This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2014.2310471, IEEE Transactions on Parallel and Distributed Systems



# Cooperative Positioning and Tracking in Disruption Tolerant Networks

Wenzhong Li, *Member, IEEE,* Yuefei Hu, *Student Member, IEEE,* Xiaoming Fu, *Senior Member, IEEE,* Sanglu Lu, *Member, IEEE,* and Daoxu Chen, *Member, IEEE*

**Abstract**—With the increasing number of location-dependent applications, positioning and tracking a mobile device becomes more and more important to enable pervasive and context-aware service. While extensive research has been performed in physical localization and logical localization for satellite, GSM and WiFi communication networks where fixed reference points are densely-deployed, positioning and tracking techniques in a sparse disruption tolerant network (DTN) have not been well addressed. In this paper, we propose a decentralized cooperative method called *PulseCounting* for DTN localization and a probabilistic tracking method called *ProbTracking* to confront this challenge. PulseCounting evaluates the user walking steps and movement orientations using accelerometer and electronic compass equipped in cellphones. It estimates user location by accumulating the walking segments, and improves the estimation accuracy by exploiting the encounters of mobile nodes. Several methods to refine the location estimation are discussed, which include the adjustment of trajectory based on reference points and the mutual refinement of location estimation for encountering nodes based on maximum-likelihood. To track user movement, the proposed ProbTracking method uses Markov chain to describe movement patterns and determines the most possible user walking trajectories without full record of user locations. We implemented the positioning and tracking system in Android phones and deployed a testbed in the campus of Nanjing University. Extensive experiments are conducted to evaluate the effectiveness and accuracy of the proposed methods, which show an average deviation of 9m in our system compared to GPS.

**Index Terms**—Disruption tolerant network, positioning, tracking, cooperation

## 1 Introduction

Disruption tolerant networks (DTNs) are sparse mobile ad hoc networks where nodes connect with each other intermittently [1]. Since DTNs allow people to communicate without network infrastructure, they are widely used in battlefields, wildlife tracking, and vehicular communications [2]. Location information is extremely important to enable context-aware and location-based applications [3]. However, due to the lack of fixed infrastructure and continuous network connection in DTNs, identifying the location of mobile users and tracking their movement trajectories are challenging.

The following scenario illustrates the localization problems in DTNs. Assume a DTN is formed by a set of wireless *nodes* (e.g., cellphones) moving within a field. Each node has a communication range of distance $r$ ($r > 0$). Two nodes can communicate when they move into each other's communication range, which is called an *encounter* of nodes. Since DTNs are sparse and highly dynamic, a constant communication path does not exist between any pair of nodes. As illustrated in Fig. 1, there are four different components in the system. The *landmarks* represent fix-deployed infrastructures like WiFi access points (APs), which can provide network service. An *infostation* is a server connecting to the APs to collect information from mobile nodes. The *GPS-nodes* are high-end mobile devices equipped with Global Positioning System (GPS). There are only a few of them in the network and they can be used as mobile reference points. The *common-nodes* are ordinary mobile phones without GPS support, which have the majority number in the system. They are only equipped with simple sensors (such as accelerometer and electronic compass), and can communicate with other nodes via WiFi or Bluetooth occasionally.

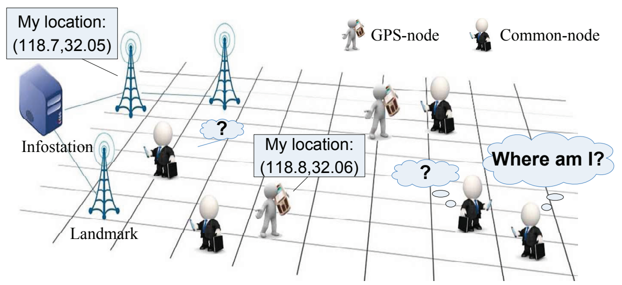



Fig. 1. The components of a DTN localization system.

The *positioning and tracking problem* in DTNs is twofold: the common-nodes (without GPS module) need to determine their locations based on the limited number of reference points (APs or GPS nodes) they encountered; and the infostation needs to track the trajectories of the common-nodes with the partial information collected by the APs opportunistically.

- *W. Li, Y. Hu, S. Lu, and D. Chen are with the State Key Laboratory for Novel Software Technology, Nanjing University, Xianlin Street 163, Nanjing 210023, China. E-mail: lwz@nju.edu.cn*
- *X. Fu is with the Institute of Computer Science, University of Goettingen, Goldschmidtstr. 7, Goettingen 37077, Germany. E-mail: fu@cs.uni-goettingen.de*

Early positioning systems rely on triangulation using physical signals from the fixed-deployed infrastructures such as GPS satellites [4], [5] and GSM celltowers [6], [7]. WiFi-based localization strategies collect the radio fingerprints quantified from the WiFi signal strengths at many physical positions and multiple APs, and identify user location by retrieving and matching the fingerprints [8], [9]. Such methods either require densely-deployed infrastructures or they need to collect a large amount of signal samples, which cannot be applied to sparse networks.

Several recent research focuses on GPS-free localization in wireless networks by incorporating fixed landmarks and surrounding characteristics. SurroundSense [10] identifies logical location using the surrounding information like sounds, lights and colors. CompAcc [11] adopts a distance estimation method using accelerometer and compass and determines location by matching to possible path signatures generated from an electronic map. Escort [12] provides a logical navigation system to help a person navigate to another person in a public place with the aid of context features. However, these methods need continuous communication with a centralized server to process a large amount of surrounding data, which are not suitable for the decentralized structure and the opportunistic communication nature of DTNs.

In this paper, we propose a decentralized cooperative method called *PulseCounting* for DTN localization and a probabilistic method called *ProbTracking* to track the movement of mobile nodes. PulseCounting evaluates the number of user walking steps using the accelerometer data, and decides the orientation of each step using the electronic compass measurements. By accumulating the segments of walking steps, it is able to form an estimation of current location. PulseCounting further takes advantage of the opportunity of encounters in DTNs to refine the location estimation: on the one hand, the encountering APs and phones equipped with GPS could be regarded as reference points; on the other hand, the encounters of two mobile nodes enable the possibility of mutual adjustment to reduce estimation error. ProbTracking detects the movement trajectory based on the partial location information reported by the other mobile nodes. It constructs a Markov chain using the movement history data and uses it to determine the most probable user walking route without the need for global location information in DTNs. We implemented the positioning and tracking system in Android phones, and deployed a testbed in the campus of Nanjing University for performance evaluation. Experiments show that the system has an average deviation of 9m compared to GPS.

## 2 Related Work

Disruption tolerant networks (DTNs) have been widely studied in the last decade. Most existing works focus on the fundamental problem of data routing in DTNs. To achieve data transmission without the need of end-to-end communication paths, several mobility-assisted routing strategies have been proposed to reduce the number of hops, the delivery delay and energy consumption [1], [2], [13], [14]. A few works addressed the issues of selfish behavior of nodes to enhance the cooperation for data relays in DTNs [15], [16]. Different from the existing works, this paper focuses on the issues of positioning and tracking mobile nodes in DTNs, which have not been well addressed in the past.

