Privacy-Preserving Compressive Sensing for Crowdsensing based Trajectory Recovery

Linghe Kong*‡, Liang He†, Xiao-Yang Liu‡, Yu Gu§, Min-You Wu‡, Xue Liu*
*McGill University, Canada, Email: linghe.kong@mail.mcgill.ca, xueliu@cs.mcgill.ca
†University of Michigan, USA, Email: lianghe.umich@gmail.com
‡Shanghai Jiao Tong University, China, Email: {linghe.kong, yanglet, mwu}@sjtu.edu.cn
§IBM Research - Austin, USA, Email: yugu@us.ibm.com

***Abstract*—Location based services have experienced an explosive growth and evolved from utilizing a single location to the whole trajectory. Due to the hardware and energy constraints, there are usually many missing data within a trajectory. In order to accurately recover the complete trajectory, crowdsensing provides a promising method. This method resorts to the correlation among multiple users' trajectories and the advanced compressive sensing technique, which significantly outperforms conventional interpolation methods on accuracy. However, as trajectories exposes users' daily activities, the privacy issue is a major concern in crowdsensing. While existing solutions independently tackle the accurate trajectory recovery and privacy issues, yet no single design is able to address these two challenges simultaneously. Therefore in this paper, we propose a novel *Privacy Preserving Compressive Sensing* (PPCS) scheme, which encrypts a trajectory with several other trajectories while maintaining the homomorphic obfuscation property for compressive sensing. Under PPCS, adversaries can only capture the encrypted data, so the user privacy is preserved. Furthermore, the homomorphic obfuscation property guarantees that the recovery accuracy of PPCS is comparable to the state-of-the-art compressive sensing design. Based on two publicly available traces with numerous users and long durations, we conduct extensive simulations to evaluate PPCS. The results demonstrate that PPCS achieves a high accuracy of <53 m and a large distortion between the encrypted and the original trajectories (a commonly adopted metric of privacy strength) of >9,000 m even when up to 50% original data are missing.**

I. Introduction

Location based services (LBSs) [18, 37] have experienced an explosive growth recently, which are evolving from utilizing a single location [7] to harness the complete trajectory of a mobile user [23, 26, 40]. For example, the *Moves* application, which automatically tracks both activities and trajectories of users, has been downloaded over 4 million times since its launch in Jan. 2013 and has been acquired by Facebook [1].

Although GPS is universally available on modern devices, the trajectory of a mobile user may always be incomplete due to none-line-of-sight to satellites [29]. In addition, since GPS consumes a significant amount of energy, it is only activated periodically to conserve energy [22]. Consequently, the trajectory recovery [30] is one of the fundamental components of LBSs to estimate the missing data in a incomplete trajectory. For instance, trippermap [4] in Flickr can automatically reproduce a user's travelling path based on her geotagged photos.

Considerable interpolation methods have been devoted to trajectory recovery. With a single user's incomplete trajectory data, the methods such as nearest neighbors [31] and linear interpolation [34] can attain only coarse-grained accuracy. More recently, Rallapalli *et al*. [29] reveal that the trajectories of multiple users within the same geographic area are strongly correlated. For instance, students in the same campus have similar time tables; vehicles in the same segment of freeway moves with similar velocities. Leveraging such correlations, the crowdsensing technology [10, 11, 16, 17, 28, 35] provides a promising recovery method, which collectively recovers all users' trajectories together using compressive sensing (CS). This crowdsensing recovery method is proved to be superior to interpolation methods with only single user data [29].

While the crowdsensing recovery method accomplishes the better accuracy, the major drawback for applying it in practice is its requirement to collect all users' location data, which poses great concerns for potential privacy leakage [32, 36]. Especially in crowdsensing, the users are willing to contribute their personal data only when their privacies are preserved. Currently, the most commonly adopted privacy-preserving approach is anonymization [24]. Nevertheless, latest studies [12, 38] reveal that the anonymization mechanism alone is inadequate. To further improve the privacy, dummification [19] and obfuscation [13, 15, 27] methods are introduced, which inject fake trajectories and perturb original trajectories, respectively. Although dummification and obfuscation methods reasonably protect user privacies, they also pollute the original data, which decreases the recovery accuracy with current crowdsensing recovery method.

To tackle the challenges of accurate trajectory recovery and privacy-preserving simultaneously, we design a novel encryption method named K-vector perturbation (KVP) to attain both objectives. The main idea of KVP is to use a private key to perturb a user's trajectory with K other trajectories while maintaining the homomorphic obfuscation property for compressive sensing. Based on KVP, we propose the privacy-preserving compressive sensing (PPCS) scheme including three major steps. First, every user encrypts her incomplete location data by KVP and transmits the data in encrypted form to the crowdsensing server. Second, the server does not need to decrypt the data but directly recovers all users' encrypted data together with CS. Third, a user downloads her corresponding recovered data and decrypts her own trajectory by inverse KVP. Under PPCS, adversaries are possible to capture the

encrypted data but do not know the private key, so that the users' privacies are preserved. Furthermore, PPCS guarantees the recovery accuracy is the same by operating CS on the encrypted data and the original data, which is named by the homomorphic obfuscation property.

The contributions of this paper are summarized as follows:

- To the best of our knowledge, this is the first work to jointly optimize the data recovery accuracy and user privacy preservation in crowdsensing.
- We propose a systematic PPCS scheme for crowdsensing based trajectory recovery, which combines the novel homomorphic obfuscation method KVP into compressive sensing framework to accomplish the recovery accuracy and the privacy preservation simultaneously. Since the design of KVP is simple, PPCS is easy to be implemented.
- We theoretically prove that PPCS achieves the same recovery error bound as CS. Moreover, we prove that the expectation of distortion between encrypted data and original data is relatively large compared with the size of the area, which indicates the effective data perturbation for privacy preservation. We also prove that PPCS can protect the user privacy as long as there are no more than K actual data being exposed, where K can be proactively controlled according to the user requirement on privacy.
- Extensive simulations are conducted to evaluate PPCS, which are based on two publicly available traces from Beijing and Shanghai with large amount of users, long durations, and mixed mobility modes including walking, biking, and driving. The evaluation results show the effectiveness of PPCS. Typically, using PPCS on Beijing traces achieves the average accuracy within 53 meters and the average distortion more than $9,000$ meters even up to 50% original data are missing.

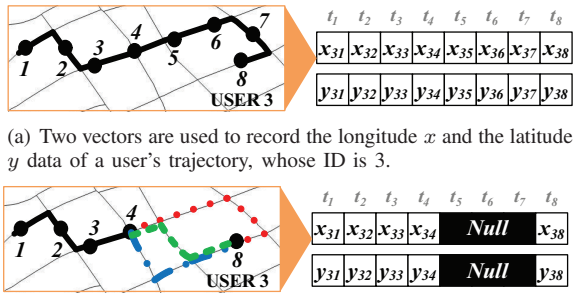
The remainder of this paper is organized as follows. In Section II, we formulate the trajectory recovery problem. In Section III, we investigate the mobility property in real traces. We describe the design of PPCS in Section IV, and analyze the theoretical recovery accuracy and privacy in Section V. In Section VI, we evaluate our scheme based on trace-driven simulations. In Section VII, we review the related work. And we conclude this work in Section VIII.

II. Preliminaries

In this section, we introduce the trajectory recovery model, the adversary model, and the formal definition of our problem.

