Enhancing Mobile Social Network Privacy

Wei Chang, Jie Wu, and Chiu C. Tan
Department of Computer and Information Sciences
Temple University, Philadelphia, PA 19122
Email: {wei.chang, jiewu, cctan}@temple.edu

***Abstract*—Privacy is an important concern for location based services (LBSs). In this paper, we consider a specific type of LBS known as a *mobile social network* (MSN). We demonstrate a new type of attack, where an adversary can combine the location and friendship information found in a MSN, to violate user privacy. We propose a fake location reporting solution that does not require any additional trusted third party deployment. We use extensive simulations to determine the validity of our scheme.**

***Index Terms*—Closeness, location privacy, mobile social networks, trajectory estimation.**

I. Introduction

The increasing popularity of smartphones has led to the rise of *mobile social networks* (MSNs). A MSN is a combination of online social networks and location based services. A MSN can provide many new services, such as friends locator.

In order to provide such services, the MSN provider has to collect the location information from users and their friends. This has led to concerns that such information may pose a privacy threat, since users may be unaware that they have revealed some sensitive information until after the fact. One apparent technique to protect privacy is to give users more control over when their locations are updated. The intuition is that individuals are the best arbiters of what locations are private, and by allowing a user to *not* upload his location at such sensitive locations, location privacy is achieved.

However, this intuition may not adequately provide location privacy against an adversary that knows both the location information, as well as social relationships. To illustrate, consider the map shown in Fig. 1 (Left), where the center region is a hospital. A user may decide that the hospital is a sensitive area, and choose not to upload his location information when he is near the hospital. However, the adversary can use trace data collected over time to determine the positions where the user did update his location, and use that information to infer that the user did visit the hospital. The adversary can also use the user's friendship information to better refine the location prediction algorithm. In Fig. 1 (Right), we see that the user does not upload his location near the hospital, but his friend continues to do so. Since the adversary is aware of the friendship information, the adversary can use the friend's location information to fill-in the location gaps of that user.

In this paper, we consider the problem of providing location privacy against an adversary having access to the data collected by the MSN provider. Our contributions are: (1) We are the first to consider an attack where the adversary uses both the historical trace data and friendship information to predict a user's movements; (2) Our proposed solutions utilizes Kalman filters to determine the best fake location to upload so as to defeat the MSN's location prediction algorithm; (3) Our solution does not require the use of trusted third parties, which may be difficult to deploy in the real world, to provide location privacy protection.

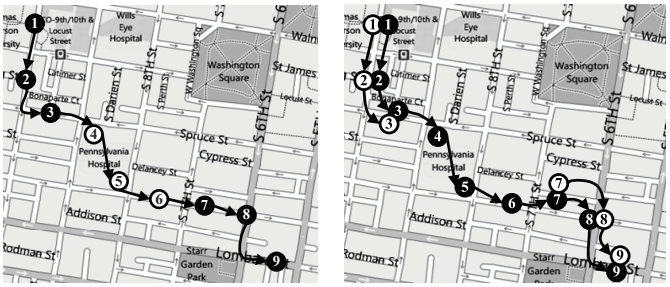


Fig. 1: Left figure: dark and light cells represent reported and unreported locations, respectively. The number in each cell indicates the observation sequence. Right figure: the trajectory is the path of two users, but one of the users does not report his location from time steps 4 to 6.

II. Related Work

Anonymity can be provided via the frequent changing of pseudonyms [1]–[4] such to make it difficult for adversaries to detect a user's movement. The system first defines several special regions named "mix-zones". In these fixed areas, a number of nodes enter the zone and another amount of nodes leave the zone. Within each mix zone, nodes change their pseudonyms at the same time, and do not report their locations in the zones. Because the transitions among nodes are simultaneous, the adversary cannot get the complete trajectories of users so that the mix zones preserve the nodes' location privacy.

However, for MSN-type applications, such as friend locator applications, the MSN provider will still have to know each pseudonym's friends, so as to deliver the correct information. As such, the frequent changing of user IDs is unlikely to mask the user's identity. Another technique of providing anonymity is to use k-anonymity techniques [5]–[8]. In such techniques, a user will only upload the location of a region, which contains $k-1$ neighbors. This will ensure that the adversary cannot pinpoint the exact user. The use of k-anonymity in a MSN is not feasible because the user's friends can only receive an approximate area, and cannot determine the location of the user. This will make a MSN less useful.