Previous research on wireless localization rely on deploying wireless infrastructures (e.g. telecommunication satellites or cell towers) and installing dedicated hardware (e.g. GPS modules or RFIDs) in the environment [17], [4]. In these systems, mobile devices measure the wireless signals to several infrastructures in known locations and estimate the actual locations based on their geometric relationships.

Cell tower triangulation is a popular technique for determining the location of a mobile device [6], [7]. Locating the position of mobile phones by measuring signals to GSM celltowers was studied in [7], which showes that GSM devices can achieve a positioning accuracy with a median error of 94-196 meters.

WiFi-based strategies rely on deploying fixed Access Points (APs) and require calibrating WiFi signal strengths at many physical positions to enable localization. RADAR [8] constructs detailed radio fingerprints of the available APs and combines empirical measurements with signal propagation modeling to determine user location . Place Lab [9] allows commodity hardware clients like PDAs and cell phones to locate themselves by listening for radio beacons of WiFi and GSM cell towers. It generates a radio map by war-driving and estimates the location of mobile devices by looking up the overhead WiFi/GSM beacons in the radio map. Several indoor localization approaches using WiFi signals was discussed in [18], [19], [20].

In the recent years, a couple of works address the issues of localization using fixed landmarks and surroundings. SurroundSense [10] identifies a user's location using the surrounding information collected by sensors and camera on mobile phones. The main idea is to fingerprint the location based on its ambient sound, light, color, RF, as well as the layout-induced user movement. However, it can only obtain a user's logical location like in Starbucks or McDonalds, but fails to provide the geographical coordinates. AAMPL [21] introduces a location estimation method using accelerometer and compass. It can estimate rough physical coordinates of mobile phones augmenting with context-aware logical localization. To improve location accuracy, CompAcc [11] uses the similar estimation method like AAMPL, and refines the location estimation by matching it against possible path

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2014.2310471, IEEE Transactions on Parallel and Distributed Systems



signatures generated from a local map. It achieves a location accuracy of less than 11 meters. However, it needs to construct path signatures from electronic maps beforehand, which is complex and time-consuming. Escort [12] provides a logical navigation system for social localization. Its goal is not to identify the physical location, but to help a person navigate to another person in a public place such as a hotel. By periodically learning the walking trails of different individuals, as well as how they encounter each other in space-time, a route is computed between any pair of persons. However, it needs global information of users' movements and their encounters to construct the navigation graph, which does not apply for DTNs.

## 3 Cooperative Positioning in DTNs

In DTNs, most of time the common-nodes have no GPS-nodes and landmarks within their communication range, which makes them hard to decide their locations. We propose the *PulseCounting* method for Cooperative Positioning in DTNs, which consists the following six steps.

### 3.1 Bootstrapping

As the first step, each node needs to know its position initially. Without the initial position, there is no reference point for location estimation. In DTNs, we assume a small number of fixed landmarks (e.g., wireless APs) are deployed in the environment with known locations. We also assume that there are a few GPS-nodes willing to report their locations to other nodes. Thus the common-nodes can obtain a rough initial location when they firstly encounter the landmarks or GPS-nodes. It is unlikely for all common-nodes to obtain their initial locations at the same time, so the initialization process is asynchronous.

With the initial location information, a map in this area will be downloaded to the user's cellphone. We use the Google Map [22] in our implementation since it provides open access to its data and APIs. The map is downloaded opportunistically when the device has a chance to access the Internet (i.e., entering the communication range of an AP). Unlike the existing positioning systems such as AAMPL [21] and CompAcc [11], the proposed PulseCounting method does not rely on the Map data to aid localization. It purposes just to help visualizing the movement trajectory on the cellphone screen.

### 3.2 Step counting

We introduce the method of using the accelerometer to measure walking steps. The accelerometer records user movement in three dimensions: X (the direction of front and back), Y (the direction of left and right), and Z (the direction of up and down). As illustrated in Fig. 2, we plot the accelerometer data of a users with the cellphone putting in three different positions: holding horizontally in hand, sticking vertically in front pocket, and sticking vertically in back pocket.

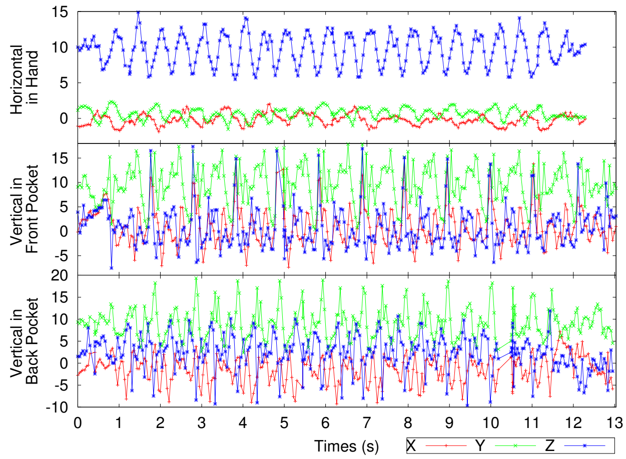



Fig. 2. The accelerometer measurement of a user.

Observation to Fig. 2 reveals several characteristics: (1) The acceleration is non-uniform. It shows a pattern of "increase-decrease" and fluctuates around some value. (2) The data is noisy. It is influenced by the way people walks and the position of their cellphones. (3) It has obvious periodicity and its shape looks like a wave. The periodical phenomenon is most clear in the direction of Z axis (up and down) of all cellphone positions. During movement, the human's center of gravity goes up and down, which causes the increasing and decreasing of his accelarations. Thus a period in the accelerometer reading corresponds to two walking steps in reality.

Based on the periodical characteristics of acceleration, we can estimate the moving distance by counting the number of steps similar to the method of [11]. The idea is simple: if the number of walking steps $m$ and the length of a step $L$ are known, the distance can be estimated by $S = m * L$. Since one period consists of two walking steps, if $\mathcal{P}$ periods in the accelerometer measurement are detected, $m$ can be approximated by $m = 2\mathcal{P}$. The step size $L$ differs from person to person. We assume $L$ is measured by users and it is known ahead. In our implementation, we let users walk through a fixed-length straight road for multiple rounds to calculate their average step lengths $\overline{L}$. For more general expression, we denote the step size of each user as a random variable and assume it follows the *Gaussian distribution*: $L \sim N(\overline{L}, \sigma^2)$, where $\overline{L}$ is the mean value and $\sigma^2$ is the variance.

