Smartphone-based Localization for Blind Navigation in Building-Scale Indoor Environments☆

Masayuki Murata[a,*], Dragan Ahmetovic[b], Daisuke Sato[a], Hironobu Takagi[a], Kris M. Kitani[b], Chieko Asakawa[b,c]

[a]*IBM Research - Tokyo, 19-21 Nihonbashi Hakozaki-cho, Chuo-ku, Tokyo, 103-8510, Japan*
[b]*Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA*
[c]*IBM Research, 1101 Kitchawan Rd, Yorktown Heights, NY 10598, USA*

Abstract

Continuous, accurate, and real-time smartphone-based localization is a promising technology for supporting independent mobility of people with visual impairments. However, despite extensive research on indoor localization techniques, localization technologies are still not ready for deployment in large and complex environments such as shopping malls and hospitals, where navigation assistance is needed most. We identify six key challenges for accurate smartphone localization related to the large-scale nature of the navigation environments and the user's mobility. To address these challenges, we present a series of techniques that enhance a probabilistic localization algorithm. The algorithm utilizes mobile device inertial sensors and Received Signal Strength (RSS) from Bluetooth Low Energy (BLE) beacons. We evaluate the proposed system in a 21,000 m^2 shopping mall that includes three multi-story buildings and a large open underground passageway. Experiments conducted in this environment demonstrate the effectiveness of the proposed technologies to improve localization accuracy. Field experiments with visually impaired participants confirm the practical performance of the proposed system in realistic use cases.

Keywords: indoor localization, smartphones, Bluetooth Low Energy, mobile sensors, accessibility

1. Introduction

Accurate localization is fundamental to enable smartphone-based turn-by-turn navigation assistance for people with visual impairments in complex large-scale real-world indoor scenarios. While some of the existing localization techniques advertise accurate positioning capabilities, they are often evaluated in controlled scenarios only. Thus, it is unclear whether current localization algorithms for mobile navigation can be successful for blind navigation assistance in building-scale real-world scenarios such as complex public facilities and commercial buildings.

When we consider localization systems for turn-by-turn navigation in real-world scenarios for people with visual impairments, the requirements are quite daunting. We identify six key challenges for accurate localization related to the large-scale nature of the navigation environments and the user's mobility, which are often overlooked in the research literature.

(1) Accurate and Continuous Localization. It is critical to achieve both *accurate* and *continuous* localization when giving turn-by-turn instructions to people with visual impairments. Preliminary tests have shown that a localization accuracy of about two meters is desirable to provide timely turn-by-turn guidance, especially at decision points such as corridor intersections or entrances. A higher localization error could lead a person with visual impairment through the wrong door or cause collisions with the environment. While probabilistic localization algorithms are designed to deal with a certain level of noise, many approaches can fail catastrophically when the fidelity of the current state estimate

☆A preliminary version of this paper appeared in the Proceedings of the IEEE International Conference on Pervasive Computing (PerCom) 2018 (Murata et al., 2018).

*Corresponding author. Tel.: +81-3-3808-5247
Email addresses: muratams@jp.ibm.com (Masayuki Murata), dragan.ahmetovic@unito.it (Dragan Ahmetovic), dsato@jp.ibm.com (Daisuke Sato), takagih@jp.ibm.com (Hironobu Takagi), kkitani@cs.cmu.edu (Kris M. Kitani), chiekoa@cs.cmu.edu (Chieko Asakawa)

degrades. Remedies to such failures (*e.g.*, modifications to the state sampling process [1, 2]) have been proposed, but such approaches can cause the location estimates to jump around discontinuously. There is no tolerance for such instability when guiding people with visual impairments.

(2) Scaling to Multi-Story Buildings. Previous methods for localizing users with a smartphone have primarily focused on 2D floor plans [3]. However, most buildings in metropolitan areas are multi-story buildings. In particular, in public facilities such as shopping centers or subway stations, people constantly transition from floor to floor and from building to building. It is therefore crucial to localize users across floors and during floor transitions. To the best of our knowledge, work addressing accurate localization over multi-story buildings has been limited.

(3) Signal Bias Adaptation at Scale. Different mobile devices observe different receiver signal strength (RSS) values from the same signal transmitter at the same location due to differences in the reception sensitivity of the underlying radio hardware. Previous works have addressed this issue [4–6] by estimating a signal strength offset value. However, in real-world applications, the number and strength of observable transmitters, *e.g.,* beacons, changes dynamically over time. There is limited prior work addressing this challenge, *i.e.,* varying RSS values over multiple devices in dynamic situations.

(4) Scaling to Large Numbers of Measurements. As the size of deployment grows to building-scale proportions, the computational costs of regression algorithms for accurately mapping between locations and RSS observations grow prohibitively expensive to run in real-time with current smartphone resources. Efficient methods are needed to accelerate the computation of localization for building-scale environments.

(5) Detecting Motion for Different Walking Profiles. Localization accuracy can be improved by factoring in the user's movement information, *e.g.*, step detection from the inertial sensor data. However, different pedestrians may have more or less accentuated steps, and therefore it is possible that for some users steps are frequently missed. This situation can cause noticeable errors in localization.

(6) Observed Signal Delay. Based on our observations, the RSS values detected and reported by mobile device operating systems tend to be delayed, *i.e.*, they describe a stale state of the system, referring to moments in the past. These delayed data affect the accuracy of localization, especially when the user is moving.

Contributions. To deal with the aforementioned challenges within a unified framework, we use a probabilistic localization algorithm as our foundation and enhance it with a series of novel innovations. This paper is an extension of our prior work [7], in which we addressed the first four challenges by the following technical innovations:

1. *Localization Integrity Monitoring* introduces an internal state machine that enables the system to be softly reinitialized during failures.
2. *Floor Transition Control* relies on changes in barometric pressure and RSS values to regularize localization during elevator or escalator use.
3. *Adaptive Signal Calibration* uses a Kalman filter to incrementally estimate signal offsets.
4. *Fast Likelihood Computation* is achieved through the use of a set of locally trained truncated regression models.

In this extension, to further improve the localization system, we consider the remaining two issues, which are related to user mobility and temporal delay in measurements, with two additional extensions:

5. *Probabilistic Motion State Detection* determines a user's motion state on the basis of a probability to alleviate the impact of errors in motion state detection.
6. *Time Sensitive Observation Modeling* adjusts the estimated location and the RSS observation model considering the temporal delay in the RSS measurements.

We implement the localization system and perform a thorough evaluation with data collected in a large and complex indoor environment composed of three multi-story buildings and a broad underground pedestrian walkway. To quantitatively validate the reliability of our method, we collected ground truth localization data using a Light Detection and Ranging (LIDAR) sensor. On the basis of this data, we evaluate the effect of our proposed enhancement modules. We also evaluate the localization accuracy in experiments with visually impaired participants in the same testing environment. The localization system is integrated into a turn-by-turn navigation application on iOS devices, and we evaluate localization error while the participants are traversing the environment with the aid of the navigation app.

2. Related Work

2.1. Indoor Localization

Indoor localization has been extensively studied for the past two decades [8–12]. Among various indoor localization techniques, localization based on the RSS of wireless signals such as WiFi or Bluetooth is one of the most popular approaches [8, 13, 1, 14, 15] due to its use of off-the-shelf mobile devices, potential for high accuracy, and relatively low infrastructure cost.

Aside from RSS-based methods, various other localization techniques based on RFID [16], UWB radios [17], ultrasound [18], *etc.* have been developed. Most of these approaches require specialized hardware for either the infrastructure or the user, and sometimes both. Image-based localization methods (*e.g.*, [6]) are promising, but they are not robust enough in scenes having few visual features and frequent appearance changes. Recently, Channel State Information (CSI) [19, 20] and Fine Timing Measurement (FTM) [21] have been investigated to achieve a higher accuracy (∼1m) localization with WiFi than RSS-based methods. Unfortunately, these approaches are still not available on commodity smartphones and WiFi access points, and therefore cannot yet be applied for our envisioned use case.

RSS-based localization methods can be divided into two categories: fingerprint-based or model-based. Fingerprint-based localization is the prevalent solution to achieve and guarantee better accuracy [8, 22, 13, 1, 6]. It is usually conducted in two phases: an offline training (site-survey) phase to collect RSS data labeled with correct locations and an online operating phase to estimate the location of a user's mobile device. Model-based methods assume a propagation model of radio waves to estimate RSS at various locations. For this purpose, the log-distance path loss model is widely used [23, 24]. Although model-based methods require much less training data than fingerprint-based methods, they are also less accurate, particularly in non-line-of-sight conditions. To reduce site-survey effort, localization methods relying on fingerprint database construction by unsupervised learning or crowdsourcing have recently been proposed [24–26]. However, they are less accurate than fingerprint-based methods and insufficient for applications with high accuracy requirements.

WiFi fingerprinting has been widely studied for indoor localization using the RSS of WiFi signals, as access points (APs) are ubiquitous in most environments [8, 13, 1, 12]. However, WiFi coverage is not tuned to provide accurate localization, and WiFi AP positioning depends on environment wiring constraints and connectivity requirements.

As an alternative, Bluetooth-based localization has gained prominence following the introduction of the Bluetooth Low Energy (BLE) protocol standard and the commercialization of off-the-shelf BLE beacons [27, 15, 28]. Compared to WiFi APs, BLE beacons are small, low-cost ($5–20 per device), and have low power consumption. Thus, they can be battery-powered and placed with fewer constraints than WiFi APs. This way the signal coverage can be controlled in order to achieve uniform and more accurate localization. Also, the RSS from BLE beacons are accessible on most commodity smartphone operating systems (iOS and Android), while WiFi scanning is currently prohibited on iOS devices.

To further improve the localization accuracy achieved by RSS fingerprint-based methods, recent approaches have performed fusion between RSS-based localization and the user motion model [11]. The fusion algorithms are used to integrate RSS fingerprint-based methods and the movement of a user that is obtained by applying pedestrian dead reckoning (PDR) methods to data from sensors embedded on a mobile device [29, 30]. This way, it is possible to produce more accurate localization from noisy observations. The particle filter is one of the most successful sensor fusion algorithms for localization [13, 1], due to its excellent extendability and ease of implementation. This approach is also suitable for indoor navigation because it is based on state space modeling and can estimate unobservable variables, *e.g.*, user's walking direction, which are important for navigation. Because of these advantages, we adopt this approach as our base localization framework. Details of our implementation of the particle filter algorithm are presented in Section 3. Although the proposed technical innovations are designed on top of our specific implementation, since we followed a typical particle filter-based scheme as our foundation, the proposed innovations can also be used to enhance similar existing systems. For the same reason, other improvements on the base localization scheme, *i.e.*, improvements in the motion model, the observation model, and the particle filter algorithm, can be further integrated with our localization system to achieve better average performance.

