Hindawi
Security and Communication Networks
Volume 2017, Article ID 7576307, 12 pages
https://doi.org/10.1155/2017/7576307

Research Article

Location Privacy Leakage through Sensory Data

Yi Liang,[1] Zhipeng Cai,[1] Qilong Han,[2] and Yingshu Li[1]

[1] *Department of Computer Science, Georgia State University, Atlanta, GA 30303, USA*
[2] *College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China*

Correspondence should be addressed to Zhipeng Cai; zcai@gsu.edu

Received 26 March 2017; Accepted 7 June 2017; Published 6 August 2017

Academic Editor: Qing Yang

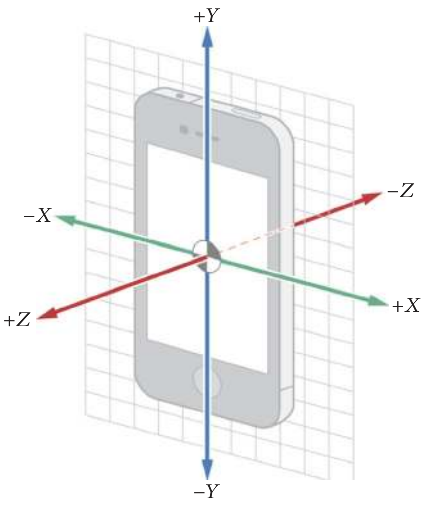
Mobile devices bring benefits as well as the risk of exposing users' location information, as some embedded sensors can be accessed without users' permission and awareness. In this paper, we show that, only by using the data collected from the embedded sensors in mobile devices instead of GPS data, we can infer a user's location information with high accuracy. Three issues are addressed which are route identification, user localization in a specific route, and user localization in a bounded area. The Dynamic Time Warping based technique is designed and we develop a Hidden Markov Model to solve the localization problem. Real experiments are performed to evaluate our proposed methods.

1. Introduction

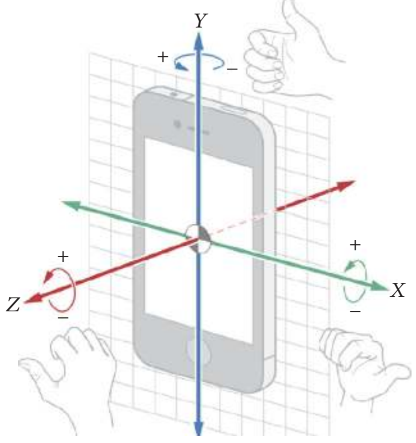
While people are enjoying the many benefits brought by mobile devices, people have to take the risk of losing privacy by leaking private information [1–8], especially location information [9–13]. People now heavily rely on services provided by third-party Apps which usually collect users' location information. Such Apps provide users with convenience, while they also threaten users' privacy. Location information is sensitive and malicious adversaries can make use of location information to attack users or threat the public. Therefore, location privacy has attracted tremendous attention from researchers who are struggling to protect location privacy without degrading service qualities of third-party Apps.

The most common way for Apps to obtain location information is to get access to the GPS [14, 15] module in a mobile device. Thus, some methods aim at controlling the access to the GPS module to protect location privacy. In reality, third-party Apps need users to authorize the access to the GPS module so that users may control the tradeoff between service quality and privacy preservation. Such a strategy seems to provide satisfiable location privacy. However, many works have pointed out that, without accessing GPS data, Apps can still infer private information, such as input on touch-screen [16] and motion status [17], through the data collected by general embedded sensors in mobile devices [18–27]. Unfortunately, few works try to utilize built-in sensors like accelerometer, magnetometer, gyroscope, and so on to do localization. These sensors are very sensitive and their readings may have lots of noises due to stochastic events such as tiny vibrations of mobile devices. Thus it is extremely challenging for inferring location information merely based on noisy sensory data. However, combined with reasonable background knowledge, readings from these sensors can be utilized to infer a user's location information. Such sensor readings are considered nonsensitive and can be obtained without user permission, which causes a big threat to location privacy [28].

In this paper, we propose a novel method to infer a user's location which only utilizes the data collected from the accelerometer and gyroscope in a mobile device. Such data can even be collected easily without users' awareness [29]. Our work is inspired by the fact that sensor readings are highly related to the route a user is taking, which can reveal the user's location. Most people generally have relatively stable life patterns in their daily lives. We then take driving pattern as a case study in this work. Driving pattern is unique for each person. We take advantage of this feature to infer users' location information through unique fingerprints collected from people's daily lives. Regarding driving pattern, we have the following observations. It is very common that a



(a) Accelerometer



(b) Gyroscope

Figure 1: Sensors in a smart phone.

person takes the same route to go to work/school or go home at specific time every weekday. A person may be jammed on the same road segments every day. There are only several reasonable routes that people would like to choose to drive to a specific destination. The time it takes every day to drive to a particular location along the same route is almost the same. If adversaries can obtain the sensory data profiles for a set of known routes in advance, they can track a mobile device on those routes by secretly gathering sensory data from that device and matching it with the prerecorded profiles. Based on these observations, we address the following three issues in this paper based on the sensory data collected from mobile devices.

(1) Given a set of possible routes that a user would like to drive along every day, how to decide which route the user is driving along?

(2) Given the selected route of a user, how to infer the user's location in real time?

(3) In a bounded area, how to trace a user?

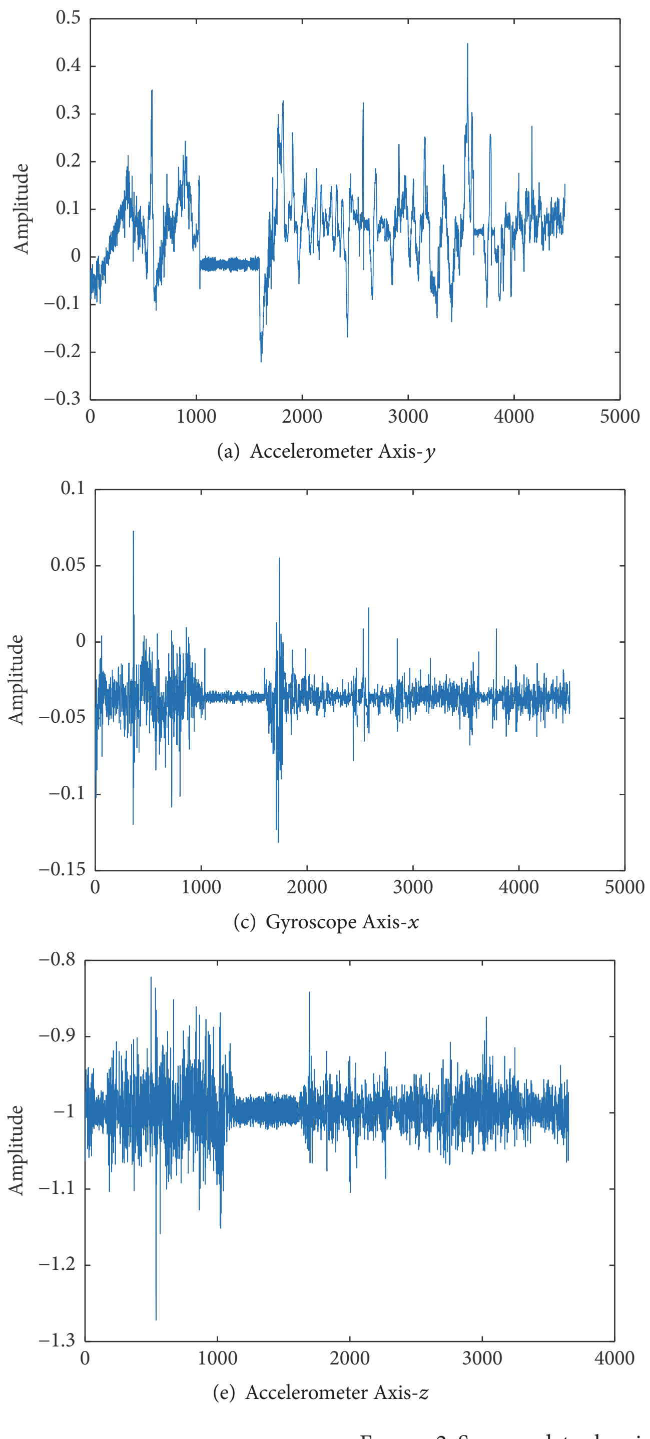
These three issues address three different aspects of location privacy, with increasing difficulty level. To the best of knowledge, this is the first work to make use of sensory data collected from embedded sensors in mobile devices to infer location information without considering GPS data. The reminder of the paper is organized as follows. Section 2 discusses how to collect sensory data, followed by the attack model in Section 3. Sections 4, 5, and 6 explain how to solve the three issues, respectively. Section 7 presents the experiment results. Section 8 reviews the related works and the paper is concluded in Section 9.

