Implicit Multimodal Crowdsourcing for Joint RF and Geomagnetic Fingerprinting

Jiajie Tan, Hang Wu, Ka-Ho Chow, and S.-H. Gary Chan, *Senior Member, IEEE*

Abstract—In fingerprint-based indoor localization, fusing radio frequency (RF) and geomagnetic signals has been shown to achieve promising results. To efficiently collect fingerprints, implicit crowdsourcing can be used, where signals sampled by pedestrians are automatically labeled with their locations on a map. Previous work on crowdsourced fingerprinting is often based on a single signal, which is susceptible to signal bias and labeling error. We study, for the first time, implicit multimodal crowdsourcing for joint RF and geomagnetic fingerprinting. The scheme, termed UbiFin, exploits the spatial correlation among RF, geomagnetic, and motion signals to mitigate the impact of sensor noise, leading to highly accurate and robust fingerprinting without the need for any explicit manual intervention. Using clustering and dynamic programming, UbiFin correlates spatially different signals and filters effectively mislabeled signals. We conduct extensive experiments on our campus and a large multi-story shopping mall. Efficient and simple to implement, UbiFin outperforms other state-of-the-art crowdsourcing schemes to construct RF and geomagnetic fingerprints in terms of accuracy and robustness (cutting fingerprint error by 40 percent in general).

Index Terms—Fingerprinting, site survey, implicit crowdsourcing, multimodal signals, RF, geomagnetic field, IMU

1 Introduction

Fingerprinting has emerged as a promising approach for indoor localization [1], [2], [3]. Fingerprint-based localization generally consists of a training (site survey) phase followed by an operation (location query) phase. In the training phase, signals are recorded at different locations in the feasible area of a map or floor plan. The signal values and their labeled locations, termed *fingerprints*, form a database. In the subsequent query phase, users are localized by matching their sampled signals with the ones in the database. Their locations are then indicated on the map.

Radio frequency (RF), such as Wi-Fi and Bluetooth, and geomagnetic field are commonly used as fingerprint signals due to their pervasiveness and location-based variation. Combining them has been shown to achieve good localization accuracy and robustness [4], [5], [6], [7]. However, the training phases of both signals involve considerable manual calibration efforts which are time-consuming and labor-intensive. Site surveys for RF signals are often conducted with dedicated surveyors traveling to every predefined location to collect and label the signals [8], [9], [10], while magnetic fields are usually collected by walking through all paths multiple times [4], [11]. Furthermore, such surveys have to be repeated frequently to keep the fingerprints up-to-date so as to accommodate environmental changes.

To make the fingerprinting process more efficient, recent works have explored *implicit crowdsourcing* [12], [13], [14], [15]. They sample signals transparently from users' mobile devices (upon users' approval, of course) and compute signal locations as labels at a server without any manual intervention [16]. Despite prior works on RF and geomagnetic crowdsourcing, they often study the signals individually and separately [12], [14], [17]. This makes the fingerprinting process vulnerable to signal noises. Furthermore, previous approaches often consider user paths independently without leveraging their correlation to mitigate signal noise and estimation error [12], [17], [18]. Designing a robust, adaptive, and highly accurate fingerprinting system based on implicit crowdsourcing for simultaneous RF and geomagnetic fingerprinting remains a challenging open problem.

We propose UbiFin, a novel and robust approach for *ubi*quitous joint RF and geomagnetic *fin*gerprinting via implicit crowdsourcing. To the best of our knowledge, this is the first piece of work of such nature. UbiFin crowdsenses multimodal signals (including received signal strength or RSS of RF, magnetic fields, and inertial sensor measurements), and automatically and efficiently constructs RF and magnetic fingerprint databases simultaneously without the need for any user labeling. The floor plan, readily available from map services, is used to constrain and calibrate effectively the crowdsourced user paths. The joint RF, magnetic, and motion design achieves highly accurate fingerprints, more efficient survey, performance robustness to large scale, applicability to constrained and open space, and deployability of RF and magnetic fusion localization. Note that though we discuss UbiFin in the context of fingerprinting for indoor localization, the labeled signals and generated user trajectories would be useful in other applications such as Wi-Fi deployment optimization [19], compass calibration, and crowd analysis [20].

UbiFin is based on the observation that RF, magnetic field, and motion signals have strong spatial and complementary features to mitigate fingerprinting errors, as presented below. RF signal is an effective location indicator over a larger scale (e.g., 5–10 meters) but shows less differentiability within a

• *The authors are with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong. E-mail: {jtanad, hwuav, khchowad, gchan}@cse.ust.hk.*

Manuscript received 25 September 2020; revised 28 May 2021; accepted 7 June 2021. Date of publication 10 June 2021; date of current version 6 January 2023. (Corresponding author: Jiajie Tan.)
Digital Object Identifier no. 10.1109/TMC.2021.3088268

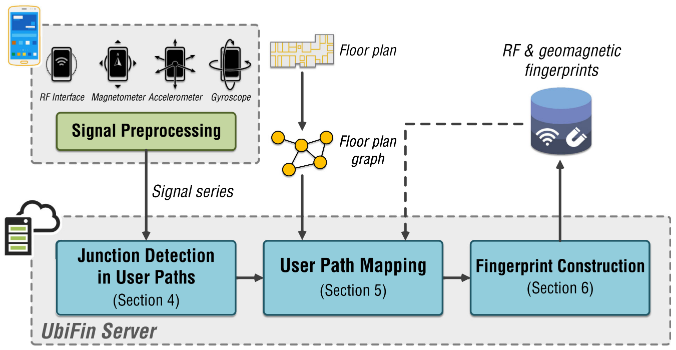

RF Interface
Magnetometer
Accelerometer
Gyroscope
Signal Preprocessing
Floor plan
Floor plan graph
RF & geomagnetic fingerprints
Signal series
Junction Detection in User Paths (Section 4)
User Path Mapping (Section 5)
Fingerprint Construction (Section 6)
UbiFin Server

Fig. 1. The system architecture of UbiFin.

short range due to signal noises. Its sampling frequency is also relatively low (one sample per second to tens of seconds, depending on the app, operating system, and device state) [14]. The sparse data make it challenging to identify the exact landmarks like turning points in a user path.[1] In contrast, magnetic field is less noisy and can be sampled at a much higher rate (tens of samples per second). Often studied based on sample sequence [17], it enjoys short-range differentiability but suffers from global location ambiguity (a sequence may be matched with multiple locations anywhere in the site). Motion signals of acceleration and angular velocity provide excellent clues on travel distance and turning direction for trajectory construction [12], [13]. However, the model suffers from user heterogeneity and cumulative error over time. By correlating the multimodal signals from all users, UbiFin mitigates the impact of signal bias and labeling errors, resulting in highly accurate fingerprints.

We illustrate in Fig. 1 the system architecture of UbiFin. When untrained pedestrians move freely in the venue, their carried smartphones collect signals via embedded RF, magnetic, and inertial sensors (accelerometer and gyroscope). The signal series are first preprocessed before being uploaded to the server for fingerprinting. The server side has the following three modules containing the major steps to construct the fingerprint databases:

1) *Junction detection in user paths*: Junctions are the key positions that connect different regions. The module uses magnetic field and motion signals jointly to detect the encountered junctions in each of the user paths. By aligning user paths based on their magnetic fields, it critically examines the unique signal patterns possibly at junctions, such as turnings and path forking/merging, and accurately identifies them with density-based clustering. User paths are then segmented based on the recognized junctions.
2) *User path mapping*: Given a segmented path, the module maps it onto the graph representation of the floor plan. To achieve it, UbiFin puts the route in the region which best matches its multimodal signals. This is a joint signal consideration: the motion signals provide clues on user activities (e.g., turning and walking distance) which fit on the map layout, while the ambient signals (e.g., RF and magnetic fields) correlate and constrain the path with the previously fingerprinted signals, if any. By formulating the path mapping problem using *dynamic programming*, UbiFin efficiently obtains the route on the map.
3) *Fingerprint construction*: The module generates fingerprints by filtering out misplaced signals. The key idea is that correct signals at a location are similar and consistent with each other, and hence the misplaced signals can be easily detected as outliers. By adopting clustering techniques to remove the possible placement mistakes in the path mapping process, fingerprint databases can be constructed with high accuracy.

1. In this paper, we refer to the term “path” as the actual trip of a user, while “route” and “trajectory” are referred to the estimated path on the floor plan.

The entire fingerprinting process works in an evolutionary and incremental manner. Initially, there is no labeled signal (i.e., fingerprint) in the site. As more signals are crowdsensed, more RF and magnetic fingerprints are gradually formed. They serve as references to improve the process of path mapping and fingerprinting. Furthermore, the evolutionary design also makes UbiFin capable of updating outdated fingerprints due to environmental changes in an online manner.

We have implemented UbiFin and conducted extensive experimental studies to validate its design and performance. The experiments are carried out in two large-scale real-world scenarios: one floor at our campus (around $14,000\,\mathrm{m}^2$) and a two-story shopping mall (around $46,000\,\mathrm{m}^2$). Our experimental results show that UbiFin substantially outperforms the state-of-the-art approaches in terms of accuracy and robustness (cutting fingerprint error by 40 percent in general).

The remainder of the paper is organized as follows. We first discuss related work in Section 2, followed by the preliminaries and data preprocessing in Section 3. In Section 4, we introduce how to segment user paths based on junction detection. We then discuss in Section 5 the efficient path mapping algorithm to localize signals using dynamic programming. Section 6 presents the technique of filtering misplaced signals and generating the fingerprint databases. We present the experimental results in Section 7, and conclude in Section 8.

2 Related Work

Prior works have studied using explicit crowdsourcing to reduce the effort of site surveys. Different from implicit methods, explicit crowdsourcing relies on users' manual inputs, such as location labels and error feedbacks [21], [22]. Though these schemes are able to ease the survey burden to some extent, the need for manual input still brings inconvenience to naïve users and negatively affects user participation. By contrast, UbiFin is designed in an implicit manner where the process of data collection and fingerprinting is entirely transparent to users.

There has also been a large body of works studying fingerprint database construction under implicit crowdsourcing. A common method is to recover user trajectories based on inertial measurement units (IMU) and then label signals with locations accordingly. Prior works devote much effort to accurate estimation of walking distance and heading for better trajectory inference [23], [24], [25]. However, it is inevitable to face severe issues of error accumulation, especially in long paths. To address that, other works propose to recognize special signal patterns to calibrate trajectories or model parameters, such as stationary devices [26], user turnings [12], and signal landmarks [27]. As a comparison, UbiFin treats user movements as regional features and learns trajectories by seeking the most compatible route on a floor plan. Furthermore, benefit from the implicit calibration at discovered junctions, UbiFin is also able to effectively mitigate the accumulated trajectory errors.

Many works explore the correlation of ambient signals, such as RF and magnetic field, to construct fingerprints [14], [26], [28]. They leverage the location-dependent nature of ambient signals to infer the correspondence among different user paths. However, they usually rely on a *single* signal, lacking robustness against signal bias. Meanwhile, most of them are designed dedicatedly for certain targeting signals and hence cannot be easily adapted to others. UbiFin, on the contrary, considers multimodal signals available on commercial smartphones. By fusing the complementary characteristics of different signals, UbiFin effectively mitigates signal bias and is robust against heterogeneous devices and environmental changes.

Integrating floor plans is not a burden for crowdsourcing systems because map resources are accessible from map service providers (e.g., Google Maps and Baidu Maps) or site owners. There have been many works that enhance fingerprinting accuracy based on map information. Some works leverage the betweenness of signals to construct latent graphs, and then align the graphs with map layouts [14], [29], [30]. However, they are vulnerable to signal noises since the generated signal graph may not be homeomorphic to the floor plan, resulting in erroneous fingerprints. Other works use layout information to constraint user movements. For instance, MapCraft [31] discretizes the floor plan and adopts conditional random fields (CRFs) to estimate user trajectories. Zee [12] applies a particle filter to model the nonlinear location distribution within the accessible regions. Those works all require high computational power and are apt to be influenced by signal noise. In UbiFin, we design an effective floor plan representation and propose a dynamic programming algorithm to efficiently localize user paths.

