基于远程渲染的移动三维 GIS 构建方法

赖冬林[1,2] 张 丰[1,2] 杜震洪[1,2] 刘仁义[2]
[1](浙江大学浙江省资源与环境信息系统重点实验室 浙江 杭州 310028)
[2](浙江大学地理信息科学研究所 浙江 杭州 310027)

摘 要 随着移动互联网的发展,移动 GIS 应用逐渐从二维 GIS 发展为三维 GIS。受制于 3D 场景数据量大、移动设备图形渲染能力低等因素,常用的移动三维 GIS 渲染效果较差,交互低效。结合预处理策略、客户端预载入策略与远程渲染技术提出一种高效的移动三维 GIS 构建方法。该方法利用 QEM 算法在服务器端预处理模型,减少移动端计算量,通过移动端预载入策略,减少帧之间的延迟,提高移动端的实时性,具有较好的渲染效果、高效的交互性和较低的耗电量。最后通过实验证明该方法的可用性。该方法适用于移动设备甚至可穿戴设备中 GIS 应用的构建。

关键词 远程渲染 移动 GIS 三维 GIS 网格化简

中图分类号 TP39 P208 **文献标识码** A DOI: 10.3969/j.issn.1000-386x.2015.07.047

A REMOTE RENDERING-BASED CONSTRUCTION METHOD FOR MOBILE 3D-GIS

Lai Donglin[1,2] Zhang Feng[1,2] Du Zhenhong[1,2] Liu Renyi[2]
[1](*Zhejiang Provincial Keylab of GIS, Zhejiang University, Hangzhou 310028, Zhejiang, China*)
[2](*Institute of Geographic Information Science, Zhejiang University, Hangzhou 310027, Zhejiang, China*)

Abstract As the mobile Internet developing, mobile GIS application has been gradually developed from 2D-GIS to 3D-GIS. Common methods of mobile GIS have poor rendering effect and low efficient interactivity due to restrained by large data amount of 3D scene and low graphics rendering capability of mobile devices. By combining the strategies of preprocessing, pre-load for clients and remote rendering technology, this paper presents an efficient construction method for mobile 3D-GIS. It uses QEM (quadric error metrics) algorithm to pre-process model at the server end to reduce the computation load at the mobile end, and reduces the delay between the frames through pre-load strategy at mobile end and improves real-time property of the mobile end, it has better rendering effect, efficient interactivity and lower power consumption. Finally the applicability of the method is proved by the experiment. This method is adapted to the construction of GIS applications for mobile devises or even the wearable devices.

Keywords Remote rendering Mobile GIS 3D-GIS Meshes simplification

0 引 言

随着移动互联网的发展,移动三维 GIS 及其渲染方式是目前移动 GIS 领域的研究热点[1]。常见移动三维 GIS 的构建方式有两种:① 利用 WorldWind、Unity3D、M3D 等已有的移动三维框架针对 GIS 需求进行改造[2,3];② 在硬件支持的图形绘制接口如 OpenGL ES 等 API 基础上开发特定的移动三维 GIS 应用[3-5]。这些构建方式都使用本地渲染作为渲染方式,服务端只用来存储 GIS 空间数据[2-5]。受限于客户端硬件计算能力,渲染效果均较低,交互性差[5]。

桌面领域的三维渲染随着云计算的发展从单机渲染逐渐跨步到集群渲染,例如利用 MapReduce 等技术进行一亿数量级三角形的超大规模三维场景渲染[6]。同时,借助面向服务构架的思想,远程渲染成为 Web 3D、视频游戏等的主要渲染方式[7-9]。远程渲染服务解决了①计算能力有限的客户端对高精度三维模型的需求、②特殊三维数据渲染的安全需求[10]。

目前国内外远程渲染主要应用在以下两个方面:①针对带宽充足可靠性高的有线网络下面向桌面 PC 的远程渲染服务[6-8];②以视频为媒介的云游戏服务[9]。面向桌面 PC 的远程渲染系统只需考虑可靠网络下的数据传输即可,例如“中国数字图书馆”项目[7]和上海市多媒体公共服务平台[8]。面向云游戏的远程渲染系统保证网络延迟在显示帧率之下,通过提供 12 fps /24 fps 的视频显示[9]。