2. Sensory Data Collected by Mobile Devices

We collected some real data to validate our observations. We adopt smart phones with accelerometer and gyroscope as our mobile devices. Almost every smart phone has at least such two kinds of sensors which have three axes x, y, and z. Each axis represents a dimension of a smart phone as shown in Figure 1.

We collected the sensory data of several routes for about 10 days and all the sensory data of the same route show similarity. Figure 2 shows an example data set for one route, which was collected for 2 different days. Figures 2(a), 2(b), and 2(c) are for one day. Figures 2(d), 2(e), and 2(f) are for the other day. As we can see, the data patterns for these two different days demonstrate high similarity. The x-axis of each subfigure in Figure 2 represents the total time to collect the data. The two days had different total data collection durations because the driving speeds in these two days are different. Then we can tell that even with different driving speeds in different days, as long as it is for the same route, similar data patterns always present. That is to say, each route has its unique data pattern. Such a fact ensures that we can definitely infer location information through sensory data collected from the sensors embedded in mobile devices without accessing GPS data. Our extensive experiment results in Section 7 also validate it.

To figure out which kind of sensory data can characterize data pattern for a specific route is a fundamental issue. We take accelerometer and gyroscope, which are two common sensing units in a mobile device, as two representative kinds of sensors in this work. Accelerometer can measure linear acceleration and gyroscope can track angular velocity of three axes of a smart phone as shown in Figure 1. Actions such as speeding up, breaking, and turning left/right are the most common driving actions. All such actions can be precisely captured by linear acceleration and angular velocity which can be conveniently measured by accelerometer and gyroscope, respectively. We conducted extensive experiments to prove our hypothesis. In our experiments, each smart phone was placed on the dashboard of a car with screen face up and the positive y-axis of accelerometer towards the



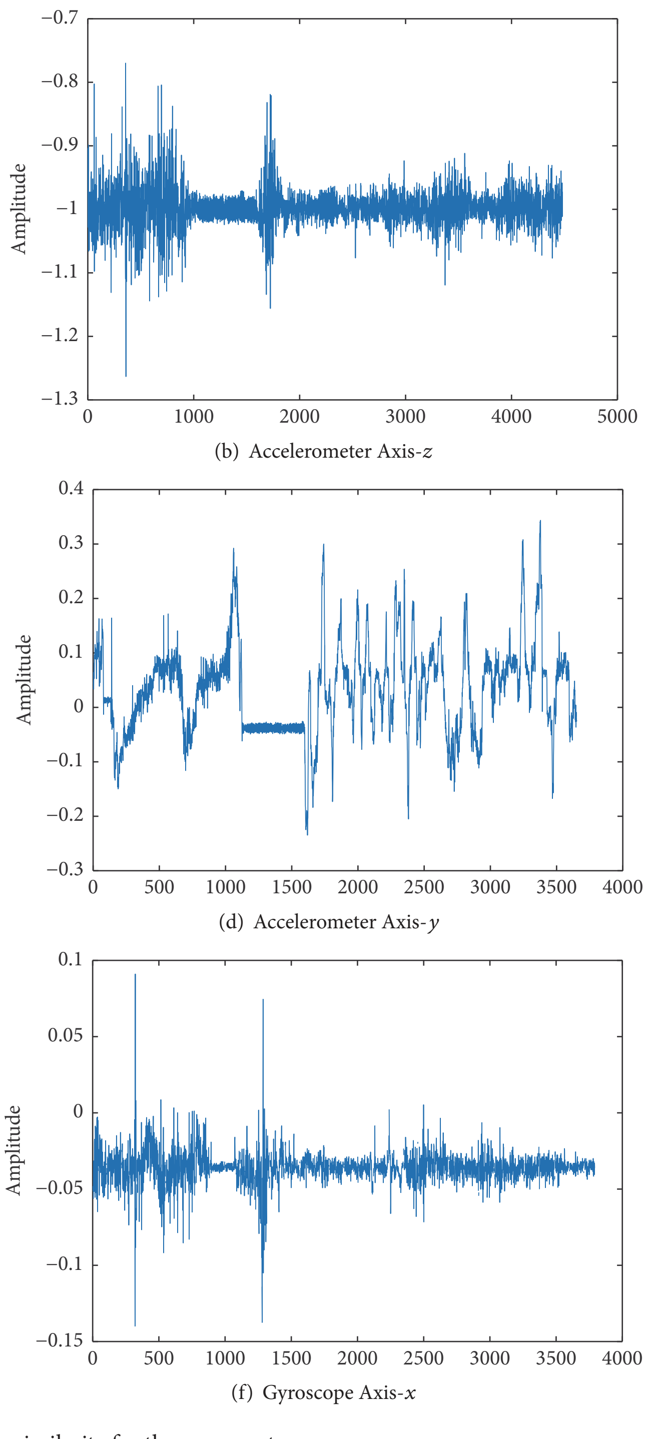


Figure 2: Sensory data showing similarity for the same route.

driving direction. Note that it is not necessary to place a smart phone in this way in real applications. Our purpose is to simplify the experiments. As depicted in Figure 2, the crests in Figures 2(a) and 2(d) represent breaks, while the troughs represent acceleration. The crests in Figures 2(c) and 2(f) represent left turns, and the troughs represent right turns. We can see that breaks and acceleration can be captured by the y-axis readings of the accelerometer, steering can be captured by the x-axis readings of the accelerometer and the x-axis and y-axis readings of the gyroscope, and road conditions (bump, downhill, slope, etc.) can be captured by the z-axis readings of the accelerometer.

