基于 Three. js 的真实三维地形可视化设计与实现

任宏康[1]，祝若鑫[1]，李风光[2]，王新量[3]

(1. 信息工程大学 地理空间信息学院，河南 郑州 450052; 2. 海军出版社，天津 300450;
3. 68029 部队，甘肃 兰州 730030)

摘　要: HTML5 的出现以及 WebGL 技术的推广，为三维 WebGIS 提供了新的解决方案。针对强调具有真实感三维地形的构建与实现，对 WebGL 第三方类库 Three. js 引擎进行了研究，提出了基于瓦片、DEM 数据的三维地形构建方法，并充分发挥 WebGL 开放性、免插件、跨平台、硬件加速的优势，创建了基于浏览器的三维地形交互系统，实验验证了该方法的可行性与有效性。

关键词: WebGL; Three. js; WebGIS; 三维地形; 瓦片; DEM
中图分类号: P209　　**文献标识码:** A　　**文章编号:** 1672 - 5867(2015) 10 - 0051 - 04

Design and Implementation of Realistic 3D Terrain Based on Three. js

REN Hong - kang[1], ZHU Ruo - xin[1], LI Feng - guang[2], WANG Xin - liang[3]

(1. College of Geography Space Information Institute, Information Engineering University, Zhengzhou 450052, China;
2. Naval Press, Tianjin 300450, China; 3. 68029 Troops, Lanzhou 730030, China)

Abstract: The emergence of HTML5 and promote of WebGL have provided a new solution for 3DWebGIS. For the construction and implementation of realistic 3D Terrain, researched the WebGL third - party library Three. js, proposed a way to construct 3D Terrain based on Tiles and DEM, and gave full play the WebGL open, free plug - in, cross - platform, hardware - accelerated advantages, created three - dimensional terrain browser - based interactive system. Do experiments to validate the feasibility and effectiveness of the method.

Key words: WebGL; Three. js; WebGIS; 3D terrain; tiles; DEM

0　引　言

近年来，计算机与互联网技术发展迅猛，Web3D 技术在此期间快速发展，并趋于成熟，在各领域都得到了广泛的应用。在 WebGIS 领域，针对客户端三维实现采用的 Web3D 技术主要有 VRML，Flash，Silverlight，Java Applet，ActiveX，X3D 等。用户在对系统访问时需要安装几十 K 到几兆的具有很强封装性的插件，这极大降低了系统的体验与实用性，同时，在系统移植方面也不灵活; 另一方面，近两年移动智能终端的性能有了飞速的发展，对于三维系统的构建，只考虑基于 PC 端是不能满足所有需求的。所以，以何种技术与方式构建开放的、跨平台的、免插件的且具有良好渲染与交互效果的三维 WebGIS 构建技术成为研究的热点和重点。

新一代的 HTML5 标准，以及新一代的 3D 图形引擎的 WebGL 技术，可以解决上述难题。它们打破了时间与空间上的限制，用户只需在网络环境中，手持任意智能设备，即可轻松对三维场景进行访问。本文针对三维 GIS 中的地形构建与可视化表达展开了研究与实验，对瓦片数据与 DEM 数据进行了分析与数据提取设计，实现了基于 WebGL 第三方库 Three. js 的实验系统，具有免插件、跨平台、硬件加速的特性，为三维地形可视化表达提供了一种新的方法与途径。

1　关键技术

本文采用的关键技术包括新一代 Web 标准 HTML5，3D 绘图标准 WebGL 以及第三方类库 Three. js。其中 HTML5 具体可分为 HTML、CSS 和 JavaScript 技术，致力于打造一个各系统平台无缝链接、更具交互功能的开放式环境，创建一种更加丰富、轻便、独立的免插件、跨平台产

收稿日期: 2015 - 06 - 11
基金项目: 国家自然科学基金项目(41271392，41401462) 资助
作者简介: 任宏康(1992 -)，男，辽宁铁岭人，地图制图学与地理信息工程专业硕士研究生，主要研究方向为三维 GIS 与地理信息系统开发。

品，因此添加了很多新的特性与元素，包括 Firefox，Google Chrome，Opera，Safari 4 +，Internet Explorer 9 + 等主流浏览器都已支持 HTML5[1]。正是利用这些特性，在进行三维表达时，无须再像 Flash、Siverlight 那样安装插，即可实现对像素级别的位图像进行 2D 与 3D 动态渲染。

