

GPU 细分着色器中的地形无缝自适应细分

王文博,殷 宏,解文彬,张绪亮

(解放军理工大学 指挥信息系统学院,江苏 南京 210007)

摘 要: 为了进一步提高大规模地形渲染的效率和真实感,提出一种利用 GPU 细分着色器进行自适应细分的 LOD 地形算法。传统细分方法在顶点着色器中进行,需要预先计算细分模板且裂缝处理较为复杂,在实时交互过程中地形的细分效率并不高。本算法首先利用固定网格投射的方法得到地形的粗糙采样网格,节省了视锥体裁剪过程,并且减少了裂缝出现的可能性。其次,在细分控制着色器中利用插值点的屏幕投影误差作为误差度量方式,不断逼近误差阈值。在此过程中,采用细分等级测度的平滑插值对误差计算过程进行修正,保证了误差度量的单调性。最后,基于地形三角形各边的细分等级进行网格三角形无模板的无缝自适应细分。实验结果表明,算法改善了网格的密度分布,与传统细分方法相比效率更高。

关键词: GPU; 自适应细分; 固定网格投射; 无缝细分

中图分类号: TP391.9

文献标识码: A

文章编号: 1673-629X(2015)12-0105-04

doi: 10.3969/j.issn.1673-629X.2015.12.024

Real-time Terrain Tessellation on GPU Using Tessellation Shaders

WANG Wen-bo, YIN Hong, XIE Wen-bin, ZHANG Xu-liang

(Institute of Command Information and System, PLA University of Science and Technology,
Nanjing 210007, China)

Abstract: In order to further improve the efficiency and reality of large-scale terrain rendering, an adaptive subdivision algorithm using GPU is proposed. The traditional method which tessellation occurred in vertex shader, needs a large number of tessellation templates and complicated treatment to cracks. Subdivision process of terrain in real time interactive process is not very efficient. To get the coarse grid of terrain, persistent grid mapping method is used. In the tessellation control shader, with screen projection error metric of interpolation point, each side of triangle is seamless adaptive subdivided. In the process, use the smooth interpolation value of subdivision level measure to modify calculation process of error, ensuring the monotonicity of error measurement. Finally, texture map is mapped to terrain using texture coordinate in the fragment shader. Experimental results show that the algorithm improves the density distribution of the grid, which is more effective than traditional adaptive subdivision method.

Key words: GPU; adaptive subdivision; persistent grid mapping; seamless tessellation

0 引 言

大规模地形渲染在大型计算机游戏、虚拟现实、军用仿真等领域一直以来都有着十分广泛的应用。在三维场景中对地形进行实时更新与渲染是计算机图形学的一大挑战和研究热点。地形数据的规模巨大和图形硬件处理能力的不足是限制实时地形渲染效率的两个主要方面,因此,简化地形模型^[1]与面向硬件进行绘制成为了地形渲染的两个优化方向。

细节层次(Level Of Detail, LOD)模型能够根据视点与地形的相对位置选择不同细分级别的网格和对应

精度的高程数据渲染地形,有效解决了大规模地形直接渲染效率低下的问题,因此,在地形算法中多采用此模型对地形进行简化。早期地形算法中面向 CPU 的 LOD 算法,通过精确的误差度量在 CPU 中删减不必要的网格顶点,有效降低实时渲染时需要传输到 GPU 中的数据规模^[2]。随着图形硬件的不断发展,GPU 的计算和渲染能力得到大幅增强。相比 CPU 的串行结构,采用并行架构的 GPU 更适合地形渲染中大规模数据的计算与处理。国内外一些学者通过对地形分块处理、采用更适合批量处理的三角形集等方法^[3-6],减少

收稿日期:2015-03-02

修回日期:2015-07-02

网络出版时间:2015-11-04

基金项目:国家自然科学基金资助项目(70971137);国家社科基金军事学项目(13QJ004-098)

作者简介:王文博(1992-),男,硕士研究生,研究方向为虚拟现实;殷 宏,教授,博士,研究方向为虚拟现实、三维可视化。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20151104.0953.068.html>

