

李莹,赵宁曦. 三种基于 Cesium 的重庆市漫游飞行算法及其对比分析. 吉林大学学报(地球科学版), 2024, 54(2): 709–720.
doi:10.13278/j.cnki.jjuese.20230029.
Li Ying, Zhao Ningxi. Three Roaming Algorithms of Chongqing City Based on Cesium and Their Comparative Analyses.
Journal of Jilin University (Earth Science Edition), 2024, 54 (2): 709–720. doi:10.13278/j.cnki.jjuese.20230029.

三种基于 Cesium 的重庆市漫游 飞行算法及其对比分析

李 莹, 赵宁曦

南京大学地理与海洋科学学院, 南京 210023

摘要:漫游飞行在数字地球、虚拟现实和仿真等领域有着广泛的应用。以往的研究多集中于客户机/服务器(client/server, C/S)结构的桌面端应用程序, 本文以浏览器/服务器(browser/server, B/S)结构下的 Web 端开源三维地理框架 Cesium 为基础构建虚拟地球, 设计了三种不同的漫游算法: 常规漫游算法、回调函数漫游算法、拉格朗日插值漫游算法, 并在地球表面成功加载了重庆市主城区的倾斜摄影模型以实现漫游飞行功能, 从帧率(frames per second, FPS)和网络延迟(network latency, NL)两方面对比分析三种算法的有效性。结果表明, 常规漫游算法、回调函数漫游算法、拉格朗日插值漫游算法前 50 s 的平均 FPS 分别为 70、74、80 Hz, 平均 NL 分别为 14.145、13.166、12.419 ms。就 FPS 而言, 回调函数漫游算法较常规漫游算法提升了 5.714%, 拉格朗日插值漫游算法较常规漫游算法提升了 14.286%。就 NL 而言, 回调函数漫游算法较常规漫游算法缩短了 6.921%, 拉格朗日插值漫游算法较常规漫游算法缩短了 12.202%。总体而言, 算法运行效率从大到小最终排序为: 拉格朗日插值漫游算法、回调函数漫游算法、常规漫游算法, 即在漫游飞行过程中, 回调函数漫游算法、拉格朗日插值漫游算法较常规漫游算法有一定的提升, 拉格朗日插值漫游算法可实现更高效、流畅的渲染效果。

关键词: Cesium; 浏览器/服务器; 漫游算法; 帧率; 网络延迟

doi:10.13278/j.cnki.jjuese.20230029

中图分类号: P208; P258

文献标志码: A

Three Roaming Algorithms of Chongqing City Based on Cesium and Their Comparative Analyses

Li Ying, Zhao Ningxi

School of Geography and Ocean Science, Nanjing University, Nanjing 210023, China

Abstract: The roaming function is widely used in the fields of digital earth, virtual reality and simulation. Previous researches mostly focused on desktop applications under client/server (C/S) structure. This paper constructs a virtual Earth based on Cesium, an open-source 3D geography

收稿日期: 2023-02-17

作者简介: 李莹(1992—), 女, 硕士研究生, 主要从事三维可视化方面的研究, E-mail: liying_gis@whu.edu.cn

通信作者: 赵宁曦(1960—), 女, 副教授, 主要从事人文地理和旅游方面的研究, E-mail: znx_njugis@163.com

基金项目: 国家自然科学基金项目(41771417)

Supported by the National Natural Science Foundation of China (41771417)

framework on the Web side under browser/server(B/S) structure and designs three different roaming algorithms: conventional algorithm, callback function algorithm and Lagrange interpolation algorithm. Then, the tilting photography model of downtown Chongqing was successfully loaded on the virtual Earth's surface to realize the roaming functions of the three different algorithms. Finally, the effectiveness of the three algorithms was compared in terms of frames per second (FPS) and network latency (NL). The results show that the average FPS in the first 50 s for conventional algorithm, callback function algorithm and Lagrange interpolation algorithm are 70, 74 and 80 Hz, respectively; The average NL in the first 50 s for conventional algorithm, callback function algorithm and Lagrange interpolation algorithm are 14.145, 13.166 and 12.419 ms, respectively. In terms of FPS, callback function algorithm is 5.714% higher than conventional algorithm, while Lagrange interpolation algorithm is 14.286% higher than conventional algorithm. In terms of NL, callback function algorithm is 6.921% shorter than conventional algorithm, while Lagrange interpolation algorithm is 12.202% shorter than conventional algorithm. Overall, the final order of the three algorithms' running efficiency from largest to smallest is: Lagrange interpolation algorithm, callback function algorithm, and conventional algorithm. Namely, callback function algorithm and Lagrange interpolation algorithm have certain improvements compared with conventional algorithm in the process of roaming, and Lagrange interpolation algorithm can achieve the most efficient and smooth operation and rendering effects.

Key words: Cesium; B/S; roaming algorithms; FPS; NL

0 引言

虚拟现实(virtual reality, VR)技术当前正处于蓬勃发展时期,早在 2015 年举行的谷歌开发者大会上,VR 就作为重要议题引发了无数探讨。VR 技术通过头盔式显示器、手部控制器等交互设备,对包括视觉、听觉、触觉在内的多种人体感知功能进行模拟,使用户完全沉浸于虚拟世界之中,并与之进行互动。虚拟地球作为 VR 的一个重要发展方向,是可视化和共享三维(three-dimensional, 3D)地理数据的一种方式,相关软件正在迅速普及。如:Google Earth、NASA World Wind、Microsoft Virtual Earth、Skyline、Vega、IMAGING Virtual GIS、VRMap、SuperMap、GeoGlobe 等。Google Earth 于 2005 年 6 月公开发布,由于能够以“上帝视角”浏览地理场景而引起了广泛的关注^[1],目前全球下载该软件的人数已经超过 10 亿。Chen 等^[2]以福州市的地上和地下空间模型为例,使用开源场景图形库(openscenegraph, OSG)和开源虚拟地球图形库(openscenegraph Earth, OSGEarth)作为 3D 可视化引擎开发了一个虚拟地球的原型系统。OSG 是视觉模拟、计算机游戏、VR、科学可视化和建模等领域的应用程序开发人员使用的开源 3D 图形应用程序编程接口^[3],而 OSGEarth 是 OSG 在虚拟地球领

域的拓展。Gong 等^[4]为了集成来自各种地理信息系统(geographic information system, GIS)的空间数据和处理功能,开发了一个名为“GeoGlobe”的虚拟地球系统,用于无缝组织全球数据模型。

漫游飞行是一种常见的 VR 技术,在 3D 场景中飞行可鸟瞰城市全貌,让用户如同置身于真实的 3D 世界中,在各大虚拟地球系统平台中都是常见功能之一,如:高辉等^[5]采用多细节层次(levels of detail, LOD)技术和 Vega 编程平台创建了一个基于真实地景的 3D 飞行再现系统;曹晶等^[6]使用 Terra Vista 软件结合 OSG 实现了大范围城市 3D 系统的快速建模及流畅漫游;段忠祥^[7]利用 Visual C++ 和开放图形库(open graphics library, OpenGL)编程,模拟飞机在一个真实感较好的背景下的漫游过程。然而,上述虚拟地球系统大都基于客户机/服务器(client/server, C/S)架构开发,用户需本地下载安装才能使用,缺乏可扩展性,难以集成到第三方应用程序和操作环境中去。相较而言,基于浏览器/服务器(browser/server, B/S)的、无需显式安装且易于跨平台的系统受到人们越来越多的欢迎。若要在 Web 浏览器中实现漫游飞行功能,传统方法是将桌面端应用程序封装成插件,如 ActiveX、COM、Flash 等,嵌入浏览器中进行显示。然而,插件式 Web 系统存在兼容性差、性能不稳定等问题,