### 3.3 Direction mapping

The other important aspect of movement is direction, which can be measured by electronic compass. The cellphone compass records the users orientation in the form of an angle with respect to magnetic north. Similar to the accelerometer data, the compass data is densely sampled (about 22 data per seconds) and appears fluctuating and noisy, thus it cannot be used directly. We proposed the *direction mapping* method to make the compass data discrete. For a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2014.2310471, IEEE Transactions on Parallel and Distributed Systems



rough estimation, we project the compass data to eight discrete directions: $North$, $Northeast$, $East$, $Southeast$, $South$, $Southwest$, $West$, and $Northwest$, which are numbered by $0-7$ accordingly. Assume $\Upsilon$ is a reading of the compass taking the value from [0, 360], we calculate its direction mapping by

$$(argmin_k\ |\Upsilon - \frac{360}{8}k|)\ mod\ 8. \quad (1)$$

With Eq. (1), a compass reading is mapped to a direction with the angle within the departure from $-22.5°$ to $+22.5°$. For example, the value $\Upsilon = 340°$ will be mapped to 0, which represents the direction "North"; $\Upsilon = 100°$ will be mapped to 2, which represents the direction "East".

To reduce the noise and fluctuation of compass measurement, we use the latest $K$ compass readings to decide the movement direction. After mapping all the $K$ readings, if a direction has the majority number, it will be taken as the movement direction. $K$ is set to 22 in our system.

Mapping to eight discrete directions is a rough estimation. It implies that the system can tolerate at most $\pm 22.5°$ of deviation in the movement direction. One can easily extend it to more fine-grained expression such as 16 or 32 directions.

### 3.4 Trajectory generation

With the results from step counting and direction mapping, we are able to describe user movement trajectories. A movement trajectory is defined as a series of segments with distance and direction:

$$T(P_0 \rightarrow P_1) = \{< S_1, \theta_1 >, \cdots, < S_M, \theta_M >\}, \quad (2)$$

where $P_0$ is the departure point and $P_1$ is the destination point of the trajectory. Each tuple $< S_i, \theta_i >$ $(i = 1, 2, \cdots, M)$ indicates a segment of the movement. $S_i$ is the moving distance of two consecutive walking steps (one period of the acceleration); $\theta_i$ is the movement direction (measured by the angle to the north) in the steps, which is obtained by the direction mapping method. $M$ is the total number of segments.

### 3.5 Location estimation

Given a trajectory $T(P_0 \rightarrow P_1)$, if the location of departure point $P_0$ is known, we can roughly estimate the location of $P_1$ by accumulating the trajectory segments. Assume the coordinate of $P_0$ is $(x_0, y_0)$. According to the segments, the user moves horizontally in the total displacement $\sum_{i=1}^{M} S_i \sin\theta_i$, and moves vertically in the total displacement $\sum_{i=1}^{M} S_i \cos\theta_i$. So the location coordinate of $P_1$ is approximated by

$$(x_0 + \sum_{i=1}^{M} S_i \sin\theta_i, y_0 + \sum_{i=1}^{M} S_i \cos\theta_i). \quad (3)$$

Theoretically, if the initial location of a node is known, we can estimate its location at any time with

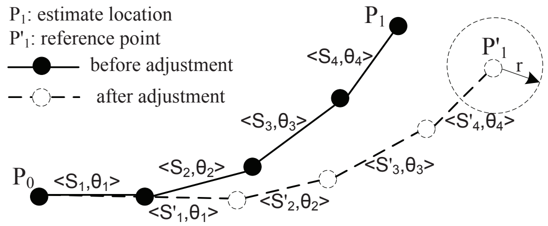



Fig. 3. Trajectory adjustment.

Eq. (3). However, due to the inaccuracy of step size and orientation measurement, errors may be introduced during the estimation of each segment. With the number of segments increase, the errors are accumulated, thus the estimated location will be far away from the actual location. To overcome this drawback, we use the encounter opportunity of nodes to improve the estimation accuracy, which is introduced in the following subsection.

### 3.6 Refinement

As mentioned previously, the measurement of user's step size is a random variable following Gaussian distribution: $L \sim N(\overline{L}, \sigma^2)$. For a trajectory with $M$ segments where each segment indicates two walking steps, the estimation of moving distance is $\sum_{i=1}^{M} S_i \sim N(2M\overline{L}, 2M\sigma^2)$. Let $\Delta = \sqrt{2M}\sigma$ be the *estimation error*. It increases with the increasing of M, making the estimation more inaccurate. Thus we need to refine the estimation from time to time. Unlike most previous work relying on densely-deployed fixed reference points to help refining location estimation [8], [9], [21], [11], we propose several novel strategies to improve location estimation using encounter opportunities of mobile nodes in sparse DTNs.

Based on the role of the encountering node (a reference point or a common-node), we apply different location refinement methods as follows.

#### 3.6.1 Refinement based on reference point

When a common-node meets a GPS-node (or a landmark), it can obtain the location from the encountering node and use it as a reference point to adjust the estimation.

If a common-node $n_i$ encounters a GPS-node $n_j$, it means the distance between them is less than the communication range $r$ (the value of $r$ depends on the communication technique used, for example, $r$ is about 2 meters for Bluetooth and about 10 meters for WiFi). Assume the estimated location is $P_1(x_1, y_1)$ and the location of reference point is $P'_1(x'_1, y'_1)$. If the Euclidean distance $||P_1 - P'_1||$ is smaller than $r$, the estimation is considered valid and no need to adjust. If $||P_1 - P'_1|| > r$, it means using $P'_1$ as the estimation will be more accurate. In the latter, we need to adjust our location estimation. But we do not simply change its value to $P'_1$: we need to further refine the estimation of each segment in the trajectory.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2014.2310471, IEEE Transactions on Parallel and Distributed Systems



The basic idea to refine the segments is to amortize the errors. For simplicity we only adjust the length of each segment and leave the approximate angle unchanged. An example of trajectory adjustment is shown in Fig. 3. Assume the trajectory after adjustment is:

$$T'(P_0 \rightarrow P_1') = \{< S_1', \theta_1 >, \cdots < S_M', \theta_M >\}. \quad (4)$$

According to Eq. (3), the coordinates of $P_1'(x_1', y_1')$ satisfy: $x_0 + \sum_{i=1}^{M} S_i' \sin\theta_i = x_1'$, and $y_0 + \sum_{i=1}^{M} S_i' \cos\theta_i = y_1'$. However, there are numerous solutions in accordance with the above equations, which infer different walking trajectories. We are interested in finding the shortest path from $P_0$ to $P_1'$, hoping that the step size after adjustment is not far away from its previous estimation, i.e. within a predefined threshold $\varepsilon$.

In summary, the trajectory adjustment problem can be expressed as:

$$minimize \ \sum_{i=0}^{M} S_i', \quad (5)$$

$$s.t.$$

$$\begin{gathered} \sum_{i=1}^{M} S_i' \sin\theta_i = x_1' - x_0, \\ \sum_{i=1}^{M} S_i' \cos\theta_i = y_1' - y_0, \\ S_i' > 0, \ i = 1, \cdots, M, \\ ||S_i' - S_i|| \leq \varepsilon, \ i = 1, \cdots, M. \end{gathered}$$

This problem is an optimization problem and it can be solved by using Linear programming [23]. Solving this problem, we can adjust $S_i$ to $S_i'$ ($i = 1, \cdots, M$). After adjustment, the estimation error becomes $\Delta \leq r$, which is more accurate than the previous estimation.

