Location Privacy-Preserving Data Recovery for Mobile Crowdsensing

TONGQING ZHOU, National University of Defense Technology, China and The Hong Kong Polytechnic University, China
ZHIPING CAI*, National University of Defense Technology, China
BIN XIAO, The Hong Kong Polytechnic University, China
LEYE WANG, Hong Kong University of Science and Technology, China
MING XU† and YUEYUE CHEN, National University of Defense Technology, China

Data recovery techniques such as compressive sensing are commonly used in mobile crowdsensing (MCS) applications to infer the information of unsensed regions based on data from nearby participants. However, the participants' locations are exposed when they report geo-tagged data to an application server. While there are considerable location protection approaches for MCS, they fail to maintain the correlation of sensory data, leading to the existence of unrecoverable data. None of the previous approaches can achieve both data recovery and data privacy preservation. We propose a novel location privacy-preserving data recovery method in this paper. Based on our discovery that the adjacency relations of non-zero elements are key to the missing data recovery in a crowdsensing data matrix, we design a correlation-preserving location obfuscation scheme to hide the participants' locations under effective camouflage. We also design an encrypted data recovery scheme based on the homomorphic encryption in order to avoid location privacy leakage from sensory data. Location obfuscation and data encryption preserve the participants' privacy, while the correlation-preserving and homomorphic properties of our method ensure data recovery accuracy. Evaluations of real-world datasets show that our privacy-preserving method can effectively obfuscate locations (e.g., yielding an average location distortion of $1.7km$ in a $2.4km \times 4km$ area for successful location hiding), and it can efficiently achieve similar data recovery accuracy to compressive sensing (which has no privacy protection).

CCS Concepts: • **Networks** → *Network privacy and anonymity*; Cloud computing; • **Human-centered computing** → **Ubiquitous and mobile computing**;

Additional Key Words and Phrases: mobile crowdsensing, location privacy, data recovery, compressive sensing

*This is the corresponding author
†This is the corresponding author

Authors' addresses: Tongqing Zhou, National University of Defense Technology, College of Computer, Changsha, China, The Hong Kong Polytechnic University, Department of Computing, Hong Kong, China, zhoutongqing@nudt.edu.cn; Zhiping Cai, National University of Defense Technology, College of Computer, Changsha, China, zpcai@nudt.edu.cn; Bin Xiao, The Hong Kong Polytechnic University, Department of Computing, Hong Kong, China, csbxiao@comp.polyu.edu.hk; Leye Wang, Hong Kong University of Science and Technology, Department of Computer Science and Engineering, Hong Kong, China, wangleye@gmail.com; Ming Xu, xuming@nudt.edu.cn; Yueyue Chen, yueyuechen@nudt.edu.cn, National University of Defense Technology, College of Computer, Changsha, China.

2474-9567/2018/9-ART151 $15.00
https://doi.org/10.1145/3264961

ACM Reference Format:
Tongqing Zhou, Zhiping Cai, Bin Xiao, Leye Wang, Ming Xu, and Yueyue Chen. 2018. Location Privacy-Preserving Data Recovery for Mobile Crowdsensing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3, Article 151 (September 2018), 23 pages. https://doi.org/10.1145/3264961

1 INTRODUCTION

The surge of sensor-rich mobile devices and the demand for pervasive sensing have led to a new sensing paradigm, known as *mobile crowdsensing (MCS)* [14]. MCS can acquire various types of environmental information (e.g., air quality [10][25], spectrum [5], noise pollution [43]) by exploiting mobile users as sensors. Due to the limited budgets of application campaigners and the large size of sensing areas, there are usually many unsensed regions whose data are unknown [34][35]. For example, participants are not always available at every block of an urban area (unlike traditional specialized sensors in wireless sensor networks), so the collected ambient noise data based on MCS tend to be very sparse [43]. In order to deal with this data sparsity (missing) problem and to construct a comprehensive sensing map, data recovery techniques are usually adopted to infer the missing data of blank regions based on reported data from nearby locations [20][38][42]. Fig. 1 (Top) illustrates a typical process of data recovery-involved MCS applications (a.k.a., compressive crowdsensing [34][38] and Sparse MCS [35]). The process consists of a sensory matrix construction phase and a data recovery phase based on compressive sensing (CS)[1].

Participants' location privacy is at risk when reporting sensory data tagged with their actual locations to the application server for data recovery [15][16]. In fact, participants would be reluctant to join a task without appropriate privacy protection, leading to a failure of such a sensing paradigm. Considerable work has been devoted to location privacy preservation for MCS [27]. We classify the existing approaches into three categories and illustrate them in Fig. 1 (Bottom). Specifically, the cloaking-based approaches use a coarse polygon area to replace a participant's precise location [9][28]; the perturbation-based approaches hide a participant's location under an obfuscated location (the end of the arrow) with techniques like differential location privacy [18][30][33][36]; and the encryption-based approaches treat locations as data and encrypt both the location information (e.g., l_1) and the sensory data (e.g., S_1) for privacy preservation based on cryptographic mechanisms [1][19].

Though effective in protecting location privacy, these approaches breach the data correlation required by data recovery. As illustrated in Fig. 1 (Bottom), cloaking and encryption would destroy the indexing role of locations, rendering the inputs (not matrices) invalid for CS-based data recovery. While perturbation maintains a matrix with the same amount of data as the input, the deduction results of the missing elements based on the perturbed matrix would be substantially deviated. For example, the 1st element in the 1st row of the sensory matrix ought to be deduced from S_1 and S_2 according to CS, but it turns out to be inferred based on S_2, S_4, and S_5 after perturbation. Therefore, it is still necessary to design a practical location privacy-preserving scheme that guarantees accurate data recovery.

In this paper, we propose a novel location privacy-preserving method, named *Spatial Camouflage Participants (SCP)*, to simultaneously conceal participants' actual locations and guarantee data recovery accuracy in MCS applications.

The building block of SCP is a *conjecture*: the inherent correlation that CS exploits for data recovery is the horizontal and vertical adjacency relations of non-zero elements in the sensory matrix. Intuitively, the data recovery results should be the same for those matrices that preserves this correlation. Based on this insight, we design a correlation-preserving location obfuscation scheme to protect participants' locations

[1]We consider compressive sensing as the data recovery technique in this paper, which we will explain in Section 3.1.2. Hence, CS and data recovery will be used interchangeably hereafter.

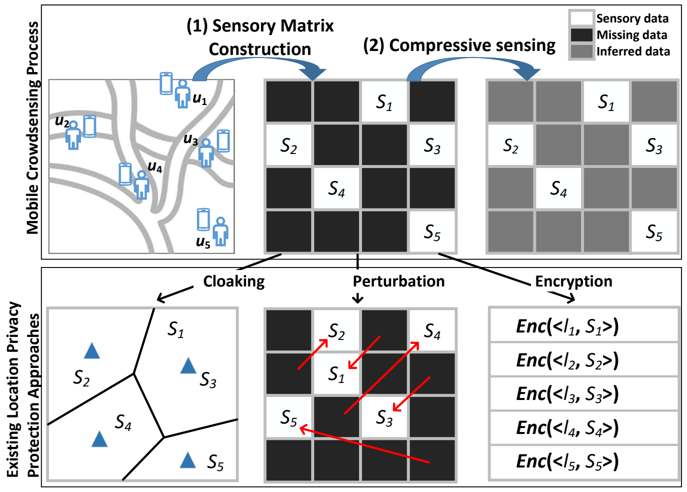


Fig. 1. A brief MCS process using CS for data recovery (Top) and some typical location privacy protection approaches (Bottom). Briefly, CS infers the unknown elements (e.g., the 1st element in the 1st row) based on the values of some known elements (e.g., S_1, S_2). Throughout the paper, a participant's location is modeled as the grid in which he is located. The grid is then mapped to an index of a sensory matrix (e.g., index $(1,3)$ represents S_1 as the location of u_1).

while still preserving their adjacency relations. Namely, for a sensory matrix in which elements are sensory data and each element is indexed with the sensing location of its data, we migrate the elements of one row (column) to another row (column) simultaneously. Furthermore, in order to perform location obfuscation in a distributed way, a blind signature is adopted in order to securely assign the constructed obfuscation to each participant. The accuracy of our obfuscation scheme is proven in theory and has been evaluated in experiments.

Participants' locations are not safe merely using location obfuscation. We note that the sensory data, formed of attributes of corresponding locations, can be used as quasi-identifiers to infer the locations of the participants. We define this privacy leakage threat as an inference attack. Specifically, given a sensory matrix in which each participant's location (i.e., index) is formally obfuscated, one can easily identify a participant's actual location as the index of its data in the recovered matrix (publicly available). In view of this threat, we design a data recovery scheme based on homomorphic encryption for participants to encrypt their data before reporting. Homomorphic encryption is widely used in privacy-preserving data aggregation. We use additively homomorphic encryption (AHE) and vector homomorphic encryption (VHE) to conduct secure computation for CS. AHE is first used to secure the report and distribution of participants' sensory data. Then we decompose the CS process into 7 atomic vector operations and leverage VHE to encrypt the sensory matrix and perform these atomic operations on the ciphertexts.

Based on SCP, participants can report to the MCS application server securely with obfuscated locations and encrypted data. Data recovery can be performed based on the secured reports with no accuracy loss. The main contributions of this paper to the field can be summarized as follows:

- To the best of our knowledge, our work is the first attempt to protect participants' location privacy against passive monitoring and inference attacks, while maintaining data recovery accuracy.
- We make a conjecture on the inherent data correlation used in CS and theoretically prove its correctness. From this conjecture, we propose a correlation-preserving location obfuscation scheme.

The scheme obfuscates participants' locations in a secure and distributed way by using blind signature.
- We propose a homomorphic encryption-based data recovery scheme to avoid location privacy leakage from the sensory data. The scheme leverages homomorphic vector encryption to perform CS in several atomic operations.
- We prove the security of our SCP method in a well-defined security model and conduct extensive experiments to evaluate its performance on real-world datasets. The results show that SCP can yield effective location obfuscation (e.g., average location distortion of $1.7km$ for a $2.4km \times 4km$ area). SCP can also achieve similar data recovery accuracy to CS, while it is much better than the baseline privacy-preserving compressive sensing (PPCS) [19].

2 RELATED WORK

We focus on the location privacy-preserving methods for data recovery in MCS. As shown above, locations can be leaked indirectly with inference attacks on sensory data. Hence, both location privacy and data security are relevant to our work. We will now review some previous work in these two areas.

2.1 Location Privacy-Preserving Approaches

Location privacy has been studied extensively because of the popularity of location-based services (LBS) [39]. We classify recent efforts on location privacy into three categories: cloaking, perturbation, and encryption.

As a widely used strategy, cloaking techniques aim to hide a precise location under a coarse area using either spatial transformation or a set of dummy locations, e.g., [9][28]. Although it does enable anonymity-based privacy, the cloaking strategy is criticized to be sensitive to adversaries' prior knowledge [33].

To avoid this issue, a rigorous method of indistinguishability [2][3] has recently been adopted in location perturbation in order to protect location privacy in LBS (i.e., differential location privacy). It involves obfuscating a participant's location in such a way that the calculated probabilities for its actual location to be one of several locations are similar in terms of the *Bayesian attack* [2]. Such techniques shed light on the design of privacy-preserving MCS tasks. Specifically, researchers in [18][30][33] focused on minimizing participants' overall travelling distance after having obfuscated their locations with differential location privacy in the participant recruitment stage. In [32], how to design an optimal location obfuscation policy for crowd coverage maximization under certain differential privacy protection was investigated. Wang *et al.* [36] tried to reduce the data inference error when adopting differential location privacy.