A. Trajectory Recovery Model

A trajectory is composed of a sequence of locations that a user traverses, represented by her corresponding longitude x and latitude y, as shown in Fig. 1(a). The user's current location (x, y) can be obtained through the GPS module on her mobile device. In an N-user system where the total duration of interests consists of T time slots, the trajectory of the i-th user is represented by two $1 \times T$ vectors, where x_{ij} and y_{ij} are the longitude and latitude at the j-th time slot respectively ($i = 1, 2, \cdots, N$ and $j = 1, 2 \cdots, T$).



(a) Two vectors are used to record the longitude x and the latitude y data of a user's trajectory, whose ID is 3.

(b) When some location data are missing, the corresponding elements in the vectors are null. It is not easy to directly recover the accurate trajectory due to several possible paths in map.

Fig. 1. Trajectory model.

The location data of a user could be partially missing due to reasons such as none-line-of-sight to GPS satellites, energy management of GPS module on mobile devices [22] and so on. In Fig. 1(b), the null elements in the vectors indicate the data missing at their corresponding time slots.

The trajectory recovery is not effective if it is performed for individual users independently. For example, as shown in Fig. 1(b), the location data for the 5-th to the 7-th time slots are missing. Even though the map matching [33] method is utilized to narrow down the field of candidates, there are still three possible trajectories provided by linear interpolation [31]. To address the weakness of the single user recovery, the crowdsensing recovery exploits the correlation among users and recovers all users' trajectories together using compressive sensing, which is verified to outperform existing methods [29] and is referred as the state-of-the-art in this paper.

The notations of the crowdsensing recovery are defined as:

- *Trajectory Matrix* is a set of N users' actual trajectories, which is defined as $X = (x_{ij})_{N \times T}$. We only illustrate the longitude X related definitions and derivations in the following sections. All results for the latitude Y are similar to X, which are omitted for conciseness.
- *Binary Index Matrix* is used to indicate whether a location data in X is missing, which is defined as

$$\Phi = (\phi_{ij})_{N \times T} = \begin{cases} 0 & \text{if } x_{ij} \text{ is missing,} \\ 1 & \text{otherwise.} \end{cases} \quad (1)$$

- *Sensed Matrix* consists of the sensed location data from GPS. Due to the potential data missing, elements in the sensed matrix S are either x_{ij} (*i.e.*, sensed location data) or 0 (*i.e.*, missing data). Thus, S can be presented by[1]

$$S = X \circ \Phi. \quad (2)$$

- *Recovered Matrix* is generated by recovering the missing data in the sensed matrix S to approximate the actual trajectories X. The recovered matrix is denoted by $\hat{X}$.
- *Compressive Sensing* (CS) is an advanced recovery technique [6, 9] utilized to recover the missing data in S. We use f_{cs} to denote the CS operation, thus $\hat{X} = f_{cs}(S)$.

[1]In this paper, '$X\Phi$' represents the matrix multiplication of X and Φ, while '$X \circ \Phi$' represents the element-wise multiplication of X and Φ.

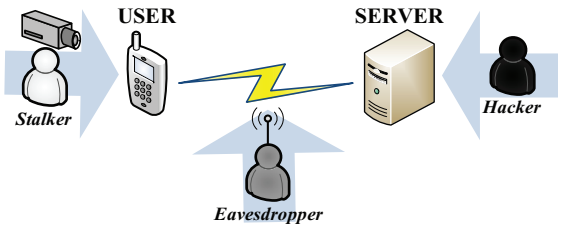


Fig. 2. Adversary models.

B. User Models and Adversary Models

We consider a system consisting of two types of mobile users: public and private users. Public users are willing to share their trajectories and private users want to avoid the exposure of their trajectories. For example, in an urban traffic scenario, buses can be treated as public users, and personal vehicles are good examples of private users.

As leakage of personal trajectories can lead to unauthorized surveillance and tracking, adversaries are motivated to obtain private users' trajectories. In Fig. 2, we illustrate the adversary models that threaten the privacy in crowdsensing recovery, which are categorized as eavesdroppers, hackers, and stalkers.

• *Eavesdroppers and hackers*: An eavesdropper could capture the data traffic between users and the crowdsensing server by hijacking the communication channels. A hacker could access and obtain all data in the server. Because eavesdroppers and hackers can obtain the same set of information, we do not differentiate them in the rest of the paper.

• *Stalkers*: A stalker can track a user for a short while and obtain k actual location data of that user. Without loss of generality, we assume that k is a small number compared with the total number of data in a complete trajectory, because a stalker cannot always tail after the user.

All adversaries potentially have the following capabilities: (i) they have the same algorithms as ours to recover the trajectory; (ii) they can exploit existing map matching methods [25, 33] as ours to further improve their estimation accuracy.

C. Problem Definition

In this paper, we consider the *accurate and privacy-preserving trajectory recovery* problem. This problem is challenging because the two objectives appear to be conflicted with each other. On one hand, a highly accurate recovery can be achieved by the crowdsensing method. However, this method requires to collect data from all users, which poses the potential privacy leakage. On the other hand, the privacy objective is to avoid the exposure of users' trajectories, which is contrary to the basic requirement of crowdsensing. Existing methods cannot satisfy the two objectives simultaneously.

To address this dilemma in crowdsensing based trajectory recovery, we propose the PPCS scheme, in which a novel homomorphic obfuscation method for CS is designed to preserve the user privacy and guarantee the recovery accuracy as well.

III. Trace Preprocessing and Validation

Before describing the design of PPCS scheme, we introduce two real traces and validate their low-rank properties, which implies the strong correlation among multiple users' trajectories within the same area [29].

TABLE I
Selected Real-World Mobility Traces

Name	Size	Area	Mobility Mode
Beijing	116 users × 355 slots	70×85 km^2	Walk/Bike/Car
Shanghai	74 users × 399 slots	100×100 km^2	Taxi/Bus

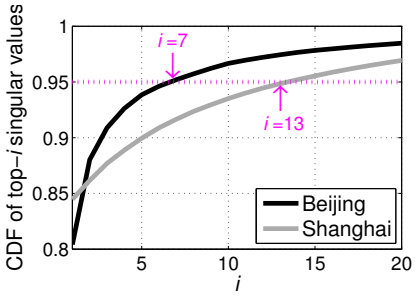


Fig. 3. Low-rank property in the investigated mobility traces.

A. Preprocessing of Real-World Mobility Traces

The evaluation of our design is based on two publicly available mobility traces: Geolife [2] and SUVnet [3]. These two traces have large amount of users, long durations, and mixed mobility modes. Geolife records the GPS trajectories of 178 users from April 2007 to October 2011 in Beijing, in which the major user mobility modes include walking, biking, and driving. SUVnet records the trajectories of over 2000 taxis and 300 buses in the urban area of Shanghai.

However, the raw traces from Geolife and SUVnet cannot be directly utilized for low rank validation, because significant amount of their data are missing. To guarantee the integrity of ground truth, we perform trace preprocessing on the raw data to select their complete subsets and build the trajectory matrices, which are then utilized in our evaluations. The description of the two selected traces including their sizes, areas, mobility modes are shown in Table I, which are denoted as Beijing and Shanghai, respectively.

B. Validating the Low Rank Property

As CS is a major component of PPCS, we first need to validate whether the trajectory matrices are low-rank, which is the requirement for the CS operation [9].

Although the low-rank property has been studied in [29], each of their traces has only one mobility mode: either human walking or car driving. The mobility mode mixed with walking, biking, and driving together in our selected traces is a more general scenario. In addition, some of their traces [29] are synthetic. However, our traces are raw data gathered from real applications, which inherently have noises. Hence, we still need to verify whether such traces are universally low-rank.

