它用来确定在视点移动时如何根据视点的位置变化来选择合适的分辨率地形。通常情况下，节点评价函数的建立和具体的应用环境有关，影响节点评价函数的因素有：视点到节点区域的距离、地形的粗糙度、漫游时视点移动的速度、视觉感知强度、数据块本身的大小等。为了减少 CPU 的运算，仅考虑部分因素。

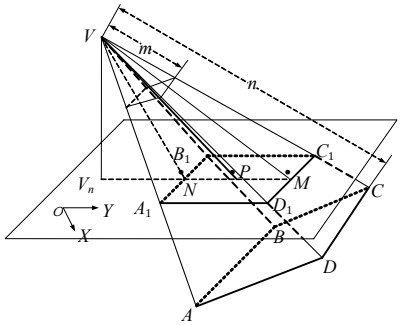

Fig.8 Schematic diagram of perspective projection
图 8 透视投影示意图

视景体的透视投影变换图如图 8 所示，其中视点为 V，视点在平面上的投影为 V_n，点 P 为视线与水平面的交点，A,B,C 在水平面上的投影点为 A_1,B_1,C_1,D_1，设点 V,P 在三维坐标系中的坐标为 $V(V_x,V_y,V_z)$、$P(P_x,P_y,P_z)$，本文用视点到目标块中心的距离代表视点到目标块的距离，则有：

$$l=\sqrt{(V_x-P_x)^2+(V_y-P_y)^2+(V_z-P_z)^2} \tag{7}$$

式(7)涉及到乘方开方运算，对速度有一定的影响，本文采取以下简化公式来计算距离：

$$l=|V_x-P_x|+|V_y-P_y|+|V_z-P_z| \tag{8}$$

根据 LOD 算法思想，离视点越近的数据块被划分得越详细，LOD 的等级越高，所以设定一个与距离相关的判定因子 C，则有距离判断法则：

$$\frac{l}{d}<C \tag{9}$$

这里 C 是一个距离调节因子，为常数；d 表示数据块的边长。当满足式(9)时，说明视点离数据块很近，需要进一步划分，使用户能够看到更多的细节层次，否则不必进行下一步的划分。

在实际的视觉效果中，人眼直视的区域最清晰，眼睛余光观察到的场景比较模糊，所以地形的详细程度不仅与视点的位置有关，还与视线的方向有关。在表达式中加入一个与视线方向相关的因子 F，则用下式求得的距离代替 l：

$$l'=l+l\times F \tag{10}$$

这里取 $F=\sin\theta$，其中 θ 为 VP 与 VO 的夹角，O 为坐标系原点。

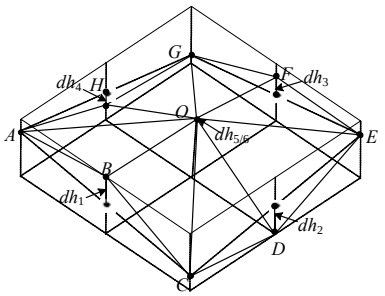

Fig.9 Schematic diagram of terrain roughness
图 9 地形粗糙度的计算示意图

地形节点的详细程度不仅与距离有关，还与地形的粗糙度有关。地形粗糙度有多种定义方式，本文采用基于块的粗糙度计算标准，用地形块的整体粗糙度来确定节点的误差大小，适用于自顶向下的遍历顺序，计算方法如图 9 所示。

当对数据块 $ACEG$ 划分时，4 个角点的坐标没有发生变化，但是 4 边中点及中心点处的高度发生了变化，引入了新的误差，6 个高度差的绝对值计算方法为：

$$\begin{aligned}
|\Delta h_1|&=\left|\frac{\text{vertex}A+\text{vertex}C}{2}-\text{vertex}B\right| \qquad |\Delta h_2|=\left|\frac{\text{vertex}C+\text{vertex}E}{2}-\text{vertex}D\right| \\
|\Delta h_3|&=\left|\frac{\text{vertex}E+\text{vertex}G}{2}-\text{vertex}F\right| \qquad |\Delta h_4|=\left|\frac{\text{vertex}G+\text{vertex}A}{2}-\text{vertex}H\right| \\
|\Delta h_5|&=\left|\frac{\text{vertex}A+\text{vertex}E}{2}-\text{vertex}O\right| \qquad |\Delta h_6|=\left|\frac{\text{vertex}C+\text{vertex}G}{2}-\text{vertex}O\right|
\end{aligned} \tag{11}$$