Some schemes treat locations as data and propose to protect location privacy with data encryption techniques [1][19]. In [1], partially homomorphic encryption was used jointly with a novel localization structure in order to perform passive localization without leaking observers' locations. Note that we also adopt homomorphic encryption in SCP (Phase III), but for the privacy preservation of sensory data against inference attacks. We protect location privacy using our correlation-preserving obfuscation scheme (Phase II). Kong *et al.* [19] encrypted locations in a trajectory (represented as a vector) by adding several public vectors to the trajectory vector before having performed trajectory recovery in the server.

We must highlight that these approaches are not effective for data recovery-involved MCS as they ignore the impact of location protection on data recovery. First, the cloaking and encryption-based approaches can destroy the location indices and provide invalid inputs (i.e., not matrices) for CS. Second, the existing perturbation-based approaches breach the inherent correlation required by CS, yielding inaccurate data recovery results. In contrast, our SCP method exploits and retains the inherent correlation used in CS to obfuscate participants' locations with no loss of accuracy for data recovery.

2.2 Data Privacy-Preserving Approaches

Data privacy/security has received considerable attention in many fields, specially database management [12] and MCS [8]. The former category generally focuses on privacy-preserving data publishing based on techniques such as generalization and suppression (i.e., replacing the values of a specific description with a less specific description [12]). In MCS, on the other hand, privacy issues are usually considered together with data aggregation based on either data perturbation or cryptographic mechanisms.

Data perturbation involves adding noises with a known distribution to the original data while allowing the computation of community statistics (e.g., the average aggregate) by using certain rules [13][40]. The differential privacy technique modifies user data by adding controlled noise (e.g., drawn from a Laplace distribution), which is believed to have a negligible effect on the query outcome. Based on differential privacy, a privacy-preserving statistics publishing framework for crowdsourced data was proposed in [37].

Applying cryptographic mechanisms in privacy-preserving data aggregation is straightforward. Homomorphic encryption is a widely adopted cryptographic technique. It allows certain computations over ciphertexts to be done in such a way that the decryption of the encrypted result matches the result of computations over plaintexts. In [45], BGV homomorphic encryption was employed to support several basic aggregation statistics (e.g., mean, variance) on encrypted data. AHE was introduced to protect sensory data privacy and supported the sum aggregate in [21] and the expectation maximum-based truth discovery in [23].

The problem settings (i.e., the aggregation process) discussed in existing work are different from those of data recovery, which aims to infer missing sensory data using CS. Both [26] and [31] have investigated privacy-preserving CS. Fully homomorphic encryption (FHE) was adopted in [26] to represent any computation via a garbled combinational circuit and then evaluate it using encrypted input. Wang *et al.* [31] criticized this FHE-based solution as being impractical and designed a problem transformation mechanism to reconstruct compressed images with privacy protection. Unlike images stored with the users, sensory matrices are constructed with participants' data in the server. Thus, data privacy would be exposed to the server if we encrypted sensory matrices based on the technique in [31].

In our SCP method, we decompose the CS process into a series of vector operations and leverage the vector homomorphic encryption in [44] to design an efficient encrypted data recovery scheme. The sensory data of participants are then protected to avoid location leakage from inference attacks.

3 PROBLEM STATEMENT

3.1 System Model: Mobile Crowdsensing

MCS tasks, especially in the urban environment, often have difficulty in getting sufficient samples due to the limited budgets [35]. For example, in a noise monitoring task, we are only expected to collect measurements that are sparsely distributed in the target area from participants [43]. To achieve full coverage of the target area (e.g., noise estimation for every block), data recovery techniques are usually adopted to infer the missing data of the overlooked regions in MCS. Below, we introduce the typical MCS system, including its data structure and the CS-based data recovery technique that it uses.

3.1.1 Data Structure. During an MCS task, the target area is divided into uniform grids with h rows and v columns. Sensory data are then collected for certain grids and are used to deduce the situation of the whole area (see Fig. 1 (Top)). Both the scale and the elements in the sensory matrix are not fixed in dynamic MCS scenarios, so sensing tasks should be conducted periodically to update the sensory matrix.

Notation: We use boldface to denote matrices (e.g., $\mathbf{A}$) and blackboard bold to denote sets and vectors (e.g., $\mathbb{L}$). We let $\mathbf{A}_i$ and $\mathbf{A}_j$ denote the i-th row and j-th column of $\mathbf{A}$, respectively. We use superscript $*$ to denote a location-obfuscated version of a vector or a matrix (e.g., $\mathbf{A}^*$). We use $\mathbb{Z}$ for the integer space.

Definition 3.1 (Environment Matrix). Environment matrix $\mathbf{G}$ is an $\mathrm{h} \times \mathrm{v}$ matrix with each element being the sensory data of a specific grid. $\mathbf{G}$ has no missing data.

Definition 3.2 (Index Matrix). Index matrix $\mathbf{I}$ is an $\mathrm{h} \times \mathrm{v}$ matrix, which indicates whether an element is missing in matrix $\mathbf{G}$ (i.e., $\mathbf{I}(i,j)$=0 if $\mathbf{G}(i,j)$ is missing, otherwise $\mathbf{I}(i,j)$=1).

Definition 3.3 (Sensory Report). The sensory report of participant u_i ($i \in [1,n]$) is a tuple $\langle l_i^1, l_i^2, S_i \rangle$, where $l_i^1 \in \mathbb{L}_1$=[2, ..., h+1] and $l_i^2 \in \mathbb{L}_2$=[2, ..., v+1] are the row index and column index of the grid in which u_i is located, and S_i denotes the sensory data. We will explain later why we number the indices from 2.

Definition 3.4 (Sensory Matrix). Sensory matrix $\mathbf{S}$ is an $\mathrm{h} \times \mathrm{v}$ matrix consisting of the sensory data of all participants with $\mathbf{S}_{l_i^1, l_i^2} = S_i$. $\mathbf{S}$ can be represented by the element-wise production of $\mathbf{G}$ and $\mathbf{I}$, namely, $\mathbf{S}$=$\mathbf{G} \odot \mathbf{I}$.

Definition 3.5 (Recovered Matrix). Recovered Matrix $\widehat{\mathbf{G}}$ is an $\mathrm{h} \times \mathrm{v}$ matrix constructed by deducing the missing values in an $\mathbf{S}$ to approximate $\mathbf{G}$ based on data recovery techniques.

3.1.2 Compressive Sensing-Based Data Recovery. We perform data recovery based on CS, which is proven to outperform (i.e., is more accurate than) most state-of-the-art data recovery methods (e.g., Singular Spectrum Analysis, K-Nearest Neighbors, Delaunay Triangulation) [20][41]. We define the CS operation as f_{cs}, thus $\widehat{\mathbf{G}} = \mathrm{f}_{cs}(\mathbf{S})$. f_{cs} is carried out based on the low-rank property:

$$\begin{aligned} &\min \ rank(\widehat{\mathbf{G}}) \\ &s.t., \ \widehat{\mathbf{G}} \odot \mathbf{I} = \mathbf{S} \end{aligned} \tag{1}$$

However, it is difficult to solve this problem because it is non-convex. To relieve this pitfall, singular value decomposition (i.e., $\widehat{\mathbf{G}} = \mathbf{U} \cdot \mathbf{V}$) is often adopted to rewrite rank minimization as minimizing the sum of the decomposition matrices' Frobenius norms:

$$\min_{\mathbf{U},\mathbf{V}} \ \lambda \cdot (\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2) + \|(\mathbf{U} \cdot \mathbf{V}) \odot \mathbf{I}\text{-}\mathbf{S}\|_F^2 \tag{2}$$

which can be further represented as:

$$\min_{\mathbf{U},\mathbf{V}} \ \lambda \cdot (\sum_{i \in [h]} \|\mathbf{U}_i\|_F^2 + \sum_{j \in [v]} \|\mathbf{V}_j\|_F^2) + \sum_{(i,j) \in \mathbb{W}} (\langle \mathbf{U}_i, \mathbf{V}_j \rangle \text{-} \mathbf{S}_{ij})^2 \tag{3}$$

where λ is a Lagrange multiplier that allows a tunable tradeoff between rank minimization and accuracy fitness, and vector $\mathbb{W}$ consists of the indices of the non-zero elements in $\mathbf{I}$. To solve this optimization problem, we use a gradient descent procedure [26] to estimate the two matrices iteratively:

$$\begin{aligned} \mathbf{U}_i(t{+}1) &= \mathbf{U}_i(t)\text{-}\gamma(\sum_{(i,j) \in \mathbb{W}} \mathbf{V}_j(t) \cdot \mathbf{E}_{ij}(t) + \lambda \cdot \mathbf{U}_i(t)) \\ \mathbf{V}_j(t{+}1) &= \mathbf{V}_j(t)\text{-}\gamma(\sum_{(i,j) \in \mathbb{W}} \mathbf{U}_i(t) \cdot \mathbf{E}_{ij}(t) + \lambda \cdot \mathbf{V}_j(t)) \end{aligned} \tag{4}$$

where $\mathbf{E}_{ij}(t)$=$\langle \mathbf{U}_i(t), \mathbf{V}_j(t) \rangle$-$\mathbf{S}_{ij}$ denotes the approximation error for each sensory measurement, and $\gamma > 0$ is a small gain factor. $\mathbf{U}(0)$ and $\mathbf{V}(0)$ consists of uniformly random elements.

3.2 Threat Model

Our security guarantees will hold under the *honest-but-curious* threat model. That is, all the involved parties (i.e., the participants, the application server and the Crypt-Service Provider (CSP)) will perform honestly according to the protocols; however, they may analyze the protocol and try to learn additional information, especially the participants' locations. Such privacy leakage may expose the participants to unexpected surveillance and tracking [16]. We consider two forms of threat models:

- **Passive Monitoring**: For a specific participant, unauthorized parties (including the server, the CSP, and other participants) could monitor the traffic of the participant to collect location related information during an MCS task. In brief, if a participant reports data with its actual location, then its privacy is at risk. Thus, participants' locations should be reported securely.
  However, even with obfuscation, such threats are still severe considering several deliberately designed attacks based on passive monitoring. With *Bayesian attacks*, adversaries can predict the posterior location of a participant based on prior knowledge of the actual location distribution and the location obfuscation probability. Similarly, *cryptographic analysis* enables the passive observers to predict participants' actual locations based on the correlation between the density of the actual and obfuscated locations. Finally, in *Sybil attacks*, an adversary can learn the obfuscation rules by manipulating several fake identities.
- **Inference Attacks**: An application server can infer the actual location of a participant by analyzing the data that it reports. Data is the attribute of a location; thus, it can be used as the quasi-identifier for one's location. Specifically, one's data would appear in both the original report (with obfuscated locations) and the final publication (with reordered locations). The server could easily find out one's actual location by matching the data in these two matrices. For example, if the server collects $\langle S_i, l^* \rangle$ in u_i's original report and finds data S_i at l_i from the final publication, then it can deduce that u_i's actual location is l_i. Hence, participants' data should also be reported secretly.

Collusion attacks are not considered in our threat model and we will discuss it in Section. 7.

Design Objectives: Mobile users would be reluctant to participate in an MCS task until the above threats are well mitigated. On the other hand, a sensing task would be uninformative if the mitigation of privacy leakage causes inaccurate data recovery. To this end, the objectives of this paper are twofold. First, participants' location privacy is protected against passive monitoring and inference attacks during data reporting and data analysis in MCS. Second, data recovery can be effectively performed based on the secured reports, with no loss of accuracy. As far as we are aware, this work is the first attempt to jointly consider these two objectives.