面向移动的远程渲染目前以图像流为主,存在一定延迟和缺乏可交互性等问题[11],不适用于交互频繁的 GIS。为了解决移动端渲染效率问题和可交互性问题,本文从远程渲染模型入手,分析面向移动三维 GIS 的可交互性远程渲染技术,提出了以 QEM 算法预先处理 GIS 空间数据的远程渲染网格预处理策略;

收稿日期:2014-01-21。国家自然科学基金项目(41001227,41101356);浙江省自然科学基金项目(401006);浙江省科技厅重点项目(2010C33146);中国国家博士后基金项目(20100481405)。赖冬林,硕士,主研领域:移动 GIS 理论及应用。张丰,副教授。杜震洪,副教授。刘仁义,教授。

利用 Markov Chains 训练视点历史位置的视点路径预测策略，提出了移动端多重请求策略的客户端预载入策略; 并利用上述研究内容创新性地提出了基于远程渲染的移动三维 GIS 的构建方法。依据此方法设计并实现一个面向移动三维 GIS 的可交互性远程渲染系统。在实验中，该系统表明本方法构建的移动三维 GIS 应用渲染效果和可交互性优于使用本地渲染方法的应用。

1　远程渲染

远程渲染是互联网环境下发展起来的一种剥离渲染部分和显示部分的渲染方式。参考文献[13]系统分析了网络环境下远程渲染的三种: 客户端方法、服务端方法和混合方法。①客户端方法以客户端渲染为主，服务端只负责场景描述数据的存储，该方法极大地简化了服务端的设计逻辑，不适用于瘦客户端环境; ②服务端方法与客户端方法相反，客户端只负责显示，所有渲染逻辑均由服务端负责，该方法能够适用于各异性客户端[11]，但增加了服务端的工作量、可交互性差，是目前远程渲染主要方式; ③混合方法即服务端与客户端同时承担一部分渲染工作，可交互性高，难点主要在于渲染工作的划分。

针对移动 GIS 的特点，混合方法能够满足显示环境弱计算性和对良好的交互性的需求。为此本文采用混合方法设计远程渲染框架，将框架分为客户端、网络和远程渲染服务器三部分。客户端视图更新时打包视点参数通过无线网络向服务器请求数据，服务器解析参数，通过路径预测、数据提取、遮挡裁剪、数据分组等操作，实现场景的渲染和数据化简压缩，通过会话返回给客户端，客户端接收数据进行场景重构和纹理贴图，完成渲染过程(如图 1 所示)。

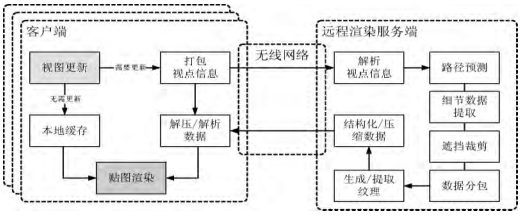


图 1　面向移动三维 GIS 的混合方法远程渲染模型

一次渲染过程所耗费的时间 T_{frame} 包含网络请求时间 rtt、服务器响应时间 T_s 和客户端渲染时间 T_c。若传输数据量为 s，带宽为 B_w，则:

$$T_{frame} = T_s + T_c + 2 \times rtt = \frac{s}{B_w} \quad (1)$$

式中，rtt 依赖于网络拥堵情况，在不可靠的移动网络下往往表现较大的波动。因此本文将从以下三方面减少客户端的渲染延迟: 减少客户端处理时间、减少服务端处理时间，和提前载入。减少客户端处理时间即增加客户端缓存机制减少额外请求、让客户端仅负责场景重构，减少服务端处理时间即设计预处理服务器，预载入即通过路径预测提前计算所需数据、优化客户端请求策略等，因此将图 1 的模型更改为图 2 模型。

