文章编号:1671-4598(2025)05-0230-09 DOI:10.16526/j.cnki.11-4762/tp.2025.05.027 中图分类号:TP391.4 文献标识码:A

基于 K-Means++和 GeoHash 的海量目标渲染系统的设计与实现

孔维江，王　宇，陈　露，陈盼君，黄途文
（航天恒星科技有限公司，北京　100086）

摘要：处理和渲染的紧迫问题；针对数据规模和复杂性，以及图像展示设备的限制，传统渲染方法无法满足实时性、高性能和基于经纬度的场景需求，提出了一种创新的海量目标渲染系统；该系统基于 Hadoop 分布式架构，首次将 K-Means++算法和 GeoHash 地理编码技术相结合，最后引入基于目标聚合的渲染策略，实现了海量目标数据的高效分类和编码，从而提高了渲染效率和质量；经实验测试和对比分析，该系统在海量数据集渲染方面具有高速性、稳定性、拓展性以及良好可视化等优势；满足了实时性、自适应性以及基于经纬度渲染的需求，具有广阔的应用前景。

关键词：K-Means++算法；GeoHash 编码；海量数据；目标渲染；分布式架构

Design and Implementation of a Massive Target Rendering System Based on K-Means ++ and GeoHash

KONG Weijiang, WANG Yu, CHEN Lu, CHEN Panjun, HUANG Tuwen
(Space Star Technology Co., Ltd., Beijing　100086, China)

Abstract: To solve the urgent issue of processing and rendering the massive data of targets in the digital era, and due to the limitations of data scale and complexity, as well as image display devices, traditional rendering methods cannot meet the requirements of real-time, high-performance and latitude and longitude based scenarios. Therefore, an innovative massive target rendering system is proposed. Based on the Hadoop distributed architecture, the system combines the K-Means ++ algorithm with the GeoHash geocoding technology for the first time. Finally, a rendering strategy based on target aggregation is introduced to realize the efficient classification and coding for the massive data of targets, thus improving the rendering efficiency and quality. After experimental testing and comparative analysis, the system has the advantages of high speed, stability, extensibility and good visualization in rendering massive data sets. It meets the requirements of real-time, adaptability, and latitude and longitude based rendering, and has a broad application prospect.

Keywords: K-Means ++ algorithm; GeoHash encoding; massive data; target rendering; distributed framework

0　引言

海量目标渲染是指利用计算机图形学技术，将大量具有空间位置信息的目标数据在二维或三维空间中进行可视化展示的过程[1]。海量目标渲染是信息处理领域的一个重要研究方向，它可以为各行各业的决策和管理提供直观和有效的信息支持[2]。然而，海量目标渲染也面临着诸多挑战，如数据的规模和复杂性、渲染的效率和质量、客户端的性能和压力等[3]。传统的海量目标渲染方法，往往采用集中式的架构，将数据存储在单一的服务器上，然后通过客户端进行数据的请求和渲染[4]。这种方法往往会面临稳定性差，共享性差，拓展性差，成本高，渲染效率低等问题，已经无法满足对海量数据的实时处理和可视化需求，因此如何高速地处理和渲染海量的目标数据已经成为当今信息处理领域的一个重要挑战。

近几年，许多研究者为了提升海量目标渲染的效率和效果做出了许多研究。目前对提升海量目标渲染效率和效果的研究主要集中在以下几个方面：1）优化数据组织和管理：对最底层数据的格式进行优化，从提高数

收稿日期:2024-11-14；　**修回日期**:2024-04-08。
基金项目:国家能源安全应用项目(F1 子项目)。
作者简介:孔维江(1996-),男,硕士,工程师。
通讯作者:黄途文(1986-),男,大学本科,工程师。
引用格式:孔维江,王　宇,陈　露,等.基于 K-Means++和 GeoHash 的海量目标渲染系统的设计与实现[J].计算机测量与控制,2025,33(5):230-238.

据的存储和检索效率的方面进而提升数据的渲染效率，例如文献［5］提出了一种基于四叉树的空间索引结构，将海量目标数据划分为不同层次的网格，实现了数据的快速检索[5]；文献［6］设计出一种新的聚类方法 实现了数据的动态加载和更新[6]；2）数据分类和编码：即从空间和位置的层面上对数据进行优化，从提高数据的传输和查询效率的方面提高数据的传输和查询效率，例如文献［7］提出了一种基于 K-means 算法的数据分类方法，将海量目标数据按照密度和分布进行聚类，实现了数据的高效组织和管理[7]；文献［8］提出了一种基于 GeoHash 的空间数据管理索引，实现了数据的紧凑表示和快速检索[8]；3）数据渲染和展示：即直接对渲染算法进行优化，从而提升渲染效率和质量。在渲染效率方面，文献［9］提出了一种基于 Shader 的 CSG 几何体的实时渲染方法，从而提高了渲染效率[9]。在渲染效果方面，文献 10 至 13 分别对基于纹理特征的渲染方法进行了优化，并提出了多种渲染方案，从而改善了渲染效果和质量[10-13]；另一方面文献［14］了一种基于几何和图像的混合方法，优化了渲染效果[14]。

上述这些方法在改进和完善目标数据渲染方法等方面取得了显著进展，对于提升渲染效果和加快渲染速度都起到了积极作用。然而，尽管这些方法在许多方面都具有优势，但仍然存在以下两个问题，目前尚未给出明确的解决方案：1）目前对于大规模目标数据在存储和实时读取方案的问题尚未找到明确的解决方案。在存储和查询方面，关系型数据库（如 MySQL、PostgreSQL 等）存在一定的局限性。当目标数据量达到一定数量级别时，这些数据库的查询效率明显下降，进而导致瓦片或目标的渲染速度变慢，无法满足海量目标数据实时渲染的需求；2）尽管现有方法在目标渲染中取得了一定的成果，但其基于笛卡尔直角坐标系的局限性限制了其使用范围。对于需要以大地坐标为基础进行渲染的场景，需要进一步探索和发展适用于大地坐标系统的渲染方法，并解决与大地坐标系相关的数据存储、查询和转换等问题。另一方面，这些方法存在以下局限性：1）稳定性不足。当数据量过大或者客户端过多时，服务器的负载会增加，导致响应速度下降，甚至出现崩溃或者数据丢失的情况；2）共享性不足。由于数据存储在单一的服务器上，不同的客户端之间难以实现数据的共享和协作，也不利于数据的更新和维护；3）拓展性不足。当数据量增加或者数据类型变化时，需要对服务器进行扩容或者升级，这会增加成本和风险。