4 SPATIAL CAMOUFLAGE PARTICIPANTS METHOD

4.1 The Framework of SCP

To address the critical privacy concerns in MCS, we propose an efficient data recovery privacy-preserving method known as spatial camouflage participants (SCP). SCP involves three entities: the participants, the application server, and the crypt-service provider (CSP), and consists of three phases as shown in Fig. 2. In Phase I, the participants perform correlation-preserving location obfuscation to hide their actual locations under spatial camouflage constructed by the CSP. In Phase II, the server conducts secure two-party computation with the CSP for data recovery based on the obfuscated and encrypted sensory matrix, wherein sensory data is encrypted to avoid location leakage from inference attacks. Finally, in Phase III, the recovered data is extracted by decryption and location reordering in the CSP for further publishing.

We will now describe Phase I in Section 4.2, Phase II in Section 4.3, and Phase III in Section 4.4.

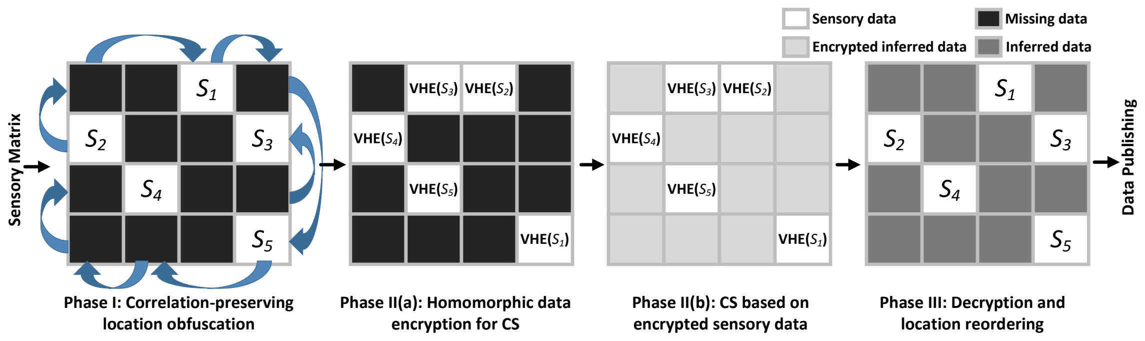

Sensory data
Missing data
Encrypted inferred data
Inferred data
Sensory Matrix
S_1 S_2 S_3 S_4 S_5
VHE(S_3) VHE(S_2) VHE(S_4) VHE(S_5) VHE(S_1)
Data Publishing
Phase I: Correlation-preserving location obfuscation
Phase II(a): Homomorphic data encryption for CS
Phase II(b): CS based on encrypted sensory data
Phase III: Decryption and location reordering

Fig. 2. Framework of our SCP method. Phase I is conducted between the participants and the CSP for location obfuscation. Phase II performs secure two-party computation for data recovery between the CSP and the application sever. The CSP extracts the recovered data through decryption and location reordering in Phase III.

4.2 Correlation-Preserving Location Obfuscation

Our location obfuscation scheme is designed based on a conjecture on the inherent data correlation of the sensory matrix. We will now describe our conjecture, then prove its correctness with a basic scheme, and finally present our detailed design following the basic scheme.

4.2.1 Conjecture on the Inherent Correlation. Existing location perturbation-based approaches (e.g., differential location privacy) perturb each participant's location separately. The non-zero elements in the sensory matrix are redistributed using these approaches. As we mentioned above, although the redistribution does not change either the number or the values of the elements, the deduced value of each missing element would be significantly deviated. It seems that some correlation is breached during their obfuscation operations.

We notice that, in the recovery process in Eq. 4, each missing piece of data is deduced using the non-zero elements in the corresponding row and column, which we refer to as the horizontal and vertical adjacent elements, respectively. Technically, the deduction result of a missing piece of data achieved by CS would not change as long as the missing piece of data is still in the same line as its horizontal and vertical adjacent elements. Thus, our conjecture is that the inherent correlation required by CS for data recovery is *the horizontal and vertical adjacency relations of non-zero elements in the sensory matrix*. As a result, the correlation is maintained if elements in the same row (column) are still found in the same row (column) after obfuscation; otherwise, the correlation is breached.

4.2.2 Basic Scheme for Correctness Analysis. We first introduce a basic obfuscation scheme in order to validate our conjecture. Our basic scheme assumes that sensory matrix $\mathbf{S}$ is available[2]. Motivated by the conjecture, the basic scheme obfuscates the participants' original locations (i.e., indices in $\mathbf{S}$) by moving the elements in one row (column) of $\mathbf{S}$ to another row (column) simultaneously. We notice that such obfuscation operations are the same as performing *elementary row and column transformation* on $\mathbf{S}$ (as shown in Phase I in Fig. 2). In this way, the basic location obfuscation process is defined by two elementary matrices $\mathbf{A}$ and $\mathbf{B}$, formalized as:

$$\mathbf{S}^{*}=\mathbf{A}\cdot\mathbf{S}\cdot\mathbf{B} \tag{5}$$

[2]We introduce this assumption here in order to clarify the basic idea and prove our conjecture. Our final obfuscation scheme can protect $\mathbf{S}$ from being revealed to each participant, the CSP, and the MCS server (see Sec. 4.2.3).

where $\mathbf{S}^*$ is the location-obfuscated version of $\mathbf{S}$ with participants' actual locations hidden. According to our conjecture, $\mathbf{S}^*$ can preserve the data correlation of sensory matrix $\mathbf{S}$ during data recovery. Thus, the recovered matrix (i.e., $\widehat{\mathbf{G}^*}$) using $\mathbf{S}^*$ should be equivalent to the result of performing elementary transformation on the recovered matrix (i.e., $\widehat{\mathbf{G}}$) using $\mathbf{S}$; in other words, our conjecture suggests $\widehat{\mathbf{G}}^*$=$\mathbf{A} \cdot \widehat{\mathbf{G}} \cdot \mathbf{B}$. We prove the accuracy of the basic scheme in Theorem 4.1.

Theorem 4.1. *The basic location obfuscation scheme of SCP (Eq. 5) does not affect the accuracy of CS-based data recovery.*

Proof. The server takes $\mathbf{S}^*$ as input, generates two decomposition matrices $\mathbf{U}^*$ and $\mathbf{V}^*$, and estimates $\mathbf{U}^*$ and $\mathbf{V}^*$ based on CS (Eq. 4).

$$\begin{aligned}
\mathrm{f}_{cs}(\mathbf{S}^*) = &\min\ (\lambda \cdot (\|\mathbf{U}^*\|_F^2 + \|\mathbf{V}^*\|_F^2) + \|\mathbf{U}^* \cdot \mathbf{V}^* \odot \mathbf{I}^* \text{-} \mathbf{S}^*\|_F^2) \\
\leftrightarrow &\min\ (\lambda \cdot (\|\mathbf{U}^*\|_F^2 + \|\mathbf{V}^*\|_F^2) + \|\mathbf{U}^* \cdot \mathbf{V}^* \odot \mathbf{I}^* \text{-} \mathbf{A} \cdot \mathbf{S} \cdot \mathbf{B}\|_F^2) \\
\leftrightarrow &\min\ (\lambda \cdot (\|\mathbf{A}^{-1} \cdot \mathbf{U}^*\|_F^2 + \|\mathbf{V}^* \cdot \mathbf{B}^{-1}\|_F^2) + \|\mathbf{A}^{-1} \cdot (\mathbf{U}^* \cdot \mathbf{V}^* \odot \mathbf{I}^* \text{-} \mathbf{A} \cdot \mathbf{S} \cdot \mathbf{B}) \cdot \mathbf{B}^{-1}\|_F^2) \\
\leftrightarrow &\min\ (\lambda \cdot (\|\mathbf{A}^{-1} \cdot \mathbf{U}^*\|_F^2 + \|\mathbf{V}^* \cdot \mathbf{B}^{-1}\|_F^2) + \|(\mathbf{A}^{-1} \cdot \mathbf{U}^* \cdot \mathbf{V}^* \cdot \mathbf{B}^{-1}) \odot (\mathbf{A}^{-1} \cdot \mathbf{I}^* \cdot \mathbf{B}^{-1}) \text{-} \mathbf{S}\|_F^2) \\
\leftrightarrow &\min\ (\lambda \cdot (\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2) + \|\mathbf{U} \cdot \mathbf{V} \odot \mathbf{I} \text{-} \mathbf{S}\|_F^2) \\
= &\ \mathrm{f}_{cs}(\mathbf{S})
\end{aligned} \tag{6}$$

where $\mathbf{U}$=$\mathbf{A}^{-1} \cdot \mathbf{U}^*$, $\mathbf{V}$=$\mathbf{V}^* \cdot \mathbf{B}^{-1}$, $\mathbf{I}$=$\mathbf{A}^{-1} \cdot \mathbf{I}^* \cdot \mathbf{B}^{-1}$, and $\|\mathbf{U}^*\|_F^2 = \|\mathbf{A}^{-1} \cdot \mathbf{U}^*\|_F^2$ as elementary transformation does not change the Frobenius norm of a matrix. By solving the above optimization problem with gradient descent, matrices $\mathbf{U}^*$ and $\mathbf{V}^*$ are iteratively adapted to their final values. Then we have:

$$\widehat{\mathbf{G}^*} = \mathbf{U}^* \cdot \mathbf{V}^* = \mathbf{A} \cdot \mathbf{U} \cdot \mathbf{V} \cdot \mathbf{B} = \mathbf{A} \cdot \widehat{\mathbf{G}} \cdot \mathbf{B} \tag{7}$$

Given the recovered matrix $\widehat{\mathbf{G}^*}$, we can easily extract matrix $\widehat{\mathbf{G}}$=$\mathbf{A}^{-1} \cdot \widehat{\mathbf{G}^*} \cdot \mathbf{B}^{-1}$ through an inverse transformation of the corresponding obfuscation operation (i.e., location reordering, see Section 4.4). Hence, we prove that our basic location obfuscation scheme in Eq. 5 does not affect the CS-based sensory matrix recovery results. □

The accuracy of this basic scheme validates our conjecture about the data correlation of CS-based data recovery (i.e., the adjacency relations of non-zero elements).

4.2.3 The Design of Our Location Obfuscation Scheme. Our basic scheme assumes that sensory matrix $\mathbf{S}$ is given during obfuscation. Meanwhile, in order to avoid location privacy leakage, all entities (the participants, the CSP, and the application server) are not supposed to acquire such an actual location-indexed matrix, as discussed in Section 3.2. Thus, the obfuscation operations of the basic scheme (i.e., elementary row and column transformation) should be performed by the participants distributively.