WebGL 是基于 OpenGL ES（OpenGL for Embedded Systems）2.0 的 JavaScript 绑定，通过前端 HTML5 的 Canvas 元素实现三维复杂场景与模型的创建。相比其他 Web3D 实现，WebGL 具有一定的优势。首先，通过脚本语言 JavaScript 实现网络交互，克服了开发网页专用渲染插件的弊端；其次，WebGL 是开放的，可提高开发效率；再次，WebGL 可以充分利用底层的图像硬件加速功能实现三维图形的渲染[2]。在 GIS 领域，当前许多公司都开始推出基于 WebGL 的 3D 地图产品，包括谷歌地图、诺基亚地图、必应地图、百度全景等，还有一些开源的项目，如 Cesium，WebGL Global，Open WebGlobal 等。

Three. js 依据 WebGL 规范，对底层 WebGL 代码进行简单封装，通过掩盖一些麻烦的细节，减轻开发者的开发负担并加快开发速率，在处理浏览器 3D 效果方面表现优异[3]。Three. js 支持多种渲染器（renderer）进行场景绘制，提供了点、线、面、向量、矩阵等三维创建时所需的基本要素，并可以简单快速地将建镜头（Cameras）、物体（objects）、光线（lights）等对象添加到场景（Scene）中[4]。Three. js 具有开放性，并提供的大量实例源代码，开发者可通过逆向工程，充分进行深入研究，进行三维场景创建，快速有效。

2 基于 Three. js 的真实三维地形设计

2.1 基本原理

真实三维地形的可视化需要顾及以下因素：DEM 数据、瓦片数据、用于显示实现的技术以及相应的纹理映射法则。考虑到地球曲率的影响以及实际应用，将地形设定为具有起伏的平面进行设计，利用 Three. js 提供的平面绘制接口 THREE. PlaneBufferGeometry 进行实现。采用基于瓦片创建三维地形的思想：地形的真实感由瓦片的级数、像素决定；绘制平面块的高、宽、片段数、位置由瓦片数据的大小、坐标、行列数决定；由于 WebGL 采用右手笛卡儿坐标，三维起伏的效果由 THREE. PlaneBufferGeometry 顶点数组 vertices 中的 y 值决定，需赋值依据瓦片获取的 DEM 高程数据；最后依据视野范围与瓦片位置，动态创建拼接平面块，生成具有真实感与实地坐标相符的三维地形。

2.2 瓦片与 DEM 数据的处理

三维地形的创建可看作瓦片数据与 DEM 数据的融合。融合需要采用统一的空间坐标系，将经纬度坐标进行转换，通常采用 Web 墨卡托投影。由于瓦片数据都是正方形，所以 X 轴与 Y 轴的取值范围都是 [-20 037 508.342 789 2，20 037 508.342 789 2]。当前几乎所有的网络地图都采用瓦片地图的方式，一个瓦片的容量小，并利用金字塔模型进行管理与组织，能够快速有效地对地图进行调度和显示[5]。一个瓦片的宽度与高度都是 256 像素，其格式大多为 png 与 jpg。瓦片除了反映地物的彩色图像内容外还包含：确定该切片在空间中位置的切图原点与瓦片行列号，确定瓦片所表示地物的范围大小的分辨率、分辨率级别和切图范围。这些信息数据量不大但均为关键性信息，这里把其编写为如表 1 的 json 文件，以便通过 JavaScript 进行读取。

表 1 Tile_info. json 文件说明

Tab. 1 Tile_info. json file description

数据名称	注释
t_left	瓦片范围：左
t_top	瓦片范围：上
t_right	瓦片范围：右
t_bottom	瓦片范围：下
t_rows	瓦片行数
t_columns	瓦片列数
t_step	瓦片大小

表 2 DEM_info. json 文件说明

Tab. 2 DEM_info. json file description

数据名称	注释
d_left	DEM 左下点 X 坐标
d_bottom	DEM 左下点 Y 坐标
d_rows	DEM 行数
d_columns	DEM 列数
d_step	DEM 栅格大小
d_elevations	保存高程数据的数组（-9999 为无值）

