ioNavi: An Indoor-Outdoor Navigation Service via Mobile Crowdsensing

XIAOQIANG TENG, National University of Defense Technology
DEKE GUO, National University of Defense Technology
YULAN GUO, National University of Defense Technology
XIAOLEI ZHOU, National University of Defense Technology
ZELIU DING, Naval University of Engineering
ZHONG LIU, National University of Defense Technology

The proliferation of mobile computing has prompted navigation to be one of the most attractive and promising applications. Conventional designs of navigation systems mainly focus on either indoor or outdoor navigation. However, people have a strong need for navigation from a large open indoor environment to an outdoor destination in real life. This paper presents ioNavi, a joint navigation solution, which can enable passengers to easily deploy indoor-outdoor navigation service for subway transportation systems in a crowdsourcing way. Any self-motivated passenger records and shares individual walking traces from a location inside a subway station to an uncertain outdoor destination within a given range, such as one kilometer. ioNavi further extracts navigation traces from shared individual traces, each of which is not necessary to be accurate. A subsequent following user achieves indoor-outdoor navigation services by tracking a recommended navigation trace. Extensive experiments are conducted on a subway transportation system. The experimental results indicate that ioNavi exhibits outstanding navigation performance from an uncertain location inside a subway station to an outdoor destination. Although ioNavi is to enable indoor-outdoor navigation for subway transportation systems, the basic idea can naturally be extended to joint navigation from other open indoor environment to outdoor environment.

Additional Key Words and Phrases: Indoor-outdoor navigation, subway station navigation, mobile crowdsensing, trace clustering, indoor localization

1. INTRODUCTION

Subway transportation systems have been well developed in more than one hundred cities around the world. Today, the subway transportation system becomes one of the dominant solutions of public transportation systems in many cities. When traveling by a subway train, a passenger has a strong need to obtain the most convenient and shortest walking path from a location inside a subway station to a nearby outdoor destination (e.g., a shopping mall). This essential requirement, however, is not easy to be satisfied due to the following reasons. First, such subway stations suffer complicat-

This work is supported in part by the National Natural Science Foundation for Outstanding Youth under grant No.61422214, the Program for New Century Excellent Talents in University, and Distinguished Young Scholars of National University of Defense Technology under grant No.JQ14-05-02, the Preliminary Research Funding of National University of Defense Technology under grant NO.ZDYYJCYJ20140601, and the National Natural Science Foundation of China under No. 61602499.
Author's addresses: X. Teng, D. Guo, X. Zhou, and Z. Liu are with the College of Information System and Management, National University of Defense Technology, Changsha, Hunan, 410073, P. R. China. Z. Y. Guo is with the College of Electronic Science and Engineering, National University of Defense Technology, Changsha, Hunan, 410073, P. R. China. Z. Ding is with the School of Electronic Engineering, Naval University of Engineering, Wuhan, Hubei, 430033, P. R. China.
E-mail: {tengxiaoqiang13, dekeguo, yulan.guo, xl.zhou, zeliuding, liuzhong}@nudt.edu.cn.

DOI: http://dx.doi.org/10.1145/0000000.0000000

ed indoor structures, especially for those interchange stations. Second, the passengers are unaware of the situation around each exit entrance of a subway station, especially when some exit entrances are on the road to other indoor buildings, such as underground shopping malls. Lack of necessary indoor-outdoor navigation service not only introduces a lot of trouble for passengers but also considerably decreases the efficiency of subway transportation systems.

Conventional methods are proposed to make each passenger learn from the posted maps, which provides coarse-grained outdoor situation around the subway station. The map information, however, is insufficient to meet the navigation demand of most passengers. Moreover, it is non-trivial for passengers to find an appropriate path by looking at the maps, especially for users with no sense of direction. Although mobile phone applications, such as Subway Navigation[1] recently developed, they just integrate existing maps of a subway station into mobile applications. Thus, they also cannot realize indoor-outdoor automatical navigation. Moreover, conventional design of navigation systems mainly focus on either indoor or outdoor scenario [Zheng et al. 2014; Shu et al. 2015; Dong et al. 2015], the joint navigation problem from an indoor location to an outdoor destination still remains unsolved.

This paper presents a joint navigation system, named ioNavi, which can be easily deployed in a crowdsourcing way without the support of comprehensive localization systems. Our ioNavi system is comprised of three major components. **First**, a self-motivated contributor makes its mobile device automatically record its walking traces and rich set of sensing data (from accelerometer, gyroscope, magnetometer, WiFi module, and 3G/4G module) along with these traces from a location inside a subway station to an outdoor destination. Those traces and sensing data then are then uploaded to a server for upcoming usages. **Second**, the server continuously collects and processes those individual traces to calculate available navigation traces. **Third**, given the starting point and the destination of a user, the optimal navigation trace is calculated and used to intelligently guide the user. After the end of navigation, the user also contributes his actual walking trace and associated sensing data to the server to improve the performance experience of our ioNavi system. If ioNavi system does not have any related navigation trace, the user can share the walking traces and associated sensing data to ioNavi system to expand the navigation areas. In this way, up-to-date navigation information is provided about the indoor and outdoor scenarios for each subway station.

Despite the above benefits, ioNavi faces several major challenges. **First**, users should start with identifying whether they are located inside a subway station and which subway station they are in before executing ioNavi system. In the case of a wrong identification of the subway station, each shared indoor-outdoor trace is not only unusable but also consumes unnecessary energy of mobile devices due to trace recording. To make things worse, such wrong traces will directly misguide the following users and considerably damage the experience of ioNavi system. To tackle this problem, motivated by the fact that most subway stations have been covered by WiFi with distinguished Access Points (APs) and the signal of celltowers, a dedicated method is proposed using such identifiable metrics to identify the subway station. In particular, the sequences of the Basic Service Set Identifiers (BSSIDs) of APs and the Identifiers (IDs) of celltowers are used. Thus, each users only triggers the ioNavi system when necessary.

Second, given a specific subway station, it remains a great challenge to locate a user inside the subway station and plan a high-quality navigation trace from incomplete, noisy, opportunistic, and unorganized traces shared by users. To address this problem,

[1]http://play.google.com/subway-navigation/

a coarse-grained fingerprint enabled method is first proposed to estimate the starting point of a user. The trace extraction method is then proposed by combining the Point of Interest (POI) detection, the dead reckoning, and the trace clustering methods. To improve the accuracy of navigation trace, trace clustering is considered as an optimization problem and a heuristic solution is proposed. In addition, a database table of walking traces is constructd to plan the optimal navigation trace.

Third, given the starting point and the destination of a user, our ioNavi system should be able to intelligently plan routes and to guide the user along the optimal trace in terms of walking distance. It is challenging to find the optimal trace because of three complicated cases: (1) there are multiple traces from the starting point to the destination. (2) None of shared traces starts at the location of a user but at least one shared trace ends at its destination. (3) None of shared traces meets the navigation requirement of a user, but the starting location of the user and its destination location are involved by some shared traces. To tackle this problem, our ioNavi system finds the optimal trace in three cases with pre-defined rules (Sec. 3.3). Besides, a friendly navigation system is expected to intelligently detect the deviation event and notify the user.

A prototype of ioNavi system is implemented and comprehensive experiments have been conducted in a subway transportation system. Experimental results demonstrate that ioNavi system exhibits outstanding navigation performance. The contributions of this paper are listed as follows:

— An architecture of the indoor-outdoor joint navigation system is proposed using mobile devices, which requires no additional hardwares, indoor and outdoor localization systems, and extra user intervention.
— A novel method is proposed to formulate the similarity between two walking traces as an optimization problem, which is NP-complete. A heuristic algorithm is then proposed based on the Hungarian algorithm for realistic solution.
— A novel trace extraction method is proposed to cluster indoor-outdoor traces by measuring the similarity between two walking traces.
— A prototype of ioNavi is implemented. The feasibility and performance is tested in a subway transportation system. Extensive experimental results demonstrate that ioNavi system makes a great progress towards widely deployed indoor-outdoor joint navigation services.

The insight of our ioNavi system is a joint architecture of indoor-outdoor navigation and a crowdsourcing solution by sharing the traces of different users. Although ioNavi is currently implemented in subway transportation systems, the design methodologies and techniques are applicable in other indoor-outdoor scenarios, such as shopping malls.

The rest of this paper is organized as follows. Section 2 presents an overview. Section 3 describes the details of ioNavi. Section 4 reports the evaluation results, followed by technical discussions in Section 5. The related work is summarized in Section 6 and this paper is concluded in Section 7.

2. OVERVIEW

2.1. Motivation

Given that indoor and outdoor navigation systems have been proposed in recent years, why is another navigation system like ioNavi necessary?

First, as outdoor navigation has become ubiquitous through GPS-enabled mobile devices, a user can be easily guided to the destination. However, traditionally used outdoor navigation methods are not available in indoor environments, because GPS

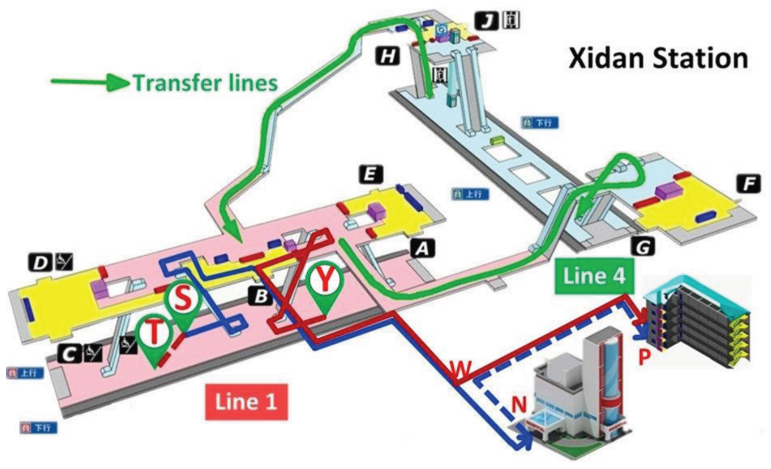


Fig. 1. The floorplan of Xidan subway station in Beijing, China. There are two subway lines and nine exits. A self-motivated passenger walks along the blue line, and then shares its individual trace with other users, starting at the indoor point S and ending the bank N.

signals are unavailable in indoor environments. Therefore, it is an essential challenge to provide accurate joint indoor-outdoor navigation services.

Second, each passenger learns from the posted map, which provides coarse-grained outdoor situation around the subway station. However, The maps are insufficient to meet the navigation demand for most passengers. That is because many outdoor destinations around a subway station are not included by the posted map. Moreover, it is non-trivial for passengers to find an appropriate navigation trace by looking at the maps, especially for users with no sense of direction. In contrast, ioNavi includes more destinations thorough shared traces from those self-motivated users. It can automatically, accurately, and quickly guides the user to the destination.