Simply, the participants can report their location indices to the CSP, which then assigns a same obfuscated location index for participants with the same location index (a.k.a., in the same row or column). The participants could also share their locations with each other to negotiate a correlation-preserving obfuscation. However, these naïve solutions expose participants' locations to the CSP or other peers. In contrast, in SCP's correlation-preserving location obfuscation phase (Fig. 3), the CSP constructs location obfuscation for all location indices and distributes the encrypted obfuscation to all participants. Each participant can only access the obfuscation of its own location index with an assigned token in this process. The following is a detailed description of the process:

- **Setup**. SCP first generates an RSA public key (e, N) and a private key (d, N), with N as the public modulus. The public key is sent to all participants. SCP also constructs two obfuscation vectors $\mathbb{L}_1^*$ and $\mathbb{L}_2^*$ by performing permutation encryption [22] (a typical transposition cipher) on $\mathbb{L}_1$ and $\mathbb{L}_2$. It is important to note that a permutation key is only used once per task. $\mathbb{L}_1^*$ and $\mathbb{L}_2^*$ refer to the obfuscated row index ℓ_1^* for $\ell_1 \in \mathbb{L}_1$ and column index ℓ_2^* for $\ell_2 \in \mathbb{L}_2$, respectively.
- **Step 1: Token Query**. Participant u_i chooses a random number which is prime to N, calculates a query $q_i{=}l_i^1 \cdot r^e \bmod N$ for its row index, and sends this blinded message to the CSP to request a token. Another query for its column index is also constructed (with a different r) and sent to the CSP separately[3].
- **Step 2: Token Answer**. On receiving a query q_i, the CSP calculates $a_i{=}(q_i)^d \bmod N{=}(l_i^1 \cdot r^e)^d \bmod N$ with its private key and sends a_i back to requester u_i.
- **Step 3: Token Calculation**. The participant removes the blinding factor r to get the token for its obfuscated index $\tau(l_i^x){=}a_i \cdot r^{-1} \bmod N{=}(l_i^1)^d \bmod N$.
- **Step 4: Obfuscation Distribution**. The CSP perturbs the location index vectors $\mathbb{L}_1^*$ and $\mathbb{L}_2^*$ with tokens, and constructs two encrypted obfuscation vectors $\widetilde{\mathbb{L}_1^*}{=}[\tau(\ell_1) \cdot \ell_1^*, \ell_1 \in \mathbb{L}_1]$ and $\widetilde{\mathbb{L}_2^*}{=}[\tau(\ell_2) \cdot \ell_2^*, \ell_2 \in \mathbb{L}_2]$ for the x-indices and y-indices. $\widetilde{\mathbb{L}_1^*}$ and $\widetilde{\mathbb{L}_2^*}$ are then distributed to all participants.
- **Step 5: Obfuscation Calculation**. Each participant uses the index token it acquires to extract the obfuscated index of its location: $(l_i^1)^*{=}\widetilde{\mathbb{L}_1^*}(l_i^1)/\tau(l_i^1){=}\tau(l_i^1) \cdot (l_i^1)^*/\tau(l_i^1)$. Note that the minimum value of location indices is set to be 2, as $\tau(\ell_1{=}1){=}1$ cannot protect the obfuscation $(\ell_1)^*$ from being revealed by other participants.

In fact, Steps 1-3 follow the *blind signature* process in [6], whereby a user can obtain a signature on a message from a signer without revealing the message to the signer. Here, the CSP is the signer that signs on the blinded location indices of each participant. Such a signature then acts as a token to access the corresponding obfuscated location index generated by the CSP. In this way, obfuscation information can be shared securely without revealing participants' actual locations. We will discuss the security of this process in Section. 5.1.

THEOREM 4.2. *Our correlation-preserving location obfuscation scheme can retain the data recovery accuracy of CS.*

PROOF. We can construct the elementary matrices **A** and **B** mentioned in the basic scheme in Section 4.2.2 based on $\mathbb{L}_1^*$ and $\mathbb{L}_2^*$ with:

$$\mathbf{A}_{ij}{=}\begin{cases} 1, & if\ \mathbb{L}_1^*(j){=}i \\ 0, & otherwise \end{cases} \quad i, j \in \mathbb{L}_1 \tag{8}$$

$$\mathbf{B}_{ij}{=}\begin{cases} 1, & if\ \mathbb{L}_2^*(i){=}j \\ 0, & otherwise \end{cases} \quad i, j \in \mathbb{L}_2 \tag{9}$$

Such a bijection from vector $\mathbb{L}_1^*$ and $\mathbb{L}_2^*$ to matrix **A** and **B** assures that participants with the same location row (column) index uniformly migrate their indices to a same row (column). According to *Theorem 3.1*, the horizontal and vertical adjacency relations can thus be preserved, which guarantees that our location obfuscation scheme will not impact the accuracy of CS-based data recovery. □

At the end of this phase, all participants acquire their obfuscated location indices (one for the row index, another for the column index), which are then used to replace the actual geo-tags in their reports.

[3]For simplicity, we only describe the obfuscation process for participants' location row indices, the same process applies to their location column indices.

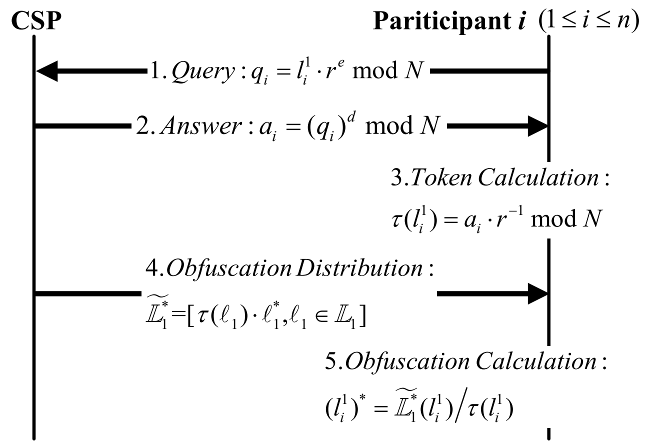

CSP
Pariticipant i $(1 \le i \le n)$
1. *Query* : $q_i = l_i^1 \cdot r^e \bmod N$
2. *Answer* : $a_i = (q_i)^d \bmod N$
3. *Token Calculation* :
$\tau(l_i^1) = a_i \cdot r^{-1} \bmod N$
4. *Obfuscation Distribution* :
$\widetilde{\mathbb{L}_1^*} = [\tau(\ell_1) \cdot \ell_1^*, \ell_1 \in \mathbb{L}_1]$
5. *Obfuscation Calculation* :
$(l_i^1)^* = \widetilde{\mathbb{L}_1^*}(l_i^1) / \tau(l_i^1)$

Fig. 3. Phase I: Correlation-preserving location obfuscation.

4.3 Encrypted Data Recovery

Even with obfuscation, a participant's location would still be exposed to potential adversaries via inference attacks, which use the participant's sensory data as quasi-identifiers for its location. In order to mitigate such attacks during data recovery, we design an encrypted data recovery scheme that is homomorphic to CS operations. Similar to location obfuscation, the data encryption should also be conducted on the participants' sides. As a result, existing homomorphic data encryption schemes for CS, such as PPCS in [19], become ineffective by assuming that participants in one row can use the same private key.

In contrast, SCP aims to encrypt sensory data at the participant's end and to perform secure two-party computation for CS-based data recovery between the application server and the CSP based on the collected encrypted data. In the following section, we will briefly introduce the adopted cryptographic primitives before going into the design details.

4.3.1 Cryptographic Primitives. Our scheme uses two cryptographic primitives: additively homomorphic encryption AHE($\cdot$) and integer vector homomorphic encryption VHE($\cdot$).

- AHE($\cdot$): We use *hash-ElGamal* [7] as AHE($\cdot$) because only a constant addition to the encrypted data is needed in our scheme, and *hash-ElGamal* can be implemented very efficiently. More details on hash-ElGamal can be found in Appendix A.
- VHE($\cdot$): As we will show later, the CS process involves a series of vector operations. Hence, in order to perform CS on the encrypted data efficiently, SCP refers to the *integer vector homomorphic encryption scheme* VHE($\cdot$) in [44]. Given an integer vector $\mathbb{X} \in \mathbb{Z}_p^m$, VHE($\cdot$) generates a matrix $\mathbf{\Psi} \in \mathbb{Z}_p^{m \times n}$ $(m < n)$ as the secret key, which satisfies:

$$\mathbf{\Psi}\mathbb{C} = \omega\mathbb{X} \tag{10}$$

with $\mathbb{C} \in \mathbb{Z}_p^n$ as the encrypted vector, and ω as an integer parameter. This symmetric encryption scheme can support the basic homomorphic operations that we need, including vector-vector addition ($\pm_v$), vectors' inner-production ($\times_v$), and vector-constant multiplication ($\times_c$), based on a key-switching technique. More details of VHE($\cdot$) are presented in Appendix B.

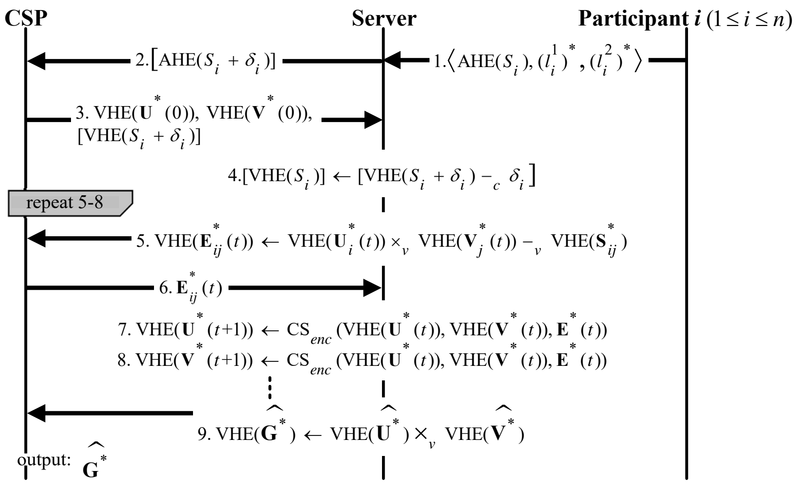

CSP
Server
Participant i $(1 \le i \le n)$
1.⟨AHE(S_i), $(l_i^1)^*$, $(l_i^2)^*$⟩
2.[AHE($S_i + \delta_i$)]
3. VHE($\mathbf{U}^*(0)$), VHE($\mathbf{V}^*(0)$), [VHE($S_i + \delta_i$)]
4.[VHE(S_i)] ← [VHE($S_i + \delta_i$) $-_c$ δ_i]
repeat 5-8
5. VHE($\mathbf{E}_{ij}^*(t)$) ← VHE($\mathbf{U}_i^*(t)$) $\times_v$ VHE($\mathbf{V}_j^*(t)$) $-_v$ VHE($\mathbf{S}_{ij}^*$)
6. $\mathbf{E}_{ij}^*(t)$
7. VHE($\mathbf{U}^*(t+1)$) ← CS$_{enc}$(VHE($\mathbf{U}^*(t)$), VHE($\mathbf{V}^*(t)$), $\mathbf{E}^*(t)$)
8. VHE($\mathbf{V}^*(t+1)$) ← CS$_{enc}$(VHE($\mathbf{U}^*(t)$), VHE($\mathbf{V}^*(t)$), $\mathbf{E}^*(t)$)
9. VHE($\widehat{\mathbf{G}^*}$) ← VHE($\widehat{\mathbf{U}^*}$) $\times_v$ VHE($\widehat{\mathbf{V}^*}$)
output: $\widehat{\mathbf{G}^*}$

Fig. 4. Phase II: CS-based data recovery on the location-obfuscated and data-encrypted sensory matrix.

