JUST-Studio: A Platform for Spatio-Temporal Data Map Designing and Application Building

Yuan Sui[1], Ruiyuan Li[2,1*], Xu Wang[1], Jun Liu[1], Juncheng Tang[1]

[1] JD Intelligent Cities Research, Beijing, China
[2] Chongqing University, Chongqing, China
{suiyuan, wangxu649, liujun513, tangjuncheng3}@jd.com; liruiyuan@whu.edu.cn

Abstract. With the proliferation of GPS (Global Position System) and other space sensors, there emerged many spatial-temporal (a.k.a. ST) urban applications, such as trajectory visualization and geofence analysis. It is time-consuming and tedious to build these applications from scratch by writing codes only. This paper introduces JUST-Studio, a holistic platform that analyzes spatio-temporal data and builds urban applications with minor efforts. JUST-Studio consists of two main components: service manager and ST-App designer. By this platform, users could: 1) upload their own ST data to the built-in data store, or connect to an existing ST data store and register it as a data source; 2) create ST models using ST data and publish ST services; 3) make ST base map and build ST applications based on ST services. JUST-Studio is not only capable of making commonly used static map scenes, but also good at rendering dynamic data. In this paper, we will use vehicle trajectories with geofence analysis to build an application for monitoring restricted areas, which is very common in urban traffic applications.

KEYWORDS: Data Virtualization; Web Mapping; Location Analysis; Application Builder

1 Introduction

With the development of spatial mapping and sensors, spatio-temporal data (a.k.a. ST data) has attracted extensive attention and been frequently applied in a variety of industries. For instance, by using status information and trajectories of taxies, a series of drop-off points can be excavated [1][2]. We can recommend these drop-off points to taxi drivers to increase vehicle utilization rate and make more money [3]. However, with the popularization of ST data, people started to realize the limitation of desk map applications. Demands of map sharing and collaboration are becoming stronger. As a result, large-scale web-based cloud GIS (Geographic Information System) system starts to come into the market.

From the web 1.0 era in 1990, web mapping has appeared. With the iteration of Internet technology, from the initial static mapping to the cloud mapping to the recent intelligent mapping, cartography has gone through nine stages of development [4]. Nowadays, many outstanding online GIS platforms give us excellent online map experience. With vector tile technology, people could not only browse the map, but also dynamically set map styles as they like. Mapbox [5] is undoubtedly a good application case, who has a ton of customization features for personalized maps. Mapbox is tile-focused, so its map display is smooth and pleasing to the eyes. However, Mapbox has two shortcomings for terminal users. First, its services only solve the base map presentation problem of GIS. It still needs a lot of coding if users want to interact with the map or have data analysis requirements. Second, the studio provided by Mapbox only supports static data. It does not provide visualization layers for dynamically changing data, e.g., dynamic trajectories. Compared to Mapbox, CARTO [6], equipped with spatial analysis services such as path planning and geocoding, provides a better experience in interactive analysis and map designing. Through CARTO builder, the results of spatial analysis can be fluently displayed on the map. Furthermore, CARTO

* Ruiyuan Li is the corresponding author.

allows users to upload data and store it in its cloud PostGIS database [6]. However, CARTO suffers from the following problems: 1) it can only define basic interactive map operations, such as `mouse click` and `drag`. 2) Like Mapbox, it does not support dynamic data analysis.

To this end, as an important part of our JUST (**J**D **U**rban **S**patio-**T**emporal data engine) project [7][8][9], we build a holistic platform, i.e., JUST-Studio, which can not only provide basic collaborative map operations, but also:

(1) Support the incorporation and analysis of dynamic spatio-temporal data. Dynamic spatio-temporal data could be displayed on the front-end map in real-time. At the same time, further complex spatio-temporal analysis operations can be performed on dynamic data.

(2) Provide analysis capabilities. We provide many out-of-the-box spatio-temporal analysis services, by which users can set their own analysis styles, drag and drop different components to build various fancy map applications without any codes.

To the best of our knowledge, JUST-Studio is the first system to build spatio-temporal urban applications for both static and dynamic scenarios without writing any codes. We also provide an online demo: http://just-studio.urban-computing.com/.