We verify the low-rank property of the selected traces by *Singular Value Decomposition* (SVD), which is an effective non-parametric technique for rank investigation [21]. According to SVD, an $N \times T$ matrix X can be decomposed as

$$X = U\Lambda V' = \sum_{i=1}^{min(N,T)} \sigma_i u_i v_i', \tag{3}$$

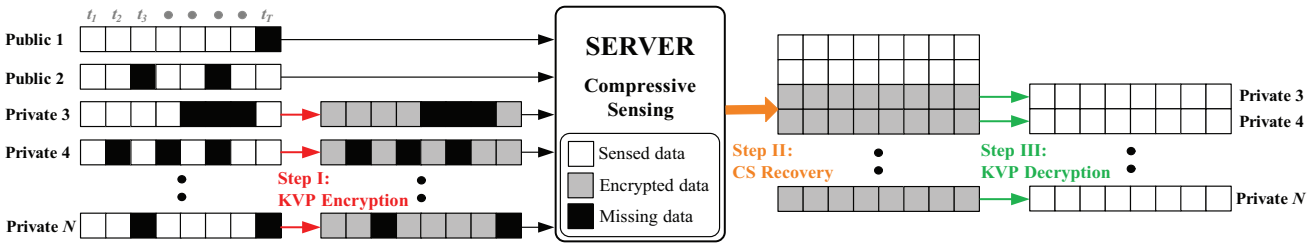


Fig. 4. PPCS overview.

where U and V are two unitary matrices, V' is the transpose of V, and Λ is an $N \times T$ diagonal matrix containing the singular value σ_i of X. Typically, the singular values σ_i are sorted as $\sigma_i \geq \sigma_{i+1}, (i = 1, 2, \cdots, min(N, T))$, where $min(N, T)$ is the number of singular values. The rank of the matrix X, denoted by r, is the number of its non-zero singular values. The matrix is low-rank if $r << min(N, T)$. If the top-$\hat{r}$ singular values have a good approximation of the total singular values, i.e.,

$$\sum_{i=1}^{\hat{r}} \sigma_i \approx \sum_{i=1}^{min(N,T)} \sigma_i, \tag{4}$$

this matrix is considered to be near low-rank, and $\hat{r}$ is treated as its rank.

The CDF of the singular values obtained from the Beijing and Shanghai traces are shown in Fig. 3, where the x-axis presents the i-th largest singular values, and the y-axis is the ratio between the sum of the top-i singular values and the sum of all singular values. We find that the total singular values are well approximated by only a few top singular values in both traces. For example, the top-7 σ_i of the Beijing and the top-13 σ_i of the Shanghai occupy more than 95% of their respective total values, while the total numbers of σ_is are 116 and 74 respectively. This observation reveals that both traces are of the near low-rank property. Hence, CS can be applied on them to achieve a promising recovery accuracy.

IV. PPCS Scheme

To address the privacy issue in conventional CS, we present a simple but efficient trajectory recovery scheme *Privacy-Preserving Compressive Sensing* (PPCS) in this section.

A. Scheme Overview

The proposed PPCS consists of three steps. First, users encrypt their sensed data and transmit the encrypted trajectories to the server. Note that the encrypted trajectories may not be complete because of the data missing issue. Second, the server performs CS on the collective data to recover the missing part of the encrypted trajectory for all users. Third, any individual user downloads the recovered and encrypted trajectory from the server, and decrypts it to obtain her original trajectory. An overview of these three steps is shown in Fig. 4. Briefly, the advantages of PPCS are:

- The design of PPCS is simple, and thus it is easy to be implemented in practice.
- PPCS tactfully takes advantage of CS to provide significant privacy preservation strength while guaranteeing the accuracy of recovered trajectories.
- The high-complexity CS recovery is computed at the centralized server side. The distributed computing at the user side is the low-complexity encryption and decryption.
- The communication overhead of every user is very small.

Analyses of these advantages are provided in Section V-A to V-D. Moreover, the case of no public vectors in PPCS are discussed in Section V-E.

B. Encrypt the Sensed Trajectories at Individual Users

The core component of PPCS is to encrypt the sensed trajectories at private users, so that only their encrypted trajectories are available at the server. Denote f_{en} as the encryption operation. With a sensed trajectory $S_{(i)}$ of the user i, the encrypted trajectory can be represented as

$$\mathbb{S}_{(i)} = f_{en}(S_{(i)}), \tag{5}$$

where $S_{(i)}$ presents the i-th row vector in the matrix S.

In the next, we explain how the encryption operates in detail. In the system under consideration, public users are willing to share their trajectories, which are available at the server. At the first phase of encryption, a private user i randomly downloads K public vectors $D_{(1)}, D_{(2)}, \cdots, D_{(K)}$ from all public vectors at the server, which is utilized to generate the encrypted vector $\mathbb{S}_{(i)}$. Only K-vector downloading does not lead to much communication overhead. In addition, random downloading brings more uncertainty for privacy preservation.

Then, user i generates a length-$(K+1)$ random vector $<\psi_{i,0}, \psi_{i,1}, \psi_{i,2} \cdots, \psi_{i,K}>$ as her *private key*, which is not shared to any other including the server. Any key satisfies $\psi_{i,j} \in (0, 1)$ and $\sum_{j=0}^{K} \psi_{i,j} = 1$. With the public vectors and the private key, user i generates her encrypted vector $\mathbb{S}_{(i)}$ as

$$\mathbb{S}_{(i)} = (\psi_{i,0} S_{(i)} + \psi_{i,1} D_{(1)} + \cdots + \psi_{i,K} D_{(K)}) \circ \Phi_{(i)}. \tag{6}$$

To demonstrate the encryption operation, let us consider the example shown in Fig. 5. Assume a private user $i = 4$ has downloaded $K = 2$ public vectors from the server (*i.e.*, $D_{(1)}, D_{(2)}$), and has generated the length-3 key $<\psi_{4,0}, \psi_{4,1}, \psi_{4,2}>$. The three vectors $S_{(4)}$, $D_{(1)}$, and $D_{(2)}$ are summed up with weight $\psi_{4,0}$, $\psi_{4,1}$, and $\psi_{4,2}$ respectively. For each null element in $S_{(4)}$, the corresponding element in the resultant sum vector is treated as the missing data and the encrypted vector $\mathbb{S}_{(4)}$ is then transmitted to the server.

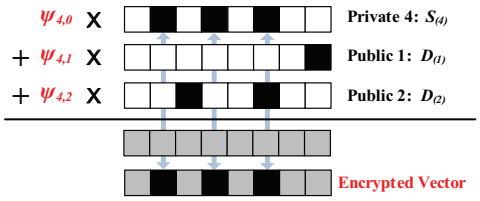


Fig. 5. KVP encryption

This encryption method is referred to as K-Vector Perturbation (KVP) in this paper, because 1) from the aspect of matrix operation, f_{en} is essentially a linear combination of K *vectors* in a matrix, and 2) the physical meaning of f_{en} is to *perturb* the user trajectory with other K public trajectories.

Intuitively, the length of private key dominates the difficulty for adversaries to decrypting the original data. Hence, the value of K determines the privacy preservation strength offered by KVP. We will further discuss the impact of K on the performance of PPCS in Section V-C.

C. Recover the Encrypted Trajectories at the Server

After collecting the encrypted trajectories from all private users and original trajectories of all public users, the server forms the encrypted matrix $\mathbb{S}$ of size $N \times T$. Then, crowdsensing recovery method applies CS on $\mathbb{S}$ and the completed encrypted trajectory matrix is obtained as $\hat{\mathbb{X}} = f_{cs}(\mathbb{S})$.