提交的渲染批次,有效利用了 GPU 的强大并行处理能力。

可编程着色器的出现赋予了渲染管线数据处理流程更多的灵活性,将地形算法的部分或全部过程放入 GPU 中^[7]进行处理可以更加有效地利用 GPU 的显存与运算资源。李白云等^[8]将四叉树的构建与地形遍历过程交由着色器完成,解放了 CPU 的内存占用,降低了 CPU 与 GPU 之间的通信消耗。聂俊岚等^[9]将地形组织为 Patch-Block 两级,并分别通过地形块的屏幕投影误差进行细分等级控制,并结合硬件细分技术减轻了网格细分层次切换时 CPU 的压力,但是分级进行误差度量,计算过程较为烦琐。

Livny 等^[10]利用光路可逆的原理,模拟视点发出光线通过视锥体近平面的固定网格对地形进行投射 (Persistent Grid Mapping, PGM),在顶点着色器 (Vertex Shader, VS) 中根据网格发出的光矢量与地面相交进行采样。由于 PGM 方法视点相关的特性,可以避免进行视锥体裁剪等过程,而且随着网格顶点到视点距离不同自然形成 LOD 效果。但是相同的网格密度无法在绘制所有地形时都适合。基于粗糙采样网格,在着色器中进行自适应细分^[11-12]能够有效改善网格密度分布。王旭等通过改进自适应细分过程中的细分模式,有效降低了细分模板数量,但是存在不同层次细分程度相差过大等问题。

文中算法首先采用 PGM 方法在顶点着色器中产生地形的粗糙采样网格;其次针对网格三角形各边分别进行视点与地形粗糙度相关的屏幕投影误差的度量,将得到的细分层次传输至细分计算着色器中控制细分过程;最后根据精细网格顶点对应的纹理坐标在片元着色器中进行纹理贴图,实现了一种利用 GPU 细分着色器进行网格自适应细分的 LOD 地形算法。

1 粗糙网格采样

文中采用固定网格投射^[10]方法对地形进行粗糙采样,同样的网格顶点间隔有利于进行硬件细分、裂缝处理等过程。固定网格投射利用光线投射的思想,从视点发出光线,光线透过屏幕空间网格照射到地面上,根据视点的方向与位置,投射的光线与地面相交在不同的位置。根据光路可逆的原理,光线投射到的区域为可见区域。固定网格投射直接与视锥体相关,可以省去传统 LOD 算法中视锥体剪裁的步骤。利用固定网格投射对地形进行采样,在顶点着色器中计算出投射到地面处顶点的世界空间坐标,输出为视点相关的粗糙网格,将粗糙网格送入细分着色器中进行下一步的自适应细分。

如图 1 所示,设地面为高度 H ,与 XOY 面平行的

平面,从视点 V 发出的任意一条光线穿过屏幕空间网格上 G 点与地面相交。光矢量 \overrightarrow{VG} 在地面投射位置 E 点的坐标可通过式(1)求得:

$$\begin{cases} E_{xy} = V_{xy} + \frac{V_z - H}{V_z - G_z}(G_{xy} - V_{xy}) \\ E_z = H \end{cases} \quad (1)$$

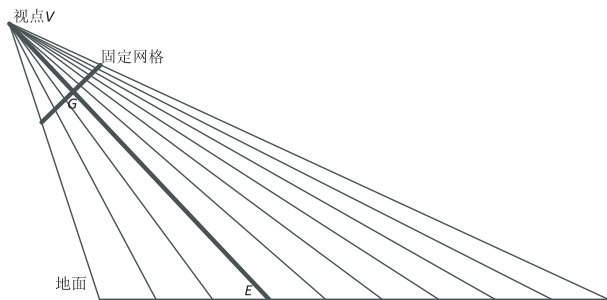


图 1 固定网格投射

2 屏幕投影误差控制的细分过程

2.1 三角形细分等级的计算

在粗糙网格形成后,以网格三角形为一个细分单位进行细分等级的计算。王旭等^[13]采用计算插值点处世界空间误差确定细分层次的方式逼近误差,并通过分裂限制因子的方式弥补未考虑视点因素的不足。杨堂^[14]对其进行改进,直接采用插值点处屏幕投影误差确定整个三角形的细分层次,但是仅对斜边中点进行计算,得到的细分等级不够精确。文中在细分控制着色器中对三角形三条边通过插值点处屏幕投影误差进行细分等级的计算。