The novelty of the trajectory adjustment method is that it is decentralized and transitive. Whenever a common-node encounters a GPS-node or a landmark, it can adjust its location estimation locally. The more frequent the encounter occurs, the more accuracy the location estimation achieves. Furthermore, by solving the optimization problem, the proposed adjustment method can not only refine the current location, but also trace back to refine previous segments, which can amortize the estimation error and obtain a more accurate trajectory.

### 3.6.2 Mutual refinement

When a common-node encounters another common-node, although both of them have no accurate location information, it is still possible for them to use each other as reference point to refine location estimation. Assume $n_i$ and $n_j$ are the two common-nodes, we discuss the following situations of encountering.

**a) $n_i$ and $n_j$ encounter in a straight road.**

When two nodes $n_i$ and $n_j$ encounter in a straight road, they will exchange their location estimation, which could be used as reference points to refine their estimation. There are four different situations for the relative locations of the encountering nodes (refer to Appendix A in the supplemental document for more details). In the following, we introduce a mutual location refinement method for the situation illustrated in Fig. 4. Note that the proposed method also applies for the other situations with slight modifications.

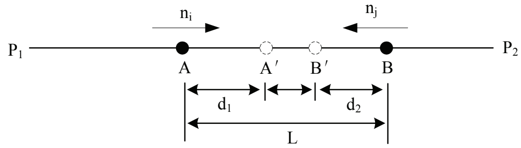



Fig. 4. Refining location estimation in a straight road.

According to Fig. 4, $n_i$ and $n_j$ move in the opposite direction, and $A$ and $B$ are their estimated locations accordingly. Due to the error of estimation, the distance between $A$ and $B$ (denoted by $L$) could be larger than the communication range $r$ (note that if $L \leq r$, no refinement is needed). Since we know the fact that $n_i$ and $n_j$ encounter each other in a straight road, their distance must be less than $r$. Such constraint can be used to refine the location estimation of $n_i$ and $n_j$. To do so, we make new estimation of the locations of $n_i$ and $n_j$, which are indicated by $A'$ and $B'$ in Fig. 4. We denote the distance $AA' = d_1$ and $BB' = d_2$. Since $n_i$ and $n_j$ are within the communication range, the distance of $A'B'$ should satisfy

$$A'B' = L - d_1 - d_2 \leq r. \quad (6)$$

We introduce a method based on *maximum-likelihood* to obtain the new estimation $A'$ and $B'$. Assume node $n_i$ moves from $P_1$ to $A'$, and the moving distance is a random variable $X_1$ described by a Gaussian distribution $X_1 \sim N(\mu_1, \sigma_1^2)$. If $P_1A'$ is the actual distance, the conditional probability that the estimated distance equals to $P_1A$ is given by

$$\begin{aligned} f_1 : \quad & Pr\{X_1 = P_1A | distance = P_1A'\} \\ &= \frac{1}{\sigma_1\sqrt{2\pi}} e^{-\frac{(P_1A - P_1A')^2}{2\sigma_1^2}} \\ &= \frac{1}{\sigma_1\sqrt{2\pi}} e^{-\frac{d_1^2}{2\sigma_1^2}}. \end{aligned} \quad (7)$$

Similarly, for node $n_j$, assume its moving distance is a random variable $X_2 \sim N(\mu_2, \sigma_2^2)$. The conditional probability of estimated distance $P_2B$ is:

$$\begin{aligned} f_2 : \quad & Pr\{X_2 = P_2B | distance = P_2B'\} \\ &= \frac{1}{\sigma_2\sqrt{2\pi}} e^{-\frac{d_2^2}{2\sigma_2^2}}. \end{aligned} \quad (8)$$

To achieve mutual refinement, we take $P_1A$ and $P_2B$ as the observed values, and we want to choose the estimated values $P_1A'$ and $P_2B'$ to maximize the likelihood function. Specifically, the objective is to decide $argmax_{\{d_1,d_2\}} f_1(d_1)f_2(d_2)$ subject to the constraint of Eq. (6).

We introduce a slack variable $a \geq 0$ to transform Eq. (6) into equality based on the fact that $L - d_1 -$

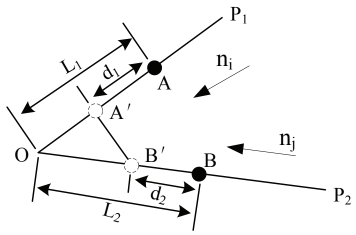



Fig. 5. Refining location estimation in an intersection.

$d_2 \leq r$ if and only if there is an $a \geq 0$ that satisfies $L - d_1 - d_2 + a = r$. With the new constraints, the optimization problem can be expressed as

$$\begin{aligned} maximize \quad & f_1(d_1)f_2(d_2) \\ s.t. \quad & L - d_1 - d_2 + a - r = 0, \\ & a \geq 0. \end{aligned}$$

The optimization problem can be solved by using the method of Lagrange multipliers. The details are found in Appendix C of the supplemental document. The solution is as follows.

$$d_1 = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}(L - r), \tag{9}$$

and

$$d_2 = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}(L - r). \tag{10}$$

Thus with the best probability that the new distance estimation are $P_1A + d_1$ for $n_i$ and $P_2B + d_2$ for $n_j$. As a result, the estimated locations of $n_i$ and $n_j$ are updated to the coordinates of $A'$ and $B'$ accordingly.

In the proposed PulseCounting method, the accumulative estimation error is the sum of the estimation error of walking steps. If $M_1$ and $M_2$ are the number of trajectory segments of $n_i$ and $n_j$ accordingly, since each segment indicates two walking steps, the accumulative estimation error of $n_i$ and $n_j$ are represented by $\sigma_1^2 = 2M_1\sigma_{n_i}^2$ and $\sigma_2^2 = 2M_2\sigma_{n_j}^2$. We can substitute $\sigma_1^2$ and $\sigma_2^2$ to Eq. (9) and Eq. (10) to obtain the refined estimation of $n_i$ and $n_j$ accordingly. Thus the encountering nodes can achieve mutual location refinement by simply exchanging their local estimations.

**b) $n_i$ and $n_j$ encounter in a road intersection.** When two nodes $n_i$ and $n_j$ move in non-parallel directions and they encounter each other, it indicates that they meet in a road intersection. Their relative locations have four different situations (refer to Appendix B in the supplemental document for more details). Knowing two nodes encountering in a road intersection can also refine their location estimations. Without loss of generality, we only discuss the refining method for the situation illustrated in Fig. 5. The rest situations can also be adapted to our method with slight modifications.