2.2. User Mobility and Temporal Delay in Observations.

In our previous work [7], we proposed four technical innovations that enhance a particle filter-based localization algorithm to enable the scaling of smartphone-based localization to building-scale environments. This article proposes two additional technical innovations that address problems in user motion detection and delay in RSS observations to improve the localization accuracy. To contextualize these additional technical innovations, we review the most relevant related works below.

Pedestrian dead reckoning estimates a user's displacement on the basis of stride length estimation. For stride length estimation, offline calibration methods [30, 1] or online calibration methods utilizing a map [31] or the fusion with wireless fingerprints [32] have been proposed. While the algorithms used in these methods are different, most methods commonly rely on step detection. Step detection can work reliably if a user moves with a steady gait. However, the displacement estimation can become unreliable when the user's motion profile changes, *e.g.*, a user moves with lower acceleration changes. A recent study [33] found significantly larger step counting error for blind walkers than for sighted walkers through analysis of a data set collected from the two groups. Considering such studies is important to achieve practical indoor navigation for people with visual impairments. To take into account the unique characteristics of blind walkers, we utilize a non-deterministic motion state detector to alleviate the effects of errors in motion state detection.

Delay in RSS observations on a smartphone is an existing but less targeted problem. We are aware of one related work [34] that addresses a problem in which outdated RSS values remain in WiFi fingerprints for a while due to the long duration of WiFi scans. While the characteristics of BLE fingerprints and WiFi fingerprints are different [15], handling delay in reported BLE RSS values on a smartphone is also essential to improve the localization performance, especially when a user is moving. Our work deals with the issue by considering an average delay in RSS reporting and the dependence of reported RSS values on the past locations.

2.3. Navigation Assistance for People with Visual Impairments

Prior research investigated various indoor navigation approaches to support people with visual impairments [35, 36]. Among these, smartphone-based turn-by-turn navigation systems using BLE beacons provide accurate navigation assistance, feature an off-the-shelf infrastructure, and are easy to deploy [37–42]. Turn-by-turn navigation is a guidance method that orients a user towards a destination through sequential vocal messages. These messages can be announcements of distances, action instructions, and additional contextual information (*e.g.*, nearby points of interest).

NavCog [39, 40] is a turn-by-turn smartphone-based navigation assistant for people with visual impairments that uses the RSS fingerprints of BLE beacons to localize the user. In its first iteration, *NavCog* used the k-nearest neighbor algorithm to perform the localization on a simplified space representation consisting of one-dimensional edges. Follow-up work [41] improved the approach by fusing the PDR and RSS-based localization. While the simplified space representation used in these seminal approaches was designed to ease the deployment workload, it had limited pose estimation and localization capabilities, which makes it unsuitable for navigation in complex environments. To enable navigation assistance for people with visual impairments in large-scale environments such as multi-story shopping malls, *NavCog3* [42, 7] is designed to rely on a location tracking method on two-dimensional floor maps that exploits not only the user's position but also the heading direction. Sato et al. [42] focused on the user interface and usability of the navigation assistant, while in this work, we present the technical details of the localization method and innovations that enable high-level indoor localization accuracy suitable for navigating people with visual impairments in building-scale environments.

3. Proposed Localization Model

We now describe the underlying probabilistic framework for state estimation using the particle filter. We first give a sketch of the base particle filter model for localizing a user with smartphone sensors and a BLE beacon network. The basic concepts introduced here will help to contextualize the key technical innovations introduced later in Section 4.

3.1. Particle Filter

The particle filter is an efficient algorithm for continuously estimating a user's location [2, 13, 1]. Let t be an index of time, $\boldsymbol{z}_t$ be the user's state (*e.g.*, 2D location), $\boldsymbol{r}_t$ be the sensor measurements (*e.g.*, RSS values from multiple transmitters), $\boldsymbol{u}_t$ be the control input (*e.g.*, the user's movement obtained from inertial measurements), and m be the map. The particle filter approximates the posterior of a user's state conditioned on all previous measurements $\boldsymbol{r}_{1:t}$ and inputs $\boldsymbol{u}_{1:t}$ by a finite set of samples (particles) $\{\boldsymbol{z}_{t|t}^{l}\}_{l=1}^{L}$, where l and L correspond to the index and the total number of particles.

Three steps are iteratively performed in the algorithm:

1. Predict each particle's state by using the motion model $p\left(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}, \boldsymbol{u}_t, m\right)$, which describes the relationship between the user's previous state $\boldsymbol{z}_{t-1}$ and current state $\boldsymbol{z}_t$ given the control input $\boldsymbol{u}_t$.
2. Compute each particle's importance weight by using the observation model $p\left(\boldsymbol{r}_t|\boldsymbol{z}_t\right)$, which describes the likelihood of the observed signal strength measurements $\boldsymbol{r}_t$ at user's state $\boldsymbol{z}_t$.
3. Resample particles with replacement from a predicted particle set according to the importance weights.

3.2. Motion Model

The motion model $p\left(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}, \boldsymbol{u}_t, m\right)$ predicts the user's location using sensory information (*e.g.,* sensors on the user's smartphone). We model this distribution using Gaussian random variables where the predicted location of the user (x_t, y_t), under the control input $\boldsymbol{u}_t = (s_t, \theta_t)^T$, can be written as:

$$x_t = x_{t-1} + s_t v_t \cos\left(\theta_t + \theta_t^{\mathrm{o}}\right)\Delta t + \xi_{x,t}, \tag{1a}$$

$$y_t = y_{t-1} + s_t v_t \sin\left(\theta_t + \theta_t^{\mathrm{o}}\right)\Delta t + \xi_{y,t}. \tag{1b}$$

The motion state of the user s_t is an indicator function (*i.e.,* user is moving: $s_t = 1$ or stopped: $s_t = 0$), θ_t is the orientation of the smartphone, and θ_t^{o} is the offset of the orientation between the user and the smartphone. v_t is the velocity of the user and Δt is the time step between $t-1$ and t. The cosine and sine represent the direction of displacement of the user location for Δt on the x-y plane. The user's motion state s_t can be detected by thresholding the standard deviation of the magnitude of acceleration in a short time window [43]. The attitude of the smartphone can be obtained by integrating the smartphone's IMU sensor data (*i.e.*, accelerometer and gyroscope data). $\xi_{x,t}$ and $\xi_{y,t}$ correspond to perturbation noise to x_t and y_t with zero mean Gaussian random variables. The θ_t^{o} and v_t are estimated through particle filtering by incorporating them into a state vector as $\boldsymbol{z_t} = [x_t, y_t, v_t, \theta_t^{\mathrm{o}}]^\top$.

3.3. Observation Model

The observation model $p\left(\boldsymbol{r}_t|\boldsymbol{z}_t\right)$ describes the likelihood of the sensor measurements $\boldsymbol{r}_t$ given the user's state $\boldsymbol{z}_t$. Our method assumes that BLE beacons are installed densely in the environment, similar to [15]. The observation model is learned from training data, as a function that maps position to RSS. Formally, we are given a set of training samples $D = \{(\boldsymbol{x}_n, r_n)\}_{n=1}^{N}$, where $\boldsymbol{x}_n$ is the input (location) and r_n is the output (RSS), and N is the number of training samples. We apply kernel ridge regression [44] to predict RSS given a vector of location $\boldsymbol{x}_*$ as

$$\mu(\boldsymbol{x}_*) = m(\boldsymbol{x}_*) + k_*^T(K + \sigma_n^2 I)^{-1}(\boldsymbol{r} - m(\boldsymbol{X})), \tag{2}$$

where k_* is the vector of the kernel function between $\boldsymbol{x}_*$ and each training point, K is the matrix of the kernel function relating each pair of points, σ_n is a regularization parameter, and $m(\boldsymbol{x}_*)$ is an explicit prior on the mean function. The mean function $m(\boldsymbol{x}_*)$ is computed using the well-known log-distance path loss model [45]:

$$m_i(\boldsymbol{x}) = -10 n_i \log\left(d(\boldsymbol{x}, b_i)\right) + A_i, \tag{3}$$

where $d(\boldsymbol{x}, b_i)$ is the physical distance between position $\boldsymbol{x}$ and BLE beacon b_i, n_i is the decay exponent, and A_i is the path loss at the reference distance of 1 m. The variance of the output, σ_i^2, is separately estimated as a position-independent constant for each BLE beacon b_i. As a result, the RSS for the i-th beacon is modeled by

$$p(r_{i,t}|\boldsymbol{x}_t) = N(r_{i,t}; \mu_i(\boldsymbol{x}_t), \sigma_i^2). \tag{4}$$

The likelihood model, given all of the RSS values from multiple beacons, is obtained from the product of single beacon likelihood terms. We use only the top K RSS signals to accelerate computation time. Formally, let $\boldsymbol{r}_t = (r_{1,t}, \ldots, r_{K,t}, \ldots, r_{M,t})^\top$ be the values of an RSS vector in descending order. The observation model with a limited number of beacons can be written as

$$p(\boldsymbol{r}_t|\boldsymbol{x}_t) = \prod_{i=1}^{K} p(r_{i,t}|\boldsymbol{x}_t)^{\alpha}, \tag{5}$$

where M is the total number of observed beacons, K is the number of beacons used for localization, and α is a smoothing coefficient to prevent the likelihood model from overconfident estimates due to dependencies between $r_{i,t}$ [2].

3.4. Initial Pose Estimation

When the localization algorithm starts tracking, it must identify the initial pose, including the location and orientation. We use the Metropolis-Hastings algorithm [46] to draw samples from $p(\boldsymbol{x}_t|\boldsymbol{r}_t)$ (equivalent to sampling from $p(\boldsymbol{r}_t|\boldsymbol{x}_t)$ when $p(\boldsymbol{x}_t)$ is assumed to be uniform) to estimate the location. To compute the initial orientation, we use the smartphone magnetometer and GPS receiver, but a very large variance is placed on the distribution to account for uncertainty.

4. Proposed Technical Innovations

The base particle filtering framework described above is sufficient to localize a user in ideal situations. However, in real-world scenarios, the system can fail catastrophically and drain smartphone computing resources if key issues are not taken into consideration. This section introduces key technical innovations based on insights from real-world use cases that enable us to scale smartphone-based localization to very large (building-scale) environments.