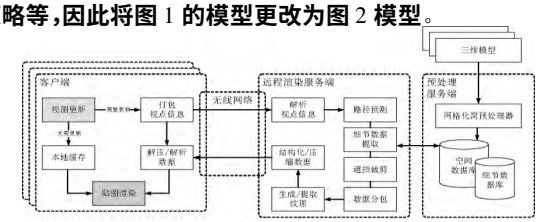


图 2　带预处理器(右)的远程渲染模型

2　服务端预处理

服务端预处理是将模型数据利用细节层次技术进行分层存储的过程。本文采用 QEM 算法作为模型的细节层次分层算法。

2.1　QEM 算法

Kho 与 Garland 在 2003 年改进了渐进网格边折叠算法，通过计算边折叠后的顶点与包含被折叠边的两个顶点的三角形的顶点的距离的平方和作为误差代价，提出了基于局部二次误差测量 QEM 为代价的边折叠算法[14]。对于顶点 u，定义其到 T_u 的平面的距离的平方和为顶点 u 的二次误差测量度:

$$\Delta(u) = \sum_{f \in T_u} d_f^2(u) = \sum_{f \in T_u} u^{\mathrm{T}}(K_f)u = u^{\mathrm{T}}\left(\sum_{f \in T_u} K_f\right)u \quad (2)$$

其中，$u = [u_x u_y u_z 1]^{\mathrm{T}}$，$f$ 表示空间中的平面:

$$ax + bx + cz + d = 0 \quad a^2 + b^2 + c^2 = 1$$

$$K_f = ff^{\mathrm{T}} = \begin{bmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & c^2 & cd \\ ad & bd & cd & d^2 \end{bmatrix}$$

令 $Q(u) = \sum_{f \in T_u} K_f$ 为顶点 u 的二次误差测量矩阵，显然，这是一个 4×4 的对称矩阵。因此对于$(v_s, v_t) \xrightarrow{ecol} v_u$ 的过程，边折叠的代价就是$\Delta(v_u) = v_u^{\mathrm{T}}\{Q(v_s) + Q(v_t)\}v_u$。因此 i 层的顶点的二次误差度量的矩阵可以由 $i+1$ 层的二次误差度量矩阵来表示。很显然，如果 v_u 离(v_s, v_t) 关联三角面的距离平方和越大，代价越大。

QEM 算法能够取得很好的简化效果，在适当降低精确度的同时保持与 Hoppe 方法的最大近似度(如图 3 所示，图中左上角为原始模型，拥有 5804 个三角形。第一排从左到右的三角形个数分别为 994、532，第二排从左到右为 248 和 64)[12]。

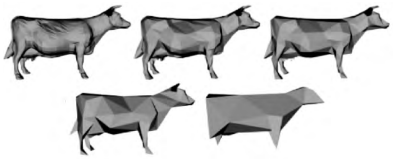


图 3　使用 QEM 算法得到的三维模型

2.2　预处理策略

在 QEM 的算法基础上构建了 QEM 引擎用于处理模型数据。首先，3D 模型数据读取(Wavefront 的标准三维模型格式或者 OSG 自定义文本存储格式)，原始纹理数据处理层、渐进网格生成(如图 4 所示)。该层先接收各类数据——包括三维模型数据 3ds Max 数据、ArcInfo 数据等，地形数据 TIFF、DEM 等，以及单纯的纹理贴图——并利用 OpenSceneGraph 工具转化成 OSG 格式，利用 QEM 引擎处理成 HLODs 数据，再对这些数据建立索引通过细节数据管理器保存到细节数据库和空间数据库中。图片纹理则在坐标校准之后通过纹理数据管理器保存到文件系统中。

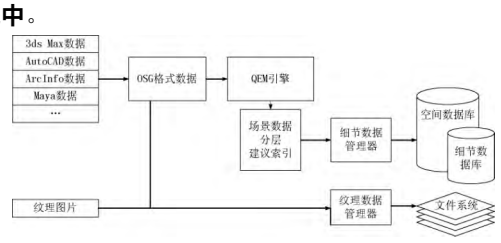


图 4　利用 QEM 引擎的三维模型数据预处理策略

3 客户端预载入

3.1 视点预测

对于路径前进预测常见的算法是使用马尔可夫链对历史数据进行训练，得到一定状态空间下的条件概率分布函数，再进行下次视点位置的预测(如图 5 所示，图中视点从 C1 运动到 C2，并预测出下一个视点位置和方向，然后在服务端预先计算所预测需要的网格，等待客户端的更新请求)。