由于在细分计算着色器中采用 equal_spacing 这种整数均分的细分模式,在细分等级为 i 时,产生在各边的新顶点分别位于 i 等分点, $i \in N$ 且 $2 \leq i \leq \max TessLevel$ 。各等分点处的世界空间误差可通过式(2)求出:

$$\begin{cases} \varepsilon_w^{ij} = (1 - \alpha)H_{V_0} + \alpha H_{V_i} - height_{ij} \\ \alpha = i/j \end{cases} \quad (2)$$

其中, ε_w^{ij} 为第 j 个 i 等分点的世界空间误差; H_{V_0} 、 H_{V_i} 分别为两个端点的采样高度; $height_{ij}$ 为第 j 个 i 等分点处的采样高度。

ε_w^{ij} 对应的屏幕投影误差可通过式(3)求得:

$$\varepsilon_m^{ij} = \frac{W \varepsilon_w^{ij}}{2 \tan(\theta/2) D_{ij}} \quad (3)$$

其中, W 为投影平面横向上的像素数量; θ 为横向的视锥体角度范围; D_{ij} 为第 j 个 i 等分点到视点的距离。

对于细分等级 i ,取 $i-1$ 个等分点处顶点的插值屏幕投影误差的最大值为细分边的细分等级测度: $\varepsilon^i = \max(\varepsilon_m^{i1}, \dots, \varepsilon_m^{ij}, \dots, \varepsilon_m^{i(i-1)})$ 。将细分等级测度 ε^i 与

阈值 τ 相比较,若 $\varepsilon > \tau$,则进行等级 $i+1$ 细分等级测度 ε^{i+1} 的计算与比较;反之,则此边的细分等级为 i 。

对于现有的图形硬件, $\max\text{TessLevel}$ 最大可以达到 64。但是随着细分层次的不加深,细分过程对算法性能的制约会越来越大,导致瓶颈转移至硬件细分过程,因此为确保算法具有较高的实时性,一般 $\max\text{TessLevel}$ 并未设定为 64。

2.2 细分等级测度的平滑插值

通过插值点屏幕投影误差判断细分程度的方法利用了曲面细分的思想,通过细分不断逼近误差阈值。但是计算过程中不同等级的细分等级测度之间没有联系,并不存在单调递减的必然性。若 $\varepsilon^i = \max(\varepsilon^2, \dots, \varepsilon^i, \dots, \varepsilon^{\max\text{TessLevel}})$ 且 $\varepsilon^i > \tau$,则会影响等级 i 之后细分等级测度的计算。

因此,文中采取平滑插值的方法避免此种问题。对于细分等级 $i = 2^k, k \in N$,让各插值点的采样高度 height_{2^k} 参与到细分等级为 2^{k+1} 时细分等级测度的计算当中;而对于 $2^k < i < 2^{k+1}$,则采取由等级 2^k 和 2^{k+1} 的细分等级测度平滑插值的方法得到等级 i 的细分等级测度:

$$\varepsilon^i = \varepsilon^{2^k} - \frac{i - 2^k}{2^{k+1} - 2^k} (\varepsilon^{2^k} - \varepsilon^{2^{k+1}}) \quad (4)$$

2.3 基于边细分因子的无缝细分

传统自适应细分算法^[11-12,14]在顶点着色器中进行模板匹配与顶点坐标的计算,细分实质上是选择模板后基于模板计算插值后顶点坐标的过程。由于在着色器中,网格三角形的细分过程独立进行,无法得知相邻三角形细分层次是否一致,对于可能出现的 T 形连接与“裂缝”的处理也更加困难。

