Journal of Network Intelligence ©2016 ISSN 2414-8105 (Online)
Taiwan Ubiquitous Information Volume **8**, Number **2**, May **2023**

Trajectory Privacy-preserving Method Combined with Prediction Perturbation in LBS

Hui Wang

School of Software
Henan Polytechnic University
Jiaozuo 454000, China
wanghui_jsj@hpu.edu.cn

Lei Song

School of Computer Science and Technology
Henan Polytechnic University
Jiaozuo 454000, China
songlei20210903@163.com

Zi-Hao Shen*

School of Computer Science and Technology
Henan Polytechnic University
Jiaozuo 454000, China
szh@hpu.edu.cn

Pei-Qian Liu

School of Software
Henan Polytechnic University
Jiaozuo 454000, China
liupeiqian@hpu.edu.cn

Kun Liu

School of Software
Henan Polytechnic University
Jiaozuo 454000, China
liukun@hpu.edu.cn

*Corresponding author: Zi-Hao Shen

Received November 14, 2022, revised January 3, 2022, accepted March 19, 2023.

Abstract. *Location-based service are widely used in the continuous query, making the protection of users' location and trajectory privacy a hot topic. Differential privacy, the mainstream privacy protection method, is widely used in location-based service, but there are many problems with it: the privacy level of location points is improperly assigned, the additional noise is independent, and it does not meet the user's demand in terms of trajectory availability. For this reason, we propose a trajectory privacy-preserving method combined with prediction perturbation in this paper. First, according to the topological relationship between geospatial points, a chain hash table is constructed to calculate privacy levels on the map; Secondly, to prevent attackers from noise filtering against independent noise, a noise sequence satisfying sequence indistinguishability is generated by combining the autocorrelation information of the user's trajectories. At the same time, the noise radius is reduced to avoid the noise location is far from the real location; Finally, a dummy location selection algorithm is proposed in combination with a Markov prediction model. It uses the genetic algorithm to filter the dummy location set, and utilizes the detection function to check the availability of the filtered perturbed site, to judge whether to use the location instead of the differentially private perturbed location. In addition, the number of user interactions with LBS is reduced by caching the query results of the fake location set. Through security analysis and experimental validation, this paper further demonstrates the effectiveness of the proposed method on privacy protection effect and service quality.*
Keywords: Trajectory privacy; sequence indistinguishability; Markov prediction model; Detection function; Genetic algorithm

1. **Introduction.** With the rapid development of communication technology, Location-Based Service (LBS) is widely used in various fields, such as healthcare and mobile social networking [1]. LBS is based on location information, with the support of Geographic Information System (GIS) and lightweight mobile devices, to provide users with value-added services, including point of interest queries [2]. Usually, when we enjoy the services brought to us by LBS, the cloud and edge servers can easily obtain the sensitive data involved in the physical and social systems, such as identity, location, and assets [3]. According to an investigation in April 2021, the personal data of approximately 533 million Facebook users was stolen by hackers. This included users' phone numbers, Facebook login IDs, names, home addresses, and emails. The compromised personal information involved 533 million Facebook users from 106 countries. The Irish Data Protection Commission fined Facebook 265 million euros as a result. Prior to this, Facebook had experienced several privacy data breaches. In September 2019, a database storing more than 400 million phone numbers associated with Facebook accounts was exposed. In December 2019, an online database containing more than 267 million Facebook user IDs, names, and phone numbers was made public. Security and privacy issues are the most concerns in various network applications and environments [4]. Therefore, more and more attention has been paid to the privacy protection of location-based service [5].

Currently, location-based service are classified into two types: snapshot query service and continuous query service. In the snapshot-based query service, the general user only initiates a query request based on the current location information, and the user only needs to protect the location's privacy at a single moment. In continuous query service, users need to continuously initiate service requests to LBS servers and obtain corresponding query results according to their location information. Users should not only protect the location privacy and security at each moment but also consider the privacy of users' trajectories not to be leaked [6]. There are three main privacy protection techniques for the user's trajectory privacy leakage problem: k-anonymity [7, 8, 9], fake data method [10, 11] and suppression method [12]. Among them, k-anonymity mainly generalizes the user's actual location into an anonymous zone containing k users, and replaces the user's real location query with the anonymous zone. However, there are problems of significant computational overhead: less accurate query results, and difficulty constructing the anonymous zone when querying continuously. The fake data method uses a fake location or trajectory instead of an actual location or trajectory for the query. The key lies in enhancing the usability of the fake location or trajectory while ensuring privacy and security. The suppression method mainly suppresses the release of sensitive location points in trajectories, but some situations can easily lead to severe distortion of trajectory data.

According to the above analysis, to ensure the usability of the user's overall published trajectory while improving the privacy and security of the user's trajectory, this paper proposes a trajectory differential privacy protection method combined with prediction perturbation. We combine differential privacy technology with prediction perturbation, consider the user's personalized privacy needs and the insufficient degree of privacy protection caused by independent noise, reasonably select the perturbed location to be published, and use the detection function to judge the availability of the perturbed location. We then cache the predicted location query results, and finally achieve the goal of improving the overall security and usability of the trajectory. The main contributions of this paper are as follows:

(1) A privacy level assignment (PLA) algorithm is proposed. The transfer probability between locations is added to the privacy level allocation to describe the correlation between locations. We construct a chained hash table to store all the privacy levels of non-initial sensitive location points computed by a initial sensitive location points and consider all initial sensitive sites connected with a non-initial sensitive location points.

(2) A noise sequence preprocessing (NSP) algorithm is proposed and applied to the trajectory privacy protection based on location service. The user's noise sequence is preprocessed to make it indistinguishable from the user's trajectory sequence, thereby obtaining noise locations that satisfy spatiotemporal correlations.

(3) We combine the Markov prediction model with the genetic algorithm and propose a false position selection (FPS) algorithm. To ensure higher service similarity between the filtered perturbed position and the actual position, we apply a genetic algorithm to filter the wrong places while making the perturbed position uncertain, then use the detection function to check their availability and cache the query results of the wrong position set to reduce the number of user interactions with LBS.

(4) Optimize the service quality at noise location points. We ensure that the noise location meets the user's service quality requirements by reducing the noise radius.

2. **Related work.**

To solve the problem of user trajectory data leakage, researchers have conducted several studies and investigations. Wu et al. [13] combined the privacy level with the differential privacy [14] budget, based on the Markov probability transition matrix, proposed a differential privacy location publishing mechanism DPLRM to protect the user's location and trajectory privacy. Ye et al. [15] divided the privacy level of spatial location points in the privacy protection scenario of continuous query, also used differential privacy technology. But they all have the problem of unreasonable allocation of privacy levels. Min et al. [16] proposed a reinforcement learning-based privacy-preserving scheme for sensitive semantic locations, which used differential privacy for random perturbation of vehicle locations, and adaptively selected a perturbation strategy based on the sensitivity of semantic sites and attack history. Gao et al. [17] extracted the stay points as the trajectory's characteristics, then confused each stay point into the target confusion sub-region through the exponential mechanism, and finally performed Laplace sampling in the target confusion sub-region to obtain the confused GPS points based on the sliding window algorithm. To realize trajectory privacy protection, Xu et al. [18] first partitioned the original trajectory sequence into different segments, and then selected appropriate positions and segments to form the confused trajectory sequence, so that the original trajectory sequence becomes a trajectory sequence satisfying differential privacy. Moreover, Laplace noise and exponential noise are added to the output in the position confusion matrix generation and trajectory sequence function generation stages, respectively. Al-Dhubhani and Cazalas [19] proposed an adaptive location protection privacy mechanism by exploiting the impact of correlation of user's obfuscated location on location privacy level, which mechanism adjusted the amount of noise required to obfuscate a user's location based on the level of correlation with its previously obscured location. Yin et al. [20] used location trajectories and check-in frequencies to set thresholds and thus classified the location sensitivity levels, then allocated the corresponding privacy budget according to the sensitivity, and added the Laplace noise that meets different privacy. But the noise they add is independent noise, this can lead to privacy breach issues.

To address the balance between privacy budget and quality of service, Zhang et al. [21] proposed a semantic and prediction-based differential privacy protection scheme for trajectory data. The scheme transformed the trajectory data into a prefix tree structure to ensure that they satisfy differential privacy, used semantic sensitivity combined with location check-in frequency to calculate the sensitivity of each location in the trajectory, and assigned the corresponding privacy budget according to the location privacy level. For protecting trajectory data and adapting to its dynamic nature, Chen et al. [22] proposed a new RNN-based trajectory privacy protection scheme, which used RNN to predict and process users' trajectory behaviors and real-time data, and employed differential privacy techniques with a prediction mechanism to improve the availability of data. Yao et al. [23] applied density-based clustering to identify hotspots and outliers, blurs the locations by generalization, and proposed a graph-based model to efficiently capture the relationship between sensitive labels and trajectory points in records, used Laplace noise to achieve differential privacy, and finally generated and publishes trajectories by traversing and updating this graph. Cheng et al. [24] proposed an optimal differential privacy mechanism for personalized trajectories, which filtered template trajectories by semantic similarity, assigned a privacy budget to the remaining trajectory location points with semantic similarity and template trajectory privacy level, and added noise. But they didn't consider the correlation between locations. According to the spatio-temporal characteristics of trajectories, Yuan et al. [25] proposed a trajectory similarity tree structure based on the R-tree index structure to realize spatial storage and query processing of trajectory data. They constructed a DPTS tree (Differential Privacy Trajectory Similarity tree, DPTS-tree) by using differential privacy technology, and added noise to the nodes' statistical values of mobile users. At the same time, random noise is added to the position data and other data in the trajectory when constructing the DPTS-tree. Ma et al. [26] stored the perturbed locations after differential privacy noise addition by introducing R-trees, used a testing mechanism to determine whether the perturbed locations are available, achieved reusability of the perturbed locations and reduced the consumption of privacy budget in continuous location privacy protection. Li et al. [27] first divided the privacy level of spatial location points according to the road network topology relationship, then allocated the privacy budget of sensitive road sections, and added Laplace noise through differential privacy technique to achieve privacy protection of location data.