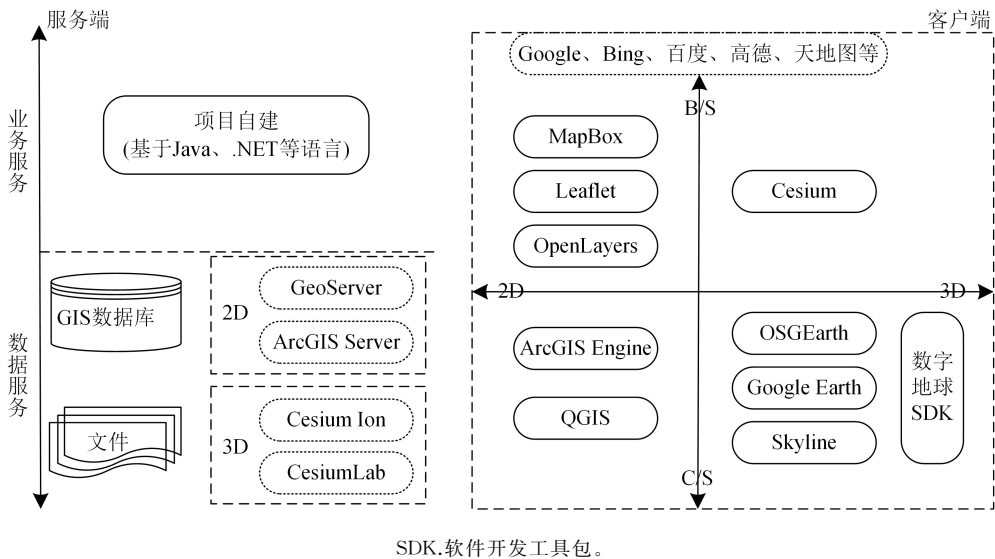
在很大程度上限制了 Web 程序的推广和使用。

随着 HTML5(一种用于构建和呈现万维网内容的语言)标准和其他 Web 技术的快速发展,传统的 2D 地图渲染过程逐渐被更丰富、更具描述性和直观性的 3D 虚拟地球系统所补充,用户无需安装任何特定的应用程序插件即可在浏览器中实现 3D 地理信息可视化,这些最新成果为地理空间应用程序中的可视化任务开辟了全新的可能性^[8]。Krooks 等^[9]利用开源数据源和开源网络服务,使用一种基于网络的 3D 图形库 WebGL(web graphics library),在任何兼容的 Web 浏览器中生成加速的 3D 图形^[10],并展示了芬兰开放地理空间数据集的 3D 环境模型。Lei 等^[11]基于 Ellipsoidal Mercator 投影和 1984 世界大地坐标系(world geodetic system 1984, WGS84),使用网络传输的 JavaScript 对象简谱(Javascript object notation, JSON)文件格式描述了流行的瓦片地图服务方法,并实现了一种基于 WebGL 技术的 3D WebGIS 系统。常用的 Web 端开源 3D 图形库有 Three.js、 Babylon.js,但二者一般用于游戏开发,几乎不涉及地理信息。由 AGI 公司计算机图形开发小组于 2011 年研发的 Cesium 是目前最为成熟和完善的 3D 地球和地图可视化开源 JavaScript 库^[12]。相较于传统 C/S 结构的桌面端 3D 地球图形库(如 OSGEarth),Cesium 的特点是基于 B/S 结构,具有跨平台、可移植性,无需安装插件或搭建开发环境,只需一个浏览器即可

在多种平台上高效、流畅地展示 3D 地形场景和模型。如:方孟元等^[13]以磁浮铁路项目为例,介绍了在 Cesium 中实现建筑信息模型(building information modeling, BIM)与 GIS 数据集成的可视化思路;郭晓非等^[14]以网络通用数据格式(network common data form, NetCDF)数据为例,实现了基于 Cesium 开源地图框架的海洋环境要素可视化;张永强等^[15]以 Cesium 和 WebGL 为技术依托,结合天地图影像服务,发布并加载了泰山地区的数字高程模型(digital elevation model, DEM)地形。上述有关 Cesium 的研究多集中于地形、影像数据的加载,停留在数据层面,而对算法的优化研究较少。因浏览器内核承载力有限,对算法的运行效率提出了更高的要求。本文以 Web 端开源 3D 地理框架 Cesium 为基础,设计了三种不同的漫游算法,实现漫游飞行功能的同时对比分析三种算法的有效性。

1 Cesium 概述

Cesium 是一种基于 WebGL 标准的支持 3D、2D、2.5D 地图展示的 Web 端开源 JavaScript 库,拥有行星级渲染和 GIS 功能。图 1 显示了 Cesium 在 GIS 中的定位。Web 端利用 Cesium 应用程序编程接口(application programming interface, API)调用地图服务。Cesium API 集成了 Knockout JS(一个可使用干净的底层数据模型创建丰富、响应式



SDK.软件开发工具包。
图 1 Cesium 在 GIS 中的定位
Fig.1 Cesium's position in GIS

显示和编辑器用户界面的 JavaScript 库)和 When JS(一个独立、可靠、小尺寸、高效的 Promises/A+ 异步编程实现)等框架,封装了一些异步 JavaScript 和 XML(asynchronous javascript and XML, Ajax)功能^[16],通过 JavaScript 语言直接调用底层硬件设备,实现数据的加载、查询、可视化等功能。

视图对象(Viewer)是 Cesium 的核心,用来实现基本虚拟地球框架的搭建以及不同类型图层的加载。Cesium 中的图层分为两类:一是普通图层,包含影像、线划等显示图层,通过影像服务接口(ImageryProvider)实现加载;二是地形图层,Cesium 中的地形系统是一种由流式瓦片数据生成地形网格的技术,可自动模拟平原、山川、海洋的 3D 效果,通过地形服务接口(TerrainProvider)实现加载。3D Tiles 技术^[17]是 Cesium 团队于 2016 年推

出的数据规范,提供了 LOD 能力,使得在 Web 环境下,海量异构的 3D 地形影像和模型数据,如遥感影像、倾斜摄影模型、3ds Max、BIM、激光探测及测距(light detection and ranging, LiDAR)点云等,可按瓦片金字塔方式在 Cesium 中实现逐层加载,如图 2 所示。此外,Cesium 还支持加载多种格式(如美国环境系统研究所公司 ESRI 开发的一种空间数据开放格式(shapefile, Shp)、Cesium 语言(Cesium language, CZML)格式、对各种地理数据结构进行编码的格式(geographical JSON, GeoJSON)、Google 旗下的 Keyhole 公司开发和维护的一种基于 XML 的标记语言(Keyhole markup language, KML)格式等)的矢量数据到 3D 场景中,并将加载的图层自动进行合并,完成地形、影像、矢量数据的综合展示。