文中算法采用在细分计算着色器中进行三角形细分过程,细分过程可以通过四个细分因子控制。如图 2 所示,对于 $\triangle ABC$,分别对边 AC 、 CB 、 BA 进行细分等级计算,并根据细分等级对三条边进行细分。为综合考虑三角形覆盖区域粗糙度,对边细分因子取均值作为三角形内部细分因子。对于相邻不同层次的三角形,由于共享边两端顶点坐标与局部 (u, v, w) 坐标一致,在同一视点下得到的细分测度是相同的。如图 2 中相邻 $\triangle ABC$ 与 $\triangle BCD$,虽然细分过程独立进行,但是由于采用基于边细分因子的策略进行细分,得到的是无缝的精细网格三角形。共享边 BC 的细分因子在 $\triangle ABC$ 与 $\triangle BCD$ 中分别进行计算,内插顶点与位置相同,有效避免了“裂缝”的出现。

对于 $\triangle ABC$,通过三个顶点世界坐标与插值顶点的局部质心坐标 (u, v, w) 计算得到新顶点的世界空间坐标:

$$\text{newPoint} = P_A \times u + P_B \times v + P_C \times w \quad (5)$$

其中, P_A, P_B, P_C 分别为三个顶点的坐标。由于粗糙网格并未对高度分量进行采样,因此得到的新顶点坐标 newPoint 的高度分量值依然为零。结合坐标在细分计算着色器中对高度图进行检索,将得到的高度值赋给新顶点坐标的高度分量。

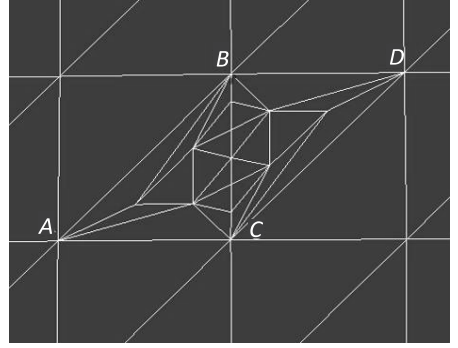


图 2 无缝细分

3 纹理坐标的计算

由于在细分过程中根据输入的网格三角形分裂形成了新的子三角形,因此纹理坐标的计算在精细网格形成后,统一在片元着色器中进行。粗糙网格经过细分着色器之后得到精细网格,将精细网格输出到片元着色器中进行纹理坐标的计算。在片元着色器中,通过顶点对应的纹理坐标检索纹理图得到对应纹理值。纹理值的计算由式(6)得出:

$$\text{fragColor} = \text{texture}(\text{colorMap}, \text{texCoord}.xy) \quad (6)$$

其中, colorMap 为地形纹理图; texCoord 为顶点对应的纹理坐标。

4 实验

根据算法设计,对变量取不同值分别在 PC 机上进行了实验。PC 机的硬件配置为:CPU Intel Core i7 Q720 1.60 GHz, 4.00 GB RAM, ATI Radeon HD 5470 显卡,显存为 1 GB。软件环境为:Windows XP SP3, Visual Studio 2010,渲染引擎采用 OpenSceneGraph,着色器语言为 HLSL。实验中采用的数据为 Pudget Sound 数据,DEM 数据和纹理数据大小同为 16 384×16 384,屏幕分辨率为 1 024×768。

在相同视点位置下,在阈值 τ 取不同值时对地形进行自适应细分。图 3~5 分别是 $\tau = 22$ 、 $\tau = 10$ 、 $\tau = 2$ 时的地形渲染网格图,图 6 为 $\tau = 2$ 时的地形渲染纹理图。

由三幅网格图可以看出,当 τ 取值较大时,地形的细分程度较小;随着 τ 取值的减小,地形粗糙度较大的山地细分程度加深,而平坦地形保持了较小的细分程度,避免了平坦区域过度细分造成的资源浪费;当 τ 取值过小时,在平坦区域造成了过度细分。在文中的实