The other category of location privacy preserving techniques is obfuscation. For this category of techniques, the user will try

to confuse the adversary through techniques, such as injective noise [9]–[11], reporting fewer locations [12], reporting false locations [9] or increasing the intervals between reported locations [13]. Our approach follows this category.

The difference is that we consider a more powerful adversary who can utilize the *friendship* information to better predict user movements. Friendship information can be combined with location information to predict user movements.

III. Overview

Our system model consists of a MSN provider and many members. The adversary may control a MSN provider. Thus, the adversary has access to all of the location updates, and can use this information to filter out fake locations.

We divide time into discrete time units. A user's request can be generated at any time, but will only be transmitted to the MSN at a predefined time step. If there are multiple requests at a time step, the phone will only provide the nearest location. In order to protect users' privacy, at each time step, users can report either their real locations, fake ones or none. We assume that friends will be able to distinguish between a real or fake location, for instance via some unique IDs. Table I contains the notations used.

1) Kalman filter: The adversary will use the Kalman filter to estimate users' unreported locations. The Kalman filter is a set of mathematical equations that provide efficient computational (recursive) means to estimate the state of a process in a way that minimizes the mean of the squared error [14]. Forward and backward Kalman filling [15] is a technique, derived from the Kalman filter, to estimate the missing data of a linear system. It uses the Kalman filer's estimated states to represent the missing data. The Kalman filling algorithm first uses some reasonable data to pre-fill the missing data at corresponding time steps. Then, it applies the Kalman filter to the data, and obtains a set of estimated states of every time step. Finally, it replaces the pre-filled data by these estimations. *Forward/backward Kalman filling* uses the data in the ascending/descending order of time.

TABLE I: Table of notation

K	Kalman gain (updating ratio)
H	Observation model matrix
F	State transition model matrix
I	Identity matrix
Δd	Extra distance provided by a user at a time step
$I(L_A; L_B)$	Mutual information between two sets of locations

Algorithm 1 Tracking algorithm based on Kalman filling

1: Input: H_R reported locations at each time step, H_E external knowledge for pre-fill the unreported ones in H_R
2: Use H_E to pre-fill the empty items in H_R, obtained H_F
3: Apply forward Kalman filter to H_F, obtained H_{KF}
4: Apply backward Kalman filter to H_F, obtained H_{KB}
5: **for** Each unreported location in H_R **do**
6: Replace it by $\alpha * H_{KF} + \beta * H_{KB}$ (α, β: weighted values)
7: Return Kalman filling result H_R

2) Adversary location estimation: Algorithm 1 illustrates how an adversary estimates a user's location. Since the trajectories of humans are continuous and the moving pattern of human beings can be modeled by a linear process with the noise, the unreported locations in a single user's trajectory can be estimated by using forward and backward Kalman filling.

The adversary will first provide some synthetic locations to pre-fill the locations set, based on some outside knowledge. For example, the missing data can be linearly interpolated. Then, the interpolated data will later be used as measurements for the Kalman filter.

The adversary can use social relationships to predict a location as follows: the adversary can first determine a distance R_S. If the distance between user A and his friend, user B, is less than R_S, the adversary can regard them as being together. For all MSN friends of A, the adversary can calculate the percentage of being together with A. When A applies an unreported location in a protected trajectory, the adversary can guess that A stays with one of his friends. The partial traces of a user's friends can be used as an external location source to estimate the hidden locations during the use of Kalman filling.

3) Location privacy preserving metric: The evaluation metric is the average error between guessing results and real locations at each reporting time. An adversary's error degree can be calculated as $W_{adv} = \frac{\sum_T \|Loc_G, Loc_R\|}{\sum amount(T)}$, where W_{adv} represents the average mistake degree of an adversary's guessing; $\| \bullet \|$ represents Euclidean distance between a guessing location Loc_G, and its corresponding real location Loc_R; $amount(T)$ represents the total amount of reported locations. If there is no fake or unreported location in the observations, the value of W_{adv} is 0. The higher the adversary error degrees are, the safer users' location privacy is.

IV. Proposed Solution

A. User traveling alone

Here, the user is traveling alone. When choosing fake locations to update, he wants to select locations that cannot be easily filtered out by the adversary.