Third, many existing navigation systems actually focus on localization. Typical localization techniques include GPS based, radio based fingerprinting [Paramvir and Venkata 2000; Yang et al. 2012a; Liu et al. 2012], dead reckoning based [Wang et al. 2012b; Rai et al. 2012; Shen et al. 2013; Li et al. 2012], visible light communication (VLC) based [Kuo et al. 2014; Li et al. 2014a; Xie et al. 2015; Hu et al. 2015], and computer vision [Alessandro et al. 2009; Gao et al. 2014a; Zheng et al. 2014; Wang et al. 2015] methods. These techniques, however, depend on full blown localization systems. In contrast, ioNavi uses the walking traces only. It does not depends on full blown localization systems.

Finally, conventional designs of navigation systems mainly focus on either indoor or outdoor environment to enable accurate navigation. Our ioNavi system is very different from those systems. It is a joint navigation system satisfying the navigation request from a large open indoor environment to an outdoor destination in real life.

2.2. A Usage Example

As shown in Fig. 1, when a trace-contributor gets off the subway train at a point S, ioNavi on a mobile device starts to collect the walking trace and associated sensing data, which is finally transmited to the server. Along the blue line in Fig. 1, the trace-contributor takes an elevator, passes through a subway turnstile, walks out the subway station from the exit B, and reaches the destination N. With the increasing number

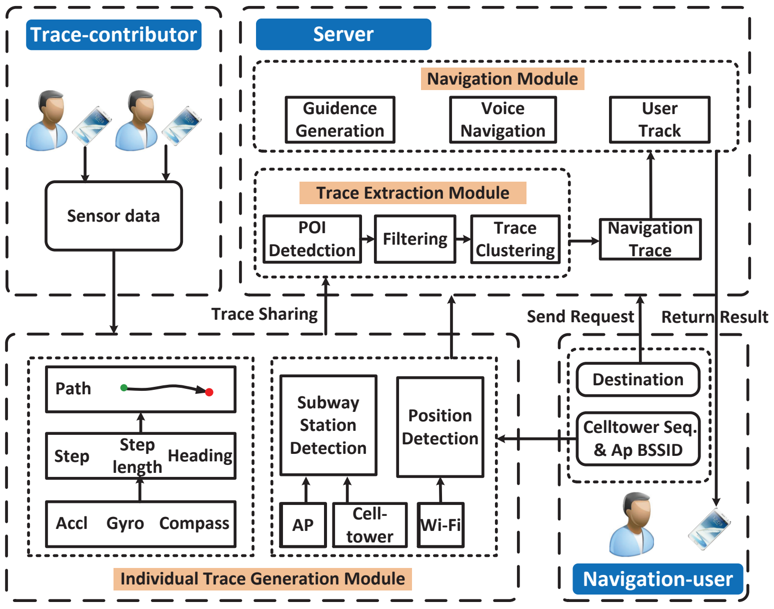


Fig. 2. System architecture.

of participators, the server collects large amounts of traces starting at any locations in the subway station and end at the same destination bank N.

A navigation-user, e.g., an upcoming passenger, gets off the train at this subway station, and wants to arrive at a destination (e.g., the bank N). Our ioNavi system first locates the user in the subway station (e.g., point S), and then recommends one navigation trace to guide the user. The walking process of the user is tracked by our ioNavi system while walking. If the user is off the correct trace, the deviation event will be detected and the user will be notified. If the user does not shut down ioNavi system after arriving at bank N, another trace ending at a shopping mall P is collected (see Fig. 1). Thus, another passenger at point S can be efficiently guided to shopping mall P by the shared new trace (i.e., $S \rightarrow B \rightarrow N \rightarrow P$). In practice, this trace may not be the optimal trace. If another user shares a new trace $S \rightarrow B \rightarrow W \rightarrow P$, it is clear that this trace is better than the prior one. If there exist no shared trace, starting from the current location T of a user to the destination (e.g., a point N), the user will be firstly guided to the closest point (e.g., point S), which is the starting point of an existing trace towards the same destination. In this way, the user can be guided to the point N.

2.3. System Architecture

Figure 2 illustrates the architecture of the proposed ioNavi system. It consists of three components: the individual trace generation module, the trace extraction module, and the navigation module.

Individual Trace Generation Module. Once a user gets off the train, the individual trace generation module starts to collect the IDs of celltowers and a sequence of nearby BSSIDs of APs. The collected data is used to identify the subway station where the user is located. Once the subway station is confirmed, it starts sampling the RSSs of WiFi to locate the user inside the subway station. Furthermore, the system uses the dead reckoning method to calculate the walking traces of a user. Specifically, the walking distance and the step length are calculated using the accelerometer readings and the heading obtained by the gyroscope and magnetometer readings. Finally, the recorded traces will be delivered to the server for further processing. A friendly navigation system performs the individual trace generation process without manual input from the users.

Trace Extraction Module. It extracts high-quality traces from large amounts of shared low-quality walking traces to enable indoor-outdoor navigation service. First, a robust POI detection method is proposed to partition each trace into segments. Second, a series of navigation segments are generated using the peak density based clustering algorithm. The resulting navigation segments are sent to the navigation module for further processing.

Navigation Module. A user first reports its navigation requirement to the server and then is recommended with a navigation trace. In this way, the user will be guided toward the destination. The only task of the user is to input his destination into the ioNavi system.

2.4. Assumptions and Limitations

As illustrated in the above example and the system architecture, ioNavi has a very different navigation model from most existing navigation systems. Instead of enabling independent indoor or outdoor navigation, ioNavi is built as a joint indoor-outdoor navigation system for subway transportation systems, which inevitably introduces several limitations. First, the user cannot be navigated in subway station-s which are not covered by either celltowers or APs, because the subway station where the user is located in and the starting point of the user cannot be estimat-ed without celltowers and APs. This may in turn introduces a robustness problem (as discussed in Sec. 5). Second, it is clear that the number of participated users dominates the navigation performance. To address this issue, similar to Jigsaw [Gao et al. 2014b] and CrowdMap [Chen et al. 2015], an incentive mechanism [Yang et al. 2012b; Luo et al. 2015; Zhang et al. 2015] can be designed to encourage users to share their walking traces. This is described in more detail in Sec. 5. Besides, we will discuss future potential improvements in Sec. 5 based on our evaluation results.

3. SYSTEM DESIGN

In this section, we first describe the individual indoor-outdoor trace modeling and the trace extraction methods. We then introduce the methods for trace planning, walking progress tracking, and deviation event notification.

3.1. Individual Indoor-outdoor Trace Modeling

The individual trace generation module is loaded into a mobile device of a user. The goal of that module is to generate the walking trace of a user. Specifically, it depends on the subway station identification, the starting point detection, and the dead reckoning methods.

3.1.1. Subway Station Identification. As aforementioned, ioNavi should intelligently detect whether a user locates in a subway station and identify the subway station before executing ioNavi system. Incorrect identification will directly misguide the following users and considerably damage the experience of ioNavi system.

Nowadays, most subway stations have provided free WiFi service for passengers by deploying multiple APs. Note that each AP has a unique identifier called BSSID. It is easy to distinguish AP via such an identifier [Sakib et al. 2014; Wu et al. 2013; Li.H. et al. 2014; Wang et al. 2016]. Besides, each subway station is covered by signals from celltowers. Each celltower possesses an identifier called ID. The mobile device

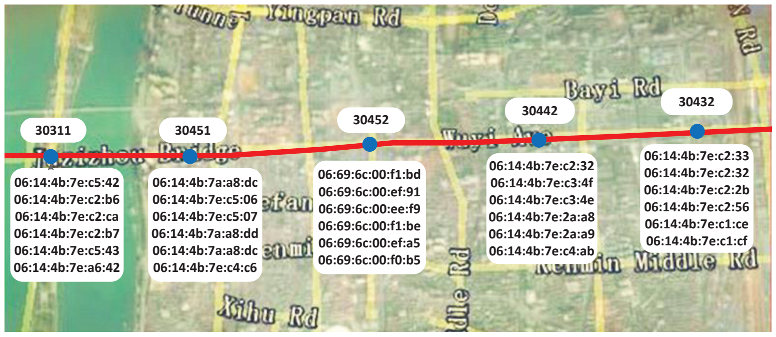


Fig. 3. The signatures of subway stations along the subway line 2 in Changsha, China.

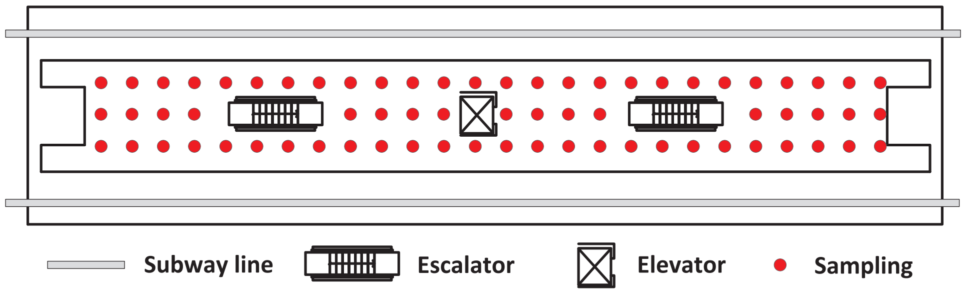


Fig. 4. An example of coarse-grained survey of fingerprint in a subway station.

collects the received signals from both APs and celltowers to extract the sequences of BSSIDs and IDs, which are used as signatures of a subway station. Therefore, the subway station where the user is located can easily be identified using such signatures.

Figure 3 illustrates signatures of each subway station of Line 2 in Changsha, China. The subway line is marked as a red line and the subway stations are marked as blue circles. For example, our measurements indicate that the mobile device connects to the celltower with ID 30452 in Wuyi Square station. Beside, the mobile device scans and records BSSIDs of six APs. The collected the sequences of IDs and BSSIDs serves as signatures of the Wuyi Square subway station.

3.1.2. Starting Point Detection. When a user gets off the train and sends a navigation request to ioNavi, ioNavi needs to know the location of the user, i.e., the starting point of a navigation trace. In this paper, a coarse-grained fingerprint method is proposed to locate a user inside the subway station. This method follows the similar idea of fingerprint based [Paramvir and Venkata 2000; Yang et al. 2012b] and sensor-based methods [Chen et al. 2011; Chen et al. 2011; Chen et al. 2014], but just needs to construct a coarse-grained fingerprint database for each subway station. Note that, the radio map only covers the area of waiting platform. Besides, ioNavi just needs to know the probable location of a user. For instance, as shown in Fig. 4, all users have to take either escalators or elevators to ground. Such escalators and elevators are regarded as crowd-points. In this way, ioNavi just needs to guide the user to crowd-points and does not care about the precise location of the user.

Fingerprint Survey. Fingerprints are surveyed for each subway station by dividing area of interests as a mesh grid, as shown in Fig. 4. In our experiments, the length l of a grid is set to be 2 meters, which is sufficient to produce a good localization performance. Let s denote the number of sampling points in an area A.

Assume m APs are deployed to cover an area A. The Received Signal Strength (RSS) at a location in A acts as its fingerprints. Due to the interference of noises, the observed RSSs at a sampling location is usually unstable and affects the accuracy of localization. To address the negative impact of noises, the unscented kalman filter (UKF) [Sun et al. 2010] is used to filter noises.