In addition, since the traditional k-anonymous trajectory data release technology cannot effectively protect user privacy from attackers with strong background knowledge, Chen et al. [28] proposed a differential privacy based (k-psi)-anonymity method to defend against re-identification and probabilistic inference attacks. This method first gave a dummy-based (k-psi)-anonymous trajectory data release algorithm, which considered the variation of thresholds on different road sections, and constructed an adaptive threshold set psi considering road network information to improve (k-delta)-anonymity. Laplace noise is then added to the output anonymized trajectory dataset. Wen et al. [29] proposed a secure perturbation region generation algorithm to dynamically calculate perturbation range for each timestamp, established an optimization problem with real-time quality loss as the optimization objective and location DP and security perturbation region as the optimization conditions, and achieved the optimal DP mechanism by solving the optimization problem. To achieved w-event differential privacy for crowd-sensing participants, Niu et al. [30] proposed a real-time mobile crowd-sensing(MCS) data collection mechanism-RDCTP, which combined with a w-event privacy model, assigned privacy budgets to trajectory locations and added noise, while formulating an optimization model to select locations from the candidate perturbation location set that satisfy the privacy-preserving effect and service quality. Based on differential privacy, Chen et al. [31] proposed a reinforcement learning-based differential privacy optimization scheme for vehicle ad hoc networks. This scheme can dynamically optimize the privacy budget allocation for each location on the vehicle trajectory, achieve a better balance between geolocation obfuscation and semantic security, and reduce the risk of geolocation and semantic location leakage. Combined with the autocorrelation constraint, Hu and Yang [32] considered the cross-correlation constraint for trajectories, superimposed the real trajectory sequences on the user noisy series satisfying the autocorrelation constraint, and established the published trajectory sequences that satisfy the cross-correlation constraint. Although the above study can protect the user's location or trajectory privacy to a certain extent, the following problems still exist:

(1) The privacy level allocation is unreasonable. In allocating privacy levels to different location points in the map, previous researches only consider the distance and connectivity between location points without considering the correlation between locations. Often, when calculating the privacy level of a location point, only a initial sensitive location point is used as the standard for privacy level allocation, which leads to unreasonable privacy level allocation and reduces the efficiency of privacy budget allocation.

(2) The degree of privacy protection is insufficient. When applying differential privacy to trajectory privacy protection, the added noise is not considered an independent homogeneous distribution variable. An attacker with background information such as autocorrelation information of user trajectory sequences can use attacks such as noise filtering to filter out interference noise, which leads to user privacy leakage.

(3) Low data availability. When employing differential privacy techniques to add noise to real locations and publish them, previous studies seldom consider the service quality in noisy areas, resulting in the proposed privacy protection methods not achieving a certain balance between user privacy security and service quality.

To address the above problems, this paper proposes a trajectory privacy-preserving method combined with prediction perturbation in LBS(CWPP). The method can achieve a reasonable allocation of privacy levels and thus allocate user privacy budgets, and prevent attackers from noisy filtering of user trajectory sequences. This will promote user service quality without disclosing user location privacy, and ensure a balance between track availability and security.

3. Preparatory knowledge.

3.1. Relevant definitions.

3.1.1. *Differential privacy.*

Definition 3.1. *ε-differential privacy. Suppose that there exist adjacent data sets D and D', D and D' differ by at most one record. Under the noise generation algorithm S, if any output results on D and D' satisfies Equation (1), then the algorithm S satisfies differential privacy.*

$$P_r[S(D) \in O] \leq e^{\varepsilon} P_r[S(D') \in O] \tag{1}$$

Where $P_r[\cdot]$ is the probability distribution, and ε is the privacy budget. The smaller the ε, the greater the noise added to the dataset and the better its privacy protection effect.

Definition 3.2. *Laplace mechanism. For a given dataset D, $f : D \to R^d$ is any query function on that dataset, and if the output of the function f satisfies Equation (2), f is proved to satisfy $\varepsilon-$differential privacy.*

$$S(D) = f(D) + Y \tag{2}$$

Where Y is the noise sequence and obeys the Laplace distribution, $Y = Laplace(\lambda)$. λ is the scale parameter, and $\lambda = \frac{\Delta f}{\varepsilon}$, Δf is the sensitivity of the query function.

Definition 3.3. *Geographical indistinguishability* [33]. *Suppose that there exist any two locations x and x', $d(x, x') \leq r$, where r is the noise radius. If the release location z satisfies Equation (3) under the action of the noise generation algorithm S, it is proved that S satisfies geographic indistinguishability.*

$$S(x)(z) \leq e^{\varepsilon d(x,x')} S(x')(z) \tag{3}$$

Definition 3.4. *User trajectory sequence. A user trajectory sequence is a set of Spatio-temporal correlation sequences containing location and time correspondences. For example, $H = \{(x_1, t_1), (x_2, t_2), (x_3, t_3)$
$, \ldots, (x_n, t_n)\}$, where $X = \{x_1, x_2, \ldots, x_n\}$ denotes the location points passed by the user and $T = \{t_1, t_2, \ldots, t_n\}$ represents the time sequence on the user's trajectory.*

Definition 3.5. *$\delta-$location set. Suppose that at the t_i moment, the user combines the probability transfer matrix with the current location to obtain the v possible location points at the next moment and forms the location set $(X')^{t_{i+1}} = \{(X')_1^{t_{i+1}}, \ldots, (X')_v^{t_{i+1}}\}$, which $P^{t_{i+1}}$=$\{p_1^{t_{i+1}}, \ldots, p_v^{t_{i+1}}\}$ represents the probability distribution of v possible position points. The $\delta-$position set is the most miniature position set whose cumulative probability value is not less than $1 - e^{1-\delta}$.*

$$\Delta X_t = \min\left\{p_j^{t_{i+1}} | \sum_{j=1}^{m} P^{t_{i+1}}[j] \geq 1 - e^{1-\delta}\right\} (\delta = m, \delta \geq 1) \tag{4}$$

Finally, from the set of v possible locations, $m(m \leq v)$ probable locations are selected to form a new location set, which is called $(X'')^{t_{i+1}} = \{(X'')_1^{t_{i+1}}, \ldots, (X'')_m^{t_{i+1}}\}$.

3.1.2. *Noise filtering.*

Definition 3.6. *Autocorrelation function* [34]. *The autocorrelation function is used to describe an interdependence of a specific stochastic process $X(t)$ at two different moments t_1 and t_2, it can represent the degree of correlation of the same sequence at different moments and is defined as $R_{XX}(t_1, t_2)$ =$E[X(t_1) * X(t_2)]$. Furthermore, if $X(t)$ is a* *smooth stochastic process, let $t_2 = t_1 + \tau$, at this time, the statistical properties of the stochastic process $X(t)$ are independent of the starting point and only related to its time interval τ, the autocorrelation function of the stochastic process $X(t)$ is a function of the time interval τ, and the autocorrelation function of the smooth stochastic process can be written as $R_{XX}(\tau)$.*

Since the user trajectory sequence is composed of position points and corresponding time points, and the correlation between the two positions before and after a short time is strong, the user trajectory sequence can be regarded as a short-term smooth process in a short time. According to the relevant knowledge of signal processing, the correlation of the short-term smooth process can be described using the autocorrelation function. Therefore, the autocorrelation function can explain the user trajectory sequence's correlation.

Definition 3.7. *Sequence indistinguishability* [35]. *If the autocorrelation function $R_{xx}(t_1, t_2)$ of the user's original trajectory sequence and the autocorrelation function $R_{zz}(t_1, t_2)$ of the trajectory sequence after adding noise satisfy*

$$R_{zz}(t_1, t_2) = R_{xx}(t_1, t_2) \tag{5}$$

Then the noise-added trajectory sequence and the user's original trajectory sequence satisfy indistinguishability for the attacker.

Since the noise added using differential privacy techniques is independent, the attacker will likely employ filtering attacks against the independent noise. This is shown in Figure 1. When independent noise is added to the user trajectory sequence using the Laplace mechanism, an attacker can use a filtering attack by using the autocorrelation information of the user trajectory sequence, which filters out the noise that is inconsistent with the user trajectory sequence, and results in a threat to the user's privacy security.

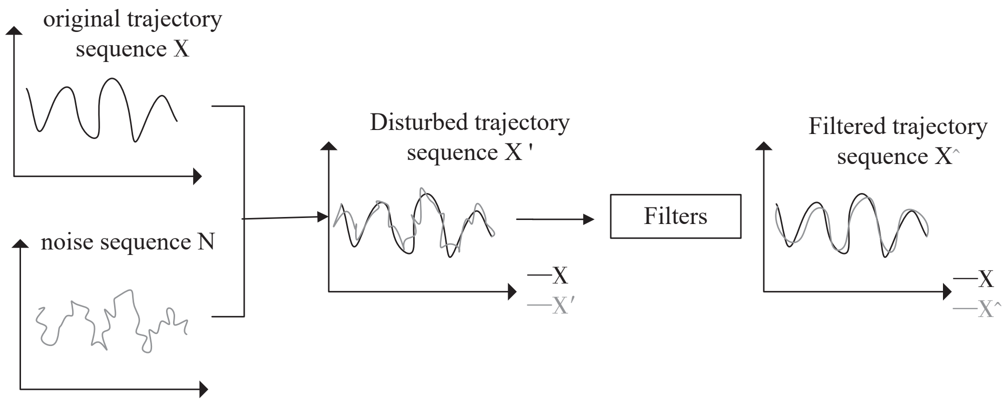

original trajectory sequence X
noise sequence N
Disturbed trajectory sequence X '
—X
—X′
Filters
Filtered trajectory sequence X^
—X
—X^

FIGURE 1. Noise filtering

We suppose the noise added to the user trajectory sequence has the same correlation with the user trajectory sequence. That is, the autocorrelation function of the noise sequence satisfies $R_Z(t_1, t_2) = R_{xx}(t_1, t_2)$, and user trajectory sequences have the same autocorrelation function as noise sequences in this case. Therefore, this ensures indistinguishability between the original trajectory sequence and the published trajectory sequence. As such, the attacker cannot filter out the noise through the filtering attack, thereby effectively protecting user privacy and security.