对于视点，以视锥为研究模型，可能变化的有视点在场景空间坐标的为 $e=(x,y,z)$，以及视点方向上的单位向量 $\hat{e}$。为了兼顾计算效率和渲染效果，简化整个状态空间集合[15]。本文中，假设可能的变化值为 Δe 和 $\Delta\hat{e}$，那么整个预测值即求两者的期望，即 $E(\Delta e)$ 与 $E(\Delta\hat{e})$。

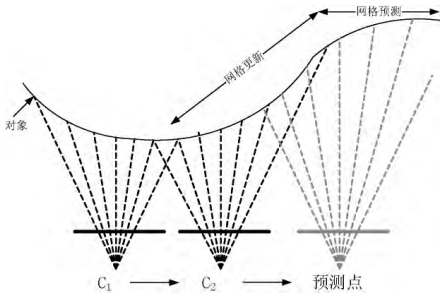


图 5 视点路径预测图

令当前时刻为 t，前一时刻为 $t-1$，后一时刻为 $t+1$。那么有:

$$\Delta e^t = e^t - e^{t-1} \tag{3}$$

$$E(e^{t+1}) = \sum_{t=0}^{t} P(e^i) \times \Delta e^i \tag{4}$$

依此类推可知 $E(\Delta\hat{e})$。

对于式(4)，计算复杂度为 $O(t)$，并且对于 $P(e)$，需要维护一个视点变化过程的顶点向量和概率集合，并且这个概率集合需要在运行时调整，降低计算效率。所以我们假定:

$$P(e^{t-i}) = \left(\frac{1}{2^{t+1}}\right)$$

所以:

$$\begin{aligned} E(e^{t+1}) &= \sum_{i=0}^{t} P(e^i) \times \Delta e^i = \sum_{i=0}^{t} \left(\frac{1}{2^{t-i+1}}\right) \times \Delta e^i \\ &= \frac{1}{2}\Delta e^t + \sum_{i=0}^{t-1} \left(\frac{1}{2^{t-i}}\right) \times \Delta e^i \\ &= \frac{1}{2}(\Delta e^t = E(e^t)) \end{aligned}$$

因此下一次的移动轨迹的期望只需要当前移动变量与当前的移动期望参与计算，时间和空间复杂度都是 $O(1)$。

3.2 客户端请求策略

在客户端方面，为了减少交互的延迟，需要弥补网络请求的时间差。我们认为一个交互中的时间差指的是两次操作(两次 Touch Up 事件)之间的时间间隔，包含视点信息请求时间、服务器处理时间、网络响应时间、本地视图渲染时间，如图 6 所示。

为了描述方便，我们忽略细节，将图 6 简化成图 7(a)。首先我们把客户端过程简化为人机交互阶段(Touch 交互)、获取更新等待时间、客户端数据解码与渲染时间。在普通模型下，获取更新等待时间和渲染时间会造成交互卡顿和延迟，为了解决这个问题，本文采用了预请求方式减少可能的交互延迟。

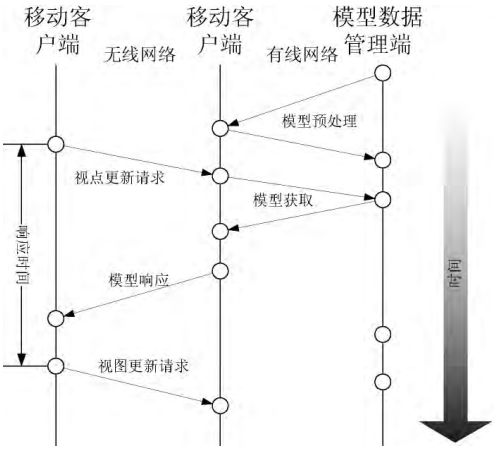
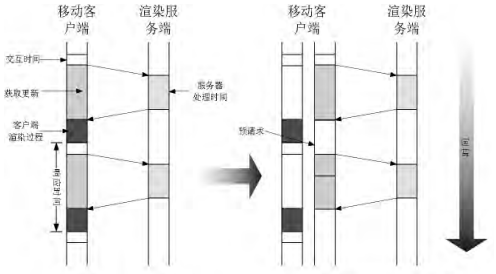
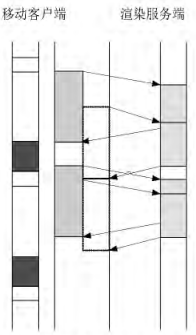