4.3.2 Detailed Design. In this phase, the participants report location-obfuscated and AHE-encrypted data to the server (Step 1). Then the server perturbs (masks) the sensory data and transmits them to the CSP (Step 2), which decrypts the AHE and constructs VHE-encrypted vectors (Step 3) for later secure CS (Steps 4-9). As shown in Fig. 4, the detailed procedure is as follows:

- **Setup**. The CSP first generates its AHE public and private key pairs. The public key is sent to the server and the participants. Next, the two SVD decomposition matrices $\mathbf{U}^*$ and $\mathbf{V}^*$ are generated with uniformly random elements. Furthermore, the CSP constructs the VHE private key[4] $\mathbf{\Psi}_0$ for the encryption of $\mathbf{U}^*$ and $\mathbf{V}^*$ and the private key $\mathbf{\Psi}_1 = \text{vec}(\mathbf{\Psi}_0' \cdot \mathbf{\Psi}_0)^T$ for the encryption of received perturbed data, where vec($\mathbf{A}$) denotes the column vector consisting of all the elements in matrix $\mathbf{A}$.
- **Step 1-3: Prepare Sensory Data for CS.** Participant u_i encrypts sensory data S_i with the AHE public key and sends data tuple $\langle \text{AHE}(S_i), (l_i^1)^*, (l_i^2)^* \rangle$ to the server. On receiving the location-obfuscated and data-encrypted reports, the server generates random masks $(\delta_1, ..., \delta_n)$ for all sensory data and sends the masked sensory data vector $\mathbb{S}_p = [\text{AHE}(S_i + \delta_i), i \in [n]]$ to the CSP.
  Next, the CSP decrypts $\mathbb{S}_p$ with its AHE private key and obtains vector $[S_i + \delta_i, i \in [n]]$. It then computes the VHE for each masked data point using Eq. 10 under key $\mathbf{\Psi}_1$ to obtain n-vectors [VHE($S_i + \delta_i$)], and for $\mathbf{U}_i^*$ and $\mathbf{V}_j^*$ with $\mathbf{\Psi}_0$ to obtain VHE($\mathbf{U}^*(0)$) and VHE($\mathbf{V}^*(0)$)[5]. These VHE-encrypted ciphertexts are then sent to the server to initiate further computation.
- **Step 4: Remove the Masks.** Receiving n encrypted vectors of sensory data, the server removes the mask for each vector by performing a vector-constant addition operation VHE($S_i + \delta_i$)-$_c\delta_i$ to obtain VHE(S_i). It then constructs $\mathbf{S}^*$ with $\mathbf{S}^*_{(l_i^1)^*,(l_i^2)^*} = \text{VHE}(S_i)$ and also constructs a vector $\mathbb{W}^* = [((l_i^1)^*, (l_i^2)^*), i \in [n]]$, recording the obfuscated indices. This step will not trigger key-switching or key updating in the VHE protocol.
- **Step 5-8: Compressive Sensing on the Encrypted Data.** SCP aims to decompose the CS process of Eq. 4 into 7 atomic operations, which are homomorphic to the corresponding VHE

[4]Without loss of generality, we allocate the same private key to $\mathbf{U}_i$ and $\mathbf{V}_j$. One can easily extend the protocol by generating a different private key for each vector.
[5]$\mathbf{U}^*$ is encrypted by row, while $\mathbf{V}^*$ by column.

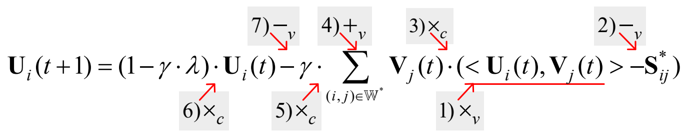

7) $-_v$
4) $+_v$
3) $\times_c$
2) $-_v$
$\mathbf{U}_i(t+1)=(1-\gamma\cdot\lambda)\cdot\mathbf{U}_i(t)-\gamma\cdot\sum_{(i,j)\in\mathbb{W}^*}\mathbf{V}_j(t)\cdot(<\mathbf{U}_i(t),\mathbf{V}_j(t)>-\mathbf{S}^*_{ij})$
6) $\times_c$
5) $\times_c$
1) $\times_v$

Fig. 5. Decomposing the CS iteration equation (Eq. 4) into 7 atomic vector operations. Each atomic operation between plaintexts is homomorphic to a basic VHE operation (in the shadowed blocks) between VHE encrypted data. For example, $\text{VHE}(\mathbf{U}_i) \times_v \text{VHE}(\mathbf{V}_j)$ is the homomorphic inner products operation for $\langle \mathbf{U}_i, \mathbf{V}_j \rangle$. We only present the decomposition of the updating function of $\mathbf{U}_i$, while $\mathbf{V}_j$ can be adapted in the same way.

operations on the VHE-encrypted data (see Fig. 5). During each iteration, for the i-th row of $\mathbf{U}^*$, the server first calculates the approximation errors of all elements in $\mathbf{S}^*_i$ based on matrices obtained from the previous iteration (we ignore the iteration number t for simplicity):

$$\text{VHE}(\mathbf{U}^*_i) \times_v \text{VHE}(\mathbf{V}^*_j)\text{-}_v\, \text{VHE}(\mathbf{S}^*_{ij}), (i,j) \in \mathbb{W}^* \tag{11}$$

First, in the vector multiplication operation, the CSP adjusts the secret key from $\mathbf{\Psi}_0$ to $\mathbf{\Psi}_1$ according to the protocol (i.e., key updating). Under key $\mathbf{\Psi}_1$, the subtraction operation gives the encrypted approximation errors $\text{VHE}(\mathbf{E}^*_{ij})$. In fact, a key-switching process is avoided here as we deliberately encrypt the sensory data with $\mathbf{\Psi}_1$, thus saving the transmission cost of the key-switching matrix. Then the server sends the approximation errors to the CSP for decryption (without their indices), and the CSP returns them in plaintext. In this way, #1, 2 in Fig. 5 are realized by Steps 5-6 in Fig. 4. Here, we decrypt the error matrix in order to perform the subsequent operations without key-switching.

The server proceeds to perform a series of constant-vector operations (# 3, 5, 6 in Fig. 5) and vector-vector addition operations (# 4, 7 in Fig. 5) based on $\text{VHE}(\mathbf{U}^*_i)$, $\text{VHE}(\mathbf{V}^*_j)$, and $\mathbf{E}^*_i$ (denoted as CS_{enc} in Fig. 4), which involve no key-switching or key updating and generate matrix $\text{VHE}(\mathbf{U}^*)$ for this iteration (Step 7). Similarly, we update each column of matrix $\text{VHE}(\mathbf{V}^*)$ (Step 8). Such an adaptation process stops until the predefined iterations (or certain stopping criteria) are achieved.

- **Step 9: Compute the Final Estimation**. Given the final version of the two SVD decomposition matrices $\text{VHE}(\widehat{\mathbf{U}^*})$ and $\text{VHE}(\widehat{\mathbf{V}^*})$, the server performs $\times_v$ operation between each row of $\text{VHE}(\widehat{\mathbf{U}^*})$ and each column of $\text{VHE}(\widehat{\mathbf{V}^*})$, and it obtains $\text{VHE}(\widehat{\mathbf{G}^*})$ with each element a VHE-encrypted vector. Estimation $\text{VHE}(\widehat{\mathbf{G}^*})$ is then sent to the CSP.

This phase results in an output of location-obfuscated and VHE-encrypted recovered matrix $\text{VHE}(\widehat{\mathbf{G}^*})$ on the CSP side.

4.4 Decryption and Location Reordering

After the encrypted data recovery process at the server, the CSP can extract the recovered matrix in plaintext by decryption and location index reordering. First, each element in $\text{VHE}(\widehat{\mathbf{G}^*})$ is decrypted using Eq. 10:

$$\widehat{\mathbf{G}^*_{ij}}=\lceil \mathbf{\Psi}_1 \cdot \text{VHE}(\widehat{\mathbf{G}^*_{ij}})/\omega \rfloor,\ i \in \mathbb{L}_1, j \in \mathbb{L}_2 \tag{12}$$

Then the CSP uses the elementary matrix $\mathbf{A}$ and $\mathbf{B}$ constructed in Eq. 8 and Eq. 9 to reorder the location indices and obtains the recovered matrix with $\widehat{\mathbf{G}}=\mathbf{A}^{-1} \cdot \widehat{\mathbf{G}^*} \cdot \mathbf{B}^{-1}$. Finally, matrix $\widehat{\mathbf{G}}$ is returned to the server for further data publishing. The decryption and reordering process can also be carried out at a specific and authenticated requester, which acquires key $\mathbf{\Psi}_1$, and matrix $\mathbf{A}$ and $\mathbf{B}$ from the CSP.

We point out that the abovementioned process is just one round of SCP. In the dynamic scenarios in which MCS tasks are performed periodically, SCP should also be conducted successively upon receiving an updated sensory matrix to protect participants' privacy in each task.

5 SECURITY ANALYSIS

In this section, we will present a security analysis of SCP's location obfuscation scheme and encrypted data recovery scheme in terms of the two threat models, passive monitoring and inference attacks, separately.

5.1 Privacy Preservation Against Passive Monitoring

Our SCP method protects participants' location privacy from passive monitoring through correlation-preserving location obfuscation. We have already discussed the accuracy of the obfuscation scheme in Section. 4.2.2. In the following section, we will analyze the security of the location obfuscation process and the encrypted data recovery process against passive monitoring. A discussion of how SCP mitigates several potential attacks is also presented.

Security During Obfuscation. The obfuscation process of SCP is conducted between each participant and the CSP. As mentioned, SCP follows the *blind signature* process to distribute the obfuscation information. In this way, our obfuscation scheme inherits the *blindness* and *unforgeability* property of blind signature [6]. The *blindness* guarantees that the CSP cannot know the actual location l_i^1 of participant u_i given query q_i, which means participants' actual locations are secure during location obfuscation. Meanwhile, the *unforgeability* assures that no one can construct valid tokens except the CSP (who keeps the secret key as a signer). Thus, u_i can only obtain the obfuscated location indices for its own row and column, which guarantees the security of the obfuscation information.

Security During Data Recovery. The server and the CSP conduct data recovery together. The participants replace their locations with the obfuscated ones in the reports to the server, so the server can only monitor obfuscated locations during data recovery (Step 1 in Fig. 4). The CSP, on the other hand, stores the actual-obfuscation mappings locally; however, it can only access the perturbed sensory data with no location tags (Step 2 in Fig. 4). Hence, the participants' location privacy is well protected against both the server's and the CSP's monitoring during data recovery.

Security Against Attacks. We analyze the location privacy-preserving capacity of SCP under *Bayesian attacks* [2], *cryptographic analysis* [11], and *Sybil attacks* [24], which are designed based on monitoring the transmissions.

In *Bayesian attacks*, an adversary is assumed to have prior knowledge about the distribution of mobile users' actual locations, denoted as $\pi(loc{=}(\ell_1, \ell_2))$ with ℓ_1 and ℓ_2 the row and column index, and it is supposed to know some information about the location obfuscation probability $P(loc^*|loc)$ [2][36]. Our SCP method proposes to hide a participant's actual location (i.e., the row and column indices of the corresponding grid) by obfuscating it to another grid in the h $\times$ v grids. The obfuscation of one's location row and column index are drawn from uniform distribution on $\mathbb{L}_1$ and $\mathbb{L}_2$ separately. Thus, the probability of transforming row ℓ_1 to each of the h rows is the same (i.e., $P(\ell_1^*|\ell_1){=}1/\text{h}$), and $P(\ell_2^*|\ell_2){=}1/\text{v}$ for column ℓ_2. Then SCP obfuscates location loc to each grid with the same probability $P(loc^*|loc){=}P(\ell_1^*|\ell_1) \cdot P(\ell_2^*|\ell_2){=}1/(\text{h} \cdot \text{v})$. As the obfuscation probability is indifferent to the actual location, an adversary can only deduce one's actual location based on prior knowledge of the distribution (i.e., calculating $P(loc|loc^*){=}\pi(loc)$), which means that SCP can preserve participants' location privacy under Bayesian attacks.

Furthermore, the permutation encryption technique that we use in location obfuscation is secure against *cryptographic analysis* [11], which tends to infer the decryption key of permutation by matching the known language statistics with the ciphertexts' statistics. Specifically, in our case, an obfuscated location with more participants than another location is considered to be from a more popular location. However, MCS usually recruits participants to cover the targeted area evenly[6]. Thus, all obfuscated locations would basically have the same number of participants, presenting no useful statistics for cryptographic analysis.

Finally, we discuss the security of our obfuscation scheme under *Sybil attacks* [24]. An adversary of the *Sybil attacks* could forge multiple identities and manipulate each identity to submit a different location query during SCP's location obfuscation phase. After having acquired the obfuscated location for each query from the CSP, the adversary could identify the actual obfuscated location mappings for the queried locations (i.e., the permutation key). This would possibly cause the location privacy leakage of the participants whose obfuscated locations are known. However, unique private/public key pairs can be assigned to each participant to securely communicate with the server and the CSP during the registration [45]. In this way, an adversarial participant is actually unable to access the others' obfuscated locations, making the deduction of one's actual location based on its obfuscation impossible. Hence, participants' location privacy is secure even under *Sybil attacks*.