Figure 3 shows the z-axis readings of the gyroscope for some sharp/slow turns made at the same intersection. We can see that the only difference between sharp turns and slow turns is the shape of the corresponding peaks which are greatly different from that of the line representing no turning. This example indicates that even

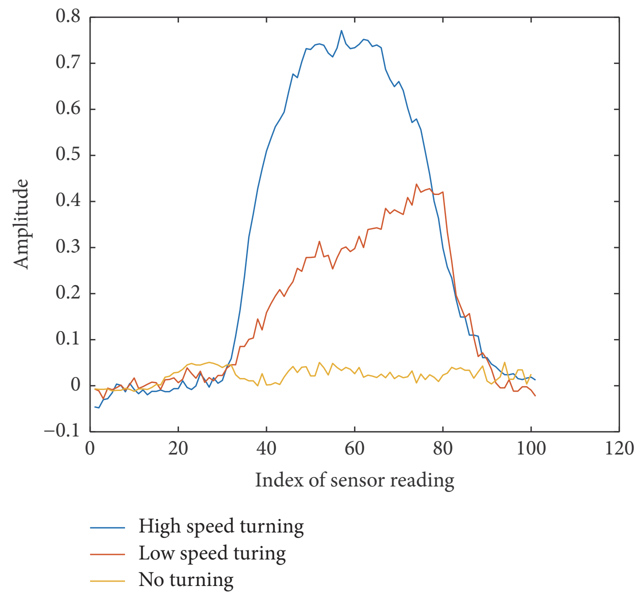


Figure 3: Influence of driving habit.

if people have different driving habits, the resultant data for the same action present the same pattern and the only difference lies in the actual values, which means that driving behaviors may affect sensory readings but have no impact on data patterns. Therefore, we can infer one's location information given precollected data for targeted routes.

Different sensory readings have different usefulness. Initially, we used y-axis readings of accelerometer to infer one's accurate location because y-axis readings of accelerometer can capture the break actions. Unfortunately, we find that this method is not quite effective for localization in practice, as one may break arbitrarily anywhere in a road segment, making it hard to precisely locate a user. This is because even if the road conditions are the same every day, the real time traffic conditions affecting one's driving speed may be quite different. Sometimes, y-axis readings of accelerometer may even hinder us from locating users. However, we find that a user may have the same break frequency or speedup frequency on some particular road segment which can help with route identification. For example, one may break very frequently on a particular road segment resulting in many crests in the collected data, and such a pattern is useful for identifying this road segment. Therefore, we make use of y-axis readings of accelerometer for route identification, not for location inference. We also find that road conditions are generally stable, as the locations of downhill, uphill, and intersections in which one needs to make turns are the same for each route.

Thanks to the aforementioned observations, we are able to locate mobile devices based on the sensory data which can be easily collected. From the sensory data, we can also extract unique fingerprints to identify road segments with high accuracy.

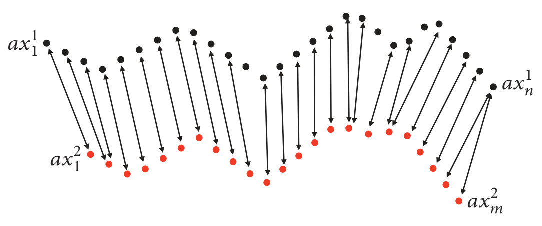
3. Attack Model

We have no special requirements regarding the attack model and our attack model is very reasonable compared with the ones in the previous works. Basically, there are just two roles in our attack model, attackers and users. Attackers are the adversarial App providers, and users are the ones who have installed these adversarial Apps. An attacker tries to obtain a user's location information secretly, assuming the attacker has successfully attracted the user to install malicious Apps on the mobile device. Then, the attacker can easily collect the user's sensory data because many sensors like accelerometer and gyroscope can be accessed by malicious Apps without user permissions. The only requirement is that the malicious Apps can upload the sensory data to the attacker's backend server through Internet so that the attacker can analyze the sensory data for location inference. All of these actions can be carried out without user's awareness. As mentioned before, sensory data collected by mobile devices may threaten privacy. Even worse, most users and many manufacturers have not even realized such a threat.

4. Route Identification

In this section, we explain how to identify a route, which is the first step towards location inference. Suppose users drive to work every weekday morning and the number of possible destinations for each user is limited. Moreover, for each destination, there are only a few reasonable routes that a user would like to take. All in all, the set of all the possible routes for each user is limited. From personal perspective and experience, we think this assumption is reasonable. If we can infer which route a user is taking, then to infer all the possible destinations for each user becomes possible which threatens user privacy.

Each route has relatively unique road conditions involving intersections, stop signs, traffic lights, and so on. Then the resultant sensory data from a user can characterize each route. For example, a user, who goes to work every weekday, may be jammed on the same road segments and stop at the same places for traffic lights and stop signs. Then the corresponding sensory data are unique and stable. Without loss of generality and for simplicity, we assume a known finite set of routes for each user. Each route in the set has a corresponding sensory data pattern as shown in Section 2. We can collect the sensory data profile for each route beforehand; then we can compare a user's sensory data with the available profiles to identify routes. However, the following challenges present. For a specific route, the data collected on multiple days may be different because of real time driving speed and traffic conditions. Many unpredictable events may occur, which also results in data difference. Furthermore, the collected data may have noises caused by shaking of cars, slight movement of smart phones, and so on. All these factors degrade the quality of the collected sensory data and make route identification even more challenging. Actually, route identification is to match sensory data patterns. If two sets of sensory data present the same pattern, we strongly believe they represent the same route. For the same route, since there are so many



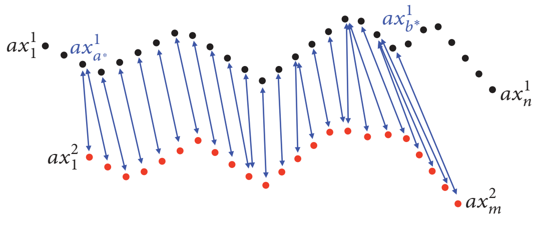


FIGURE 4: Two examples for DTW-based sequence alignment.

factors causing data difference, we cannot expect two sets of sensory data representing the same route to present exactly the same pattern. In order to address these challenges, we first need to define similarity between two sets of sensory data. The data collection durations for different routes vary greatly. Then simple measurement Euclidean distance is obviously ineffective in measuring similarity because Euclidean distance can only be used for phase aligned sequence. In order to accommodate noises and various data collection durations, it is better to consider the shape of a sequence of sensor readings for distinguishment.

4.1. Dynamic Time Warping. Dynamic Time Warping (DTW) is a powerful tool to measure a distance-like quantity between two time series which may vary in speed and duration [30]. The obtained distance-like quantity reflects the similarity between these two nonlinearly aligned time series. This is exactly what we need for sensory data matching, since real time traffic is unpredictable resulting in various data collection durations. Therefore, we employ DTW to find out along which route a user is driving given a set of possible routes that the user would like to drive along.

Let ax, ay, az and gx, gy, gz be the x, y, z-axis values of accelerometer and gyroscope, respectively. Assume $ax^1 = (ax_1^1, ax_2^1, ax_3^1, \ldots, ax_n^1)$ and $ax^2 = (ax_1^2, ax_2^2, ax_3^2, \ldots, ax_m^2)$ are an accelerometer's x-axis readings for two different days, where n and m are their collection durations, respectively. If n and m are different, the Euclidian distance is not proper for measuring the similarity between these two sequences. Our primary task is to compare two sensor reading sequences collected on different days for the same route even if they have different collection durations. Then we define similarity based on a time warping path. First, we use an $n \times m$ matrix M to represent the point-to-point distance between two sensor reading sequences ax^1 and ax^2. Figure 4(a) shows two sensor reading sequences with similar data collection durations. Figure 4(b) shows two sensor reading sequences with more different data collection durations. Entry M_{ij} in M indicates the way we align ax_i^1 and ax_j^2. Then we can derive a time warping path $P = (p_1, p_2, p_3, \ldots, p_K)$ to represent the alignment and matching relationship between ax^1 and ax^2, where $p_k = (i, j)$ $(1 \leq k \leq K)$ indicates the alignment and matching between ax_i^1 and ax_j^2 with $\min(m, n) \leq K \leq m + n - 1$. The different data collection durations of different sensor reading sequences resulted from the different driving speeds in different days. Since DTW can align multiple sensor readings in one sequence to a particular sensor reading of another sequence, we are able to successfully align two sequences with different data collection durations. Based on the obtained time warping path $P = (p_1, p_2, p_3, \ldots, p_K)$, we define the distance between two sensor reading sequences ax^1 and ax^2 as follows:

$$Dist\left(ax^1, ax^2\right) = \sum_{k=1}^{K} d_k, \tag{1}$$

where d_k denotes the distance between ax_i^1 and ax_j^2.