Localization. Given a localization requirement of a user inside a subway station, the measured RSSs are used as the input. The bayesian decision theory is used to estimate the location of the user. The bayesian decision theory is a statistical approach, which quantifies the tradeoff between various decisions using the probability and the decision cost [Dieter et al. 2003]. It usually has two assumptions. The first one is that the probability of each decision is known in advance, i.e., the total probability. The second one is that the probability distribution of WiFi signal is known, i.e., the priori probability.

Given a RSS vector R=$(RSS_1, RSS_2, ..., RSS_m)$, which resorts to the Gaussian assumption. The prior probability is

$$p(x)=\frac{1}{\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \tag{1}$$

where x is the received RSS, μ is the expected value of RSS, σ is the standard deviation of RSS. Consequently, the posteriori probability R in each sampling location (S) is:

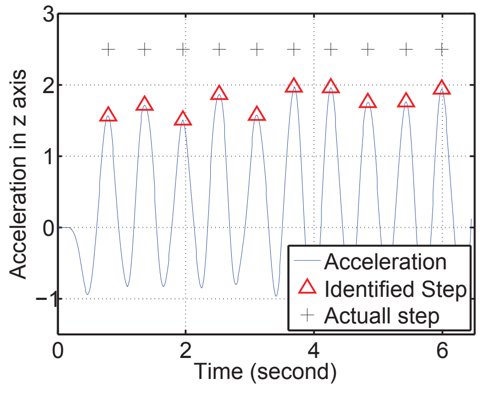
$$p(S_i|R)=\frac{p(R|S_i)p(S_i)}{p(S)}=\frac{p(R|S_i)p(S_i)}{\sum_{k=1}^{n}p(R|S_k)p(S_k)}, \tag{2}$$

where $p(R|S_i)$ is the probability of RSS at a known location S_i, $p(S_i)$ is the priori probability in the i-th sampling point. Since RSSs are independent from different APs, we can have $p(R|S_i)$=$p(R_1|S_i)p(R_2|S_i)\dots p(R_n|S_i)$. When a detected RSS comes, the bayesian decision theory computes the posterior probability of RSSs for each sampling location in the subway station using the prior probability and the total probabilities. The location with the highest posteriori probability is considered as the optimal selection of the location.

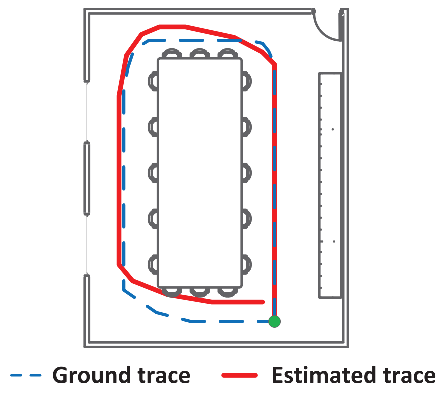
3.1.3. Indoor-outdoor Trace Generation. After locating the starting point of a user in the subway station, the walking traces of a user are estimated using the dead reckoning method. Several dead reckoning methods have been proposed in the literatures [Wang et al. 2012b; Shen et al. 2013; Yang et al. 2012a; Rai et al. 2012], and a brief working flow is presented in this paper. It consists three components: step counting, orientation estimation, and step length estimation.

Step Counting. Several methods have been proposed to count the walking steps of a user [Rai et al. 2012; Wang et al. 2012b; Li et al. 2012]. In this paper, accelerometer readings are is used to count the walking steps of the user by detecting the peaks of acceleration. To improve the accuracy of step detection, the Finite Impulse Response (FIR) filtering method is used to filter out both the high frequency and low frequency noises in acceleration measurements. Figure 5(a) shows that the accuracy of step detection is over 99%, where the black crosses and red triangles represent the actual steps and detected steps from the readings of the accelerometer sensor embedded in a mobile device, respectively.

Orientation Estimation. The magnetometer and gyroscope sensors are used to estimate the orientation of user while walking. Specifically, the magnetometer sensor is used to estimate the absolute direction in the earth coordinate system and the gyroscope sensor is used to estimate the relative direction changes with respect



(a)



(b)

Fig. 5. The detection accuracy of the step counting and the turning points using the dead reckoning method.

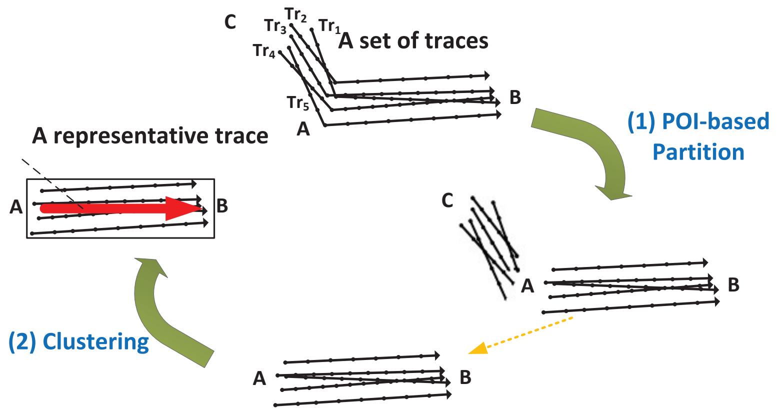


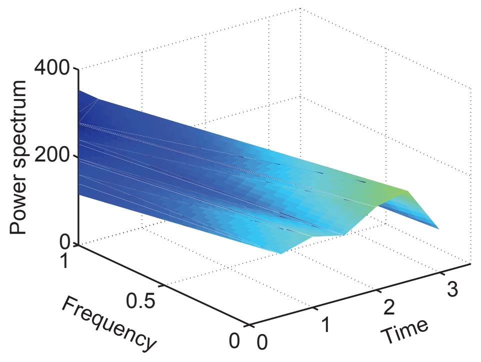
Fig. 6. An illustrative example of our trace extraction method.

to the mobile device platform in our ioNavi system. Such method has been demonstrated to be remarkably accurate in [Wang et al. 2012b; Alzantot and Youssef 2012; Shen et al. 2013]. However, the magnetometer sensor is considerably noisy in indoor environments due to electromagnetic interference. Therefore, several particular directions are used to calibrate the orientation of a user in this paper. For example, the directions of the escalator are in and out. As shown in Fig. 5(b). It can be seen that the turning points are detected with an accuracy of 99% in our experiments. It is sufficient for our trace extraction method (described in Sec. 3.2) the subsequent section.

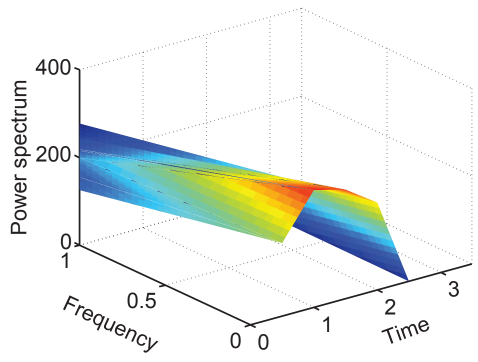
Step Length Estimation. To estimate the step length, a frequency model is adopted, which is represented as $L{=}a{\times}f{+}b$. With the historical knowledge of a user, we can determine the value of coefficients a and b [Li et al. 2012].

3.2. Trace Extraction

Note that those self-motivated users are glad to share their walking traces with other users. Consequently, the server collects large amounts of walking traces for any pair of the starting and end points. However, the crowdsourced traces are inherently incomplete, opportunistic, and noisy, recorded by different users with a wide variety of



(a) Normal walking



(b) Walking through a subway turnstile

Fig. 7. An illustrative example of subway turnstile detection.

Table I. The used POIs in ioNavi system.

	POIs
Indoor space	The starting points of users, Turning points, Elevators, Escalators, Stairs, Subway turnstiles, Exit entrances, WiFi marks
Outdoor space	Turning points, General buildings, Bus stations

mobile devices. Hence, our ioNavi system should be robust and be able to handle heterogeneous, redundant, inaccurate and low-quality data. To solve this issue, a trace extraction method is proposed. First, POI-based partition method is proposed to divide a trace into segments using the detected POIs. Second, an effective similarity measurement algorithm is proposed to calculate the similarity between a pair of traces, which is formulated as a combinatorial optimization problem and is NP-complete. Third, a density peak based clustering algorithm is used to cluster the crowdsourced traces, to estimate the representative traces and thus to remove outlier traces. Last but not least, a basic navigation trace database is constructed to plan an optimal walking trace for a user. The flow chart of our trace extraction method is shown in Fig. 6.

3.2.1. POI-based Partition Method. To divide a trace into segments, the notions of POIs are introduced. POIs are specific points in the environment with unique sensor signatures that can be accurately detected. Users perform distinguishable motion patterns passing through these POIs. For example, the up and down stairs motion is distinctive, compared to normal walking patterns. In this paper, we identify two classes of POIs based on inertial sensors (i.e., accelerometer, gyroscope, and magnetometer) embedded in a mobile device: those based on indoor space and those based on outdoor space. All POIs used in our ioNavi system are listed in Table I. Since many existing works have been proposed to detect different POIs, including elevators, escalators, stairs, exit entrances, WiFi marks, and bus stations [Alzantot and Youssef 2012; Shen et al. 2013; Li et al. 2015; Valentin et al. 2014], we focus on the detection of the subway turnstiles in this paper.

Passengers mostly pass by a subway turnstile to get the train. The walking behaviors of a user nearby a subway turnstile usually consist of stop and walking, stop (find a credit card or coin), then quickly walking (passing through the subway turnstile in a limited opening time), and normal walking finally. As shown in Fig. 7, the Fourier transform is used to calculate the walking frequency from data of the accelerometer of a mobile device. It can be found that the walking pattern via sensing walking through subway turnstiles can be distinguished with a high accuracy.

3.2.2. Traces Clustering. Once a large number of traces are divided into segments using POIs, the next step is to calculate the similarity between two walking traces. These similar walking traces are clustered using the density peak based clustering algorithm [Alex and Alessandro 2014].

Similarity Measurement Between Two Traces. Recall that a trace is divided into a series of segments by POIs. A segment consists of walking points which are the step down points detected by the dead reckoning method. Due to the variation in walking velocity and trace length, it is challenging to accurately calculate the similarity between any pair of traces. Therefore, the similarity measurement problem is considered as an optimization problem and a heuristic solution is proposed. The key idea is that the similarity between two segments is first calculated and the similarity between two traces is then calculated.

Given two traces under the coordinate system, γ^1 and γ^2, where $\gamma=\{x_i, y_i, p_i\}$ and p_i denotes the i-th walking point of a trace, the traces are firstly divided into segments using POIs, i.e., $\gamma^1=\{s^1_1, s^1_2, \ldots, s^1_n\}$, $\gamma^2=\{s^2_1, s^2_2, \ldots, s^2_m\}$, where n and m are the numbers of POIs for γ^1 and γ^2, respectively. Let $d_{i,j}$ denote the Euclidean distance between walking points p_i and p_j, let Z denote the objective value. The object is to find the matching of walking points between two segments such that the total distance of two segments is minimized.