3.1.3. *Genetic algorithm.* A genetic algorithm is a kind of randomized search algorithm implemented using a genetic mechanism based on natural selection. It is an algorithm for solving optimal solutions for complex non-linear optimization problems. It has many applications in many fields, such as artificial intelligence and combinatorial optimization. The basic idea of a genetic algorithm is to start from multiple solutions when solving a problem and then iterate step by step through specific rules to generate new solutions. Its essential elements are: (1) Determine the encoding scheme. (2) Construct the fitness function according to the objective function. (3) Selecting the strategy. (4) Control of parameters. (5) Selection, crossover, and variation. (6) Specify a maximum evolutionary generation to make the algorithm stop.

3.1.4. *Service similarity matrix.* In the LBS service, users send requests to the LBS server based on their location information, and the LBS server returns query results to users based on their location information. The query results of these locations have a certain similarity, so the service similarity between locations is defined as follows. Suppose that there are two locations g and h, then, the service similarity between them can be defined as the following equation.

$$Q = sim(g,h) = \frac{|R_k(x_g,y_g) \cap R_k(x_h,y_h)|}{k} (0 \le Q \le 1) \tag{6}$$

Where $R_k(x_g,y_g)$ indicates the ranking result of the top k nearby interest points queried with location point g as the query point, $|\cdot|$ represents the number of elements in the set, that is, the number of common parts in the first k interest points queried by the two location points.

According to Equation (6), the service similarity between different locations can be calculated. Then the different location points in the map are labelled with the ordinal number $0, 1, 2, \ldots, c$, where $c + 1$ is the number of locations in the map, from which we can construct the service similarity matrix. The construction formula is as follows.

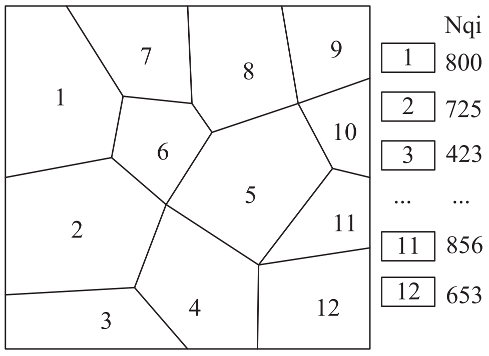
$$a_{ij} = \begin{cases} sim(g,h) & i \neq j \\ 1 & i = j \end{cases} \tag{7}$$

Where i, j denotes the user location point serial number, according to the above equation, we can get the service similarity matrix, as shown in Equation (8).

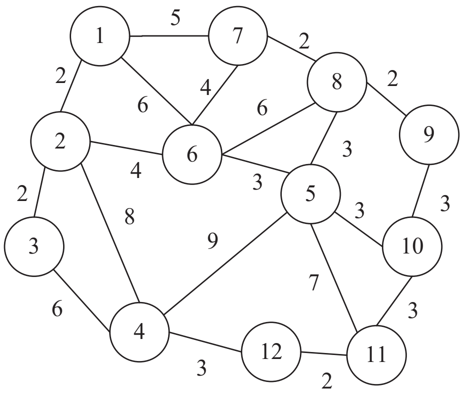
$$A = \begin{bmatrix} a_{00} & a_{01} & \cdots & a_{0c} \\ a_{10} & a_{11} & \cdots & a_{1c} \\ \vdots & \vdots & \ddots & \vdots \\ a_{c0} & a_{c1} & \cdots & a_{cc} \end{bmatrix} \tag{8}$$

Where the element a_{ij} in matrix A represents the service similarity between locations and $0 \le a_{ij} \le 1$.

3.1.5. *Definition of maps at the time of privacy level calculation.* We assume that the location points set constituted by all location points are $L = \{loc_0, loc_1, \ldots, loc_c\}$ in the map. In this paper, we use the plane sweep algorithm (because the algorithm is mature and not the focus, it is not repeated here) to divide the location points with different location points. Each location point will construct a Voronoi polygon. Only one discrete location point exists in each Voronoi polygon (the Voronoi graph's nature). Finally, the whole location map is partitioned into a Voronoi diagram, which ensures that each discrete location unit is not adjacent and guarantees the physical dispersion of the final selected predicted locations, as shown in Figure 2(a).



(a) Spatial location segmentation



(b) Geographic topology

FIGURE 2. Map definition when calculating privacy level

where Nqi represents the number of historical queries per location unit and the historical query probability of each location unit is $q_i = \frac{Nqi}{\sum_{i=0}^{c} Nqi}$, $0 \leq i \leq c$.

When calculating the privacy level, the user's map $G = (V, E, W)$ is an undirected graph composed of vertex set V, edge set E, and weights W on the edges. The vertex set V represents the set of all location points on the map. At the same time, different position regions are labelled with different numbers in the vertex set V. The edge set E represents if two location points are reachable between them, then two location points are connected by edges.

In addition, to calculate the sensitivity of each location point more accurately, in addition to introducing the distance between locations and the connectivity to measure, the transfer probability between location points is added at the same time, and this is used to measure the correlation between sites. According to the different location points in the map, we first construct the state space map based on Markov model, as shown in Figure 3.

where p_{ij} represents the transfer probability between positions, and

$$p_{ij} = P\{X_{t+1} = loc_j | X_t = loc_i\} = \frac{P\{X_{t+1} = loc_j, X_t = loc_i\}}{P\{X_t = loc_i\}} \tag{9}$$

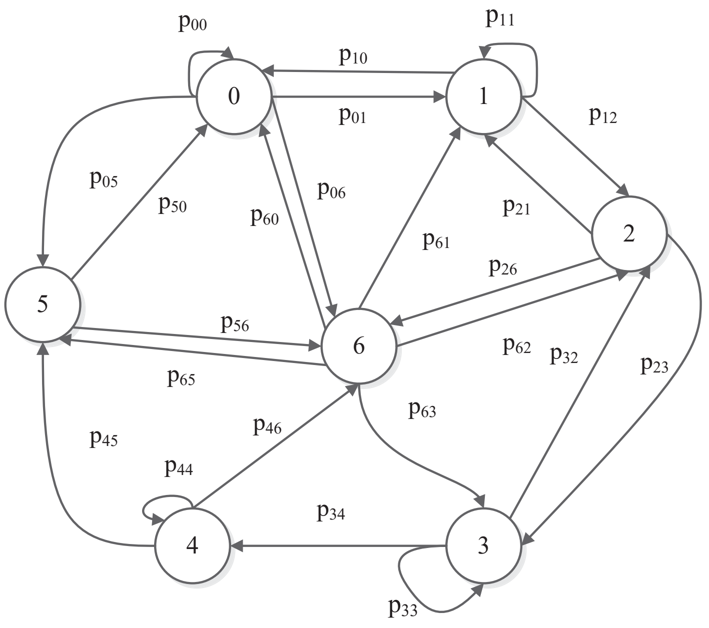


Figure 3. State space diagram

it represents the one-step transfer probability from location loc_i to loc_j at the current moment t. The edge server calculates the transfer probability between locations by measuring the user's historical trajectory data. The probabilistic transfer matrix M can be calculated from it, and

$$M = \begin{bmatrix} p_{00} & p_{01} & \cdots & p_{0c} \\ p_{10} & p_{11} & \cdots & p_{1c} \\ \vdots & \vdots & \ddots & \vdots \\ p_{c0} & p_{c1} & \cdots & p_{cc} \end{bmatrix} = \begin{bmatrix} \varphi_0 \\ \varphi_1 \\ \vdots \\ \varphi_c \end{bmatrix} \tag{10}$$

M is stored in the database of the edge server.

By calculating the transfer probability between locations, the edge weights

$$w = \begin{cases} \dfrac{dis_{ij}}{\min\{p_{ij}, p_{ji}\}} & |p_{ij} - p_{ji}| \le \zeta \\ \dfrac{dis_{ij}}{\bar{p}} & |p_{ij} - p_{ji}| > \zeta \end{cases} \tag{11}$$

where ζ is the probability difference threshold, it can be set according to user needs. dis_{ij} is the distance between two location points, $\bar{p} = \frac{p_{ij}+p_{ji}}{2}$. Therefore, this paper abstracts the spatial location partitioning graph as a location undirected graph, as shown in Figure 2(b).

Definition 3.8. *Location privacy level initialization. Before dividing the sensitivity levels, the user first needs to make an initial assignment of the privacy level of the map locations. According to the user's privacy needs, users first specify some initial sensitive areas, which form the set* $PL = \{PL_1, PL_2, \ldots, PL_{pl}\}$ *and the corresponding privacy level set* $PL^{initial} = \{PL_1^{initial}, PL_2^{initial}, \ldots, PL_{pl}^{initial}\}$*. Then the remaining locations on the map are further divided into non-initial sensitive position and geographically inaccessible places and form groups* $NPL = \{NPL_1, NPL_2, \ldots, NPL_{npl}\}$ *and*

$NA = \{NA_1, NA_2, \ldots, NA_{na}\}$. Thus, according to the user-defined privacy level assignment, all location points on the map are divided into $G_{map} = \{PL, NPL, NA\}$.

3.2. System model.

3.2.1. *System structure.* The system architecture of the proposed mechanism is shown in Figure 4, which mainly consists of three parts: user side, edge server, and LBS server. The user side primarily obtains the current location through the GPS positioning module and saves it in the database, and the user obtains the current location information from the database and sends it to the edge server; the edge server is divided into preprocessing data module, fake location generation module, query result screening module and cache list module, which mainly generates mock locations for the location information sent by users and filters the query results returned by the LBS server. At the same time, the preprocessing data module divides the map location privacy level and generates the noise sequence corresponding to the user's trajectory, and saves the result in the database; the LBS server mainly queries the corresponding result in the database for the location information sent by the edge server, and then sends the result back to the edge server.