We collected the sensory data along the 6 dimensions for each route in a set of routes for several days. These data are used as our training data. Then we computed the distance between the test data and our training data. For each dimension, we derive a similarity score between the test data and training data. The final similarity score for each route is the sum of these 6 similarity scores. The route with the smallest similarity score is the identified one that matches a route in the training data set. As sensory data have a lot of noises, we need to smooth the data before computing similarity scores. Furthermore, we use two classic methods to optimize the DTW algorithm whose time complexity is $O(nm)$, where n and m are the data collection durations for two sensor reading sequences. The first method is based on the fact that although the durations vary, their difference is limited. Suppose the maximum time duration difference for the same route is $\mathrm{MD} = \max(|m - n|)$. Then we can reduce the searching space in our algorithm. Assuming that the sampling rate is r, each alignment and matching in $P = (p_1, p_2, p_3, \ldots, p_K)$ does not exceed $r * \mathrm{MD} * 60$. That is, for every $p_k = (i, j)$ $(1 \leq k \leq K)$

$$\max\left|i - j\right| \leq r * \mathrm{MD} * 60. \tag{2}$$

For all of our testing routes, the maximum difference for the same route is 4 minutes. We limit the searching space within a bounded area to increase the searching speed. The second method is called the multiscale DTW. Because we just identify the route with the smallest similarity distance as the result, exact similarity distance is not necessary. Then we can resample the sensory data sequences to reduce the dimension of matrix M. This method also substantially speeds up the searching speed.

5. Location Inference on a Particular Route

This section discusses how to locate a user on a particular route given real time sensory data. We assume an attacker knows along which route a user is driving. In this case, the attacker can collect the data for a small road segment from malicious Apps installed in the user's mobile device. To locate a user, subsequence matching needs to be performed between real time sensory data and the data for the entire route. Here, real time sensory data is the test data, and the data for the entire route is the training data.

DTW can also be used for subsequence matching with minor modifications. For instance, for the ax dimension, a route's sensory data is $ax^1 = (ax_1^1, ax_2^1, ax_3^1, \ldots, ax_n^1)$ and the query segment's sensory data is $ax^2 = (ax_1^2, ax_2^2, ax_3^2, \ldots, ax_m^2)$, where $n \gg m$. It is different from route identification in which the start and end points of ax^1 are aligned with the start and end points of ax^2 as shown in Figure 4(a). In subsequence matching, the start and end points of ax^1 can be aligned to any points in ax^2. That is, in route identification, $p_1 = (1, 1)$ and $p_K = (n, m)$. While for subsequence matching, this requirement is not necessary.

Our goal is to find subsequence $ax^1(a^* : b^*) = (ax_{a^*}^1, ax_{a^*+1}^1, ax_{a^*+2}^1, \ldots, ax_{b^*}^1)$ with $1 \le a^* \le b^* \le n$ minimizing the time warping distance to ax^2 over all possible subsequences of ax^1. In other words,

$$(a^*, b^*) = \underset{(1 \le a \le b \le n)}{\arg\min} \left(Dist\left(ax^1 (a : b), ax^2 \right) \right). \tag{3}$$

The algorithm is pretty simple which can be found in [30]. Let $P^* = (p_1^*, p_2^*, p_3^*, \ldots, p_K^*)$ be the identified warping path; we have $p_1^* = (a^*, 1)$ and $p_K^* = (b^*, m)$ as shown in Figure 4(b). Then we can infer b^* is the current location of the user.

Roughly speaking, to infer a user's location, we employ the modified DTW algorithm to find the most likely subsequence along a route and consider the end point of the subsequence as the inferred location of the user. The most challenging issue is that there may exist many similar subsequences along a very long route; for example, a user is driving along a highway with constant velocity. In this case, we need to take account of other information such as time and traffic conditions. The simplest method to deal with this issue is to consider the time difference. We assume the start time of a training sequence for the given route and the start time of the test subsequence are both known. According to all the training sequences for a given route, we can reduce the searching space to a specific range to reduce inference error. Even though we cannot completely eliminate such errors, in practice, the dynamically changing road and traffic conditions, sudden events, and climate reasons can all help with characterizing sequences. Then the number of the similar subsequences along a route is not large. In our experiments, such an issue does not present.

6. Location Inference in a Bounded Area

There are some works for tracing a user in a bounded area based on private location information of users without user awareness [18, 19]. However, some assumptions in these works may not be practical. For example, users may choose to detour due to traffic jam or emergencies. In this case, since there are no training data for the new route, it may be impossible to identify the route. Another challenge is that there are so many possible routes and it is impossible to collect the training data for all the possible routes. Furthermore, it is infeasible to compute the similarity distance between the query sensory data and the entire database data.

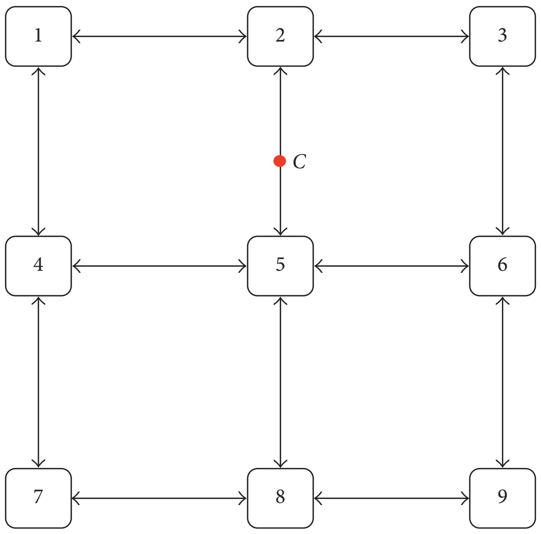


Figure 5: Road map.

In order to develop a more general method for location inference, we employ a Hidden Markov Model (HMM). As shown in Figure 5, we split all the routes into small segments based on intersections. Each rounded rectangle represents an intersection and each arrow represents a road segment. Bidirectional arrows represent two-way roads, while directional arrows represent one-way roads. Let I denote the set of intersections and S denote the set of road segments. A road segment is denoted by $s = (i, j)$, indicating s is between intersection i and intersection j. We consider a road segment s as a state, and the transition probability of s is determined by s's outgoing degree. For example, from segment s_1, a user can go to segments s_2 and s_3. Then the transition probability from s_1 to s_2 or s_3 is 0.5. We may also define transition probability based on real time traffic. For example, 30% of cars go from s_1 to s_2; then the transition probability from s_1 to s_2 is 0.3. This method requires real time traffic information at each intersection and usually it is impractical. Thus, we make use of outgoing degree to define transition probability.