The decision variables $x_{i,j}$ is denoted as:

$$x_{i,j}=\begin{cases}1, & \text{assigning } p_i \text{ to } p_j,\\ 0, & \text{others.}\end{cases} \tag{3}$$

Thus our objective is formulated as:

$$minZ=\sum_{i=1}^{n}\sum_{j=1}^{m} d_{i,j}x_{i,j} \tag{4}$$

$$s.t. \sum_{i=1}^{n} x_{i,j}=1, i=1,2,\ldots,n \tag{5}$$

$$\sum_{j=1}^{m} x_{i,j}=1, j=1,2,\ldots,m \tag{6}$$

$$x_{i,j}=0 \; or \; 1, i=1,\ldots,n, j=1,\ldots,m. \tag{7}$$

We formulate the similarity measurement problem between two traces, called the Measurement of Similarity (MoS) problem.

THEOREM 3.1. *MoS problem is NP-complete.*

PROOF. First we show that MoS problem belongs to NP. It is easy to verify whether $\sum_i \sum_j d_{i,j}x_{i,j} \leq c$ is satisfied in polynomial time, where c is an integer.

Since the Quadratic Assignment Problem (QAP) is known to be NP-complete [Sahni and Gonzalez 1976], we prove that QAP is reducible in polynomial time to our MoS problem. The decision problem of QAP describes that there are two sets: $A=\{1, 2, \ldots, n\text{-}1\}$ and $B=\{1, 2, \ldots, n\text{-}1\}$, two functions $g(i, j)$ and $h(i, j)$, and a constant c. It determines whether there exists an one-to-one mapping $f : A \rightarrow B$, such that $\sum_i \sum_j g_{i,j}h_{i,j} \leq c$. Therefore, QAP is reducible to our MoS problem as follows: $d_{i,j}x_{i,j}=g_{i,j}h_{i,j}(i \neq j)$ and $d_{i,j}x_{i,j}=c'(i=j)$, where c' is a sufficient large constant. Thus the MoS problem is NP-complete. □

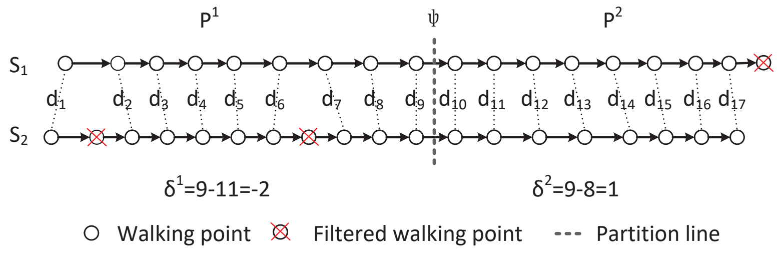


Fig. 8. An illustration of the MoS algorithm. The Euclidean distance between segments S_1 and S_2 is $d_{1,2}$=d_1+d_2+, . . . , +d_{17}.

A MoS algorithm is proposed to find an approximation solution in reasonable time for the MoS problem based on the Hungarian algorithm. We first present a brief introduce of the Hungarian algorithm, which can be easily referred in the literatures [Papadimitriou and Steiglitz 1998; Keysers et al. 2004]. We assume that the weights of the edges are given by the entries of a matrix W and we assume that both components of the graph have N vertices and thus $W{\in}\mathbb{R}^{N\times N}$ is square. The goal of the algorithm is to find a permutation $\pi : \{1, \ldots, N\}{\mapsto}\{1, \ldots, N\}$ to minimize $\sum_{n=1}^{N} W_{n\pi(n)}$ [Papadimitriou and Steiglitz 1998; Keysers et al. 2004].

Despite of its wide usage, the Hungarian algorithm cannot be directly used to solve our MoS problem. That is because that the number of walking points of two segments maybe unequal in our MoS problem due to the difference in walking velocity of users. To solve this issue, a step length based adaptive partition algorithm is proposed. First, the segment is partitioned with ψ interval, where the value of ψ is adaptively changed according to the step length (l) of a user. Second, the redundant walking points are filtered with a pre-defined pattern for each partitioned segment with larger number of walking points. Let l_{s_i} denote the length of the i-th segment, n_{s_i} denote the number of walking points of the i-th segment, and ϵ denote a constant. The value of ψ can be calculated by:

$$\psi{=}\begin{cases}\frac{l_{s_i}\times l}{\epsilon}, & n_{s_i}{\geq}\epsilon,\\ \frac{l_{s_i}\times l}{2}, & n_{s_i}{<}\epsilon.\end{cases} \tag{8}$$

Furthermore, the redundant walking points are filtered based on the partition results for the segment with larger number of walking points. Let P^i denote the set of partitioned subsegments for the i-th segment, i.e., $P^i{=}\{p_k^i|1{\leq}k{\leq}\lceil\frac{l_{S_i}}{\psi}\rceil\}$. Let $n_{p_k^i}$ denote the number of walking points of the k-th subsegment for the i-th segment. Given two subsegments, p_k^1 and p_k^2, we take $\delta{=}|n_{p_k^1}{-}n_{p_k^2}|$. Our ioNavi system randomly filters δ walking points of a subsegment with a larger number of walking points. After that, the Hungarian algorithm is used to solve the MoS problem. Algorithm 1 illustrates the MoS algorithm.

Example. Figure 8 shows an example of the MoS algorithm for two segments S_1 and S_2. The walking points are marked in circles and the filtered walking points are marked in circles and crosses. In this example, the numbers of walking points in segments S_1 and S_2 are different (i.e., $|S_1|$=18 and $|S_2|$=19). According to the constant ϵ=9 in this setting, the segments are divided into two subsegments, i.e., P^1 and P^2, respectively. In subsegment P^1, the number of walking points in segment S_1 is 9 and the number of walking points in segment S_2 is 11. Then, we take δ=-2. Therefore, two points should be filtered for segment S_2 in subsegment P^1. Similarly, one walking point should be filtered for segment S_1 in subsegment P^2. Finally, the Euclidean dis-

ALGORITHM 1: The Measurement of Similarity Algorithm

```
Input: Two segments $S_1$ and $S_2$, a constant $\epsilon$ and step length $l$.
Output: The distance between two segments $Z$.
$\psi$=0, $\delta$=0;
if $|S_1|$=$|S_2|$ then
    Use the Hungarian algorithm;
else
    if $n_{S_1} \geq \epsilon$ then
        $\psi = \frac{l_{S_1} \times l}{\epsilon}$;
    else
        $\psi = \frac{l_{S_1} \times l}{2}$;
    end
    Partition $S_i$ into $P^i$, $P^i = \{p^i_k | 1 \leq k \leq \lceil \frac{l_{S_1}}{\psi} \rceil\}$;
    for $k$=1 to $\lceil \frac{l_{S_1}}{\psi} \rceil$ do
        $\delta = n_{p^1_k} - n_{p^2_k}$;
        if $\delta > 0$ then
            Randomly delete $\delta$ walking points of $p^1_k$ partition;
        else
            Randomly delete -$\delta$ walking points of $p^2_k$ partition;
        end
        $\delta$=0;
    end
    Use the Hungarian algorithm;
end
```

tance between segments S_1 and S_2 is calculated using the Hungarian algorithm, i.e., $d_{1,2}$=d_1+d_2+, . . . , +d_{17}. Systematic evaluation results will be presented in Sec. 4.3.7.

Once the similarities between all pairs of traces are calculated, the next step is to cluster the crowdsourced traces using the trace extraction algorithm (as illustrated in Algorithm 2). Let TR denote a set of traces, TR=$\{\gamma^i | 1 \leq i \leq |TR|\}$, where $|TR|$ is the number of traces. Let S denote a set of segments of a trace and T_c denote the clustered trace. First, the traces are divided into segments using POIs. Second, Algorithm 1 is used to calculate the similarity between two segments. Third, the similarity between two traces is calculated by accumulating the similarity values between segments. Finally, the clustered trace is given using the density peak based algorithm [Alex and Alessandro 2014]. This algorithm has a number of advantages, for example, it is not required to chooses a density threshold, the number of clusters is not required before carrying out clustering, the detected clusters can be represented in an arbitrary shape, and it is very simple. Once clusters are obtained, the representative traces are estimated inside their clusters. The Gaussian Kernel function is used to calculate the local density in the density peak based algorithm [Alex and Alessandro 2014]. The parameter t is a parameter for the cutoff distance (d_c). We take t=2% by referring to the work [Alex and Alessandro 2014]. We do not present the details of the clustering algorithm due to space constraints, for more details, the reader can refer to the literature [Alex and Alessandro 2014].

3.2.3. Trace as Database Tables. The trace clustering results are stored in the database table with fields "ID", "POI_1", "POI_2", "distance", and "route code". ID is the identifier of a trace, POI_1 and POI_2 are the starting and end POIs in a trace, respectively. The distance means the walking distance along a trace. A route code (integer index from 1 to n) is saved instead of the coordinates of the route. For instance, a trace-contributor

ALGORITHM 2: The Trace Extraction Algorithm

```
Input: The set of traces $TR$.
Output: The clustering trace $TR_c$.
$Z$=0, $index$=0, $n$ is the number of POIs of a trace;
for each trace $\gamma^i \in TR$ do
    Divide $\gamma^i$ into segments, $S^i=\{s^i_k|1 \leq k \leq n+1\}$;
end
for $i$=0 to $|TR|$-2 do
    for $j$=$i$ to $|TR|$-1 do
        for $k$=0 to $n$ do
            Use the MoS algorithm (Algorithm 1) for segments $s^i_k$ and $s^j_k$;
        end
        $Z(index)=\sum_{k=0}^{n} z_k$;
        $index$++;
    end
end
Use the density peak based clustering algorithm to generate $TR_c$;
```

Table II. An example of trace database tables.

ID	POI_1	POI_2	distance	route code
0	A	C	5m	1
1	B	C	25m	2
2	C	D	45m	3

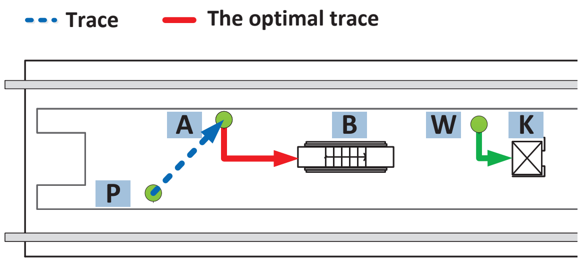
walked along the segment ($A \rightarrow C$), with a total length of around 5m. Another trace-contributor walked along the trace ($B \rightarrow C \rightarrow D$), with a total length of around 70m. The database table is shown in Table II. A navigation-user, e.g., an upcoming passenger, currently locates in point A and wants to arrive at destination D. According to the trace database table, ioNavi takes the combination of two segments, $A \rightarrow C$ and $C \rightarrow D$, to generate a new trace, which is not shared by other users, i.e., $A \rightarrow C \rightarrow D$. Therefore, our ioNavi system plans an optimal trace to guide users to their destinations using this database table.