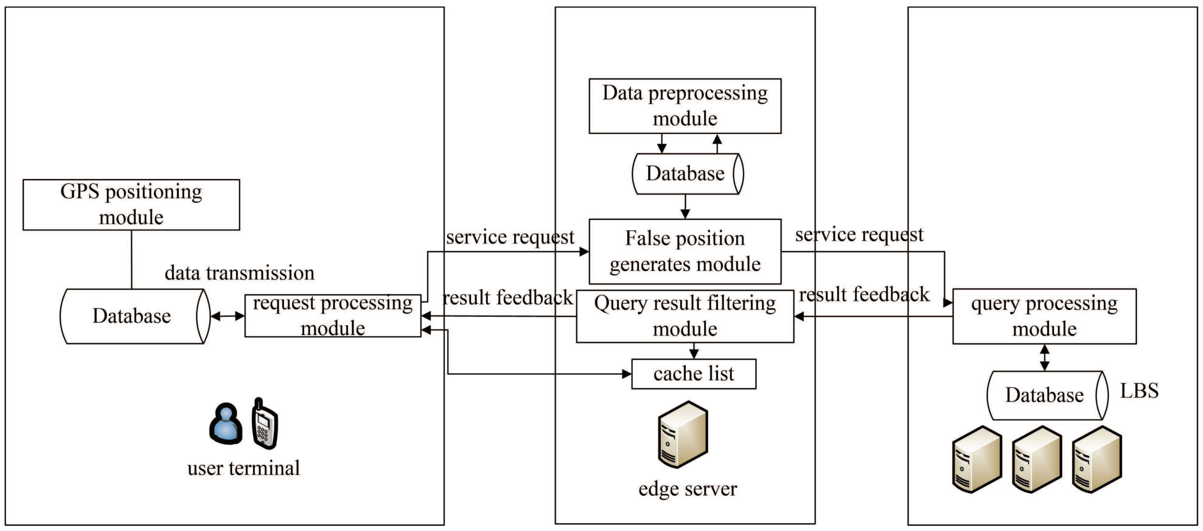


FIGURE 4. System structure diagram

3.2.2. $(R, \varepsilon)-$*Location differential privacy preservation model.* For the differential privacy protection approach, assigning the same privacy budget to different location points may not only waste a certain amount of privacy budget but also not meet the individual privacy needs of users. According to the sensitivity of the current site, we need a reasonable allocation of the privacy budget to ensure that different privacy budgets are allocated to varying locations. Therefore, a $(R, \varepsilon)-$privacy protection model is proposed in this paper, where $R = \varepsilon * pl$, and $R \in (0, 1]$. ε and pl indicate the privacy budget and privacy level of the current location, respectively. When $pl = 1$, the privacy budget parameter ε is a fixed constant, the privacy budget does not change with user demand or location sensitivity. In this case, the $(R, \varepsilon)-$privacy protection model is equivalent to the traditional $\varepsilon-$differential privacy protection model. The privacy budget parameter can be changed by introducing the $(R, \varepsilon)-$privacy protection model, which allows the dynamic addition of interference noise to change the privacy protection effect. Moreover, from the above equation, $\varepsilon = \frac{R}{pl}$, the current location privacy budget can be obtained by the given system parameters R and the calculated location point's privacy level pl. When R given, pl is inversely proportional to ε, that is, the smaller pl is, the larger the assigned privacy budget is, the smaller the privacy protection strength is, and when $pl \to 0$, $\varepsilon \to \infty$.

4. **Trajectory Privacy-preserving Method Combined with Prediction Perturbation in LBS.**

The CWPP method proposed in this article is suitable for the trajectory privacy protection scenarios in LBS. The core idea is to protect the service quality of the user's requested location while preserving the user's privacy, thereby improving the overall trajectory availability. Its main components include the following aspects:

(1) Privacy Level Assignment Mechanism

This mechanism allows users to customize the location sensitivity and reasonably assign the privacy level of each location point in terms of the distance between locations, the in/out-degree in the location undirected graph, and the transfer probability between sites.

(2) Noise Filtering

For the independent noise added by differential privacy, the noise sequence is pre-processed based on the filtering principle to obtain the noisy locations that satisfy the Spatio-temporal correlation, so that the attacker cannot filter the user trajectory sequence for the attack.

(3) Prediction Perturbation

We use the Markov prediction model and genetic algorithm to obtain perturbation locations that satisfy high data availability and privacy-preserving effects, and introduce detection function to check the availability of the perturbation location.

The specific process is shown in Algorithm 1.

Algorithm 1 CWPP

```
Input: user's current real location $x$, user trajectory sequence $H$, service similarity matrix
    $A$, location undirected graph $G$, privacy level division graph $G_{map}$, system parameters
    $R$, initial privacy level set $PL^{initial}$, sensitivity threshold $\Delta$, service similarity threshold
    $\alpha$, probability transfer matrix $M$
Output: user query result $SR_{user}$
 1: $\varepsilon = sensitivity\ assignment(G, G_{map}, R, PL^{initial}, \Delta)$
 2: $Z = noise\ filter(H, \varepsilon_H, \alpha)$      /* Initialize the noise sequence */
 3: if $findingofcache(x, cache) == 1$ then
 4:     $return$
 5: else
 6:     if $x.pl > \Delta$ then
 7:         $X^{(k-1)} = prediction(x, M)$
 8:         $x' = GA(X^{(k-1)})$
 9:         if $test(x, x', A, \alpha) = 1$ then
10:             $z = x'$
11:         else
12:             selecting the corresponding moment noise from the noise sequence $Z$
13:             if $(Z'.flag == 1)$ then
14:                 $z = x + Z'$ and $AOR_z = AOI_x$
15:             else
16:                 $z = x + Z'$ and $AOR_z = AOI_x + dis(z, x)$
17:             end if
18:         end if
19:         $X^{(k)} = X^{(k-1)} + z$
```

```
20:         $SR = X^{(k)} \rightarrow LBS$
21:         $cache \leftarrow SR_{X^{(k-1)}}$
22:         $SR_{user} = SR_z$
23:     **else**
24:         $SR_{user} = x \rightarrow LBS$
25:     **end if**
26: **end if**
27: $return\ SR_{user}$
```

In this algorithm, AOR_z represents the retrieval range of the user using the publishing location, AOI_x denotes the range of the user's interest points. The function $sensitivity\ assignment(G, G_{map}, R, PL^{initial}$
$, \Delta)$ allocates the privacy budget, calculates the privacy budget mainly according to the system parameters R and the sensitivity of each position point in the map, and saves the results into the set ε. See Algorithm 2 for details. The function $noise\ filter(H, \varepsilon_H, \alpha)$ mainly initializes the noise sequence to ensure that the user trajectory and noise sequence are not distinguishable. See Algorithm 3 for more information. The function $prediction(x, M)$ is the false position generation algorithm, which uses the prediction mechanism to generate an incorrect position set. See Algorithm 4 for details. In addition, the function $findingofcache(x, cache)$ is a lookup algorithm, which mainly looks up the corresponding results in the edge server cache list. If the corresponding result is found, it is returned directly, otherwise the scrambled position is generated instead of the user's real position for query. In Algorithm 1, a genetic algorithm is also used to screen false locations to obtain an area with the highest similarity to the user's actual location's service quality. Finally, the detection function $test(x, x', A, \alpha)$ judges whether the actual site and the disturbance location screened by the genetic algorithm meet the service similarity threshold.

4.1. **Sensitivity division.** According to the personalized privacy needs of users, different locations should be assigned different privacy levels. Although the literature [13,15,32] considers users' personalized privacy needs, the proposed method still has room for improvement. In addition, when calculating the sensitivity of a non-initial sensitive point, the existing methods often take only one initial sensitive location as the calculation criterion. However, in real situations, there may be more than one initial sensitive location connected with a non-initial sensitive point. In this paper, under the premise of fully considering users' personalized privacy requirements, we accurately calculate the sensitivity of locations by specifying some initial sensitive location points on the map in advance and converting the map into a location undirected graph. While we introduce the transfer probability between locations based on literature [13,15,32], and consider all initial sensitive locations connected with a non-initial sensitive points comprehensively by constructing a chain hash table, which assigns privacy levels to all location points more accurately and improves the efficiency of the privacy budget allocation. This is shown in Figure 5.

Suppose the privacy level of the sensitive location point s is pl, and the node connected to the sensitive node is l, and the formula of its sensitivity assignment is as follows.

$$l.pl = \frac{\left[\frac{1}{l.w}\right] * (s.pl)}{\sum_{l' eneighborset} \frac{1}{l'.w}} \tag{12}$$

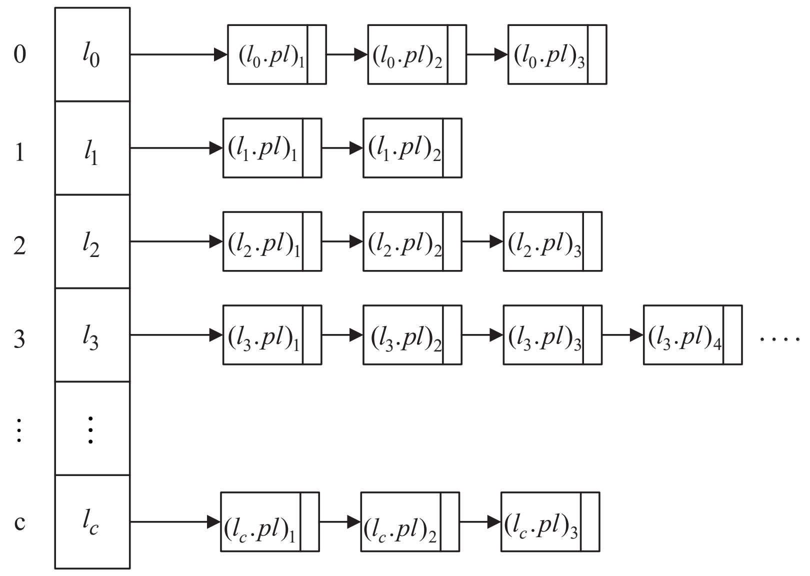


FIGURE 5. Chained hash table

Where $l.pl$ is the privacy level assigned to the node l, $neighborset$ is the connection setting of the sensitive node s, and its size is equal to the degree of the node s in the map. w denotes the weight between the node l and the sensitive node s. According to the above privacy level assignment method and calculation formula, this paper proposes the following privacy level assignment algorithm.