To calculate a user's probability of arriving at a particular location, we have the following strategy. As shown in Figure 5, when a user is driving along road segment $(2, 5)$, if the user passes intersection 5, the state changes from 2 to 5. Otherwise, suppose the user stops at a particular location C on segment $(2, 5)$, and C is the location to be inferred. Then the probability of arriving at C is determined by the similarity distance between the observed segment sensory data for $(2, C)$ and the training sensory data for $(2, 5)$. The similarity distance can be computed using the DTW algorithm presented in the previous section. With this probability, we are able to identify the final location of the user on segment $(2, 5)$.

```
Require: $\pi$, $T_{\text{Prob}}$, $OB$, $RD^i$ for each road segment $s_i$
Ensure: The most likely trajectory $MPS$ for $OB$
(1)  for all stage $t$ do
(2)      if ($t == 1$) then
(3)          [Dist, $end\_point_1$] = FRS(1)
(4)          $MPS_i^t = \max(\pi_i * Dist)$
(5)          Let the end_point be the start_point at the next stage
(6)      else
(7)          for all segment $s_i$ do
(8)              $MPS_i^t = \max(MPS_i^{t-1}) * T_{\text{Prob}}$
(9)          end for
(10)         [$Dist, end\_point_t$] = FRS($end\_point_{t-1}$)
(11)         $MPS_i^t = MPS_i^t *$ Dist
(12)     end if
(13) end for
```

Algorithm 1: Modified Viterbi algorithm.

In summary, to trace a user in a bounded area, we first need to collect the sensory data for all the road segments within the area, which is possible since the number of the road segments in the bounded area is limited. Obviously, the route traversed by a user is a concatenation of a subset of all the road segments. Once we collect the sensory data from a user's mobile device, we can infer the most possible route traversed by the user. Then the user can be located in a bounded area. The following are the details of this location inference method in a bounded area.

6.1. The Modified Viterbi Algorithm. The Viterbi algorithm [31] is a dynamic programming algorithm to find the most likely sequence of hidden states which generates the input sequence of observed events. In our work, each road segment is regarded as a hidden state, and the sensory data from a mobile device are regarded as the input sequence of observed events. Then given the sensory data of some road segments from a user's mobile device in terms of a sequence, the Viterbi algorithm can help us identify the most likely route traversed by this user. However, the Viterbi algorithm deals with discrete events, while our collected sensory data are continuous. Besides, the data collection duration for a road segment may not be fixed, and it is hard to find the intersections that divide the sensory data into road segments. Thus, we modify the Viterbi algorithm so that it can be used to locate a user in a given area.

For each road segment, we have the corresponding training data. Let RD^i be the training data for road segment s_i. Let $T_{\min}^i$ and $T_{\max}^i$ be the shortest and longest time to traverse s_i, respectively. $T_{\min}^i$ and $T_{\max}^i$ can be obtained from the training data of s_i. Let l_{ob} be the data collection duration of query route OB. The collected sensory data for OB, which is a sensor reading sequence, correspond to the entire route traversed by the user and this route consists of road segments. The purpose is to identify the route traversed by the user through matching the sequence of OB with the training data. Our basic idea is to break OB into different road segments so that we can employ the modified Viterbi algorithm to locate a user.

To break OB into road segments, at each stage, we need to cut OB utilizing the DTW algorithm. For example, at the first stage, we start from the first point of OB. Then for all the possible next road segments s_i where $i \in 1, 2, \dots, m$, we compute the similarity distance between s_i and a subsequence of OB. Let this subsequence be $\widehat{s}_i$. As we know, $\widehat{s}_i$ starts from the first point of OB. Since we do not know the user's exact travel time of s_i, the end point of $\widehat{s}_i$ could be reached in time $T_{\min}^i$ to $T_{\max}^i$ as mentioned above. We need to compute the similarity distance between all such possible $\widehat{s}_i$ and s_i and choose the smallest similarity distance as the similarity distance for s_i. The end point of $\widehat{s}_i$ is the point that derives the smallest similarity distance for s_i. The end point of $\widehat{s}_i$ is regarded as the start point at the next stage. Then the above process is repeated. In this way, we can break OB into road segments. Algorithm 1 is the pseudocode for the modified Viterbi algorithm which breaks OB into road segments. The input of Algorithm 1 includes the initial probabilities π_i $(1 \le i \le n)$ for a user to start from road segment s_i, transition probability matrix T_{Prob} among road segments with size $K * K$, observed sensory data OB, and the training data RD^i for road segments s_i $(1 \le i \le n)$. Let MPS_i^t be the probability of road segment s_i being determined for stage t. After running Algorithm 1, we derive vector MPS_i for road segment s_i for all the stages. By tracing back from the final stage, we can obtain the most possible trajectory for OB.

Algorithm 2 is to determine the exact end point of a particular road segment $\widehat{s}_i$ given the start point of $\widehat{s}_i$. Let $Dist_i$ be the similarity distance between $\widehat{s}_i$ and the training data and end_point_i be the end point of $\widehat{s}_i$. mps_{i-1} is the end point of the previous stage and the step size τ is 1 second. Actually, we consider vector $Dist$ as the emission probability that generates the observation at each stage.

A user may finally stop at an intermediate point on a road segment. That is, the collected sensory data OB is a concatenation of several complete segments and a partial segment. However, Algorithm 1 can only derive a rounded

```
Require: End point $mps_{i-1}$,
Ensure: Similarity distances for all possible road segments
        and corresponding end points $mps_i$
(1) for all $RD^i$ do
(2)     if $T^i_{\min} + \tau$ of $RD^i$ is smaller than $l_{ob} - mps_{i-1}$ then
(3)         $Dist_i = \min_{\tau \in [0, T^i_{\max} - T^i_{\min}]} Dist\left(ob\left(mps_{i-1} : T^i_{\min} + \tau\right), RD^i\right)$
(4)     else
(5)         $Dist_i = Infinity$
(6)     end if
(7)     return $Dist$ and $mps_{i-1} + \tau + 1$
(8) end for
```

Algorithm 2: Finding road segment (FRS) for stage *i*.

sequence representing a set of complete road segments. Let us call the last complete road segment derived from Algorithm 1 as the final complete road segment, after which the next possible segment that the user would like to go to must be an adjacent road segment of it. Then we can employ the method of location inference on a particular route introduced in the previous section to find the most likely subsequences for $OB(mps_{\text{final}} : l_{ob})$ on all the adjacent road segments. Among all these subsequences, we consider the one with the smallest similarity distance as the road segment that the user finally selects. The end point of this subsequence is the final location of the user.

7. Experiment Results

7.1. Data Collection. All the data employed in our experiments are real data. For route identification, we drove around Atlanta, USA, and Wuhan, China, to collect data. The sensor data for 48 unique routes were collected, with 32 routes in Atlanta and 16 routes in Wuhan. The lengths of the routes vary from about 1 kilometer to 3 kilometers. All the data were collected by iPhone 5, iPhone 5s, iPhone 6 plus, and iPhone 6s. For a specific route, we collected its data in at least consecutive 5 days. Thus, we have at least 5 sensor data profiles for each route. For localization in a particular route, we collected the data for a very long route. For localization in a bounded area, we collected the data of all the road segments in a limited area located at the Decatur county in Atlanta, USA.

7.2. Assumptions. It is obvious that the way a smart phone is placed in a car greatly affects the collected sensor data. In our work, we assume that the query data follow the same dimensions of the collected training data. Even though different users may place their smart phones in different ways, the similarity between the sensor data for different days of the same route does not change. In our experiments, the only requirement is that a volunteer places the smart phone the same way every day. Since it is easy to detect the position of a smart phone, we believe it is possible to project the sensor data into a uniform position coordinate system and this is out of the scope of this paper.

7.3. Route Identification. For route identification, 16 volunteers participated in collecting sensor data along their daily routes. We do not have any strict requirements about the start time. We find that it does not have much impact on the experiment results. Totally, we have 48 routes, some of which overlap with each other. For any pair of routes, the overlap rate is from 0 to 70%. However, we can still distinguish them efficiently. We also find that the longer the route is, the easier the route is distinguished from other routes, because the longer the route, the more unique the features it has. For each route, the volunteers are required to collect sensor data for at least a week so that we can evaluate the impact of the size of the training data.