图 6 一个客户端请求流程

客户端通过和服务端同样的算法预测视点的下一步动作，并且在计算完成即向服务器进行请求(见图 7(b))。当前这种情况下只有一个请求队列，在频繁操作、特别是网络状况不好的情况下，如果预请求不符合要求，合法请求依旧需要等待预请求返回结果之后才能再请求。因此在此基础上，结合文献[16]在 PC 计算机上的并行处理策略，使用多个优先队列作为 UDP 请求队列进行异步请求(如图 7(c))。



(a) 简化的请求流程 (b) 带有预请求的请求流程



(c) 使用多重请求队列的请求流程

图 7 请求流程描述

4 系统实现与分析

4.1 系统构建

客户端采用基于 iOS 的 OpenGL ES 2 渲染程序，通过可编程管线构建。编程环境为 Xcode，客户端由负责网络通信的网络模块、负责显示和交互的显示模块、负责数据缓存提取存储的数据管理模块等三个模块组成。网络模块根据 4.2 节使用多个请求队列进行异步请求，在实验中帧与帧平均延迟均在 30 ms 左右，如果使用使用单队列，平均延迟 100 ~ 300 ms 左右(如图 8

所示,旋转视角变换三角形数量测试性能,从左到右三角形数分别为 7634、5010、8012)。

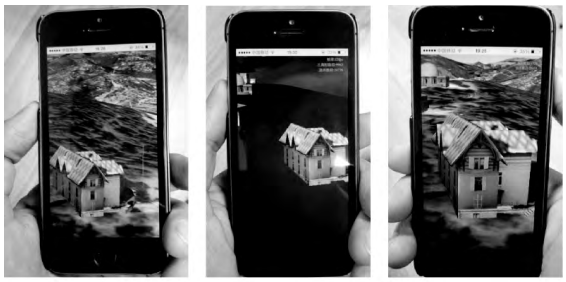


图 8　客户端展示结果

服务端构建采用 OpenSceneGraph 作为远程渲染的基本框架,在此框架下通过插件的模式实现了 QEM 引擎插件、数据库交互插件、网络请求插件。由于移动客户端本身的若连接性,为提高服务器的并发性能,实现中采用以下策略: 在客户端与服务器的交互中维持一个弱会话,即通过一个会话 ID 识别客户端,并采用短连接的形式,在客户端离线超时或者主动关闭会话时才认为连接结束。此策略能够实现更高的并发性能并且服务端在维持会话 ID 期间能实现客户端请求预处理。

4.2　实现结果与分析

本文所使用数据源包含三角形网格数 2 501 948,顶点数 4 837 550,纹理数 843。本文以 osgEarth for iOS 作为本地渲染的实验程序,与本研究提出的远程渲染模型进行对比实验。实验采用分组多次试验,每组实验测量根据操作的频率,至少收集 20 次数据。每次试验均在移动客户端上拖动/旋转场景,监控客户端渲染帧率、服务端和客户端的平均处理三角形数、服务端和客户端的平均处理顶点数、服务器的平均处理时间。远程渲染中,服务器平均处理一次渲染请求在 10 ~ 20 ms 之间,客户端帧率 11 ~ 46 fps 之间,客户端渲染的三角形占服务器处理三角形 5% ~ 15% 之间,对比本地渲染帧率平均增加 20% ~ 113%,客户端处理三角形数平均减少 13% ~ 97%。同等渲染效果下,交互流畅度远程渲染高于本地渲染。

4.3　与现有方法比较

(1) 与传统远程渲染对比

目前传统的远程渲染平台如 OpenGL VizServer、HP Remote Graphics Software 等解决方案均无在移动 GIS 中应用,其存在客户端渲染平台限制、交互性差等问题。本文所述方法对于客户端只需遵循数据解析协议即可,渲染平台无限制,在交互上良好、网络稳定的情况下可达交互不卡顿的效果。