The outline of this paper is as follows. In Section 2, we give the system overview of JUST-Studio. The main components of JUST-Studio are described in detail in Section 3 and Section 4. Finally, we demonstrate the practical effects of JUST-Studio by building a map application in two scenarios in Section 5.

2 System Overview

Figure 1 presents the system overview of JUST-Studio, which consists of two main components: **Service Manager** and **ST-App Designer**. Service Manger includes three layers: 1) data store layer, which stores and manages both static and dynamic ST data, such as geographic data, remote sensing data, trajectory data and mobile phone data; 2) ST-model layer, which offers a variety of ST application models on ST data, such as basic layer model, road network model, GOI (Geometry of Interests) model and trajectory model; 3) service layer, which provides two types of services, i.e., layer services and control services. Based on the services provided by Service Manager, users can construct various fancy urban applications with ST-App Designer in a pulling-dragging manner.

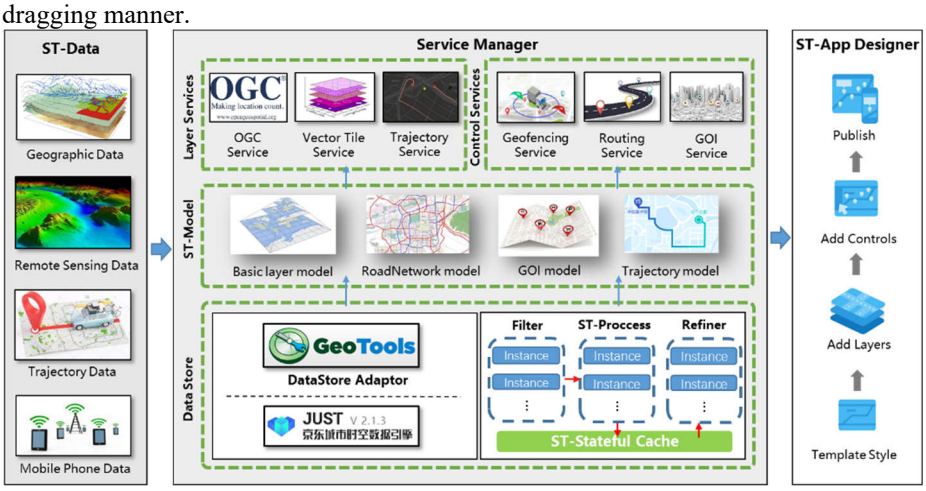

Figure 1: System Framework

3 Service Manager

Service Manager provides a wealth of services for ST-App Designer to call. It acts as a solid foundation for map application construction. For the convenience of users, Service Manager simplifies the whole service creation process into three steps: 1) Data Source Register, 2) Model Creating, and 3) Model Publishing. They are used to correlate data, define algorithmic models, and publish algorithmic capabilities, respectively.

3.1 Data Source Register

Data source configuration is the first step to use ST data. Classified by the means of data access, there are two modes of configuration. First, users upload local files in ShpFile or GeoPackage formats; second, users register existing data sources, which can be both static and dynamic. Static data source refers to the data that is not accessed and updated in real time. It includes traditional relational databases, such as PostgreSQL, and distributed NoSQL databases, such as HBase [10]. Dynamic data sources refer to data sources that are accessed in a real-time fashion, such as Kafka [11], a message-oriented middleware.

Data Source Register has a strict inspection process. It includes: 1) *data connectivity testing*, which examines whether the target data source is connectable or whether the uploaded data is complete; 2) *spatial property checking*, which inspects whether the target data source contains correct spatial fields, such as longitude, latitude, or WKT (Well-Know Text) [12] fields; 3) *special data spatial check*, which is related to specific models. For example, the road network model needs to check the connectivity of road networks, because an unconnected road network will lead to wrong path planning results. Users will be ready to use these data sources after the inspection process. Next, we will briefly explain the static and dynamic data supported by JUST-Studio.