为了解决上述问题并实现海量目标数据的实时渲染，从而提升海量数据集的渲染效率同时优化渲染效果。本研究首次将 K-Means＋＋算法和 GeoHash 地理编码技术相结合（表 1），并在此基础上进一步考虑了数据存储、数据读取、使用场景以及渲染效果等因素。最终，设计出一套基于 K-Means＋＋算法和 GeoHash 地理编码的海量目标渲染系统。相比于单节点的传统渲染方式，该系统具有以下优势：1）Hadoop 分布式架构：该系统将数据分散存储在多个服务节点上，从而提升了系统的可靠性、共享性及稳定性；2）该系统利用 K-Means＋＋算法对海量目标数据进行动态聚类，根据数据的密度和分布，自适应地确定最佳的聚类中心和聚类个数，提升了数据的组织和管理效率；3）该系统利用 GeoHash 地理编码技术对聚类中心进行编码，将聚类中心的空间位置信息转化为一维的字符串，提升了数据的压缩和索引效率；4）基于目标聚合的渲染策略：根据客户端的视角和视距，将一个簇内的数据进行聚合和计数，实现了数据的自适应和优化，提高了渲染的效果和质量。

综上所述，该系统一方面提高了海量目标数据的渲染效率和质量，低了客户端的计算和存储压力；另一方面，该系统满足了基于经纬度场景的监测需求，为海量目标渲染提供了一种新的解决方案，具有广阔的应用前景。

表 1　本研究使用的技术与其他类似研究使用的技术之间的比较表

序号	文献	研究方法			结果	
		数据组织	数据编码	渲染算法	提升效率	优化效果
1	（张芩等人.，2004；杨帆等人.，2007）	√			√	
2	（Shi 等人.，2010；Liu 等人.，2014）		√		√	
3	（陈国军等人.，2022）			√	√	
4	（彭湃等人.，2017；Shin 等人.，2018；Caban 等人.，2008；Hu 等人.，2022.，Debevec 等人.，1996）			√		√
5	基于 K－means＋＋和 GeoHash 的海量目标渲染系统	√	√	√	√	√

1　K-Means＋＋算法与 GeoHash 编码

1.1　K-Means＋＋聚类算法

K-Means＋＋算法是一种经过改进的聚类算法，其目的在于解决聚类问题[15]。传统的 K-means 算法在初始聚类中心的选择方面存在明显的缺陷。此外，K 值的选取对 K-means 算法的稳定性也具有重要影响。为了提升聚类算法的稳定性和效果，K-Means＋＋算法对初始质心的选择进行了优化。该优化策略旨在确保初始聚类中心之间的距离尽可能最大，以增强算法的效果。

K-Means＋＋算法的核心思想是根据数据点之间的距离来确定初始聚类中心的选择[16]。该算法的第一步是随机选择一个数据点作为第一个聚类中心，接着按照

一定顺序选择下一个聚类中心，直到选取出 k 个聚类中心为止。K-Means++算法在选择聚类中心时，会通过计算每个数据点与已选中聚类中心的最短距离，并选取最远距离的数据点作为下一个聚类中心[16]。这种方法可以有效地减少聚类结果的重复性，同时提高聚类的效果。该算法的具体步骤（如图 1 所示）如下：

1）从数据集中随机选择 1 个数据点作为初始质心，并将其标记为 P_1；

2）对于尚未选择的数据集中的每个数据点 EPi（$i=1, 2, 3\cdots n$），计算并保存其与已知初始质心之间的最短距离；

3）基于加权概率分布，依据各数据点与已知初始质心的最短距离成正比的概率，随机选择一个新的数据点作为质心。

4）重复计算上述，直到选择了 k 个质心即 $n=k$。

5）应用 K-Means++聚类算法得到最终的 k 个质心，并获得点的分配结果。

1.2 GeoHash 地理编码

GeoHash 是一种高效的地址编码方法，由学者 Gustavo Niemeyer 于 2008 年提出[17]。它通过将地球划分为多个矩形区域，并为每个区域分配一个唯一的编码来实现其功能。其核心思想是对地球进行区域二分，即将地球视为一个二维平面，并将其递归均分为更小的子块[18]。这种编码方式具有几个显著特点。首先，GeoHash 能够将地理位置信息转换为字符串表示形式，从而方便进行存储和传输。这样的表示形式可以简化数据的处理和管理，使得地理位置信息更易于处理。其次，在 GeoHash 中，相似的地理位置具有相近的编码。这意味着我们可以通过比较编码来快速计算地理位置之间的距离和相似度。这种特性对于许多地理位置相关的任务，如位置搜索和地理聚类，都非常有用。第三，GeoHash 的编码长度可以根据需求进行调整。这就意味着我们可以在精度和存储空间之间找到平衡点。较长的编码长度可以提供更高的精度，但会占用更多的存储空间，而较短的编码长度则可以减少存储要求，但也可能导致精度降低。具体而言，GeoHash 的编码过程可以按照（图 2）以下步骤进行：

1）将地球表面的经度范围［－180，180］和纬度范围［－90，90］分别映射到对应的二进制编码空间；

2）初始时，将经度的范围［－180，180］和纬度的范围［－90，90］平均分为两个区间，用 0 和 1 分别表示，其中左区间记作 0，右区间记作 1；

3）对于每个区间，再次将其平均分为两个子区间，继续用 0 和 1 表示。

4）重复上述步骤，不断细分每个区间，并采用 Z 字曲线对区域进行编码，直到达到所需的精度或者编码长度为止。

最终，根据每个区间的划分情况，生成一个由 0 和 1 组成的编码字符串。然后，将该编码按照 5 位为一组进行划分，并使用 Base32 编码的方式计算出每组的十进制值，从而得到该位置的 GeoHash 编码。通过这种方式，相邻的区域在 GeoHash 中具有相近的编码，可以通过比较编码来快速计算地理位置间的距离和相似度。同时，GeoHash 的编码长度可以根据需要进行调整，从而在精度和存储空间之间进行权衡。