Other techniques such as simultaneous localization and mapping (SLAM) and signal reconstruction have also been studied to eliminate or reduce the cost of site surveys. Some works employ SLAM to correlate opportunistic signals with motion patterns [32], [33], [34]. They yet rely on high-precision sensors to calibrate trajectories, while UbiFin does not have assumptions on sensors or user paths. On the other hand, exponential moving average (EMA) [35], Gaussian process regression (GPR) [36], and matrix completion [37] have been applied to reconstruct or update fingerprint databases based on partially sampled signals. TuRF [38] introduces a path-based fingerprint collection method in which signals are collected while users are walking along predefined paths. The works in [30] and [39] apply transfer learning to adapt the outdated fingerprint databases to the current environment. Although the above works alleviate the workload of site surveys, prior knowledge of signals, and manual intervention is still required. In UbiFin, we construct and update fingerprints via implicit crowdsourcing with neither existing fingerprints nor explicit user participation.

TABLE 1
Major Notations Used in UbiFin

Notation	Definition
$\mathcal{L}$	Set of RPs
$\boldsymbol{l}_i$	Coordinate of the ith RP
$\mathcal{D}_R$	RF fingerprint database
$\boldsymbol{r}_i$	RF RSSI vector at $\boldsymbol{l}_i$
$\mathcal{D}_B$	Magnetic field fingerprint database
$\boldsymbol{b}_i$	Magnetic feature vector at $\boldsymbol{l}_i$
P	User path
S	Signal series bundle in a user path
$\boldsymbol{s}_i$	The ith signal series segment in S
$\mathcal{G}$	Floor plan graph
V	Vertex set in $\mathcal{G}$
v_i	The ith vertex in $\mathcal{G}$
E	Edge set in $\mathcal{G}$
M	Path mapping

3 Preliminaries and Preprocessing

In this section, we present the preliminaries and the preprocessing module of UbiFin. We introduce the modeling of fingerprints and crowdsourced signals in Section 3.1. In Section 3.2, we present the graph representation of a floor plan. Finally, we discuss in Section 3.3 the signal preprocessing before uploading signals to the processing server. The major notations used in the paper are summarized in Table 1.

3.1 Fingerprint and Signal Modeling

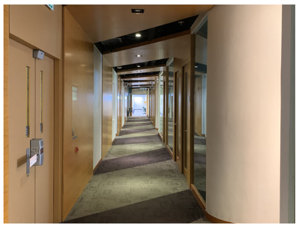
A *fingerprint* refers to the signals associated with a specific location (known as a reference position or RP). Let the set of RPs in the area of interest be $\mathcal{L} = \{\boldsymbol{l}_i \mid 1 \leq i \leq n\}$, where n is the total number of RPs. The fingerprint database that contains all the fingerprints is denoted by $\mathcal{D} = \{\langle \boldsymbol{l}_i, \boldsymbol{u}_i \rangle \mid \boldsymbol{l}_i \in \mathcal{L}\}$, where $\boldsymbol{u}_i$ is a fingerprint signal at RP $\boldsymbol{l}_i$. In this work, we discuss the construction of the RF (mainly Wi-Fi) fingerprint database $\mathcal{D}_R$ and the geomagnetic field database $\mathcal{D}_B$. Specifically, the Wi-Fi signal at $\boldsymbol{l}_i$ is an RSS vector $\boldsymbol{r}_i = \langle r^i_1, r^i_2, \ldots \rangle$, where r^i_j is the RSS from the jth AP at $\boldsymbol{l}_i$; the magnetic signal at $\boldsymbol{l}_i$ is an attitude-invariant feature vector $\boldsymbol{b}_i = \langle b^i_s, b^i_v \rangle$, where b^i_s is the magnetic field intensity, and b^i_v is the vertical component of the field along with the gravity direction. It is worth noting that different types of signals may correspond to different RP sets because of their diverse sampling rates and spatial distributions.

Signals in UbiFin are collected via implicit crowdsourcing, in which untrained users carrying their sensing devices roam in the site and record signals without manual operation. We refer to the location series of a user trip as a *path* P. Multimodal signals collected along P form a bundle of *signal series*[2] $S = \{A, \boldsymbol{\Omega}, R, B\}$, where A, $\boldsymbol{\Omega}$, R, and B are the time series of accelerations, angular velocities, Wi-Fi RSS vectors, and geomagnetic field features, respectively. Please note that the locations of signal samples in S are not labeled by users during the collection and hence need to be determined in the work.

3.2 Floor Plan Representation

We model an indoor space as a composition of multiple *region*s. From hundreds of indoor maps examined, we classify

2. For the sake of simplicity, a "signal series bundle" is sometimes called a "signal series".


(a) TYPE-I (1-D space).

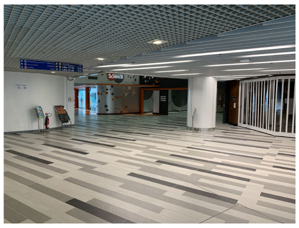
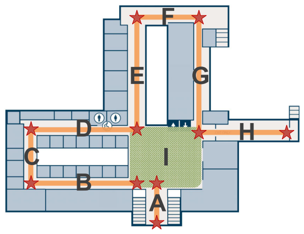
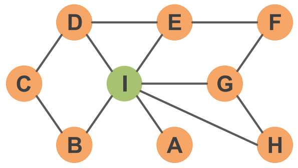

(b) TYPE-II (2-D space).

Fig. 2. Typical examples of the two types of regions.



(a) Floor plan.



(b) Graph representation.

Fig. 3. An illustrative example of the floor plan representation.

the majority of regions into two categories, namely *TYPE-I* and *TYPE-II*. A TYPE-I region is an abstraction of 1-D space where people within always travel along a single line (see Fig. 2a for example). For instance, corridors and hallways can be regarded as such type (the width of a TYPE-I region is usually less than $3\,\mathrm{m}$ and there is no intersection or turn in the middle). In addition, we also view certain traffic facilities (e.g., staircases and escalators) as TYPE-I regions since user movements inside are generally constrained. TYPE-II regions correspond to broader spaces such as squares, atrium, and rooms (see Fig. 2b for example), where pedestrians can move in any direction. We practically treat all the regions other than TYPE-I as TYPE-II. To describe the connectivity between regions, we abstract the locations that connect neighboring regions as *junction*s. Junctions are often corners, cross-sections, doors, and so on.

Based on the region definition above, we formally model the floor plan as an undirected graph $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$, where $\boldsymbol{V}$ and $\boldsymbol{E}$ are the sets of vertices and edges, respectively. A vertex $v_i \in \boldsymbol{V}$ represents a region in the map (either TYPE-I or TYPE-II). An edge $(v_i, v_j) \in \boldsymbol{E}$ indicates the connectivity between two adjacent regions v_i and v_j. According to the definition of junctions, each edge is associated with a junction. Note that there can be multiple representations of a floor plan due to different interpretations of region types. However, it does not affect path mapping performance much because UbiFin can always find the most compatible routes in the given floor plan graph (details are explained in Section 5).

We show in Fig. 3 an illustrative example of a floor plan and its graph representation. Fig. 3a shows the layout and its region partition (for the sake of simplicity, our illustration involves only the public area). The space is divided into eight TYPE-I regions (the orange lines labeled from A to H) and one TYPE-II region (the green area labeled as I). We mark the junctions as red stars. The constructed graph is demonstrated in Fig. 3b, where vertices correspond to the regions with the same labels in Fig. 3a and edges indicate their connectivity.

3.3 Signal Series Preprocessing

Crowdsourced data are generally noisy due to the diverse and unpredictable user behaviors. Preprocessing is thus required before feeding them to the later modules of UbiFin. To achieve this, UbiFin filters signal series by recognizing user activities and device attitudes.

User Activity Recognition. UbiFin expects the signals collected while users are walking naturally in the venue. However, unpredictable user activities, such as making phone calls or shaking phones, distort regular signals and lead to incorrect junction detection and erroneous path mapping. To mitigate the influence, we apply classification techniques to detect whether a segment of signal series is under normal walking status. The classifier is implemented based on the support vector machine (SVM) [40], [41]. Both time- and frequency-domain features are extracted from acceleration series of a 10 seconds. We collect a set of offline data from 12 users to train the model. By performing recognition on the segments of crowdsourced signal series, UbiFin truncates the paths and keeps only the continuous parts collected while walking (with a duration of at least 20 seconds).

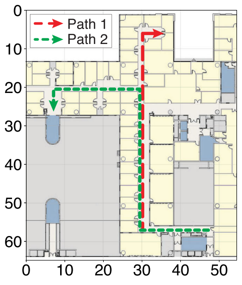
Phone Attitude Recognition. Users have diverse preferences for placing smartphones. For instance, some people are used to holding their phones in hand, while others prefer to put them in pockets or handbags. Different phone attitudes and positions result in inconsistent signal measurements, even at the same location. In this work, we are interested in the signals collected when users hold smartphones in front of their bodies (as the normal position of reading text or viewing maps). Such attitude provides stable signal measurements and is consistent with the application scenarios of location-based services (e.g., using a navigation app). In UbiFin, we adopt a simplified implementation of A^3 [42] to estimate device attitude based on gyroscope and accelerometer. The method employs a Butterworth filter [43] to filter out high-frequency noise. Attitudes are then derived by continuously integrating angular velocities and opportunistically calibrated by gravity readings. Given accurate device attitudes in terms of *pitch* and *roll*, we retain the series with desired poses ($-90^\circ \leq pitch \leq 15^\circ$ and $-15^\circ \leq roll \leq 15^\circ$) [23].

4 Junction Detection in User Paths

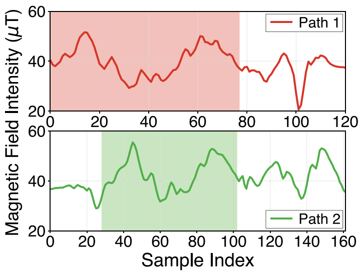
Users generally have common behaviors at junctions, such as turning or entering new regions. It inspires us to recognize junctions according to such patterns. In this section, we introduce a novel and robust junction detection method by correlating magnetic fields and motion patterns in multiple user paths. We discuss in Section 4.1 how to find the common sections in paths using magnetic fields. In Section 4.2, we then present the robust junction recognition algorithm.

4.1 Correlating Paths Using Magnetic Fields

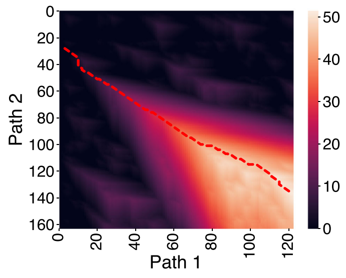
Commercial off-the-shelf smartphones can measure magnetic fields at much higher rates ($\geq$50 Hz) than RF ($\leq$1 Hz) due to the characteristics of sensors and the restriction of operating systems. The high-frequency measurements provide fine-grained signal dynamics in paths. UbiFin hence leverages magnetic fields to correlate different paths, i.e., to identify their common sections in paths.



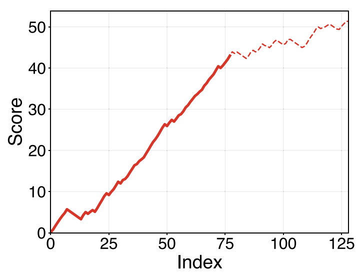
(a) Paths.



(b) Magnetic fields.



(c) Scoring matrix.