Algorithm 2 Privacy Level Assignment (PLA) algorithm

```
Input: location undirected graph $G$, privacy level partitioning graph $G_{map}$, system parameters $R$, initial sensitive region privacy level set $PL^{initial}$, sensitivity threshold $\Delta$
Output: the privacy budget set $\varepsilon$
1:  for $all(l \in NPL\ \&\ \&\ l \notin NA)$ do
2:      $L[] = l$
3:  end for
4:  for $all\ s \in PL$ do
5:      $s.\varepsilon = \frac{R}{s.pl}$
6:      $\varepsilon = \varepsilon \cup \{s.\varepsilon\}$
7:  end for
8:  $l = NPL.head()$
9:  while $l! = null$ do
10:     $neigborset' = findneigborsets(l)$      /* find the neighboring vertices of node $l$ */
11:     for $all\ s \in\ neigborset'$ do
12:         if $(s \notin NPL \cap s \notin NA)$ then
13:             $neigborset =\ findneigborsets(s)$
14:             $newpl = allocprivacylevel(l)$      /*privacy level calculation according to Equation (12) */
15:             $l.pl = newpl$
16:             place $l.pl$ at the end of the single-linked table corresponding to the node $l$
17:         end if
18:     end for
19:     $l = l.next()$
```

```
20: end while
21: for $all\ \ l \in L[]$ do
22:     $\overline{l.pl} = \sum_{i=1}^{j} \frac{l_1.pl+l_2.pl...l_j.pl}{j}$
23:     if $\overline{l.pl} \le \Delta$ then
24:         $l.\varepsilon = 0$
25:     else
26:         $l.\varepsilon = \frac{R}{l.pl}$
27:         $\varepsilon = \varepsilon \cup \{l.\varepsilon\}$
28:     end if
29: end for
30: $return\ \ \varepsilon$
```

In this algorithm, the $findneigborsets(s)$ function obtains the neighboring vertices set of the current node. Since each non-initial sensitive point may have more than one initial sensitive node connected, so we use the chain storage structure of the hash table to store the privacy level calculated for each time according to different sensitive nodes, its average value is used as the final sensitivity of the location point. The sensitivity threshold Δ can play a specific regulatory role. When the calculated privacy level value is less than the sensitivity threshold, we do not assign the corresponding privacy budget for the point. That is, we make the privacy budget value of the point 0, and the location point can be directly used as the release location. Finally, we get the map's privacy budget for all reachable location points.

4.2. **Noise preprocessing.** The noise added to the user trajectory sequence belongs to independent noise. As a result, the attacker with autocorrelation information of the user trajectory sequence will likely adopt a filtering attack to leak user privacy information. Therefore, in this paper, based on the literature [35], by calculating the autocorrelation function $R_{xx}(\tau)$ of the user trajectory sequence and the power spectral density N_0 of the Gaussian white noise, we can design the impulse response function $h(\tau) = \sqrt{\frac{R_{xx}(\tau)}{16\pi N_0}}$ of the filter. We convert Y_j through the impulse response function of the filter into a correlation Gaussian noise sequence $Y_j^{'}$, and the autocorrelation function of $Y_j^{'}$ satisfies $R_{Y_j^{'}}(\tau) = \sqrt{\frac{R_{xx}(\tau)}{8}}(j = 1, 2, 3, 4)$, we let

$$Z = {Y_1^{'}}^2 + {Y_2^{'}}^2 - {Y_3^{'}}^2 - {Y_4^{'}}^2 \tag{13}$$

then

$$\begin{aligned} R_Z(\tau) = {} & E[({y_1^{'}}^2(t) + {y_2^{'}}^2(t) - {y_3^{'}}^2(t) - {y_4^{'}}^2(t)) \cdot ({y_1^{'}}^2(t+\tau) + {y_2^{'}}^2(t+\tau) - \\ & {y_3^{'}}^2(t+\tau) - {y_4^{'}}^2(t+\tau))] \\ & = 4[R^2{}_{Y_1^{'}}(\tau) + R^2{}_{Y_2^{'}}(\tau) - R^2{}_{Y_3^{'}}(\tau) - R^2{}_{Y_4^{'}}(\tau)] \\ & = 8R^2{}_{Y^{'}}(\tau) \end{aligned} \tag{14}$$

From this we can get $R_Z(\tau) = R_{xx}(\tau)$, so that the noise sequence and the user trajectory sequence satisfy the sequence indistinguishability. At the same time, we consider the user's personalized privacy requirements, assign different privacy budgets to different location points on the trajectory, and further consider the user's service quality.

For user location $x = (x_1, y_1)$, when the privacy budget ε is specific, if after adding the noise, the probability density function of the generated position z satisfies $P(x)(z) = \frac{\varepsilon^2}{2\pi}e^{-\varepsilon d(x,z)}$, and the noise addition mechanism is said to fulfill $\varepsilon-$location differential privacy. The above equation is the two-dimensional noise probability density function in Cartesian coordinates. To facilitate the calculation, we convert it to polar coordinates. In polar coordinates, the probability density function is shown below.

$$p_\varepsilon\left(r,\theta\right) = \frac{\varepsilon^2}{2\pi}re^{-\varepsilon r} \tag{15}$$

From this, we can obtain the marginal probability density function on r and θ by its two-dimensional probability density function.

$$p_\varepsilon(r) = \varepsilon^2 re^{-\varepsilon r} \tag{16}$$

$$p_\varepsilon(\theta) = \frac{1}{2\pi} \tag{17}$$

Where r is the noise radius, that is, the distance between the published location and the user's actual location, the range of values of r is $r \in (0, \infty)$, and when r is more considerable, the user's actual location is farther away from the published site, and the service quality is poorer, so the range of r needs to be limited. To ensure the service quality for users, we set a threshold value $r_{\max}$ of noise radius, so that $r \leq r_{\max}$. The new noise radius takes the value range of $0 \leq r \leq r_{max}$. To make each generated noise radius satisfy $r \leq r_{\max}$, we first find the cumulative probability distribution function of $p_\varepsilon(r)$, as the following equation.

$$C_\varepsilon(r) = \int_0^r p_\varepsilon(\mu)d\mu = 1 - (1 + \varepsilon r)e^{-\varepsilon r} \tag{18}$$

Then $r = C_\varepsilon^{-1}(\omega)$, where ω denotes the probability distribution value of the noise radius r, and there

$$C_\varepsilon^{-1}(\omega) = -\frac{1}{\varepsilon}(W_{-1}(\frac{\omega - 1}{e}) + 1) \tag{19}$$

W_{-1} denotes the Lambert W function (-1 branch), and since $C_\varepsilon^{-1}(\omega)$ is an increasing function, let

$$r_{\max} = -\frac{1}{\varepsilon}(W_{-1}(\frac{\omega - 1}{e}) + 1) \tag{20}$$

We get

$$\omega_{\max} = [-(\varepsilon r_{\max} + 1)e^{-(\varepsilon r_{\max}+1)}]e + 1 \tag{21}$$

From this, we can limit the range of noise radius r by $\omega_{\max}$, only needing to make the generated ω satisfy $0 \leq \omega \leq \omega_{\max}$, then we can get $0 \leq r \leq r_{max}$.

From the above ideas, we can get the algorithm for noise preprocessing under the simultaneous consideration of privacy level and service quality. Its specific algorithm is as follows.

Algorithm 3 Noise Sequence Preprocessing (NSP) algorithm

Input: user trajectory sequence H, the privacy budget set corresponding to the user trajectory sequence ε_H, the service similarity threshold α

Output: noise sequence Z

```
1: for all $\varepsilon_i \in \varepsilon_H$ do
2:     if $(\varepsilon_i \neq 0)$ then
3:         $\exists Y_j \in Z \ \ (j = 1, 2, 3, 4)$
4:         $S(D) = f(D) + Laplace\left(\frac{\Delta f}{\varepsilon_i}\right)$
5:         $\Pr[x] = \frac{1}{2\lambda} e^{-\frac{|x-\mu|}{\lambda}}$
6:         generate four sets of Gaussian white noise, and $Y_j \sim N(0, \sqrt{2\lambda})$
7:     end if
8: end for
9: for all $Z' \in Y_j$ do
10:    if $(sim(x, z_x) \geq \alpha)$ then
11:        $Z'.flag = 1$
12:    else
13:        Generate a random number $\omega$ in $[0, \omega_{\max}]$
14:        Recalculate $\theta$ and $r$ from Equations (17) and (19), respectively
15:        $Z' = \langle r' cos(\theta), r' sin(\theta) \rangle$
16:        $Z'.flag = 0;$
17:    end if
18: end for
19: Calculate the autocorrelation function of the user trajectory sequence $R_{xx}(\tau)$
20: $h(\tau) = \sqrt{\frac{R_{xx}(\tau)}{16\pi N_0}}$
21: $Y_1' = Y_1 \otimes h(\tau), Y_2' = Y_2 \otimes h(\tau), Y_3' = Y_3 \otimes h(\tau), Y_4' = Y_4 \otimes h(\tau)$   /*$\otimes$ stands for convolution*/
22: $Z = {Y_1'}^2 + {Y_2'}^2 - {Y_3'}^2 - {Y_4'}^2$
23: return $Z$
```

In Algorithm 3, when the privacy budget ε_i is not 0, we add noise to the user location according to the Laplace mechanism. When the privacy budget is 0, no noise is added to the user location, and the user's actual location is taken as the published location. Then the Gaussian white noise is convolved with the filter impulse response function $h(\tau)$ to obtain the correlated Gaussian noise sequence. Finally, this noise sequence has a consistent autocorrelation function with the user trajectory sequence. In the process of privacy budget allocation, the privacy budget consumed is $\sum_{i=1}^{n} \varepsilon_i$, as shown by the serial combinatoriality of differential privacy.

4.3. **Forecasting phase.** Since the user's mobile trajectory is time series, the user's next mobile position is usually only related to the current position, not to the past, which satisfies the Markov stable non-sequential property, so this paper adopts the first-order Markov model to simulate the correlation between the user's positions, and the probability transition matrix M is used to calculate the transition probability of users between various locations.