1.3 基于 K-Means++和 GeoHash 的结合算法

综上所述，K-Means++通过在选择初始聚类中心时考虑样本之间的距离，通过最小化数据点与中心点之

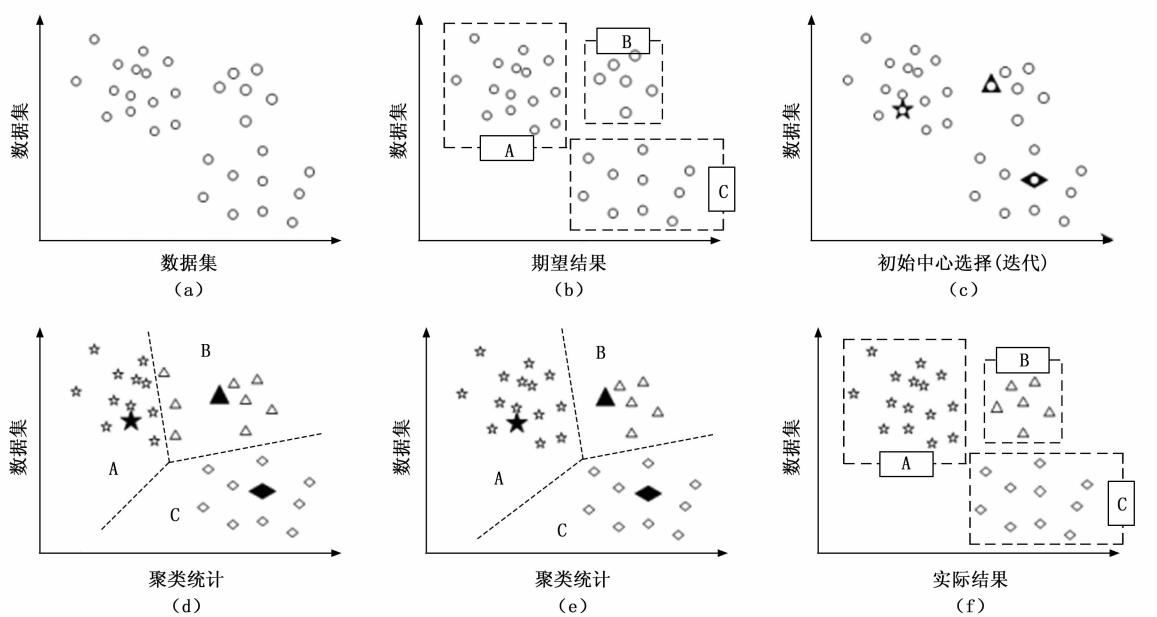

图 1 K-Means++流程示意图

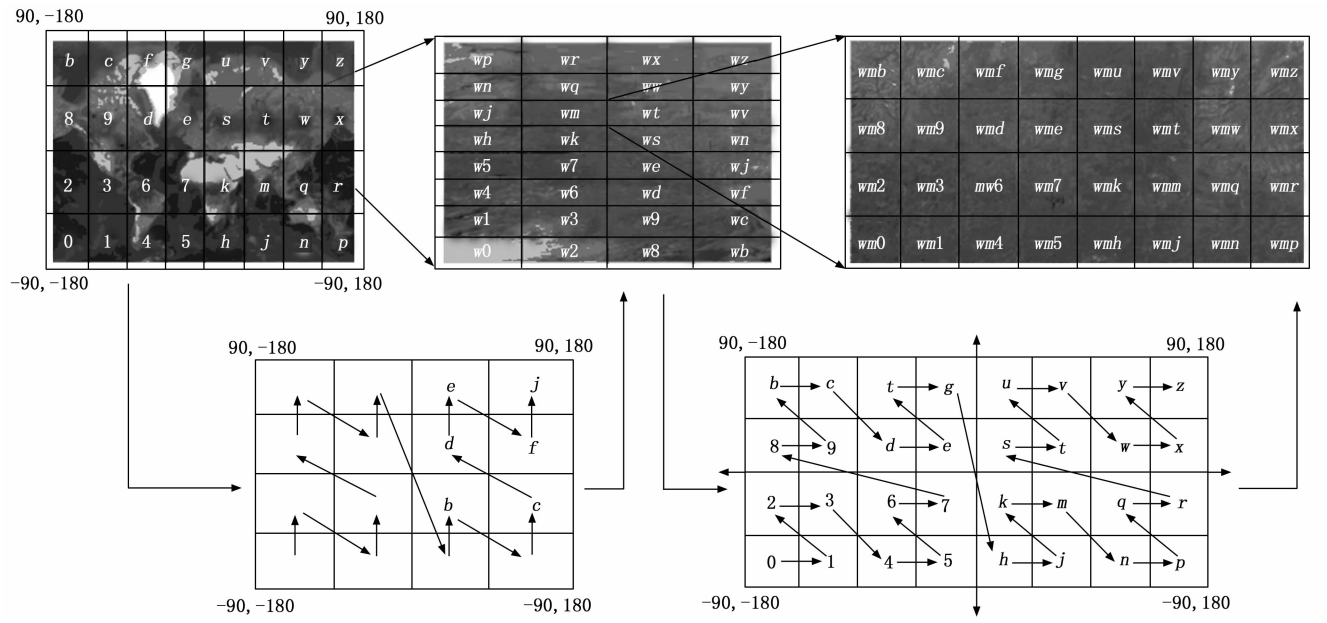

图 2　GeoHash 编码流程示意图

间的距离来划分数据集进而寻找数据集中的相似群体，其有效提升数据管理效率[19]。GeoHash 可以将地理坐标转化为一串字符，用于空间上进行快速的检索和查询。将 K-Means＋＋分类算法和 GeoHash 地理编码技术相结合，可以在空间数据集中进行聚类分析，准确来说，可以更精细地实现对空间数据的聚类和检索。首先使用 K-Means 算法对地理坐标进行聚类，将地理位置点划分为不同的簇，提高聚类的准确性和稳定性；其次利用 GeoHash 技术进行编码，将连续的经纬度坐标转换为离散的编码，减少数据量，提高聚类的效率；此外，为了优化最终渲染的效果，在两者结合进行渲染的同时进一步引入基于目标聚合的渲染策略，实现同簇数据的快速统计。该算法利用了 K-Means＋＋算法和 GeoHash 地理编码技术的优势，实现了海量目标数据的高效分类和编码，不仅解决了大规模数据集在数据管理和检索方面效率的问题，提高了渲染的质量；另一方面，该算法能够有效地处理大规模地理位置数据，快速压缩数据的体积，完美解决了传统渲染方法不能适用于基于经纬度的场景需求。