In Fig. 5, the movement orientations and the estimated locations of $n_i$ and $n_j$ (denoted by $A$ and $B$ in Fig. 5) are known, so the intersection (denoted by $O$) of their trajectories are uniquely determined. We assume the length $AB > r$ (otherwise the two nodes are close enough and no refinement is needed). The new estimated locations $A'$ and $B'$ are to be decided.

Similar to the analysis in the previous subsection, we assume the estimated moving distances of $n_i$ and $n_j$ are random variables following Gaussian distribution $X_1 \sim N(\mu_1, \sigma_1^2)$ and $X_2 \sim N(\mu_2, \sigma_2^2)$ accordingly. The conditional probabilities of the difference between the observed location and actual location are given by Eq. (7) and Eq. (8).

The new estimated distance of $A'$ and $B'$ should satisfy $A'B' \leq r$. However, such constraint is non-linear, which yields a complicated optimization problem. To reduce the computation complexity, we explore an enhanced problem. We consider a stronger constraint: $OA' + OB' \leq r$. Due to the triangle inequality, $A'B' \leq OA' + OB'$, so $OA' + OB' \leq r$ is a sufficient condition for $A'B' \leq r$. The optimization problem with enhanced constraints can be expressed as

$$\begin{aligned} maximize \quad & f_1(d_1)f_2(d_2) \\ s.t. \quad & L_1 - d_1 + L_2 - d_2 + b - r = 0, \\ & b \geq 0, \end{aligned}$$

where $b$ is a slack variable.

Similarly, this problem can be solved by using the Lagrange multipliers method. We omit the details and give the final results:

$$d_1 = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}(L_1 + L_2 - r), \tag{11}$$

and

$$d_2 = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}(L_1 + L_2 - r), \tag{12}$$

where $\sigma_1^2$ and $\sigma_2^2$ can be represented by the accumulative estimation error of walking steps: $\sigma_1^2 = 2M_1\sigma_{n_i}^2$ and $\sigma_2^2 = 2M_2\sigma_{n_j}^2$.

As a result, we can achieve the maximal probability when the distances are estimated by $P_1A + d_1$ for $n_i$ and $P_2B + d_2$ for $n_j$. Accordingly, the estimated locations of $n_i$ and $n_j$ are updated to the coordinates of $A'$ and $B'$.

The novelty of mutual refinement is that the encountering nodes can take each other as a reference point, and apply the maximum-likelihood method to reduce the estimation errors. In DTNs, a node may encounter multiple nodes during movement. The proposed refining method allows the node to adjust its location estimation on each pair-wise contact. Thus the encountering opportunities in DTNs are not only benefit to data dissemination, but also benefit to cooperative positioning as described in our work.

One can further consider the meeting duration of node pairs to improve the estimation accuracy. If the meeting duration exceeds a predefined threshold, the mobile nodes should update their location estimation periodically using the proposed refinement strategies.

By doing so, if the encountering node is a reference point (i.e., a landmark or a GPS node), the mobile node can calibrate its location estimation periodically. If the encountering node is a common-node, they can do multiple rounds of mutual refinement, which will also increase the accuracy of location estimation.

## 4 PROBTRACKING: TRACING MOBILE USERS IN DTNS

In this section, we introduce the ProbTracking method for the infostation to track user movement in DTNs. Most previous works assume continuous communication between users and infostation, thus tracking a user is easily achieved by acquiring the user location constantly. However, due to the opportunistic communication nature of DTNs, tracking the users' trajectory without continuous connection is non-trivial.

In DTNs, the infostation can only communicate with the nodes passing by the landmarks. Each node keeps logging its trajectory while moving, as well as the partial trajectory information obtained from the encountering nodes. For a node $n_i$, its log is in the form $\{T_{n_i}(\cdot)\} \bigcup \{T_{n_j}(\cdot) | \forall n_j \in \phi_i\}$, where $T_{n_i}(\cdot)$ is the trajectory of $n_i$; $T_{n_j}(\cdot)$ is the partial trajectory of an encountering node $n_j$; and $\phi_i$ is the set of encountering nodes of $n_i$. When a node encounters a landmark, it uploads its log to the inforstation. Thus the infostation has the information of a set of trajectories: $\{T_{n_i}(\cdot) | i = 1, \cdots, N\}$. Such information is *incomplete*: not all trajectories of all nodes are recorded; and it is *non-realtime*: it is updated intermittently. The tracking problem is: *given the set of partial trajectory, how to determine the movement of nodes*.

To address this issue we look to the patterns of user movement. Intuitively user mobility is not random, and it will follows some patterns. For example, there are daily activities like going to office at 8:00, going for lunch at 13:00, and being back home at 19:00. By exploring the mobility patterns we are possible to recreate the user's movement trajectory with some known knowledge.

A previous research [24] shows that human mobility follows reproducible patterns with their trajectories characterized by a significant probability to return to a few highly frequented locations. In the DTN system, we use landmarks as the fixed characteristic locations, and calculate the frequency and probability for the users moving from one landmark to another. The mobility pattern of a user can be naturally described as a *Markov chain*: the landmarks correspond to the *states* and the probabilities moving to different landmarks form the *transition matrix* of the Markov chain.

In the system implementation, we calculate the visiting frequency and the transition probabilities as follows. For each trajectory uploaded by a user, we use $T_k^{in}$ and $T_k^{out}$ to indicate the time stamps that the user enters and leaves the range of landmark $k$. Denoted by $v_{kj}$ the number of visits from landmark $k$ to landmark $j$ in an observing duration $T^{obs}$. For each $T_k^{out}$, we find the next time stamp $T_j^{in} = min\{T_i^{in} | T_i^{in} > T_k^{out}\}$ that the user enter another landmark. If $T_j^{in} - T_k^{out}$ is less than a predefined threshold $\Delta T$, the user is considered heading landmark $j$, and $v_{kj}$ is increased by 1. In our system, $\Delta T$ is set to 1 hour since most locations in the campus are with the walking distance of an hour. If $T_j^{in} - T_k^{out} > \Delta T$, the user is considered to enter an unknown place (which corresponds to a special state in the Markov chain). After having $v_{kj}$, the probability that the user moving from landmark $k$ to $j$ can be estimated by $Pr\{j|k\} = \frac{v_{kj}}{\sum_i v_{ki}}$. The transition matrix is updated weekly in the system, thus we set the observing period $T^{obs}$ to be one week.

At the beginning, the infostation needs to collect enough information of user movement trajectories to construct the Markov chain. We assume there is a "warm-up" stage in the tracking system. During warm-up, the system only collects historical data and it cannot provide any tracking information. The warm-up stage can last for one day or one week depending on the amount of information collected.

Once the Markov chain is formed, the infostation can recreate and predict the missing trajectories of user movement. For example, if the infostation knows that a user appears in location $P_i$ at time $t$, it can check the transition matrix of the Markov chain to obtain the probability that the user moving from $P_i$ to other locations. With such information, the inforstation can estimate the location of the user at time $t + \Delta t$ by exploring the most probable historical trajectory from the trace. An additional example of using Markov chain to decide user movement trajectory is shown in Appendix D in the supplemental document.