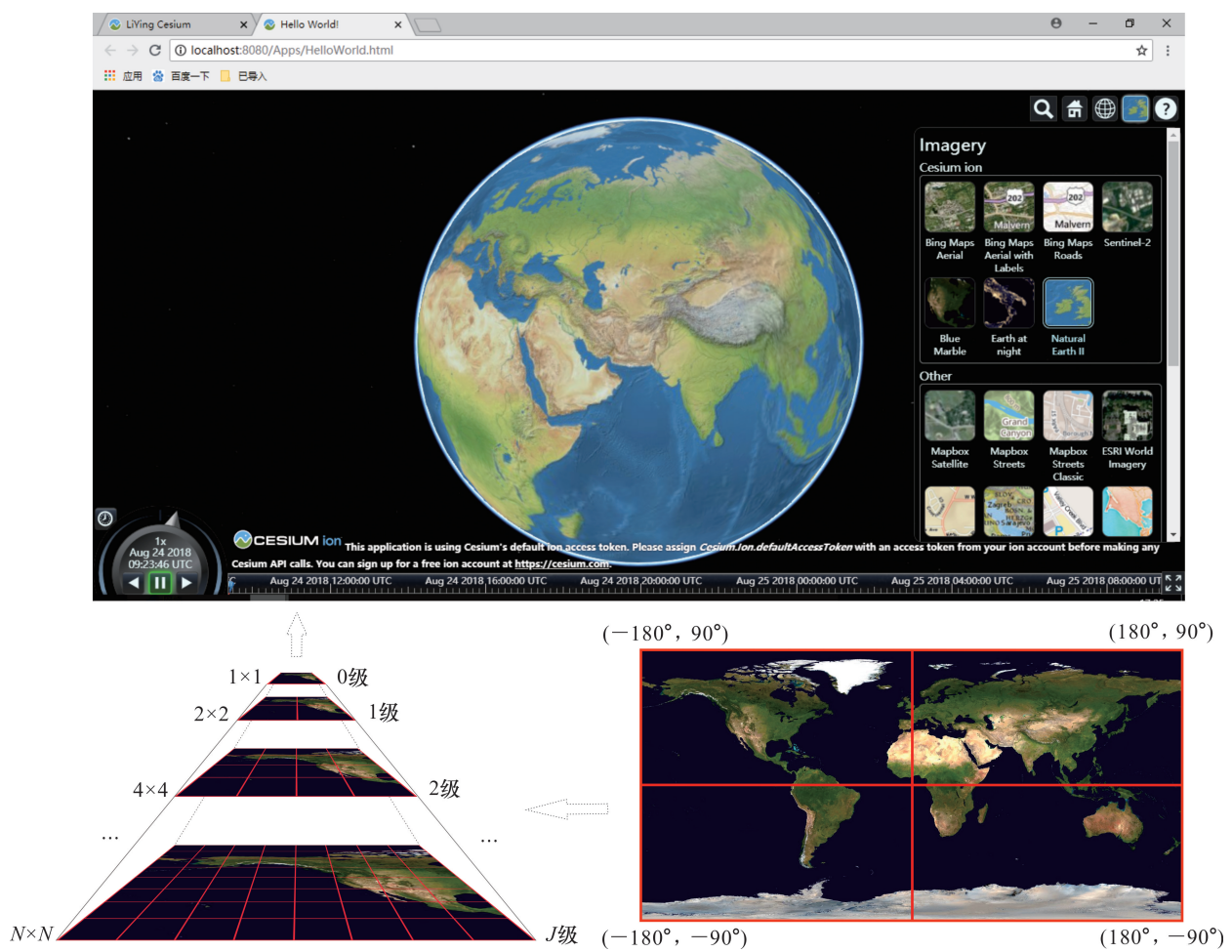


图 2 Cesium 影像加载过程
Fig.2 Image loading process of Cesium

2 研究方法

2.1 漫游系统构建

图 3 展示了 Cesium 漫游功能实现的流程,主要包含两大步骤。

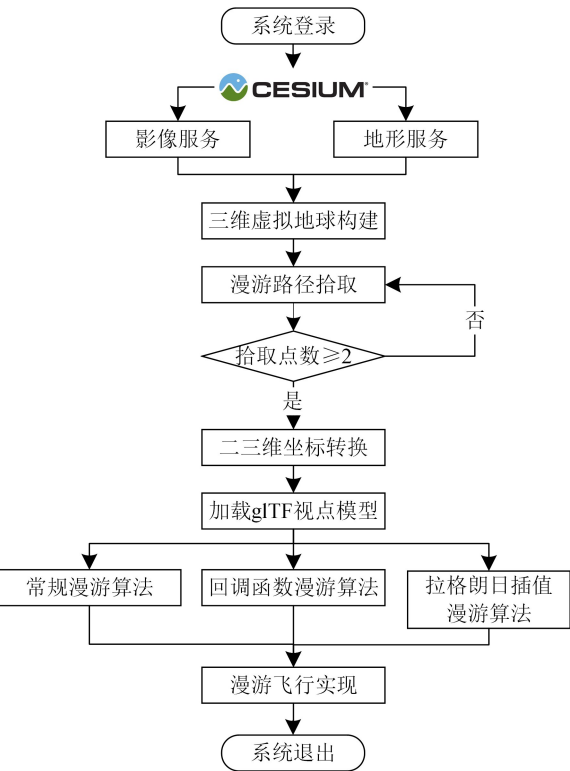


图 3 Cesium 漫游功能实现的流程图

Fig.3 Flow chart of Cesium's roaming implementation

1)漫游路径的拾取。用鼠标在屏幕上点选的是 2D 屏幕坐标,但漫游路径是 3D 世界坐标,因此需要进行二三维坐标转换,其过程如图 4 所示。在透视投影下, C 为相机, θ 为水平视角, N 为近屏幕, F 为远屏幕。近屏幕上一点 $A(x,y)$ 的像素深度值为 z ,经过视口变化,乘以投影逆矩阵和模型视图逆矩阵等进行一系列反投影操作后,转换成笛卡尔空间直角坐标系下的 3D 坐标 $A'(x',y',z')$ 。

2)漫游飞行的实现。首先需要加载 3D 对象,模拟人眼提供观测视角。Cesium 常用的 3D 对象模型为图形语言传输格式 (graphics language transmission format, glTF)^[18],即由 OpenGL 和 3D 图形标准组织 Khronos 所定义的 Web 端 3D 模型通用标准。glTF 使用 JSON 进行描述,包含场景、像机、动画、网格、材质、纹理、渲染技术、着色器等信息,几乎每个框架都支持 glTF。本文设计了三种不同的算法以实现漫游飞行功能,如表 1 所示,从算法原理和优、缺点等方面对各算法进行详细的对比分析。

2.2 漫游算法设计

2.2.1 常规漫游算法

常规漫游算法的思路为:设漫游路径由一系列点 $[P_0,P_1,\dots,P_n]$ 组成, $P_i(x_i,y_i,z_i)$, $i=0,1,\dots,n$ 。图 5a 为漫游飞行路径的示意图,每一个转折点 P_i 与起点 P_0 的距离用 l_i 表示,请注意, l_i 不是两点间的直线距离,而是视点飞行至 P_i 点途经的所有路径折线的总长度,记为 $\Delta l_i=l_i-l_{i-1}$, $i\neq$