4.1. Localization Integrity Monitoring

The Localization Integrity Monitoring (LIM) module observes whether the localization is working as expected and switches the behavior of the system in times of uncertainty. Specifically, we implement a state machine to monitor the integrity of localization to control how sensor inputs are used by the localization algorithm. Figure 1 shows a diagram of the state machine, which consists of four states: *unknown, locating, tracking,* and *unreliable*. The state starts from *unknown* and changes depending on the RSS vector input $\boldsymbol{r}_t$. After a first RSS vector input is given, the state changes to *locating*, in which the localization system begins initialization. The *locating* state is repeated until the uncertainty of the current location decreases below a certain level. Once initialized, the state transits to *tracking*. If the uncertainty remains high, the *locating* state returns to *unknown*. In the *tracking* state, the localization system tracks the user's state by means of the particle filter.

The important issue here is the reaction of the system when the *tracking* state changes to the *unreliable* state. Formally, let $X_{t|1:t}$ and $X_{t|t}$ be a set of particles approximating the belief distribution $p(\boldsymbol{x}_t|\boldsymbol{r}_{1:t}, \boldsymbol{u}_{1:t})$ and a set of particles drawn from $p(\boldsymbol{x}_t|\boldsymbol{r}_t)$, respectively. The set of particles $X_{t|1:t}$ is obtained as the filtered states by the particle filter and $X_{t|t}$ can be obtained by generating samples using the method in Section 3.4. Abnormality is measured as the ratio of the maximum likelihood given $X_{t|1:t}$ and the maximum likelihood given $X_{t|t}$, which is formally defined as

$$a(X_{t|1:t}, X_{t|t}, \boldsymbol{r}_t) = \frac{\max\limits_{x_t \in X_{t|1:t}} p(\boldsymbol{r}_t|\boldsymbol{x}_t)}{\max\limits_{x_t \in X_{t|t}} p(\boldsymbol{r}_t|\boldsymbol{x}_t)}. \tag{6}$$

The numerator and denominator take similar values when the device location is successfully tracked. Otherwise, the numerator is much smaller than the denominator because the region with high likelihood is not covered with the tracked particles. As a result, an abnormal situation can be detected by checking whether $a(X_{t|1:t}, X_{t|t}, \boldsymbol{r}_t)$ takes an exceedingly small value, *e.g.*, 0.01. When an abnormal situation is detected, the *locating* state changes to the *unreliable* state. The role of the *unreliable* state is to buffer the decision to revert the state to *unknown*. When an abnormal situation is subsequently detected during the *unreliable* state, the state moves to *unknown*. In this state, the localization system stops updating the unreliable belief distribution conditioned by past inputs and restarts localization from the initialization.

4.2. Floor Transition Control

The Floor Transition Control (FTC) module seamlessly bridges the estimated location of the user from a source floor to a target floor to reduce localization failure related to floor transitions. To apply the localization algorithm in multi-story environments, we need to introduce transitions between two-dimensional floors into the localization algorithm. We denote the floor on which a user is as f_t and augment the location vector $\boldsymbol{x}_t$ as $\boldsymbol{x}_t = (x_t, y_t, f_t)^\top$.

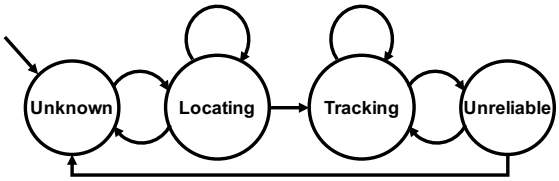


Figure 1: State machine in localization integrity monitoring.

We also define a subset of states called *floor transition areas* that include staircases, escalators, and elevators. When the motion model predicts the user's state, floor to floor transitions can only occur at a floor transition area. In cases where the user's location is estimated to be on an escalator, the motion model adds a constant velocity motion to the user's state to model the passive movement of the user carried by the escalator without taking any steps.

We propose a multi-modal observation model to allow seamless transitions between floors. The standard observation model defined above becomes unstable when transiting from floor to floor because the number of visible beacon signals can be very sparse in that situation (*e.g.*, due to elevator walls blocking BLE signal). To reduce the risk of increased error during floor transition, we use the smartphone barometer in addition to beacon RSS to actively update the user's location. Although barometer readings alone are known to be very noisy [47], they can be used in conjunction with RSS to detect floor changes. The barometer can be used to detect changes in height by thresholding the standard deviation of estimated height over a short period of time.

Formally, let variable c_t denote a detected floor change (*i.e.*, changing: $c_t = 1$ or not changing: $c_t = 0$) and A_T be floor transition areas including stairs, escalators and elevators. We introduce a conditional motion model and a modification to the observation model to take into account changes in floors. Specifically, we introduce $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, c_t)$ and $p(c_t|\boldsymbol{x}_t)$ into the particle filter. In the motion model, the location of a particle $\boldsymbol{x}_{t-1}^l$ is moved to the closest point in A_T with a probability $\max(0, p_{A_T} - \sum_{\boldsymbol{x}_t^l \in A_T} w(\boldsymbol{x}_t^l))$, where p_{A_T} is an acceptable lower limit for $p(\boldsymbol{x}_t \in A_T|c_t = 1)$. In the observation model, detected height changes are used to update the weights of the particles as

$$w_t(\boldsymbol{x}_t^l) \leftarrow \frac{w_t(\boldsymbol{x}_t^l)p(c_t = 1|\boldsymbol{x}_t^l)}{\sum_{k=1}^{L} w_t(\boldsymbol{x}_t^k)p(c_t = 1|\boldsymbol{x}_t^k)}, \tag{7}$$

where $p(c_t = 1|\boldsymbol{x}_t)$ is the likelihood of $\boldsymbol{x}_t$ given $c_t = 1$, which satisfies that $p(c_t = 1|\boldsymbol{x}_t \notin A_T) < p(c_t = 1|\boldsymbol{x}_t \in A_T)$.

In addition to obtaining reliable estimates of location at all times, predictive location estimates (before the user arrives at a location) are desired so that the navigation application can issue instructions as early as possible. Further enhancements to the floor transition module can be achieved by exploiting the observation model of BLE beacon RSS observations. In addition to the sampling of the floor variable in the motion model, the floor variable is selectively sampled by the observation model when a floor change is detected ($c_t = 1$). More specifically, the floor variable is drawn according to the likelihood of the observation where variables, except for the floor, are fixed:

$$f_t^l \sim p(\boldsymbol{r}_t|f_t^l, \boldsymbol{x}_t^{l,\backslash f}), \tag{8}$$

where $\boldsymbol{x}_t^{l,\backslash f}$ denotes the location of the l-th particle except for the floor variable f_t^l. This sampling scheme improves the response of the estimated location to actual floor transitions. In practice, this allows us to localize the user a few seconds before the actual floor arrival and gives us enough time to notify the user.

4.3. Adaptive Signal Calibration

The Adaptive Signal Calibration (ASC) module adjusts the device RSS offset with a time-series filtering algorithm modified for a truncated observation vector. A basic approach for adjusting signal offset can be implemented by minimizing the distance in RSS space between two devices with respect to an RSS offset [5, 6]. We pose a similar optimization problem to calibrate for offsets in signal reading but apply the updates in an online manner using a lightweight time-series filter based on the Kalman filter [2]. This approach also enables automatic adaptation to the uncertainty in the offset estimation due to dynamical changes in the number of observed beacons.

Formally, we denote RSS $r_{i,t}$ of device i at time step t. To differentiate the RSS observed during training time and test time, we use the notation $r_{i,t}^A$ for the training time signal and $r_{i,t}^B$ as the test time signal. We denote the current predicted mean of RSS $\mu_i(\boldsymbol{x}_t)$ in shorthand notation as $\mu_{i,t}$. We assume that at test time, there will be an offset

introduced to the RSS due to a different device being used (*e.g.*, a different iPhone) or some global changes to the signal environment (*e.g.*, weakening of the beacon signal strength). Formally, we assume

$$r^B_{i,t} = r^A_{i,t} + r^{\mathrm{o}}_t, \tag{9}$$

where r^{o}_t is the signal strength offset at time t.

Since the observation model $p(r^A_{i,t}|\boldsymbol{x}_t) = N(r^A_{i,t}; \mu_{i,t}, \sigma^2_{i,t})$ is Gaussian, we can denote the test time observation model as

$$p(\boldsymbol{r}^{B(1:K)}_t|\boldsymbol{x}_t, r^{\mathrm{o}}_t) \ = \prod_{i=1}^{K} N(r^B_{i,t}; \mu_{i,t} + r^{\mathrm{o}}_t, \sigma^2_{i,t}), \tag{10}$$

where the mean of the Gaussian has been modified by the latent offset value r^{o}_t.

The number of visible beacons can vary across time steps, so computing the full likelihood can be expensive when many beacons are visible. We can truncate the number of terms in the likelihood product by setting a small value for K and taking a product of the K largest RSS signals. Taking this truncation into account, we can properly estimate the true distribution from this truncated distribution in the following way. Extracting the largest subset $\boldsymbol{r}^{B(1:K)}_t$ from the original RSS vector $\boldsymbol{r}^{B(1:M)}_t$, we can see that the values of $\boldsymbol{r}^{B(1:K)}_t$ are generated from a truncated distribution with a lower limit. The largest value in the discarded value, $r^B_{K+1,t}$, can be such a lower limit. Let $N_{tr}(r^B_{i,t}; \mu_{i,t} + r^{\mathrm{o}}_t, \sigma^2_{i,t}, r^B_{K+1,t} < r^B_{i,t})$ be the truncated probability distribution for $r^B_{i,t}$ with a lower limit value $r^B_{K+1,t}$. By approximating this truncated distribution to a normal distribution with the same mean and variance, the observation model for $\boldsymbol{r}^{B(1:K)}_t$ incorporated with the effect of truncation $q(\boldsymbol{r}^{B(1:K)}_t|\boldsymbol{x}_t, r^{\mathrm{o}}_t)$ can be approximated by

$$q(\boldsymbol{r}^{B(1:K)}_t|\boldsymbol{x}_t, r^{\mathrm{o}}_t) \ \approx \prod_{i=1}^{K} N(r^B_{i,t}; \mu'_{i,t} + r^{\mathrm{o}}_t, \sigma'^2_{i,t}), \tag{11}$$

where $\mu'_{i,t} + r^{\mathrm{o}}_t$ is the mean and $\sigma'^2_{i,t}$ is the variance for the truncated distribution of $r^B_{i,t}$.

The above moments can be obtained as follows [48]:

$$\mu'_{i,t} = \mu_{i,t} + \sigma_{i,t} \frac{\phi(a_{i,t})}{1 - \Phi(a_{i,t})}, \tag{12a}$$

$$\sigma'^2_{i,t} = \sigma^2_{i,t} \left[1 + \frac{a_{i,t}\phi(a_{i,t})}{1 - \Phi(a_{i,t})} - \left(\frac{\phi(a_{i,t})}{1 - \Phi(a_{i,t})} \right)^2 \right], \tag{12b}$$

where $a_{i,t} = [r^B_{K+1,t} - (\mu_{i,t} + r^{\mathrm{o}}_t)]/\sigma_{i,t}$. $\phi(a)$ and $\Phi(a)$ are the probability distribution function and the cumulative probability function of a standard normal distribution $N(a; 0, 1)$.