The f_{cs} operation adopted in this paper is the standard CS recovery. Briefly, the procedures of f_{cs} are as follows:

- Assume that $\hat{\mathbb{X}}$ can be divided into L and R matrices according to SVD factorization.

$$\hat{\mathbb{X}} = U\Lambda V' = LR', \tag{7}$$

  where $L = U\Lambda^{1/2}$, $R = V\Lambda^{1/2}$.
- Estimate L and R matrices by

$$\min((||L||_2^2 + ||R'||_2^2) + \lambda||B \circ (LR') - \mathbb{S}||_2^2), \tag{8}$$

  where the Lagrange multiplier λ allows a tunable tradeoff between rank approximation (due to the real data is near low-rank but not exact low-rank) and accuracy fitness. In Eq. (8), 1) B and $\mathbb{S}$ are known, 2) any $||\cdot||_2^2$ is non-negative, 3) the optimal values approximate 0 by minimizing all non-negative parts. Hence, L and R can be obtained by iterative computing, *e.g.*, genetic algorithm.

In summary, f_{cs} is equal to solve $\hat{\mathbb{X}} = LR'$ from Eq. (8) with input $\mathbb{S}$. Please refer to [6] [29] for the detailed CS operation.

D. Decrypting the Recovered Trajectories at Individual Users

After the encrypted trajectories are recovered at the server, any individual user can download her corresponding encrypted trajectory and apply the decryption operation. Specifically, user i downloads $\hat{\mathbb{X}}_{(i)}$ from the server, and locally decrypts it with the public vectors and her private key as

$$\hat{X}_{(i)} = (\hat{\mathbb{X}}_{(i)} - (\psi_{i,1}D_{(1)} + \cdots + \psi_{i,K}D_{(K)}))/\psi_{i,0}, \tag{9}$$

where $\hat{X}_{(i)}$ is the approximation of $X_{(i)}$, *i.e.*, the recovered complete trajectory of user i. Due to the local decryption and the private key, $\hat{X}_{(i)}$ is only known by user i herself. Then, a user can exploit map matching methods [33], which normally adjust the recovered trajectory by matching the nearest roads in the map, to further improve the accuracy.

At the end of this design, we discuss the impact of $\psi_{i,0}$. In PPCS, $\psi_{i,0}$ determines the weight of a original vector in the encrypted vector. On one hand, $\psi_{i,0}$ cannot be too small. When $\psi_{i,0} \to 0$, the weight of $X_{(i)}$ in the encrypted $\mathbb{X}_{(i)}$ is too small, which will result in a poor recovery accuracy. On the other hand, $\psi_{i,0}$ cannot be too large. When $\psi_{i,0} \to 1$, $X_{(i)} = \mathbb{X}_{(i)}$, which losses the effect of encryption. Empirically, we find that setting $\psi_{i,0}$ in the range $[0.2, 0.8]$ can guarantee a high recovery accuracy and privacy. The other weights still satisfy $\psi_{i,j}|_{j\neq 0} \in (0,1)$ and $\sum_{j=0}^{K}\psi_{i,j} = 1$.

V. PPCS Analysis

In this section, we analyze the performance of PPCS in three metrics: the trajectory recovery accuracy, the privacy preservation against eavesdroppers, and the privacy preservation against stalkers. Its complexity analysis is also presented.

A. Accuracy Analysis

Although recovering trajectory by CS has been shown to achieve a promising accuracy [29], we still need to make sure the KVP encryption operation does not degrades the accuracy.

We adopt the same metric in [29] to evaluate the recovery accuracy, namely, the *recovery error* ϵ. For user i, its recovery error $\epsilon_{(i)}$ is the geometric mean of the distance between the actual trajectory and the recovered trajectory, defined as

$$\epsilon_{(i)} = \frac{||X_{(i)} - \hat{X}_{(i)}||_2}{T}, \tag{10}$$

where $||X_{(i)} - \hat{X}_{(i)}||_2 = \sqrt{\sum_{j=1}^{T}(x_{ij} - \hat{x}_{ij})^2}$, and T is the total number of time slots along the trajectory.

With this accuracy metric, we have the following theorem stating that the KVP encryption operation does not degrade the recovery accuracy.

Theorem 5.1: The proposed KVP is a homomorphic obfuscation method for CS. We define the homomorphic obfuscation property as follows. If a matrix X is near low-rank, the recovery accuracy of a user i satisfies

$$\sup||X_{(i)} - \hat{X}_{(i)}||_2 = \sup||X_{(i)} - \tilde{X}_{(i)}||_2, \tag{11}$$

where sup is the upper bound of $||\cdot||_2$, $\hat{X}$ is the trajectories recovered by CS with KVP (*i.e.*, $\hat{X} = f_{de}(f_{cs}(f_{en}(X \circ \Phi)))$), $\tilde{X}$ is trajectories recovered by CS directly (*i.e.*, $\tilde{X} = f_{cs}(X \circ \Phi)$), and $\hat{X}_{(i)}$ is the recovered trajectory of user i.

Proof: When a matrix is near low-rank and the value of approximate rank is r, the value $\sum_{i=1}^{min(N,T)}\sigma_i - \sum_{i=1}^{r}\sigma_i$ can be considered as noise [6], which is denoted as ξ.

According to existing work on the CS-based matrix completion [6] [14], we have the accuracy upper bound as

$$\sup||X - \tilde{X}||_2 = 4\sqrt{\frac{2\min(N,T)}{(1-\alpha)}}\xi_1, \tag{12}$$

where α is the data loss ratio in X, and ξ_1 is the noise of X.

Similarly, the accuracy upper bound of $||\mathbb{X}||_2$ can be represented as

$$\sup ||\mathbb{X} - \hat{\mathbb{X}}||_2 = 4\sqrt{\frac{2\min(N,T)}{(1-\alpha)}}\xi_2, \tag{13}$$

where ξ_2 is the noise of $\mathbb{X}$.

From Fig. 5, we know that the KVP operation does not change the number of missing data. Consequently, the loss ratio α in Eq. (12) and in Eq. (13) has the same value. Combining Eq. (12) and Eq. (13), we have

$$\frac{\sup ||\mathbb{X} - \hat{\mathbb{X}}||_2}{\sup ||X - \tilde{X}||_2} = \frac{\xi_2}{\xi_1}. \tag{14}$$

It is difficult to obtain the exact value of ξ_1 and ξ_2, which highly depends on the specific data. However, because KVP is a basic linear transformation, which can be presented as $\mathbb{X} = \Psi X$ and Ψ is the matrix of private keys ψ. Treating this transformation as a measurement operation in CS, we can obtain the noise ratio according to CS theory [6],

$$\frac{\xi_2}{\xi_1} = \frac{|\mu(\Phi, \Psi)|}{|\mu(\Phi, I)|}. \tag{15}$$

Recall that Φ is the binary index matrix indicating the missing data. The coherence operation μ in Eq. (15) is defined as

$$\mu(\Phi, I) = \max_{1 \le i \ne j \le T} | < \Phi^{(i)}, I^{(j)} > |. \tag{16}$$

where $\Phi^{(i)}$ is the i-th column vector of $\Phi_{N\times T}$, and $< \Phi^{(i)}, I^{(j)} >$ is the inner product of two vectors, *i.e.*, $< \Phi^{(i)}, I^{(j)} >= (\Phi^{(i)})' I^{(j)}$.