考虑数据块的边长为 d，定义一个变量 H，H 的计算方法为：

$$H=\frac{1}{d}\max\left|\Delta h_i\right| \tag{12}$$

H 能够反映数据块在划分前后粗糙度的变化，所以可以得到与粗糙度相关的判断法则：

$$\frac{l}{H}<c \tag{13}$$

这里 c 表示粗糙度的调节因子，是一个可调节的全局变量，当粗糙度满足上式时，说明粗糙度较高，需要进一步划分。

综上所述，可以得到这个判定公式(14)：

$$f=\frac{l(1+\sin\theta)}{d\bullet H\bullet C\bullet c}<1 \tag{14}$$

式(14)说明，当 $f<1$ 时，意味着当前地形因为距离很近或者粗糙度太高，需要进一步划分。从式(14)可以看到，当地形十分平坦时 H 为零，会造成零除，当 H 为一个很小的数值时，可能造成离视点很近的数据块也得不到充分的分割，所以在实际应用中做如下改动：

$$f=\frac{l(1+\sin\theta)}{d\bullet C\bullet\max(c\bullet H,1)}<1 \tag{15}$$

上述因素综合形成了节点评价函数，但是在实验中发现随着视点的移动，有时会发生几何形变，即视点改变时有些细节会突然出现或消失。为了减少这种情况的出现，对距离 l 设定一个阈值 L，当 $l<L$ 时，只用地形粗糙度评价公式进行节点划分，这样虽然不能完全消除几何形变，但是也在一定的范围之内控制了几何形变。

1.5 裂缝的处理

在基于四叉树的地形节点简化算法中，不可避免地会出现“裂缝”问题[12]。在对地形进行 LOD 处理时，使得不同地形块以及同一地形块的不同节点具有不同的分辨率，当渲染地形时，不同分辨率的节点之间就可能产生裂缝，如图 10 所示。

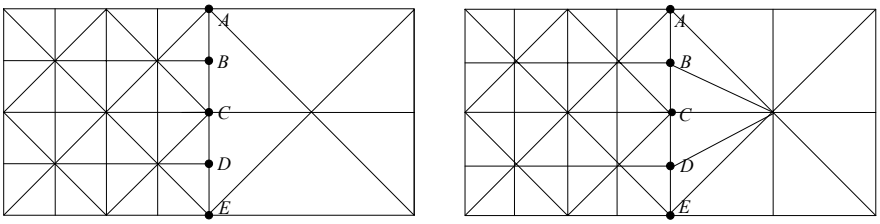

Fig.10 Produce and eliminate cracks
图 10 裂缝的产生与消除

图 10 中左边的节点在 B 点和 D 点使用了高程数据绘制了三角形，而右边的节点分辨率较低，没有绘制以 B 点和 D 点为顶点的三角形，其中 B 点的高程值取 A 点和 C 点高程值的平均值，D 点的高程值取 C 点和 E 点高程值的平均值，这样同一个点在 2 个不同的节点中就可能具有不同的高程值，于是渲染地形时就会形成裂缝。

本文使用添加顶点的方法来消除裂缝，即在分辨率较低的地形节点中添加顶点，使得该顶点与相邻的节点中的顶点具有相同的高程值，重新组织低分辨率节点，即增加 1 条边生成新的三角形来消除裂缝。

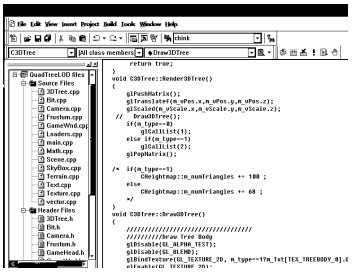

Fig.11 Interface of terrain simulation
图 11 地形仿真操作界面

2 仿真结果及分析

本算法利用 OpenGL 在 Pentium(R) 4 CPU 2.80 GHz,1.00 GB 的内存，Windows 2003，VC6.0 环境中进行仿真，屏幕分辨率为 1 024 × 768，生成的地形大小为 512 × 512，仿真操作界面截图如图 11 所示。