First of all, we regard each mobile process of the user as a Markov process, suppose the probability distribution of the user at $t-1$ moment is λ_{t-1}, then the probability distribution at moment t satisfies $\lambda_t = \lambda_{t-1}M$. If we know that the user's current location is x, then the probability distribution at this time is $\lambda_i = [0,\ldots,1,\ldots,0,\ldots,0]$, when the user sends its actual location to the edge server at the moment t_i, after the prediction, the probability distribution of the next moment is λ_{i+1}, and $\lambda_{i+1} = \lambda_i M = \varphi_i = [p_{i0}, p_{i1}, \ldots, p_{ic}]$, we remove some position points with probability 0 and will eventually get a predicted site set $(X')^{t_{i+1}} = \{(X')_1^{t_{i+1}}, \ldots, (X')_v^{t_{i+1}}\}$, then use the $\delta-$position set to filter out some possible positions with low probability to get a new place set, denoted as $(X'')^{t_{i+1}} = \{(X'')_1^{t_{i+1}}, \ldots, (X'')_m^{t_{i+1}}\}(m \le v)$.

In addition, we consider that the attacker may infer the possibility of query through the location attack, so the query probability generalization is performed on the filtered predicted positions. Firstly, the edge server calculates the average query probability $\bar{q}$ of the filtered places, and then by setting the threshold parameter β, such that the standard deviation of the query probability of the final selected $k-1$ locations from the average query probability $\bar{q}$ does not exceed β, that is, the selected $k-1$ positions satisfy the following equation.

$$X^{(k-1)} = \left\{ X_i^{(k-1)} | \sqrt{\frac{\sum_{i=1}^{k-1}(q_i-\bar{q})^2}{k-1}} \le \beta \right\} (k-1 \le m) \tag{22}$$

Where $X^{(k-1)}$ denotes the selected false location set, as this paper divides the spatial location segmentation stage with a Voronoi diagram, so that the filtered wrong area set will satisfy both physical dispersion and query probability generalization, thus preventing attackers from inferring the actual location of the user at the next moment through the predicted site set. The specific algorithm is as follows.

Algorithm 4 False Position Selection (FPS) algorithm

```
Input: actual user location $x$, probability transfer matrix $M$
Output: the false position set $X^{(k-1)}$
1: Construct a probability transfer matrix $M$ based on the user's historical track record
2: Predict a dummy location set satisfying physical dispersion from the user's actual
   location $x$ and the probability transfer matrix $M$, noted as $X^{(v)}$
3: $X = \emptyset$
4: for all $X^{(m)} \in X^{(v)}$ do
5:     if $\sum_{j=1}^{m} P[j] \ge 1 - e^{1-\delta}$ then
6:         $X = X \cup X^{(m)}$
7:     end if
8: end for
9: for all $X^{(m)} \in X$ do
10:    The smallest value of $\sum_{j=1}^{m} P[j]$ is selected
11: end for
12: The final pseudo-position set $X^{(k-1)}$ is obtained by selecting the position set according
    to Equation (22)
13: return $X^{(k-1)}$
```

The above algorithm can finally obtain a predicted location set with the same query probability and satisfying physical dispersion, we use this set as the dummy location set. At the same time, using predicted locations as a collection of dummy locations and caching their query results can reduce the number of interactions between users and LBS and the risk of user privacy leakage.

4.4. **Location release phase.** After the edge server generates the $k-1$ predicted locations, the edge server also needs to filter the predicted generated $k-1$ locations using a genetic algorithm, whose specific steps are as follows.

(1) For the obtained $k-1$ predicted locations, as the initial population.

(2) Define the fitness policy: we perform service similarity detection on the predicted $k-1$ location and the user's real location x, and rank them between each site and the actual location. At the same time, we set a survival rate e to select the first few predicted locations with the highest service similarity. In the selection process, we use the fitness proportion method of selection. The probability of an individual being selected is

$$\rho_i = \frac{Q_i}{\sum_{i=1}^{k-1} Q_i}(1 \leq i \leq k-1) \tag{23}$$

Where Q_i denotes the service similarity between the location set $X^{(k-1)}$ and the user's actual location. When the service similarity is higher, its probability of being selected is more significant.

(3) Crossover: for any two location points $m_1(x_1, y_1)$ and $m_2(x_2, y_2)$, we swap their coordinate information to get two new location points $m_1'(x_1, y_2)$ and $m_2'(x_2, y_1)$.

(4) Variation: for a position point (x_1, y_1), we generate a random number r, and the mutated position point is $(x_1 + r, y_1 + r)$. After the crossover and variation operation, the elements in the filtered position set reach $k-1$ again.

(5) The predetermined number of algorithm cycles is $maxiter$. If the number of cycles is not reached, the algorithm returns to step 2 to continue the process. Otherwise, the algorithm ends and produces a location with the highest service similarity to the user's actual location and records it as x'. For the detection part, we compare the service similarity of the location x' generated by the genetic algorithm and the user's actual location in this paper. Its detection function $test(x, x', A, \alpha)$ is as follows.

$$test(x, x', A, \alpha) = \begin{cases} 1 & sim(x, x') \geq \alpha \\ 0 & else \end{cases} \tag{24}$$

If the detection function is 1, then x' is the release location. Otherwise, the location that adds the perturbed noise to the user's actual location is used as the publishing location. This published location z is then sent to the LBS server along with the set of predicted locations to obtain query results. Further, for the query result returned by the LBS server, the current scrambled location query result is substituted for the user's actual location query result, and the predicted location query result is saved in the edge server cache list for use in the following query.

5. **Security analysis.** In this paper, we assume that the LBS server is an untrustworthy entity and an attacker with a lot of background knowledge, which can infer user privacy based on general background information. This scheme can effectively protect user privacy for the LBS server, the details are as follows.

When the user sends its location information to the edge server, it first predicts a false location set based on the user's location information. It then generates a perturbed site based on a genetic algorithm along with the predicted location to the LBS server for the query. At the same time, the LBS server with more vital attack capability attempts to infer user privacy based on the location information sent by the user. However, firstly, to prevent adjacent predicted locations from being sent by users, this paper uses a spatial location map for Voronoi diagram segmentation, which provides the physical dispersion of the fake areas; secondly, the sites in the counterfeit location set processed by the edge server have the same query probability, that is, they satisfy $q_i \approx q_j$ (q_i, q_i denotes the historical query probability of loc_i, loc_j). According to the formula of location entropy $H(x) = -\sum_{i=1}^{k\text{-}1} f_i \log(f_i)$ (where $f_i = \frac{q_i}{\sum_{i=1}^{k\text{-}1} q_i}$), the entropy is higher when the query probability is closer, i.e., the attacker has more uncertainty about the user's location, preventing the attacker from inferring the actual location of the user in the next moment by the predicted site sent. Moreover, the perturbed site filtered by the genetic algorithm has uncertainty, from the location, the attacker has difficulty obtaining the user's privacy. Meanwhile, the edge server stores the query results returned by the LBS server in its cache list, and the user first queries in the cache list each time, which reduces the number of interactions with the LBS server and the risk of privacy leakage. Also caching with predicted location query results can improve cache hit rate. Furthermore, the user's actual location is never sent during the whole service request process, and the probability that an attacker can infer the user's actual location is low.

In addition, considering the attacker's filtering attack based on the autocorrelation information of the user trajectory sequence, the scheme in this paper initializes the noise added to the user trajectory sequence, we convert the Gaussian white noise into a correlated Gaussian noise sequence with an autocorrelation function satisfying $R_{Y'_j}(\tau) = \sqrt{\frac{R_{xx}(\tau)}{8}} (j = 1, 2, 3, 4)$, and finally superimpose it to form a correlated Laplace noise sequence, so that the autocorrelation function of the user trajectory sequence and the noise sequence satisfy $R_{xx}(t_1, t_2) = R_Z(t_1, t_2)$, and the user trajectory and the noise sequence meet sequence indistinguishability, which avoids the attacker to obtain the user trajectory privacy through noise filtering.

6. **Experiment and analysis.**

6.1. **Experimental setup.**

In this paper, based on the architectural model of the location privacy protection system, we conduct simulation experiments on this scheme in terms of privacy protection degree, data availability and average time cost, and compare it with the methods of literature [15], DP-UR [32], and AGENT [26]. The simulation experiments are implemented using the Python language, and the experimental environment is configured with a 4.0 GHz CPU, 12 GB RAM, and a 64-bit Windows 10 platform. GeoLife [36] was chosen as the testing dataset.

6.2. **Experimental results and analysis.**

First we set some experimental parameters, the relevant parameters involved in the experiments are shown in Table 1. Then we analyze the experimental results below.

6.2.1. *Degree of privacy protection.*

The experiments in this section mainly analyze the privacy protection degree of the method in this paper, which is divided into two aspects: mutual information and noise coverage, compared with the literature [15] method, DP-UR method, and AGENT method.

Table 1. Experimental parameter setting table

Experimental parameters	Range of values	Default Value
track length L	$10 \le L \le 35$	15
privacy Budget ε	$0.2 \le \varepsilon \le 0.9$	0.4
sensitivity Threshold Δ	$0.06 \le \Delta \le 0.16$	0.12
availability Threshold α	$0.2 \le \alpha \le 1.0$	0.3
Dataset size $\lvert D\rvert$	$400K \le \lvert D\rvert \le 600K$	600K
the number of initial sensitive locations η	$4 \le \eta \le 14$	6

1) Mutual information. Mutual information is a useful measure of information in information theory. As a kind of information, privacy can be quantified by information entropy. We use MI to represent the interdependence between two sets, which is represented by the probability that an attacker uses filtering attacks and query probability inference attacks to identify the real location of the user.

This is shown in Figure 6. When $\varepsilon = 0.4$, $\Delta = 0.06$, and $\alpha = 0.2$, the MI of all four methods increases with the increase of the track length, since both the method in this paper and DP-UR preprocess the noisy sequences to avoid the attacker from filtering the independent noise using the filtering attack, their MI is lower than the remaining two methods. In addition, the method in this paper reduces the number of interactions with the LBS server to a certain extent through the caching mechanism, and generalizes the query probability of the predicted locations when sending the set of predicted locations to the LBS server, which avoids the query probability inference attack of the attacker and further reduces the MI of the method in this paper. The literature [15] method combines the prediction mechanism with the differential privacy mechanism for privacy protection. In contrast, the AGENT method entirely consists of the scrambled location generated by differential privacy for the service request, so the MI is higher.