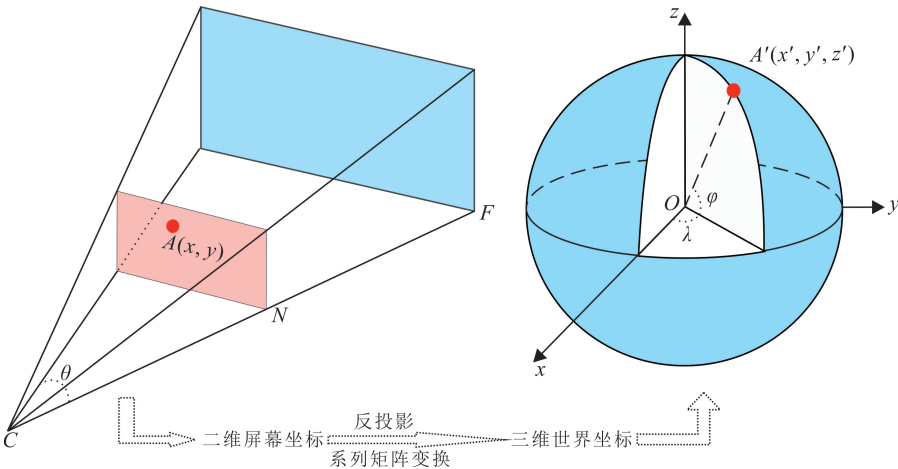


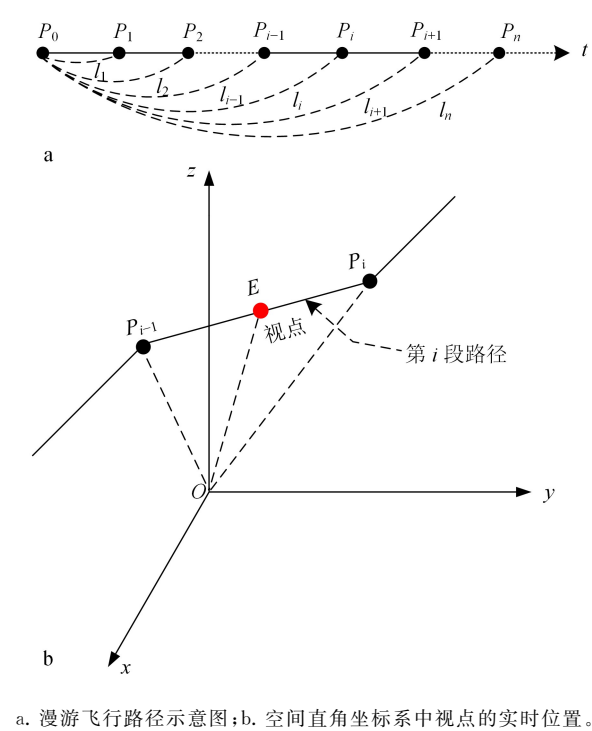
图 4 二三维坐标转换过程

Fig.4 Transformation process from 2D coordinate to 3D coordinate

表 1 三种漫游算法优缺点对比分析

Table 1 Comparative analyses of advantages and disadvantages of the three roaming algorithms

算法名称	原理	优点	缺点
常规漫游算法	使用 Cesium 的位置点类存储(时刻,3D 点坐标)序列,静态设置视点位置和飞行起止时间等参数	算法简单,参数较少,无需设计复杂的函数来计算视点位置,漫游飞行过程可由系统自行配置和运行	灵活性低,位置、时间、模型等相关参数需预先设置。未采用插值法动态生成视点位置,故运行至拐点处突然转向,缺少缓冲过程,视觉冲击较大,流畅度较低
回调函数漫游算法	利用点向式插值法和回调函数动态更新视点位置	回调函数可实现异步调用和代码精简化,灵活性高且功能强大,维护了 JavaScript 代码的抽象性及可读性;点向式插值法较为简单,容易实现	回调函数按帧动态刷新视点位置,当场景数据量较大时,对浏览器的内核造成一定的负担。此外,回调函数的构造和调用较为复杂,对算法设计者来说是一种考验
拉格朗日插值漫游算法	结合拉格朗日插值法和动态缓存 CZML 文件以实现视点位置实时更新	CZML 文件本质是 JSON 格式的,JSON 数据可实现高效的网络传输。此外,采用拉格朗日插值法实时内插视点位置,拟合程度较高,可减小运行至拐点处的视觉冲击	当数据较为繁杂时,若不借助计算机,则求解起来较为困难。当路径上点较多时,拉格朗日插值多项式的次数会很高,数值不稳定,容易出现较大误差。且动态缓存 CZML 文件需要设置的参数较多,易出错



a. 漫游飞行路径示意图;b. 空间直角坐标系中视点的实时位置。

图 5 空间直角坐标系中直线的点向式原理
Fig.5 Principle of pointwise of a line in the Cartesian coordinate system

0。设视点 E 飞行至第 i 段折线的起点为 $P_{i-1}(x_{i-1}, y_{i-1}, z_{i-1})$, 终点为 $P_i(x_i, y_i, z_i)$, 该线段的方向向量为 $\overrightarrow{P_{i-1}P_i} = (x_i - x_{i-1}, y_i - y_{i-1}, z_i - z_{i-1})$ 。则可计算每一段折线长度 Δl_i 以及 l_i :

$$\Delta l_i = \left| \overrightarrow{P_{i-1}P_i} \right| = \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2 + (z_i - z_{i-1})^2}; \quad (1)$$
$$l_i = \sum_{j=1}^i \Delta l_j. \quad (2)$$

设 $\bar{v}$ 为视点匀速飞行的速度, t_0 为初始时刻, t_i 为飞行至 P_i 点的时刻, $\Delta t = l_i / \bar{v}$ 表示飞行时长, 则 $t_i = t_0 + \Delta t$ 。依此类推, 可得到一组包含(时刻, 3D 点坐标)的数值序列 $[t_0, (x_0, y_0, z_0), t_1, (x_1, y_1, z_1), \dots, t_n, (x_n, y_n, z_n)]$ 。Cesium 内置的位置点类 (SampledPositionProperty) 相当于一个容器, 可以以(时刻, 3D 点坐标)的组合形式添加多个点对象, 使用的方法为 `addSample[ $t_i, (x_i, y_i, z_i)$ ]`, 这里用来存储飞行路径点集合。随后, 定义视点对象 (Entity), 并设置一系列参数, 如: `availability` (漫游起止时间)、`position` (位置)、`orientation` (方向)、`model` (glTF 视点模型), `model` 参数还包含 `uri` (模型链接) 和 `minimumPixelSize` (最小像素) 等信息。最后, 调用 Cesium 的 `viewer.entities.add()` 函数将视点对象添加到 3D 场景中, 启动飞行函数实现漫游功能。

2.2.2 回调函数漫游算法

回调函数漫游算法的核心思想是利用 Cesium 中的回调函数 (CallbackProperty) 实时更新视点位置。回调函数也被称为高阶函数, 是一个被作为参数传递给另一个函数的函数。当触发某一条件时,

系统会自动调用该函数。回调函数漫游算法使用线性插值来实时计算视点的位置坐标,并通过回调函数赋值给视点的 position 参数,实现视点位置按帧刷新。如图 5b 所示,视点 E 当前坐标为 (x, y, z) ,介于 P_{i-1} 和 P_i 两点之间,位于第 i 段折线上。依据空间解析几何中的点向式原理,计算三维空间直线上任意一点 $E(x, y, z)$ 的 3D 坐标的公式为:

$$\begin{cases} x = k(x_i - x_{i-1}) + x_{i-1}; \\ y = k(y_i - y_{i-1}) + y_{i-1}; \\ z = k(z_i - z_{i-1}) + z_{i-1}. \end{cases} \quad (3)$$

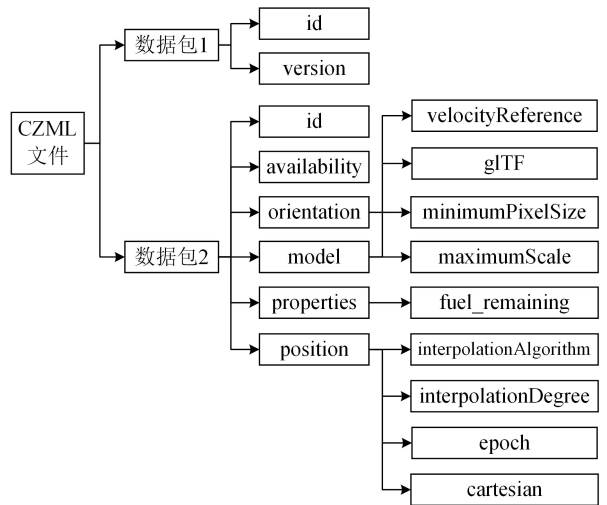
式中,比率 $k = |\overrightarrow{P_{i-1}E}| / |\overrightarrow{P_{i-1}P_i}|$,随时间 t 的变化而变化。由此可知, t_h 时刻视点飞行的总距离 $s_h = \Delta t \bar{v}$,其中, $\Delta t = t_h - t_0$ 。将 s_h 与 l_{i-1} 和 l_i 进行循环比较,直到满足条件 $l_{i-1} < s_h < l_i$ 时停止,则可判断视点飞行到第 i 段折线上;随后,计算视点在第 i 段折线上与该段折线起点 P_{i-1} 的距离差 $\Delta s_h = s_h - l_{i-1}$ 。根据上述点向式原理中 k 的定义可得 $k = \Delta s_h / \Delta l_i$,最后回带入式(3),即可得点 E 的坐标 (x, y, z) 的计算公式为:

$$\begin{cases} x = \frac{[(t_h - t_0)\bar{v} - l_{i-1}](x_i - x_{i-1})}{\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2 + (z_i - z_{i-1})^2}} + x_{i-1}; \\ y = \frac{[(t_h - t_0)\bar{v} - l_{i-1}](y_i - y_{i-1})}{\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2 + (z_i - z_{i-1})^2}} + y_{i-1}; \\ z = \frac{[(t_h - t_0)\bar{v} - l_{i-1}](z_i - z_{i-1})}{\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2 + (z_i - z_{i-1})^2}} + z_{i-1}. \end{cases} \quad (4)$$

2.2.3 拉格朗日插值漫游算法

拉格朗日插值漫游算法的核心思想是利用 Cesium 的 CZML 文件来实现漫游飞行功能。CZML 数据本质是一种 JSON 格式的字符串,其文件后缀为 .czml,用于描述与时间有关的动画场景,包含点、线、地标、模型和其他的一些图形元素,并指明了这些元素如何随时间而变化,使得 Cesium 炫酷地展示动态数据成为可能。每个 CZML 文件包含两个数据包,且每个数据包拥有一个 id(数据包的唯一值编码)。数据包 1 的 id 规定为“document”,包含了 Cesium 场景的时间轴范畴、当前时刻、倍速等信息;数据包 2 描述某一段时间内视点的行为。

CZML 文件的具体结构如图 6 所示。



Version. 版本号; properties. 属性; velocityReference. 参考速度; maximumScale. 最大缩放尺寸; fuel_remaining. 剩余燃料占比; interpolationAlgorithm. 内插算法; interpolationDegree. 内插间隔度数; epoch. 计时起始日期; cartesian. 实时更新的视点坐标序列。

图 6 CZML 文件结构

Fig.6 Structure of CZML file

然而,难点在于漫游路径需鼠标实时点选并绘制,不能以静态文件的形式存放于服务器中。本文设计了一种动态缓存 JSON 数据的算法以实现 CZML 文件的实时读写,将路径的坐标序列和视点模型信息记录在缓存的 CZML 文件中,以便实时传输和调用。在飞行进程中,视点的实时位置需通过鼠标拾取的路径点内插来动态更新。本文选择拉格朗日插值法,可实现较好的拟合效果。若已知 $y = f(x)$ 在互异的 $n+1$ 个点 $x_0, x_1, \dots, x_n$ 处的函数值 $y_0, y_1, \dots, y_n$,则可构造一个经过这 $n+1$ 个点的、不超过 n 次的多项式 $y = L_n(x)$,使其满足 $L_n(x_k) = y_k, k = 0, 1, \dots, n$ 。如估计任一点 ξ 的函数值 $y = f(\xi)$,则可利用 $L_n(\xi)$ 近似替代准确值 $f(\xi)$ 。拉格朗日插值多项式^[19]的定义为

$$L_n(x) = \sum_{i=0}^n y_i \prod_{\substack{j=0 \\ j \neq i}}^n \frac{x - x_j}{x_i - x_j} \quad (5)$$

如图 7 所示,设飞行路径由 P_0, P_1, P_2, P_3 4 个点组成, $P_i(x_i, y_i, z_i), i = 0, 1, 2, 3$ 。已知飞行速度为 $\bar{v}$,可计算视点飞行至 P_i 点处的时刻 $t_i = l_i / \bar{v}$ 。因点 E 的坐标 (x, y, z) 随时间 t 的变化而变化,故 x, y, z 的值都与 t 有关。设 $x = f_1(t), y = f_2(t),$

$z=f_3(t)$,依据拉格朗日插值多项式计算任意时刻 t 的 E 点坐标 $(\hat{x},\hat{y},\hat{z})$ 为

$$\left\{\begin{aligned}\hat{x}&=\frac{(t-t_1)(t-t_2)(t-t_3)}{(t_0-t_1)(t_0-t_2)(t_0-t_3)}f_1(t_0)+\\&\frac{(t-t_0)(t-t_2)(t-t_3)}{(t_1-t_0)(t_1-t_2)(t_1-t_3)}f_1(t_1)+\\&\frac{(t-t_0)(t-t_1)(t-t_3)}{(t_2-t_0)(t_2-t_1)(t_2-t_3)}f_1(t_2)+\\&\frac{(t-t_0)(t-t_1)(t-t_2)}{(t_3-t_0)(t_3-t_1)(t_3-t_2)}f_1(t_3);\\\hat{y}&=\frac{(t-t_1)(t-t_2)(t-t_3)}{(t_0-t_1)(t_0-t_2)(t_0-t_3)}f_2(t_0)+\\&\frac{(t-t_0)(t-t_2)(t-t_3)}{(t_1-t_0)(t_1-t_2)(t_1-t_3)}f_2(t_1)+\\&\frac{(t-t_0)(t-t_1)(t-t_3)}{(t_2-t_0)(t_2-t_1)(t_2-t_3)}f_2(t_2)+\\&\frac{(t-t_0)(t-t_1)(t-t_2)}{(t_3-t_0)(t_3-t_1)(t_3-t_2)}f_2(t_3);\\\hat{z}&=\frac{(t-t_1)(t-t_2)(t-t_3)}{(t_0-t_1)(t_0-t_2)(t_0-t_3)}f_3(t_0)+\\&\frac{(t-t_0)(t-t_2)(t-t_3)}{(t_1-t_0)(t_1-t_2)(t_1-t_3)}f_3(t_1)+\\&\frac{(t-t_0)(t-t_1)(t-t_3)}{(t_2-t_0)(t_2-t_1)(t_2-t_3)}f_3(t_2)+\\&\frac{(t-t_0)(t-t_1)(t-t_2)}{(t_3-t_0)(t_3-t_1)(t_3-t_2)}f_3(t_3).\end{aligned}\right. \tag{6}$$

将计算得到的坐标值赋给 CZML 文件中的视点的位置参数(positon 和 cartesian),即可动态更新视点位置。

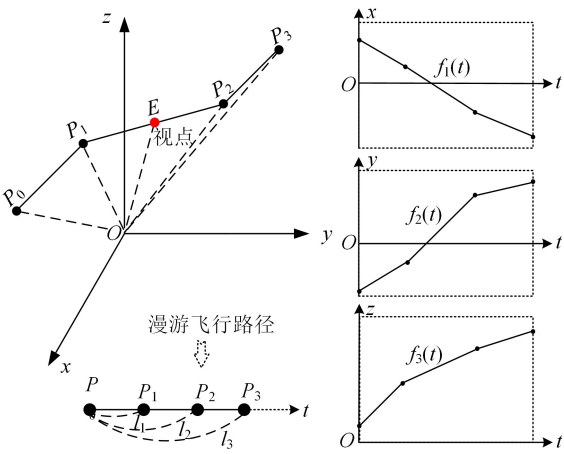


图 7 拉格朗日插值原理

Fig.7 Principle of the Lagrange interpolation

3 实验与分析

3.1 漫游功能实现

Cesium 漫游系统在基于 B/S 架构的 Web 端实现。实验的硬件环境为中央处理器 (central processing unit, CPU)为 Intel(R) Core(TM) i7 - 10750H,显卡为 NVIDIA GeForce GTX 1660 Ti (6 GB);软件环境为 Node.js 14、Cesium - 1.94.1、Google Chrome 101。实验区范围为重庆市区 (106°24 'E—106°36 'E,29°26 'N—29°39 'N)。