2　系统的架构设计

对于海量目标渲染系统的设计，本研究主要采用“分布式云架构”的设计思想，将整个渲染系统分为 7 个层次，分别是提供基础运算环境的设施层，用于进行数据存储的存储层，进行数据集分类运算的数据分类层，进行地理空间编码的数据编码层，提供数据处理及对接的数据控制层，提供服务接口的服务层以及应用层，详细构架如图 3 所示。

1）设施层，作为关键层级，通过采用虚拟化技术，

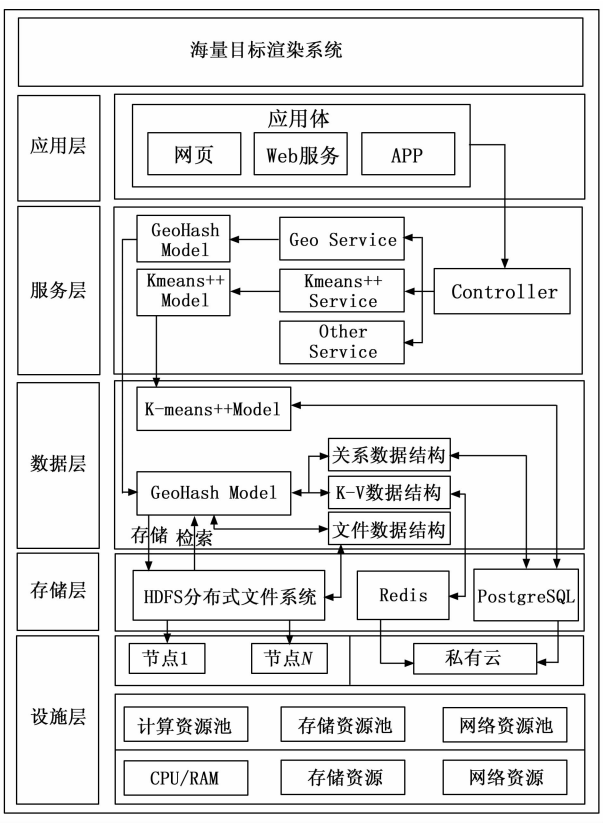

图 3　目标数据渲染系统架构设计图

对物理计算机的 CPU、内存和网络进行虚拟化，从而创建了计算资源池、存储资源池和网络资源池。这样的虚拟化环境为目标数据渲染提供了高效的计算环境[20]。

2）存储层：存储层的主要功能是提供可靠的数据存储和访问能力。本研究在该层级中分布部署 HDFS 分布式文件系统，PostgreSql 数据库以及 Redis 缓存数据库分别存储上层处理完成的各种数据。PostgreSql 和

HDFS需要进行数据互通和共享，同时，该层也负责管理目标数据的读取和写入操作，使得系统能够方便地对数据进行访问和修改。

3）数据层：数据层是进行业务逻辑处理的一个层级，主要功能是根据客户端传递过来的需求，对需求进行识别并对海量目标数据集进行聚类和编码以及下发等操作，以满足不同应用场景对于目标数据渲染的需求。该层又可以细分为数据分类层和数据编码层；(1) 数据分类层：该层负责在每个服务器上对负责的数据进行聚类，实现了数据的高效组织和管理。该层主要采用了K-Means＋＋算法对数据进行聚类，同时维护了一个局部的聚类表，记录了每个聚类的中心和个数以及同簇中数据的总数，同时将该数据维护到数据层的 PostgreSql 数据库中。(2) 数据编码层：该层负责在每个服务器上对负责的数据进行编码，实现了数据的紧凑表示和快速检索。该层采用了 GeoHash 地理编码技术对数据进行编码，同时维护了一个局部的编码表，记录了每个数据的编码和属性，并存放至存储层的 Redis 中。

4）服务层：服务层主要负责提供服务接口给上层模块。本研究将 K-Means＋＋聚类和 GeoHash 地理编码等操作全部封装成一个个小模块放入服务层中，从而实现海量目标的并行计算，从而提升渲染的效率。

5）数据渲染层：该层负责在客户端上对请求的数据进行渲染和展示，实现了数据的自适应和优化。该层根据客户端的视角和视距，动态地展示最终目标渲染的效果。

3 系统与渲染算法的实现

3.1 系统的构建与配置

本研究实现了一种基于 K-Means＋＋和 GeoHash 的海量目标渲染系统，该系统在 Hadoop 分布式架构的基础上，结合了 K-Means＋＋算法、GeoHash 地理编码技术以及目标聚合的渲染策略，实现了海量目标数据的高效分类和编码，提高了渲染的效率和质量。用于构建海量目标渲染系统的方法（图 4）简要介绍如下。

1）虚拟环境设置：为了实现多节点的分布式架构，本研究在 Windows 10 操作系统上安装 VMware Workstation Pro 软件。利用此软件虚拟出多台虚拟机，并在虚拟机的基础上继续安装 Linux 操作系统，使该系统在不消耗物理硬件设施的情况下完成大量目标数据的存储、检索和渲染；

2）基础环境配置：为了实现多台虚拟机之间的相互通信，需要在各台虚拟机上进行修改 IP 地址、修改主机名、时间校正、实现域名映射、关闭防火墙等一系列操作；

3）Hadoop 数据处理平台的接入：为了提升大规模数据集的存储效率和计算速度，需要引入分布式架构，本研究选择 Hadoop 架构作为目标渲染系统的基础架构，具体操作主要需要进行 HDFS 的相关配置、2 MapReduce 的相关配置、Yarn 的相关配置；

4）基于 K-Means＋＋与 Geohash 的渲染算法设计：采用 SpringBoot 框架结合 Redis Geo 设计目标渲染算法，从而提升海量目标的渲染效率。

本系统使用 CentOS 7.9（CentOS-7-x86 _ 64-Minimal-2009 _ 2.iso）作为镜像，以 Apache Hadoop-3.2.1 作为支撑架构，以 Postgresql-14.10 和 Redis 6.2.14 作为系统数据层的数据库，以实现具有 Linux 系统的计算节点的虚拟化和基础环境配置。具体的服务器情况如表