5.2 Privacy Preservation Against Inference Attacks

SCP's CS-based encrypted data recovery phase protects participants' sensory data and mitigates inference attacks against their location privacy.

During the preparatory steps of this phase (Steps 1-3 in Fig. 4), the sensory data are AHE-encrypted under CSP's public key, so the server cannot reveal the received data; the CSP, on the other hand, can decrypt the AHE ciphertext with its private key, but it can only obtain data that is already masked by the server. In the following data recovery steps, homomorphic vector operations are conducted on the VHE-encrypted sensory data and decomposition matrices in the server (Steps 5,7,8 in Fig. 4). The CSP only receives intermediate approximation errors (Step 5 in Fig. 4) and the final encrypted estimation matrix (Step 9 in Fig. 4), which reveal nothing about the participants' raw sensory data.

When encrypting data with VHE, we ignore the noise term introduced in the original protocol (see Appendix B). This is because in our case, the CSP would not share (old) secret keys with other entities, so there is no need to introduce noise terms to prevent other entities from learning new secret keys based on the old ones (a.k.a., known plaintext attacks). Some may argue that the server can access both encrypted matrix VHE($\widehat{\mathbf{G}^*}$) and the recovered matrix $\widehat{\mathbf{G}}$, likely leading to known plaintext attacks. We emphasize that the server cannot map the ciphertexts to their corresponding plaintexts through index matching between these two matrices (i.e., by regarding VHE($\widehat{\mathbf{G}^*_{ij}}$) as the encryption for $\widehat{\mathbf{G}_{ij}}$) as the matrix has been transformed during obfuscation. Hence, publishing $\widehat{\mathbf{G}}$ would not result in data leakage or location exposure.

Overall, the CSP gets to know $\widehat{\mathbf{G}^*}$ and $\widehat{\mathbf{G}}$, and the server can acquire AHE($\mathbf{S}^*$) and $\widehat{\mathbf{G}}$, but none of them can deduce matrix $\mathbf{S}$ or $\mathbf{S}^*$ based on the available information at hand. Both locations and data privacy are well protected in our SCP method.

6 PERFORMANCE EVALUATION

6.1 Experimental Setup

6.1.1 Datasets. We evaluate our SCP method on two real-world datasets taken from *SensorScope* [17].

[6]Privacy preservation in MCS task assignment is not within the scope of this paper. One can refer to [33].

- **LUCE**: We use the ambient temperature and humid measurements of *SensorScope*'s LUCE deployment. The target area, the EPFL campus ($260m \times 400m$), is divided into 90 equal-sized grids ($29m \times 40m$) with 41 grids deployed with at least one sensor.
- **Glacier**: We use the ambient temperature and humid measurements of *SensorScope*'s Plaine Morte (a glacier) deployment. The target glacier ($2.4km \times 4km$) is divided into 64 equal-sized grids ($0.3km \times 0.5km$), of which 14 are deployed with at least one sensor.

As the environment matrix is not available, we use the assumptions postulated in [4] instead of validating the (approximate) low-rank property [29] to show the feasibility of performing CS-based data recovery on these datasets. Specifically, we divide the target areas as above to yield an *even data distribution*, and the recorded measurements contain *small data uncertainty* (i.e., there is little noise in the sampled entries, so the missing values can be inferred accurately). We emphasize that these two assumptions would remain satisfied in SCP. In other words, the correlation-preserving property of SCP can preserve the evenness of data distribution (i.e., data in the same row (column) of the sensory matrix are still found in the same row (column) after location obfuscation, and vice versa), while the encryption scheme introduces no noise to the sensory data to enlarge data uncertainty. For a grid with more than one sensor, we use the mean value of the sensors' measurements as the grid's data. During the evaluation, data about the sensed grids are used to deduce the missing elements of the environment matrix.

6.1.2 Baselines. We compare the *data recovery performance* of SCP with CS-based data recovery (CS) [29] and privacy-preserving compressive sensing (PPCS) [19][7]. Briefly, CS follows the process described in Section. 3.1.2; PPCS encrypts each row of a sensory matrix by adding several public vectors to it. PPCS's encryption is not effective in MCS, where the values of missing data in a row are all 0. Thus, in a row of encrypted data, elements with the same value are the ciphertexts of the missing data and the other elements are the encryption of the participants' data. We can then recover the (weighted) sensory data by subtracting from their ciphertexts the ciphertext of the missing data.

6.1.3 Metrics. To evaluate location obfuscation performance, we calculate the location distortion of an obfuscation as:

$$ld(i)=\sqrt{(((l_i^1)^*\text{-}l_i^1)\cdot g_x)^2+(((l_i^2)^*\text{-}l_i^2)\cdot g_y)^2} \tag{13}$$

where g_x and g_y are the row and column granularity of the grids. A larger ld indicates a stronger privacy-preserving capability against passive monitoring.

Furthermore, we evaluate the data recovery accuracy with two metrics named recovery difference ratio ($\mathbb{R}_d$) and recovery error ratio ($\mathbb{R}_e$). $\mathbb{R}_d$ describes the differences between the data recovered by ($\widehat{\mathbf{G}_{(\cdot)}}$) and the data recovered by CS ($\widehat{\mathbf{G}_{cs}}$):

$$\mathbb{R}_d^{(\cdot)}=\text{vec}(|\widehat{\mathbf{G}_{(\cdot)}}\text{-}\widehat{\mathbf{G}_{cs}}|\oslash\widehat{\mathbf{G}_{cs}}) \tag{14}$$

where $\widehat{\mathbf{G}_{(\cdot)}}$ consists of the recovered data based on method $(\cdot)$ (CS, SCP, or PPCS), and $\oslash$ denotes element-wise division. A small $\mathbb{R}_d^{(\cdot)}$ indicates that method $(\cdot)$'s recovery performance is similar to CS. On the other hand, we iteratively regard one element in the sensory matrix as a missing piece of data, and use method $(\cdot)$ to deduce its value based on the remaining sensory data. Then we calculate $\mathbb{R}_e^{(\cdot)}$ by comparing the recovered value with the true value:

$$\mathbb{R}_e^{(\cdot)}(i)=[|S_i\text{-}\widehat{\mathbf{G}_i}(l_i^1,l_i^2)|/S_i],\ i\in[n] \tag{15}$$

[7]Both CS and PPCS cannot protect location privacy. We adopt them merely for the comparison of data recovery accuracy.

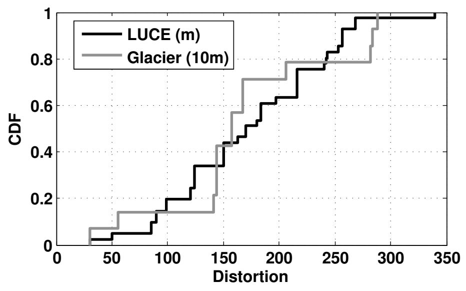


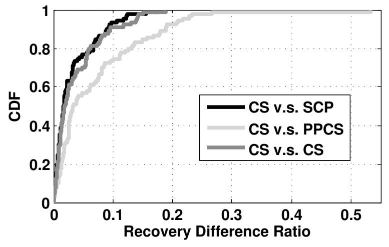
Fig. 6. CDF of location distortion with SCP. The distortion magnitude is $1m$ for LUCE , and $10m$ for Glacier.

where $\widehat{\mathbf{G}_i}$=$f_{(\cdot)}(\mathbf{S}_{l_i^1 l_i^2}$=0) denotes the recovered matrix based on the S_i-removed matrix $\mathbf{S}$. A large $\mathbb{R}_e^{(\cdot)}$ indicates that method $(\cdot)$ introduces a large recovery error.

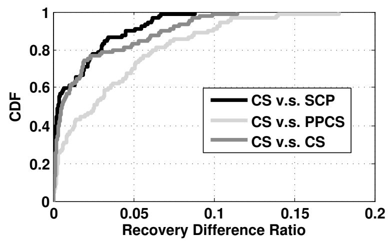
Finally, the data recovery iteration is set to 1000 for all three methods. We implement SCP and the baselines using Matlab, and we conduct our experiments on a workstation with 4GB memory and a 2.94GHz Intel(R) CPU.

6.2 Experimental Results

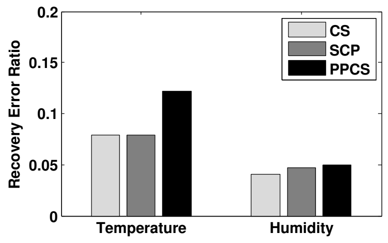
6.2.1 Location Obfuscation Performance. We first investigate the location distortion of SCP with Eq. 13 to evaluate its location obfuscation performance. As shown in Fig. 6, the distortions between the participants' original locations and the obfuscated locations are enormous. For example, 50% of the participants in LUCE obtain a distortion of more than $160m$, while the distortions for 50% participants in Glacier are more than $1.5km$. The average distortion for participants in LUCE and Glacier are $173m$ and $1.7km$, respectively. Compared to the size of the target areas ($260m \times 400m$ and $2.4km \times 4km$), the obtained distortions are relatively large. Meanwhile, the distortions are random (with no obvious pattern). Hence, our SCP can provide effective spatial camouflage for participants.



(a) Recovery difference for the temperature data.



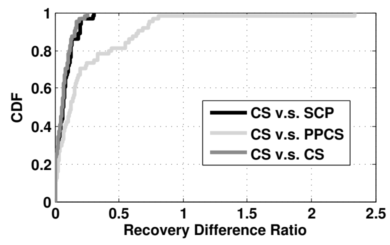
(b) Recovery difference for the humidity data.



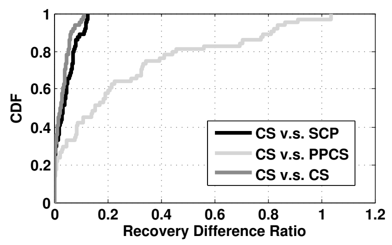
(c) Recovery error of CS, SCP, and PPCS.

Fig. 7. Recovery accuracy analysis of the LUCE dataset.

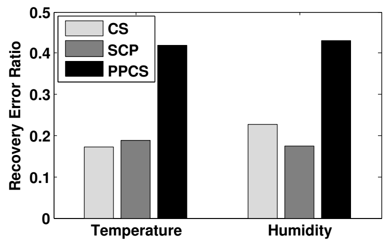
6.2.2 Data Recovery Accuracy. Next, we examine the data recovery performance of SCP by comparing it with CS and PPCS. We first compare the difference between the two in terms of recovery (i.e., Eq. 14) using two measurements and two datasets. Fig. 7(a) and Fig. 7(b) show the distributions of $\mathbb{R}_d^{scp}$ (CS v.s. SCP), $\mathbb{R}_d^{ppcs}$ (CS v.s. PPCS), and $\mathbb{R}_d^{cs}$ (CS v.s. CS) for temperature and humidity data in LUCE, where



(a) Recovery difference for the temperature data.



(b) Recovery difference for the humidity data.



(c) Recovery error of CS, SCP, and PPCS.

Fig. 8. Recovery accuracy analysis of the Glacier dataset.

$\mathbb{R}_d^{cs}$ represents the recovery difference ratio[8] between two runs of CS. As shown, distribution of $\mathbb{R}_d^{scp}$ is similar to distribution of $\mathbb{R}_d^{cs}$, which means SCP introduces no additional recovery error compared to CS. On the other hand, the PPCS method shows bigger recovery difference compared to the other two. We can observe a more obvious performance advantage of SCP over PPCS in the tests for Glacier (Fig. 8(a) and Fig. 8(b)). For example, in terms of the temperature data in Glacier, the biggest difference ratios of $\mathbb{R}_d^{scp}$ and $\mathbb{R}_d^{cs}$ are both around 0.25 for all the data points, while 20% of the data points' difference ratios exceed 0.5 (i.e., $\mathbb{R}_d^{ppcs} > 0.5$) under PPCS.