(d) Scores along the trace in Fig. 4c.

Fig. 4. An illustrative example of correlating paths using Smith-Waterman (SW) algorithm. Fig. 4a shows two paths with a common section. Fig. 4b demonstrates their magnetic field series. Fig. 4c visualizes the scoring matrix constructed by SW. Fig. 4d plots the scores along the selected trace in the scoring matrix (the dotted line in Fig. 4c).

Speed variation and device heterogeneity are two major challenges in matching magnetic series. The variation of speeds causes shape distortion on magnetic series, such as stretching (i.e., sample insertion due to a low speed) and compression (i.e., sample deletion due to a high speed) [44]. The heterogeneity of devices, on the other hand, results in measurement offsets at same locations. These make it difficult to perform series comparisons directly. To address the above, UbiFin applies a modified Smith-Waterman (SW) algorithm to extract the local alignment of matched sub-series [44], [45]. The alignment between magnetic samples reflects their location correspondence (and thus the common path sections).

Given two series of magnetic fields (Fig. 4b as an example), the algorithm first constructs a scoring matrix $\boldsymbol{W}$ using dynamic programming [44]. At each step (say, comparing the pair of $\boldsymbol{b}_i$ in the first path and $\boldsymbol{b}_j$ in the other path), SW considers the possible cases of sample matching, insertion, or deletion, and records in $\boldsymbol{W}_{ij}$ the maximum score of potential alignments ending there. In particular, we employ a re-scaled Gaussian kernel to represent the score between two matched samples, i.e.,

$$f(\boldsymbol{b}_i, \boldsymbol{b}_j) = 2\exp\left(-\frac{\left\|\boldsymbol{b}_i - \boldsymbol{b}_j\right\|^2}{\sigma^2}\right) - 1, \tag{1}$$

where the kernel width σ controls the degree of similarity. We also empirically designate the gap penalty of -0.4 for the case of sample insertion or deletion.

To address the device heterogeneity issues in terms of measurement offsets, we consider replacing the magnetic vector $\boldsymbol{b}$ in Equation (1) with a mean-removed vector $\boldsymbol{b}'$, i.e.,

$$\boldsymbol{b}' = \boldsymbol{b} - \bar{\boldsymbol{b}}, \tag{2}$$

where $\bar{\boldsymbol{b}}$ is the mean magnetic vector in the past 5 seconds. Fig. 4c visualizes an example of the constructed scoring matrix.

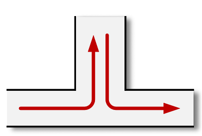
Generally, a pair of matched sub-series corresponds to a series of successive sample alignments and hence generates a segment of monotonically increasing scores, as demonstrated in Fig. 4d. We thus search the scoring matrix W for paths with increasing scores to obtain possible matchings. Specifically, we start from the highest score in $\boldsymbol{W}$ and trace back towards the previously highest scores (the dotted line in Fig. 4c). Note that the obtained scores along the traced path may contain unmatched portions because of the slow decay in scores. Plus, due to signal fluctuation and speed variation, even the matched segment may not be strictly monotonically increasing. To take these into consideration, we adopt sliding window techniques and extract the segment (the solid line in Fig. 4d) whose gradient is greater than a predefined threshold (say, 0.8 of the overall gradient). The corresponding coordinates in W thus form the alignment between two series.

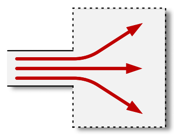
4.2 Junction Recognition

Given the common sections in different user paths, UbiFin correlates specific activity patterns to recognize junctions. On one hand, users usually take turns at junctions (Fig. 5a). At any junction, there should be many paths turning at a similar location. Otherwise, it may be a false positive such as entering a room or avoiding obstacles. On the other hand, junctions are usually the positions where path forking or merging happens. Path forking means that common path sections diverge (Fig. 5b), while path merging indicates that different paths begin to follow the same way (Fig. 5c). We can thus recognize junctions based on turning, path forking, and path merging.

Fig. 6 demonstrates a toy example to explain the above idea. The example consists of three connected corridors (Fig. 6a). Three different user paths, labeled as P_1, P_2, and P_3, pass through the area. We can easily recognize turnings by detecting the drastic change in the angular velocities (Fig. 6b). However, it cannot find all the junctions in P_2 and P_3. On the other hand, magnetic series can be used to correlate different user paths. By recognizing the common sections among paths, it cross-verifies the first turnings of P_1 and P_2 are actually taken at the same junction, and so do the last turnings in P_1 and P_3. Besides, magnetic fields reveal the forking and merging points in paths. For instance, P_1 and P_2 have similar magnetic fields until encountering B, which indicates that B is a forking point. Likewise, junction A is a merging point for P_1 and P_3. By combining the above, we can learn that, besides the turning points, P_2 also passes by junction B and P_3 encounters junction A.

Algorithm 1 presents the major procedures of detecting junctions in the path P^*. We use $\boldsymbol{\Omega}_i$ and B_i to denote the series of angular velocities and magnetic fields of S_i, respectively.


(a) Turning.


(b) Forking.

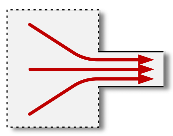
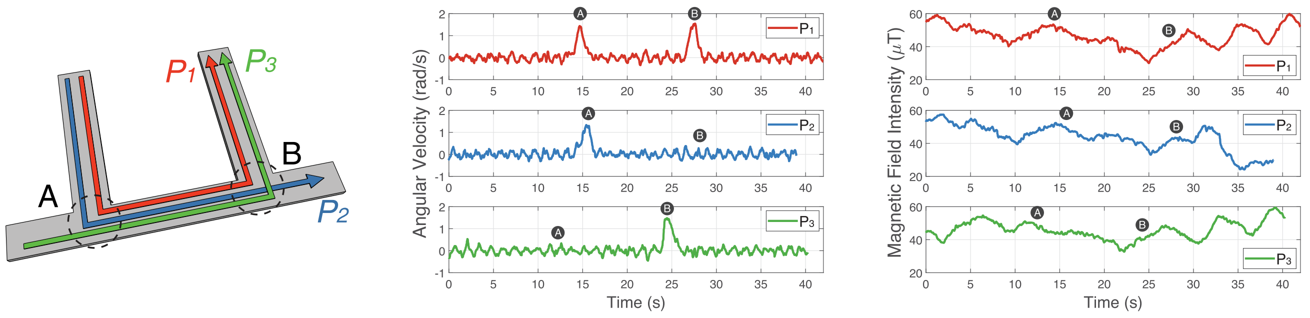

(c) Merging.

Fig. 5. Illustrations of typical behaviors at junctions.



(a) Layout and walking paths. (b) Angular velocities around Z axis. (c) Geomagnetic fields.

Fig. 6. An illustrative example of junction detection. A and B are two junctions. P_1, P_2, and P_3 are three walking paths. The circled letters indicate the time of encountering corresponding junctions.

The set T^* contains the timestamps at which potential junctions are detected in P^*. We first identify turnings by applying the conventional peak-detection method which compares points with neighboring values to find significant local maxima in the angular velocities of the Z-axis [46] (lines 1 and 3). From line 2 to line 9, we convey the junction information from other paths by correlating their magnetic fields. Let H_i be the alignment between B^* and B_i obtained by the modified SW algorithm (Section 4.1), where $H_i(t_j^i)$ represents the timestamp of which the magnetic sample in B^* corresponds to the sample in B_i at t_j^i (line 4). If there is a turning at t_j^i in S_i, we believe that a turning is also possible at the corresponding timestamp $H_i(t_j^i)$ in S^* and hence add $H_i(t_j^i)$ to T^* (line 6). On the other hand, the endpoints of each pair of matched subseries may indicate merging or forking points. We also add the timestamps of them to T^* (line 8). Finally, we employ the DBSCAN clustering [47], [48] on T^* (line 10) to merge the redundant potential points and eliminate false-positive detections. DBSCAN is a density-based method that groups elements close to each other. We employ DBSCAN because it does not require the number of clusters in advance and is effective to find outliers. The centroids of the resulted clusters C are regarded as the timestamps of encountering actual junctions (line 11).

Algorithm 1. Junction Detection

```
input: A signal series $S^*$ of path $P^*$ and a batch of signal
       series $\mathcal{S} = \{S_1, S_2, \ldots\}$
output: The set of timestamps when encountering junctions
        in the path $P^*$
1  $T^* \leftarrow$ TurningDetection($\Omega^*$);
2  foreach $S_i \in \mathcal{S}$ and $S_i \neq S^*$ do
3      $T_i \leftarrow$ TurningDetection($\Omega_i$);
4      $H_i \leftarrow$ SmithWaterman($B^*, B_i$);
5      foreach $t_j^i \in T_i$ and $H_i(t_j^i) \neq \emptyset$ do
6          Add $H_i(t_j^i)$ to $T^*$;
7      end
8      Add the first and the last timestamps of each matched
       segment in $S^*$ to $T^*$;
9  end
10 $C \leftarrow$ DBSCAN($T^*$);
11 return the centroids of the clusters in $C$;
```

Given the above, UbiFin partitions a path P into multiple *path segments* according to the detected junctions. The corresponding signal series S can be segmented into $\{s_i \mid 1 \leq i \leq m\}$, where s_i is the *signal segment* between two consecutive junctions and m is the number of segments.

5 User Path Mapping

We present in this section a novel and efficient path mapping algorithm that localizes the segmented signal series on the given floor plan using *dynamic programming* (DP) [49]. We first describe the path mapping problem in Section 5.1. In Section 5.2, we introduce in detail the cost function which measures the mapping error between a signal segment and a region. Finally, we present in Section 5.3 the DP implementation for computing the optimal route with the minimum mapping cost.

5.1 Path Mapping Problem and Formulation

We first introduce the path mapping problem and its formulation. Recall that signal segments are designed to be associated with vertices in the floor plan graph, and junction points correspond to edges. To map the path P (that is, finding a route on the floor plan), we can equivalently find a subgraph in the graph $\mathcal{G}$ which best matches its signal segments S.

Formally, we state the *path mapping* problem as: given a series of signal segments $S = \{s_i \mid 1 \leq i \leq m\}$ and a graph of floor plan $\mathcal{G} = (V, E)$, we look for a mapping $f : S \rightarrow V$ such that the total mapping cost $\sum_i \Delta(s_i, f(s_i))$ is minimized, where $\Delta(\cdot, \cdot)$ denotes the cost function that evaluates the incompatibility between a signal segment and a vertex.

The objective in the path mapping problem is to minimize the total mapping cost $\sum_i \Delta(s_i, f(s_i))$. However, the misdetection of junctions may cause the segmentation inconsistent with the floor plan and hence degrades the overall accuracy. To address this, we consider mapping a concatenation of q $(q \geq 1)$ consecutive segments in the mapping process. Specifically, we attempt to find a matched vertex for the segment concatenation $s_{i-q+1:i}$ in the programming, where $s_{i-q+1:i}$ denotes the concatenation of segments $\langle s_{i-q+1},\ s_{i-q+2},\ \ldots,\ s_i \rangle$. Therefore, the optimal value function in DP is formulated as

$$J_i = \min_{q,j} \left\{ J_{i-q} + \Delta(s_{i-q+1:i}, v_j) \right\}, \tag{3}$$

where J_i is the optimal mapping cost after mapping the signal segment concatenation ending with s_i.

To effectively reduce the computational complexity in DP and improve mapping accuracy, we further discuss some constraints in the formulation. Note that the mapped vertices in two consecutive stages correspond to two successive signal segments (or concatenations of signal segments) in a user path. To make sure the path continuity, we require that $\tilde{v}$ and v_j are connected, i.e.,

$$(\tilde{v}, v_j) \in \boldsymbol{E}, \tag{4}$$

where $\tilde{v}$ is the vertex mapped in the previous stage.