Static data. In order to access multiple heterogeneous data sources, we choose JUST [7] to store static data. JUST provides a unified JustQL language for multiple data sources, which makes the management and analysis of massive ST data easier. Users can switch different execution engines for both spatio-temporal query scenarios and spatio-temporal mining cases. When users only conduct applications and analysis in small or medium-sized data scenarios, they can use local engine mode, which can not only meet the requirements of computing efficiency, but also does not occupy too many resources. When it comes to city-level massive data scenarios, they can switch to the distributed Spark [13] computing engine. This scalable mode can cover almost all ST analysis and computing scenarios. JUST-Studio extended its OGC (Open Geospatial Consortium) [14] adapter to achieve a unified layer management mode based on GeoTools framework [15].

Dynamic data. In order to better access and process real-time data, we design a real-time data management framework, as shown in the right part of Data Store in Figure 1. The framework consists of two components. One is the upper ST processing engine, and the other is the underlying ST state storage. The processing engine is responsible for accessing real-time ST data. Currently, JUST-Studio supports access to distributed Kafka messages. After access, the engine filters out unwanted data in real time according to parameters set by users. Remained data will flow into the processor, where ST data will be transformed and assembled according to predefined logic. The results will be stored into stateful storage. Stateful storage is a cache component of ST data that receives information from the processor in real time and updates the old information. Cache data will be purged periodically at a preset time. When a user accesses real-time data, the request would be sent to Refiner. Refiner extracts the latest data from the stateful storage, refines the data based on the user's request parameters, and returns the final result to the caller.

Real-time data management framework is a predetermined flow framework. Different data and different processing logic will get different results. We will show you in Section 3.2 and Section 3.3 how to implement specific real-time data services through dynamic model definitions.

3.2 Data Model Creating

Data models are the bridges between data sources and spatial-temporal services. A data model must be associated with at least one data source. Parameters are then set according to different model types, which together with the data determine the final service result. We can think of the model as a kind of ST algorithm. For example, if we want to build a path planning application, we first need to create a road network model. When we associate a road network data source, we need to set the parameters of the model, such as geometry field, direction field, speed limit field. After that, we have a

path planning algorithm model based on the specific road network data.

ST-Static Model. There are three static temporal and spatial models in JUST-Studio: layer model, road network model and GOI model. Layer model mainly orients to map presentation scenarios. Relatively, the configurations of road network models and GOI models are more abundant. The configuration of road network models has been briefly introduced before. Users need to map some data source fields to the parameters required by our path planning algorithm. However, field mapping is not mandatory. Users can create models without any configuration. In that way, JUST-Studio would use default parameters to provide path planning services. The GOI model is used to serve keyword retrieval for general geometry objects. Users must specify at least one keyword field.

ST-Dynamic Model. Dynamic ST data refers to the data whose properties change over time. For instance, trajectory data, whose GPS point positions are time-varying. Dynamic ST model is an algorithm package of ST dynamic data. Before creating the trajectory model, the user needs to associate a Kafka data source that receives GPS points. JUST-Studio would partition Kafka's incoming data based on the user-defined trajectory key, and then allocates a thread to each partition processor. With each thread, filtering and processing operations would be conducted. As shown in Figure 2, taking the vehicle trajectories as an example, using license plate number as the key, GPS points are assigned to different Kafka cluster partitions. Consuming thread pool in downstream system would subscribe GPS points of all partitions and perform filtering and processing operations in the form of Pipeline. This strategy of partitioned processing can greatly improve the throughput of real-time processing. It is important for time-sensitive scenarios, such as real-time visualizations. Users can configure the way of trajectory processing, such as setting the map matching [16] processor. If the accessed Kafka data is the original GPS points, notable errors may occur. Setting map-matching can rectify GPS points to the correct position in real-time. Since map-matching requires a benchmark road, a road network model must be associated with it. In addition to the processing methods, the system also provides indexed configurations. Users can choose the latest point of trajectory data or the trajectory line as the index, as shown in Figure 3. When the point index is used, JUST-Studio would create a `gps_point` column for the trajectory table, which stores the last GPS point for each vehicle. When querying the trajectories, JUST-Studio would filter GPS points by the nearest point of the vehicle. When users choose the line index, it will establish a `traj_line` column to store the latest trajectory line object, as shown in Figure 4. In this case, the trajectory will be filtered by the whole vehicle line. These two storage methods both have their own application scenarios. In the next section we will give a more detailed description.