For the 48 routes, we have a testing set and a training set. The sizes of the testing set and the training set are both 48. Each route has only one profile. We run our algorithm for each route in the testing set. If our algorithm can identify the route in the training set, we consider it as successful. Our success rate is 100% for identifying a route. It indicates our method is effective in identifying a route even if the routes may overlap with each other, as our method makes use of the data collected from 6 dimensions which vividly depict the unique features of a route. Our method outperforms the work in [18] that employs power footprint collected from the base station. As the number of the reference profiles increases, we obtain better results even if we try to identify more unique routes.

7.4. Localization in a Specific Route. For localization in a specific route, we randomly select a long route which is about 20 kilometers and collect 10 sensor data profiles for it. We choose one of the profiles as the training data. We randomly select another profile as the testing data. That means we only have one profile in each experiment. We split the testing data into several road segments as if they are collected from a user's smart phone in real time. First, we want to know whether our method could distinguish road segments. We are also interested in the impact of the number of profiles. The results are shown in Table 1. When there are 10 road segments, the success rate is 84.2%. When there are 5 road segments, the success rate increases to 90.3%. The main reason is that if there are only 5 road segments, each road segment is longer

Table 1: Road segment identification.

# of road segments	# of profiles	Ave. success rate
10	1	84.2%
5	1	90.3%
10	5	100%

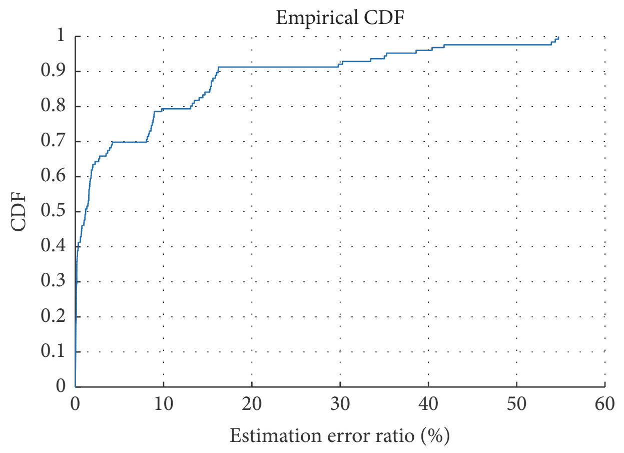


Figure 6: Estimation error cumulative distribution.

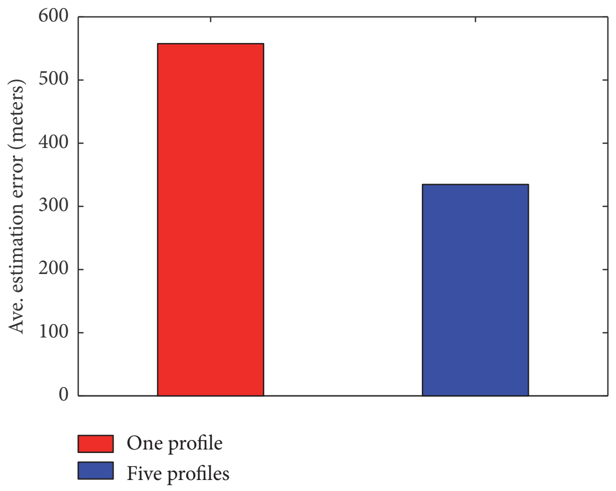


Figure 7: Localization accuracy with more training profiles.

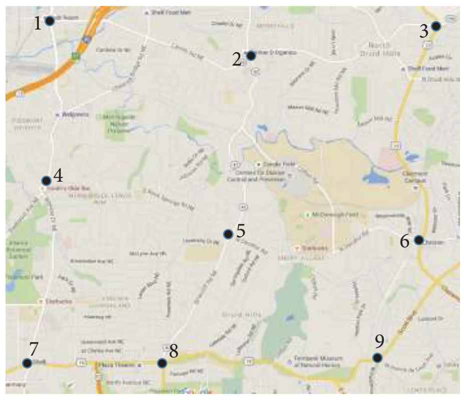


Figure 8: A bounded area.

and more unique so that it is easier to distinguish them. If we increase the number of the profiles to 5, the success rate increases to 100% even if there are 10 road segments.

By using our algorithm, it is easy to know which road segment a user is traveling through. However, we still want to locate a user more accurately. In our experiments, the minimum length of a subsequence is 1 minute, and the step size is 5 seconds. One of the 15 days' sensor data pieces is chosen as the testing data, and we randomly select another one as the training data. The total length of the route is 19 kilometers. The x-axis of Figure 6 shows the estimation error ratio with respect to the total length. It can be seen that almost 40% of the estimations are error free, and almost 80% of the estimations have an error less than 2 kilometers. Even though sometimes there are big errors, they can be avoided if we take time into consideration. Since we already know which route a user is traversing, based on the time information, we can narrow down the search space to avoid a big error.

For the impact of the size of training set, we fix the length of a subsequence as 4 minutes. We compute the location for each training route in the training set for the query subsequence; then the averaged location is regarded as the final estimation. As we increase the size of the training set, we can obtain a more accurate result. As shown in Figure 7, when we use more route profiles to localize a user, the average estimation error is reduced.

7.5. Localization in a Bounded Area. For localization in a bounded area, we collected data from an area shown in Figure 8. This area locates at the center of Atlanta, USA. We select 9 intersections and 12 segments determined by these intersections. That means, in the HMM model, we have 12 states. The average length of the road segments is about 3 kilometers. We assume the probability of a road segment to be the starting segment of a user is 1/12. There are many methods to determine transition probability. In our experiments, we adopt the simplest one. The transition probability for road segments is evenly distributed over all the possible transitions. For each road segment, we collected at least one profile. One of them is chosen as the testing data, and the rest are considered as the training data. The probability of some sensor data to be related to a specific road segment can be calculated by the DTW algorithm. Actually, this is the observation probability. Now, we have the initial probability, transition probability, and observation probability; then by using our method we can infer the route and location of a user.

First, we want to make sure that our method can successfully infer the route traversed by a user. For simplicity, we only consider one direction in the map which is from top left to bottom right. All the possible routes have been tested as listed in Table 2. We tested all the possible routes from intersection 1 to intersection 9. For a full route which means a car stops immediately after passing intersection 9, we want to know whether we can infer the route correctly. The results are shown in Table 2.

Table 2: Route inference.

Routes	# of profiles	Ave. success rate
1-2-3-5-9	3	95%
1-2-4-5-9	3	98%
1-2-4-8-9	3	100%
1-3-4-5-9	3	100%
1-3-6-8-9	3	100%

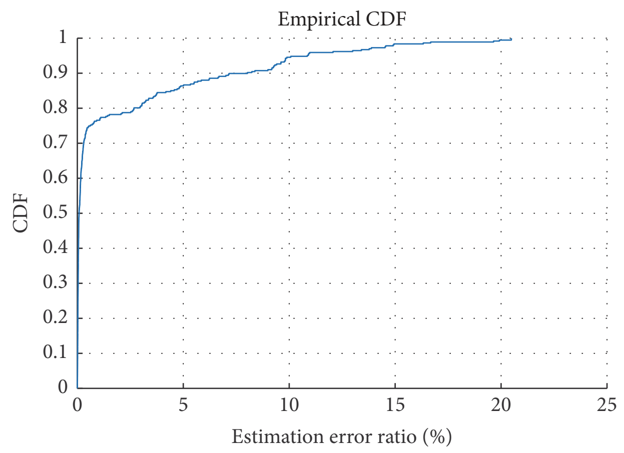


Figure 9: Estimation error for localization in a bounded area.