By transforming the term on the right in (11), the probability distribution of r^{o}_t conditioned on $\boldsymbol{r}^{B(1:K)}_t$ can be obtained as $p(r^{\mathrm{o}}_t|x_t, \boldsymbol{r}^{B(1:K)}_t) \ = N(r^{\mathrm{o}}_t; \widehat{r}^{\mathrm{o}}_t, \widehat{\sigma}^{\mathrm{o}2}_t)$, where $\widehat{r}^{\mathrm{o}}_t$ is the mean and $\widehat{\sigma}^2_r$ is the variance calculated by

$$\widehat{r}^{\mathrm{o}}_t = \widehat{\sigma}^{\mathrm{o}2}_t \sum_{i=1}^{K} \frac{r^B_{i,t} - \mu'_{i,t}}{\sigma'^2_{i,t}}, \quad \widehat{\sigma}^{\mathrm{o}2}_t = \left(\sum_{i=1}^{K} \sigma'^2_{i,t} \right)^{-1}. \tag{13a}$$

By considering this probability distribution as the observation process of r^{o}_t, and assuming its state transition probability as a normal distribution with the mean r^{o}_{t-1} and the variance $\sigma^{\mathrm{b}2}$, *i.e.*, $p(r^{\mathrm{o}}_t|r^{\mathrm{o}}_{t-1}) = N(r^{\mathrm{o}}_t; r^{\mathrm{o}}_{t-1}, \sigma^{\mathrm{b}2})$, the r^{o}_t can be estimated using the linear Kalman filter [2]. Let the mean and the variance of the posterior of r^{o}_t be $r^{\mathrm{o}}_{t|t}$ and $\sigma^{\mathrm{o}2}_{t|t}$, respectively. By iteratively updating $r^{\mathrm{o}}_{t|t}$ and $\sigma^{\mathrm{o}2}_{t|t}$ on the basis of the Kalman filter for each particle, the RSS offset can be estimated under location uncertainty. As a result of this extension, the state vector in the particle filter is augmented by additional variables as

$$\boldsymbol{z}_t = (x_t, y_t, f_t, v_t, \theta^{\mathrm{o}}_t, r^{\mathrm{o}}_{t|t}, \sigma^{\mathrm{o}2}_{t|t})^\top. \tag{14}$$

4.4. Fast Likelihood Computation

The Fast Likelihood Computation (FLC) module decomposes the regression model into a set of many local regression models to speed up the computation of the predicted values of RSS given a location. The number of beacons deployed in very large buildings can reach into the hundreds or even thousands. In order to build an accurate regression model, the fingerprinting process (*i.e.*, measuring RSS at various points in the building) may result in dozens or hundreds of thousands of measurements. Serious computational issues can arise when the number of fingerprint

points reaches this level of magnitude. In the observation model described in Section 3.3, the nonparametric regression based on a kernel function has a computational complexity time of $O(N)$ for predicting RSS given a location. The computation grows linearly with the number of fingerprint points N. This is clearly not practical because the particle filter must perform an $O(N)$ operation for every observation, every particle, and every beacon considered. Therefore, it is necessary to reduce the computational complexity of RSS prediction for large areas.

To mitigate this computational complexity issue, we split the regression model into small parts–local models–in input (location) space. For each local model, the computational complexity becomes $O(N_{\mathrm{sp}})$, where N_{sp} is the average number of training data assigned to each local model by splitting. N_{sp} is much smaller than the total number of fingerprint points N. At test time, we find the top M_{sp} local models closest to the current location estimate $\boldsymbol{x}_t$ and use their kernel-weighted average to compute the predicted values of RSS, similar to [49]. The computational complexity to predict the value of RSS can be reduced from $O(N)$ to $O(N_{\mathrm{sp}}M_{\mathrm{sp}})$. The M_{sp} is typically selected as a small natural number, *e.g.*, $M_{\mathrm{sp}} = 3$, to reduce the computational complexity as much as possible. In our implementation, the input space is split by the k-means clustering algorithm in the training phase.

4.5. Probabilistic Motion State Detection

When the localization accuracy is sufficiently improved, a slight detection failure of a user's motion state s_t can result in noticeable localization error of a few meters. Therefore, improving the motion state detection helps to reduce large localization error cases. In general, the motion state s_t is deterministically detected by thresholding the standard deviation of the magnitude of acceleration, σ^{acc} [43]. In real use cases, the localization system occasionally falls into a situation where the user's motion state is misclassified when a user walks very slowly or steps on soft flooring, *e.g.*, a mat or carpet. On the other hand, false detections have a negative effect on navigation because an estimated location becomes unstable even when the user is not moving. Decreasing the threshold value, therefore, does not resolve the situation.

As an alternative to the deterministic motion state detection, we propose probabilistic motion state detection. We construct a motion state detector that determines a user's motion state s_t for each particle according to a probability distribution $p(s_t|\sigma^{\mathrm{acc}})$ depending on the standard deviation. Specifically, we utilize a sigmoid function, $\varsigma(\sigma^{\mathrm{acc}}) = 1/(1 + \exp(-k_\varsigma(\sigma^{\mathrm{acc}} - \sigma^{\mathrm{acc}}_{0.5})))$, as the probability distribution function $p(s_t = 1|\sigma^{\mathrm{acc}})$ (shown in Fig. 2), where $\sigma^{\mathrm{acc}}_{0.5}$ is the σ^{acc} value of the sigmoid's midpoint and k_ς is the steepness of the curve. k_ς is determined by the threshold value $\sigma^{\mathrm{acc}}_{\mathrm{th}}$ and the corresponding function value $\varsigma_{\mathrm{th}} := \varsigma(\sigma^{\mathrm{acc}}_{\mathrm{th}})$ as well as $\sigma^{\mathrm{acc}}_{0.5}$. A moving state ($s_t = 1$) is sampled with the probability of $\varsigma(\sigma^{\mathrm{acc}})$, otherwise, a stopped state ($s_t = 0$) is sampled. The probabilistic motion state detection allows us to generate a moderate population of moving and stopped particles even in the case that the standard deviation is an intermediate value somewhat less than $\sigma^{\mathrm{acc}}_{\mathrm{th}}$ and motion detection tends to be uncertain.

4.6. Time Sensitive Observation Modeling

The time synchronized assumption of the observation model (Section 3.3), in which the RSS measurement $\boldsymbol{r}_t$ is time synced with the user's state $\boldsymbol{z}_t$, is only true when the user stops or moves very slowly. Since the RSS is computed by summing the power over a short time interval, the RSS value is actually dependent on a short history of the user's state, for example, the trajectory of the user in the last second. When the user moves very slowly, the past trajectory is essentially a single location and thus allows us to assume temporal synchronization in our model. However, when the user moves at a fast pace, we must take into account the fact that the RSS value reported by the smartphone is actually a 'stale' or 'time-averaged' measurement taken over a very recent history of user states. If we do not take into account the temporal delay in the measurements, it will result in a consistent lag in localization accuracy when the user moves quickly.

We consider two methods to mitigate the localization errors due to the delay in the measurements. The first method predicts a latest location from slightly stale observations, and the second one modifies the observation model to depend on locations for the last few seconds.

Method 1: Short-Time Location Prediction

The time increment Δt between the two time indices t and $t + 1$ is determined by the interval between two consecutive BLE beacon observations. Due to the interval Δt, an element of an RSS vector $\boldsymbol{r}_t$, *i.e.* the RSS value for a beacon $r_{i,t}$, can be reported with a delay up to Δt. Assuming that each element of the RSS vector is received almost evenly in the interval, the RSS vector can be regarded as being delayed $\Delta t/2$ on average. Therefore, the estimated

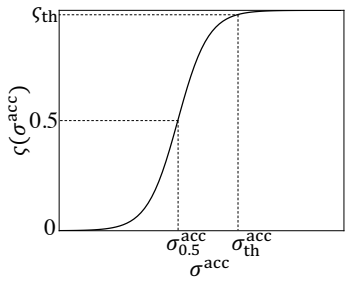
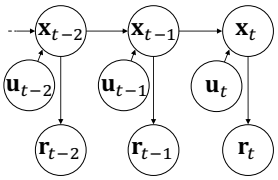
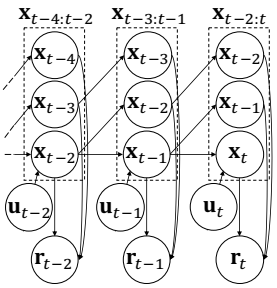


Figure 2: Probability function for probabilistic motion state detection.



(a) Observation model without delay



(b) Observation model with delay ($T = 2$)

Figure 3: A schematic representation of the state space model with different observation models. Location, input, and RSS vectors are displayed.

location $\boldsymbol{x}_{t|t}$ at a current time index t given all previous measurements $\boldsymbol{r}_{1:t}$ points to a location $\Delta t/2$ before the actual current location. We compensate for the delay in the time index by predicting the location of the user at a near future time index $t + 1/2$ rather than using the outdated location $\boldsymbol{x}_{t|t}$. We denote the predicted location at time index $t + 1/2$ given all previous measurements and inputs as $\boldsymbol{x}_{t+1/2|t}$. The $\boldsymbol{x}_{t+1/2|t}$ can be computed from the location, velocity, and orientation in the filtered particles $\{z^l_{t|t}\}^L_{l=1}$ by using the motion model (Section 3.2).

Method 2: Observation Model with Delay

According to our observations of RSS values reported by a smartphone operating system (iOS), RSS values reported at a certain moment reflect not only the device position at that moment but also the positions of the device in the past one or several seconds. With this method, we consider this effect by modifying the observation model so that it depends on the positions in the past several seconds. Figure 3 compares the observation model without delay (a) and with delay (b) regarding the dependency of an RSS vector on location vectors. For simplicity, only location, input, and RSS vectors are depicted; other variables are not shown. Formally, let $\boldsymbol{x}_{t-T:t}$ be the history of the location in the past few time indices $(\boldsymbol{x}_{t-T}, ..., \boldsymbol{x}_t)$, where T represents a delay time window. In the observation model (Section 3.3), we model the RSS for the i-th beacon by a Gaussian distribution with the mean $\mu_i(\boldsymbol{x}_t)$ and the variance σ_i^2. Assuming that a reported RSS value is smoothed by a moving average filter, the observation model for the i-th beacon considering the delay can be obtained by replacing its mean with the moving average of $\mu_i(\boldsymbol{x}_t)$ given the past location history, as

$$\mu_i(\boldsymbol{x}_{t-T:t}) = \frac{1}{T+1} \sum_{s=0}^{T} \mu_i(\boldsymbol{x}_{t-s}). \tag{15}$$

The RSS model for the i-th beacon can be modified to $p(r_{i,t}|\boldsymbol{x}_{t-T:t}) = N(r_{i,t}; \mu_i(\boldsymbol{x}_{t-T:t}), \sigma_i^2)$. To evaluate this likelihood, we extend the state vector $\boldsymbol{z}_t$ in the particle filter so that it stores not only the latest location $\boldsymbol{x}_t$ but also the location history $\boldsymbol{x}_{t-T:t}$.