1) Relationship between estimation error and fake location distance: Suppose that in order to protect the privacy of his own trace, a user wants to use N% locations as fake locations. We assume that the user uses fake location $(x + \Delta x, y + \Delta y)$ at time k, where x and y represent the real location. Δx and Δy can be thought as user specified *noise*. Now, we want to calculate the relation between fake locations and their corresponding estimated locations by Kalman filter. The estimation error of the Kalman filter can be evaluated by a covariance matrix $P_{k|k}$, which stores the covariance of the state at time k based on observation at k.

$$P_{k|k} = (I - P_{k|k-1}H^T(HP_{k|k-1}H^T + R)^{-1}H)P_{k|k-1}, \quad (1)$$

where I is an identity matrix, $P_{k|k-1}$ is the covariance of the state at time k based on the past observation at time $k-1$, H is an observing matrix, R is the covariance of noise, k is a time instance and H^T is the transposed matrix of H.

If the covariance of *noise* in a user's historical location set is relatively larger, the covariance between observation and prediction will be greater. In other words, the further a fake location is from the real location, the greater the estimation error will be.

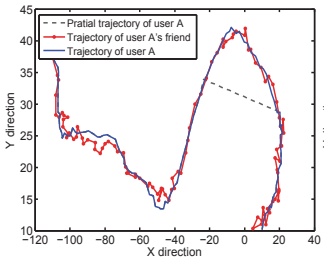


Fig. 2: Two users travel together

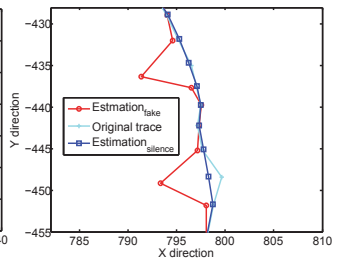


Fig. 3: One user defense

2) Estimation error of Kalman filter: The Kalman gain K determines the trust between a predicting system to its observation, and its predicting result [14]. Under conditions where R is constant, both the estimation error covariance $P_{k|k}$ and K will stabilize quickly and remain constant [14].

Now, we consider the total estimation error led by *noise* in a 1-D space. Assume that we add Δd to one location in x or y direction. Then, the additional estimation error Δe in the following n^{th} step can be represented as follows:

$$\Delta e = ((I - KH)F)^n K \Delta d, n = 0, 1, 2, ..., \quad (2)$$

where I is an identity matrix, K is the Kalman gain, H is an observing matrix and F is the state transition model. The trace of a user can be divided into x and y directions, which means the moving trace of a user can be seen as the combination of two traces from the 1-D space.

The Kalman filter has the capacity to deal with noise because the expected value of noise is zero. As a result, if we provide *noise* in the same direction to a real location, the influence of fake locations can be accumulated and disturb the estimated result of the Kalman filter. However, the value of Δe will decrease exponentially when n goes up: If we provide a fixed value Δd to a group of traces, the extra estimation error is a fixed value: $\sum_n ((I - KH)F)^n K \Delta d$. We use $\lambda \Delta d$ to represent this fixed value. Suppose that we use N% of our reported locations as the fakes. Since a fake location can be considered as a real location adding Δd_i, the average estimation error of a trace is $N\% \times \lambda \overline{\Delta d_i}$.

3) Fake locations reporting pattern: There are two options when reporting fake locations: 1. Concentratively report fake locations in some time, then continually report the real ones; 2. Provide the fake locations separately. The estimation error of Kalman filter at a place is the sum of errors caused by previous fake locations. However, as the parameter decreases rapidly, an intensive reporting of fake locations will cause the remainder of the locations to be less protected. Therefore, we will separately report fake locations to protect the whole trace. A user will only provide intensive fake location updates when protecting a specific sensitive location, not the entire trace. Hence, if a user wants to protect a certain location, he can intensively report some fake locations. As for the condition of using fake locations to distort the whole trajectory, it is better to report the fake locations separately.

4) The optimal fake location: Users' moving trajectories are a group of points in a two-dimensional space. To find the best fake locations at each point means to find the maximum value of $\Delta x^2 + \Delta y^2$ with speed restrictions. Suppose that (x_0, y_0), (x_1, y_1) and (x_2, y_2) are three consecutive points in a trajectory. Assume that (x, y) represent the corresponding fake location to point (x_1, y_1). With the limitation of speed, the maximum distance a user can reach in the time interval is R. The fake location (x, y) should satisfy the following:

$$R = S_{max} * \Delta t; \quad (3)$$

$$(x-x_0)^2+(y-y_0)^2 \leq R^2; (x-x_2)^2+(y-y_2)^2 \leq R^2; \quad (4)$$

$$Goal : maximize(\sqrt{(x-x_1)^2+(y-y_1)^2}) \quad (5)$$

Normally, a user makes a U-turn far less than others. In order to reduce the complexity of the algorithm, we always use the intersection point of two speed limitation circles as the best fake location. Hence, the best fake location (x, y) can be computed as follows:

$$a = (\frac{y_2 - y_0}{x_2 - x_0})^2 + 1 \quad (6)$$

$$q = \frac{(x_2 - x_0)(x_2 + x_0) + (y_2 - y_0)(y_2 + y_0)}{2(x_2 - x_0)} - x_0 \quad (7)$$

$$b = -2(y_0 + \frac{y_2 - y_0}{x_2 - x_0} q), c = y_0^2 - R^2 + q^2 \quad (8)$$

$$(x, y) = (q + x_0 - \frac{y_1(y_2 - y_0)}{x_2 - x_0}, \frac{-b + \sqrt{b^2 - 4ac}}{2a}), \quad (9)$$

$$or (x, y) = (q + x_0 - \frac{y_2(y_2 - y_0)}{x_2 - x_0}, \frac{-b - \sqrt{b^2 - 4ac}}{2a}) \quad (10)$$

However, the best fake locations are obtained based on the assumption of knowledge of the next location. We assume that there is an application that can predict the next position (x_2, y_2) by using a current real location and a current instantaneous speed, or users may provide the next position intended for hiding current sensitive locations.

Among all of the optimal fake locations of a trace, we note that the fake locations, which bring more errors to Kalman filling, are always the turning points' corresponding fake locations. In our method, we let each user's application record the distribution of the distances between each of the real locations and their corresponding fake ones. Therefore, reporting N% fake locations means finding a distance threshold D such that $\sum Dist(d \geq D) = N\%$. At each time interval T, if the distance between a fake and real location is greater than D, the application will report this fake location, otherwise it will report the real location.

Algorithm 2 Fake location generation algorithm

1: Input: Previous location (x_0, y_0), current location(x_1, y_1), current speed S_C, time interval T, maximum speed S_{max}
2: Use (x_1, y_1) and S_C to predict next location (x_2, y_2)
3: Compute the two potential fake locations based on formula 9-10, and calculate the distances from them to (x_1, y_1)
4: Return the furthest potential fake location (x, y)

Algorithm 3 Single user trace protection

1: Input: Real location (x_1, y_1), fake reporting percentage N, fake distance distribution $Dist$ and candidate fake location (x, y)
2: Compute the distance Δd between (x_1, y_1) and (x, y)
3: **if** $\sum_d Dist(d > \Delta d) \leq N$ **then**
4: Return the fake location (x, y)
5: **else**
6: Return the current location (x_1, y_1)

B. Users traveling in a group

The probability of two users appearing together is proportional to the closeness of them. We use the distance distribution between two users (A and B) to measure the closeness of them. We apply the concept of mutual information to analyze the problem. The historical location sets of A and B can be represented by two random variables L_A and L_B. We use $I(L_A; L_B)$ to represent the mutual information between A and B's locations sets. A high amount of mutual information between two users means that they always appear in the same regions at the same time.

The closeness of two users is subjective. We use a predefined threshold to determine the state of *being together*. We first compute the distance's distribution of two users. Then, we calculate the probability that two users appear in the same region at the same short period of time, based on a predefined threshold. The probability represents the chance that the two users stay together.

According to Information Theory, mutual information also represents how much the uncertainty in guessing L_B has been removed by knowing L_A. Therefore, if user A reports his location at time t while B doesn't, the uncertainty of those unreported locations of B will be reduced since adversaries can get more information from A's locations. According to the definition of mutual information, by increasing the uncertainty of getting together, or by increasing the probability of arriving at some location alone, the location-binding mutual information between users will become less.

A high amount of mutual information between A and B also means a high value of $Prob(L_B|L_A)$. The adversary can guess the position of B at a certain time t by $Prob(L_B) = Prob(L_B|L_A) \times Prob(L_A)$. If we can increase the uncertainty of A's reported locations, then B's locations can be protected better, and vice versa.

Our fake location generating method works as follows: when the distance between A and his friend, B, is less than a predefined *being together* distance threshold, the two users will report their real location and fake location at the same time, in turns. For example, A reports his real location at time t_1 and B reports a fake location. Then, at time t_2, B reports the real location and A reports the fake one. Through this method, we reduce the mutual information between the two users, and also increase the uncertainty of reported locations.