Basically, we can infer all the routes successfully; then we can know the final location of a user is at intersection 9. However, it is quite possible that a user may stop at any point in the area. The ultimate goal of our work is to infer a user's location. Thus, we also test some routes ending at any point in the area. Totally, we tested 200 subroutes of the full routes in the previous group of experiments. These subroutes are randomly taken from the full routes. The total length of each route is up to 11 kilometers. The idea is to infer the part of the route consisting of several complete road segments. We can get most possible intersection of a user. As we know, there are many possible associated road segments for each intersection. By using the method introduced in Section 5, we can compute the similarity distance between the testing partial road segment and all the possible roads. The one with the minimum similarity distance is the inferred location of a user. The results are shown in Figure 9. It can be seen that almost 65% of the estimations are error free, and almost 86% of the estimations have an error less than 0.5 kilometers.

8. Related Works

There are many previous works trying to acquire user's privacy information by analyzing sensor data collected from mobile devices. We roughly classify them into four categories based on the type of privacy information.

The first category focuses on deanonymizing mobile devices. Dey et al. [23] conducted extensive experiments to show that the accelerometer readings are identical for each user so that they can be used to infer user identity, while in [32], in order to identify an individual device, the speaker of a smart phone is used to construct the fingerprint of a user. Different from [32], the work in [33] proposed and implemented two approaches, one based on analyzing the frequency response of the speakerphone-microphone system and the other is based on studying device-specific accelerometer calibration errors to construct a fingerprint to deanonymize mobile devices. No matter what the embedded sensors they make use of to extract unique fingerprints of users, the works fully prove that user privacy is being threatened by smart phones.

The second category aims at getting users' location information. Without making use of GPS information, Han et al. [19] proposed an approach to locate users only based on accelerometer readings. Their method firstly tries to reconstruct motion trajectory given the acceleration measurements collected from a user's phone. Then it matches the constructed trajectory with the map information to infer the user's location. Their work is similar to ours. However, their method is mainly based on probability and statistic models which need tremendous background information. Thus, it may have limited ability to infer location. Similar to [19], Constandache et al. [34] try to make use of a smart phone's accelerometer and electronic compass to get the moving speed and the direction so that they can construct a directional trail which can be matched with the local area map. In this way, they can infer a user's location based on the best matched path segment. But they need to use GPS information to get the initial location which cannot be satisfied in many situations. Two works related to open permission sensors have been proposed by Michalevsky et al. In [18], they argue that a smart phone's location greatly affects the power consumed by the phone's cellular radio which is the most power-intensive part. Thus, they can use a mobile device's aggregated power consumption profile to learn the location information based on the cellular radio map. But the power consumption in smart phones can be affected greatly and many factors such as playing game affect localization accuracy heavily. Alzantot and Youssef [20] designed a step counting method based on a lightweight finite state machine to estimate the walking distance so that they can track pedestrians. Their method is too simple to deal with complicated scenarios. The most popular methods to get user's location indoors without using GPS component is to utilize the WiFi signal. Krumm and Horvitz [22] designed and implemented a system called LOCADIO to infer the motion and location of a user. This kind of works cannot work without WiFi device (outdoor). The work in [35] tries to explore the possibility of developing an electronic escort service by inferring the walking trail of a user. This work is not trying to get user's privacy information secretly. It requires users to share their location information with others which is not preferred by most users. Azizyan et al. [36] argue that logical location, which means location fingerprint characterized by surrounding sound, color, light, and so on can be captured by the embedded sensors in smart phones. They try to utilize location logical fingerprint matching to localize users indoors. It is obvious that this is infeasible outdoors. Different from all these works, our work is the first one that combines two kinds of sensor readings to infer a user's location information outdoors without using GPS information.

All other privacy issues were considered in category three. These works open an interesting way to make use of in-built sensors to poach privacy information. As in [37], the authors proposed a method to steal the acoustic signals by using gyroscope in a smart phone. The work in [16] studies the feasibility of getting a user's tap inputs through motion sensors embedded in cell phones. Accelerometer is used at [17] to infer if the user is taking a metro. In this paper, they first extract the feature of the accelerometer sensor data and then utilize supervised learning based classifier to infer the interval of riding a metro, while [38] focuses on inferring a user's private information in Android system leveraging the system bugs.

Some other miscellany works were grouped into category four. Attackers not only want to infer privacy information, but also try to do it efficiently [39]. In order to save energy, Yadav et al. [21] proposed their low cost GSM-based localization method based on Cell Broadcast Messages and war-driving. To tackle the problem that the Maximum Likelihood estimator for received signal strength (RSS) based localization is nonconvex, Ouyang et al. [40] proposed an Semidefinite Programming (SDP) relaxation technique to solve this problem. Further, even some works have been proposed to improve the service quality instead of getting privacy information from user. Actually, using the integrated sensor in phone to monitor the road condition and traffic problem has been proposed by Mohan et al. [41]; however, their work focuses on detecting rough road condition and traffic jam. In [42], the author argued that the slight localization error may cause inconvenient result, so they proposed using accelerometer signatures to mark user's location to place mobile phone in a right context. However, the accelerometer signatures were just used as side channel to give a more meaningful localization information for user when using GPS.

As we can see, most of the aforementioned related works either have strong assumptions about their application scenarios or have limited inference ability.

9. Conclusion

User privacy is being threatened by the sensors embedded in mobile devices, as these sensors may release data without users' awareness. In this paper, we show that a user's location information can be inferred by utilizing the sensory data collected from embedded sensors in users' mobile devices. We make use of the sensory data to construct fingerprints of routes and Dynamic Time Warping is employed to perform route inference. We address three issues including route identification, localization in a specific route, and localization in a bounded area. Real experiments were performed to evaluate our work. The extensive experiment results show that we can effectively identify routes and localize a user in a real time manner unconsciously.

Conflicts of Interest

There are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is partly supported by the NSF under Grant no. CNS-1252292 and the National Natural Science Foundation of China under Grants nos. 61502116 and 61370084.

References

[1] T. Song, N. Capurso, X. Cheng, J. Yu, B. Chen, and W. Zhao, "Enhancing GPS with lane-level navigation to facilitate highway driving," *IEEE Transactions on Vehicular Technology*, vol. 99, 2017.

[2] H. Zhong, J. Wen, J. Cui, and S. Zhang, "Efficient conditional privacy-preserving and authentication scheme for secure service provision in VANET," *Tsinghua Science and Technology*, vol. 21, no. 6, pp. 620–629, 2016.

[3] H. Huang, T. Gong, P. Chen, R. Malekian, and T. Chen, "Secure two-party distance computation protocol based on privacy homomorphism and scalar product in wireless sensor networks," *Tsinghua Science and Technology*, vol. 21, no. 4, pp. 385–396, 2016.

[4] L. Zhang, Z. Cai, and X. Wang, "Fakemask: a novel privacy preserving approach for smartphones," *IEEE Transactions on Network and Service Management*, vol. 13, no. 2, pp. 335–348, 2016.

[5] Z. He, Z. Cai, Y. Sun, Y. Li, and X. Cheng, "Customized privacy preserving for inherent data and latent data," *Personal and Ubiquitous Computing*, no. 1, pp. 1–12, 2016.

[6] X. Zheng, Z. Cai, J. Li, and H. Gao, "Location-privacy-aware review publication mechanism for local business service systems," in *Proceedings of the 36th Annual IEEE International Conference on Computer Communications (INFOCOM 2017)*, pp. 1–9, 2017.