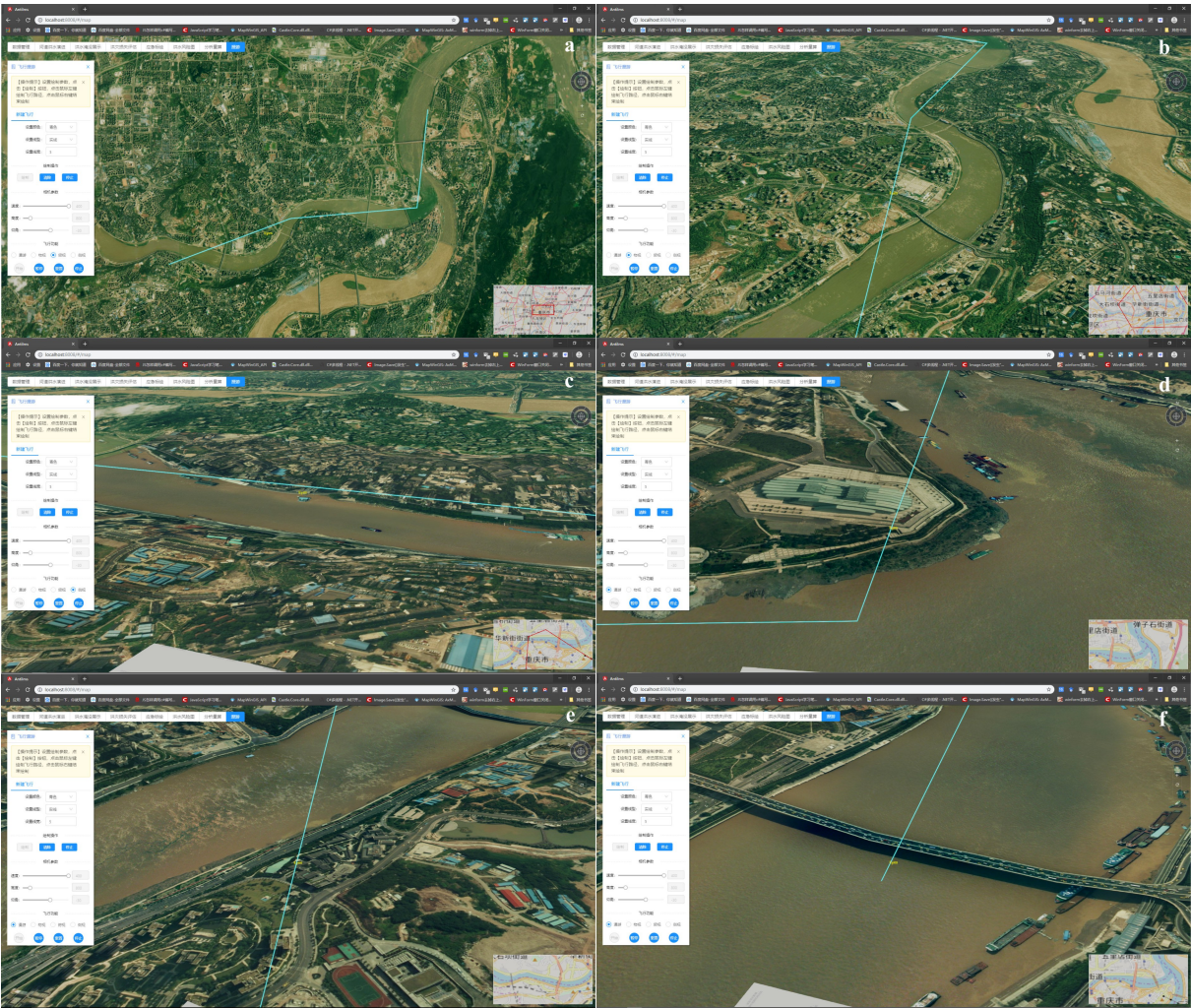
首先,登录系统,加载重庆市区的倾斜摄影模型。待数据加载成功后,在系统交互面板上设置绘制漫游路径的颜色、线型、线宽以及飞行速度、视点高度和仰角等参数。然后,在屏幕上用鼠标拾取任意 n 个点,由点连接成线,绘制漫游飞行路径。最后,点击开始按钮进行漫游飞行,在飞行过程中可调整视点的视角和姿态,进行俯视、侧视或物视。飞行过程中可见长江穿城而过,与嘉陵江在朝天门处交汇,呈现泾渭分明的两种颜色,以及黄花园和千厮门嘉陵江大桥、重庆大剧院、洪崖洞、朝天门广场等标志性景点和建筑。界面右下角的鹰眼视图可实时显示视点当前位置和可视范围。采用上述三种漫游算法均可实现如图 8 所示的漫游功能,整个飞行过程视觉效果流畅,VR 沉浸感强烈。

3.2 三种算法运行效率对比分析

为了定量分析三种算法在漫游时的渲染效率,需选取评估指标。帧率(frames per second, FPS)也称为刷新率,单位为 Hz,指画面每秒传输帧数。FPS 越大,视频画面越流畅^[20]。网络延迟(network latency, NL)单位为 ms。NL 越小,网速越快,反之则越慢。二者满足关系式:

$$y_{NL}=1000/x_{FPS} \tag{7}$$

式中: y_{NL} 为网络延迟; x_{FPS} 为帧率。FPS 与 NL 成反比。在动态可视化领域,常用这两个指标来定量评估算法的可视化效率。测试三种算法在相同软、硬件环境下前 50 s 的 FPS 和 NL,以作对比分析。实验时,为避免阻挡视线,需暂时移除倾斜摄影模型背景,以凸显漫游路径和效果。将实验重复进行 10 次取平均值以减小随机误差。图 9a—c 分别对应于三种漫游算法的运行效果图,图 9d 为前三幅图中红色椭圆虚线框内视点对象模型放大后的效果。



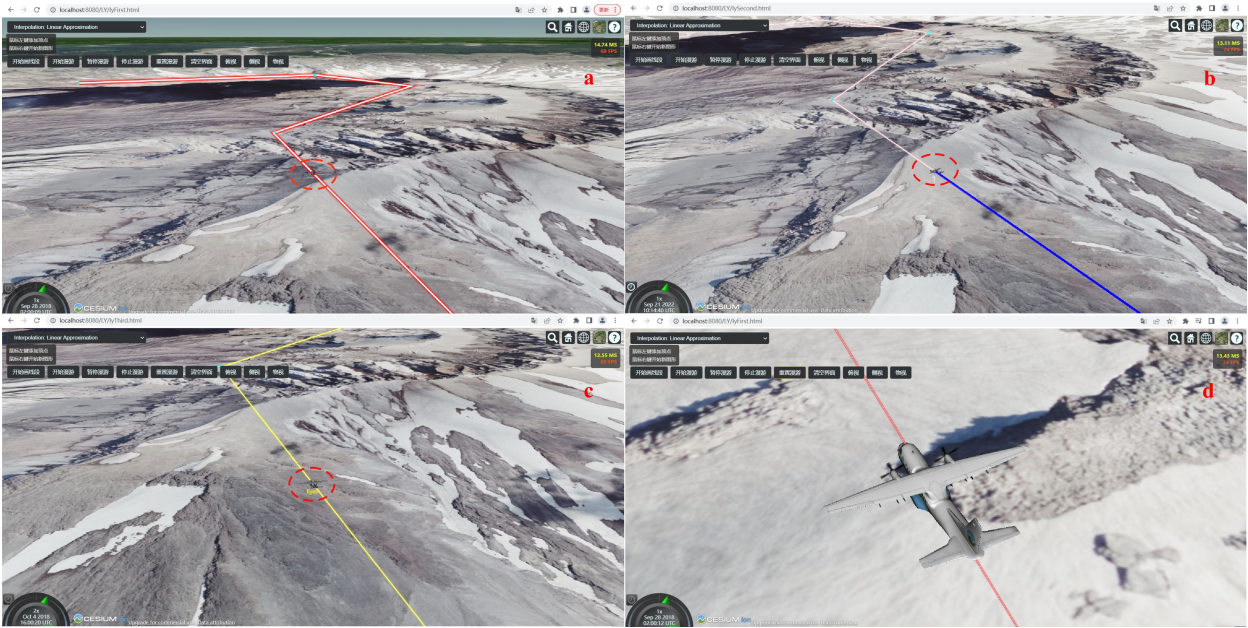
a. 绘制漫游路径;b. 视点开始沿路径飞行;c. 视点沿嘉陵江飞行侧视图;d. 视点飞跃重庆大剧院和朝天门广场;e. 视点横跨嘉陵江俯视图;f. 视点飞跃重庆黄花园嘉陵江大桥。