5. Performance Evaluation

We evaluated the proposed localization system in a real-world environment: a shopping mall spanning three multi-story buildings and an underground public space that connects them. We first describe the experimental settings (environment and data collection) and evaluate the overall impact of the proposed improvements on the localization accuracy of the system. We then perform ablative evaluations of the proposed improvements. Specifically, for each improvement, we compare the localization accuracy of the complete system against a version of the localization approach with the selected improvement removed. The primary focus of this experiment is to investigate the effect of the proposed technical innovations, and not the accuracy of the base algorithm itself. Indeed, the proposed technical innovations are not necessarily limited to our specific base localization scheme; they can also be applied to similar systems in order to address specific problems in smartphone-based blind navigation in large and building-scale environments.

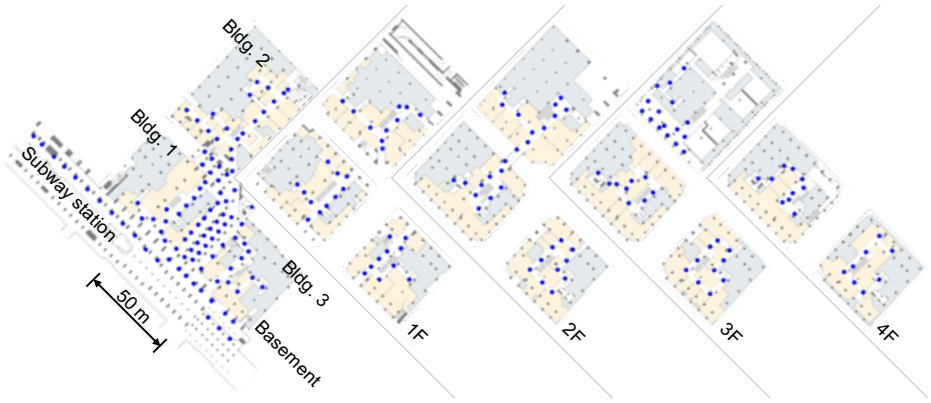


Figure 4: Floor maps of the experimental site from the basement to the 4th floor. Blue dots indicate beacon locations.

5.1. Experimental Settings

5.1.1. Environment

Figure 4 shows the floor maps of the experimental site. The testing environment covers three buildings: 1) a building with four floors and one basement, 2) a building with three floors and one basement, and 3) a building with four floors and one basement. The basement floors of these buildings are connected through an underground public pedestrian walkway. The total area of the experimental site is about 21,000 m^2. We extracted the information on accessible areas and floor transition areas (escalators and elevators) from the floor plans. A total of 218 beacons were installed in this environment, with about a distance of about 5–10 m between beacons, to enable indoor localization.

5.1.2. Data Collection

To evaluate our localization system, we collected fingerprint data and test data using the data acquisition equipment described below. Note that we collected both fingerprint and test data during business hours in which the shopping facilities were open. Thus, the evaluation conditions reflect the real-world use case, as the data were affected by crowds traversing the testing environment.

Data Collection Equipment: To reliably evaluate the localization accuracy, especially in situations where the user is moving, it is important to reduce human error in assigning ground truth locations to fingerprints and test data. Manually labeling all collected data with actual location information is a long and error-prone process, so we automated the data collection and ground truth assignment procedures using dedicated data collection equipment. Figure 5 shows our equipment, which is composed of the following items:

- a Velodyne VLP-16 LIDAR to record a point cloud of the surrounding environment.
- an Xsens Mti-30 Inertia Measurement Unit (IMU) to compensate for rotational movement of LIDAR during data collection.
- an Apple iPhone 6 smartphone to collect embedded sensor data and Bluetooth RSS data.
- an Apple iPhone 7 smartphone to collect embedded sensor data and Bluetooth RSS data for evaluation of adaptive signal calibration.
- a laptop computer cabled to other components to simultaneously record LIDAR, IMU, and smartphone data.

The collected data were later processed to reconstruct the measurement positions with a three-dimensional SLAM algorithm based on point cloud registration using Normal Distribution Transformation (NDT) [50, 51]. We then tested and validated the collected data by confirming that the projection of the registered point cloud onto a ground plane was in good agreement with the floor plans of the environment.

Fingerprint Collection: We collected fingerprints at roughly 1-meter intervals throughout the environment. The data acquisition equipment was fixed to an electric wheelchair (WHILL Model A[1]) during fingerprint collection to

[1] `http://whill.us/model-a-personal-mobility-device-personal-ev`

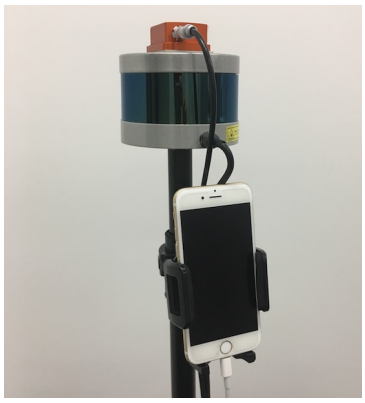


Figure 5: Data acquisition equipment.

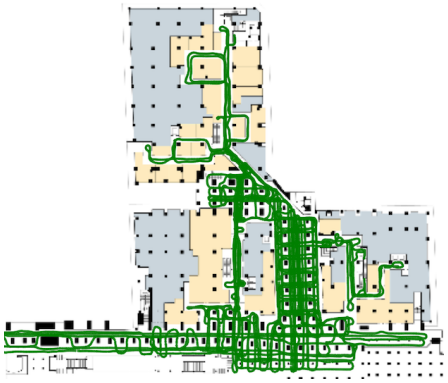


Figure 6: Basement floor fingerprint collection sequences.

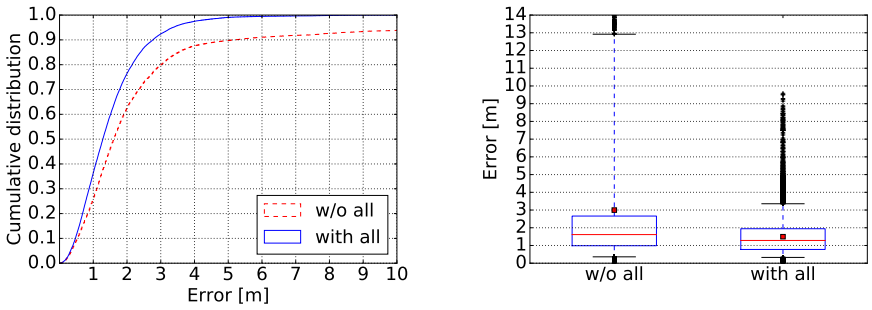
keep the movement of the equipment stable, with one experimenter controlling the movement of the wheelchair. We collected one sequence for each floor, per building, and three sequences for the underground public area, for a total of 17 sequences and 17,745 data samples. As an example, Fig. 6 shows the fingerprint positions on the basement floor.
Test Data Collection: In contrast to the fingerprint data, which only included pairs of locations and respective RSS vectors, our test data included additional sensor data used for the localization. We collected a time-series of iPhone sensor data (accelerometer, attitude, barometric pressure, heading, and BLE RSS data) with correct location labels. Even though the use of SLAM made it easier to assign correct location labels to test data, SLAM lacks semantic information (such as the time a user reached a target floor using an escalator or an elevator), which is important to evaluate performance. This additional information was input externally using a smartphone. For the testing, we collected three distinct datasets:

1. Static: On a single floor, walk along one route. Stop every 4–10 m for about fifteen seconds. (11,028 seconds, with iPhone 6)
2. Walk: On a single floor, walk along one route, from the starting point to the end point. (8,274 seconds, with iPhone 6)
3. Walk with Floor Transition: Walk along one route, from the starting point to the end point. A floor transition takes place using a vertical transportation device, *i.e.*, escalator or elevator. (2,477 seconds, with iPhone 6 and iPhone 7)

To evaluate the performance of the localization with different smartphones, the "Walk with Floor Transition" dataset 3) was collected with two different devices at the same time. Note that the second device (iPhone 7) has different signal receiving characteristics from the first device (iPhone 6), which is also the fingerprint collection device. On the devices, the update frequencies of the accelerometer, attitude, barometric pressure, and heading data are about 100 Hz, 100 Hz, 1 Hz, and 50 Hz, respectively. The report interval of BLE RSS readings is 1 second, and localization errors are calculated for each RSS update.

5.2. Overall Localization Error

We measured the overall localization accuracy on a global dataset comprising data from the three datasets collected with iPhone 6: 1) "Static", 2) "Walk", and 3) "Walk with Floor Transition". The comparison of localization errors between the base localization algorithm without any improvements and our localization system with all improvements is shown in Fig. 7, where (a) shows the localization error on the horizontal axis and the cumulative distribution of the error on the vertical axis. The localization error is calculated as Euclidean distance in (x,y) space. Figure 7b shows the 5th percentile, 25th percentile, median, 75th percentile, and 95th percentile error. We also mark the mean as a red point. Although the median error of both the base algorithm (1.6 m) and our system (1.3 m) shows a small difference, in large error cases, the base algorithm without the improvements performed significantly worse than our proposed system. In particular, the 95th percentile error decreased from 12.9 m to 3.4 m with our system, resulting in a mean accuracy improvement from 3.0 m to 1.5 m. These values indicate that our system can manage localization much



(a) Cumulative error distribution

(b) Localization error box plot

Figure 7: Overall localization error.

better in critical cases, which makes it a better approach for real-world applications. In the following sections, we assess the impact of the six enhancements to localization: (1) Localization Integrity Monitoring, (2) Floor Transition Control, (3) Adaptive Signal Calibration, (4) Fast Likelihood Computation, (5) Probabilistic Motion State Detection, and (6) Time Sensitive Observation Modeling.