The novelty of the proposed ProbTracking system is that it can create the most probable user trajectory from incomplete observations. According to the historical movement data, it describes the user's mobility as a finite state Markov chain, and generates a rough trajectory for the mobile user based on partial location records (the encountering locations observed by other mobile users). However, there are several limitations of the proposed tracking system. First, it is non-realtime. Although it can achieve "post-tracking" by fulfilling a trajectory using partial information, it is hard to obtain the real-time location of a mobile user. Second, it only has limited ability of prediction. One may apply Markov chain to predict the user's current movement, and one can obtain the information from the system such as "with probability $p$ the user is moving on the way from $P_i$ to $P_j$". However, such information is valuable only when the probability $p$ is high (the accuracy depends on the routine of user movement, and it varies from person to person). To achieve accurate real-time tracking of mobile users in DTN systems still remains an open question.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2014.2310471, IEEE Transactions on Parallel and Distributed Systems



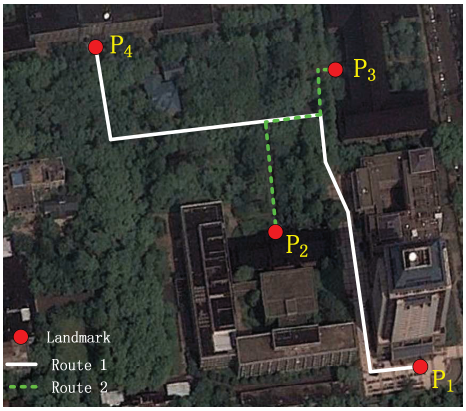



Fig. 6. Experimental scenario in the NJU campus.

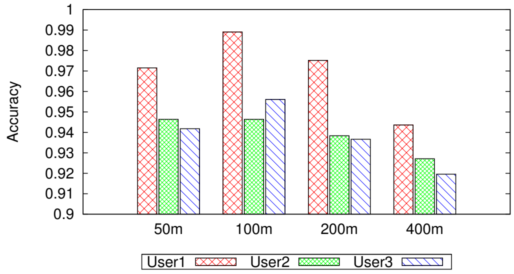



Fig. 7. Accuracy of distance estimation.

TABLE 1
The accuracy of directions with different users.

| *Methods* | User a | User b | User c |
|---|---|---|---|
| *Compass Reading* | 80.12% | 59.34% | 71.43% |
| *Direction Mapping* | 90.03% | 75.12% | 87.89% |

## 5 Implementation and Evaluation

In this section, we discuss the implementation issues and evaluate the system performance.

### 5.1 Implementation

We implemented the PulseCounting localization method in the HTC Wildfire phone. The HTC Wildfire Android smartphone has built-in GPS, WiFi and Bluetooth communication modules, as well as sensors such as accelerometer and compass. The proposed positioning system was implemented as an Android App and tested by 12 volunteers.

To track the movement of users, we implemented the ProbTracking method in a PC server, where MySQL and PHP are adopted to store user historical data and calculate the movement trajectories. The tracking service was deployed in an Apache server, and it can be accessed from a web browser with JavaScript.

We deployed a testbed in the Gulou campus of Nanjing University (NJU) to evaluate the system performance. The experimental environment is illustrated in Fig. 6. Four APs were deployed in the campus which work as landmarks and provide connection to Internet. The chosen landmarks corresponds to four typical locations in the campus: $P_1$ (the Department of Computer Science), $P_2$ (the library), $P_3$ (the Southeast building), and $P_4$ (the academic building).

We design several scenarios to test the system. As shown in Fig. 6, we specify two routes $P_1 \leftrightarrow P_4$ (marked by the full white line) and $P_2 \leftrightarrow P_3$ (marked by the dashed orange line), which represent a longer and a shorter walking paths accordingly. Both routes have straight sections and turns, and they intersect each other. The volunteers were asked to walk along the two routes back and forth, with the phones placed in their front pockets. The WiFi modules of the smart phones were turned on. Some random users were appointed to play the role of GPS-nodes. The mobile users have the opportunities to encounter on the straight road or intersections, and their location estimation are refined dynamically using the methods proposed in section 3.6. All GPS locations of the users were logged, which are used as ground truth for performance comparison.

### 5.2 Accuracy of distance estimation

We investigate the accuracy of distance estimation using PulseCounting. We asked the volunteers to walk through a 100-meter straight road for several rounds. Using the measurement data from the cellphone accelerometer, the number of steps is obtained, thus the average step length of each user can be calculated.

After measuring their average step sizes, we estimate their moving distance using the PulseCounting method. We chose four different road segments in the NJU campus with length 50m, 100m, 200m, and 400m accordingly. We asked the users to walk through the road sections and compared their distance estimation to the actual distance. The experiment results of three different users are shown in Fig. 7. According to the experiments, the distance estimation in the system is rather accurate: most of them have the accuracy higher than 90%, which indicates that the average deviation between the estimated location and the real location is lower than 10%. The accuracy is also various from user to user. Some user has high accuracy approaching to 98%, while other users have lower accuracy due to estimation errors.

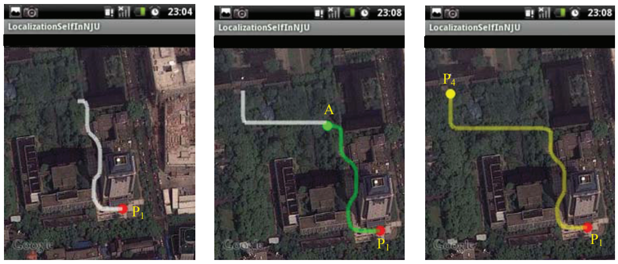
### 5.3 Accuracy of direction mapping

We conducted an experiment to evaluate the accuracy of the direction mapping method. In this experiment, we asked the users walk an octagon shape in the playground. The eight edges of the octagon correspond to the eight discrete directions described in Section 3.3. Starting from the north direction, the users walk along the eight directions sequentially.

The direction accuracy of three different users are shown in Table 1. According to the table, the compass reading data is fluctuating and achieves low accuracy, which is 80.12% for user $a$, 59.34% for user $b$, and 71.43% for user $c$. With the direction mapping method,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2014.2310471, IEEE Transactions on Parallel and Distributed Systems






(a) Starting. (b) Midway. (c) Arriving.

Fig. 8. Screen shots of user movement trajectory.

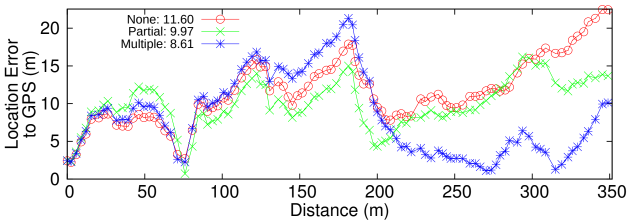



Fig. 9. Instantaneous deviation compared to GPS.