验环境下, τ 取值为 8 时细分效果最好。

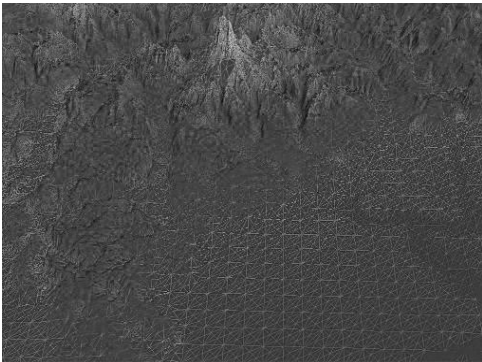


图 3 $\tau = 22$ 时的细分网格图

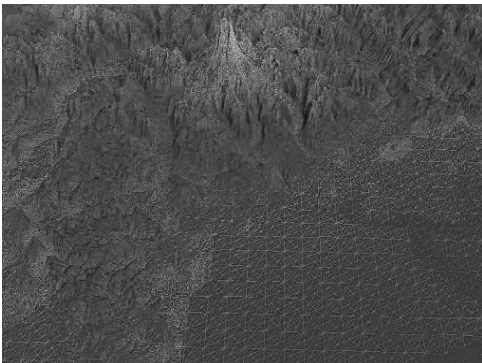


图 4 $\tau = 10$ 时的细分网格图

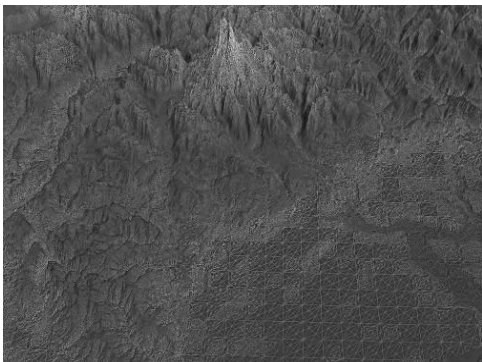


图 5 $\tau = 2$ 时的细分网格图

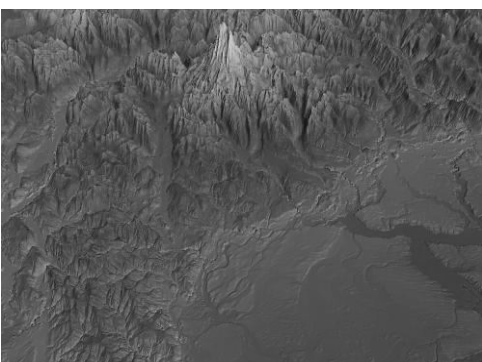


图 6 $\tau = 2$ 时的渲染纹理图

表 1 给出了文中算法、文献[12]中算法和 PGM 算法的性能比较。

在对应最大细分等级下,文中算法相比文献[12]中基于顶点着色器进行自适应细分的算法在效率上有

了明显提高。PGM 算法并没有细分过程,因此在对应的采样密度下与文中算法比较。文中算法在大部分情况下渲染一帧所花费的时间较少,在自适应细分层次较高,采样网格边长较大的情况下,与 PGM 算法相当。

表 1 算法性能比较

最大 细分 等级	采样 网格 边长	文中算法		文献[12]中算法		PGM 算法		
		三角 形数量 / K	帧速 /fps	对应 细分 等级	三角形 数量 / K	帧速 /fps	三角形 数量 / K	帧速 /fps
7	50	365	146		320	100	320	100
	70	715	108	3	627	88	627	88
	90	1 183	84		1 037	54	1 037	70
14	40	810	111		819	85	819	85
	60	1 822	62	4	1 843	45	1 843	54
	80	3 238	37		3 277	31	3 277	37