Besides, some POIs represent the landmarks (e.g., Starbucks) visited by trace-contributors in outdoor environment. Therefore, our ioNavi system queries the input destinations by users in the database table using the boyer-moore algorithm [Tarhio and Ukkonen 1993], which is a simple and quick algorithm. Systematic evaluation results will be presented in Sec. 4.3.8.

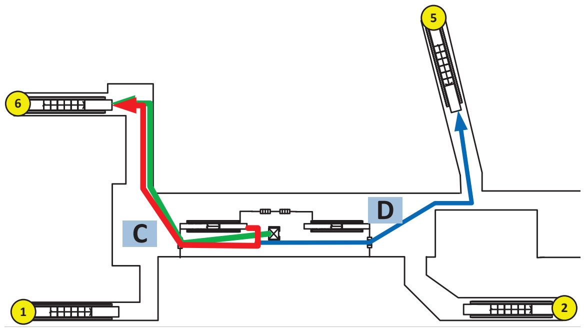
Scale Expansion of Database Tables. A navigation service should automatically extend its areas of navigation service. The GPS readings are used to extend the scale of database tables. The GPS readings provide the latitude and longitude coordinates of a user in space. In order to turn these coordinates into human-readable and searchable text, the reverse geo-coding algorithm is adopted to turn the coordinates into their textual description and attach this text to those traces ending [Feldman et al. 2015]. Our ioNavi system then adds such texts as searchable destinations into the database table to expand the areas of navigation service.

3.3. Traces Based Navigation

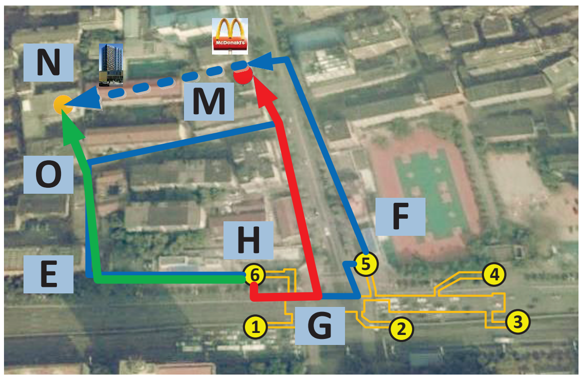
Given the starting point and the destination of a user as well as the database of traces, our ioNavi system should be able to intelligently plan routes and guide a user along the optimal trace in terms of walking distance. The work proposed in [Zheng et al. 2014] detected overlapping segments by measuring their similarity of magnetic field signals and WiFi fingerprint sequences. Two situations were considered, i.e., finding shortcuts



(a) Traces on floor one



(b) Traces on floor two



(c) Outdoor traces

Fig. 9. An example of traces from locations in a subway station to outdoor destination.

from multiple traces with overlapping segments and finding shortcuts from crossing traces. This approach works well when existing walking traces are available, but fails if none of shared trace is available. To overcome this issue, three complicated situations are considered and pre-defined rules are proposed in our ioNavi system. Besides, if the user is off the correct trace, the deviation event will be detected and the user will be notified.

Case 1: There exist many traces from the starting point of a user to its destination. ioNavi recommends the trace with the shortest walking distance. For the i-th trace, the server calculates the walking distance d_i. Then the smallest d_i is obtained for all traces. As shown in Fig. 9, the red line represents the optimal trace. In Fig. 9(a), the user first walks towards an escalator after getting off the train, and then walks away from the subway station via Exit 6 after passing the subway turnstiles, as shown in Fig. 9(b). Finally, the user arrives at the outdoor destination (i.e., the McDonald's as shown in Fig. 9(c).

Case 2: None of shared traces starts at the location of a user but at least one shared trace ends at his destination. As shown in Fig. 1, the user located at point T wants to arrive at bank N. However, none of shared traces starts from the location of the user. Our ioNavi system would recommend a trace whose starting point is closest to the location of the user (e.g., point S). The entire recommended trace is depicted as a red dashed line, i.e., $T \rightarrow S \rightarrow B \rightarrow N$.

Case 3: None of shared traces meets the navigation requirement of a user, but the starting location of the user and its destination location are involved by some shared traces. As shown in Fig. 1, a trace-contributor starts from point S and wants to arrive at bank N, while another trace-contributor visits a shopping mall P starting from point Y. If a new user wants to visit bank N starting from point Y, our ioNavi system finds that none of shared individual traces can directly meet the navigation requirement of the user. A feasible approach is to download two traces, $S \rightarrow N$ and $Y \rightarrow P$. The user is first guided to point P, and then visits point N. It is clear that this walking trace is not the best one in terms of walking distance. Accordingly, a trace combination method is proposed to efficiently generate the optimal trace. Note that, each shared trace can be partitioned into a sequence of segments by the detected POIs. Thus, the Dijkstra [Skiena 1990] algorithm is used to estimate the reachable walking traces among all segments. As shown in Fig. 1, the optimal trace is obtained, i.e., $Y \rightarrow B \rightarrow W \rightarrow N$.

3.3.1. Tracking and Deviation Handling. Our ioNavi system tracks the navigation procedure of a user based on the dead reckoning method. If the user is off the recommended trace, the deviation event will be detected and the user will be notified. Note that, due to interference of a large volume of passenger flow, it will be difficult to perfectly follow the navigation trace for a navigation-user. Therefore, the deviation events are wrong turnings while walking. The turning direction is calculated based on gyroscope and magnetometer readings. If a user has a wrong turning direction against with navigation trace, an alert is then given to the user. Furthermore, a new trace is calculated based on the current location of the user.

4. PERFORMANCE EVALUATION

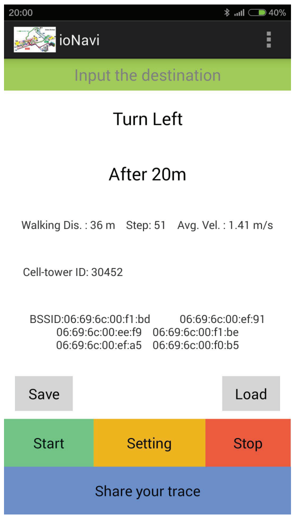
In this section, the implementation details of our ioNavi system is first presented, and the evaluation methodology and setups is introduced. The performance of each component of our ioNavi system is then tested.

4.1. Implementation

The implementation of our ioNavi system consists of two components: a mobile client and a navigation pipeline. The mobile client runs on an Android mobile device and the navigation pipeline works on a server. The mobile client interface can be installed in different Andorid mobile devices which support WiFi, 3G/4G, GPS, light sensor, and inertial sensors (i.e., accelerometer, gyroscope, and magnetometer). The navigation pipeline was implemented on two types of servers, including a Lenovo computer (with WiFi, 32GB RAM, i7 CPU) and a cloud server on Microsoft Azure platform using a network optimized A9 virtual machine (with 16 cores, 112GB RAM and Ubuntu Linux). For most of our experiments, WiFi or 3G/4G networks were used for communication between the mobile devices and the server.

Mobile Client. The mobile client software was used to record signals of celltowers and WiFi as well as readings of inertial sensors (i.e., accelerometer, gyroscope, and

(a) Yingbin Road station



Turn Left

After 20m

Walking Dis. : 36 m Step: 51 Avg. Vel. : 1.41 m/s

Cell-tower ID: 30452

BSSID:06:69:6c:00:f1:bd 06:69:6c:00:ef:91
06:69:6c:00:ee:f9 06:69:6c:00:f1:be
06:69:6c:00:ef:a5 06:69:6c:00:f0:b5

(b) A snapshot of ioNavi system

Fig. 10. An illustration of our experimental environment and a snapshot of our ioNavi client.

Table III. The details of Android mobile phones used in our experiments.

	WiFi	GPS	3G/4G	Light sensor	Inertial sensor	RAM	Processor
Sumsung Galaxy S4	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	2GB	Exynos 5410
MI	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	2GB	Nvidia Tegra4
Nubia Z5	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	2GB	Qualcomm APQ

magnetometer) with timestamps. Instant readings of inertial sensor are used to estimate the walking traces of users. The data of celltowers and WiFi is used to locate users in subway stations. Those walking traces, locations in subway stations, and data of sensors are then automatically packed and transmitted to the server with WiFi or 3G/4G networks.

Server Configuration. The server was implemented in a computer and a cloud virtual machine with Ubuntu Linux system using two threads. The first thread was used to receive and store incoming data from the mobile client, the second thread was used to process those data (e.g., POIs detection and trace clustering), generate navigation instructions, and track the walking progress of a user.

4.2. Evaluation Methodology and Setups

A prototype system of ioNavi was implemented in different Android mobile phones, including Sumsung Galaxy S4, MI, and Nubia Z5. The details of those Android mobile phones are shown in Table III. We conducted experiments on the subway Line 2 in Changsha, China[2]. The subway Line 2 in Changsha has 19 stations with different numbers of exit entrances. For example, the Yingbin Road station has 6 exit entrances, while the Wuyi Square station has 8 exit entrances. Figure 10(a) shows the experimental environment of the Yingbin Road station. Figure 10(b) shows the GUI of our ioNavi system, where navigation instructions are provided. Once the destination is given and the start button is pressed by a user, an optimal trace is recommended from the server and the instructions will be displayed on the screen.

[2]http://www.railway-technology.com/projects/

Table IV. The trace planning performance of our ioNavi system.

	Dataset A	Dataset B	Dataset C
F metric	0.96	0.98	0.96

20 volunteers were invited to participate in our experiments, including 12 passengers and 8 members in our research group. Besides, the volunteers are different in genders, heights and weights. Volunteers themselves determine the walking traces. Our ioNavi system recorded the BSSIDs of access points and IDs of celltowers to identify the subway station in subway Line 2. Meanwhile, it recorded inertial sensor readings to estimate the walking traces of a user. Note that, we randomly selected 3 subway stations to test the navigation performance of our ioNavi system. The subway station identification performance of our ioNavi system was tested in the entire subway Line 2. In order to bootstrap our ioNavi system, 268, 205, and 230 walking traces were collected in those three stations, respectively. At least 3 walking traces were collected for a volunteer in each station using different Android mobile devices. These volunteers walked towards their destinations under navigation instructions provided by our ioNavi system (See Fig. 10(b)) and the mobile devices were hold in hand. These traces covered most experimental areas within one kilometer around each subway station. In order to evaluate the navigation performance of our ioNavi system, volunteers were asked to collect 150 (dataset A), 102 (dataset B), and 120 (dataset C) walking traces in those three stations.

4.3. Performance Evaluation

4.3.1. Trace Planning. Given a navigation request, our ioNavi system guides users to their destinations by providing the optimal navigation traces in terms of walking distance. To test the trace planning performance of our ioNavi system, datasets A, B, and C were used and the F metric is proposed, that is:

$$F=\frac{Number\ of\ the\ optimal\ traces}{Number\ of\ traces}. \quad (9)$$

It can be seen from Table IV that the F values are larger than 0.96 for those three datasets. Furthermore, we present more details of trace planning. As shown in Fig. 9, four contributed traces were considered from an indoor location A in the Yingbin Road subway station to an outdoor destination M (i.e., the McDonald's). That is, there are three blue traces and one red trace from A to M, while the red trace is the optimal one. Volunteers were asked to participate in this experiment. Note that, they were not aware of the navigation routes at the beginning, but only knew the destination. The trace planning performance was evaluated under three cases.