[7] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, 2017.

[8] T. Qiu, R. Qiao, and D. Wu, "EABS: an event-aware backpressure scheduling scheme for emergency internet of things," *IEEE Transactions on Mobile Computing*, 2017.

[9] J. H. Ziegeldorf, O. G. Morchon, and K. Wehrle, "Privacy in the internet of things: threats and challenges," *Security and Communication Networks*, vol. 7, no. 12, pp. 2728–2742, 2014.

[10] K. Zhao, H. Jin, D. Zou, W. Dai, and Y. Xiang, "A privacy-preserving location tracking system for smartphones based on cloud storage," *Security and Communication Networks*, vol. 8, no. 3, pp. 446–458, 2015.

[11] Q. Yang, A. Lim, X. Ruan, X. Qin, and D. Kim, "Location-preserved contention-based routing in vehicular ad hoc networks," *Security and Communication Networks*, vol. 9, no. 9, pp. 886–898, 2016.

[12] L. L. J. L. Meng Han, Q. Han, and Y. Li, "Maximizing influence in sensed heterogenous social network with privacy preservation," *International Journal of Sensor Networks*, 2017.

[13] N. Capurso, T. Song, W. Cheng, J. Yu, and X. Cheng, "An android-based mechanism for energy efficient localization depending on indoor/outdoor context," *IEEE Internet of Things Journal*, 2016.

[14] Y. Zhao, "Mobile phone location determination and its impact on intelligent transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 1, pp. 55–64, 2000.

[15] M. Han, J. Li, Z. Cai, and Q. Han, "Privacy reserved influence maximization in gps-enabled cyber-physical and online social networks," in *Proceedings of the IEEE International Conference on Social Computing and Networking (SocialCom '16)*, pp. 284–292, Atlanta, Ga, USA, October 2016.

[16] Z. Xu, K. Bai, and S. Zhu, "TapLogger: inferring user inputs on smartphone touchscreens using on-board motion sensors," in *Proceedings of the 5th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pp. 113–124, Tucson, Ariz, USA, April 2012.

[17] J. Hua, Z. Shen, and S. Zhong, "We can track you if you take the metro:tracking metro riders using accelerometers on smartphones," preprint https://arxiv.org/abs/1505.05958.

[18] Y. Michalevsky, A. Schulman, G. A. Veerapandian, D. Boneh, and G. Nakibly, "Powerspy: location tracking using mobile device power analysis," in *Proceedings of the 24th USENIX Security Symposium (USENIX Security 15)*, vol. 15, pp. 785–800, 2015.

[19] J. Han, E. Owusu, L. T. Nguyen, A. Perrig, and J. Zhang, "Accomplice: Location inference using accelerometers on smartphones," in *Proceedings of the 2012 4th International Conference on Communication Systems and Networks, COMSNETS 2012*, January 2012.

[20] M. Alzantot and M. Youssef, "UPTIME: ubiquitous pedestrian tracking using mobile phones," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '12)*, pp. 3204–3209, IEEE, April 2012.

[21] K. Yadav, V. Naik, P. Singh, and A. Singh, "Alternative localization approach for mobile phones without GPS," in *Proceedings of the Middleware'10 Posters and Demos Track*, ACM, December 2010.

[22] J. Krumm and E. Horvitz, "LOCADIO: inferring motion and location from Wi-Fi signal strengths," in *Proceedings of the 1st Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS '04)*, pp. 4–13, Boston, Mass, USA, August 2004.

[23] S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi, "AccelPrint: imperfections of accelerometers make smartphones trackable," in *Proceedings of the Network and Distributed System Security Symposium (NDSS '14)*, San Diego, Calif, USA, February 2014.

[24] T. Qiu, A. Zhao, R. Ma, V. Chang, F. Liu, and Z. Fu, "A task-efficient sink node based on embedded multi-core soC for Internet of Things," *Future Generation Computer Systems*, 2017.

[25] S. Cheng, Z. Cai, J. Li, and X. Fang, "Drawing dominant dataset from big sensory data in wireless sensor networks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '15)*, pp. 531–539, April 2015.

[26] S. Cheng, Z. Cai, and J. Li, "Curve query processing in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 11, pp. 5198–5209, 2015.

[27] S. Cheng, Z. Cai, J. Li, and H. Gao, "Extracting kernel dataset from big sensory data in wireless sensor networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 4, pp. 813–827, 2017.

[28] N. Eagle, A. Pentland, and D. Lazer, "Inferring friendship network structure by using mobile phone data," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 106, no. 36, pp. 15274–15278, 2009.

[29] Online; accessed Dec-2016, https://developer.android.com/guide/topics/sensors/index.html.

[30] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *KDD Workshop*, vol. 10, pp. 359–370, Seattle, WA, USA, 1994.

[31] G. D. Forney, "The Viterbi Algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[32] A. Das and N. Borisov, "Poster: fingerprinting smartphones through speaker," in *Poster at the IEEE Security and Privacy Symposium*, Citeseer, 2014.

[33] H. Bojinov, Y. Michalevsky, G. Nakibly, and D. Boneh, *Mobile device identification via sensor fingerprinting*, https://arxiv.org/abs/1408.1416.

[34] I. Constandache, R. R. Choudhury, and I. Rhee, "Towards mobile phone localization without war-driving," in *Proceedings of the IEEE 29th Conference on Computer Communications (INFOCOM '10)*, pp. 1–9, San Diego, Calif, USA, March 2010.

[35] I. Constandache, X. Bao, M. Azizyan, and R. R. Choudhury, "Did you see Bob?: human localization using mobile phones," in *Proceedings of the 16th Annual Conference on Mobile Computing and Networking, MobiCom 2010*, pp. 149–160, ACM, September 2010.

[36] M. Azizyan, I. Constandache, and R. R. Choudhury, "Surround-Sense: mobile phone localization via ambience fingerprinting," in *Proceedings of the 15th Annual ACM International Conference on Mobile Computing and Networking (MobiCom '09)*, pp. 261–272, ACM, September 2009.

[37] Y. Michalevsky, D. Boneh, and G. Nakibly, "Gyrophone: recognizing speech from gyroscope signals," in *Proceedings of the 23rd USENIX Security Symposium (USENIX Security 14)*, pp. 1053–1067, 2014.

[38] X. Zhou, S. Demetriou, D. He et al., "Identity, location, disease and more: Inferring your secrets from android public resources," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, pp. 1017–1028, ACM, 2013.

[39] L. Zhang, X. Wang, J. Lu, P. Li, and Z. Cai, "An efficient privacy preserving data aggregation approach for mobile sensing," *Security and Communication Networks*, vol. 9, no. 16, pp. 3844–3853, 2016.

[40] R. W. Ouyang, A. K.-S. Wong, C.-T. Lea, and V. Y. Zhang, "Received signal strength-based wireless localization via semidefinite programming," in *Proceedings of the 2009 IEEE Global Telecommunications Conference, GLOBECOM 2009*, pp. 1–6, December 2009.

[41] P. Mohan, V. Padmanabhan, and R. Ramjee, "Nericell: rich monitoring of road and traffic conditions using mobile smartphones," in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys '08)*, pp. 323–336, ACM, Raleigh, NC, USA, November 2008.

[42] A. Ofstad, E. Nicholas, R. Szcodronski, and R. R. Choudhury, "Aampl: Accelerometer augmented mobile phone localization," in *Proceedings of the First ACM International Workshop on Mobile Entity Localization and Tracking in GPS-Less Environments*, pp. 13–18, ACM, New York, NY, USA, 2008.