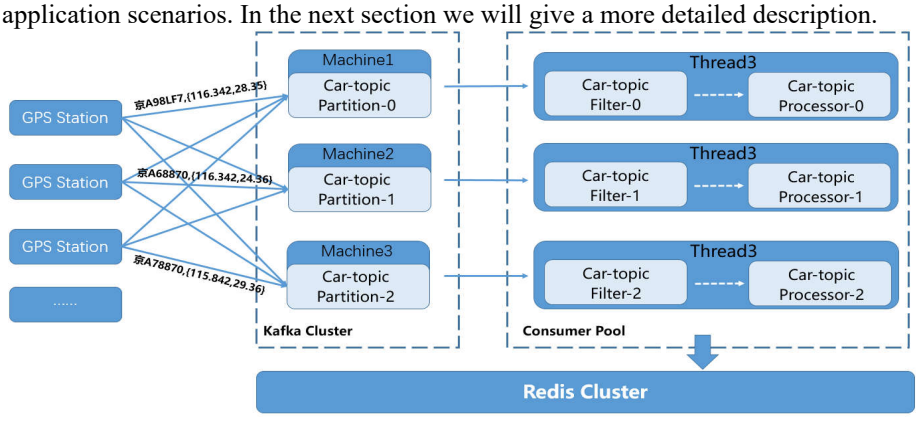

Figure 2: ST-Dynamic Data Process Architecture

Key	Value	
gps_point	traj_series	properties
Latest $GPSPoint_1$	LinkedList<GPS Point>	$number_1$ $speed_1$ ...
Latest $GPSPoint_2$	LinkedList<GPS Point>	$number_2$ $speed_2$ ...
...	...	...
Latest $GPSPoint_n$	LinkedList<GPS Point>	$number_n$ $speed_n$ ...

Figure 3: Point indexed trajectory

Key	Value	
traj_line	traj_series	properties
Latest $LineString_1$	LinkedList<GPS Point>	$number_1$ $speed_1$ ...
Latest $LineString_2$	LinkedList<GPS Point>	$number_2$ $speed_2$ ...
...	...	...
Latest $LineString_3$	LinkedList<GPS Point>	$number_n$ $speed_n$ ...

Figure 4: Line indexed trajectory

3.3 Service Publishing

Services are outlets for model capabilities and bridges between data and visual interfaces when building ST applications. Classified by ST application scenarios, JUST-Studio provides two categories of services. One is the ST map configuration service, called Layer Service. The other is the ST application configuration service, also known as Control Service. This kind of service capabilities are displayed in JUST-Studio through ST analysis components.

Layer Service. Static ST data supports two types of layer services. One is the standard OGC service, including WMS (Web Map Service) [17], WFS (Web Feature Service) [18] and WMTS (Web Map Tile Service) [19]. The other is a vector tile service. We extend the algorithm based on the vector tile standard structure [20] proposed by Mapbox. We add a pixel-based spatial thinning algorithm to balance the integrity of data expression and display efficiency. In addition, dynamic ST data can also publish layer services. For example, the trajectory service can provide real-time trajectory visualization. When visiting from the front end, to obtain the trajectories of all vehicles within a spatial range at a specific moment, users only need to dynamically input the current map space range. Note that the trajectory visualization displays the latest location information of the vehicle, so the trajectory model depends on the needs to be built with the point index.

Control Service. Unlike Layer Service, Control Service concentrates on ST analysis scenarios. They are the service basis for ST-App Designer (simplify Designer later). In Designer, one or more control services are packaged into an analysis component, allowing users to build applications easily. Here we focus on the control service of dynamic ST data, i.e., geofencing service. A geofence is a virtual perimeter for a real-world geographic area [21]. The use of geofence is called geofencing. There are many applications involved in geofencing. When used to children's location service, geofence can notify parents if a child leaves a designated area [22]. Another case is the vehicle tracking. When a car drives into a user- predefined perimeter, it would send some warning messages immediately. With JUST-Studio, users can add or remove geofence geometry objects for the geofencing service, and then associate a trajectory model with it. Once geofencing service is online, the system would monitor the positions of vehicles in real-time. If a vehicle enters the fence, one record will be added to the monitoring table. The system will continuously monitor the entered vehicles until they leave the fence. Eventually the information such as entering time, leaving time, driving routes would be recorded in the monitoring table. Given that fence monitoring has a short time difference, the geofencing needs to capture the trajectory routes of all vehicles. Therefore, the trajectory model associated with the geofencing service must be stored as a line index.