By the design of KVP, we have the Ψ matrix as follows, which is an example when $K = 2$ as shown in Fig. 5

$$\Psi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \psi_{3,1} & \psi_{3,2} & \psi_{3,0} & 0 & 0 \\ \vdots & \vdots & 0 & \ddots & 0 \\ \psi_{N,1} & \psi_{N,2} & 0 & 0 & \psi_{N,0} \end{bmatrix}. \tag{17}$$

Combine Eq. (14), Eq. (15), Eq. (16), and Eq. (17), we can calculate the recovery error of user i as

$$\frac{\sup ||\mathbb{X}_{(i)} - \hat{\mathbb{X}}_{(i)}||_2}{\sup ||X_{(i)} - \tilde{X}_{(i)}||_2} = \frac{|\mu(\Phi, \Psi^{(i)})|}{|\mu(\Phi, I^{(i)})|} = \frac{\psi_{i,0}}{1}. \tag{18}$$

Because of the reasons that 1) the decryption operation $f_{de}(\mathbb{X}_{(i)}) = \hat{X}_{(i)}$ is also a linear transformation according to Eq. (9); 2) all other variables such as Ds and ψs are known; and 3) since $\sum_{j=0}^{K} \psi_{i,j} = 1$, we know the error is linearly amplified according to weights

$$\frac{\sup ||X_{(i)} - \hat{X}_{(i)}||_2}{\sup ||\mathbb{X}_{(i)} - \hat{\mathbb{X}}_{(i)}||_2} = \frac{\psi_{i,0} + \psi_{i,1} + \cdots + \psi_{i,p}}{\psi_{i,0}} = \frac{1}{\psi_{i,0}}.$$

Combining the above two equations, we have

$$\sup ||X_{(i)} - \hat{X}_{(i)}||_2 = \sup ||X_{(i)} - \tilde{X}_{(i)}||_2, \tag{19}$$

and the theorem is proved. ■

B. Privacy Preservation against Eavesdroppers

Privacy preservation is offered by PPCS. We discuss how PPCS protects privacy leakage against eavesdroppers (in this subsection) and stalkers (in the next subsection).

The location data are encrypted by individual users before transmitting them to the server. In this way, only encrypted data (the encrypted sensed trajectories sent from the users $\mathbb{S}$, or the complete encrypted trajectories $\hat{\mathbb{X}}$ recovered by CS) can be captured by eavesdroppers. These eavesdroppers can only infer the original user trajectory based on the exposed encrypted data $\hat{\mathbb{X}}$. Therefore, we adopt the *distortion* δ defined in [40] to measure the similarity between the encrypted and the original data of every user

$$\delta_{(i)} = \frac{\sum_{j=1}^{T} |\hat{\mathbb{X}}_{(i,j)} - X_{(i,j)}|}{T}. \tag{20}$$

The value of δ presents the average per-location distortion between the encrypted and the original trajectories, and a larger δ indicates a stronger privacy preservation against eavesdroppers. In practice, the complete trajectory X is not always available due to the missing data issue. In this case, we adopt the recovered $\hat{X}$ to replace X in Eq. (20) for computing.

The PPCS scheme exploits KVP to obfuscate the user's personal trajectory. Since several trajectories are perturbed into one trajectory, even if an eavesdropper steals this combined trajectory, it is not easy to distinguish the original one. In the next, we derive the distribution of the distortion δ.

The encrypted vector is obtained via linearly combining K public vectors with weights ψs, and this encryption operation demonstrates significant randomness in that 1) the K public vectors are randomly selected from all public vectors and 2) the weight vector $<\psi_{i,0}, \psi_{i,1}, \cdots, \psi_{i,K}>$ is randomly generated. With these randomness, the original locations are mapped to other locations but still in the area of interests. As a result, we can use the random distance distribution to approximate the distortion of a given location and its encrypted data.

Consider a $w \times h$ rectangle area, the distortion distribution $\mathbb{P}(\delta \le d)$ can be presented by a piecewise function [5]

$$\mathbb{P}(\delta \le d) = \begin{cases} \frac{2}{w^2h^2}(G(d) - G(0)) & d \in [0, h] \\ \frac{2}{w^2h^2}(G(h) - G(0)) & d \in (h, w] \\ \frac{2}{w^2h^2}(G(h) - G(\sqrt{d^2 - w^2})) & \\ +F_h(\sqrt{d^2 - w^2}) & d \in (d, \eta] \end{cases}, \tag{21}$$

where

$$\begin{aligned} G(z) &= \int (h - z)\sqrt{d^2 - z^2}(2w - \sqrt{d^2 - z^2})\mathrm{d}z, \\ F_h(z) &= 1 - (1 - z/h)^2, \text{ and } \eta = \sqrt{w^2 + h^2}. \end{aligned}$$

With this distribution, the average distance between randomly selected points (*i.e.*, the expectation of distortion $\bar{\delta}$) can be easily obtained.

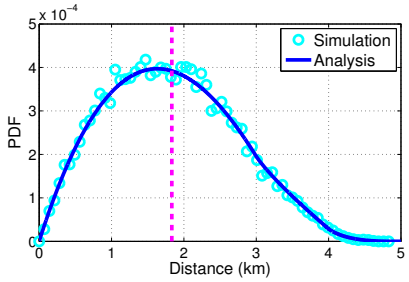


Fig. 6. PDF of the distortion by KVP.

To valid our analysis on the distortion distribution, we simulate in a rectangle area with $w = 4$ km and $h = 3$ km, and randomly generate a set of trajectories including a total number of 1×10^6 locations. Then we apply KVP on these trajectories and record the distances between the original locations and their corresponding encrypted locations. The statistic distribution of these distances is shown in Fig. 6, along with the probability distribution calculated according to Eq. (21). The average distance of these location pairs is also shown in the figure (*i.e.*, $\bar{\delta} \approx 1.83$ km).

Two observations are obtained from these results. First, the distortion shows significant randomness over a large distance range. Second, the average distortion between the original and encrypted locations is relatively large compared to the area. These observations verify that there is no obvious pattern to infer the original locations through the encrypted locations.

C. Privacy Preservation against Stalkers

Another adversary model is the stalker, who can obtain k actual location data of a user's trajectory. A stalker has two alternative methods to recover the trajectory: 1) Crack the private key based on the k data. 2) Run the crowdsensing recovery using these k data.

In the first method, to protect the user privacy, it is required that the trajectory $\hat{X}_{(i)}$ in Eq. (9) is unsolvable, even if the stalkers know the encrypted data $\hat{\mathbb{X}}_{(i)}$, the decryption function f_{de}, and k ($k < K$) actual location data of $X_{(i)}$. The PPCS scheme resorts to the *private keys* against stalkers. From Eq. (9), we know

$$\hat{\mathbb{X}}_{(i)} = \psi_{i,0}\hat{X}_{(i)} + \psi_{i,1}D_{(1)} + \cdots + \psi_{i,K}D_{(K)}. \tag{22}$$

It is possible for a stalker to obtain $\hat{\mathbb{X}}_{(i)}$ by hacking the server, and she may also obtain the public vectors D_i ($i = 1, 2, \cdots, K$). However, because the private keys $\psi_{i,j}$ ($j = 0, 1, \cdots, K$) are only known by the users themselves, a stalker resolving Eq. (22) needs the knowledge of at least $K + 1$ elements of $X_{(i)}$ according to the theory of underdetermined system [8]. As a result, the stalker cannot resolve the original trajectory as long as the condition $k \leq K$ holds. As K is the control parameter adopted in PPCS, we can proactively adjust the number of public vectors used in the encryption operation according to the requirement of individual users.

In the second method, a stalk pretends herself as a private user and joins in the crowdsensing recovery. Many practical factors significantly affects the privacy-preserving such as the number of exposed location data k, the mobility model of users, the map structure, and etc. Some of these practical factors are not easy to be formulated. Hence, for this method, we conduct real trace based simulations to verify the privacy preservation in practice (refer to Section. VI-B).