In order to gain more insights into recovery accuracy performance, we investigate the data recovery error (i.e., Eq. 15) of the three methods by iteratively regarding each sensory measurement as missing and inferring its value based on the other measurements. The results of the analyses of LUCE and Glacier are shown in Fig. 7(c) and Fig. 8(c), respectively. As expected, SCP can yield a similar recovery error to CS, and PPCS shows a much bigger recovery error than CS. For example, the recovery error of PPCS (i.e., 0.4) is twice each of the errors of CS and SCP (i.e., 0.2) for Glacier. SCP succeeds in maintaining the recovery error of CS in all the tests, which validates both the correctness of *Theorem 3.2* and the effectiveness of CS-based encrypted data recovery in SCP.

6.2.3 Execution Cost. Finally, we analyze the execution cost of three methods. The statistics about execution times are illustrated in Table 1. In all the tests, SCP requires around 6 times the execution time (mainly taken up by the homomorphic operations) of the two baselines. The comparatively high time cost is a sacrifice for the protection of participants' locations and data. In real deployment, the computation process of SCP is conducted in the application server, which has a strong computational capacity to assure a real-time response. We also emphasize that the update operations of **U**'s rows and **V**'s columns (Steps 5-8 in Fig. 4) can be carried out in parallel to further reduce the time cost.

Compared to traditional CS, SCP requires some additional communication: Steps 1, 2, and 4 in Phase I (see Fig. 3) and Steps 2, 3, 5, and 6 in Phase II (see Fig. 4). For example, we assume a 50×50 sensory matrix with 60 sensory data (each 10bit) and assume the size of an encrypted location and the size of an AHE (VHE) ciphertext to be both 100bit. Then, for Phase I, the levels of data exchange for Steps 1, 2, and 4 are 2×100, 2×100, and $2 \times 50 \times 100$. Altogether, this phase increases 1.3KB of communication for each participant, which can be considered as a negligible overhead for mobile users; furthermore, for Phase II, the levels of additional data exchange between the server and the CSP in Step 2, 3, 5, and 6 are

[8]The difference exists due to the random initialization of matrices **U** and **V**.

Table 1. Execution times of CS, SCP and PPCS.

Execution time (s)		**CS**	**SCP**	**PPCS**
LUCE	*Temperature*	0.75	4.62	0.76
	Humid	0.76	4.61	0.78
Glacier	*Temperature*	0.31	1.87	0.3
	Humid	0.3	1.88	0.31

100×60, $(50 \times 50+50 \times 50+60) \times 100$, $1000 \times 60 \times 100$, and $1000 \times 60 \times 10$ for 1000 iterations, which amounts to around 870KB. This cost is also very light for the server and the CSP.

7 DISCUSSION

Limitations. In our SCP method, if the CSP and the application server collude with each other, then the participants' locations would unfortunately be revealed. Specifically, the server could decrypt $\text{AHE}(\widehat{\mathbf{S}^*})$ and recover $\mathbf{S}$ from $\mathbf{S}^*$ with the AHE private key and the permutation key (i.e., $\mathbf{A}$ and $\mathbf{B}$) of the CSP. Collusion can also take place between the participants and the server. In this case, the server could construct obfuscated actual location mappings of the malicious participants. For an honest participant who reports with an obfuscated location, the server can deduce its actual location if its obfuscated location is recorded in the mappings. The impacts of such a participants-involved collusion are decided by the distribution of malicious participants' locations. In other words, if the malicious group covers each row and each column of the sensory matrix, then all other participants' location privacy would be breached. However, at least $\max(\mathrm{h}, \mathrm{v})$ malicious participants are needed for this purpose, which is not feasible in real scenarios. Finally, we point out that collusion between the participants and the CSP would not put other participants' locations in danger.

Hence, SCP only works when there is no collusion between the server and the CSP. Meanwhile, SCP cannot protect one's location privacy if some participants in its corresponding line (row and column) collude with the server. This also specifies the security boundary of our method.

Implications. The implications of the proposed method are twofold. First, when designing location privacy-preserving methods for MCS applications, the impacts of location protection on data analysis should be carefully considered and well mitigated. Traditionally, location privacy and data analysis are studied in the data collection step and the data analytics step separately. However, as two essential steps of MCS tasks [42], the processed data in the former step will inevitably impact the analytical results of the latter. Even with perfect privacy protection, poor analytical results would destroy the application. In other words, it is important to preserve participants' location privacy, but the protective techniques should be compatible with corresponding data analysis tasks. Second, we emphasize that our SCP method, especially the conjecture on the data correlation of CS-based data recovery (Section. 4.2.1), is promising in providing privacy-preserving reconstruction for matrices compressed by CS (e.g., images [31]). For example, owners can perform elementary transformation on the visual matrices (i.e., images) before outsourcing them to the cloud for reconstruction[9], thus protecting the images' privacy with no reconstruction accuracy loss in a lightweight way.

[9]Images can be compacted by CS sampling to save storage. The reconstruction process for the compressed images is usually outsourced to the cloud due to its high computational complexity.

8 CONCLUSION AND FUTURE WORK

By taking both effective location protection and accurate data recovery into consideration, for the first time, we have proposed a novel location privacy-preserving method (i.e., SCP) for MCS in this paper. A conjecture on the inherent data correlation required by CS has been presented with theoretical proof. Based on this conjecture, we designed a correlation-preserving location obfuscation scheme to provide spatial camouflage for participants' locations. In addition, we identified the threat of location privacy leakage with sensory data as quasi-identifiers. To this end, an encrypted data recovery scheme was designed to protect participants' data based on homomorphic encryption techniques. The proposed method was verified with both a security analysis and extensive experiments on real-world datasets. The results show that participants' locations can be obfuscated with sufficient distortions comparable to the size of the target area, and missing data can be recovered with similar accuracy to CS.

As to future work, we will look into the privacy preservation of data recovery for high-dimensional sensory matrices. In fact, CS is only adequate for two-dimensional environment matrices, while for matrices with higher dimensions (e.g., noise tensor in [43]), the tensor decomposition technique is commonly used. Hence, we plan to extend our method by tailoring it to support tensor decomposition-based data recovery.

ACKNOWLEDGMENTS

The work is supported by the National Key Research and Development Program of China under Grant No.: 2016YFB1000302 and the National Natural Science Foundation of China under Grant Nos.: 61672195, 61433019, U1435217, 61772446, 61501482. Tongqing Zhou would like to thank Miaomiao Hu for her care and support.

REFERENCES

[1] Amr Alanwar, Yasser Shoukry, Supriyo Chakraborty, Paul Martin, Paulo Tabuada, and Mani Srivastava. 2017. PrOLoc: Resilient localization with private observers using partial homomorphic encryption. In *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 41–52.

[2] Miguel E Andrés, Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2013. Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 901–914.

[3] Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2014. Optimal geo-indistinguishable mechanisms for location privacy. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 251–262.

[4] Emmanuel J Candes and Yaniv Plan. 2010. Matrix completion with noise. *Proc. IEEE* 98, 6 (2010), 925–936.

[5] Ayon Chakraborty, Md Shaifur Rahman, Himanshu Gupta, and Samir R Das. 2017. SpecSense: Crowdsensing for efficient querying of spectrum occupancy. In *Proceedings of the IEEE Conference on Computer Communications (Infocom)*. IEEE, 1–9.

[6] David Chaum. 1983. Blind Signatures for Untraceable Payments. *Proceedings of CRYPTO*, 199–203.

[7] Benoît Chevalliermames, Pascal Paillier, and David Pointcheval. 2006. Encoding-free ElGamal encryption without random oracles. In *Proceedings of the International Conference on Theory and Practice in Public-Key Cryptography (PKC)*. Springer, 91–104.

[8] Delphine Christin, Andreas Reinhardt, Salil S. Kanhere, and Matthias Hollick. 2011. A survey on privacy in mobile participatory sensing applications. *Journal of Systems and Software* 84, 11 (2011), 1928–1946.

[9] Cory Cornelius, Apu Kapadia, David Kotz, Dan Peebles, Minho Shin, and Nikos Triandopoulos. 2008. Anonysense: Privacy-aware people-centric sensing. In *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys)*. ACM, 211–224.

[10] Srinivas Devarakonda, Parveen Sevusu, Hongzhang Liu, Ruilin Liu, Liviu Iftode, and Badri Nath. 2013. Real-time air quality monitoring through mobile sensing in metropolitan areas. In *Proceedings of the ACM SIGKDD International Workshop on Urban Computing*. ACM, 1–8.

[11] A Dimovski and D Gligoroski. 2003. Attacks on the transposition ciphers using optimization heuristics. *Proceedings of ICEST* (2003), 1–4.

[12] Benjamin C. M. Fung, Ke Wang, Rui Chen, and Philip S. Yu. 2010. Privacy-preserving data publishing: A survey of recent developments. *Comput. Surveys* 42, 4 (2010), 53.

[13] Raghu K. Ganti, Nam Pham, Yu En Tsai, and Tarek F. Abdelzaher. 2008. PoolView: Stream privacy for grassroots participatory sensing. In *Proceedings of the ACM International Conference on Embedded Networked Sensor Systems (SenSys)*. ACM, 281–294.

[14] Raghu K Ganti, Fan Ye, and Hui Lei. 2011. Mobile crowdsensing: Current state and future challenges. *IEEE Communications Magazine* 49, 11 (2011), 32–39.

[15] Bin Guo, Zhu Wang, Zhiwen Yu, Yu Wang, Neil Y Yen, Runhe Huang, and Xingshe Zhou. 2015. Mobile crowd sensing and computing: The review of an emerging human-powered sensing paradigm. *ACM Computing Surveys (CSUR)* 48, 1 (2015), 33.

[16] Chao Huang, Dong Wang, and Shenglong Zhu. 2017. Where are you from: Home location profiling of crowd sensors from noisy and sparse crowdsourcing data. In *Proceedings of the IEEE Conference on Computer Communications (Infocom)*. IEEE, 1–9.

[17] François Ingelrest, Guillermo Barrenetxea, Gunnar Schaefer, Martin Vetterli, Olivier Couach, and Marc Parlange. 2010. Sensorscope: Application-specific sensor network for environmental monitoring. *ACM Transactions on Sensor Networks (TOSN)* 6, 2 (2010), 32.

[18] Xiaocong Jin, Rui Zhang, Yimin Chen, Tao Li, and Yanchao Zhang. 2016. Dpsense: Differentially private crowdsourced spectrum sensing. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 296–307.

[19] Linghe Kong, Liang He, Xiao Yang Liu, Yu Gu, Min You Wu, and Xue Liu. 2015. Privacy-preserving compressive sensing for crowdsensing based trajectory recovery. In *Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 31–40.

[20] Linghe Kong, Mingyuan Xia, Xiao Yang Liu, Guangshuo Chen, Yu Gu, Min You Wu, and Xue Liu. 2014. Data loss and reconstruction in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems (TPDS)* 25, 11 (2014), 2818–2828.

[21] Qinghua Li, Guohong Cao, and Thomas F La Porta. 2014. Efficient and privacy-aware data aggregation in mobile sensing. *IEEE Transactions on Dependable and Secure Computing (TDSC)* 11, 2 (2014), 115–129.

[22] Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. 1996. *Handbook of applied cryptography.* CRC press.