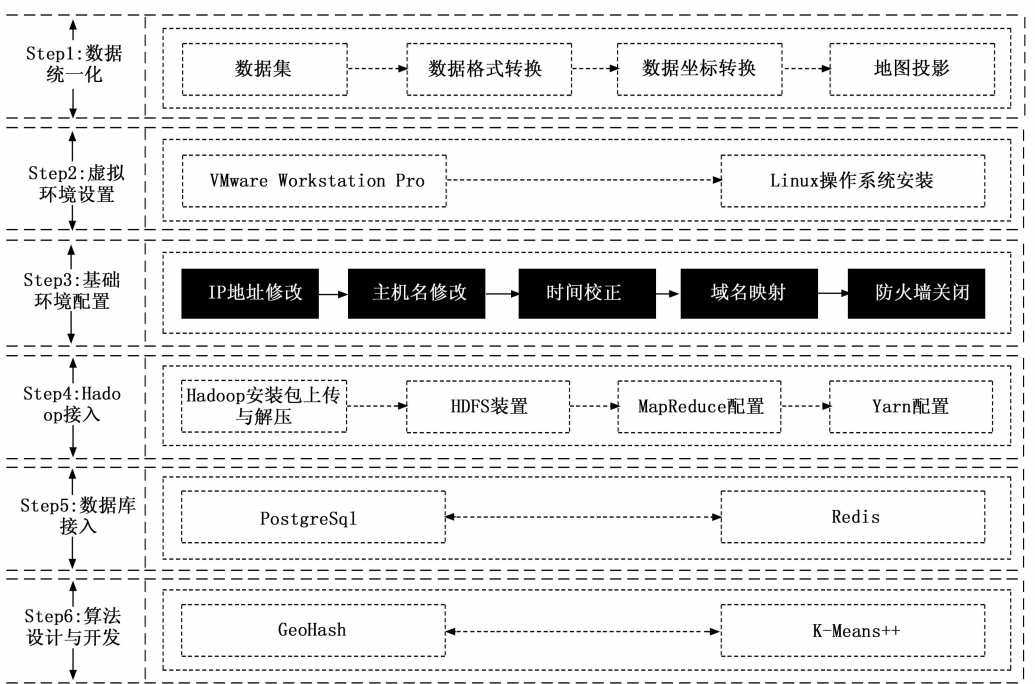

图 4 目标数据渲染系统构建流程图

2 所示。

表 2　目标数据渲染系统配置表

类型	IP	主机名	CPU	RAM	硬盘	备注
虚拟机	192.168.0.161	v1.bdglpt,com.cn	16	64 GB	942 GB	
虚拟机	192.168.0.162	v2.bdglpt,com.cn	4	16 GB	943 GB	
虚拟机	192.168.0.163	v3.bdglpt,com.cn	16	16 GB	937 GB	
虚拟机	192.168.0.164	v4.bdglpt,com.cn	16	20 GB	939 GB	
虚拟机	192.168.0.165	v5.bdglpt,com.cn	16	32 GB	966 GB	
虚拟机	192.168.0.166	v6.bdglpt,com.cn	16	32 GB	957 GB	
物理机	192.168.0.171	hdfs1.cmpc.com.cn	8	32 GB	1.1 T	HDFS
物理机	192.168.0.172	hdfs2.cmpc.com.cn	8	32 GB	1.3 T	HDFS

3.2　基于 K-Means＋＋与 GeoHash 的渲染算法设计与实现

本研究的目标是在提升海量数据的渲染效率的同时，进一步提升海量数据集的渲染效果。为此本研究结合了 K-Means＋＋聚类与 GeoHash 编码的特点，同时引入基于目标集合渲染的策略，对渲染算法进行了设计和优化。该算法的实现环境如表 3 所示，利用 JDK1.8，SpringBoot2.2.1 版本进行开发与实现。

该算法的核心思想是利用 GeoHash 编码技术对 K-Means＋＋的聚类中心的地理位置进行离散化操作，从而提升渲染的效率；利用基于目标集合渲染的策略，提升渲染的效果。首先本研究利用 K-Means＋＋聚类算法对海量目标进行分类并记录每个质心，将每个质心和各个簇中的数据进行统一标识并保存到数据库中，通过记录同簇数据的总数。其次采用 GeoHash 的编码方式将聚类结果转换为 GeoHash 编码，根据 GeoHash 编码将每个聚类中心映射到相应的地理位置上，并在地图或其他可视化界面上显示。通过这个算法，能够直观地展示原始数据的分组情况，有助于发现数据的分布特征和规律。另外，该算法运行在整个系统的服务层和数据层中，能够有效解决目标数据在存储和读取速度方面的问题。最后，由于 GeoHash 的辅助，该算法适用于以经纬度为基准的场景。

该算法的实施过程由 6 个模块和 17 个运算部分组成，具体包括计算环境配置、数据分布模式设计、K-Means＋＋聚类算法的实现、GeoHash 编码的实现以及渲染算法的设计。这些步骤共同构成了基于 K-Means＋＋与 GeoHash 的渲染算法的实施框架，具体计算过程如图 5 所示。

1）计算环境配置：该模块主要涉及导入开发所需的基础组件以及与 GeoHash 和 K-Means＋＋相关的依赖项。这些组件和依赖项将为后续的算法实现提供必要的支持。

2）数据分布模块：该模块首先将海量目标数据分布在多个服务器上，每个服务器负责一定范围的数据。为了实现数据的均匀分布和快速定位，该模块采用了哈希函数将数据按照哈希值进行分配，同时维护了一个全局的哈希