5 结束语

文中基于 GPU 实现了一种单 pass 的网格自适应细分算法,相比多 pass 算法中先由将顶点的细分级别渲染至纹理,并由 CPU 回读的做法,有效降低了 CPU 与 GPU 的信息交换。算法在粗糙采样网格基础上,利用细分着色器中对网格三角形进行基于边细分等级的细分,相比针对三角形整体的细分方法,在裂缝处理、不同层级间的平滑过渡更有优势。通过实验结果表明,文中算法相比传统自适应细分方法性能更佳,与固定网格投射方法相比同样具有一定提高。

参考文献:

[1] 邹海,陈保柱. 三维动态多分辨率地形模型的研究[J]. 计算机技术与发展,2014,24(5):239-242.

[2] 陈希亮,曹雷,崔平. 基于 ROAM 算法的实时地形可视化研究[J]. 计算机技术与发展,2013,23(1):237-241.

[3] Ulrich T. Rendering massive terrains using chunked level of detail control[J]. SIGGRAPH Course Notes,2002,3(5):162-166.

[4] de Boer W. Fast terrain rendering using geometrical mipmapping [EB/OL]. 2000. <http://www.ipcode.com/tutorials/geomipmaps.pdf>.

[5] 张兵强,张立民,艾祖亮,等. 面向 GPU 的批 LOD 地形实时绘制[J]. 中国图象图形学报,2012,17(11):582-588.

[6] 宫晓辉,温慧明,于卓. 基于 Clipmap 的海量地形及纹理实时绘制方法[J]. 计算机技术与发展,2012,22(10):22-26.

[7] 刘浩,赵文吉,段福洲,等. 利用可编程 GPU 实现大规模地形场景的高性能漫游[J]. 计算机应用研究,2011,28(10):3775-3777.

的隐藏传输带来完整性影响。

针对上述的第一种情况,可以利用网络协议簇中多层协议的协同合作来实现隐通道信道容量的提升,即同时利用一个数据包中的多层协议中的可用对等元素编码进行信息的隐藏传输,通信双方需要就数据分片、重组和校验进行设计和实现,如图 3 所示。针对第二种情况,可以使用冗余传输的方式来防止数据丢包造成的隐藏信息完整性问题,如添加校验机制和重传机制等。

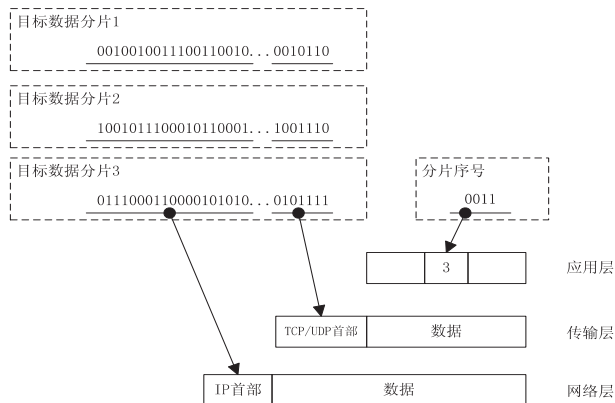


图 3 利用多层协议协同式构建隐通道

5 结束语

文中从隐通道的概念、分类以及相应的弊端开始,阐述了网络协议对等元素和类对等元素的定义,并就利用对等元素进行编码的方式进行了研究和描述,同时,着重研究网络协议中几种可利用对等元素构建隐通道的协议及其相应的构建方式,并通过实验仿真实现了这些隐通道的发送和接收,对其信道容量进行了比对和总结,最后给出了对等元素编码隐通道的协同式增强方法。文中系统地归纳总结和理论提炼了一种隐通道构建方法,对更系统和理论地研究隐通道构建和应用的方式具有一定的参考和借鉴意义。

参考文献:

[1] Chaum D L. Untraceable electronic mail return addresses, and digital pseudonym[J]. Communications of the ACM,1981,24(2):84-88.

[2] Tsai C R, Gligor V D, Chandrasekaran C S. A formal method for the identification of covert storage channels in source code[C]//Proc of IEEE symposium on security and privacy. [s. l.]:IEEE,1987:74-86.