UbiFin encodes turning direction as an indication to reduce the ambiguity of similar paths. Let $\angle(\cdot,\cdot)$ be the path angle between two neighboring regions, and $\theta(\cdot,\cdot)$ be the walking direction change between consecutive signal segments. $\angle(\cdot,\cdot)$ is usually obtained from the given floor plan based on basic geometry, while $\theta(\cdot,\cdot)$ can be estimated by integrating the horizontal angular velocities over a turning period. The direction constraint can be expressed as

$$|\angle(\tilde{v}, v_j) - \theta(\boldsymbol{s}_{i-q}, \boldsymbol{s}_{i-q+1})| < \Theta, \tag{5}$$

where Θ is the tolerance of angle difference. Considering the precision of the gyroscope and common indoor layouts, we empirically set $\Theta = 30^\circ$ to adapt to the majority of cases.

For efficiency concern, we limit the length of segment concatenation q by setting up an upper bound Q, i.e.,

$$1 \leq q \leq Q. \tag{6}$$

In UbiFin, we set $Q = 2$ to balance complexity and accuracy.

5.2 Mapping Cost Functions

The mapping cost function $\Delta(\cdot,\cdot)$ measures the dissimilarity between a signal segment (or a concatenation of signal segments) and a vertex. Since a single type of signal is vulnerable to signal bias and environmental changes such as the adjustment of APs' antennas and the magnetization of metals, we consider a multimodal model to evaluate mapping costs, including motion (in terms of displacement), RF (typically Wi-Fi RSS), and geomagnetic field. Specifically, we define the mapping cost function $\Delta(\boldsymbol{s}, v)$ as a linear combination of the costs from individual signals, i.e.,

$$\Delta(\boldsymbol{s}, v) = \lambda\Delta_D(\boldsymbol{s}, v) + \alpha\Delta_R(\boldsymbol{s}, v) + \beta\Delta_B(\boldsymbol{s}, v), \tag{7}$$

where $\boldsymbol{s}$ is a signal segment (or a concatenation of consecutive signal segments) and v is a vertex in the floor plan graph; Δ_D, Δ_R, and Δ_B measure the mapping costs on displacement, RF, and geomagnetic field, respectively; λ, α, and β are their corresponding weights. We further require $\lambda + \alpha + \beta = 1$ to normalize $\Delta(\boldsymbol{s}, v)$.

In the following, we introduce how to obtain the mapping costs of different modalities.

Displacement. $\Delta_D(\boldsymbol{s}, v)$ compares the estimated displacement according to the inertial measurements in the signal segment $\boldsymbol{s}$ with the physical dimensions of the region corresponding to the vertex v.

Let $\boldsymbol{D}$ be the estimated destination of the signal segment. We employ pedestrian dead reckoning (PDR) techniques to compute $\boldsymbol{D}$. That is, given the initial states (i.e., the initial position and walking direction), user locations are estimated by integrating their walking directions and speeds over time [23]. The initial position is the junction between the previous region and the current one v, while the initial heading can be obtained from the geometric relationship between the previous and

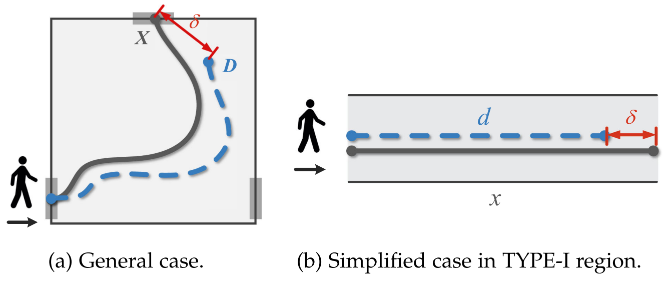


Fig. 7. Illustrations of the displacement difference in two types of regions. The solid gray lines show the ground-truth paths. The dotted blue lines are the estimated trajectories.

current regions in the floor plan. When moving in the region, user's walking direction is determined according to the gyroscope measurements, and walking speed can be obtained through a pedometer [50] or supervised learning methods [51]. On the other hand, the actual destination should be one of the exit junctions through which users can leave the region (and enter another). As the exit is unknown beforehand, we treat the one with the minimum distance to $\boldsymbol{D}$ as the expected destination (see Fig. 7a as an illustration). Let $\boldsymbol{X}_k$ be the coordinate of the kth exiting junction in v. The displacement error is represented by

$$\delta = \min_k \|\boldsymbol{D} - \boldsymbol{X}_k\|_2, \tag{8}$$

where $\|\cdot\|_2$ is the euclidean norm of a vector. Note that the influence of accumulated error commonly seen in PDR is rather limited in our cases since UbiFin performs PDR only in a small area [27], [52].

Particularly, in a TYPE-I region where users have only one degree of freedom to move, the computation of displacement error can be further reduced to walking distance error (see Fig. 7b). It helps mitigate the error introduced by walking direction estimation. In such a case, the displacement difference in a TYPE-I region can be represented as

$$\delta = |d - x|, \tag{9}$$

where $|\cdot|$ denotes the absolute value, d is the estimated walking distance, and x is the physical length of the region.

Finally, the displacement cost can be normalized by

$$\Delta_D(\boldsymbol{s}, v) = \min\left(\frac{\delta}{d}, 1\right). \tag{10}$$

RF. $\Delta_R(\boldsymbol{s}, v)$ reflects the dissimilarity between RF series in $\boldsymbol{s}$ and the region of vertex v. We compare the RF signals along the path segment with those existing fingerprints in the region.

Recall that we have obtained the PDR trajectory in the region when computing $\Delta_D(\boldsymbol{s}, v)$. Given the starting and ending points (i.e., the entrance and exit), we further calibrate the trajectory by rotating and scaling the trajectory to fit the region. For each RSS vector $\boldsymbol{r}_k$ in the path, we compare it with its nearest fingerprint $\boldsymbol{r}_{\text{fpr}}(k)$ (with respect to the calibrated trajectory) if $\boldsymbol{r}_{\text{fpr}}(k)$ has been generated and their distance is within 5m. We introduce a binary indicator function $\phi(\boldsymbol{r}_u, \boldsymbol{r}_v)$ to determine whether two RSS vectors $\boldsymbol{r}_u$ and $\boldsymbol{r}_v$ match, i.e.,

$$\phi(\boldsymbol{r}_u, \boldsymbol{r}_v) = \begin{cases} 0, & \text{if } \|\boldsymbol{r}_u - \boldsymbol{r}_v\|_2 / N_{\mathrm{AP}} \le \xi_R, \\ 1, & \text{otherwise}, \end{cases} \tag{11}$$

where $\|\cdot\|_2$ is the euclidean norm of the vector, N_{AP} is the number of APs appeared in both $\boldsymbol{r}_u$ and $\boldsymbol{r}_v$, and ξ_R is a threshold standing for the maximum difference between two RF signals at the same location [14]. We use the empirical setting of $\xi_R = 3\,\mathrm{dB}$ in UbiFin.

Let Φ be the index set of the RSS vectors that have valid neighboring fingerprints. The normalized RF cost function is thus defined as

$$\Delta_R(\boldsymbol{s}, v) = \frac{1}{|\Phi|} \sum_{k \in \Phi} \phi(\boldsymbol{r}_k, \boldsymbol{r}_{\mathrm{fpr}}(k)), \tag{12}$$

where $|\Phi|$ is the cardinality of Φ.

Geomagnetic Field. $\Delta_B(\boldsymbol{s}, v)$ measures the dissimilarity between the signal segment $\boldsymbol{s}$ and the vertex v in terms of geomagnetic field. A similar method to RF cost can be applied to computing the magnetic cost. Specifically, the indicator function of magnetic field is defined as

$$\psi(\boldsymbol{b}_u, \boldsymbol{b}_v) = \begin{cases} 0, & \text{if } \|\boldsymbol{b}_u - \boldsymbol{b}_v\|_2 / 2 \le \xi_B, \\ 1, & \text{otherwise}, \end{cases} \tag{13}$$

where $\|\cdot\|_2$ is the euclidean norm of the vector, ξ_B is the threshold controlling the maximum difference between two matched magnetic signals ($\xi_B = 3\mu\mathrm{T}$ in the paper) [45]. The cost of magnetic field $\Delta_B(\boldsymbol{s}, v)$ becomes

$$\Delta_B(\boldsymbol{s}, v) = \frac{1}{|\Psi|} \sum_{k \in \Psi} \psi(\boldsymbol{b}_k, \boldsymbol{b}_{\mathrm{fpr}}(k)), \tag{14}$$

where $\boldsymbol{b}_{\mathrm{fpr}}(k)$ is the nearest magnetic fingerprint to $\boldsymbol{b}_k$, Ψ is the index set of the samples whose neighboring fingerprints are available within 1m, and $|\Psi|$ is the number of elements in Ψ.

Algorithm 2. User Path Mapping

```
   input: A segmented signal series $S = \{s_i \mid 1 \le i \le m\}$ and a
          floor plan graph $\mathcal{G} = (V, E)$
   output: Path mapping $M$
   /* Compute the minimum mapping cost. */
 1 $J_0 \leftarrow 0$;
 2 for $1 \le i \le m$ do
 3     for $1 \le q \le Q$ and $i - q \ge 0$ do
 4         $v^{(i,q)} \leftarrow \arg\min_j \{J_{i-q} + \Delta(s_{i-q+1:i}, v_j)\}$, subject to the
           constraints (4) and (5);
 5         $J_{i,q} \leftarrow J_{i-q} + \Delta(s_{i-q+1:i}, v^{(i,q)})$;
 6     end
 7     $q^{(i)} \leftarrow \arg\min_q J_{i,q}$;
 8     $v^{(i)} \leftarrow v^{(i,q^{(i)})}$;
 9     $J_i \leftarrow J_{i,q^{(i)}}$;
10 end
   /* Recover the path mapping. */
11 $M \leftarrow \emptyset$;
12 $\gamma \leftarrow m$;
13 while $\gamma \ge 1$ do
14     Add $\langle s_{\gamma - q^{(\gamma)}+1:\gamma}, v^{(\gamma)} \rangle$ to $M$;
15     $\gamma \leftarrow \gamma - q^{(\gamma)}$;
16 end
17 return $M$;
```

5.3 DP Implementation

Given the above, we can implement the efficient path mapping algorithm using DP techniques. Algorithm 2 illustrates the major procedures.

The algorithm first conducts a forward pass to compute the optimal mapping cost (from line 2 to line 10). A vector J is used to store the optimal values in different stages, where J_i represents the total minimum cost till the ith stage. In particular, we impose an initial value $J_0 = 0$ for bootstrapping. Based on the recurrence relationship in Equation (3), the optimal costs can be determined stage by stage. Meanwhile, we cache the optimal choices of the segment concatenation length and the matched vertex in each stage, denoted by $q^{(i)}$ and $v^{(i)}$ for the ith stage, respectively. Formally

$$q^{(i)}, v^{(i)} = \operatorname*{arg\,min}_{q, v_j} \left\{ J_{i-q} + \Delta(\boldsymbol{s}_{i-q+1:i}, v_j) \right\}, \tag{15}$$

subject to the constraints (4), (5), and (6). $q^{(i)}$ and $v^{(i)}$ act as the backward pointers that lead us to the previous optimal states efficiently. Note that our DP can easily adapt to the case where a user manually inputs opportunistic positions in the path by setting the corresponding mapping costs to 0.