DEM 是按一定方式测定的一定数量离散点的平面位置和高程值，是对地形起伏的数字表达，狭义上讲，也可以看作是按照规则网格间隔进行采集的地面高程值的集合[6]。DEM 数据文件的存储使用二进制方式或 ASCII 码的形式，其格式（扩展名）多种多样，一般可直接利用 txt 记事本打开。格式如国家测绘局所制定的 *. grd、吉奥公司的 *. dem、ARC/INFO 公司的 *. asc 与 *. bil 等。其数据内容大致相同，包括文件标识、格网的行数列数、格网范围的坐标参数、网格间隔的大小以及按一定顺序排列的高程值等。本文采用 ARC/INFO 的 GRID 格式（*. asc）的数据，通过编写好的文件读取操作输出为表 2 所示的 json 文件。

2.3 基于 DEM 的高程值获取

THREE. PlaneBufferGeometry 是基于规则网格进行平面绘制的，所以用于提取高程值的 DEM 数据模型同样采用规则网格模型，简单便捷，且具有高效的读取效率。对于不规则网格数据，应先处理成规则网格。如图 1 所示，通过叠加分析从 DEM 网格提取的所需目标瓦片网格内的高程值，依据对应的位置信息由 DEM 网格点插值获取，下面对提取过程进行详细表述。

首先，获取坐标位置信息。其中瓦片格网与 DEM 格网的左下点的坐标都是已知，设为（TX_0，TY_0）与（DX_0，DY_0），同时两个格网的各自间隔都是已知，设为 D_t 与 D_d，

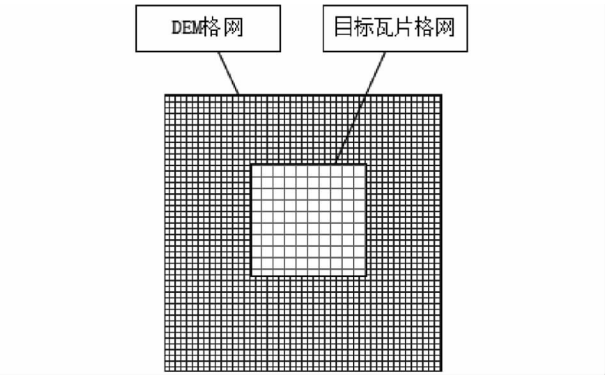

图 1　网格叠加分析

Fig. 1　Grid overlay analysis

则 DEM 网格中点 D_{ij} 的坐标与单独瓦片网格 T_{ij} 的左下角点坐标为(其中 i,j 为行列数):

$$\begin{cases} D_{ij} = (DX_0 + D_d \times i, DY_0 + D_d \times j) \\ T_{ij} = (TX_0 + D_t \times i, TY_0 + D_t \times j) \end{cases} \tag{1}$$

其次,根据所求瓦片网格点的坐标检索其周围的 DEM 网格点。为了与 THREE. PlaneBufferGeometry 绘制参数对应,确保地形的起伏效果,瓦片网格的行列数应与片段数 widthSegmen,heightSegments 对应,则第 T_{ij} 瓦片第 m 行 n 列网格点的坐标为:

$$T_{ij}(m,n):(TX_0 + D_t \times i + \frac{D_t}{\text{seg}} \times m, TY_0 + D_t \times j + \frac{D_t}{\text{seg}} \times n) \tag{2}$$

式中,seg 为片段数(宽、高相同),那么,该网格点最邻近的左下角 DEM 网格点的行列号 M,N 为(其中,⌊⌋为向下取整函数):

$$\begin{cases} M = \lfloor \frac{T_{ij}(m,n)\ y - DY_0}{D_d} \rfloor \\ N = \lfloor \frac{T_{ij}(m,n)\ x - DX_0}{D_d} \rfloor \end{cases} \tag{3}$$

DEM 网格点的高程存放在 DEM_info. json 文件的 d_elevations 数组中,数据的存储是按先从上到下,再从左至右的顺序排列的。见表 3,提取瓦片网格点 $T_{ij}(m,n)$ 周围 4 点的高程值:

表 3　$T_{ij}(m,n)$ 周围四点高程值

Tab. 3　$T_{ij}(m,n)$ around four point elevation

点位置	高程值
左下	d_elevations [d _rows * N + M]
左上	d_elevations [d _rows * N + M - 1\]
右上	d_elevations [d _rows * (N+1) + M\]
右下	d_elevations [d _rows * (N+1) + M - 1\]