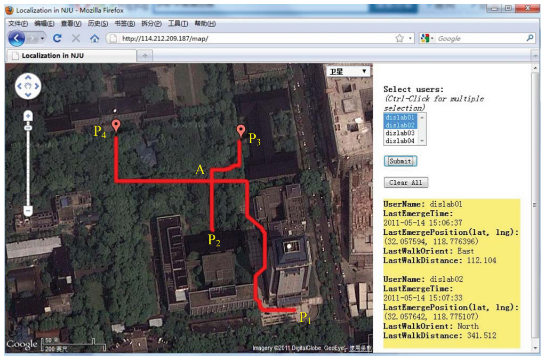



Fig. 10. Screen shots of the tracking system.

the accuracy reaches to 90.03% for user $a$, 75.12% for user $b$, and 87.89% for user $c$, which implies a significant improvement in performance.

### 5.4 The positioning system

As mentioned above, the proposed cooperative positioning system was implemented as an Android App and installed in the HTC phones. The App can estimate user's location and demonstrate the moving trajectory in the screen. Fig. 8 illustrated the screen shots of a user walking from $P_1$ to $P_4$. At the beginning, the user leaved $P_1$ and headed north (Fig. 8(a)). In the midway, the user encountered a GPS-node at position $A$ (Fig. 8(b)). After information exchange, the positioning system used the received GPS coordinates to refine his trajectory (from $P_1$ to $A$). The refined trajectory is demonstrated in colored lines in Fig. 8. Thereafter, the user moved towards west, then turned to the north, and finally arrived the destination $P_4$. Since $P_4$ was a landmark, the latest trajectory (from $A$ to $P_4$) was refined using the obtained location information, and the updated trajectory was shown in the user's screen as Fig. 8(c).

To show the accuracy of the positioning system and the effectiveness of the location refinement methods, we use the GPS logs as ground truth and calculate the deviation of PulseCounting to GPS during movement. Fig. 9 shows the *instantaneous deviation* of a random user moving from $P_1$ to $P_4$. The "None" data (red circle dots) mean that no refinement is done to the estimation. The "Partial" data (green cross dots) indicates that the location estimation is refined only using the information from other mobile nodes. The "Multiple" data (blue star dots) indicates that the node considers all possible information from landmarks and from other mobile nodes to refine its location estimation. As shown in the figure, the instantaneous deviation is fluctuating, which means that the relative estimation error compared to GPS varies from time to time. For shorter distance (e.g., shorter than 200m), the three curves are close to each other. For longer distance (e.g., longer than 200m), the deviation of "None" data is the highest. The location estimation refined by only mobile node is slightly lower than non-refinement. The location refined by landmarks and mobile nodes achieves the lowest deviation, which is significantly lower than the others.

The *average deviations* of the three cases are shown beside their corresponding symbols in Fig. 9. It can be seen that the average deviation of the three methods are 11.60m, 9.97m, and 8.61m accordingly.

Note that the positioning accuracy depends on the communication range of the mobile devices. Theoretically the proposed approach is applicable to different wireless communication techniques including WiFi, BlueTooth, ZigBee, etc, and different communication technique will yield different accuracy accordingly. In our system, we choose WiFi for implementation since it is widely available and equipped in most smart phones. The typical communication range of WiFi is about 10 meters, and the average accuracy of 9 meters (compared to GPS) is achieved using the proposed positioning and refinement approaches.

### 5.5 The tracking system

The tracking system was implemented in a PC server, which was connected to the Internet and could be accessed via web browsers. The tracking server can emulate the trajectories of mobile nodes and display their traces on the map.

Fig. 10 illustrates the trajectories of two mobile nodes in our experiment. In this scenario, node 1 moved from $P_1$ to $P_4$ and node 2 moved from $P_2$ to $P_3$. They encountered at an intersection $A$. Node 2 reached to its destination first (since it walked along a shorter path). The server collected data from node 2 via the AP in $P_3$. Based on the knowledge of history statistics and the trajectories uploaded by node 2, the server determined that node 1 was moving to $P_4$ with high probability and infer its movement with a historical trajectory. Thus the browser can display the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2014.2310471, IEEE Transactions on Parallel and Distributed Systems



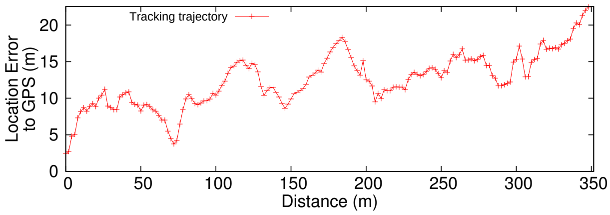



Fig. 11. Deviation of the tracking system.

walking routes of both users without communicating with them in realtime.

Fig. 11 compares the deviation of the tracking trajectory to the GPS records. According to the figure, in the walking distance of 350 meters, the deviation varies from 5 meters to 25 meters. The average deviation is 12.49 meters. Note that such accuracy achieves only when the system makes a correct prediction of the user's route. If the system makes a wrong prediction (based on Markov chain, this will happen with some probability), the deviation will be far away from the real trajectory. Improving the prediction accuracy and reducing the tracking errors will be our future work.

### 5.6 Energy consumption

We evaluate the energy efficiency of the proposed positioning system, and show that its energy consumption is slightly higher than online radio listening, but much lower than online video watching. The details are found in Appendix E in the supplemental document.

### 5.7 Discussion and comparisons

We further discuss the pros and cons of the proposed positioning and tracking system, and provide comparisons with other related works such as AAMPL [21], CompAcc [11], and Escort [12]. The details are found in Appendix F in the supplemental document.

## 6 Conclusion

Localization in DTNs faces two major difficulties: the mobile node can only use sparse reference points to estimate its location, and the tracking server need to determine and predict movement trajectories with partial location information. To overcome these difficulties, we propose *PulseCounting* and *ProbTracking* for positioning and tracking in DTNs. We implement the system in Android phones and evaluate its performance in a testbed in the NJU campus. Extensive experiments show that the proposed system achieves an average deviation less than 9m compared to GPS.

## Acknowledgement

This work was partially supported by the National Natural Science Foundation of China (Grant Nos. 61373128, 91218302, 61321491), the EU FP7 IRSES MobileCloud Project (Grant No. 612212), and the funding from Alexander von Humboldt Foundation.