Case 1: There are several traces from the starting point of a user to its destination. In this experiment, the volunteers started using the McDonald's as their destination and point A as their starting point (see Fig. 9). Then, a walking trace is given to their mobile devices by our ioNavi system. It can be seen that all volunteers can successfully reach their destination along the recommended trace (i.e., the red trace). This experiment confirms that ioNavi system is able to recommend an optimal navigation trace in this case.

Case 2: None of shared traces starts at the location of a user but at least one shared trace ends at its destination. As shown in Fig. 9, none of those shared traces starts at point P and ends at the destination M. If a volunteer input the McDonald's as its destination, ioNavi returned a blue dash trace from P to A as the first stage and another trace from point A to the destination M. Accordingly, all volunteers can successfully reach the destination M.

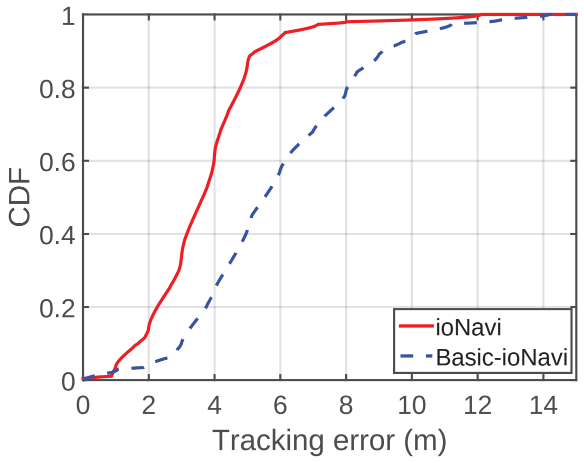


Fig. 11. The tracking errors achieved by our ioNavi and Basic-ioNavi systems.

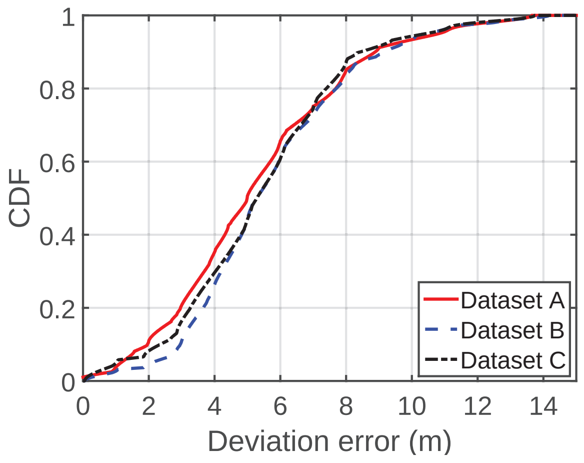


Fig. 12. The deviation events detection accuracy.

Case 3: None of shared traces meets the navigation requirement of a user, but its starting point and destination location are involved by some shared traces. In Fig. 9, the trace-contributor shared two traces, $A \rightarrow M$ and $W \rightarrow N$. In our experiments, volunteers can be guided from A to N by ioNavi system. Obviously, none of shared traces meets such navigation requirement. However, a new trace can be generated by ioNavi system to guide volunteers to their destinations, i.e., $A \rightarrow B \rightarrow C \rightarrow H \rightarrow E \rightarrow O \rightarrow N$.

4.3.2. Navigation Performance. Datasets A, B, and C were used to test the navigation performance of our ioNavi system. We compared the navigation performance of ioNavi system with the navigation system without trace clustering (denoted as Basic-ioNavi). In our experiments, all volunteers can successfully arrive at their destinations along their optimal traces under the guidance of ioNavi and Basic-ioNavi systems. Figure 11 shows that 90% tracking errors produced by ioNavi are less than 5.2m while 90% tracking errors produced by Basic-ioNavi are less than 9.2m. That is because the dead

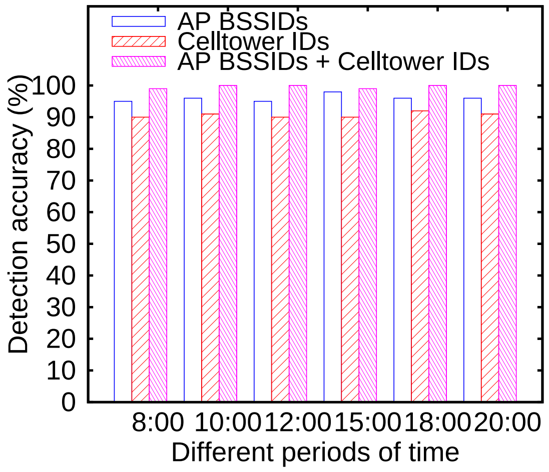


Fig. 13. The subway station detection accuracy.

reckoning method produces drift tracking errors as the walking distance increases. Our ioNavi system uses trace clustering to provide high-quality traces to users while the Basic-ioNavi provides low-quality traces to users in most cases without trace clustering.

4.3.3. Deviation Detection. Furthermore, the deviation detection performance of our ioNavi system was tested on datasets A, B, and C. The location where a deviation event happened was recorded and the distance between the start point and the location where the deviation event happened can be calculated. Figure 12 shows that 90% deviation errors produced by ioNavi are less than 9.2m for datasets A, B, and C.

4.3.4. Subway Station Identification Accuracy. The celltower IDs and AP BSSIDs were collected at different subway stations during experiments in three days and finally a subway station identification database was generated. Note that, the constructed database is updated every month. The average identification accuracy achieved by celltower IDs, AP BSSIDs, and their combination on different periods is presented in Fig 13. It is shown that the identification accuracy is 100% using the combination of celltower IDs and AP BSSIDs on different period. The subway station identification accuracies achieved by celltower IDs and AP BSSIDs are 91% and 95%, respectively. Experimental results demonstrate that our method can effectively identify the subway station on any period in a day.

4.3.5. Detection Accuracy. Our ioNavi system uses the fingerprint method to invoke user starting point detection inside a subway station when the user gets off the train. We first tested the localization performance on datasets A, B, and C. We then compared the localization performance of our ioNavi system with the RADAR system proposed in [Paramvir and Venkata 2000]. The localization errors are shown in Fig. 14. Our ioNavi system achieves an average location error of 3.2m, which is significantly smaller than the RADAR system (4.5m). Such localization accuracy is sufficient to guide a user.

Furthermore, we tested the localization performance of our ioNavi system under different numbers of available APs in a subway station. We randomly selected 20 sample locations in the Yingbin Road station and the number of APs was increased from 1 to 6. As shown in Fig. 15, the localization error decreases as the number of APs is increased

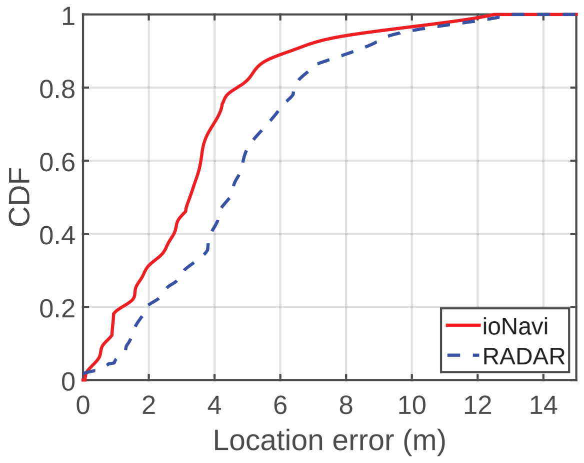


Fig. 14. Localization accuracy achieved on the subway station data.

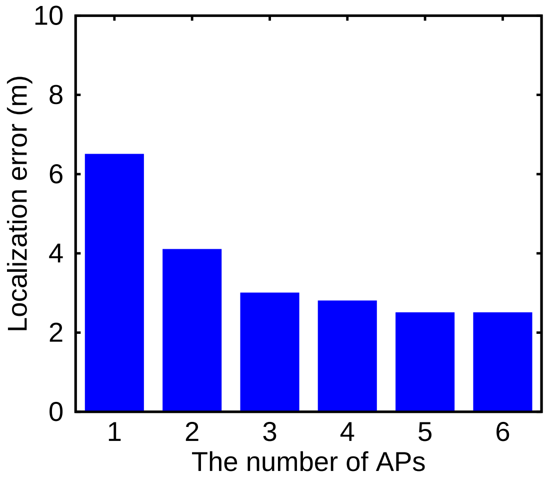


Fig. 15. Localization accuracy achieved under different numbers of APs.

to 6. However, the localization error becomes stable as the number of APs is further increased from 3. Therefore, at least 3 APs are needed to locate a user in subway stations using our ioNavi system.

4.3.6. POI Detection Accuracy. We further tested the POI detection accuracy on Datasets A, B, and C using inertial sensor readings (accelerometer, gyroscope, and magnetometer). Those datasets covered six types of POIs, including turning points, elevators, escalators, stairs, exit entrances, and subway turnstiles. The ground truth of POIs was generated manually. Table V shows the detection matrix for the six types of POIs. We can see that each POI can be detected with a high accuracy.

4.3.7. Trace Clustering Performance. The trace clustering performance is crucial for indoor-outdoor navigation. The trace clustering performance was evaluated at the Yingbin Road station. For this evaluation, the volunteers were asked to walk along the

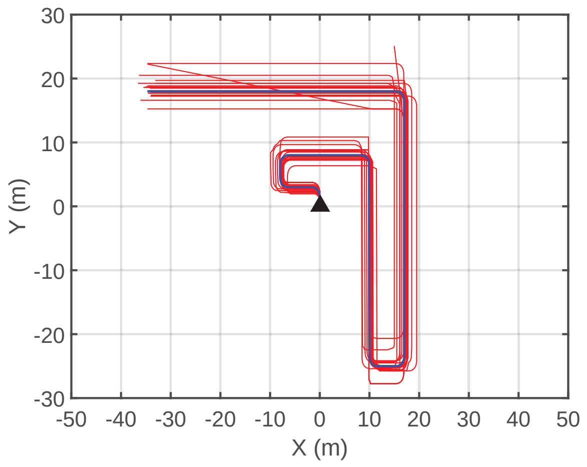


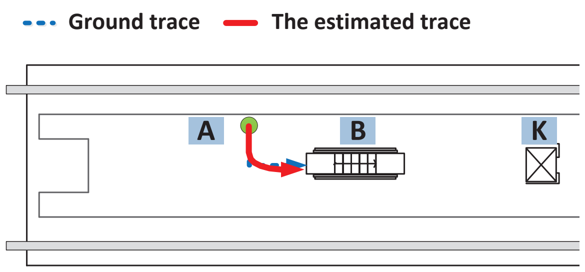
Fig. 16. The walking traces of volunteers and the clustered trace. The red curves represent walking traces of volunteers, the blue curve represents the clustering trace, and the triangle represents the starting point of volunteers.