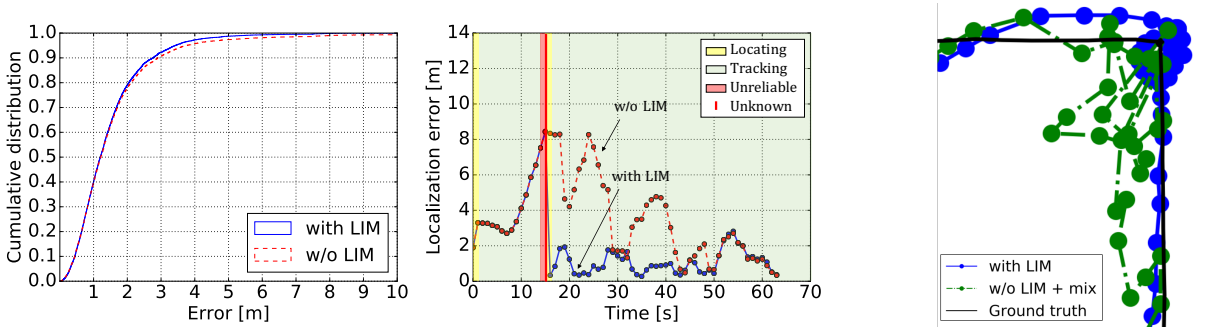
5.3. Localization Integrity Monitoring

The effect of the localization integrity monitoring module is evident in situations in which a large number of particles in the particle filter fail to approximate the probability distribution of the localization state. This may happen when the user's movement model diverges from the expectation and the resulting errors accumulate. To assess the effect of the localization integrity monitoring module in this case, we evaluated the localization error with and without the module using the test dataset 2) "Walk". Figure 8 plots the effects of the localization integrity monitoring module, where (a) is the cumulative distribution of localization error and (b) is an example time series of localization error. In Fig. 8a, we can see that the large localization error is reduced by applying the localization integrity monitoring module, although the amount of improvement is smaller compared to the preliminary version [7] due to the overall improvement of the system. To investigate the effect of the localization integrity monitoring module, one example time series that includes catastrophic failures in localization is extracted in Fig. 8b. The localization error both with and without localization integrity monitoring is the same until around 20 s after localization starts, after which the localization error without localization integrity monitoring hits a peak and gradually decreases to the initial level. In contrast, the localization integrity monitoring module manages to re-initiate the localization before the error peak, and thus provides a higher localization accuracy.

We assessed the continuity of the tracked trajectories obtained by our localization system. As an intuitive metric to measure the continuity, we compare the traveled distance between the ground truth and the estimated trajectories. The traveled distance becomes longer when location estimates jump around excessively. For relative evaluation, we also measured the traveled distance obtained by a naïve approach for improving robustness [1] in which a small fraction (10 %) of the particles in the particle filter are sampled from incoming RSS measurement and mixed into the particles (hereafter, referred to as "the mix approach"). Compared to the travel distance of the ground truth trajectories (6784 m), the estimated trajectories by our approach and by the mix approach were 6893 m (1.6 % increase) and 7007 m (3.3 % increase), respectively. Figure 8c shows one of the most prominent examples of the trajectories obtained by our approach and by the mix approach. The mix approach sometimes causes the location estimates to jump around discontinuously as shown in Fig. 8c, which makes the traveled distance excessively long. In contrast, the localization integrity monitoring approach is able to provide more consistent location estimates while reducing catastrophic failures.

5.4. Floor Transition Control

We compare (i) the localization system without the floor transition control module and (ii) the system complemented with our floor transition control module (Section 4.2). We compute the localization error on the test dataset 3) "Walk with Floor Transition" with iPhone 6, which contains routes that include the use of escalators or elevators.

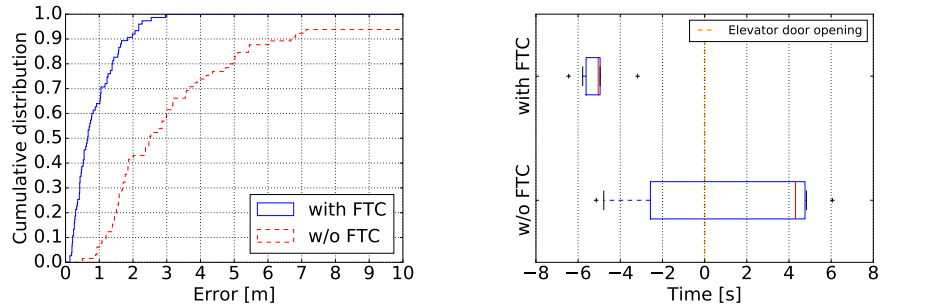


(a) Cumulative localization error

(b) Example time series of localization error

(c) Example estimated trajectories

Figure 8: Effect of Localization Integrity Monitoring.



(a) Cumulative localization error during five seconds after floor transition

(b) Time of detection of arrival to floor with respect to actual arrival

Figure 9: Effect of Floor Transition Control.

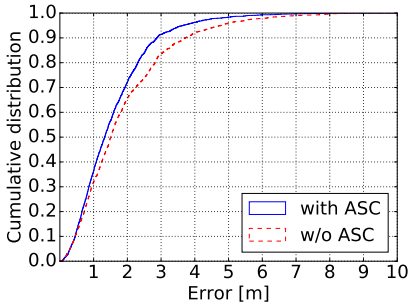
The goals of the floor transition control module are to a) relocalize the user correctly upon leaving the elevator and b) notify the user of the floor change in time (about five seconds before actually reaching the floor). Because the floor transition control impacts the localization only around floor transitions, we evaluated the errors in a short period (five seconds) after floor transition is finished.

Figure 9 depicts the effect of the floor transition control on the localization, where (a) shows the cumulative distribution of the localization error for a short period of time (5 seconds) after floor transition and (b) shows times when the localization system detected the arrival to the target floor with respect to the actual arrival. The arrival time of the elevator to the target floor was annotated at the moment the elevator doors started to open. A negative value means that the localization on the floor happened before the actual arrival to the target floor, while a positive value means that the device has been detected on the target floor after the actual arrival. Considering the voice navigation assistant use case, it is preferred that the application notices the arrival to the target floor a few seconds before the actual arrival because it takes few seconds to notify a user of the arrival and next actions to perform by voice instructions.

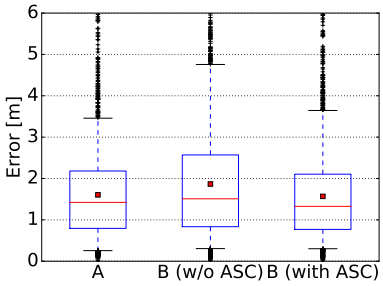
In Fig. 9a, the localization error with the floor transition control is visibly better than without floor transition control. This is further backed up in Fig. 9b, which shows that the floor transition control module detected that the device had reached the target floor an average of five seconds before the actual arrival, which is the desired result. In contrast, without floor transition control, the average moment at which the floor transition is detected ranges from about two seconds before to five seconds after reaching the target floor.

5.5. Adaptive Signal Calibration

We evaluate the localization with and without the adaptive signal calibration on the test dataset 3) “Walk with Floor Transition” that was recorded by two different devices. For simplicity, we denote the device for fingerprint collection (Apple iPhone 6) as “A” and the other device (Apple iPhone 7) as “B”. Figure 10 plots the effect of the

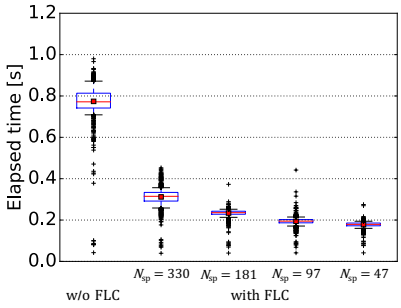


(a) Cumulative localization error by device B

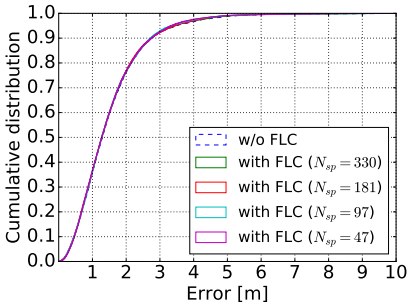


(b) Comparison of device A and device B

Figure 10: Effect of Adaptive Signal Calibration.



(a) Elapsed time to process one second of input



(b) Cumulative localization error

Figure 11: Effect of Fast Likelihood Computation.

adaptive signal calibration on localization error, where (a) is the cumulative error distribution and (b) is a boxplot to compare statistical values of device A, device B without adjustment, and device B with adjustment. Compared to the previous two modules that significantly impact a limited part of the data, this module moderately reduces localization error across the whole sequence, as seen in Fig. 10a. To investigate this effect in detail, we compare the error obtained by device B and device A in Fig. 10b. As shown, this module reduces the error of device B to the level of error obtained by device A which has no RSS offset.

5.6. Fast Likelihood Computation

We evaluate the impact of the fast likelihood computation on computation times and localization error. We run the localization system with and without this enhancement on an Apple iPhone 6 smartphone. Figure 11 plots the effect of the fast likelihood computation module, where (a) indicates the time required to process one second of input and (b) indicates the cumulative localization error. We investigated the input space partitioning with four different settings: $N_{\text{sp}} = 330, 181, 97, 47$. The top $M_{\text{sp}} = 3$ local models are used to predict RSS values. The other parameters that affect computational times were set to fixed values ($L = 300$ and $K = 10$). The average elapsed time decreased by 77 percent (from 0.77 s to 0.18 s) thanks to the computational efficiency improvement (Fig. 11a). At the same time, the localization error does not show any increase (Fig. 11b). With this improvement, the localization system achieved a sufficiently small computational burden to run flawlessly on a commodity smartphone device.

5.7. Probabilistic Motion State Detection

We evaluate the effect of the probabilistic motion state detection on the whole dataset collected with iPhone 6. Because the dataset was collected in a large testing environment over several periods of time, it includes some cases in which motion states tend to be overlooked, *e.g.*, walking slowly or on a carpet in a movie theater. The threshold

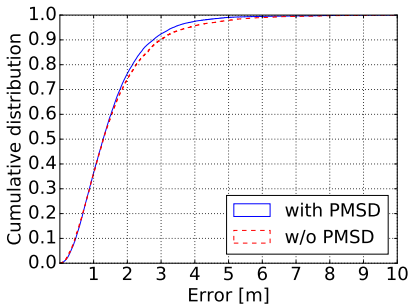


Figure 12: Effect of Probabilistic Motion State Detection.

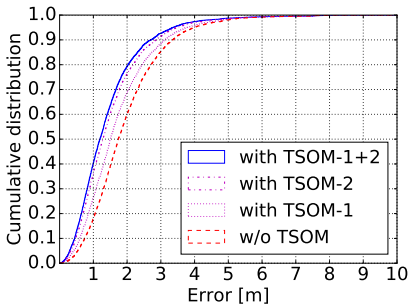


Figure 13: Effect of Time Sensitive Observation Modeling.