图 8 重庆市区漫游飞行效果图

Fig.8 Effect pictures of roaming in downtown Chongqing

图 10 为三种算法运行效率对比分析的折线图。由图 10a 可见:常规漫游算法的 FPS 基本维持在 70 Hz,且波动幅度不大;回调函数漫游算法的 FPS 保持在 74 Hz 上下,明显位于常规漫游算法之上;而拉格朗日插值漫游算法的 FPS 较前两种算法又提高了一个层次,维持在 80 Hz 左右。三者波动幅度均较为平缓。对比而言,三种算法的 NL 在某些时刻出现交错重叠,表明三者的差别并不大,但整体上还是存在一定的分层(图 10b):常规漫游算法的 NL 最高,波动幅度最大,表明其漫游飞行过程中的内核渲染效率最慢,且最不稳定;回调函数漫游算法的 NL 略高于拉格朗日插值漫游算法,但略低于常规漫游算法,表明回调函数漫游算法的运行效率居中。

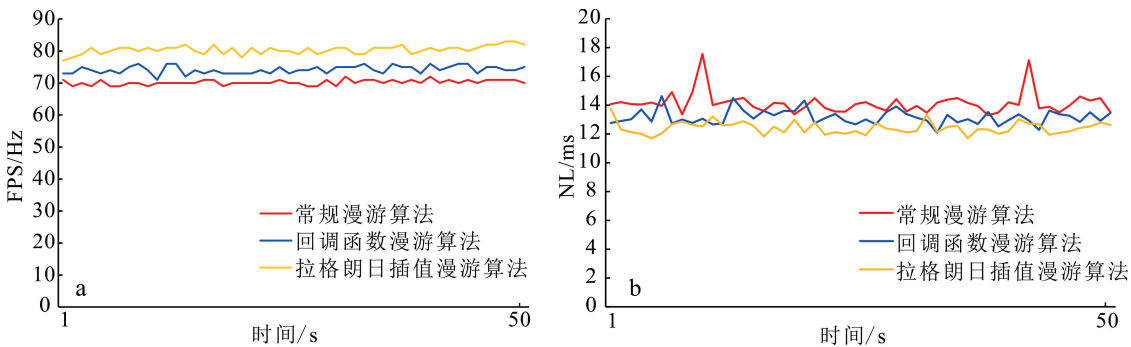
如表 2 所示:常规漫游算法的平均 FPS 为 70 Hz,平均 NL 为 14.145 ms;回调函数漫游算法的平均 FPS 为 74 Hz,较常规漫游算法提升了 5.714%,平均 NL 为 13.166 ms,较常规漫游算法缩短了 6.921%,表明该算法在收敛速度上比常规漫游算法更快;拉格朗日插值漫游算法的平均 FPS 为 80 Hz,较常规漫游算法提升了 14.286%,平均 NL 为 12.419 ms,较常规漫游算法缩短了 12.202%,其 FPS 优化率和 NL 缩短率均为最佳,视觉流畅度最高。总体而言,算法运行效率从大到小最终排序为拉格朗日插值漫游算法、回调函数漫游算法、常规漫游算法。



a. 常规漫游算法的运行效果;b. 回调函数漫游算法的运行效果;c. 拉格朗日插值漫游算法的运行效果;d. 视点对象模型。

图 9 三种漫游算法运行效果对比

Fig.9 Comparison of the operation effect of the three roaming algorithms



a. 三种漫游算法的 FPS 对比;b. 三种漫游算法的 NL 对比。

图 10 三种漫游算法运行效率折线图

Fig.10 Line charts of the running efficiency of the three roaming algorithms

表 2 三种漫游算法运行效率对比分析结果

Table 2 Comparative analysis results of the running efficiency for the three roaming algorithms

算法名称	编程语言	平均 FPS/Hz	FPS 优化率/%	平均 NL/ms	NL 优化率/%
常规漫游算法	JavaScript	70	0	14.145	0
回调函数漫游算法	JavaScript	74	5.714	13.166	6.921
拉格朗日插值漫游算法	JavaScript	80	14.286	12.419	12.202

4 讨论

整体而言,常规漫游算法未经优化,运行效率最低;回调函数漫游算法通过构造回调函数来动态更

新视点坐标,可提升运行效率;拉格朗日插值漫游算法采用拉格朗日插值法计算视点坐标,动态构建 CZML 文件来实现漫游功能,CZML 文件的数据量比可扩展标记语言(extensible markup language,

XML)更小,更易于网络传输和机器解析生成。回调函数漫游算法计算视点坐标时,所使用的点向式插值是一种简单的线性插值,拟合效果较差,而拉格朗日插值漫游算法所使用的拉格朗日插值在路径点较为密集的情况下,拟合效果更好。因此,拉格朗日插值漫游算法拥有比前两种算法更高的渲染效率和视觉流畅度。

目前,Cesium 是最受欢迎的 Web 端开源 3D 虚拟地球渲染引擎,在许多 3D GIS 中,涉及到一些复杂的功能,如数字地形模型的组织和调度、3ds Max 模型的加载和可视化、海量矢量和栅格数据的集成等。因浏览器内核承载力有限,面对大数据量时容易出现延迟或卡顿现象。如:Cheng 等^[20]在研究大区域(面积为 927.86 km²)范围内的气体泄露与扩散问题时,虚拟地球表面气体扩散速率的平均FPS为 41.45 Hz;Liang 等^[21]在模拟地球表面 Isabel 飓风行进过程时,采用复杂体积投影算法研究大范围区域(83°W—62°W,23.7°N—41.7°N)内的飓风数据,渲染的平均FPS低于 50 Hz(2 000~4 147 km 视距范围);Yang 等^[22]在研究地球表面粉尘颗粒分布可视化时,采用千万级顶点的 NetCDF 数据和多核 CPU 多线程八叉树的算法,渲染的平均FPS低于 30 Hz。通常,避免动态可视化不流畅的最低 FPS 是 30 Hz。因此,面对大范围场景和海量数据时,系统对运行算法的效率提出更高要求。

本文所提出的三种漫游算法均未涉及大区域范围或海量数据,渲染区域仅限于重庆市区,故平均FPS均高于 70 Hz。若涉及到大区域范围内海量异构数据场景的漫游飞行问题,需要对算法进行更深层次的优化。Cesium 的优点之一在于可使用图形处理器(graphics processing unit, GPU)对场景进行高效渲染,GPU 的并行计算能力极大地增强了 3D 图形渲染的功效。因此,利用 GPU 并行计算对大范围异构数据地理环境的漫游算法进行优化,将会是下一个研究重点。