Once obtaining the optimal mapping cost, UbiFin employs a backtracking process to get the path mapping (from line 11 to line 16). Starting from the last mapping (i.e., the one from $\boldsymbol{s}_{m-q^{(m)}+1:m}$ to $v^{(m)}$, where m is the number of segments), UbiFin follows the cached $q^{(i)}$ and $v^{(i)}$ to find the preceding mappings iteratively. More generally, the optimal mapping is represented as

$$\boldsymbol{M} = \{ \langle \boldsymbol{s}_{\gamma_k - q^{(\gamma_k)}+1:\gamma_k}, v^{(\gamma_k)} \rangle \}, \tag{16}$$

where γ_k is the index of DP stage in the kth backtracking iteration with $\gamma_1 = m$ and $\gamma_k = \gamma_{k-1} - q^{(\gamma_{k-1})}$ when $k > 1$.

6 Fingerprint Construction

In this section, we discuss the construction of fingerprints based on the mapped paths. Section 6.1 presents a clustering-based approach to filter out misplaced signals at each RP. In Section 6.2, we introduce the comprehensive algorithm for constructing RF and magnetic fingerprint databases. Finally, we discuss the adaptation to environmental changes in Section 6.3.

6.1 Filtering Out Misplaced Signals

An inevitable challenge in constructing fingerprints is that the path mapping results may contain inconsistency and errors due to misplaced paths and signal fluctuation. To mitigate such influence, we propose a clustering-based approach to filter out outlier signals and aggregate the correct signals.

Given the resulting mapping of a crowdsourced signal series S, we assign the signal samples (either RF or magnetic field) to their nearest RPs. At each RP, a set of signals are assigned after multiple paths pass through it. Intuitively, signals at the same RP are similar to each other if they are correctly assigned, and, conversely, misplaced signals are sparsely distributed and become outliers. We hence adopt the density-based clustering method DBSCAN [48] on the assigned signal set to distinguish between the correctly mapped signals and outliers. We regard the cluster with the most members as the correct one and use its centroid to represent the fingerprint.

6.2 Fingerprint Database Construction

Algorithm 3. Fingerprint Database Construction

```
input: A batch of signal series $\mathcal{S} = \{S_1, S_2, \ldots\}$, a floor plan
       graph $\mathcal{G}$, and previously generated RF and magnetic
       fingerprint databases $\mathcal{D}_R$ and $\mathcal{D}_B$ (if existed)
output: RF fingerprint database $\mathcal{D}_R$ and magnetic fingerprint
        database $\mathcal{D}_B$
1  for each $S_i \in \mathcal{S}$ do
2    Segment $S_i$ by junction detection (Section 4);
3    Initialize $S_i$.attempts $= 0$;
4  end
5  Initialize $\mathcal{Q} \leftarrow \emptyset$;
6  Sort $\mathcal{S}$ according to the estimated traveling distances in
   descending order and then insert them to $\mathcal{Q}$;
7  while $\mathcal{Q} \neq \emptyset$ do
8    $S^* \leftarrow$ PopFront($\mathcal{Q}$);
9    if $S^*$.attempts $\geq \Omega$ then
10     continue                                    // Discard
11   end
12   Map $S^*$ to $\mathcal{G}$ and obtain the mapping $M$ with cost $J$
     (Section 5);
13   if $J/|M| \leq \kappa$ then                     // Accept
14     Assign signals in $S^*$ to corresponding RPs, and generate
       fingerprints (Section 6.1);
15   else                                          // Reject
16     PushBack($\mathcal{Q}, S^*$);
17     $S^*$.attempts $\leftarrow S^*$.attempts $+ 1$;
18 end
19 Generate remaining fingerprints using GPR;
20 return $\mathcal{D}_R$, $\mathcal{D}_B$;
```

Algorithm 3 summarizes the overall workflow of constructing fingerprint databases given a batch of crowdsourced signal series $\mathcal{S} = \{S_1, S_2, \ldots\}$. The algorithm first detects junctions in each path and segments them accordingly (line 2). To better manage the processing order of multiple user paths, we introduce a queue $\mathcal{Q}$ that contains the signal series ready to be mapped. We sort signal series by their estimated walking distances and put them into $\mathcal{Q}$ (line 6). This is because longer paths usually contain more topological information and thus have better mapping accuracy when few ambient fingerprints are available.

For each path in $\mathcal{Q}$, we apply the path mapping module to determine its route on the map (line 12). We use the average mapping cost ($J/|M|$) to evaluate whether the mapping fits the floor plan and previously generated fingerprints (line 13). If the cost is lower than the predefined threshold κ, we accept the mapping (line 14). We assign signals in the path to corresponding RPs and filter out misplaced ones. Otherwise, the mapping is not considered trustworthy. We postpone processing the signal series and put it back into $\mathcal{Q}$ (line 16). As the process goes on, more fingerprints are available in the databases. We thus have more knowledge of ambient signals in the site and are able to map previously rejected paths with more confidence. A further study on the impact of κ will be discussed in Section 7.2.

To ensure that the algorithm ends, we tag the number of attempts on each signal series. If a signal series has been rejected more than Ω times, we consider it to be incompatible with the floor plan and/or the existing fingerprints and thus discard it (line 10). In our prototype designing, we set $\Omega = 3$ empirically.

At the locations where few users visit, we do not have sufficient signals to generate fingerprints. To obtain full fingerprint databases, we apply Gaussian process regression (GPR) to interpolate the signal values according to their surrounding fingerprints [36], [47].

6.3 Discussion on Environmental Change Adaptation

Fingerprints usually vary over time due to environmental changes, for example, modification of network configurations, movement of network devices, renovation, etc. [36]. UbiFin is capable of adapting to the changes, and keeps the databases up-to-date. The reasons are twofold. On the one hand, UbiFin is robust against signal noise and partial signal changes. Recall that UbiFin leverages multimodal signals to map user paths to the floor plan. Local signal changes do not affect much on the mapping accuracy. On the other hand, UbiFin is designed in an incremental mode. As time goes by, more and more users walk through the region and contribute the latest signals to the changed fingerprints. In an affected fingerprint, updated signals gradually gather to form a new cluster. It eventually becomes the dominant cluster and replaces the outdated fingerprint.

7 Experimental Evaluation

UbiFin is simple to implement. In this section, we evaluate UbiFin through extensive experiments. We introduce the implementation details and experimental settings in Section 7.1, followed by illustrative results in terms of crowdsourcing localization accuracy and fingerprinting accuracy in Section 7.2. In Section 7.3, we further discuss the positioning performance resulted from various indoor positioning applications using the generated fingerprints.

7.1 Experimental Settings

Implementation & Devices. Our implementation of UbiFin consists of a client app built on smartphones and a processing server running on a PC. The client app is developed on Android (API level 21). It gathers sensor readings at the highest sampling frequency, including accelerations, angular velocities, Wi-Fi RSS values, magnetic fields, and the like. We have tested it on various models of devices and different versions of operating systems. Specifications of some selected experimental devices are listed in Table 2. The server is implemented on Python 3. It receives crowdsourced data from clients and constructs fingerprint databases of both RF and geomagnetic fields.

Experimental Environments. We conduct experiments in two large-scale indoor environments of different characteristics. The first venue is a floor in the academic building of our campus, named *campus* (Fig. 8). The venue consists mainly of corridors of different widths, with a total area of

TABLE 2
Specifications of Selected Devices in the Experiments

Model	Operating System	Wi-Fi Sampling Interval (Frequency)	Magnetic Sampling Interval (Frequency)
Google Nexus 5	Android 5.1	910.3ms (1.10Hz)	16.8ms (59.58Hz)
Sony Xperia Z2	Android 6.1	5178.3ms (0.19Hz)	20.1ms (49.65Hz)
Redmi 4a	Android 7.1	10023.8ms (0.10Hz)	5.0ms (198.58Hz)
Samsung Galaxy S8	Android 8.0	3050.4ms (0.33Hz)	20.0ms (50.02Hz)
Vivo Y50	Android 10	3445.5ms (0.29Hz)	9.9ms (100.84Hz)
Redmi Note 9	Android 10	18238.4ms (0.05Hz)	10.0ms (100.00Hz)

around 14000 m^2. The second site is a two-story shopping mall building, short for *mall* (Fig. 9). It contains lots of wide corridors, a large atrium, and two escalators through which users can reach the other floor. The complex venue covers an area of approximately 46000 m^2.

Data Collection. The experimental data are collected by multiple volunteers over several days. Participants carry their smartphones and roam freely in the venues. They can walk at their own pace without pre-planning. In total, there are 95 paths of signals collected by 8 participants on the campus, and 223 paths are collected by 15 users in the mall. Note that we have no restriction on how users carry their phones (e.g., holding, putting in pockets, etc.). In the experiments, we adopt the preprocessing techniques described in Section 3.3 to filter appropriate user activities and phone attitudes for fingerprint database construction.

To record the true locations of user paths for evaluation, participants are required to label locations on the app when passing landmarks (e.g., junctions, counters, etc.). Besides, we conduct additional site surveys to obtain ground-truth fingerprints. Surveyors stand still at each RP for 10 seconds and record Wi-Fi RSS values or geomagnetic fields. Both the labeled locations and the surveyed signals are used only for performance evaluation.

Evaluation Metrics. In the experiments, we adopt the following performance metrics:

- *Crowdsourcing localization accuracy*: We use *crowdsourcing localization error* to evaluate the accuracy of estimated routes. Specifically, we take samples every 5 seconds along each path. For a sampled point, the error is defined as the euclidean distance between the estimated location and its true position.
- *Fingerprinting accuracy*: We use the metrics of *RF fingerprint error* and *magnetic fingerprint error* to evaluate the correctness of generated fingerprints, respectively. For each RP, RF fingerprint error is defined as the mean absolute error (MAE) of the generated RF fingerprint against the site-surveyed one, and magnetic fingerprint error is the MAE between the generated magnetic fingerprint and its actual fingerprint.

Comparison Schemes. We compare UbiFin with the following state-of-the-art schemes:

- *GROPING* [17]: The work leverages both crowdsourced magnetic fields and opportunistic location labels to construct fingerprint databases. Different user paths are associated according to their magnetic correspondence. To make a fair comparison, we integrate floor plans into the system so that user paths can be associated with the floor plan directly.
- *UnLoc* [27]: The work employs clustering techniques to discover landmarks with unique signal patterns (e.g., Wi-Fi, magnetic field, acceleration, etc.). The estimated locations of landmarks are then used to calibrate walking directions and stride lengths of different users in PDR trajectories.
- *LiFS* [14]: This work considers a shared latent structure between the fingerprint space and the physical space (represented by a stress-free floor plan). It adopts multidimensional scaling (MDS) to align the spaces and thus fingerprints Wi-Fi signals. Note that LiFS detects doors as anchors for space alignment, while they are not available in our scenarios. To adapt to our experiments, we manually label all the signals at junctions, which have similar density and distribution to the door case.

Default Parameters. Unless otherwise stated, we use *campus* as the default testing venue. In both sites, we adopt the supervised learning based speed estimation [51] in the PDR

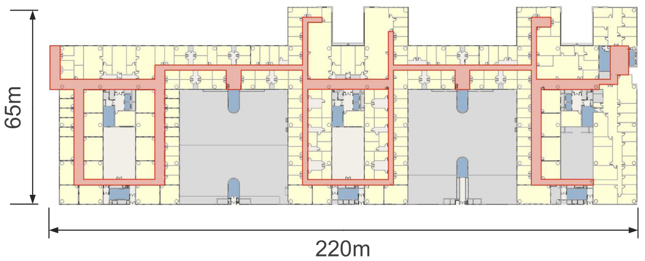


Fig. 8. The floor plan of one floor in our campus building.