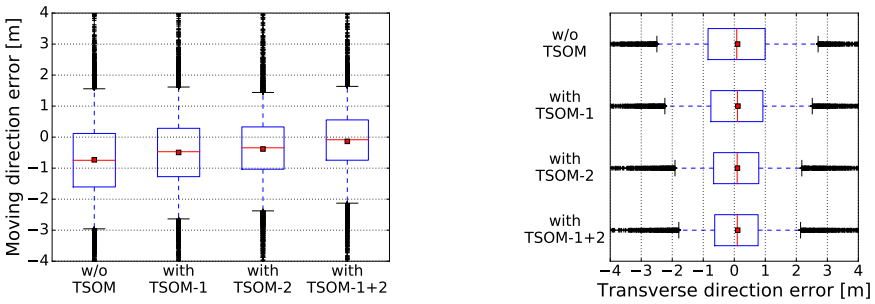
value for motion state detection $\sigma_{\mathrm{th}}^{\mathrm{acc}}$ is set to 0.5 g, where 1 g = 9.8 m/s^2. The standard deviation of the acceleration is calculated over a window of 0.8 s. For the cumulative distribution function, we set $\sigma_{0.5}^{\mathrm{acc}}$ to $\sigma_{\mathrm{th}}^{\mathrm{acc}}/2$ and ς_{th} to 0.95, and k_ς is calculated from these values. Figure 12 plots the effect of the probabilistic motion state detection on the localization error. Localization error larger than about two meters is slightly reduced thanks to the decrease in error corresponding to failure of motion state detection for a couple of seconds, which results in the 95th percentile error decrease from 3.8 m to 3.4 m and the 99th percentile error decrease from 6.0 m to 4.9 m.

5.8. Time Sensitive Observation Modeling

We evaluate the effects of the time sensitive observation modeling methods on localization by using the "Walk" dataset collected in the environment. On iOS, the interval between two consecutive BLE beacon notifications is one second; therefore, the Δt is set to one second. On the basis of a preliminary experiment to determine the response of RSS values to position changes on an iPhone 6, we set the delay time window T to two through this experiment. Figure 13 compares the localization error achieved under four different settings: without time sensitive observation modeling, with Method 1 (Short-time Location Prediction), with Method 2 (Observation Model with Delay), and with both Method 1 and Method 2. The localization error becomes smaller in the order as written above. To clearly understand the effect of the time sensitive observation modeling methods, we decompose the localization error in the moving direction (Fig. 14a) and the transverse direction (Fig. 14b). The moving directions are calculated from the trajectories of ground truth locations when the locations are changing faster than 0.1 m/s. A positive value of the localization error in the moving direction means the estimated location is ahead of the ground truth location, while a negative value means the estimated location is behind it. The localization error in the transverse direction is defined such that a positive value indicates the estimated location is on the right side of the ground truth location and a negative value indicates the left side. In Fig. 14a, the mean localization error in the moving direction is improved from −0.7 m (without time sensitive observation modeling) to −0.1 m (with Method 1 and Method 2), which means that the delays in the estimated locations are compensated as desired. As a subsequent effect, both the 5th percentile and the 95th percentile error in the transverse direction are improved about 0.6 m with time sensitive observation modeling (both Method 1 and Method 2), as shown in Fig. 14b.

6. Evaluation with Users

We integrated our localization system in a turn-by-turn navigation application [42] and then evaluated the localization accuracy while actual users with visual impairments traversed the experiment field with the navigation application. The localization and navigation results in this user experiment were obtained with the previous version of the localization system [7].



(a) Localization error in the moving direction

(b) Localization error in the transverse direction

Figure 14: Effect of Time Sensitive Observation Modeling on localization error decomposed in two directions.

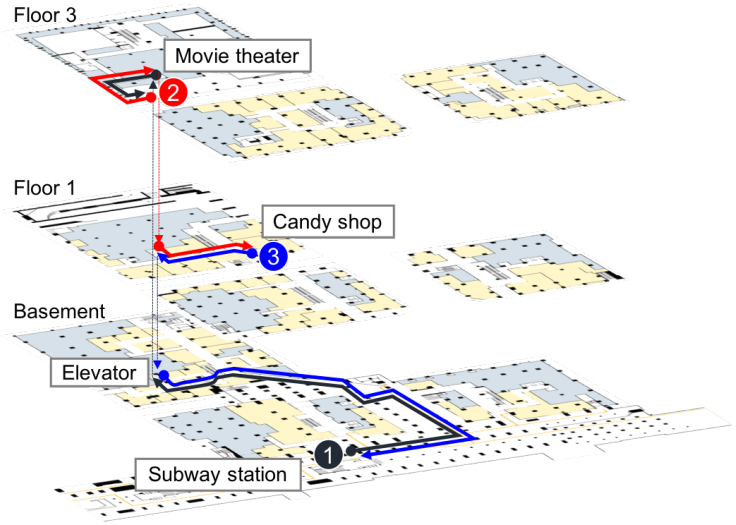


Figure 15: Three routes for the navigation experiment.

6.1. Method

We recruited ten participants (4 m/6 f) with visual impairments (6 legally blind and 4 low vision) aged from 33 to 54 (M=44, SD=5.9) years. One participant brought her guide dog while the others used white canes while navigating with the system. During the study, participants were asked to navigate three fixed routes in the shopping mall (as shown in Fig. 15): 1) from the area in front of the subway station on the basement floor to a movie theater on the 3rd floor (177 m), 2) from the movie theater to a candy shop on the 1st floor (54 m), and 3) from the candy shop back to the subway station. The total number of turns in these routes was 26. Note that each route included a transition between floors via an elevator. All participants were asked to wear a waist bag with the phone attached to it in order to free their hands from holding the phone during navigation, which is important for users with visual impairments as their hand is often occupied holding a cane or a guide dog's leash. An experimenter followed close behind all participants and videotaped them with a camera. We visually annotated participants' actual location every second, except for when they were inside an elevator. We also annotated their turn performance, *i.e.*, whether they successfully made a turn without the experimenter's help.

6.2. Results and Discussion

We evaluate the localization error between users' actual locations and corresponding estimated locations. The number of annotated location points is 7,641 from all routes for all participants. We obtained 1.7 m mean localization

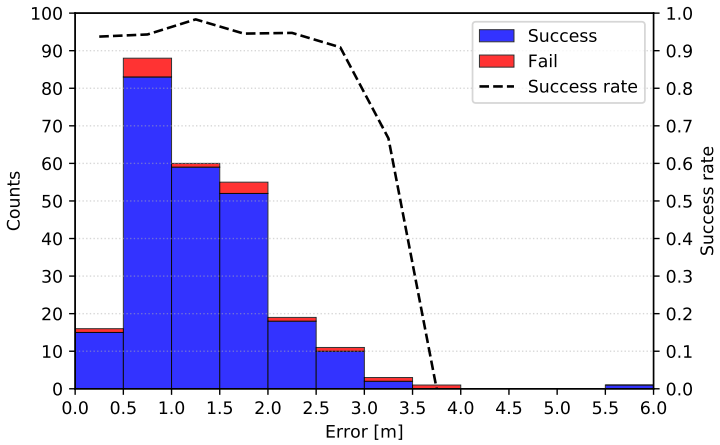


Figure 16: Localization error and turn performance.

error and 3.2 m localization error at the 95th percentile during navigation. We also investigate the performance of navigation relying on our localization system. Of the 260 turns in total (26 turns per participant), 243 turns (93.5%) were successful without the experimenter's help. These results demonstrate that the proposed system can provide actual visually impaired users with location information sufficient for navigation.

To examine the impact of the localization error on the turn performance, we plot the localization error at the turns, the count of turn results (success or fail), and the rate of success divided into 0.5-m intervals (see Fig. 16). Note that a success in the 5.5–6.0 m range was counted because the participant was eventually able to return to the navigation route after walking for a while, although the navigation error caused by the large localization error made the participant turn far before the corner.

We discuss the findings from the experiment conducted with the previous version of the localization system and the benefits of the technical innovations introduced in this paper. Figure 16 shows that the turn success rate does not change much when the localization error is lower than 3 m. This confirms our initial intuition that the localization error in the range of about two meters is appropriate for turn-by-turn navigation assistance for people with visual impairments. However, recent research highlights that environment [52–54] and user performance [55] may also influence navigation assistance requirements, and therefore further investigations may be needed.

On the other hand, the turn success rate drops with localization error larger than 3 m. Therefore, to improve the navigation performance on top of the speech-based navigation system, reducing large error values has a profound effect. These findings obtained with the prior version of the localization system highlight the benefit of the additional extensions presented in Sections 4.5 and 4.6 that improved the extreme localization error values, as shown in Sections 5.7 and 5.8.

7. Conclusion

To enable automated turn-by-turn indoor navigation assistance for individuals with visual impairments, a localization system that can achieve high levels of accuracy in building-scale real-world environments is essential. In this paper, we considered challenges for accurate localization related to the large-scale nature of the environments and the user's mobility. To address these challenges within a unified framework, we designed and implemented a localization system that enhances a probabilistic localization algorithm with a series of innovations in order to achieve accurate real-time localization We performed a series of experiments with ground truth data collected in a large indoor environment composed of three multi-story buildings and an underground passageway. The experimental evaluations validated the effect of the enhancement modules to improve the localization accuracy and provide real-time navigation capabilities in real-world scenarios. Specifically, the mean localization error decreased from 3.0 m to 1.5 m. We also

evaluated the practical performance of the system in a study with visually impaired participants and found that the proposed localization system helps their independent mobility.

Acknowledgments

We thank Shimizu Corporation and Mitsui Fudosan for their collaboration.

Conflict of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

References

[1] S. Hilsenbeck, D. Bobkov, G. Schroth, R. Huitl, E. Steinbach, Graph-based data fusion of pedometer and WiFi measurements for mobile indoor positioning, in: Proc. of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp), ACM, 2014, pp. 147–158.

[2] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics, MIT press, 2005.

[3] D. Lymberopoulos, J. Liu, X. Yang, R. R. Choudhury, V. Handziski, S. Sen, A realistic evaluation and comparison of indoor location technologies: Experiences and lessons learned, in: Proc. of the 14th International Conference on Information Processing in Sensor Networks (IPSN), ACM, 2015, pp. 178–189.

[4] A. W. Tsui, Y.-H. Chuang, H.-H. Chu, Unsupervised learning for solving RSS hardware variance problem in WiFi localization, Mobile Networks and Applications 14 (5) (2009) 677–691.

[5] L. Li, G. Shen, C. Zhao, T. Moscibroda, J.-H. Lin, F. Zhao, Experiencing and handling the diversity in data density and environmental locality in an indoor positioning service, in: Proc. of the 20th Annual International Conference on Mobile Computing and Networking (MobiCom), ACM, 2014, pp. 459–470.