D. Complexity Analysis

1) Computational Complexity: The KVP operation is locally run at user side. In KVP, $K + 1$ vectors with size $1 \times T$ need to be processed in encryption and decryption steps, which requires a computational complexity of $\mathcal{O}((K + 1)T)$. This complexity costs negligible computing time owe to the capability of current GHz-level mobile devices.

At the server side, the main task is CS computing, which requires a computational complexity of $\mathcal{O}(rNT\varrho)$ [29], where r is the rank of the to-be-recovered matrix and ϱ is the iteration numbers. Our evaluation experiences with Beijing and Shanghai traces reveal that $\varrho \leq 5$ in most cases. Since the server always has a strong computational capability, the CS operation is responsive in real-time.

2) Communication Overhead: In order to execute f_{en}, a user should download K public vectors $D_{(i)}$ from the server and then upload a encrypted vector $\mathbb{S}_{(i)}$. Hence, the communication overhead is $\mathcal{O}((K+1)T)$. Moreover, in order to execute f_{de}, the user should download $\hat{\mathbb{X}}_{(i)}$, requiring another communication overhead of $\mathcal{O}(T)$. As an example, with $K = 10$, $T = 500$ and a 16-bit operating system, the total amount of data exchange is about $(10+2) \times 500 \times 16/8 = 12$ KB, which is a very light overhead for modern mobile applications.

E. Design Discussion

At the end of the analysis, we discuss the interesting design concern: *no public users*.

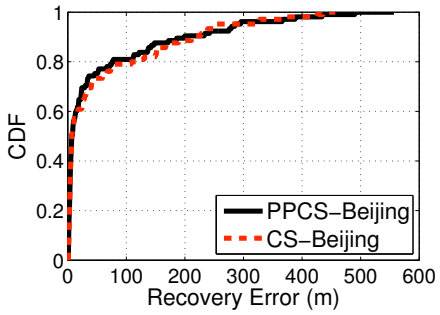
Public users are optional in PPCS. Even there is no public users, PPCS still works. To replace the roles of public vectors, the server can provide historical vectors (*e.g.*, any trajectory with the same time interval yesterday), as long as the low-rank property is maintained. In addition, when a user has a high privacy requirement, she will demand a large K public vectors according to the analysis in Section V-C. This inadequate K problem could also be solved by historical vectors.

VI. Performance Evaluation

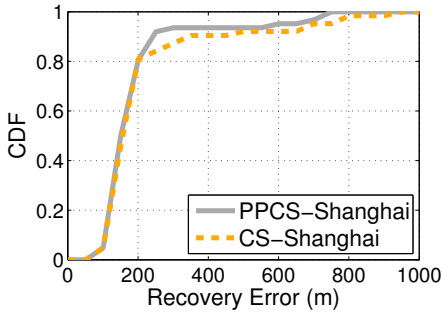
In this section, we evaluate the performance of PPCS in terms of both the data accuracy and the privacy.

A. Simulation Settings

We evaluate PPCS based on two real-world traces including walk, bike, and car data in Geolife [2], and taxi and bus data in SUVnet [3]. Using the same method in Section III, we preprocess the raw data of Geolife and SUVnet by selecting complete trajectories as our ground truth to conduct our simulations. The selected traces are named Beijing traces with a size of 116 users $\times$ 355 slots and Shanghai traces with a size of 74 users $\times$ 399 slots, whose detailed descriptions are listed in Table I.

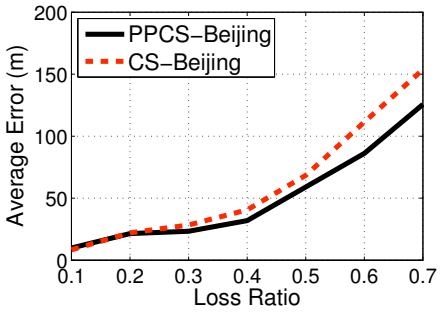


(a) CDF of recovery error in Beijing traces

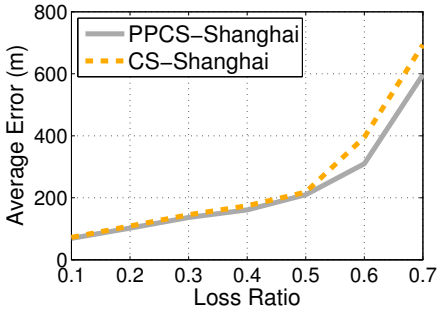


(b) CDF of recovery error in Shanghai traces

Fig. 7. Recovery accuracy comparison.

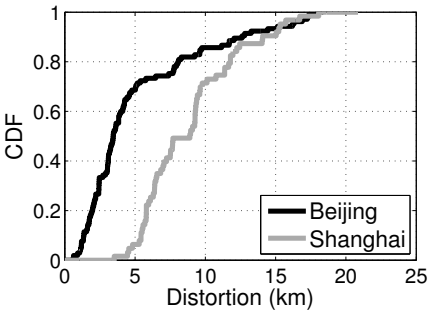


(a) Recovery error vs loss ratio in Beijing traces

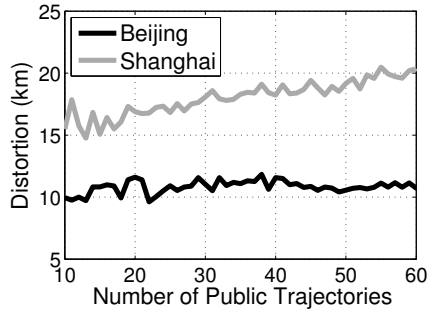


(b) Recovery error vs loss ratio in Shanghai traces

Fig. 8. Recovery accuracy vs loss ratio.



(a) Distortion distribution



(b) Distortion vs K

Fig. 9. Distortion against eavesdroppers.

In the trace-driven simulations, we randomly generate a 0-1 matrix Φ with the same size as the original data trace. The element in Φ takes the value of 0 if its corresponding element in the data trace is missing and 1 otherwise. The ratio of the 0 elements to the total number of elements in Φ is controlled by the data loss ratio α, which is set 0.5 by default unless otherwise specified. Then, we generate the sensed matrix S according to Eq. (2), *i.e.*, $S = X \circ \Phi$. The proposed PPCS is applied on the sensed matrix S with K public vectors, and the recovered matrix $\hat{X}$ is obtained. Without loss of generality, the top-K rows in the original traces are treated as the public traces, and $K = 10$ by default. The reported results in the following are averaged over 100 simulation runs.

We adopt the state-of-the-art CS-based crowdsensing recovery method in [29] as a baseline, which is referred as CS in the remaining of this section. With respect to the adversary model, we consider that an eavesdropper can steal all encrypted data $\hat{\mathbb{X}}$. And the stalker has a part ($< 10\%$) of real trajectory data.

B. Performance Analysis

1) Recovery Accuracy: We first evaluate the recovery accuracy with a default setting of $\alpha = 0.5$ and $K = 10$. The distributions of the recovery accuracies obtained by PPCS and CS with the two real-world traces are shown in Fig. 7. For example, Fig. 7(a) shows that the recovery errors of 50% users' trajectories are less than 10 meters and the average recovery error is 53 m when applying PPCS on Beijing traces. Two observations are obtained from Fig. 7. First, the recovery accuracy obtained with PPCS and CS are comparable in both traces, which validates the correctness of Theorem 5.1 that PPCS can achieve similar recovery accuracy as the state-of-the-art CS method. Second, the recovery errors are small. For instance, the recovery errors of 80% users with Beijing and Shanghai are less than 100 m and 200 m, respectively. These errors are tolerable in many cases because mechanisms such as map matching [25, 33] can eliminate their impacts on the final recovered trajectories.