## References

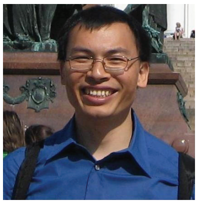
[1] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proc. of SIGCOMM '03*. ACM, 2003, pp. 27–34.
[2] A. Lindgren, A. Doria, E. Davies, and S. Grasic, "Probabilistic routing protocol for intermittently connected networks," RFC 6603, Internet Research Task Force (IRTF), Aug. 2012.
[3] S. Gaonkar, J. Li, R. R. Choudhury, L. Cox, and A. Schmidt, "Micro-blog: sharing and querying content through mobile phones and social participation," in *Proc. of MobiSys '08*. ACM, 2008, pp. 174–186.
[4] E. D. Kaplan, *Understanding GPS Principles and Applications*. Artech House Publishers, Feb 1996.
[5] J. Soubielle, I. Fijalkow, P. Duvaut, and A. Bibaut, "GPS positioning in a multipath environment," *IEEE Transactions on Signal Processing*, vol. 50, no. 1, pp. 141–150, Jan 2002.
[6] J. Yang, A. Varshavsky, H. Liu, Y. Chen, and M. Gruteser, "Accuracy characterization of cell tower localization," in *Proc. of Ubicomp '10*. ACM, 2010, pp. 223–226.
[7] M. Y. Chen, T. Sohn, D. Chmelev, D. Haehnel, J. Hightower, J. Hughes, A. LaMarca, F. Potter, I. Smith, and A. Varshavsky, "Practical metropolitan-scale positioning for gsm phones," in *Proc. of UbiComp'06*. Springer-Verlag, 2006, pp. 225–242.
[8] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building rf-based user location and tracking system," in *Proc. of Infocom'00*. IEEE, 2000, pp. 775–784.
[9] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, J. Tabert, P. Powledge, G. Borriello, and B. Schilit, "Place lab: device positioning using radio beacons in the wild," in *Proc. of PERVASIVE'05*. Springer-Verlag, 2005, pp. 116–133.
[10] M. Azizyan, I. Constandache, and R. Roy Choudhury, "Surroundsense: mobile phone localization via ambience fingerprinting," in *Proc. of MobiCom '09*. ACM, 2009, pp. 261–272.
[11] I. Constandache, R. R. Choudhury, and I. Rhee, "Towards mobile phone localization without war-driving," in *Proc. of INFOCOM'10*. IEEE Press, 2010, pp. 2321–2329.
[12] I. Constandache, X. Bao, M. Azizyan, and R. R. Choudhury, "Did you see bob?: human localization using mobile phones," in *Proc. of MobiCom '10*. ACM, 2010, pp. 149–160.
[13] N. Banerjee, M. D. Corner, and B. N. Levine, "Design and field experimentation of an energy-efficient architecture for dtn throwboxes," *IEEE/ACM Transactions on Networking*, vol. 18, no. 2, pp. 554–567, 2010.
[14] Y. Li, Y. Jiang, D. Jin, L. Su, L. Zeng, and D. Wu, "Energy-efficient optimal opportunistic forwarding for delay-tolerant networks," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 9, pp. 4500–4512, Nov 2010.
[15] Y. Li, P. Hui, D. Jin, L. Su, and L. Zeng, "Evaluating the impact of social selfishness on the epidemic routing in delay tolerant networks," *IEEE Communications Letters*, vol. 14, no. 11, pp. 1026–1028, November 2010.
[16] P. Hui, K. Xu, V. Li, J. Crowcroft, V. Latora, and P. Lio, "Selfishness, altruism and message spreading in mobile social networks," in *Proc. of INFOCOM Workshops '09*, April 2009.
[17] S. Gezici, "A survey on wireless position estimation," *Wireless Personal Communications*, vol. 44, no. 3, pp. 263–282, Feb. 2008.
[18] S. Sen, B. Radunovic, R. R. Choudhury, and T. Minka, "You are facing the mona lisa: spot localization using phy layer information," in *Proc. of MobiSys '12*. ACM, 2012, pp. 183–196.
[19] H. Liu, Y. Gan, J. Yang, S. Sidhom, Y. Wang, Y. Chen, and F. Ye, "Push the limit of wifi based localization for smartphones," in *Mobicom '12*. ACM, 2012, pp. 305–316.
[20] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: wireless indoor localization with little human intervention," in *Proc. of Mobicom '12*. ACM, 2012, pp. 269–280.
[21] A. Ofstad, E. Nicholas, R. Szcodronski, and R. R. Choudhury, "AAMPL: accelerometer augmented mobile phone localization," in *Proc. of MELT '08*. ACM, 2008, pp. 13–18.
[22] Google, "The google map," https://maps.google.com/, 2012.
[23] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, March 2004.
[24] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, pp. 779–782, June 2008.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2014.2310471, IEEE Transactions on Parallel and Distributed Systems



**Wenzhong Li** received his B.S. and Ph.D degree from Nanjing University, China, both in computer science. He was a Humboldt Scholar at the University of Göttingen, Germany. He is now an associate professor in the Department of Computer Science, Nanjing University. Dr. Li's research interests include wireless networks, pervasive computing, and social networks. He has published over 40 research papers at international conferences and journals, which include ICDCS, IWQoS, ICPP, ICC, WCNC, IEEE Transactions on Wireless Communications, IEEE Transactions on Vehicular Technology, etc. Dr. Li was also the winner of the Best Paper Award of ICC 2009. He is a member of IEEE and ACM.

**Yuefei Hu** received his B.S. degree in Computer Science from Jilin University, Changchun, China, and received his M.s. degree in Computer Science from Nanjing University, Nanjing, China. He is now working as a staff in the Zhengzhou Commodity Exchange, one of the four futures exchanges in China. His research interests include data routing in wireless sensor networks, wireless media streaming system and localization in delay tolerant networks.



**Xiaoming Fu** received his bachelor and master degrees from Northeastern University, China, and his Ph.D. degree from Tsinghua University, Beijing, China. He was a research staff at the Technical University Berlin until joining the University of Göttingen, Germany in 2002, where he has been a professor in computer science and heading the Computer Networks Group since 2007. His research interests include network architectures, protocols, and applications. He is currently an editorial board member of IEEE Communications Magazine, IEEE Transactions on Network and Service Management, Elsevier Computer Networks, and Computer Communications, and has published over 100 papers in journals and international conference proceedings. He is the coordinator of EU FP7 GreenICN and MobileCloud projects, and the recipient of the IEEE LANMAN 2013 Best Paper Award and the 2005 University of Göttingen Foundation Award for Exceptional Publications by Young Scholars.

**Sanglu Lu** received her B.S., M.S., and Ph.D. degrees from Nanjing University in 1992, 1995, and 1997, respectively, all in computer science. She is currently a professor in the Department of Computer Science and Technology and the deputy director of State Key Laboratory for Novel Software Technology. Her research interests include distributed computing, pervasive computing, and wireless networks. She is a member of IEEE and ACM.

**Daoxu Chen** was a visiting professor in Purdue University and City University of Hong Kong. He is now a professor in the Department of Computer Science, Nanjing University. His research interests include distributed computing, parallel processing, and computer networks. He has published over 100 research papers in international conference proceedings and journals. He is a member of IEEE, ACM and a senior member of the China Computer Federation.