5 结论

本文基于 Cesium 设计了三种不同的漫游算法,以重庆市主城区为例实现了漫游功能,并对比分析了三种算法的运行效率,得出如下结论:

1)常规漫游算法使用 Cesium 的位置点类存储(时刻,3D 点坐标)序列来实现漫游功能,回调函数漫游算法的核心是构造回调函数来动态更新视点位

置坐标,拉格朗日插值漫游算法利用拉格朗日插值法动态计算视点实时位置并生成 CZML 文件以实现漫游飞行。

2)从 FPS 和 NL 两个方面对三种算法的渲染效率进行对比,结果表明,渲染效率从大到小最终排序为拉格朗日插值漫游算法、回调函数漫游算法、常规漫游算法,即回调函数漫游算法和拉格朗日插值漫游算法均在常规漫游算法的基础上实现了一定程度的优化,且拉格朗日插值漫游算法的运行效率最佳。

参考文献(References):

- [1] Sheppard S, Cizek P. The Ethics of Google Earth: Crossing Thresholds from Spatial Data to Landscape Visualization [J]. Journal of Environmental Management, 2009, 90(6): 2102–2117.
- [2] Chen Q Y, Liu G, Ma X G, et al. A Virtual Globe-Based Integration and Visualization Framework for Aboveground and Underground 3D Spatial Objects[J]. Earth Science Informatics, 2018, 11(4): 591–603.
- [3] 王锐, 钱学雷. OpenSceneGraph 三维渲染引擎设计与实践[M]. 北京: 清华大学出版社, 2009.
Wang Rui, Qian Xuelei. OpenSceneGraph 3D Rendering Engine Design and Practice[M]. Beijing: Tsinghua University Press, 2009.
- [4] Gong Jianya, Chen Jing, Xiang Longgang, et al. GeoGlobe: Geo-Spatial Information Sharing Platform as Open Virtual Earth [J]. Acta Geodaetica et Cartographica Sinica, 2010, 39(6): 551–553.
- [5] 高辉, 张茂军. 基于真实地景的三维飞行再现系统的设计与实现[J]. 计算机工程, 2006, 32(12): 232–234.
Gao Hui, Zhang Maojun. Design and Realization of 3D Flight Replay System Based on Real Terrains [J]. Computer Engineering, 2006, 32(12): 232–234.
- [6] 曹晶, 邵远征, 张煜, 等. 大范围地形及城市三维快速建模与漫游[J]. 长江科学院院报, 2008, 25(4): 40–43, 47.
Cao Jing, Shao Yuanzheng, Zhang Yu, et al. Fast Modeling and Roaming of Large Terrain[J]. Journal of Yangtze River Scientific Research Institute, 2008, 25(4): 40–43, 47.
- [7] 段忠祥. 基于 OpenGL 的飞行漫游系统的设计[J]. 广西轻工业, 2008(8): 102–104.
Duan Zhongxiang. Design of Flight Roaming System Based on OpenGL [J]. Guangxi Journal of Light

- Industry, 2008(8): 102 - 104.
- [8] Kim H W, Kim D S, Lee Y W, et al. 3 - D Geovisualization of Satellite Images on Smart Devices by the Integration of Spatial DBMS, RESTful API and WebGL[J]. Geocarto International, 2015, 30 (4): 422 - 440.
- [9] Krooks A, Kahkonen J, Lehto L, et al. WebGL Visualization of 3D Environmental Models Based on Finnish Open Geospatial Data Sets[J]. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2014(3):163 - 169.
- [10] Chen B, Xu Z. A Framework for Browser-Based Multiplayer Online Games Using WebGL and WebSocket[C]// 2011 International Conference on Multimedia Technology. Hangzhou: IEEE, 2011: 471 - 474.
- [11] Lei F, Wang C, Li C, et al. A Research for 3D WebGIS Based on WebGL[C]// Proceedings of 2011 International Conference on Computer Science and Network Technology. Harbin: IEEE, 2011: 348 - 351.
- [12] 李俊金. 基于 3D GIS Cesium 的数字城市建模技术[J].信息 with 电脑, 2016(19): 45 - 46.
- Li Junjin. Digital City Modeling Technology Based on 3D GIS Cesium[J]. Information and Computer, 2016 (19): 45 - 46.
- [13] 方孟元, 罗年学, 许毅, 等. 基于 Cesium 的 BIM 与实景三维模型集成可视化研究[J].测绘地理信息, 2022, 47(2): 111 - 114.
- Fang Mengyuan, Luo Nianxue, Xu Yi, et al. Research on Integrated Visualization of BIM and Real-Scene 3D Model Based on Cesium[J]. Journal of Geomatics, 2022, 47(2): 111 - 114.
- [14] 郭晓非, 朱俊利, 万剑华, 等. 基于 Cesium 的海洋环境要素三维可视化研究[J].海洋科学, 2021, 45(5): 130 - 136.
- Guo Xiaofei, Zhu Junli, Wan Jianhua, et al. 3D Visualization of Marine Environmental Elements Based on Cesium[J]. Marine Sciences, 2021, 45(5): 130 - 136.
- [15] 张永强, 王波, 申茂廷. 基于 Cesium 的 3DWebGIS 三维场景加载及开发[J].河南科技, 2021, 40(21): 8 - 10.
- Zhang Yongqiang, Wang Bo, Shen Maoting. 3D Scene Loading and Development of 3DWebGIS Based on Cesium[J]. Henan Science and Technology, 2021, 40(21): 8 - 10.
- [16] 李俊金. 基于 Cordova 和 Cesium 的移动 3DWebGIS 系统实现[J].电子技术与软件工程, 2017(8): 55 - 57.
- Li Junjin. Implementation of Mobile 3D WebGIS System Based on Cordova and Cesium [J]. Electronic Technology and Software Engineering, 2017 (8): 55 - 57.
- [17] 熊英. 基于 Cesium 3D Tiles 技术的模型分级加载研究[J].人民珠江, 2019, 40(8): 117 - 122.
- Xiong Ying. Research on Models Loading with LOD Based on Cesium 3D Tiles[J]. Pearl River, 2019, 40 (8): 117 - 122.
- [18] 孙晓鹏, 张芳, 应国伟, 等. 基于 Cesium.js 和天地图的三维场景构建方法[J].地理空间信息, 2018, 16 (1): 65 - 67.
- Sun Xiaopeng, Zhang Fang, Ying Guowei, et al. 3D Scene Construction Method Based on Cesium.js and Map World [J]. Geospatial Information, 2018, 16 (1): 65 - 67.
- [19] 刘金健. 基于拉格朗日插值法的 GPS 卫星轨道位置拟合[J].科技创新与生产力, 2018, 294(7): 11 - 13.
- Liu Jinjian. Fitting of GPS Satellite Orbit Location Based on Lagrange Interpolation Method[J]. Sci-tech Innovation and Productivity, 2018, 294(7): 11 - 13.
- [20] Cheng R Z, Chen J, Cao M. A Virtual Globe-Based Three-Dimensional Dynamic Visualization Method for Gas Diffusion [J]. Environmental Modelling and Software, 2019, 111:13 - 23.
- [21] Liang J M, Gong J H, Li W H, et al. Visualizing 3D Atmospheric Data with Spherical Volume Texture on Virtual Globes[J]. Computers & Geosciences, 2014, 68(7): 81 - 91.
- [22] Yang C W, Wu H Y, Huang Q Y, et al. Using Spatial Principles to Optimize Distributed Computing for Enabling the Physical Science Discoveries [J]. Proceedings of the National Academy of Sciences of the United States of America, 2011, 108(14): 5498 - 5503.