V. Evaluation

To perform the simulations, we established a Cartesian coordinate system where the point $(0, 0)$ is the beginning of a user's trajectory. We set the initial user speed to be 1 unit of distance per unit of time, with a maximum speed of 7 units. At each step, we provide a normal distributed noise as the state noise, and another normal distributed noise as observing noise. We use the state noise to represent the speed and moving direction change of a user. Both observation noise and state noise are used to randomize the movement change pattern of a user. The average speed of a user is about 3.5 units of distance per time. In experiments, we vary observing time, continuous unreported locations and the closeness between two friends.

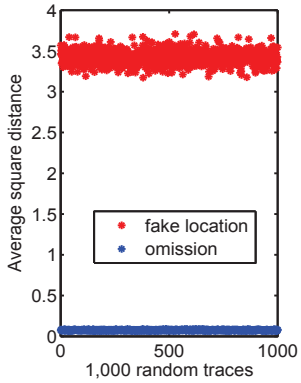
1) Single user results: We compare our scheme against the *location omission* strategy (the user will pick $N\%$ locations and not report them), as shown in Fig. 3. The figure shows the original trajectory of a user and the trajectory estimation results before and after using our fake locations-based method. Fig. 3 is a part of a random trajectory consisting of 2,000 points. We select 10% as a fake location reporting ratio. Fig. 3 shows that the Kalman filling result is inaccurate by our choice of fake locations.

For evaluating the effectiveness of our method to different random trajectories, we conducted the following simulation: we fixed the maximum speed and the initial position, left the other parameters randomized and generated 1,000 different trajectories. In this experiment, each trajectory was observed 2,000 times. The average speed of these traces vary from 2.73 to 5.39 units of distance per time. Fig. 4 shows the average estimation error per observation time between protecting via fake locations and omission, and we can see that our method can significantly improve the traces' privacy of users.

Next, we want to show that our method is always effective, regardless of the length of adversaries' observations. We generate 1,000 random traces, and the observing times of each trace changes from 100 to 10^5. The fake reporting ratio and the maximum speed are the same as the previous setting. Fig. 5 shows the average error's change pattern as time goes on: the average distance approaches a constant number. This result agrees with our research result in Section IV.

2) Two users results: The adversary may compute the closeness between two users using a statistic closeness metric. When the distance between two users is less than a threshold, we regard them as being together. If the closeness of two users is $k\%$, then when one user does not report his locations, there is $k\%$ probability that he is near his friend.

In Fig. 7, we let the number of continuous unreported locations vary from 1 time step to 50 time steps. We compute the average estimation error among 1,000 random trajectories, and to each trajectory, we observed 1,000 steps. We found



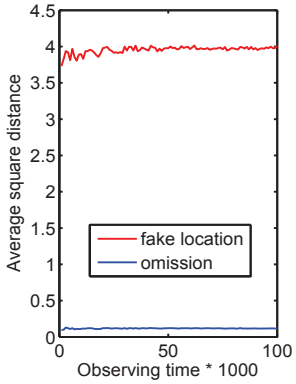


Fig. 4: Error by fake or omission *Fig. 5:* Error vs. observing time

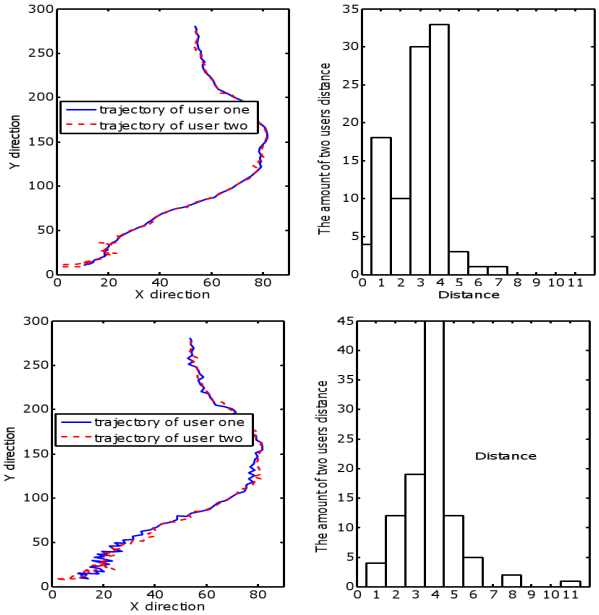


Fig. 6: The trajectories and distance histogram between two friends before and after using our proposed method. Upper left is the trajectories before using fake locations. Down left is the result after using fake locations. Upper right and down right show the difference between users' closeness before and after using our proposed method.

that when users do not report their locations in a short period of time, Kalman filling has better estimation results, while, if users do not report their locations for a long time, using the position of users' friends has a better estimation result.