最后,通过双线性插值对点 $T_{ij}(m,n)$ 赋值。双线性插值同时考虑到了插值精度与计算机开销的问题,由于简单高效直观,常用于实际工程中。插值点与周围 4 点都具有线性关系,过渡平滑,质量好,适用于这种当前层重采样。设该 4 点的高程值依次用 $D1$,$D2$,$D3$,$D4$ 表示,其计算公式为:

$$\begin{aligned} T_{ij}(m,n) = & \frac{(T_{ij}(m,n)\ x - DX_0)\ \%\ D_d}{D_d} \times \frac{(T_{ij}(m,n)\ y - DY_0)\ \%\ D_d}{D_d} \times D1 + \frac{1 - (T_{ij}(m,n)\ x - DX_0)\ \%\ D_d}{D_d} \times \frac{(T_{ij}(m,n)\ x - DX_0)\ \%\ D_d}{D_d} \times D2 + \\ & \frac{(T_{ij}(m,n)\ x - DX_0)\ \%\ D_d}{D_d} \times \frac{(T_{ij}(m,n)\ y - DY_0)\ \%\ D_d}{D_d} \times D3 + \frac{1 - (T_{ij}(m,n)\ x - DX_0)\ \%\ D_d}{D_d} \times \frac{1 - (T_{ij}(m,n)\ y - DY_0)\ \%\ D_d}{D_d} \times D4 \end{aligned} \tag{4}$$

其中,% 为求余计算。最后通过循环遍历,计算出 T_{ij} 瓦片所包含的所有网格点的高程值(共(seg + 1) ×(seg + 1)),将其存入瓦片高程值数组 Tiles_vertices [i] [j] 中,返回给函数 THREE. PlaneBufferGeometry,供其进行地形绘制。

2.4　LOD 模型的组织设计

LOD(Levels of Detail)细节层次模型技术是在不影响视觉效果的条件下,依据模型节点在三维场景中的位置及其重要性,逐次简化模型的细节与几何复杂度,提高绘制算法的效率,对物体的渲染重新进行资源分配等[7]。LOD 技术又分为离散型与连续型,离散型在不同 LOD 模型之间切换时具有很大的视觉突跳感,本文采用连续的 LOD,利用 three. js 提供的 THREE. LOD 接口进行三维地形 LOD 模型的组织,预先构建生成了 4 层 LOD 模型,采用静态的方式依据模型与摄像机的距离选择合适的细节层次模型。该 4 层 LOD 模型采用同样的纹理 floorMaterial,根据距离参数逐级进行模型的简化。

3　系统实现

首先,利用 Three. js 封装好的接口进行三维场景的基本搭建。包括场景 THREE. Scene、透视投影摄像机(THREE. PerspectiveCamera)、平行光源(THREE. DirectionalLight)、半球光源(THREE. HemisphereLight)、WebGL 渲染器(THREE. WebGLRenderer)以及相机控制器(THREE. OrbitControls)。下面进行三维地形模型的加载。

Three. js 对三维场景中模型的创建加载有两种方式,一种是直接导入 json,obj,vtk 等模型文件,另一种是使用 THREE. mesh 三维网格动态创建模型。THREE. Mesh 的实现需要两部分: Geometry 几何函数与 Material 纹理。这里 Geometry 加载的是 PlaneBufferGeometry,其属性 attributes. position. array 以 x,y,z 顺序逐点存储顶点的位置信息,决定地形的起伏与位置。Material 加载的是 MeshPhongMaterial(冯氏材质类型,具有光泽的材质感),其中 MeshPhongMaterial 的 map 参数则要加载采用文件方式存储的瓦片图片。绘制流程如图 2 所示。