(2) 与传统移动三维 GIS 应用对比

现在传统移动三维 GIS 应用以“先下载再显示”为主,存在模型数据大需要客户端大量存储空间、带宽受限等情况,本文所述方法通过边显示边传输方式,减少传输的数据量和显示的三角形数,达到理想的效果。由于 GIS 数据敏感性,传统方法通过对数据进行加密保证数据安全性,加大客户端计算压力。本方法对数据进行预处理,不增加客户端计算量下保证了数据安全。

5　结　语

本文介绍了一种基于远程渲染的移动三维 GIS 构建方法。该方法通过模型预处理策略、视点路径预测、客户端多重载入策略等多种方式优化客户端显示、降低客户端交互延迟,从而保证 GIS 应用大场景的渲染和友好的交互环境。该方法通过远程服务器承担大量渲染任务的方式减少客户端的计算量,在移动端拥有良好的计算性能,减少电量消耗,在各种移动设备,如手机导航、车载导航、GIS 行业应用或者是可穿戴式设备中都有良好的应用前景。

参　考　文　献

[1] Müller M U,Medyckyj-Scott D,Cowie A,et al. Current Status and Future Directions of Mobile GIS [C] //Proceedings of GIS and Remote Sensing Research Conference. SIRC NZ 2013,August 29-30,2013.

[2] Montgomery K,Addison J. High performance mobile 3D GIS-IVJ/WWJ on Android [C] //Proceedings of the 1st International Conference and Exhibition on Computing for Geospatial Research & Application. ACM,2010: 52.

[3] Noguera J M,Barranco M J,Segura R J,et al. A mobile 3D-GIS hybrid recommender system for tourism [J]. Information Sciences,2012,215: 37-52.

[4] Li X,Xu W,Zhu Q,et al. A Multi-Level Cache Approach for Realtime Visualization of Massive 3D GIS Data [J]. International Journal of 3-D Information Modeling (IJ3DIM),2012,1(3): 37-48.

[5] Jacynthe P,Sylvie D,Frédéric H,et al. Progress and New Trends in 3D Geoinformation Sciences [M]. Springer,2013.

[6] Zhang H X,Zhu B,Chan W. Interactive Rendering for Large-Scale Mesh Based on MapReduce [C] //Proceedings of The 13th International Conference on Computer-Aided Design and Computer Graphics,Nov 16-18,2013.

[7] 任政,杨旭波,肖双九,等. 远程渲染分发管理系统的设计与实现 [J]. 计算机应用与软件,2008,25(7): 6-7,10.

[8] 金平,张海东,齐越,等. 基于远程渲染的三维模型发布系统 [J]. 北京航空航天大学学报,2006,32(3): 337-341.

[9] Huang C Y,Chen K T,Chen D Y,et al. GamingAnywhere—The First Open Source Cloud Gaming System [J]. ACM Transactions on Multimedia Computing,Communications and Applications,2010,2(3): 1-20.

[10] Koller D,Turitzin M,Levoy M,et al. Protected Interactive 3D Graphics Via Remote Rendering [J]. ACM Transactions on Graphics,August 2004,23(3): 695-703.

[11] Paravati G,Sanna A,Lamberti F,et al. An open and scalable architecture for delivering 3D shared visualization services to heterogeneous devices [J]. Concurrency and Computation: Practice and Experience,2011,23(11): 1179-1195.

[12] Hoppe H. Progressive Meshes [C] //Proceedings of SIGRAPH. ACM SIGRAPH,August 1996: 99-108.

[13] Martin I M. Adaptive Rendering of 3D Models over Networks Using Multiple Modalities [R]. IBM T. J. Watson Research Center,2000.

[14] Garland M,Heckbert P. Surface Simplification using Quadric Error Metrics [C] //Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques,1997: 209-216.

[15] Grabner M. Feature preservation in view-dependent multiresolution meshes [C] //Proceedings of the 18th spring conference on Computer graphics. Budmerice,Slovakia,2002: 153-162.

[16] Zheng Z,Prakash E,Chan T K Y. Interactive View-Dependent Rendering over Networks [J]. IEEE Transactions on Visualization and Computer Graphics,2008,14(3): 576-589.