Next, we illustrate the results before and after using our two users privacy-preserving method. Fig. 6 upper left shows the original trajectories of two users, and the upper right picture is the histogram of their distance. If we defined the distance between them as less than 4 units of distance apart, then 64% of their locations are satisfied. However, after using our method, the percentage reduces to 38%.

In order to see the relation between our method and the closeness threshold, we tested 1,000 pairs of users. Each pair was observed 1,000 times. The average speed of these tested trajectories varies from 2.82 to 3.76 units of distance per time. The maximum speed is set to 5 units of distance per time. Fig. 8 shows the change pattern of average estimation error

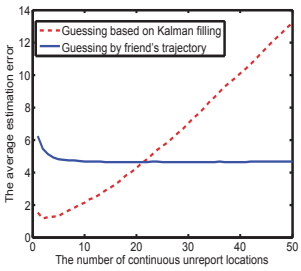


Fig. 7: Error vs. unreported num.

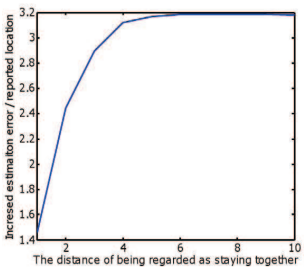


Fig. 8: Error vs. threshold

when closeness distance standard changes from 1 to 10 units of distance. As the adversary increases the threshold, his guessing result will depend more on the locations of the user's friends, since the closeness of the users will be increased, leading to the estimation error increase.

VI. Conclusion

In this paper, we consider the problem of improving location privacy for MSNs. We introduce a new privacy attack where the adversary uses both historic movements and friendship information to estimate a user's trajectory. Our solution allows a user to upload fake locations to protect his privacy. In our future work, we intend to consider incorporating mobility models, such as Levy-walk, to improve our scheme.

Acknowledgments

This research was supported in part by NSF grants ECCS 1128209, CNS 1065444, CCF 1028167, CNS 0948184, and CCF 0830289.

References

[1] M. Gruteser, "Enhancing location privacy in wireless LAN through disposable interface identifiers:A quantitative analysis," in *WMASH 2003*.
[2] L. Huang, K. Matsuura, H. Yamane, and K. Sezaki, "Enhancing wireless location privacy using silent period," in *IEEE WCNC 2005*.
[3] J. Freudiger, M. Manshaei, J. Hubaux, and D. Parkes, "On non-cooperative location privacy: a game-theoretic analysis," in *ACM CCS 2009*.
[4] T. Jiang, H. J. Wang, and Y.-C. Hu, "Preserving location privacy in wireless LANS," in *Mobisys 2007*.
[5] L. Kulik, "Privacy for real-time location-based services," *SIGSPATIAL Special*, vol. 1, July 2009.
[6] C.-Y. Chow and M. F. Mokbel, "Privacy in location-based services: a system architecture perspective," *SIGSPATIAL Special*, vol. 1, July 2009.
[7] Y. Ouyang, Y. Xu, Z. Le, G. Chen, and F. Makedon, "Providing location privacy in assisted living environments," in *PETRA 2008*.
[8] M. L. Damiani, E. Bertino, and C. Silvestri, "Protecting location privacy against spatial inferences: the PROBE approach," in *ACM SPRINGL 2009*.
[9] H. Kido, Y. Yanagisawa, and T. Satoh, "An Anonymous Communication Technique using Dummies for Location-based Services," in *ICPS 2005*.
[10] J. Krumm, "Inference attacks on location tracks," in *PERVASIVE 2007*.
[11] B. Hoh and M. Gruteser, "Protecting Location Privacy Through Path Confusion," in *Securecom 2005*.
[12] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Preserving privacy in GPS traces via uncertainty-aware path cloaking," in *ACM CCS 2007*.
[13] ——, "Enhancing Security and Privacy in Traffic-Monitoring Systems," *IEEE Pervasive Computing 2006*.
[14] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," Chapel Hill, NC, USA, Tech. Rep., 1995.
[15] L. S. Tekumalla and E. Cohen, "A hole-filling algorithm for triangular meshes," School of Computing, University of Utah, Tech. Rep., 2004.