3.4 Massive Data Optimization

With the support of JUST, JUST-Studio could access huge amounts of ST data. However, on the other hand, it poses a big challenge to the Studio's processing power. Taking map service as an example, the visualization of 1,000 geometries and 100,000 geometries has completely different requirements for the processing efficiency of the server side. Users usually do not care about the amount of data in the database, and they just want to see the results on the map in seconds. JUST-Studio has made many optimizations for this. Say the user wants to display 100,000 geographical elements on a map. We first partition the geographic spatial area of the user request based on the slicing strategy of map tiles, and then retrieve the data in the database in parallel using the divided sub-areas (data thinning parameters can be set here). After obtaining the data of each region, we simplify the data using Douglas-Peucker algorithm [23], of which the simplification coefficient is calculated according to the pixel value of each region. After that, we transform the geographic coordinate system into a local coordinate system. Finally, we compress the data using zigzag compression method [24]. The procedure is shown in Figure 5. By this algorithm, JUST-Studio can easily render millions of geographical elements in the front end in seconds. If we switch our execution engine to Spark mode, the amount of data we can handle will be even larger.

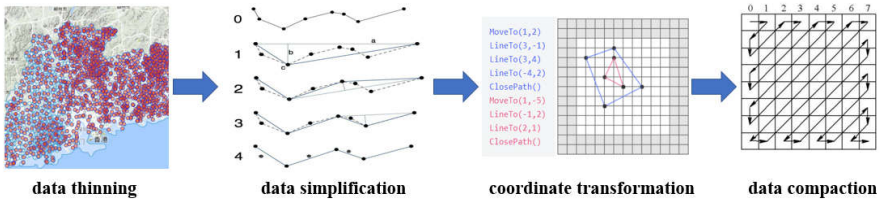

Figure 5: Massive ST Data Map Service Optimization

4 ST-App Designer

ST-App Designer is a configuration module of JUST-Studio in the front end. There are mainly two functions. One is to configure map styles, and the other is to construct ST applications based on the configured map styles. Users can publish the applications. Other users can directly access the designed ST applications, or use the front-end SDK provided by Designer to parse the map and process a secondary development.

4.1 Map Designing

ST-App Designer is built based on Mapbox GL JS framework. We choose Mapbox because of its high rendering efficiency for vector tile services. As mentioned earlier, Service Manager provides real-time update vector tile services. ST-APP Designer can synchronously read all vector tile services published by the user. Based on the layer services released by Service Manager, the system supports various types of rendering effects. 1) *Point style rendering*. For the point data, ST-APP Designer can set the point icon style and label information. 2) *Line style rendering*. It provides line type, line width, color and other style settings; 3) *Surface style rendering*. It supports the texture, color, boundary style and other configurations. 4) 2.5*D surface rendering*. It provides a three-dimensional stretching effect for surface data. 5) *Trajectory rendering*. It supports real-time display of trajectory data and can change the styles of trajectory line and trajectory icon.

4.2 Application Designing

In addition to the configuration of the map style, Designer also provides users with an environment for constructing spatial-temporal application scenarios. We define an ST application as the combination of a map and some spatial-temporal controls. The user selects one of several designed map styles as the base map of an ST application, and then adds the required controls on top of this. JUST-Studio provides three categories of controls. The first is heat map control, which allows user to build a map with high and low render effects. The second is analysis control, which is the most important element of scenario construction. At present, the system provides POI query, path planning, geographical fence monitoring and other capabilities. All of these controls allow users to drag and drop to change positions and sizes. The analysis capabilities of the controls come from the control services published by Service Manager. Users only need to associate interface controls with service models and configure parameters to add corresponding functions to the applications. The third is general control, such as map scale, north finger, layer control and other common elements in the map.

4.3 Sharing

Both map styles and scenario styles can be shared with other users. There are two modes of sharing: 1) Webpage sharing, which is the fastest sharing method. Other users can access ST applications directly when they get the sharing link. A common application is to nest iframes into their own web pages. 2) Style file sharing, which is a more flexible sharing method. After users get the link, they can use JUST-Map-SDK (simplify SDK later) to restore the application quickly. At the same time, they can use the interface provided by SDK for a secondary development.