Table V. The detection accuracy of POIs

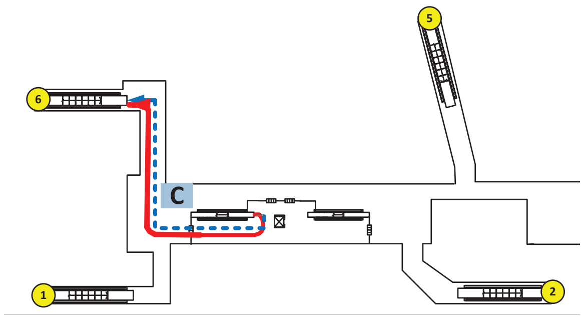
	Elevators	Escalators	Stairs	Exit entrances	Turning	Turnstiles	Total
Elevators	27	1	0	0	0	0	28
Escalators	1	11	0	0	0	0	12
Stairs	0	0	15	0	0	0	15
Exit entrances	0	0	0	20	0	0	20
Turning	0	0	0	0	81	0	81
Turnstiles	0	2	0	0	0	18	20

trace $A \rightarrow B \rightarrow C \rightarrow H \rightarrow M$ (Fig. 17), where point A represents the starting point of a volunteer, point M represents the end point of a volunteer, and points B, C, H represent POIs along the trace, respectively. Our ioNavi system recorded inertial sensor readings to estimate the walking traces of volunteers. The traces of volunteers and POIs are shown in Fig. 16, where the red curves represents the walking traces of volunteers, the blue curve represent the clustering trace, and the triangle represents the starting point of volunteers. It can be seen that our trace clustering algorithm can effectively extract high-quality trace from a large number of low-quality traces. Figure 17 shows the recommended navigation trace and the ground-truth trace. The blue line denotes the ground-truth trace, while the red line denotes the recommended trace. It is clear that our method can successfully and effectively guide the user to reach the destination along a recommended indoor-outdoor trace.

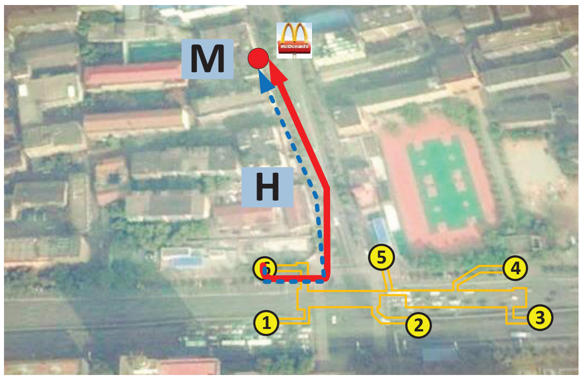
4.3.8. Response Delay. We deployed ioNavi using different Andoird mobile devices and servers. The server has two types including a Lenovo computer and a cloud server on Microsoft Azure platform. The response delay incudes user starting point detection, destination query, and path planning. For the fingerprint based starting point detection, it took 0.1s in average to locate a mobile device in the subway station using either a Lenovo computer or a cloud server. For a destination querying, it took 0.01s in average to query destination from 200 destinations using either a Lenovo computer or a cloud server in average. For path planning, it took 0.25s and 0.2s in average to provide an optimal walking path using a Lenovo computer and a cloud server in average, respectively. In summary, the response delay teste on two types of servers were about 0.36s



(a) Traces on floor one



(b) Traces on floor two



(c) Outdoor traces

Fig. 17. The accuracy of indoor-outdoor navigation trace extraction. The blue line denotes the ground-truth trace, the red line denotes the recommended trace.

and 0.31s, respectively, which are acceptable for most users. Note that, the response delay can further be reduced using more powerful machines.

5. DISCUSSION AND FUTURE WORK

In this paper, an architecture is proposed for joint indoor-outdoor navigation, which involves a series of design concerns. Although those essential design concerns have been tackled in this paper, the following issues should be further addressed in our future work.

Robustness to Celltowers and APs. Currently, our ioNavi system uses signals of celltowers and APs to identify the subway station where a user is located in and the starting points of a user. Therefore, the user cannot be navigated in subway stations which are not covered by either celltowers or APs. In our future work, we will investigate more techniques to improve the navigation robustness in places without celltowers and APs.

Scalability Issue. It is clear that the number of participated users dominates the navigation performance and the scalability of ioNavi. In order to improve the experience of our ioNavi system, a large number of users should be invited to participate in the trace collection process. It is challenging to motivate users in crowdsourcing systems. To address this problem, a reward mechanism [Yang et al. 2012b; Luo et al. 2015; Zhang et al. 2015] is necessary to be designed in our future work.

Ability of Constructing Indoor-outdoor Maps. A precondition for our ioNavi system is the availability of an indoor-outdoor map around the subway station. Although many indoor and outdoor localization systems simply assume that the required maps are available, the joint indoor-outdoor navigation systems always lack available maps. Many existing works have been proposed to construct indoor maps [Alzantot and Youssef 2012; Philipp et al. 2014; Gao et al. 2014b; Chen et al. 2015; Jiang et al. 2013]. CrowdInside [Alzantot and Youssef 2012], Mapgenie [Philipp et al. 2014], and the work in [Jiang et al. 2013] rely on aggregated user motion traces generated from inertial data. Jigsaw [Gao et al. 2014b] and CrowdMap [Chen et al. 2015] use both images and inertial data to reconstruct the indoor floorplans. Our ioNavi system is significantly different from these systems. It gradually obtains an indoor-outdoor map of the subway station using the trace clustering method.

Other Indoor Environments. Although our ioNavi system was evaluated in subway stations, it can be customized to other indoor environments that have similar construction structure of buildings, e.g., underground shopping malls. Specially, semi-unsupervised or unsupervised learning techniques [Wang et al. 2012a; Valentin et al. 2014; Prakash and Nithya 2014] can be used to extend our method to new indoor environments.

Information Privacy. Users are allowed to share their walking traces in our ioNavi system. However, a user might not want others to know his locations and walking traces after sharing. Therefore, information privacy protection should be considered in our future work.

6. RELATED WORK

Mobile Device-based Indoor Localization Techniques. In outdoor environment, the users uses GPS to locate their locations. In complex indoor environments, indoor localization face serious challenges due to the unavailability of GPS signal. Recently, a variety of indoor localization methods have been proposed, such as the fingerprint based [Yang et al. 2012a; Liu et al. 2012; Li et al. 2014b] and the dead reckoning based methods [Alzantot and Youssef 2012; Rai et al. 2012; Wang et al. 2012b; Shen et al. 2013; Li et al. 2012].

With the increasing deployment of APs, many indoor localization systems are proposed using the WiFi signal as fingerprint [Yang et al. 2012a; Liu et al. 2012]. The navigation systems based on conventional WiFi localization methods require the indoor radio map to cover the entire indoor area. Thus, the computation complexity and the time consumption considerably increases with the spread of the service area. Currently, we find that a coarse-grained radio map only covers the area around trains, but does not cover the entire area inside each subway station. In this case, the WiFi based localization methods can only provide a coarse-grain initial location for a user when getting off the train. Therefore, it is insufficient to provide an indoor navigation service via the conventional WiFi localization technology. Moreover, the outside of each subway station lacks stable WiFi signals. Thus, although those WiFi based localization methods are feasible to enable indoor navigation in theory, they are not suitable for indoor-outdoor navigation in the scenario of subway systems. Moreover, the WiFi localization method only provides the localization services and has to deploy path planning algorithm to enable online navigation services.

For the dead reckoning method, the current location of a user is obtained by computing the step counts, the step length, and the walking heading, given a previous location [Harle 2013]. A fundamental challenge is how to control the accumulation of localization error. The built-in inertial sensor in a smart phone is usually low in cost and the sensor readings are always with strong noises. On the other hand, the irregular walking pattern of users, such as arm swing, is an essential factor, which influences the localization accuracy. UnLoc [Wang et al. 2012b] imposed a constraint on the location of the phone since it relies on the gyroscope readings for dead reckoning and landmark matching. Walkie-Markie [Shen et al. 2013] used landmarks to calibrate the drift errors of the dead reckoning method. Some approaches try to improve the accuracy of dead reckoning by reducing the orientation estimating error. Li et al. [Li et al. 2012] corrected the magnetometer heading using the map based particle filtering.

Mobile Device Based Navigation Systems. Recently, many research organizations and companies begin to provide navigation systems based on smart phones [Lee et al. 2011; Zheng et al. 2014; Glanzer and Walder 2010; Shu et al. 2015; Dong et al. 2015; Teng et al. 2015]. Nerimi [Lee et al. 2011] proposed a WiFi-based subway stop notification system for Seoul subways in Korea. Travi-Navi [Zheng et al. 2014] motivated each volunteer to actively generate and share a trace with associated vision information. Thus, a follower can download a trace and follow that trace to arrive at the end of the trace. Our ioNavi system is different from Travi-Navi in several aspects. First, Travi-Navi only focuses on indoor navigation. Our ioNavi system provides joint indoor-outdoor navigation service, which has not been addressed by any existing system. Second, the user needs to hold the mobile device uprightly and steadily during walking to obtain a good image quality in Travi-Navi, and the overhead caused by image downloading and uploading is very high. The work in [Glanzer and Walder 2010] introduced a pedestrian navigation system with human motion recognition based on magnetic field map. FollowMe [Shu et al. 2015] used the geomagnetic field to guide a user by providing the "scent" left by the leaders or previous travelers. iMoon [Dong et al. 2015] used crowdsourced photos to build a smartphone-based indoor navigation system. Our ioNavi system is significantly different from the work proposed in [Glanzer and Walder 2010], FollowMe [Shu et al. 2015], and iMoon [Dong et al. 2015] systems. Our ioNavi system is a joint indoor-outdoor navigation system.

Trace Clustering. Due to the large amounts of collected mobility data, trajectory analysis becomes a very active research topic. Many methods have been proposed [Ferreira et al. 2013; Lee et al. 2007; Lee et al. 2008] to analyze a set of outdoor traces. However, different from outdoor traces, indoor traces are more challenging due to the complex interaction and diverse sensor readings. Therefore, existing outdoor trace analysis methods cannot be directly used in the case of indoor traces. In this paper, an efficient method is proposed to perform indoor trace clustering.

7. CONCLUSION

In real life, people have a strong need for navigation from a large open indoor environment to an outdoor destination. This paper presents a framework to enable a joint indoor-outdoor navigation service (called ioNavi). It enables passengers to easily deploy indoor-outdoor navigation service for subway transportation systems in a crowdsourcing way without any comprehensive localization system. The system attracts some passengers to generate and share their walking traces, and then extracts navigation traces from these inaccurate traces. A subsequent user can then guided to the destination by a recommended navigation trace. Extensive experiments were conducted in subway transportation systems. The experimental results show that ioNavi achieves excellent navigation performance from an uncertain location inside a subway station to a nearby outdoor destination. Although ioNavi is motivated to enable indoor-outdoor navigation for subway transportation systems, it also provides a general design framework for joint indoor-outdoor navigation in other scenarios.