[6] H. Xu, Z. Yang, Z. Zhou, L. Shangguan, K. Yi, Y. Liu, Enhancing WiFi-based localization with visual clues, in: Proc. of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp), ACM, 2015, pp. 963–974.

[7] M. Murata, D. Ahmetovic, D. Sato, H. Takagi, K. M. Kitani, C. Asakawa, Smartphone-based indoor localization for blind navigation across building complexes, in: Proc. of the 2018 IEEE International Conference on Pervasive Computing and Communications (PerCom), IEEE, 2018, pp. 254–263.

[8] P. Bahl, V. N. Padmanabhan, RADAR: An in-building RF-based user location and tracking system, in: Proc. IEEE International Conference on Computer Communications (INFOCOM), Vol. 2, IEEE, 2000, pp. 775–784.

[9] H. Liu, H. Darabi, P. Banerjee, J. Liu, Survey of wireless indoor positioning techniques and systems, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 37 (6) (2007) 1067–1080.

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

[11] Z. Yang, C. Wu, Z. Zhou, X. Zhang, X. Wang, Y. Liu, Mobility increases localizability: A survey on wireless indoor localization using inertial sensors, ACM Computing Surveys 47 (3) (2015) 54:1–54:34.

[12] S. He, S.-H. G. Chan, Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons, IEEE Communications Surveys & Tutorials 18 (1) (2016) 466–490.

[13] B. Ferris, D. Haehnel, D. Fox, Gaussian processes for signal strength-based location estimation, in: Proc. of Robotics Science and Systems, 2006.

[14] F. Subhan, H. Hasbullah, A. Rozyyev, S. T. Bakhsh, Indoor positioning in Bluetooth networks using fingerprinting and lateration approach, in: Proc. of International Conference on Information Science and Applications, IEEE, 2011.

[15] R. Faragher, R. Harle, Location fingerprinting with Bluetooth Low Energy beacons, IEEE Journal on Selected Areas in Communications 33 (11) (2015) 2418–2428.

[16] J. Wang, D. Katabi, Dude, where's my card?: RFID positioning that works with multipath and non-line of sight, ACM SIGCOMM Computer Communication Review 43 (4) (2013) 51–62.

[17] S. Gezici, Z. Tian, G. B. Giannakis, H. Kobayashi, A. F. Molisch, H. V. Poor, Z. Sahinoglu, Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks, IEEE Signal Processing Magazine 22 (4) (2005) 70–84.

[18] P. Lazik, N. Rajagopal, O. Shih, B. Sinopoli, A. Rowe, ALPS: A Bluetooth and ultrasound platform for mapping and localization, in: Proc. of the 13th ACM Conference on Embedded Networked Sensor Systems (SenSys), ACM, 2015, pp. 73–84.

[19] Z. Yang, Z. Zhou, Y. Liu, From RSSI to CSI: Indoor localization via channel response, ACM Computing Surveys 46 (2) (2013) 25:1–25:32.

[20] D. Vasisht, S. Kumar, D. Katabi, Decimeter-level localization with a single WiFi access point, in: Proc. of the 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2016, pp. 165–178.

[21] L. Banin, U. Schatzberg, Y. Amizur, WiFi FTM and map information fusion for accurate positioning, in: 2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2016.

[22] T. Roos, P. Myllymäki, H. Tirri, P. Misikangas, J. Sievänen, A probabilistic approach to WLAN user location estimation, International Journal of Wireless Information Networks 9 (3) (2002) 155–164.

[23] Y. Gwon, R. Jain, Error characteristics and calibration-free techniques for wireless LAN-based location estimation, in: Proc. of the Second International Workshop on Mobility Management & Wireless Access Protocols, ACM, 2004, pp. 2–9.

[24] K. Chintalapudi, A. Padmanabha Iyer, V. N. Padmanabhan, Indoor localization without the pain, in: Proc. of the 16th Annual International Conference on Mobile Computing and Networking (MobiCom), ACM, 2010, pp. 173–184.
[25] Z. Yang, C. Wu, Y. Liu, Locating in fingerprint space: wireless indoor localization with little human intervention, in: Proc. of the 18th Annual International Conference on Mobile Computing and Networking (MobiCom), ACM, 2012, pp. 269–280.
[26] C. Gleason, D. Ahmetovic, S. Savage, C. Toxtli, C. Posthuma, C. Asakawa, K. M. Kitani, J. P. Bigham, Crowdsourcing the installation and maintenance of indoor localization infrastructure to support blind navigation, Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 2 (1) (2018) 9.
[27] X. Zhao, Z. Xiao, A. Markham, N. Trigoni, Y. Ren, Does BTLE measure up against WiFi? a comparison of indoor location performance, in: Proc. of the 20th European Wireless Conference (EW), 2014.
[28] F. Palumbo, P. Barsocchi, S. Chessa, J. C. Augusto, A stigmergic approach to indoor localization using Bluetooth Low Energy beacons, in: Proc. of the 12th IEEE International Conference on Advanced Video and Signal Based Surveillance, 2015.
[29] O. Woodman, R. Harle, Pedestrian localisation for indoor environments, in: Proc. of the 10th International Conference on Ubiquitous Computing (UbiComp), ACM, 2008, pp. 114–123.
[30] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, F. Zhao, A reliable and accurate indoor localization method using phone inertial sensors, in: Proc. of the 2012 ACM Conference on Ubiquitous Computing (UbiComp), ACM, 2012, pp. 421–430.
[31] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, R. Sen, Zee: Zero-effort crowdsourcing for indoor localization, in: Proc. of the 18th Annual International Conference on Mobile Computing and Networking (MobiCom), ACM, 2012, pp. 293–304.
[32] S. He, S.-H. G. Chan, L. Yu, N. Liu, Calibration-free fusion of step counter and wireless fingerprints for indoor localization, in: Proc. of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp), ACM, 2015, pp. 897–908.
[33] G. H. Flores, R. Manduchi, WeAllWalk: An annotated dataset of inertial sensor time series from blind walkers, ACM Transactions on Accessible Computing 11 (1) (2018) 4.
[34] C. Wu, Z. Yang, Z. Zhou, Y. Liu, M. Liu, Mitigating large errors in WiFi-based indoor localization for smartphones, IEEE Transactions on Vehicular Technology 66 (7) (2017) 6246–6257.
[35] N. A. Giudice, G. E. Legge, Blind navigation and the role of technology, The Engineering Handbook of Smart Technology for Aging, Disability, and Independence (2008) 479–500.
[36] N. Fallah, I. Apostolopoulos, K. Bekris, E. Folmer, Indoor human navigation systems: A survey, Interacting with Computers 25 (1) (2013) 21–33.
[37] J.-E. Kim, M. Bessho, S. Kobayashi, N. Koshizuka, K. Sakamura, Navigating visually impaired travelers in a large train station using smartphone and Bluetooth Low Energy, in: Proc. of the 31st Annual ACM Symposium on Applied Computing, ACM, 2016, pp. 604–611.
[38] S. A. Cheraghi, V. Namboodiri, L. Walker, Guidebeacon: Beacon-based indoor wayfinding for the blind, visually impaired, and disoriented, in: Proc. of IEEE International Conference on Pervasive Computing and Communications (PerCom), IEEE, 2017, pp. 121–130.
[39] D. Ahmetovic, C. Gleason, K. M. Kitani, H. Takagi, C. Asakawa, NavCog: turn-by-turn smartphone navigation assistant for people with visual impairments or blindness, in: Proc. of the 13th Web for All Conference, ACM, 2016, pp. 9:1–9:2.
[40] D. Ahmetovic, C. Gleason, C. Ruan, K. Kitani, H. Takagi, C. Asakawa, NavCog: a navigational cognitive assistant for the blind, in: Proc. of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services, ACM, 2016, pp. 90–99.
[41] D. Ahmetovic, M. Murata, C. Gleason, E. Brady, H. Takagi, K. Kitani, C. Asakawa, Achieving practical and accurate indoor navigation for people with visual impairments, in: Proc. of the 14th Web for All Conference (W4A), 2017, pp. 31:1–31:10.
[42] D. Sato, U. Oh, K. Naito, H. Takagi, C. Asakawa, K. Kitani, NavCog3: An evaluation of a smartphone-based blind indoor navigation assistant with semantic features in a large-scale environment, in: Proc. of the 19th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS), ACM, 2017, pp. 270–279.
[43] A. Brajdic, R. Harle, Walk detection and step counting on unconstrained smartphones, in: Proc. of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp), ACM, 2013, pp. 225–234.
[44] K. P. Murphy, Machine Learning. A Probabilistic Perspective, MIT press, 2012.
[45] J. Fink, V. Kumar, Online methods for radio signal mapping with mobile robots, in: Proc. of the 2010 IEEE International Conference on Robotics and Automation, 2010, pp. 1940–1945.
[46] C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
[47] K. Muralidharan, A. J. Khan, A. Misra, R. K. Balan, S. Agarwal, Barometric phone sensors: More hype than hope!, in: Proc. of the 15th Workshop on Mobile Computing Systems and Applications, ACM, 2014, pp. 12:1–12:6.
[48] W. H. Greene, Econometric Analysis (7th Edition), Pearson Education, 2011.
[49] D. Nguyen-Tuong, J. Peters, Local gaussian process regression for real-time model-based robot control, in: Proc. of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2008, pp. 380–385.
[50] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, T. Hamada, An open approach to autonomous vehicles, IEEE Micro 35 (6) (2015) 60–68.
[51] M. Magnusson, The three-dimensional normal-distributions transform: an efficient representation for registration, surface analysis, and loop detection, Ph.D. thesis, Örebro universitet (2009).
[52] D. Ahmetovic, U. Oh, S. Mascetti, C. Asakawa, Turn right: Analysis of rotation errors in turn-by-turn navigation for individuals with visual impairments, in: Proc. of the 20th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS), ACM, 2018, pp. 333–339.
[53] J. Guerreiro, E. Ohn-Bar, D. Ahmetovic, K. Kitani, C. Asakawa, How context and user behavior affect indoor navigation assistance for blind people, in: Proc. of the 15th International Cross-Disciplinary Conference on Web Accessibility, W4A '18, ACM, New York, NY, USA, 2018, pp. 2:1–2:4.
[54] H. Kacorri, E. Ohn-Bar, K. M. Kitani, C. Asakawa, Environmental factors in indoor navigation based on real-world trajectories of blind users, in: Proc. of the 2018 CHI Conference on Human Factors in Computing Systems, CHI '18, ACM, New York, NY, USA, 2018, pp. 56:1–56:12.
[55] E. Ohn-Bar, J. Guerreiro, K. Kitani, C. Asakawa, Variability in reactions to instructional guidance during smartphone-based assisted navigation of blind users, Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 2 (3) (2018) 131.