5 Demonstration

As shown in Figure 6, JUST-Studio initial interface is a dashboard structure with the configured map templates provided by the system at the top. By clicking the cards in the area of **System Default Templates**, users can quickly create a map style based on a template and enter the map design interface. Users can also create maps in the **Map Style Dashboard** area, which also supports the import of existing native styles. As shown in the figure, the designer allows users to create a new scenario application. Users click **New Scene Style** or select a basic map style to enter the scene editing area. Below, we will show how a designer works through a geofence monitoring scenario.

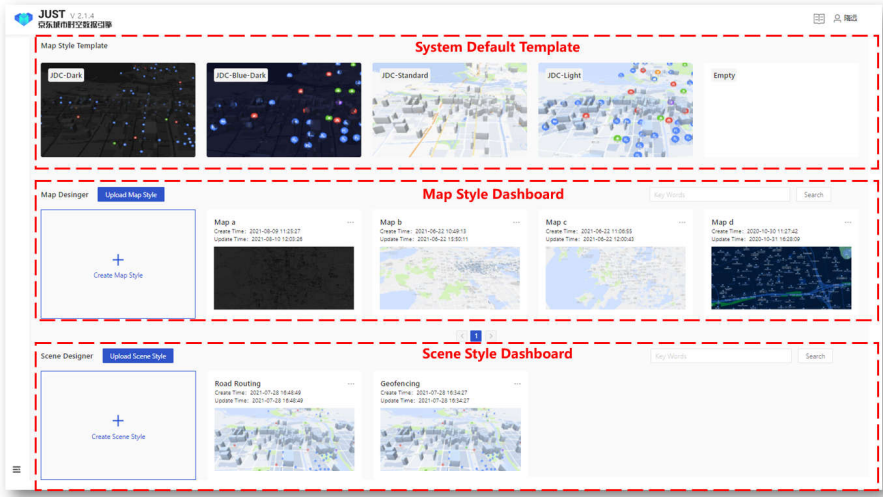

Figure 6： Main Page of ST-AP Designer

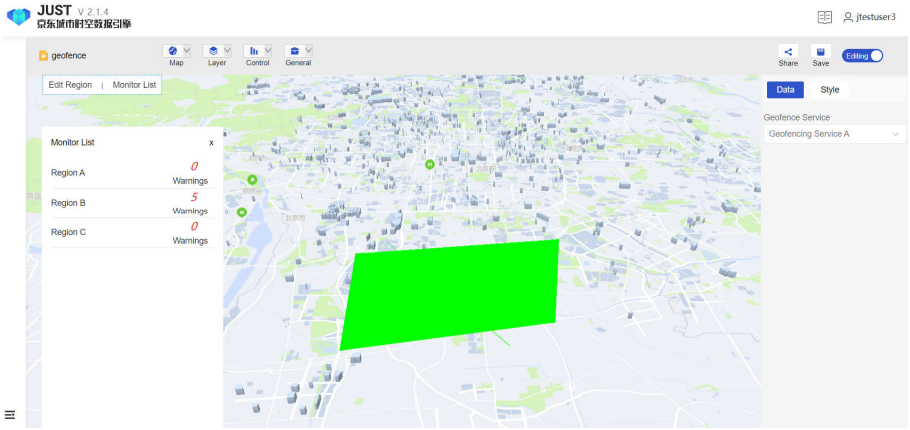

Figure 7：Scene Designer Workbench for Geofencing

Geofence Monition Scene. Geofencing monitors vehicle information, so we need to receive vehicle trajectory data first. We configure a Kafka data source to receive real-time vehicle data by Service Manager. The Kafka data source needs to configure the topic of the message, as well as the longitude, latitude and direction fields. We then create a dynamic trajectory model associated with the previous Kafka data source, as shown in Figure 7. As mentioned in Section 3.3, in order to monitor the vehicle's path within the fence, we use the line index to store trajectory data. In this demo, the vehicle GPS position Kafka got is the original position, which may have large errors. Therefore, we need to open the map matching option and associate it with a road network model. Then, we need to create a geofencing model, which is configured by associating a trajectory model simply. Finally, by launching the geofencing model, we have all services associated with geofencing applications.