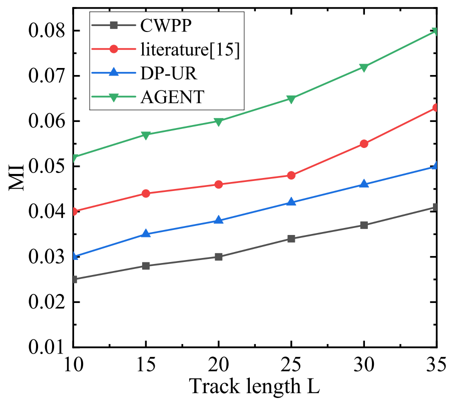


Figure 6. Mutual information

2) Noise coverage. The noise coverage indicates that when the privacy budget ε is specific, the proportion of the noise that can be added to the user's actual location accounts for the track length. When there are many position points in the trajectory or the privacy budget is low, the noise coverage will affect the user's privacy protection effect, and when the noise added by differential privacy is independent noise without considering, the higher the noise coverage, the better the privacy protection effect.

When $L = 35, \Delta = 0.10, \alpha = 0.6$, according to Figure 7, the noise coverage increases with an increase in the privacy budget. Since the AGENT method uses the R-tree to store the perturbed positions after noise addition by the differential privacy mechanism, when perturbing the position point, it first searches for the disturbed sites that meet the requirements in the R-tree. When there is no place to meet the requirements, the Laplace mechanism is used to add noise to the current actual location. Therefore, the noise coverage is higher than that of the independent differential privacy noise-addition mechanism. In this paper, because the privacy level of all locations is calculated using a chained hash table in the initial privacy level allocation stage, the privacy budget allocation for each site is also more reasonable. Its noise coverage is higher than that of the remaining three methods. So, without considering the attacker filtering attack, the privacy protection effect of this paper is still higher than that of the remaining three methods.

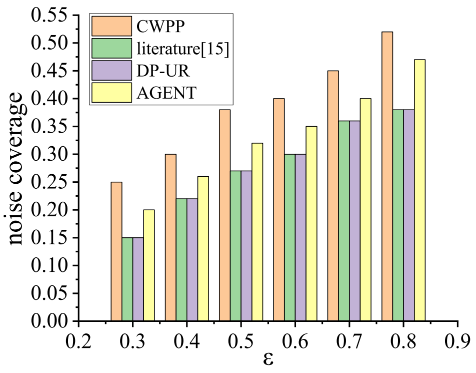


FIGURE 7. Noise coverage

6.2.2. *Data availability.* In this paper, we find that the detection success rate directly affects the usability and privacy risk of LBS. Therefore, improving the detection success rate is crucial for the privacy protection scheme combined with the prediction mechanism. Figure 8 analyzes the relationship between the availability threshold and detection success rate and compares the method used in this paper and the literature [15]. When $\varepsilon = 0.6$, $\Delta = 0.07$, and $L = 25$, the detection success rate decreases with increasing availability thresholds for both, but the detection success rate of the method in this paper is always higher than that in the literature [15], mainly because compared to the literature [15], this paper uses a genetic algorithm for location screening. In contrast, the exponential mechanism used in the literature [15] only selects a predicted location from a specified location set (the selection range is fixed in that predicted location set), while the optimal solution selected by the genetic algorithm is not necessarily in that predicted location set (it can be other locations), so the final selected location has a better service quality and higher service similarity compared to the actual location, and therefore can pass the detection function more easily.

To evaluate the usability of the final user-posted trajectory, this paper uses the root mean square error between the user's actual trajectory and the posted trajectory to measure. Where the user's actual trajectory location and the user's published trajectory location are respectively $X = \{x_1, x_2, \ldots, x_n\}$ and $Z = \{z_1, z_2, \ldots, z_n\}$, then

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} \left(d_i - \bar{d}\right)^2}{|L|}} \tag{25}$$

Where d_i denotes the distance between the user's actual trajectory position and the published trajectory position, $\bar{d}$ represents the average Euclidean distance between the

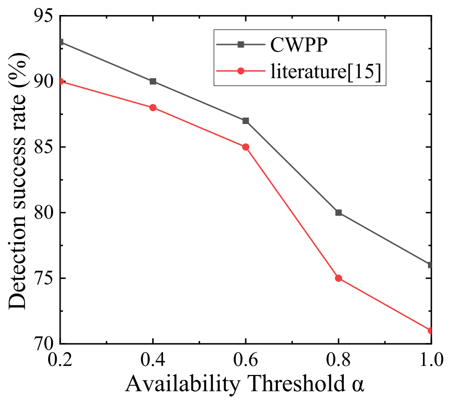


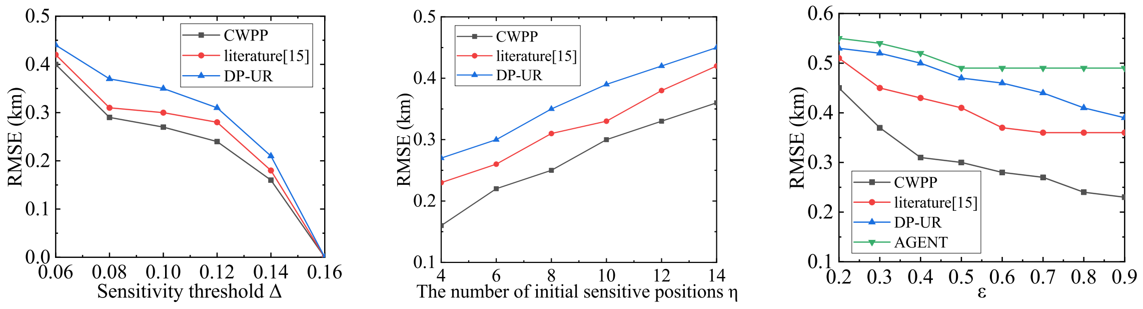
Figure 8. Detection success rate

user's actual trajectory position and the published trajectory position on the trajectory, $|L|$ is the trajectory length. In addition, for the convenience of calculating the trajectory availability, when the user queries the service information through the caching mechanism, the published position is the user's current actual position by default.

Figure 9(a) gives the relationship between sensitivity thresholds and trajectory availability for the three methods in this paper, the literature [15] and the DP-UR. We assume that the upper limit of the sensitivity threshold is $\Delta = 0.16$, and when $L = 30$, $\varepsilon = 0.7$, $\alpha = 0.4$, as the sensitivity threshold increases, the RMSE values of all three methods decrease. However, the RMSE values of this paper's approach are always lower than the remaining two schemes, which indicates that the trajectory availability of this paper's method is always the best. It is mainly because of the way this paper uses a genetic algorithm in the dummy location screening stage, considers the service quality of noisy locations, and introduces a caching mechanism, so usability is the best. Moreover, because the literature [15] method incorporates a detection mechanism to detect the availability of published locations, its RMSE value is lower than that of the DP-UR method. When the sensitivity threshold reached 0.16, the RMSE values of all three methods reached 0 simultaneously because all location points on the trajectory are less sensitive, are not sensitive locations, and can be published directly, so the average RMSE of the trajectory is 0.

Figure 9(b) gives the relationship between the number of initial sensitive locations η and RMSE for the three methods. When $L = 32, \varepsilon = 0.7, \Delta = 0.07, \alpha = 0.3$, as the number of initial sensitive locations η increases, the trajectory availability of all three ways decreases. More locations in the map are assigned to the privacy level due to the rise in η, which leads to fewer directly publishable locations and worse trajectory availability. The method in this paper and the literature [15] combine the prediction disturbance and use the detection function to detect the availability of the disturbance location. Compared with the independent differential privacy mechanism, its trajectory availability is better.

Figure 9(c) presents the relationship between the privacy budget and RMSE for the four methods. When $L = 32$, $\Delta = 0.07$, $\alpha = 0.3$, with the increase of the privacy budget, the RMSE continues to decrease. The method in this paper and the literature [15] combine the prediction disturbance and detect the disturbance position availability through the detection function. Compared with DP-UR and AGENT, its RMSE value is lower. In addition, the method in this paper improves the noise location availability by reducing the noise radius, so that the published location retrieval range covers the actual location interest point range, so the trajectory availability is the highest. Since the method in literature [15] adopts the prediction and detection mechanism before noise disturbance, the privacy budget spent when the successful prediction is smaller than the budget spent on generating the disturbed location. When its privacy budget is $\varepsilon = 0.7$, it reaches the maximum privacy budget consumed by its entire service request. Because the AGENT method uses R-tree to store the perturbation position to achieve the reusability of the perturbation position, compared with the method in the literature [15], it does not use the exponential mechanism to consume the privacy budget in the false position screening stage. When its privacy budget is $\varepsilon = 0.5$, its privacy budget consumption reaches the maximum value, and its RMSE value will not change when the privacy budget is increased again.

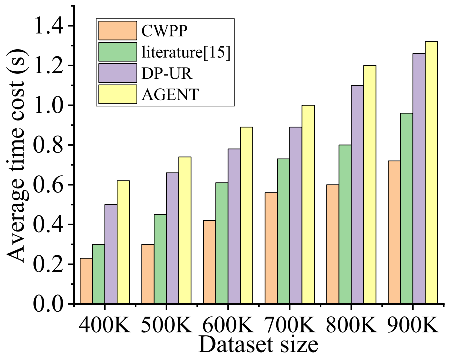


(a) Influence of Δ on RMSE (b) Influence of η on RMSE (c) Influence of ε on RMSE

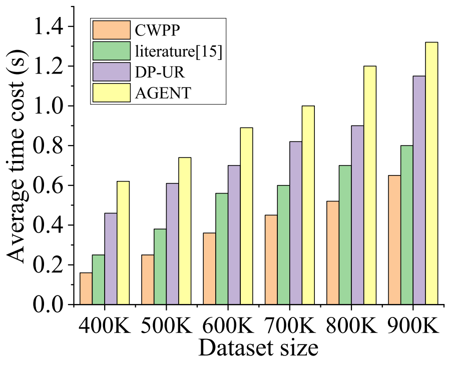
FIGURE 9. Influence of different parameters on RMSE

6.2.3. *Average time cost.* Figure 10 analyzes the relationship between dataset size and algorithm average time overhead for different sensitivity thresholds. The average time cost represents the average time spent by the user to generate the location to be published each time. The query is performed by randomly truncating subsets of different sizes from the dataset Geolife. According to Figure 10, the average time overhead increases as the dataset becomes larger. Since the method in this paper caches the predicted location query results, the user first looks up the corresponding results in the cache each time, which greatly reduces the average time cost of generating the location to be published, while AGENT needs to look up the corresponding results in the R-tree each time, and its average time cost is higher. In addition, the average time overhead decreases when the sensitivity threshold increases, because as the sensitivity threshold increases, more locations become available for direct publishing. This reduces the average time cost of generating the locations to be published.