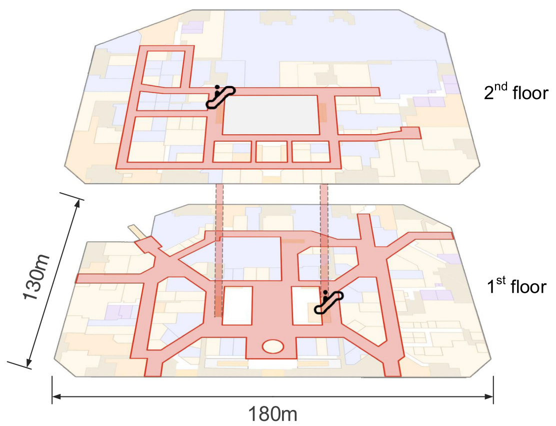


Fig. 9. The floor plan of a two-story shopping mall.

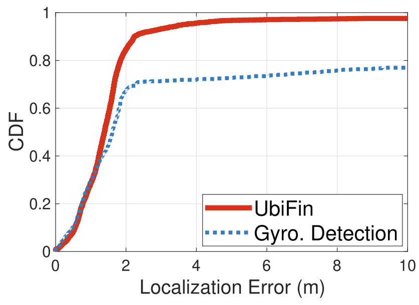


Fig. 10. CDF of crowdsourcing localization errors with different junction detection approaches.

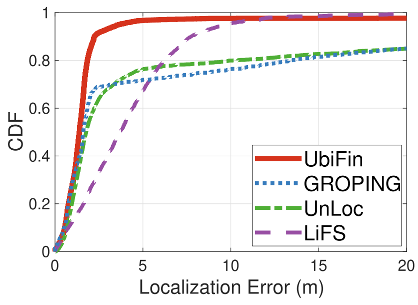


Fig. 11. CDF of crowdsourcing localization errors (campus).

implementation. The default system parameters are $\alpha = 0.4$, $\beta = 0.3$, and $\kappa = 0.1$.

7.2 Illustrative Results

Performance of Junction Detection. Junction detection is an important component in UbiFin as it segments user paths to correlate them with the floor plan graph. In Fig. 10, we compare crowdsourcing localization errors under different junction detection approaches, i.e., the proposed method (labeled as *UbiFin*) and the classic turning detection using gyroscope (labeled as *Gyro. Detection*). We can observe a significant improvement in UbiFin compared with the traditional way. This is because UbiFin leverages user behaviors in multiple paths to cross-verify the recognized junctions and hence achieves accurate and robust detection. By contrast, the traditional method is prone to the influence of unexpected user activities and sensor noise.

Performance of User Path Mapping. We show in Fig. 11 crowdsourcing localization errors among different schemes at the campus site. Benefit from the multimodal nature, UbiFin gains the lowest mean error (3.08 m) and outperforms the others by a large margin. By contrast, UnLoc fails to identify accurate locations of landmarks while GROPING suffers from inaccurate segmentation and error accumulation. We also observe a long tail in the CDF caused by occasional incorrect mapping decisions. However, it has limited influence on fingerprinting because the majority of paths can be mapped accurately (the 95th percentile is 3.60 m), and the proposed clustering mechanism can further recognize the misplaced signals as outliers (Section 6.1).

Fig. 12 compares the crowdsourcing localization performance in the mall. Likewise, UbiFin achieves a satisfactory accuracy with a mean error of 4.97 m and significantly outperforms the competitors. Compared with the case on the campus, the localization accuracy in the mall is degraded to some extent due to noisier signals and more complex environments. Among them, LiFS shows the most serious drop ($\sim$70%) since the ambiguity of RF signals disturbs the stitching accuracy among different paths. However, the performance of UbiFin remains satisfactory and exceeds the others substantially, validating its robustness against heterogeneous environments.

Accuracy of Fingerprint Databases. We evaluate the accuracy of the generated RF and magnetic fingerprint databases on the campus. Fig. 13 shows the CDFs of RF fingerprint errors under different schemes. UbiFin achieves the minimum RF fingerprint error (3.48 dB) and greatly outperforms the others (cutting the error by 24 to 48 percent). Fig. 14 plots the CDF comparison of magnetic fingerprint errors. Note that we have not involved LiFS here because LiFS does not guarantee the continuity of signals and hence does not apply to fingerprinting magnetic fields. The results show that UbiFin achieves the lowest fingerprint error (1.78 μT), which is at least 40% less than GROPING and UnLoc. The remarkable fingerprinting performance comes mainly from the accurate path mapping (also illustrated in Fig. 11) and the effective misplaced signal filtering.

We also evaluate the fingerprint accuracy in the large shopping mall. Fig. 15 shows the CDFs of RF fingerprint errors. We can see the lowest RF fingerprint error (4.45 dB) and the shortest tail achieved by UbiFin. Similar to the previous results on the campus (Fig. 13), the accuracy of UbiFin exceeds the others by a large margin. Fig. 16 presents the CDF plots of magnetic fingerprint errors. Unsurprisingly, the magnetic fingerprint error of UbiFin is the lowest (3.44 μT) among all the evaluated schemes, outperforming GROPING by 38.8% and UnLoc by 52.5%. The stable results also prove the applicability of UbiFin in diverse scenarios.

Impact of System Parameters. We study the impact of signal weights α, β, and λ in the mapping cost function (Equation (7)). The weights reflect our trust towards different signal modalities, i.e., RF (controlled by α), geomagnetic field (controlled by β), and displacement (controlled by $\lambda = 1 - \alpha - \beta$). Fig. 17 shows the distribution of mean crowdsourcing localization errors over different weight settings. On the one hand, we can observe a high localization error when ambient signals are missing (i.e., both α and β are set to 0). Without any ambient signal, UbiFin maps crowdsourced paths based on mobility information and floor plans only. This results in severe route ambiguity, i.e., a path may be mapped to multiple possible routes in the floor plan, leading to large localization errors in the experiments. On the other hand, the error bars on the diagonal (i.e., $\alpha + \beta = 1$) demonstrate that lack of displacement information leads to high localization errors (with only ambient signals such as RF and geomagnetic fields without motion

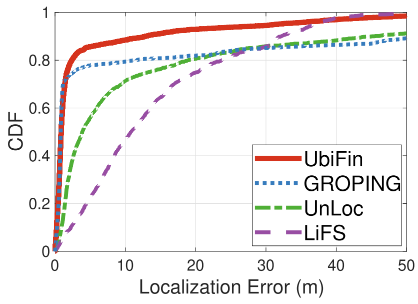


Fig. 12. CDF of crowdsourcing localization errors (mall).

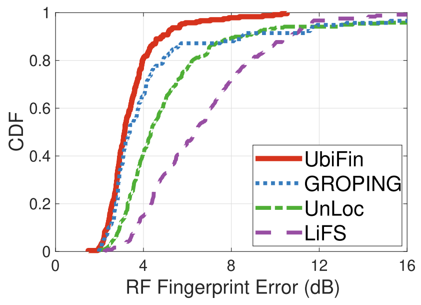


Fig. 13. CDFs of RF fingerprint errors (campus).

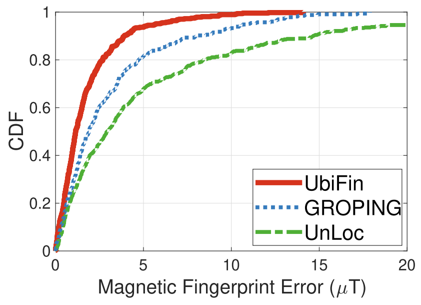


Fig. 14. CDFs of magnetic fingerprint errors (campus).

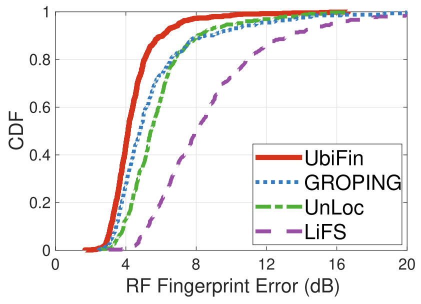


Fig. 15. CDFs of RF fingerprint errors (mall).

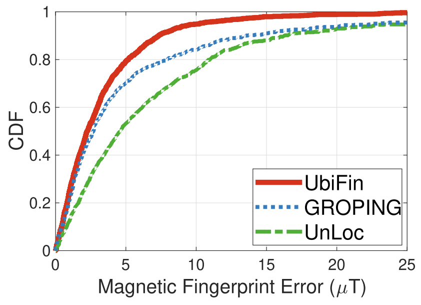


Fig. 16. CDFs of magnetic fingerprint errors (mall).

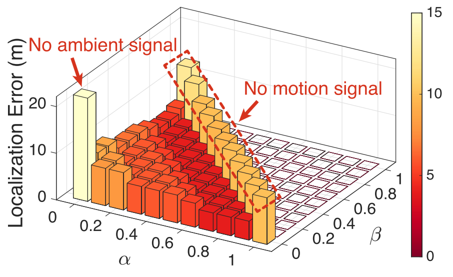


Fig. 17. Impact of signal weights α and β on crowdsourcing localization errors.

information). In this case, errors usually occur when mapping paths in the area without adequate previous fingerprints. The above shows that either signal modality alone is not sufficient to accurately determine path trajectories. By contrast, with both ambient (RF and magnetic field) and motion signals, UbiFin achieves low localization errors, as shown in the middle region where $\alpha \neq 0$, $\beta \neq 0$, and $\lambda \neq 0$. In practice, we set $\alpha = 0.4$, $\beta = 0.3$, and thus $\lambda = 0.3$, to balance the importance of signal modalities and hence achieve robust performance.

Fig. 18 demonstrates the impact of acceptance threshold κ (Algorithm 3) on crowdsourcing localization errors and mapping rejection rates (the percentage of mapping that cannot be accepted for fingerprinting). The threshold controls whether a mapping result is trustworthy or not. A small κ tightens up the requirement of mapping confidence, and vice versa. We observe a large localization error when κ is very small (say, 0.05). This is because the strict threshold rejects a large number of paths and results in insufficient ambient information for later mapping. On the other hand, a loose value (i.e., large κ) fails to recognize the incompatible mappings and hence cannot yield satisfactory accuracy either. With a proper threshold $\kappa = 0.1$, UbiFin achieves extraordinary mapping accuracy in both sites where over 90 percent of paths are correctly mapped.

Impact of Heterogeneous Devices. Fig. 19 compares the performance of junction detection among different devices (Table 2). In the experiment, we consider a detected junction to be correct if the time difference between the detection time and the actual encountering time is less than 3 s. We use *precision* to reflect whether detected junctions are correct and *recall* to describe whether all junctions are detected. *F-score* is the harmonic mean of precision and recall, which gives an overall evaluation. Benefit from the robust magnetic matching (Section 4.1) and clustering-based junction recognition (Section 4.2), UbiFin achieves satisfactory performance on all the devices (most precisions are greater than 0.95 and all recalls are greater than 0.9).

We study in Fig. 20 the crowdsourcing localization performance on heterogeneous devices. The CDFs show that all the devices achieve similar localization performance. Over 90% of the paths collected by each device are accurately located with low errors (less than 5 m). Surprisingly, we notice that two smartphones with Android 10 (i.e., Vivo Y50 and Redmi Note 9) perform slightly worse than the others. This is possibly because of the scanning restriction (e.g., reduced Wi-Fi scanning frequency) on the Android platform of a higher version [53]. Despite the low RF sampling rate on certain devices,

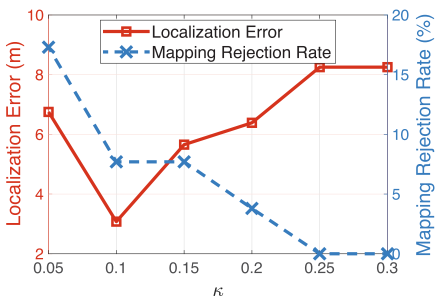


Fig. 18. Impact of acceptance threshold κ on crowdsourcing localization errors and mapping rejection rates.