[23] Chenglin Miao, Lu Su, Wenjun Jiang, Yaliang Li, and Miaomiao Tian. 2017. A lightweight privacy-preserving truth discovery framework for mobile crowd sensing systems. In *Proceedings of the IEEE Conference on Computer Communications (Infocom)*. IEEE, 1–9.

[24] James Newsome, Elaine Shi, Dawn Song, and Adrian Perrig. 2004. The sybil attack in sensor networks: analysis & defenses. In *Proceedings of the ACM/IEEE International Symposium on Information Processing in Sensor Networks (IPSN)*. ACM, 259–268.

[25] Evangelos Niforatos, Athanasios Vourvopoulos, Marc Langheinrich, Pedro Campos, and Andre Doria. 2014. Atmos: a hybrid crowdsourcing approach to weather estimation. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. ACM, 135–138.

[26] Valeria Nikolaenko, Stratis Ioannidis, Marc Joye, Marc Joye, Nina Taft, and Boneh Dan. 2013. Privacy-preserving matrix factorization. In *Proceedings of the ACM Sigsac Conference on Computer and Communications Security (CCS)*. ACM, 801–812.

[27] Layla Pournajaf, Daniel A Garcia-Ulloa, Li Xiong, and Vaidy Sunderam. 2016. Participant privacy in mobile crowd sensing task management: A survey of methods and challenges. *ACM SIGMOD Record* 44, 4 (2016), 23–34.

[28] Layla Pournajaf, Li Xiong, Vaidy Sunderam, and Slawomir Goryczka. 2014. Spatial task assignment for crowd sensing with cloaked locations. In *Proceedings of the International Conference on Mobile Data Management (MDM)*, Vol. 1. IEEE, 73–82.

[29] Swati Rallapalli, Lili Qiu, Yin Zhang, and Yi-Chao Chen. 2010. Exploiting temporal stability and low-rank structure for localization in mobile networks. In *Proceedings of the International Conference on Mobile Computing and Networking (Mobicom)*. ACM, 161–172.

[30] Hien To, Gabriel Ghinita, and Cyrus Shahabi. 2014. A framework for protecting worker location privacy in spatial crowdsourcing. *Proceedings of the VLDB Endowment* 7, 10 (2014), 919–930.

[31] Cong Wang, Bingsheng Zhang, Kui Ren, and Janet M Roveda. 2013. Privacy-assured outsourcing of image reconstruction service in cloud. *IEEE Transactions on Emerging Topics in Computing* 1, 1 (2013), 166–177.

[32] Leye Wang, Gehua Qin, Dingqi Yang, Xiao Han, and Xiaojuan Ma. 2018. Geographic differential privacy for mobile crowd coverage maximization. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. AAAI, 1–10.

[33] Leye Wang, Dingqi Yang, Xiao Han, Tianben Wang, Daqing Zhang, and Xiaojuan Ma. 2017. Location privacy-preserving task allocation for mobile crowdsensing with differential geo-obfuscation. In *Proceedings of the International Conference on World Wide Web (WWW)*. International World Wide Web Conferences Steering Committee, 627–636.

[34] Leye Wang, Daqing Zhang, Animesh Pathak, Chao Chen, Haoyi Xiong, Dingqi Yang, and Yasha Wang. 2015. CCS-TA: Quality-guaranteed online task allocation in compressive crowdsensing. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing (Ubicomp)*. ACM, 683–694.

[35] Leye Wang, Daqing Zhang, Yasha Wang, Chao Chen, Xiao Han, and Abdallah M'Hamed. 2016. Sparse mobile crowdsensing: challenges and opportunities. *IEEE Communications Magazine* 54, 7 (2016), 161–167.

[36] Leye Wang, Daqing Zhang, Dingqi Yang, Brian Y Lim, and Xiaojuan Ma. 2016. Differential location privacy for sparse mobile crowdsensing. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*. IEEE, 1257–1262.

[37] Qian Wang, Yan Zhang, Xiao Lu, Zhibo Wang, Zhan Qin, and Kui Ren. 2016. RescueDP: Real-time spatio-temporal crowd-sourced data publishing with differential privacy. In *Proceedings of the IEEE Conference on Computer Communications (Infocom)*. IEEE, 1–9.

[38] Liwen Xu, Xiaohong Hao, Nicholas D Lane, Xin Liu, and Thomas Moscibroda. 2015. More with less: Lowering user burden in mobile crowdsourcing through compressive sensing. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing (Ubicomp)*. ACM, 659–670.

[39] Zhiwen Yu, Huang Xu, Zhe Yang, and Bin Guo. 2016. Personalized travel package with multi-point-of-interest recommendation based on crowdsourced user footprints. *IEEE Transactions on Human-Machine Systems* 46, 1 (2016), 151–158.

[40] Fan Zhang, Li He, Wenbo He, and Xue Liu. 2012. Data perturbation with state-dependent noise for participatory sensing. In *Proceedings of the IEEE Conference on Computer Communications (Infocom)*. IEEE, 2246–2254.

[41] Yin Zhang, Matthew Roughan, Walter Willinger, and Lili Qiu. 2009. Spatio-temporal compressive sensing and internet traffic matrices. In *ACM SIGCOMM Computer Communication Review*, Vol. 39. ACM, 267–278.

[42] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. 2014. Urban computing: Concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 5, 3 (2014), 55.

[43] Yu Zheng, Tong Liu, Yilun Wang, Yanmin Zhu, Yanchi Liu, and Eric Chang. 2014. Diagnosing New York city's noises with ubiquitous data. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing (Ubicomp)*. ACM, 715–725.

[44] Hongchao Zhou and Gregory Wornell. 2014. Efficient homomorphic encryption on integer vectors and its applications. In *Information Theory and Applications Workshop*. 1–9.

[45] Gaoqiang Zhuo, Qi Jia, Linke Guo, Ming Li, and Pan Li. 2016. Privacy-preserving verifiable data aggregation and analysis for cloud-assisted mobile crowdsourcing. In *Proceedings of the IEEE Conference on Computer Communications (Infocom)*. IEEE, 1–9.

A HASH-ELGAMAL ENCRYPTION

Let $\mathbb{G}$=$\langle g \rangle$ be a multiplicative cyclic group of order p and H : $\mathbb{G} \rightarrow \mathbb{Z}_M$ be a cryptographic hash function, where M is a sufficiently large integer. One can generate a public key pk=(g, y) where y=g^{sk} with sk $\in \mathbb{Z}_\mathrm{p}$ as a randomly chosen secret key. Given pk and sk, message $m \in \mathbb{Z}_M$ is encrypted using: $c = (g^\rho, (m+H(y^\rho)) \bmod M) = (c_1, c_2)$ for a random value $\rho \in \mathbb{Z}_p$. The ciphertext c can be masked (which equals constant addition) by anyone with any chosen random mask $\mu \in \mathbb{Z}_M$ with: $\hat{c} = (c_1, (c_2+\mu) \bmod M) = (\hat{c_1}, \hat{c_2})$. Finally, a masked message can be obtained by decrypting $\hat{c}$:

$$\hat{m} = \hat{c_2}\text{-}H(\hat{c_1}^{\mathrm{sk}}) \bmod M\text{=}\ (c_2+\mu)\text{-}H(c_1^{\mathrm{sk}})\text{=}\ ((m+H(g^{\mathrm{sk}\rho}))+\mu)\text{-}H(g^{\mathrm{sk}\rho})\text{=}\ m+\mu$$

B INTEGER VECTOR HOMOMORPHIC ENCRYPTION

Let $\mathbb{X} \in \mathbb{Z}^m$ be an integer vector to be encrypted and $\mathbb{C} \in \mathbb{Z}^n$ be the ciphertext of $\mathbb{X}$. The secret key is a matrix $\mathbf{S} \in \mathbb{Z}^{m \times n}$ that satisfies $\mathbf{S}\mathbb{C}$=$\omega\mathbb{X}$+$\mathbb{E}$, where $\mathbb{E}$ is a noise vector and ω is an integer much bigger than $|\mathbf{S}|$ and $|\mathbb{E}|$ ($|\cdot|$ denotes the maximum value in a vector or a matrix).

(**Key-switching**) The encryption process is based on a key-switching technique, which allows changing a key-ciphertext pair to another specific secret key while still encrypting the same vector. Briefly, suppose the secret key and the ciphertext vector for vector $\mathbb{X} \in \mathbb{Z}_p^m$ are $\mathbf{S} \in \mathbb{Z}_p^{m\times n}$ and $\mathbb{C} \in \mathbb{Z}_p^n$ and the desired secret key is $\mathbf{S}' \in \mathbb{Z}_p^{m\times n'}$. A key-switching matrix $\mathbf{M} \in \mathbb{Z}_p^{n'\times n}$ is constructed such that $\mathbf{S}'\mathbf{M}$=$\mathbf{S}$+$\mathbf{E}$ for some noise matrix $\mathbf{E}$ with small magnitude. If $\mathbf{S}'$ is in the form of $[\mathbf{I}, \mathbf{T}]$, which is a concatenation of an identity matrix $\mathbf{I}$ and some matrix $\mathbf{T}$, then the key-switching matrix can be constructed by:

$$\mathbf{M}=\begin{pmatrix} \textbf{-TA+S+E} \\ \mathbf{A} \end{pmatrix}$$

where $\mathbf{A} \in \mathbb{Z}^{(n'\text{-}m)\times n}$ is a random matrix. The new ciphertext vector can then be defined as $\mathbb{C}'$=$\mathbf{M}\mathbb{C}$.

(**Encryption and decryption**) Let $\mathbf{I}$ be an $m \times m$ identity matrix, then $\mathbf{I}(\omega\mathbb{X})$=$\omega\mathbb{X}$. Matrix $\mathbf{I}$ can be treated as a secret key and $\omega\mathbb{X}$ as a ciphertext. Given the actual secret key $\mathbf{S}$, one can switch $\mathbf{I}$ to it with key-switching, by which a new ciphertext $\mathbb{C}$=$\mathbf{M}\omega\mathbb{X}$ can be obtained. Based on $\mathbf{S}$, the decryption process is straightforward: $\mathbb{X}$=$\lceil \mathbf{S}\mathbb{C}/\omega \rfloor$.

(**Operations on the encrypted data**) This scheme supports the homomorphic operations for vector additions, and inner products. For two plaintext-ciphertext pairs $(\mathbb{X}_1, \mathbb{C}_1)$ and $(\mathbb{X}_2, \mathbb{C}_2)$ with secret key $\mathbf{S}_1$ and $\mathbf{S}_2$, these operations work as follows:

- **Addition**. Key-switching is first used to convert the two different keys to the same one $\mathbf{S}'$. The ciphertexts are transformed to $\mathbb{C}_1'$ and $\mathbb{C}_2'$ accordingly. Then the valid encryption for $\mathbb{X}_1$+$\mathbb{X}_2$ can be computed as $\mathbb{C}_1'$+$\mathbb{C}_2'$ with the secret key $\mathbf{S}'$.
- **Inner Products**. The valid encryption for $\mathbb{X}_1 \cdot \mathbb{X}_2$ that should be computed is $\lceil \text{vec}(\mathbb{C}_1 \cdot \mathbb{C}_2^T)/\omega \rfloor$. Meanwhile, the secret key should be updated to $\text{vec}(\mathbf{S}_1^T \cdot \mathbf{S}_2)^T$ (i.e., key updating[10]).

As to the vector-constant operations, the encryption for $\mathbb{X}_1$+a is $\mathbb{C}_1$+$\omega[a^T, 0, ..., 0]^T$, while the encryption for $a \cdot \mathbb{X}_1$ is $a \cdot \mathbb{C}_1$, both under the secret key $\mathbf{S}_1$ and without key updating.

Received February 2018; revised April 2018; accepted September 2018

[10]Key-switching adjusts a secret key to another key, while key updating is for the decryption of encrypted operation results.