To gain more insights on the impact of data loss in recovery accuracy, we apply PPCS on the two traces with varying α from 0.1 to 0.7, and the results are shown in Fig. 8. A clear increasing trend of recovery errors with the increase of α can be observed. For example, with Beijing traces, PPCS achieves an average recovery error of 20 m when $\alpha = 0.2$, and the error is increased to 124 m with an $\alpha = 0.7$. The recovery errors with PPCS and CS are comparable in all the explored cases, which agrees with the observation in Fig. 7. An interesting observation is that PPCS slightly improves the accuracy performance of CS. A possible reason is that the linear transformations in KVP make the low-rank property of the trajectory matrix even more obvious, and thus improves the recovery accuracy.

2) Privacy against Eavesdroppers: Keeping $\alpha = 0.5$ and $K = 10$, next we investigate the perturbation distortion obtained with PPCS. In Fig. 9(a), we can see that the distortion between original and encrypted trajectories is enormous. For example, the distortion of 50% trajectories are over 4,000 m and the average distortion is more than 9,000 m with Beijing traces. Such distortion distances are quite large when compared with the road segment length in Beijing city. As a result, even if the encrypted trajectory is exposed to adversaries, the information leakage on the original trajectory

TABLE II
RECOVERY ERROR WHEN STALKERS HAVE PARTIAL ORIGINAL DATA.

Recovery error ϵ	Stalker (5%)	Stalker (10%)	User (50%)
ϵ of Beijing	409.36 m	366.47 m	38.99 m
ϵ of Shanghai	2510.48 m	1723.16 m	189.96 m

is small, indicating a strong privacy preservation level. Another observation is that the distortion distribution shows no clear patterns. For example, the distortion distribution with Shanghai traces is nearly linear, but that with Beijing traces is more like a piecewise function. This patternless feature indicates that even the adversaries can obtain a large amount of the encrypted trajectories, the training methods based on these information would not facilitate them to infer the original trajectories.

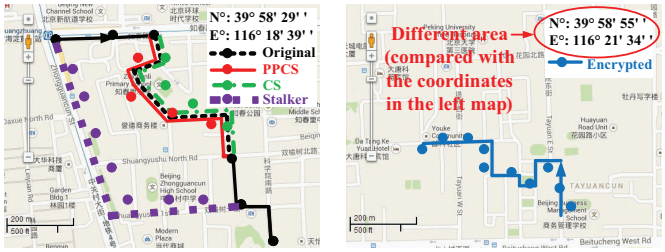
As analysis in Section V, PPCS needs a number of public (or historical) trajectories to perform the encryption. To investigate the impact of the amount of public trajectories on the distortion, we apply PPCS on the two traces with the number of public traces varying from 10 to 60 (the total number of users in Beijing and Shanghai are 116 and 74 respectively). The results in Fig. 9(b) demonstrate that there is no clear relation between the distortion and the number of available public traces. This observation alleviates our concern on whether the available number of public trajectories will significantly degrade the distortion performance of PPCS. So the privacy-preserving level in PPCS is independent to the number of public vectors.

3) Privacy against Stalkers: A stalker can treat the exposed k data of a user as a $(T-k)$ missing data trajectory and then utilizes PPCS to recover this trajectory. Next, we evaluate the privacy preservation offered by PPCS against stalkers, and the results are shown in Table II. Recall that k is a small number in the stalker model. We set and evaluate the cases when 5% and 10% actual location data are captured by the stalks. In the case of 5%, the recovery error is more than $2,500$ m with Shanghai traces and more than 400 m with Beijing traces. It is difficult to obtain the actual trajectory with such large errors. Even if a stalker has 10% actual trajectories, she cannot achieve a promising recovery accuracy. On the contrary, when the data loss ratio is $\alpha = 0.5$, the private users under PPCS has an excellent accuracy that always under 40 m with Beijing traces. In summary, our PPCS solution is able to effectively protect the privacy even when a few original data are exposed.

C. Illustrative Results

To demonstrate a clear view of the results obtained with PPCS, we show the recovered trajectory by PPCS/CS/Stalker and the encrypted trajectory against eavesdropper in Fig. 10, using a 10-location original trajectory. All trajectories are fitted to roads by the map match method proposed in [33].

In Fig. 10(a), the recovered trajectories by PPCS and CS [29] are drawn when 4 locations along the trajectory are missing. We can see when 40% of original data are missing, PPCS still recovers the original trajectory with a high accuracy that is comparable to the result of CS. Moreover, Fig. 10(a) also shows the recovered trajectory of a stalker who applies PPCS with 3 stalked actual locations, which is a totally different trajectory compared with the original one.



(a) The recovered trajectories by PPCS and CS are similar to the original one, but the trajectory recovered by a stalker is much different.

(b) The encrypted trajectory distorts the original one to a different area (refer to the latitude and the longitude) against eavesdroppers.

Fig. 10. Illustrative results of PPCS. (The dots are the PPCS / CS / Stalker / Encrypted results. The lines are the map matching results based on the dots.)

The encrypted trajectory is shown in Fig. 10(b). Comparing the latitude N° and the longitude E° in Fig. 10(b) with those in Fig. 10(a), we find that the distortion between the encrypted trajectory and the original one is relatively large, indicating a strong defense against eavesdroppers. Furthermore, the encrypted results also form a sound trajectory in the map. This indicates that the eavesdroppers cannot easily determine whether the hacked trajectories are encrypted or not.

In addition, even an adversary can eavesdrop and stalk simultaneously, she can only obtain two separate results: 'encrypted' as shown in Fig. 10(b) and 'stalker' in 10(a), but no further improvement on inferring the original trajectory.

VII. RELATED WORK

In this section, we discuss the related work in literature. There are two important research topics involved in this work: trajectory recovery and trajectory privacy.

A. Trajectory recovery

We classify existing efforts on trajectory recovery into two categories: *single user recovery* and *crowdsensing recovery*.

The single user recovery is to reconstruct a trajectory based on a user's own location data. Plenty of classic missing data estimation methods such as nearest neighbors (NN) [34] and linear interpolation [31] can be utilized to recover a trajectory in a user's own mobile device. These methods avoid the data leakage issue because no data exchange is required; however, their recovery accuracy is usually limited [29].

The crowdsensing recovery is to reconstruct all users' trajectories together based on their trajectory correlations, and thus significantly improves the recovery accuracy when compared with the single user recovery. Currently, compressive sensing (CS) [9] is an advanced recovery technique in diverse applications [20, 39]. For trajectory recovery, CS-based crowdsensing recovery [29] also produces the near-optimal approximation for missing data recovery. Although CS provides high accuracy, it requires data transmission and a computing server, and thus degrades user privacy.

B. Trajectory privacy

Existing trajectory privacy works have three primary methods: *anonymization*, *dummification*, and *obfuscation*. First, a

user adopting anonymization method [24] is to transmit her location data attached with an anonymity instead of her ID. However, latest studies [12, 38] reveal that the anonymization mechanism alone is inadequate to preserve the privacy well. Second, a user adopting dummification method [19] is to transmit her location data with a set of generated fake data. Although the dummification increases the privacy, it introduces additional data and influences the original correlations, which decreases the recovery accuracy. Third, the obfuscation method either perturbs a user's location data by mixing other trajectories [15, 27] or cloaks the data into a spatial region [13]. Existing obfuscation methods blur the original data, and thus contradict with the consideration of the accurate recovery.