[3] 王保华,马新强,李丹宁,等.安全数据库隐藏通道的标识技术与实例分析[J].计算机技术与发展,2007,17(2):233-235.

[4] 王永吉,吴敬征,曾海涛,等.隐蔽信道研究[J].软件学报,2010,21(9):2262-2288.

[5] 徐杰峰.基于TCP/IP协议的网络隐藏通道研究[J].北京邮电大学学报,2003,26(S):144-150.

[6] 陈夕华.TCP/IP协议上隐藏通道机制的研究及实现[D].上海:上海交通大学,2005.

[7] 郭强.基于时延特性的网络隐藏信道的研究[D].上海:上海交通大学,2008.

[8] 张令通,罗森林.基于TCP协议首部的网络隐藏通道技术研究[J].计算机工程与科学,2014,36(6):1072-1076.

[9] 王相林,洪伟烨.网络隐通道构建技术的研究[J].杭州电子科技大学学报,2009,29(6):29-32.

[10] 王永杰,刘京菊.基于DNS协议的隐藏通道原理及性能分析[J].计算机工程,2014,40(7):102-105.

[11] Forouzan B A. TCP/IP协议族[M].第4版.王海,张娟,朱晓阳,等.北京:清华大学出版社,2011.

[12] 常晶.基于HTTP隧道技术的保密通信关键技术的研究[D].北京:北京邮电大学,2013.

[13] 邹昕光.基于FTP协议的命令序列隐藏信道[J].哈尔滨工业大学学报,2007,39(3):424-426.

[14] 解培岱.面向内容过滤的协议扩展技术研究与实现[D].长沙:国防科学技术大学,2008.

[15] 倪红彪.基于IPv6的互联网络安全研究[J].信息安全与技术,2014,5(2):28-30.

(上接第 108 页)

[8] 李白云,赵春霞.GPU实时构建四叉树的快速地形渲染算法[J].计算机辅助设计与图形学学报,2010,22(12):2259-2264.

[9] 聂俊岚,张精卫,郭栋梁.细节平滑过渡的GPU构网地形渲染[J].小型微型计算机系统,2014,35(1):121-125.

[10] Livny Y, Sokolovsky N, Grinshpoun T, et al. A GPU persistent grid mapping for terrain rendering[J]. The Visual Computer, 2008,24(2):139-153.

[11] Boubekeur T, Schlick C. A flexible kernel for adaptive mesh refinement on GPU[J]. Computer Graphics Forum, 2008,27

(1):102-114.

[12] Lorenz H, Döllner J. Dynamic mesh refinement on GPU using geometry shaders[C]//Proceedings of the 16th international conference in Central Europe on computer graphics, visualization and computer vision. Campus Bory: UNION Agency-Science Press, 2008:97-104.

[13] 王旭,杨新,王志铭.在GPU上实现地形渲染的自适应算法[J].计算机辅助设计与图形学学报,2010,22(10):1741-1749.

[14] 杨莹.改进固定网格和自适应细分实现地形实时绘制[J].计算机工程与设计,2013,34(11):3939-3943.

GPU细分着色器中的地形无缝自适应细分

作者：[王文博](#)，[殷宏](#)，[解文彬](#)，[张绪亮](#)，[WANG Wen-bo](#)，[YIN Hong](#)，[XIE Wen-bin](#)，[ZHANG Xu-liang](#)

作者单位：[解放军理工大学 指挥信息系统学院, 江苏 南京, 210007](#)

刊名：[计算机技术与发展](#)

英文刊名：[Computer Technology and Development](#)

年，卷(期)：2015, 25(12)

引用本文格式：[王文博](#). [殷宏](#). [解文彬](#). [张绪亮](#). [WANG Wen-bo](#). [YIN Hong](#). [XIE Wen-bin](#). [ZHANG Xu-liang](#) [GPU细分着色器中的地形无缝自适应细分](#) [期刊论文] - [计算机技术与发展](#) 2015 (12)