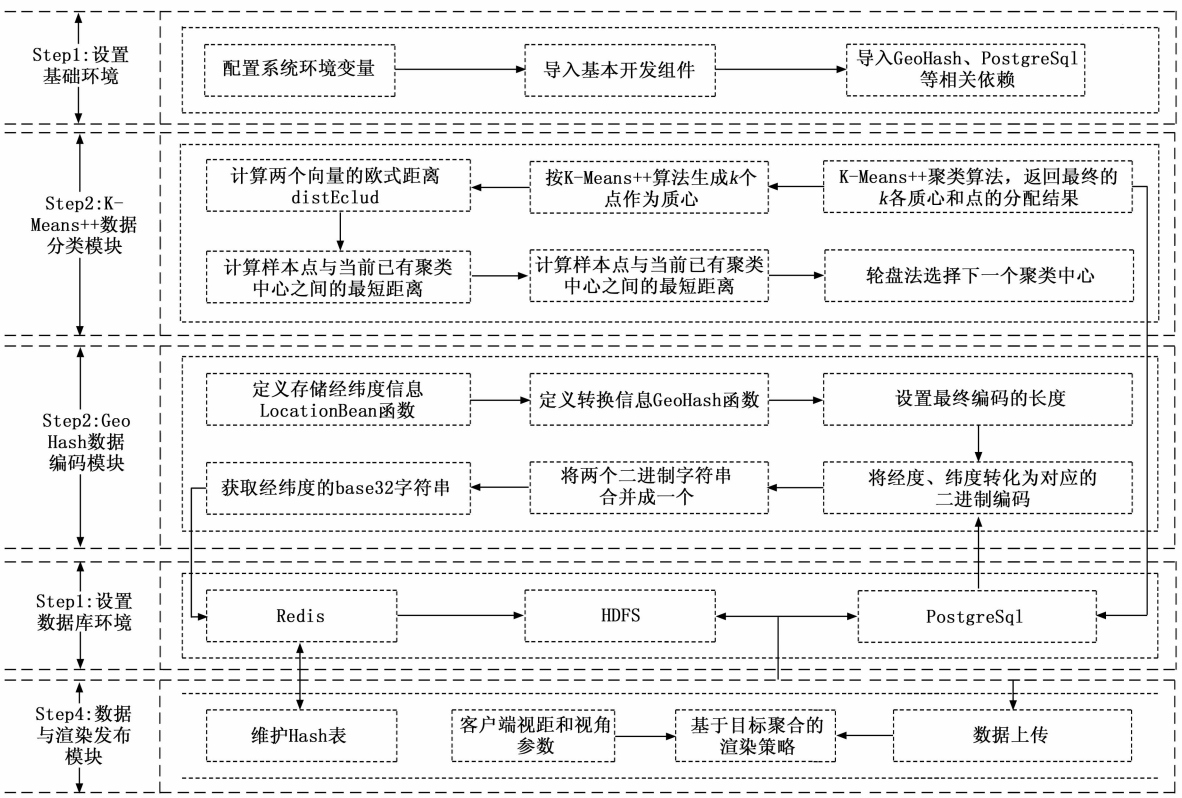

图 5　渲染算法运算流程图

表，记录了每个服务器的地址和负责的数据范围。

3）数据分类模块。该模块对在每个服务器上对负责的数据进行聚类，将数据按照密度和分布进行分组。为了实现数据的高效组织和管理，该算法采用了 K-Means＋＋算法对数据进行聚类，同时维护了一个局部的聚类表，记录了每个聚类的中心和个数。该算法主要包含以下步骤：初始化 k 个质心，通过欧式距离计算将数据点分配到最近的质心，使用轮盘法选择下一个聚类中心，迭代更新质心直到收敛等。通过这一系列的操作，能够对数据进行有效的聚类。

4）数据模块：该模块最后在每个服务器上对负责的数据进行编码，将数据的空间位置信息转化为一维的字符串。为了实现数据的紧凑表示和快速检索，该算法采用了 GeoHash 地理编码技术对数据进行编码，同时维护了一个局部的编码表，记录了每个数据的编码和属性。具体而言，该过程主要包括动态设置编码长度以适应不同精度要求。以二进制编码的方式将地理位置映射到字符串表示，并进行编码合并的操作。通过这样的处理，能够将聚类结果转换为可用于渲染的 GeoHash 编码。

5）数据渲染模块：为了在提升大规模目标数据渲染效率的同时，进一步优化数据渲染的效果。为此，该模块主要采用了基于目标聚合的渲染策略，具体而言是通过客户端传递过来的视距参数和视角参数，动态的对同簇数据进行统计并进行展示，从而实现数据的动态和实时渲染。

4 实验结果与分析

本文提出了一种基于 K-Means＋＋和 GeoHash 的海量目标渲染系统，该系统利用了 K-Means＋＋算法和 GeoHash 地理编码技术，实现了海量目标数据的高效分类和编码，提高了渲染的效率和质量。为了验证该系统的有效性和优越性，本研究在软硬件配置相同和同一网络环境下，通过模拟大规模目标数据集，设计了一系列的实验，对该系统进行了测试和评估。最后，本研究将该系统与传统的基于 K-means 的单节点渲染方法（以下简称传统渲染）进行了对比，分别从效果效果、存储能力、稳定性以及拓展性等方面进行了分析和讨论。

4.1 渲染效果

传统渲染方法首先需要获取质心，再通过质心去计算其他数据。这样的计算过程非常耗时和资源密集，特别是在渲染大型复杂的场景时，渲染时间可能需要数小时，导致工作效率下降和项目延期。而本研究提出的基于 K-Means＋＋和 GeoHash 的海量目标渲染系统，从渲染效果方面，具有以下几个优势：1）渲染效果真实。该系统利用了 GeoHash 地理编码技术，将海量目标数据的空间位置信息转化为一维的字符串，实现了数据的紧凑表示和快速检索。该技术可以保证数据的空间位置信息不会丢失或者变形，从而保证了渲染效果的真实性；2）渲染效果简洁明了。该系统不对所有数据进行逐一渲染，而是利用了目标聚合策略，根据客户端的视角和视距，动态地聚合同簇数据并进行渲染和展示（如图 6 所示）。该策略可以实现数据的自适应和优化，从而保证了渲染效果的立体性和简洁性