表 1 规则格网算法和本文算法对比
Table1 Regular grid algorithm and the proposed algorithm

	size of terrain	number of triangles	rendering time /s	frame rate /fps	simplified rate/%
regular grid	512×512	5 084	0.76	56	50.6
proposed algorithm	512×512	2 512	0.63	64	

表 1 为规则格网算法和本文算法的有关数值(渲染的三角形数目、渲染地形所需时间、帧率)的比较；图 12~图 15 为地形仿真部分效果截图。

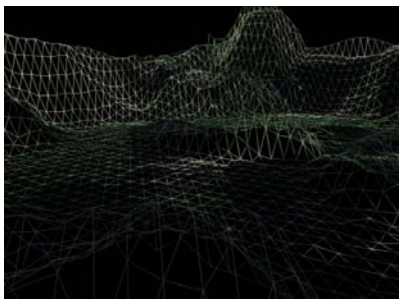

Fig.12 Terrain rendering of regular grid
图 12 规则格网地形效果图

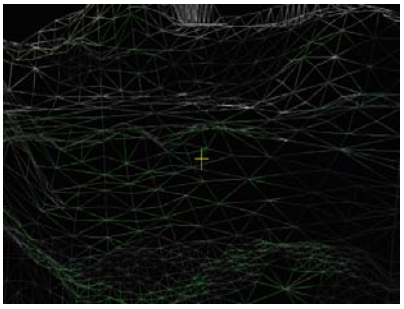

Fig.13 Terrain rendering of proposed algorithm
图 13 本文算法地形效果图

Fig.14 Generation and amplification of cracks
图 14 裂缝的产生与放大

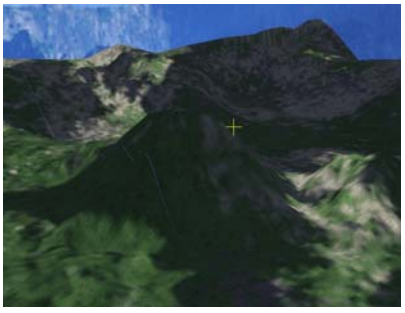

Fig.15 Texture rendering of terrain roaming
图 15 漫游后地形纹理效果图

从仿真结果可以看出，用规则格网得到的地形图，不管是在平坦区域还是离视点较远的区域，都是采用同样的精确度进行渲染，于是增加了许多不必要的三角形，降低了渲染地形的效率。而本文给出的基于四叉树视点相关的动态 LOD 地形渲染算法，首先根据数学模型对原始格网进行简化，在渲染地形时根据视点在场景中的位置，对可视区域进行裁剪和数据组织调度，然后通过设定合适的距离调节因子 C 和粗糙度调节因子 c，根据视点到节点的距离等因素，选择合适的节点分辨率评价函数，渲染地形时最大限度地减少所渲染的三角形数目。

3 结论

仿真实验结果表明，使用本文渲染算法绘制地形时，在较远地区能够减少三角形的绘制量，提高渲染地形的效率，且对地形的逼真度影响较小；若是平坦地区或者地形起伏变化缓慢的区域所占比重较大，本文算法绘制三维地形的效率更高。今后还需对规则格网模型的简化算法、数据存储结构与组织方式等方面继续分析研究，选择出更合适的模型简化算法、数据调度方式、效率更高的渲染算法来实现地形的可视化，尽可能地降低渲染算法的时间和空间开销。

参考文献：

[1] 卞海红,王峰. 基于三维 GIS 的地形可视化研究及实现[J]. 计算机技术与发展, 2006,16(7):230-233. (BIAN Haihong, WANG Feng. Research and implementation of terrain visualization based on 3D-GIS[J]. Computer Technology and Development, 2006,16(7):230-233.)

[2] HOPPE H. Smooth view dependent Level-of-Detail control and its application to terrain rendering[C]// IEEE Visualization'98. Durham USA:[s.n.], 1998:35-42.