After completing the geofencing service configuration, we need to build the application in Designer. First, we select a map template and enter a workbench. Next, we add another geofencing control, resize and style it, associate a geofencing model in the parameters box, and then publish the application. By doing so, we complete the

construction of a geofencing application. Now we have a trajectory, but where is the fence? Note that JUST-Studio creates a geofencing application, so the fence must be created by the users using the application. Various areas of interest can be created through our fence tool. Once saved, these areas begin to monitor the vehicles. The warnings of entering the fence will be displayed in real time in the results bar.

Acknowledgments

This work is supported by National Key R&D Program of China (2019YFB2103201) and National Natural Science Foundation of China (61976168).

Reference

[1] Hu Y, Ruan S, Ni Y, et al. SALON: A Universal Stay Point-Based Location Analysis Platform[C]//ACM SIGSPATIAL, 2021.
[2] Ruan S, Li R, Bao J, et al. Cloudtp: A cloud-based flexible trajectory preprocessing framework[C]//ICDE. IEEE, 2018: 1601-1604.
[3] Yuan J, Zheng Y, Zhang L, et al. Where to find my next passenger[C]//Proceedings of the 13th international conference on Ubiquitous computing. 2011: 109-118.
[4] Veenendaal B, Brovelli M A, Li S. Review of web mapping: Eras, trends and directions[J]. ISPRS International Journal of Geo-Information, 2017, 6(10): 317.
[5] Mapbox Studio. https://studio.mapbox.com/, 2021.
[6] Carto. https://carto.com/, 2021
[7] Li R, He H, Wang R, et al. Just: Jd urban spatio-temporal data engine[C]//ICDE. IEEE, 2020: 1558-1569.
[8] Li R, He H, Wang R, et al. Trajmesa: A distributed nosql storage engine for big trajectory data[C]//ICDE. IEEE, 2020: 2002-2005.
[9] Li R, He H, Wang R, et al. TrajMesa: A Distributed NoSQL-Based Trajectory Data Management System[J]. TKDE, 2021.
[10] Vora M N. Hadoop-HBase for large-scale data[C]//ICCSNT. IEEE, 2011, 1: 601-605.
[11] Thein K M M. Apache kafka: Next generation distributed messaging system[J]. International Journal of Scientific Engineering and Technology Research, 2014, 3(47): 9478-9483.
[12] Well-known text. https://en.wikipedia.org/wiki/Well-known_text_representation_of_geometry, 2021.
[13] Zaharia M, Chowdhury M, Das T, et al. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing[C]//NSDI 12. 2012: 15-28.
[14] OGC. https://en.wikipedia.org/wiki/Open_Geospatial_Consortium, 2021.
[15] GeoTools Project. https://www.geotools.org/, 2021.
[16] Newson P, Krumm J. Hidden Markov map matching through noise and sparseness[C]//Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems. 2009: 336-343.
[17] WMS. https://en.wikipedia.org/wiki/Web_Map_Tile_Service, 2021.
[18] WFS. https://en.wikipedia.org/wiki/Web_Feature_Service, 2021.
[19] WMTS. https://en.wikipedia.org/wiki/Web_Map_Tile_Service, 2021.
[20] Mapbox Vector Tiles. Accessed: 2015-03-12. url: https://www.mapbox.com/developers/vector-tiles/.
[21] Abbas A H, Habelalmateen M I, Jurdi S, et al. GPS based location monitoring system with geo-fencing capabilities[C]//AIP Conference Proceedings. AIP Publishing LLC, 2019, 2173(1): 020014.
[22] LaMarca A, De Lara E. Location systems: An introduction to the technology behind location awareness[J]. Synthesis Lectures on Mobile and Pervasive Computing, 2008, 3(1): 1-122.
[23] Hershberger J E, Snoeyink J. Speeding up the Douglas-Peucker line-simplification algorithm[M]. Vancouver, BC: University of British Columbia, Department of Computer Science, 1992.
[24] Protocol Buffers Encoding. https://developers.google.com/protocol-buffers/docs/encoding, 2021.