REFERENCES

M. Alessandro, W. Daniel, and S. Dieter. 2009. Indoor positioning and navigation with camera phones. *IEEE Pervasive Computing.* 8, 2 (2009), 22–31.

R. Alex and L. Alessandro. 2014. Clustering by fast search and find of density peaks. *Science.* 344, 6191 (2014), 1492–1496.

M. Alzantot and M. Youssef. 2012. Crowdinside: Automatic construction of indoor floorplans. In *Proc. of SIGSPATIAL*. Redondo Beach, California.

S. Chen, M. Li, K. Ren, and C. Qiao. 2015. CrowdMap: Accurate reconstruction of indoor floor plans from crowdscourced sensor-rich videos. In *Proc. of IEEE ICDCS*. Ohio, USA.

T. Chen, D. Guo, Z. Yang, H. Chen, and X. Luo. 2011. Improving the efficiency of localization-oriented network adjustment in wireless sensor networks. *IEEE communications letters* 15, 9 (2011), 983–985.

T. Chen, Z. Yang, Y. Liu, D. Guo, and X. Luo. 2011. Localization in non-localizable sensor and ad-hoc networks: a localizability-aided approach. In *Proc. of IEEE Infocom*. Shanghai, China.

T. Chen, Z. Yang, Y. Liu, D. Guo, and X. Luo. 2014. Localization-oriented network adjustment in wireless Ad Hoc and sensor networks. *IEEE Transactions on Parallel and Distributed Systems* 25, 1 (2014), 146–155.

F. Dieter, H. Jeffrey, L. Lin, S. Dirk, and B. Gaetano. 2003. Bayesian filtering for location estimation. *IEEE Pervasive Computing.* 2, 3 (2003), 24–33.

J. Dong, Y. Xiao, M. Noreikis, Z. Ou, and A. Yla-Jaaski. 2015. iMoon: Using smartphones for image-based indoor navigation. In *Proc. of ACM SenSys*. Seoul, South Korea.

D. Feldman, C. Sung, A. Sugaya, and D. Rus. 2015. idiary: From gps signals to a text-searchable diary. *ACM Transactions on Sensor Networks.* 11, 4 (2015), 60.

N. Ferreira, J. Klosowski, ScheideggerC., and SilvaC. 2013. Vector field k-means: Clustering trajectories by fitting multiple vector fields. *Eurographics Conference on Visualization.* 32, 3 (2013), 201–210.

R. Gao, Y. Tian, F. Ye, K. Bian, Y. Wang, T. Wang, and X. Li. 2014a. Sextant: Towards ubiquitous indoor localization service by photo-taking of the environment. *IEEE Transactions on Mobile Computing.* 13, 9 (2014), 1–14.

R. Gao, M. Zhao, T. Ye, F. Ye, Y. Wang, K. Bian, T. Wang, and X. Li. 2014b. Jigsaw: Indoor floor plan reconstruction via mobile crowdsensing. In *Proc. of ACM MobiCom*. Maui, Hawaii.

G. Glanzer and U. Walder. 2010. Self-contained indoor pedestrian navigation by means of human motion analysis and magnetic field mapping. In *Proc. of IEEE WPNC*. Dresden, Germany.

R. Harle. 2013. A survey of indoor inertial positioning systems for pedestrians. *IEEE Communications Surveys.* 15, 3 (2013), 1281–1293.

Y. Hu, Y. Xiong, W. Huang, X.Y. Li, Y. Zhang, X. Mao, and C. Wang. 2015. Lightitude: Indoor iositioning using ubiquitous visible lights and COTS devices. In *Proc. of IEEE ICDCS*. Ohio, USA.

Y. Jiang, Y. Xiang, X. Pan, K. Li, Q. Lv, R.P. Dick, and M. Hannigan. 2013. Hallway based automatic indoor floorplan construction using room fingerprints. In *Proc. of ACM UbiComp*. Zurich, Switzerland.

D. Keysers, T. Deselaers, and H. Ney. 2004. Pixel-to-pixel matching for image recognition using hungarian graph matching. *Pattern Recognition.* (2004), 154–162.

Y. Kuo, P. Pannuto, K. Hsiao, and P. Dutta. 2014. Luxapose: Indoor positioning with mobile phones and visible light. In *Proc. of ACM MobiCom*. Maui, Hawaii.

I. Lee, G. Yoon, and D. Han. 2011. Nerimi: WiFi-based subway navigation system. In *Intelligent Radio for Future Personal Terminals*. Daejeon, South Korea.

J. Lee, J. Han, X. Li, and H. Gonzalez. 2008. TraClass: Trajectory classification using hierarchical region-Based and trajectory based clustering. In *Proc. of ACM PVLDB*. Auckland, New Zealand.

J. Lee, J. Han, and K. Whang. 2007. Trajectory clustering: A partition-and-group framework. In *Proc. of ACM SIGMOD*. Beijing, China.

F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao. 2012. A reliable and accurate indoor localization method using phone inertial sensors. In *Proc. of ACM UbiComp*. Pittsburgh, Pennsylvania, United States.

L. Li, P. Hu, C. Peng, G. Sen, and F. Zhao. 2014a. Epsilon: A visible light based positioning system. In *Proc. of USENIX NSDI*. Seattle, WA.

L. Li, G. Shen, C. Zhao, T. Moscibroda, J.H. Lin, and F. Zhao. 2014b. Experiencing and handling the diversity in data density and environmental locality in an indoor positioning service. In *Proc. of ACM MobiCom*. Maui, Hawaii.

M. Li, P. Zhou, Z. Yang, Z. Li, and G. Shen. 2015. IODetector: A generic service for indoor/outdoor detection. *ACM Transactions on Sensor Networks.* 11, 21 (2015), 28.

Li.H., K. Wu, Q. Zhang, and L.M. Ni. 2014. CUTS: improving channel utilization in both time and spatial domain in WLANs. *IEEE Transactions on Parallel and Distributed Systems* 25, 6 (2014), 1413–1423.

H. Liu, Y. Gan, J. Yang, S. Sidhom, Y. Wang, Y. Chen, and F. Ye. 2012. Push the limit of Wi-Fi based localization for smartphones. In *Proc. of ACM MobiCom*. Istanbul, Turkey.

T. Luo, S. Kanhere, H. Tan, F. Wu, and H. Wu. 2015. Crowdsourcing with Tullock contests: A new perspective. In *Proc. of IEEE INFOCOM*. Hong Kong.

C.H. Papadimitriou and K. Steiglitz. 1998. Combinatorial optimization: Algorithms and complexity. *Dover Publications* (1998).

B. Paramvir and P. Venkata. 2000. RADAR: An in-building RF-based user location and tracking system. In *Proc. of IEEE Infocom*. Israel.

D. Philipp, P. Baier, C. Dibak, F. Durr, K. Rothermel, S. Becker, and D Fritsch. 2014. Mapgenie: Grammar-enhanced indoor map construction from crowd-sourced data. In *Proc. of IEEE PerCom*. Budapest, Hungary.

V.J. Prakash and D.L. Nithya. 2014. A survey on semi-supervised learning techniques. In *arXiv preprint arXiv:1402.4645*.

A. Rai, K. Chintalapudi, V. Padmanabhan, and R. Sen. 2012. Zee: Zero-effort crowdsourcing for indoor localization. In *Proc. of ACM MobiCom*. Istanbul, Turkey.

S. Sahni and T. Gonzalez. 1976. P-complete approximation problems. *J. ACM.* 23, 3 (1976), 555–565.

M.N. Sakib, J.B. Halim, and C.T. Huang. 2014. Determining location and movement pattern using anonymized WiFi access point BSSID. In *Proc. of IEEE SecTech*. Hainan, China.

G. Shen, C. Zhuo, P. Zhang, M. Thomas, and Y. Zhang. 2013. Walkie-Markie: Indoor pathway mapping made easy. In *Proc. of USENIX NSDI*. Lombard, IL.

Y. Shu, K. Shin, T. He, and J. Chen. 2015. Last-Mile navigation using smartphones. In *Proc. of ACM MobiCom*. Paris, France.

S. Skiena. 1990. Dijkstra's Algorithm. *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica, Reading, MA: Addison-Wesley.* (1990), 225–227.

C. Sun, H. Kuo, and C. Lin. 2010. A sensor based indoor mobile localization and navigation using Unscented Kalman Filter. In *Proc. of IEEE LPLANS*. Indian Wells, CA, USA.

J. Tarhio and E. Ukkonen. 1993. Approximate boyer-moore string matching. *Society for Industrial and Applied Mathematics.* 22, 2 (1993), 243–260.

X. Teng, D. Guo, X. Zhou, and Z. Liu. 2015. Poster: An indoor-outdoor navigation service for subway transportation systems. In *Proc. of ACM SenSys*. Seoul, Republic of Korea.

R. Valentin, K. Panagiota, S. Rik, and M. Mahesh. 2014. A semi-supervised learning approach for robust indoor-outdoor detection with smartphones. In *Proc. of ACM Sensys*. Memphis, TN.

G. Wang, S. Zhang, K. Wu, Q. Zhang, and L.M. Ni. 2016. Tim: Fine-grained rate adaptation in WLANs. *IEEE Transactions on Mobile Computing* 15, 3 (2016), 748–761.

H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R.R. Choudhury. 2012a. No need to war-drive: unsupervised indoor localization. In *Proc. of ACM MbiSys*. United Kingdom.

H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. Choudhury. 2012b. Unsupervised indoor localization. In *Proc. of ACM MobiSys*. United Kingdom.

S. Wang, S. Fidler, and R. Urtasun. 2015. Lost shopping! Monocular localization in large indoor spaces. In *Proc. of IEEE ICCV*. Santiago, Chile.

K. Wu, H. Li, L. Wang, Y. Yi, Y. Liu, D. Chen, X. Luo, Q. Zhang, and L.M. Ni. 2013. hJam: Attachment transmission in WLANs. *IEEE Transactions on Mobile Computing* 12, 12 (2013), 2334–2345.

B. Xie, G. Tan, and T. He. 2015. Spinlight: A high accuracy and robust light positioning system for indoor applications. In *Proc. of ACM SenSys*. Seoul, Korea.

D. Yang, G. Xue, X. Fang, and J. Tang. 2012b. Crowdsourcing to smartphones: incentive mechanism design for mobile phone sensing. In *Proc. of ACM MobiCom*. Istanbul, Turkey.

Z. Yang, C. Wu, and Y. Liu. 2012a. Locating in fingerprint space: wireless indoor localization with little human intervention. In *Proc. of ACM MobiCom*. Istanbul, Turkey.

X. Zhang, G. Xue, R. Yu, D. Yang, and J. Tang. 2015. Truthful incentive mechanisms for crowdsourcing. In *Proc. of IEEE INFOCOM*. Hong Kong.

Y. Zheng, G. Shen, L. Li, C. Zhao, M. Li, and F. Zhao. 2014. Travi-Navi: Self-deployable indoor navigation system. In *Proc. of ACM MobiCom*. Maui, Hawaii.