7. **Conclusion.** As a result of insufficient privacy protection and service quality due to independent noise in the existing trajectory differential privacy protection, the proposed solution addresses problems of unreasonable privacy level allocation and insufficient level of privacy protection, etc. Firstly, the privacy level for each location point is reasonably assigned, and the noise sequence is initialized so that it meets the sequence indistinguishability with the user trajectory sequence. Secondly, the Markov prediction model and genetic algorithm are used to obtain perturbed locations with high availability and high privacy protection effect. At the same time, the detection function are introduced to check the availability of the perturbed location. In addition, a caching mechanism is used to reduce the number of interactions between users and LBS. Finally, the comparison of experimental results shows that the proposed method has significant effects on privacy protection level and service quality.



(a) $\Delta = 0.12$



(b) $\Delta = 0.15$

FIGURE 10. The relationship between average time cost and dataset size

In the current study, we should also consider the semantic information of the predicted locations and the fact that the prediction accuracy can be further improved. In the future, we plan to consider the semantic similarity of predicted locations and build deep-learning prediction models to improve the accuracy of predicted locations. At the same time, we will incorporate query content privacy protection to increase the effect of user privacy protection.

8. **Acknowledgement.** This work was supported by the National Natural Science Foundation of China (61300216), Key scientific research projects of colleges and universities in Henan Province (23A520033) and Doctoral Scientific Fund of Henan Polytechnic University(B2022-16).

REFERENCES

[1] S. Zhang, X. Li, Z. Tan, T. Peng, and G. Wang, "A caching and spatial k-anonymity driven privacy enhancement scheme in continuous location-based services," *Future Generation Computer Systems*, vol. 94, pp. 40–50, 2019.

[2] Y. Qiu, Y. Liu, X. Li, and J. Chen, "A novel location privacy-preserving approach based on blockchain," *Sensors*, vol. 20, no. 12, pp. 3519–3536, 2020.

[3] Q. Mei, H. Xiong, Y.-C. Chen, and C.-M. Chen, "Blockchain-enabled privacy-preserving authentication mechanism for transportation cps with cloud-edge computing," *IEEE Transactions on Engineering Management*, pp. 1–12, 2022.

[4] K. Renuka, S. Kumar, S. Kumari, and C.-M. Chen, "Cryptanalysis and improvement of a privacy-preserving three-factor authentication protocol for wireless sensor networks," *Sensors*, vol. 19, no. 21, pp. 1–15, 2019.

[5] A. Tadakaluru, "Context optimized and spatial aware dummy locations generation framework for location privacy," *Journal of Geovisualization and Spatial Analysis*, vol. 6, no. 2, pp. 1–12, 2022.

[6] L. Zhang, J. Li, S. Yang, Y. Liu, X. Zhang, and Y. Sun, “A markov prediction-based privacy protection scheme for continuous query,” *Journal of Circuits, Systems and Computers*, vol. 28, no. 09, pp. 1–20, 2019.

[7] S. Zhang, X. Mao, K.-K. R. Choo, T. Peng, and G. Wang, “A trajectory privacy-preserving scheme based on a dual-k mechanism for continuous location-based services,” *Information Sciences*, vol. 527, pp. 406–419, 2020.

[8] K. Kita, Y. Koizumi, and T. Hasegawa, “Private retrieval of location-related content using k-anonymity and application to icn,” *Computer Networks*, vol. 209, pp. 1–14, 2022.

[9] S. Zhang, G. Wang, M. Z. A. Bhuiyan, and Q. Liu, “A dual privacy preserving scheme in continuous location-based services,” *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 4191–4200, 2018.

[10] J. Xiong and H. Zhu, “Real-time trajectory privacy protection based on improved differential privacy method and deep learning model,” *Journal of Cloud Computing*, vol. 11, no. 1, pp. 1–15, 2022.

[11] K. Qian and X. Li, “Lbs user location privacy protection scheme based on trajectory similarity,” *Scientific Reports*, vol. 12, no. 1, pp. 1–12, 2022.

[12] C.-Y. Lin, “Suppression techniques for privacy-preserving trajectory data publishing,” *Knowledge-Based Systems*, vol. 206, pp. 1–15, 2020.

[13] C. Wu, H. Cheng, S. Zhao, W. Liang, Y. Wu, C. Li, and X. Zhang, “Differentially private trajectory protection based on spatial and temporal correlation,” *Chinese Journal of Computers*, vol. 41, no. 2, pp. 309–322, 2018.

[14] C. Dwork, “Differential privacy,” in *Automata, Languages and Programming*, M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1–12.

[15] A. Ye, L. Meng, Z. Zhao, Y. Diao, and J. Zhang, “Trajectory differential privacy protection mechanism based on prediction and sliding window,” *Journal on Communications*, vol. 41, no. 04, pp. 123–133, 2020.

[16] M. Min, W. Wang, L. Xiao, Y. Xiao, and Z. Han, “Reinforcement learning-based sensitive semantic location privacy protection for vanets,” *China Communications*, vol. 18, no. 6, pp. 244–260, 2021.

[17] Z. Gao, Y. Huang, L. Zheng, H. Lu, B. Wu, and J. Zhang, “Protecting location privacy of users based on trajectory obfuscation in mobile crowdsensing,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6290–6299, 2022.

[18] C. Xu, L. Zhu, Y. Liu, J. Guan, and S. Yu, “Dp-ltod: Differential privacy latent trajectory community discovering services over location-based social networks,” *IEEE Transactions on Services Computing*, vol. 14, no. 4, pp. 1068–1083, 2021.

[19] R. Al-Dhubhani and J. M. Cazalas, “An adaptive geo-indistinguishability mechanism for continuous lbs queries,” *Wireless Networks*, vol. 24, no. 8, pp. 3221–3239, 2018.

[20] C. Yin, X. Ju, Z. Yin, and J. Wang, “Location recommendation privacy protection method based on location sensitivity division,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, pp. 1–13, 2019.

[21] J. Zhang, Y. Li, Q. Ding, L. Lin, and X. Ye, “Successive trajectory privacy protection with semantics prediction differential privacy,” *Entropy*, vol. 24, no. 9, pp. 1172–1190, 2022.

[22] S. Chen, A. Fu, J. Shen, S. Yu, H. Wang, and H. Sun, “Rnn-dp: A new differential privacy scheme base on recurrent neural network for dynamic trajectory privacy protection,” *Journal of Network and Computer Applications*, vol. 168, pp. 1–11, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804520302101

[23] L. Yao, Z. Chen, H. Hu, G. Wu, and B. Wu, “Privacy preservation for trajectory publication based on differential privacy,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 13, no. 3, pp. 1–21, 2022.

[24] W. Cheng, R. Wen, H. Huang, W. Miao, and C. Wang, “Optdp: Towards optimal personalized trajectory differential privacy for trajectory data publishing,” *Neurocomputing*, vol. 472, pp. 201–211, 2022.

[25] S. Yuan, D. Pi, X. Zhao, and M. Xu, “Differential privacy trajectory data protection scheme based on r-tree,” *Expert Systems with Applications*, vol. 182, pp. 1–12, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417421006485

[26] X. Ma, J. Ma, H. Li, Q. Jiang, and S. Gao, “Agent: an adaptive geo-indistinguishable mechanism for continuous location-based service,” *Peer-to-Peer Networking and Applications*, vol. 11, no. 3, pp. 473–485, 2018.

[27] H. Li, X. Ren, J. Wang, and J. Ma, “Continuous location privacy protection mechanism based on differential privacy,” *Journal on Communications*, vol. 42, no. 8, pp. 164–175, 2021.

[28] H. Chen, S. Li, and Z. Zhang, "A differential privacy based (k-psi)-anonymity method for trajectory data publishing," *CMC-COMPUTERS MATERIALS & CONTINUA*, vol. 65, no. 3, pp. 2665–2685, 2020.

[29] R. Wen, R. Zhang, K. Peng, and C. Wang, "Protecting locations with differential privacy against location-dependent attacks in continuous lbs queries," in *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2021, pp. 379–386.

[30] X. Niu, H. Huang, and Y. Li, "A real-time data collection mechanism with trajectory privacy in mobile crowd-sensing," *IEEE Communications Letters*, vol. 24, no. 10, pp. 2114–2118, 2020.

[31] X. Chen, T. Zhang, S. Shen, T. Zhu, and P. Xiong, "An optimized differential privacy scheme with reinforcement learning in vanet," *Computers & Security*, vol. 110, pp. 1–15, 2021.

[32] Z. Hu and J. Yang, "Differential privacy protection method based on published trajectory cross-correlation constraint," *Plos One*, vol. 15, no. 8, pp. 1–25, 2020.

[33] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, 2013, pp. 901–914.

[34] B. Basrak, *The sample autocorrelation function of non-linear time series.* Rijksuniversiteit Groningen Groningen, Netherlands, 2000.

[35] H. Wang, Z. Xu, L. Xiong, and T. Wang, "Clm: differential privacy protection method for trajectory publishing," *Journal on Communications*, vol. 38, no. 06, pp. 85–96, 2017.

[36] Y. Zheng, X. Xie, and W.-Y. Ma, "Geolife: A collaborative social networking service among user, location and trajectory." *IEEE Data(base) Engineering Bulletin*, vol. 33, no. 2, pp. 32–39, 2010.