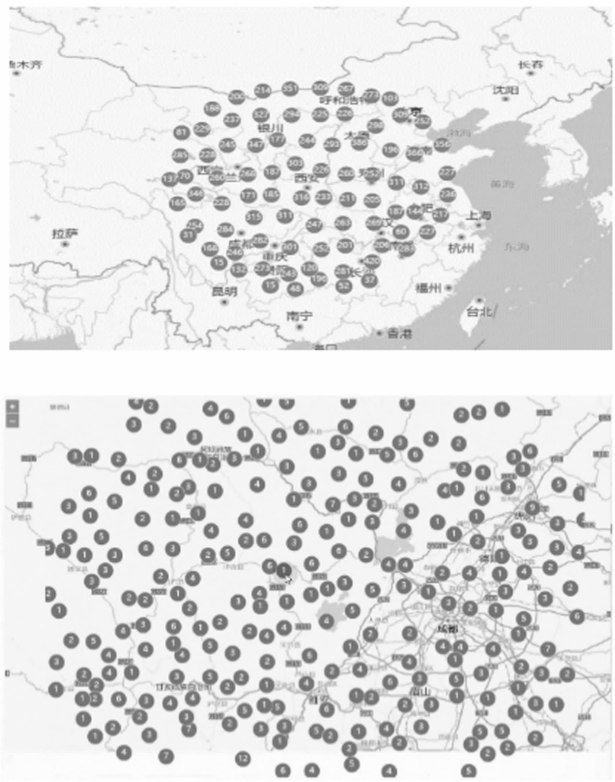

图 6 系统渲染效果图

4.2 渲染速度

该系统对大量目标数据集的渲染时间是衡量其性能的关键指标。传统的渲染法需要将目标数据全部加载到渲染系统中，然后，通过设置参数和渲染参数，确定渲染视角和渲染方式。它虽然能够直观地展示目标数据的属性和空间关系，有助于用户对数据进行分析和理解。然而，由于整体渲染法需要对整个目标数据集进行渲染计算，处理大规模目标数据时可能面临计算复杂度和存储需求高的问题，从而会降低大规模目标数据的渲染效率。与传统的整体渲染法相比，本研究设计的海量目标渲染系统由于采用了分布式架构，能够从根本上改善数据存储和检索的问题；此外该系统采用了基于 K-Means＋＋算法和 GeoHash 编码的渲染模型，能够快速渲染海量目标数据，其速度对比曲线如图 7 所示，据统计，在目标数据量为 400 条时，该系统对目标数据的渲染效率提升了 91.75％；在数据量为 10 000 条时，渲染效率

提升了 95.36%。

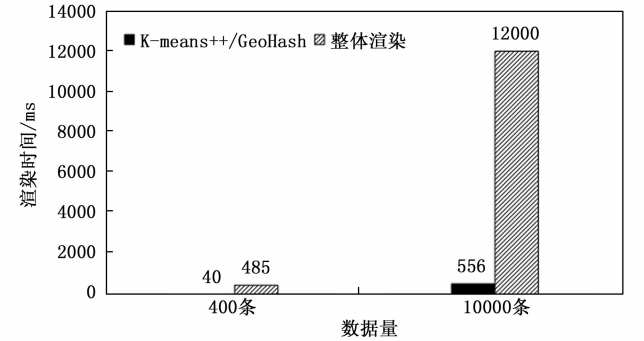

图 7　传统渲染法与该系统渲染速度的比较分析图

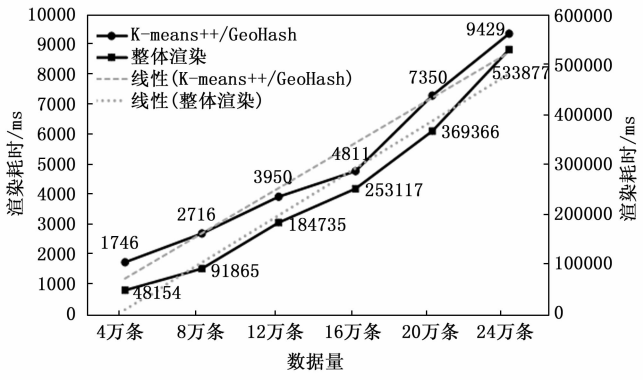

图 8　传统渲染法与该系统渲染速度的比较分析图

为了进一步验证该系统的渲染效率，本研究选择了 4 万条、8 万条、12 万条、16 万条、20 万条和 25 万条目标数据作为数据源，测试了在相同硬件和同一网络带宽条件下，随着数据量的增加，渲染所需时间的变化情况。从图 8 中可以看出，虽然随着数据量的不断增多，不管是该系统还是整体渲染法，对目标数据的渲染耗时都有所增加，但是，基于 K-Means＋＋和 GeoHash 的渲染算法的处理时间要全面低于整体渲染法。当目标数据量为 4 万条时，传统整体渲染法的处理时间约为 48 154 毫秒，而该系统的处理时间仅为 1 746 毫秒，渲染效率提升了 96.37%；当目标数据为 12 万条时，传统渲染法的处理时间为 184 735 毫秒，而该系统的处理时间仅为 3 950 毫秒，渲染效率提升了 97.86%；当目标数据为 24 万条时，传统渲染法的处理时间约为 533 877 毫秒，而该系统的处理时间仅为 9 429 毫秒，渲染效率提升了 98.23%。综上所述，本研究提出的渲染系统可以提高目标数据的渲染效率，此外随着目标数据集的不断扩大，处理效率也在不断提升，因此本系统完全可以试用于海量数据集的情况。

4.3　稳定性

为了探索系统的渲染性能，本研究定义了一个性能指数 I。其中 I 表面渲染一条数据所要耗费的时间（秒），T 表示时间消耗量，D 代表数据条数。

图 9（b）显示了该渲染系统的性能指标曲线。可以看出，性能曲线先下降后上升。当目标数据量较小时，渲染时间相对较短，影响性能曲线的主要因素是检索数据的时间以及节点之间相互通信所花费的时间。但是当数据量线性增加时，影响性能的主要因素转化成数据量本身。但是从图 9（a）中可以看出，当数据量较大时，该系统的渲染性能还是要由于传统渲染的性能。