VIII. Conclusion

With the increasing popularity of location based services, it is important to simultaneously consider the quality of service and user privacy. Focus on the trajectory recovery service, in this paper, we design a novel PPCS scheme using crowdsensing to accurately recover the trajectories with the consideration of privacy. The core design of PPCS leverages the matrix transformation to include the privacy preservation into compressive sensing. Through extensive trace-based simulations, we demonstrate that PPCS not only effectively preserves the user privacy against eavesdroppers and stalkers, but also accomplishes comparable accuracy as the state-of-the-art CS design. Although we focus on the trajectory recovery in this work, the general PPCS can also be utilized in other privacy-preserving data recovery applications.

Acknowledgment: This research was supported in part by the NSERC Discovery Grant 341823, NSERC Collaborative Research and Development Grant CRDPJ418713, Canada Foundation for Innovation (CFI) Leaders Opportunity Fund 23090, NSFC grant No. 61373155, No. 91438121, No. 61303202, and China Postdoctoral Science Foundation grant No. 2014M560334.

References

[1] Facebook acquires company behind Moves fitness app. http://www.theverge.com/2014/4/24/5647084/facebook-acquires-moves-fitness-app.

[2] GeoLife Data Collected by Microsoft Research Asia. http://research.microsoft.com/en-us/projects/geolife/default.aspx.

[3] SUVnet Data Collected by Shanghai Jiao Tong University. http://wirelesslab.sjtu.edu.cn/download.html.

[4] Trippermap service in flickr. http://www.flickr.com/services/apps/5121/.

[5] V. S. Alagar. The distribution of the distance between random points. *Journal of Applied Probability*, pages 558–566, 1976.

[6] E. J. Candes and Y. Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

[7] C.-Y. Chow, M. F. Mokbel, and W. G. Aref. Casper: Query processing for location services without compromising privacy. *ACM Transactions on Database Systems*, 34(4):24–48, 2009.

[8] J. W. Demmel and N. J. Higham. Improved error bounds for underdetermined system solvers. *SIAM Journal on Matrix Analysis and Applications*, 14(1):1–14, 1993.

[9] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[10] Z. Feng, Y. Zhu, Q. Zhang, L. M. Ni, and A. V. Vasilakos. Trac: Truthful auction for location-aware collaborative sensing in mobile crowdsourcing. In *IEEE INFOCOM*, 2014.

[11] R. K. Ganti, F. Ye, and H. Lei. Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine*, 49(11):32–39, 2011.

[12] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan. Private queries in location based services: anonymizers are not necessary. In *ACM SIGMOD*, 2008.

[13] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *ACM MobiSys*, 2003.

[14] C. Hegde, P. Indyk, and L. Schmidt. Approximation-tolerant model-based compressive sensing. In *ACM/SIAM SODA*, 2014.

[15] B. Hoh and M. Gruteser. Protecting location privacy through path confusion. In *IEEE SecureComm*, 2005.

[16] X. Ju, H. Zhang, and D. Sakamuri. Neteye: a user-centered wireless sensor network testbed for high-fidelity, robust experimentation. *International Journal of Communication Systems*, 25(9):1213–1229, 2012.

[17] J. Jun, Y. Gu, L. Cheng, B. Lu, J. Sun, T. Zhu, and J. Niu. Social-loc: Improving indoor localization with social sensing. In *ACM SenSys*, 2013.

[18] T. Jung and X.-Y. Li. Search me if you can: privacy-preserving location query service. In *IEEE INFOCOM*, 2013.

[19] H. Kido, Y. Yanagisawa, and T. Satoh. An anonymous communication technique using dummies for location-based services. In *IEEE ICPS*, 2005.

[20] L. Kong, M. Xia, X.-Y. Liu, M.-Y. Wu, and X. Liu. Data loss and reconstruction in sensor networks. In *IEEE INFOCOM*, 2013.

[21] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft. Structural analysis of network traffic flows. In *ACM SIGMETRICS*, 2004.

[22] J. Liu, B. Priyantha, T. Hart, H. S. Ramos, A. A. Loureiro, and Q. Wang. Energy efficient GPS sensing with cloud offloading. In *ACM SenSys*, 2012.

[23] S. Liu, S. Wang, K. Jayarajah, A. Misra, and R. Krishnan. Todmis: Mining communities from trajectories. In *ACM CIKM*, 2013.

[24] C. Y. Ma, D. K. Yau, N. K. Yip, and N. S. Rao. Privacy vulnerability of published anonymous mobility traces. In *ACM MOBICOM*, 2010.

[25] P. Newson and J. Krumm. Hidden markov map matching through noise and sparseness. In *ACM SIGSPATIAL*, 2009.

[26] D. Niculescu and B. Nath. Trajectory based forwarding and its applications. In *ACM MOBICOM*, 2003.

[27] D. Quercia, I. Leontiadis, L. McNamara, C. Mascolo, and J. Crowcroft. Spotme if you can: Randomized responses for location obfuscation on mobile phones. In *IEEE ICDCS*, 2011.

[28] K. K. Rachuri, C. Mascolo, M. Musolesi, and P. J. Rentfrow. Sociablesense: exploring the trade-offs of adaptive sampling and computation offloading for social sensing. In *ACM MOBICOM*, 2011.

[29] S. Rallapalli, L. Qiu, Y. Zhang, and Y.-C. Chen. Exploiting temporal stability and low-rank structure for localization in mobile networks. In *ACM MOBICOM*, 2010.

[30] R. Rosales and S. Sclaroff. 3D trajectory recovery for tracking multiple objects and trajectory guided recognition of actions. In *IEEE CVPR*, 1999.

[31] G. Scaglia, A. Rosales, L. Quintero, V. Mut, and R. Agarwal. A linear-interpolation-based controller design for trajectory tracking of mobile robots. *Elsevier Control Engineering Practice*, 18(3):318–329, 2010.

[32] I. Singh, M. Butkiewicz, H. V. Madhyastha, S. V. Krishnamurthy, and S. Addepalli. Twitsper: Tweeting privately. *IEEE Security & Privacy*, 11(3):46–50, 2013.

[33] A. Thiagarajan, L. Ravindranath, H. Balakrishnan, S. Madden, L. Girod, et al. Accurate, low-energy trajectory mapping for mobile devices. In *USENIX NSDI*, 2011.

[34] W. K. Wong, D. W.-l. Cheung, B. Kao, and N. Mamoulis. Secure kNN computation on encrypted databases. In *ACM SIGMOD*, 2009.

[35] W. Xi, J. Zhao, X.-Y. Li, K. Zhao, S. Tang, X. Liu, and Z. Jiang. Electronic frog eye: Counting crowd using wifi. In *IEEE INFOCOM*, 2014.

[36] M. Xia, L. Gong, Y. Lv, Z. Qi, and X. Liu. Effective real-time android application auditing. In *IEEE S&P*, 2015.

[37] T. Xu and Y. Cai. Feeling-based location privacy protection for location-based services. In *ACM CCS*, 2009.

[38] H. Zang and J. Bolot. Anonymization of location data does not work: A large-scale measurement study. In *ACM MOBICOM*, 2011.

[39] B. Zhang, X. Cheng, N. Zhang, Y. Cui, Y. Li, and Q. Liang. Sparse target counting and localization in sensor networks based on compressive sensing. In *IEEE INFOCOM*, 2011.

[40] J. Zhu, K.-H. Kim, P. Mohapatra, and P. Congdon. An adaptive privacy-preserving scheme for location tracking of a mobile user. In *IEEE SECON*, 2013.