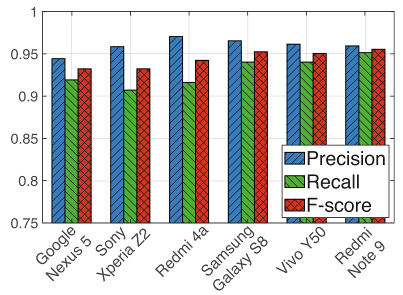


Fig. 19. Junction detection performance among heterogeneous devices.

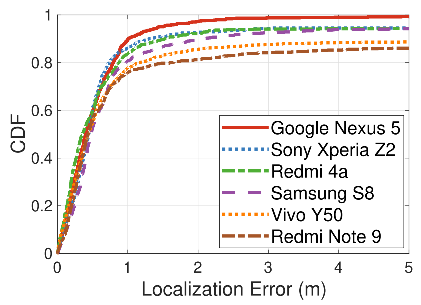


Fig. 20. CDFs of crowdsourcing localization errors among heterogeneous devices.

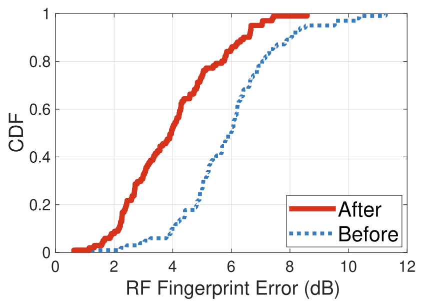


Fig. 21. CDFs of RF fingerprint errors before/after adaptation to environmental changes.

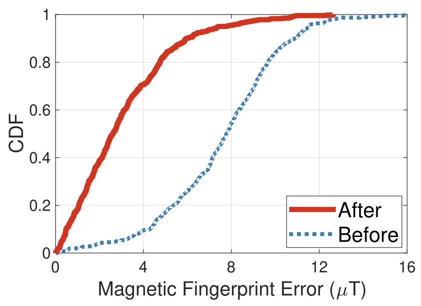


Fig. 22. CDFs of magnetic fingerprint errors before/after adaptation to environmental changes.

UbiFin takes advantage of multimodal signals and hence shows robustness against device heterogeneity.

Performance of Environmental Change Adaptation. UbiFin is capable of adapting to environmental changes and updating fingerprints accordingly. To validate this, we design an extra experiment to evaluate system performance under environmental changes. Specifically, we treat the same campus site after 12 months as a changed environment. There is

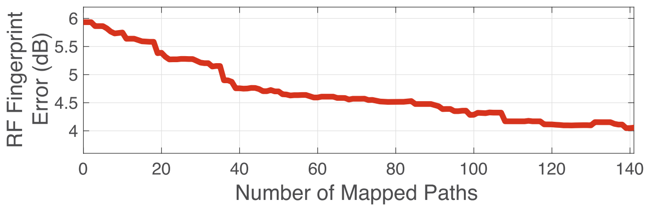


Fig. 23. RF fingerprint errors over the number of mapped paths in environmental change adaptation.

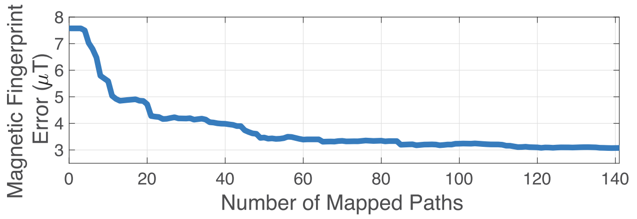


Fig. 24. Magnetic fingerprint errors over the number of mapped paths in environmental change adaptation.

no obvious change on the floor plan. However, due to the partial renovation and the upgrade of network facilities, both Wi-Fi and magnetic fields in much of the region have been changed. We gather another set of signal series via crowdsourcing (157 paths collected over several days) and apply UbiFin on top of the outdated databases (constructed by the previous crowdsourcing).

Figs. 21 and 22 demonstrate the RF and magnetic fingerprint errors before and after performing UbiFin, respectively. Due to signal changes in the environment, the previous fingerprint databases deviate from the current true fingerprints by a large margin (5.93 dB in RF and 7.57 μT in magnetic fields on average). After applying UbiFin, the fingerprint databases are gradually updated and the fingerprint errors are greatly reduced (cutting the RF error by 32.3% and the magnetic error by 59.5%).

To have a close look at the process of the environmental adaptation, we further present the RF and magnetic fingerprint errors over the number of mapped paths in Figs. 23 and 24, respectively. As time goes by, more paths are correctly mapped and outdated fingerprints are replaced with the latest ones, and thus both the errors steadily decrease. We also notice that the RF errors converge at a slower pace. This is mainly because of the much lower sampling frequency of Wi-Fi and higher dimensionality in RSS measurements. By contrast, since magnetic fields can be measured densely along paths, more magnetic RPs are thus updated and the fingerprint errors drop rapidly in the initial 60 paths.

7.3 Indoor Positioning Applications Based on the Generated Fingerprints

To verify the effectiveness and compatibility of our generated fingerprint databases, we further investigate the positioning accuracy under the following mature indoor positioning applications:

- *RADAR* [8]: This is a pioneering Wi-Fi fingerprint positioning algorithm that utilizes the K-nearest neighbor (KNN) algorithm. Cosine similarity is applied to

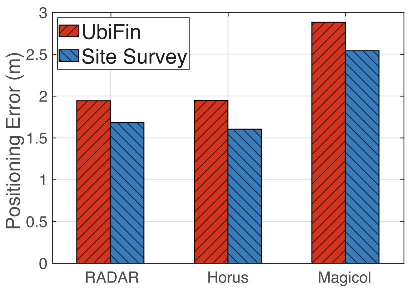


Fig. 25. Positioning errors with different fingerprint databases (campus).

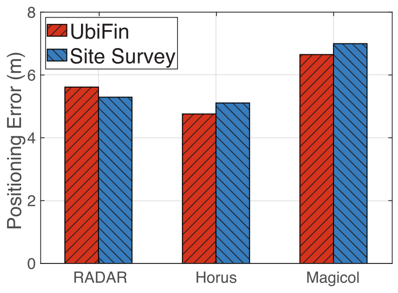


Fig. 26. Positioning errors with different fingerprint databases (mall).

compute the similarity between any two RSS vectors. We set $K = 3$ in our implementation.

- *Horus* [9]: This is a probabilistic Wi-Fi fingerprint positioning scheme that assumes RSS values follow Gaussian distributions. The algorithm aims to find the fingerprints which have the maximum likelihood of the observed RSS vectors. A weighted K-nearest neighbor (WKNN) algorithm is then used for precise location estimation ($K = 3$ in our implementation).
- *Magicol* [4]: The algorithm exploits dynamic time warping (DTW) on sequences of magnetic fields to compare the observed signals with the given fingerprint database. It also adopts a particle filter to further improve positioning accuracy.

We evaluate the positioning errors under the fingerprint databases generated by UbiFin and by an ordinary site survey, respectively. Fig. 25 shows the errors on the campus. We can see that UbiFin performs well in both Wi-Fi and magnetic field positioning applications. It yields competitive positioning accuracy compared with the time-consuming site survey among all the schemes. The error difference between UbiFin and the site survey is less than 0.4m. We also plot Fig. 26 the performance comparison in the shopping mall. Though the positioning errors in the mall are generally larger than those on the campus due to noisy signals and complex environments, UbiFin achieves similar results to the campus case where the positioning accuracy is close to (and sometimes even better than) the site survey. This fully validates the effectiveness of UbiFin in practical positioning applications.

8 Conclusion

In this paper, we propose *UbiFin*, a novel and robust fingerprinting scheme based on the implicit multimodal crowdsourcing of RF, magnetic, and motion (acceleration and angular velocity) signals. By correlating and analyzing these signals among different user paths, UbiFin mitigates signal bias and path mapping error to construct fingerprints of both RF and geomagnetic fields simultaneously. As far as we know, this is the first piece of work considering multimodal correlation for joint RF and geomagnetic fingerprinting.

UbiFin first employs a junction detection algorithm based on the motion and geomagnetic signals. It partitions user paths into multiple segments indicated by the junction points in the floor plan. Using a dynamic programming formulation, UbiFin then efficiently maps the segments as user trajectories in the floor plan. It subsequently detects and filters out those misplaced fingerprints to construct highly accurate fingerprint databases of both RF and geomagnetic field. We have implemented UbiFin and conducted extensive experiments on our campus and a multi-story shopping mall. The experimental studies show that UbiFin outperforms other state-of-the-art crowdsourcing schemes in terms of accuracy, robustness, and scalability (cutting fingerprint error by 40 percent in general).

Acknowledgments

This work was supported by the Hong Kong General Research Fund under Grant 16200120.

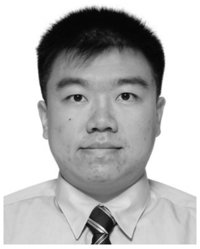
References

[1] MarketsandMarkets, "Indoor location market by component, deployment mode, application, vertical, and region - global forecast to 2022," MarketsandMarkets, Chicago, IL, USA, Tech. Rep., Oct. 2017.
[2] S. He and S.-H. G. Chan, "Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons," *IEEE Commun. Surv. Tut.*, vol. 18, no. 1, pp. 466–490, Jan.–Mar. 2016.
[3] S. He and K. G. Shin, "Geomagnetism for smartphone-based indoor localization: Challenges, advances, and comparisons," *ACM Comput. Surv.*, vol. 50, no. 6, pp. 1–37, 2017.
[4] Y. Shu, C. Bo, G. Shen, C. Zhao, L. Li, and F. Zhao, "Magicol: Indoor localization using pervasive magnetic field and opportunistic WiFi sensing," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 7, pp. 1443–1457, Jul. 2015.
[5] H. Wu, Z. Mo, J. Tan, S. He, and S.-H. G. Chan, "Efficient indoor localization based on Geomagnetism," *ACM Trans. Sensor Netw.*, vol. 15, no. 4, pp. 1–25, 2019.
[6] X. Guo, W. Shao, F. Zhao, Q. Wang, D. Li, and H. Luo, "WiMag: Multimode fusion localization system based on Magnetic/WiFi/PDR," in *Proc. Int. Conf. Indoor Positioning Indoor Navigation*, 2016, pp. 1–8.
[7] W. Zhang, R. Sengupta, J. Fodero, and X. Li, "DeepPositioning: Intelligent fusion of pervasive magnetic field and WiFi fingerprinting for smartphone indoor localization via deep learning," in *Proc. 6th IEEE Int. Conf. Mach. Learn. Appl.*, 2017, pp. 7–13.
[8] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *Proc. 19th Annu. Joint Conf. IEEE Comp. Commun. Soc.*, 2000, pp. 775–784.
[9] M. Youssef and A. Agrawala, "The Horus WLAN location determination system," in *Proc. 3rd Int. Conf. Mobile Syst., Appl., Serv.*, 2005, pp. 205–218.
[10] S. He and S.-H. G. Chan, "TileJunction: Mitigating signal noise for fingerprint-based indoor localization," *IEEE Trans. Mobile Comput.*, vol. 15, no. 6, pp. 1554–1568, Jun. 2016.
[11] H. Xie, T. Gu, X. Tao, H. Ye, and J. Lu, "A reliability-augmented particle filter for magnetic fingerprinting based indoor localization on smartphone," *IEEE Trans. Mobile Comput.*, vol. 15, no. 8, pp. 1877–1892, Aug. 2016.
[12] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: Zero-effort crowdsourcing for indoor localization," in *Proc. 18th Annu. Int. Conf. Mobile Comput. Netw.*, 2012, pp. 293–304.
[13] S. Yang, P. Dessai, M. Verma, and M. Gerla, "FreeLoc: Calibration-free crowdsourced indoor localization," in *Proc. 32nd Annu. IEEE Int. Conf. Comput. Commun.*, 2013, pp. 2481–2489.