[3] Duchaineau M,Wolinsky M,Sigeti D,et al. Roaming terrain: real-time optimally adapting meshes[C]// IEEE Visualization'97. USA:IEEE Xplore, 1997:81-88.

[4] 张兵强,张立民,李垠广,等. 分块 LOD 连续非线性分布的地形渲染[J]. 计算机工程与应用, 2013,49(4):213-218. (ZHANG Bingqiang,ZHANG Limin,LI Yinguang,et al. Terrain rendering of blocks LOD continuous non-linear distribution[J]. Computer Engineering and Applications, 2013,49(4):213-218.)

[5] 刘良彪,吴晓红,王正勇,等. 基于 COM 组件技术的岩心图像在线三维重建[J]. 太赫兹科学与电子信息学报, 2011, 9(6):770-773. (LIU Liangbiao,WU Xiaohong,WANG Zhengyong,et al. Online three-dimensional reconstruction of core

images based on COM component technology[J]. Journal of Terahertz Science and Electronic Information Technology, 2011,9(6):770-773.)

[6] 孔川,罗大庸. 利用动态多分辨率 LOD 技术的地形简化研究[J]. 计算机工程与应用, 2010,46(27):156-159. (KONG Chuan,LUO Dayong. Research of simplification on terrain using LOD techniques of dynamic multi-resolution[J]. Computer Engineering and Applications, 2010,46(27):156-159.)

[7] 王芳,丛文静,祝海涛. 基于顶点法向量重要度的模型简化算法研究[J]. 计算机与数学工程, 2011,39(7):6-8. (WANG Fang,CONG Wenjing,ZHU Haitao. Research of simplification algorithm for modeling based on normal important degree of triangular facets[J]. Computer & Digital Engineering, 2011,39(7):6-8.)

[8] 吴颖,张新家,茹芬. 基于四叉树分割的连续 LOD 漫游地形绘制[J]. 计算机技术与发展, 2011,21(4):5-8. (WU Ying, ZHANG Xinjia,RU FEN. Continuous LOD roaming terrain rendering based on quad-tree division[J]. Computer Technology and Development, 2011,21(4):5-8.)

[9] 张剑飞,王艳涛,程杰. 大规模三维地形的生成和漫游[J]. 哈尔滨理工大学学报, 2010,15(2):28-30. (ZHANG Jianfei, WANG Yantao,CHENG Jie. Large scale 3D terrain's generation and walkthrough[J]. Journal of Harbin University of Science and Technology, 2010,15(2):28-30.)

[10] 林炎光,孙红胜,岳春生. 一种基于嵌入式地形的三维数据建模与调度方法[J]. 测绘科学, 2010,35(2):99-101. (LIN Yanguang,SUN Hongsheng,YUE Chunsheng. A method of modeling and scheduling of three-dimensional topographical data on the basis of embedded system[J]. Science of Surveying and Mapping, 2010,35(2):99-101.)

[11] 曾艳阳,康凤举,杨慧珍. 自适应多特征融合的真实感地形快速绘制[J]. 中国图像图形学报, 2013,18(6):724-729. (ZENG Yanyang,KANG Fengju,YANG Huizhen. Fast rendering of realistic terrain based on adaptive multiple features fusion[J]. Journal of Image and Graphics, 2013,18(6):724-729.)

[12] 吴慧欣,苏锦旗,薛慧锋. 一种视点相关的大规模地形快速实时生成算法[J]. 计算机应用研究, 2008,25(10):3081-3084. (WU Huixin,SU Jinqi,XUE Huifeng. View-dependent real-time rending algorithm for large-scale terrain[J]. Application Research of Computer, 2008,25(10):3081-3084.)

作者简介:

郭雪峰(1987–),男,河南省商丘市人,在读硕士研究生,主要研究方向为嵌入式系统与智能信息处理.email:gxf870422@126.com.

孙红胜(1969–),男,湖北省长阳县人,副教授,主要研究方向为测控系统中的信号处理.

岳春生(1966–),男,山西省忻州市人,教授,主要研究方向为智能信息处理、嵌入式系统等.