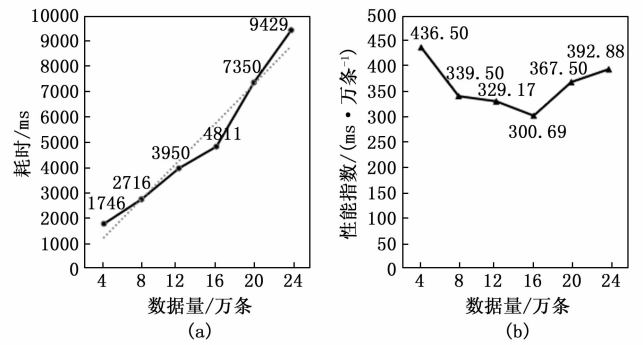

图 9　性能分析图

4.4　拓展性

从图 10 中可以看出，随着目标数据规模的不断扩

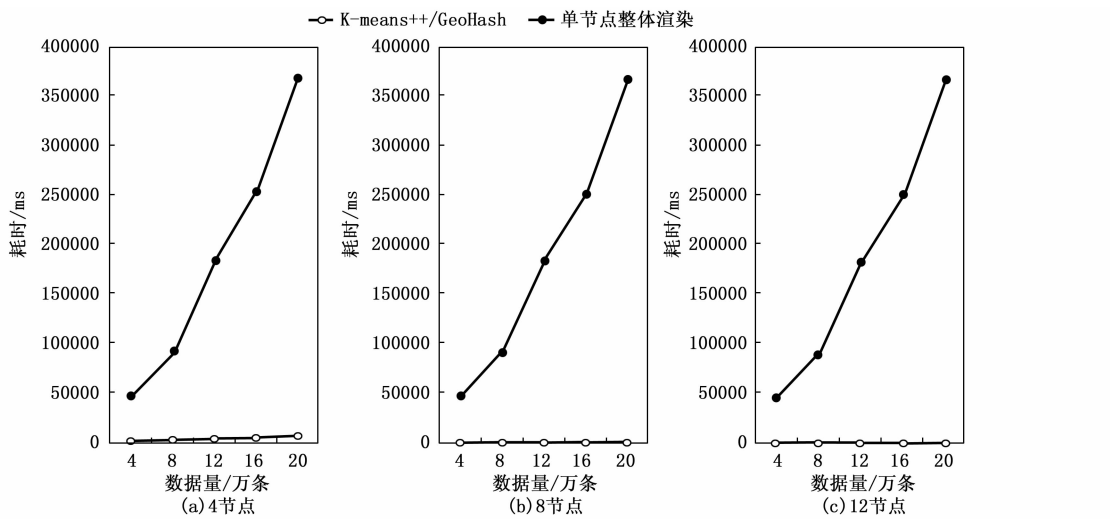

图 10　系统拓展性与稳定性分析图

大、基于 Hadoop 架构的多节点渲染可以加速数据的渲染速度。虽然在数据量较小的情况下，该系统由于多节点处理所需的时间可能低于单节点整体渲染法，这主要是因为该系统采用了 Hadoop 分布式计算，节点之间的相互通信是影响渲染效率的主要元素。但是，随着数据量的不断增加，虽然多节点处理的速度也会降低，但是整体缺低于单节点的整体渲染。此外，当输入数据量增加时，系统的渲染时间都呈似线性变化的趋势，也就表明当数据量大而计算资源不足时，节点数量的增加可以有效提高目标数据的渲染效率，由此可见该系统具备高拓展性。

5 结束语

为了提升大规模目标数据集的渲染效率和效果，并满足基于经纬度场景的实时、动态目标渲染需求，设计了一种基于 K-Means＋＋和 GeoHash 的海量目标渲染系统。虽然该系统从不同层面优化了海量目标数据的渲染效率和效果，但仍存在一些潜在的缺陷和改进空间。首先由于硬件条件的影响，本研究使用的数据集并未真正达到海量的级别，系统面对百万级别的数据规模时的渲染效率还有待考证；其次，本研究所使用的基于目标聚合的渲染策略虽然能够优化数据渲染效果的，但也存在一些过度聚合、数据的细节丢失等问题，需要进一步完善渲染策略，以提高渲染的效果和质量。

参考文献：

[1] 范亚兵，王明海，范亚洲，等. 一种基于 OSG 的海量地形快速渲染方法 [J]. 测绘与空间地理信息，2014，7：190－192.

[2] 张锦华. 基于目标分布场的实时图形跟踪渲染算法 [J]. 微电子学与计算机，2015，32 (7)：95－98.

[3] 牛景波. 自动渲染系统的设计与实现 [J]. 现代电影技术，2013 (9)：9－13.

[4] 孙 前，邱国庆. 基于重要目标区的三维地形快速渲染方法 [J]. 计算机与数字工程，2009，37 (4)：127－129.

[5] 张 芩，王振民. QR-树：一种基于 R-树与四叉树的空间索引结构 [J]. 计算机工程与应用，2004，40 (9)：100－103.

[6] 杨 帆，米 红. 一种基于网格的空间聚类方法在区域划分中的应用 [J]. 测绘科学，2007 (z1)：66－69.

[7] NA S，XUMIN L，YONG G. Research on k-means clustering algorithm：An improved k-means clustering algorithm [C] //2010 Third International Symposium on intelligent information technology and security informatics. Ieee，2010：63－67.

[8] LIU J，LI H，GAO Y，et al. A geohash-based index for spatial data management in distributed memory [C] //2014 22Nd international conference on geoinformatics. IEEE，2014：1－4.

[9] 陈国军，尹 鹏，裴 利，等. 基于 Shader 的 CSG 几何体的实时渲染 [J]. 计算机与数字工程，2022，50 (1)：190－194.

[10] 彭 湃，李少梅. 基于纹理合成的三维地形水墨风格渲染研究 [J]. 测绘与空间地理信息，2017，40 (3)：118－121.

[11] SHIN S，CHOI S. Geometry-based haptic texture modeling and rendering using photometric stereo [C] //2018 IEEE Haptics Symposium (HAPTICS). IEEE，2018：262－269.

[12] CABAN J J，RHEINGANS P. Texture-based transfer functions for direct volume rendering [J]. IEEE Transactions on Visualization and Computer Graphics，2008，14 (6)：1364－1371.

[13] HU H，SONG A. Haptic Texture Rendering of 2D Image Based on Adaptive Fractional Differential Method [J]. Applied Sciences，2022，12 (23)：12346.

[14] DEBEVEC P E，Taylor C J，Malik J. Modeling and rendering architecture from photographs：A hybrid geometry-and image-based approach [M] //Seminal Graphics Papers：Pushing the Boundaries，Volume 2. 2023：465－474.

[15] ARTHUR D，VASSILVITSKII S. K-means＋＋ the advantages of careful seeding [C] //Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. 2007：1027－1035.

[16] AGARWAL M，JAISWAL R，PAL A. k-means＋＋ under Approximation Stability [J]. Theoretical Computer Science，2015，588：37－51.

[17] HUANG K，LI G，WANG J. Rapid retrieval strategy for massive remote sensing metadata based on GeoHash coding [J]. Remote Sensing Letters，2018，9 (11)：1070－1078.

[18] ZHOU C，LU H，ANG Y，et al. GeohashTile：Vector geographic data display method based on geohash [J]. ISPRS International Journal of Geo-Information，2020，9 (7)：418.

[19] YODER J，PRIEBE C E. Semi-supervised k-means＋＋ [J]. Journal of Statistical Computation and Simulation，2017，87 (13)：2597－2608.

[20] KONG W，WANG T，LIU L，et al. A novel design and application of spatial data management platform for natural resources [J]. Journal of Cleaner Production，2023，411：137183.