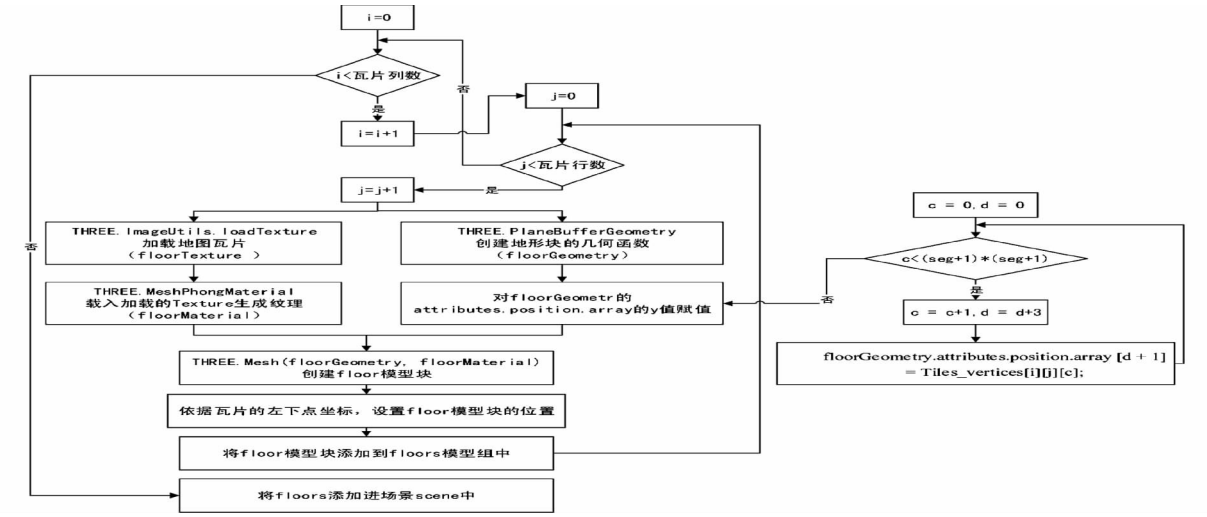

图 2 三维地形实现流程图

Fig. 2 3D terrain realization flow chart

通过浏览器进行访问，效果如图 3 所示。

图 3 三维地形效果图

Fig. 3 3D topographic map

对于 LOD 的实现，通过减少 THREE. PlaneBufferGeometry 的片段数进行简化操作，从底层依次逐级减少 positon 数组的数量。4 层 LOD 模型分别为 20 × 20，10 × 10，5 × 5，2 × 2 网格。如图 4 是 5 × 5 网格从 10 × 10 网格提取高程值的方法：

```
for( var c = 0,d = 0; c < 36; c + + ,d + = 3)
{
vertices1 [d + 1] = Tiles_vertices [i] [j] [Math. floor( c/6) × 22 + ( c%6) × 2];
}
```

THREE. LOD 的使用需要添加具体的 Mesh 地形模型以及距离参数，前面的 4 层网格对应的距离分别为 300，800，1 500，3 000。将 LOD 添加到场景 scene 以此来进行层次的转换。图 4(1) 为 5 × 5 网格地形图与 4(2) 为 10 × 10 网格地形图，为两个层次下 Mesh 网格的绘制情况，网格相差的倍数为 4 倍，通过相机移动可流畅切换，无跳跃感。由此完成三维地形 LOD 模型的组织与创建，快捷高效，效果良好。

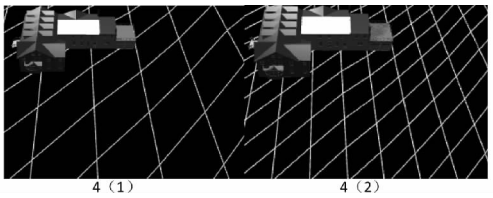

图 4 Lod 模型简化

Fig. 4 LOD model simplification

4 结束语

本文利用 HTML5 与 WebGL 第三方类库 Three. js 技术，基于瓦片信息对真实三维地形进行了设计与实现。首先对瓦片数据与 DEM 数据进行了分析研究，继而基于瓦片信息从 DEM 数据中提取所需高程值，再利用 Three. js 接口 THREE. PlaneBufferGeometry 进行了三维地形可视化，最后使用 THREE. LOD 创建组织 4 层 LOD 模型。该方案具有开放、免插件、跨平台的优势，是一种新型有效的 3DWebGIS 方案，为用户带来了全新的良好体验。

参考文献:

[1] Lawson B，Sharp R. IntroducingHTML5 [M]. United States of America: Pearson Education，2010.

[2] 刘爱华，韩勇，张小垒，等. 基于 WebGL 技术的网络三维可视化研究与实现 [J]. 地理空间信息，2012(5) : 79 - 81.

[3] Cui Peng. The Research and Design Of 3D Web Guide System Based On WebGL [A]. 第 26 届中国控制与决策会议论文集，2012.

[4] 许虎，聂云峰，舒坚. 基于中间件的瓦片地图服务设计与实现 [J]. 地球信息科学学报，2010(4) : 562 - 567.

[5] 李志林，朱庆. 数字高程模型 [M]. 武汉: 武汉大学出版社，2001.

[6] 杜剑侠，李凤霞，战守义. LOD 算法研究及其在地形实时显示中的应用 [J]. 计算机工程与应用，2005(13) : 211 - 213.

[编辑: 任亚茹]