[14] C. Wu, Z. Yang, and Y. Liu, “Smartphones Based crowdsourcing for indoor localization,” *IEEE Trans. Mobile Comput.*, vol. 14, no. 2, pp. 444–457, Feb. 2015.
[15] B. Jang and H. Kim, “Indoor positioning technologies without offline fingerprinting map: A survey,” *IEEE Commun. Surv. Tut.*, vol. 21, no. 1, pp. 508–525, Jan.–Mar. 2019.
[16] B. Lashkari, J. Rezazadeh, R. Farahbakhsh, and K. Sandrasegaran, “Crowdsourcing and sensing for indoor localization in IoT: A review,” *IEEE Sens. J.*, vol. 19, no. 7, pp. 2408–2434, Apr. 2019.
[17] C. Zhang, K. P. Subbu, J. Luo, and J. Wu, “GROPING: Geomagnetism and crowdsensing powered indoor navigation,” *IEEE Trans. Mobile Comput.*, vol. 14, no. 2, pp. 387–400, Feb. 2015.
[18] Z. Xiao, H. Wen, A. Markham, and N. Trigoni, “Lightweight map matching for indoor localisation using conditional random fields,” in *Proc. 13th Int. Symp. Inf. Process. Sensor Netw.*, 2014, pp. 131–142.
[19] P. Croak and Y. Kim, “Site Survey guidelines for WLAN deployment,” Feb. 2020. [Online] Available: https://www.cisco.com/c/en/us/support/docs/wireless/5500-series-wireless-controllers/116057-site-survey-guidelines-wlan-00.html
[20] B. Zhan, D. N. Monekosso, P. Remagnino, S. A. Velastin, and L.-Q. Xu, “Crowd analysis: A survey,” *Mach. Vis.appl.*, vol. 19, no. 5, pp. 345–357, Oct. 2008.
[21] D. H. Kim, K. Han, and D. Estrin, “Employing user feedback for semantic location services,” in *Proce. 13th Int. Conf. Ubiquitous Comput.*, 2011, pp. 217–226.
[22] J.-G. Park *et al.* “Growing an organic indoor location system,” in *Proc. 8th Int. Conf. Mobile Syst., Appl., Serv..*, 2010, pp. 271–284.
[23] B. Huang, G. Qi, X. Yang, L. Zhao, and H. Zou, “Exploiting cyclic features of walking for pedestrian dead reckoning with unconstrained smartphones,” in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2016, pp. 374–385.
[24] C. Chen, X. Lu, A. Markham, and N. Trigoni, “IONet: Learning to cure the curse of drift in inertial odometry,” in *Proc. 32nd AAAI Conf. Artif. Intell. AAAI*, Feb. 2018.
[25] S. Herath, H. Yan, and Y. Furukawa, “RoNIN: robust neural inertial navigation in the wild: Benchmark, evaluations, new methods,” in *Proc. IEEE Int. Conf. Robot. Automat.*, May 2020, pp. 3146–3152.
[26] C. Wu, Z. Yang, and C. Xiao, “Automatic Radio Map Adaptationradio map adaptation for Indoor Localization Using Smartphones,” *IEEE Transactions on Mobile Computing*, vol. 17, no. 3, pp. 517–528, Mar. 2018.
[27] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, “No need to war-drive need to war-drive: Unsupervised indoor localization,” in *Proc. 10th. Mobile Syst., Appl., Serv.*, Jun. 2012, pp. 197–210.
[28] Q. Wang, H. Luo, F. Zhao, and W. Shao, “An indoor selfindoor self-localization algorithm using the calibration of the online magnetic fingerprints and indoor landmarks,” in *Proc. Indoor Positioning Indoor Navigation*, Oct. 2016, pp. 1–8.
[29] C. Wu, Z. Yang, Y. Liu, and W. Xi, “WILL: Wireless indoor localization without site survey,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 4, pp. 839–848, Apr. 2013.
[30] S. J. Pan, J. T. Kwok, Q. Yang, and J. J. Pan, “Adaptive localization in a dynamic WiFi environment through multi-view learning,” in *Proc. 22nd AAAI Conf. Artif. Intell.*, 2007, pp. 1108–1113.
[31] Z. Xiao, H. Wen, A. Markham, and N. Trigoni, “Indoor tracking using undirected graphical models,” *IEEE Trans. Mobile Comput.*, vol. 14, no. 11, pp. 2286–2301, Nov. 2015.
[32] G. Younes, D. Asmar, E. Shammas, and J. Zelek, “Keyframe-based monocular SLAM: Design, survey, and future directions,” *Robot. Auton. Syst.*, vol. 98, pp. 67–88, 2017.
[33] S. Wang, H. Wen, R. Clark, and N. Trigoni, “Keyframe based large-scale indoor localisation using geomagnetic field and motion pattern,” in *Proc. IEEE/RSJ Int. Conf. Intell. RobotsSyst.*, Oct. 2016, pp. 1910–1917.
[34] Q. Liang and M. Liu, “An automatic site survey approach for indoor localization using a smartphone,” *IEEE Trans. Automat. Sci. Eng.*, vol. 17, no. 1, pp. 191–206, Jan. 2020.
[35] S. He, B. Ji, and S.-H. G. Chan, “Chameleon: Survey-free updating of a fingerprint database for indoor localization,” *IEEE Pervasive Comput.*, vol. 15, no. 04, pp. 66–75, Oct.–Dec. 2016.
[36] S. He, W. Lin, and S.-H. G. Chan, “Indoor localization and automatic fingerprint update with altered AP signals,” *IEEE Trans. Mobile Comput.*, vol. 16, no. 7, pp. 1897–1910, Jul. 2017.
[37] S. Nikitaki, G. Tsagkatakis, and P. Tsakalides, “Efficient multichannel signal strength based localization via matrix completion and Bayesian sparse learning,” *IEEE Trans. Mobile Comput.*, vol. 14, no. 11, pp. 2244–2256, Nov. 2015.
[38] C. Li, Q. Xu, Z. Gong, and R. Zheng, “TuRF: Fast data collection for fingerprint-based indoor localization,” in *Proc. Int. Conf. Indoor Positioning Indoor Navigation*, 2017, pp. 1–8.
[39] Z. Sun, Y. Chen, J. Qi, and J. Liu, “Adaptive localization through transfer learning in indoor Wi-Fi environment,” in *Proc. Seventh Int. Conf. Mach. Learn. Appl.*, 2008, pp. 331–336.
[40] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, “Activity recognition from accelerometer data,” in *Proc. 17th Conf. Innovative Appl. Artif. Intell.*, 2005, pp. 1541–1546.
[41] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, “Activity recognition using cell phone accelerometers,” in *Proc. 4th Int. Workshop Knowl. Discov. Sensor Data*, 2011, pp. 74–82.
[42] P. Zhou, M. Li, and G. Shen, “Use it free: Instantly knowing your phone attitude,” in *Proc. 20th Annu. Int. Conf. Mobile Comput. Netw.*, 2014, pp. 605–616.
[43] S. Butterworth, “On the theory of filter amplifiers,” *Exp. Wirel. Wirel. Eng.*, vol. 7, pp. 536–541, 1930.
[44] T. Smith and M. Waterman, “Identification of common molecular subsequences,” *J. Mol. Biol.*, vol. 147, no. 1, pp. 195–197, 1981.
[45] H. Wu, S. He, and S.-H. G. Chan, “Efficient sequence matching and path construction for geomagnetic indoor localization,” in *Proc. Int. Conf. Embedded Wirel. Syst. Netw.*, 2017, pp. 156–167.
[46] T. C. O’haver, *A Pragmatic Introduction to Signal Processing: With Applications in Scientific Measurement*, 2nd ed. Scotts Valley, CA, USA: CreateSpace, 2016.
[47] C. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.
[48] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *Proc. 2nd Int. Conf. Knowl. Discov. Data Mining*, 1996, pp. 226–231.
[49] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA, USA: MIT Press, 2009.
[50] A. Brajdic and R. Harle, “Walk detection and step counting on unconstrained smartphones,” in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2013, pp. 225–234.
[51] J.-g. Park, A. Patel, D. Curtis, S. Teller, and J. Ledlie, “Online pose classification and walking speed estimation using handheld devices,” in *Proc. ACM Conf. Ubiquitous Comput..* 2012, pp. 113–122.
[52] I. Constandache, R. R. Choudhury, and I. Rhee, “Towards mobile phone localization without war-driving,” in *Proc. 29th Conf. Comput. Commun.*, 2010, pp. 1–9.
[53] Google, “Wi-Fi scanning overview,” 2021. [Online]. Available: https://developer.android.com/guide/topics/connectivity/wifi-scan

Jiajie Tan received the Bachelor of Engineering degree from Zhejiang University, Hangzhou, Zhejiang, China, in 2012. He is currently working toward the PhD degree at the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong. His research interests include wireless location sensing, Internet of Things, and mobile computing.

Hang Wu received the bachelor of computer science (ACM honored class) degree from Shanghai Jiao Tong University, in 2015 and the master of philosophy degree in computer science and engineering from the Hong Kong University of Science and Technology, in 2017. His research interests include indoor localization, mobile computing, machine learning, and data mining.



Ka-Ho Chow received the BEng and MPhil degrees in computer science from the Hong Kong University of Science and Technology. He is currently working toward the PhD degree with the School of Computer Science, Georgia Institute of Technology. He is a member of Distributed Data Intensive Systems Laboratory. His research interests include robust machine learning, cybersecurity, ML for systems, and mobile computing.

S.-H. Gary Chan (Senior Member, IEEE) received the BSE degree (highest honor) from Princeton University, Princeton, NJ, with certificates in applied and computational mathematics, engineering physics, and engineering and management systems, and the MSE and PhD degrees with a minor in business administration from Stanford University, Stanford, CA, all in electrical engineering. He was a visiting professor and a researcher with Microsoft Research, Princeton University, Stanford University, and the University of California, Davis. He was the director of Entrepreneurship Center, the director of Sino Software Research Institute, the co-director of Risk Management and Business Intelligence program, and the director of Computer Engineering Program, Hong Kong University of Science and Technology (HKUST). He is currently a professor at the Department of Computer Science and Engineering, HKUST, Hong Kong, and an affiliate professor with Innovation, Policy, and Entrepreneurship Thrust Area, HKUST, Guangzhou. His research interests include smart sensing and IoT, cloud and fog or edge computing, indoor localization and mobile computing, video or location or user or data analytics, and IT entrepreneurship. He was an associate editor for the *IEEE Transactions on Multimedia*, the vice-chair of Peer-to-Peer Networking and Communications Technical Sub-Committee of the IEEE Comsoc Emerging Technologies Committee, and the guest editor of the *ACM Transactions on Multimedia Computing,* the *Communications and Applications*, the *IEEE Transactions on Multimedia*, the *IEEE Signal Processing Magazine*, and the *IEEE Communication Magazine* . He is currently the chair of the Committee on Entrepreneurship Education Program, HKUST, the board director of Hong Kong Logistics and Supply Chain MultiTech R&D Center, a steering committee member, the TPC chair of the IEEE Consumer Communications and Networking Conference, and the area chair of the multimedia symposium of the IEEE Globecom and the IEEE ICC. He has co-founded and transferred his research results to several startups. Due to their innovations and commercial impacts, his startups and research projects have received local and international accolades. He was the recipient of the Hong Kong Chief Executive's Commendation for Community Service for "outstanding contribution to the fight against COVID-19" in 2020, the Google Mobile 2014 Award, the Silver Award of Boeing Research and Technology, the Charles Ira Young Memorial Tablet and Medal, and the POEM Newport Award of Excellence at Princeton. He was a William and Leila fellow with Stanford University, a member of honor societies Tau Beta Pi, Sigma Xi and Phi Beta Kappa, and a chartered fellow of The Chartered Institute of Logistics and Transport.