IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 15, 2021, accepted March 24, 2021, date of publication March 29, 2021, date of current version April 6, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3069261

A Precise Tracking Algorithm Using PDR and Wi-Fi/iBeacon Corrections for Smartphones

TUAN D. VY, (Student Member, IEEE), THU L. N. NGUYEN, AND YOAN SHIN, (Senior Member, IEEE)
School of Electronic Engineering, Soongsil University, Seoul 06978, South Korea
Corresponding author: Yoan Shin (yashin@ssu.ac.kr)

This work was supported by the NRF Grant funded by the Korean Government (MSIT) under Grant 2020R1A2C2010006.

ABSTRACT Considering smartphone-based indoor localization and tracking, combination of Wi-Fi fingerprinting and pedestrian dead reckoning (PDR) becomes prevalent due to wide deployment of indoor Wi-Fi access points and easily access of inertial measurement units (IMUs) on smartphones. Since Wi-Fi fingerprinting depends on the similarities between online and offline received signal strengths, it suffers from the fluctuation of radio signal measurements. Meanwhile, the PDR relies on the IMUs, which undergo accumulated errors in long walking distance. In this paper, we aim to improve the indoor tracking accuracy, while filling up the gap between two popular Android and iOS platforms. In particular, we propose a hybrid tracking algorithm that integrates the PDR with Wi-Fi/iBeacon signal features. There are three key points in our paper. First, we process Wi-Fi/iBeacon signals to build conversion functions to convert signals to related proximity distances. Besides, in order to support mobile tracking, we introduce a placement methodology for iBeacon installation for a specific region of interest. Second, a mobile smartphone uses an improved PDR to localize itself, while incorporating Wi-Fi/iBeacon data for estimating a starting point and correcting its location along the walking path. Finally, we build an iOS app to implement the proposed scheme and display visual tracking results. Experiment results show that our proposed scheme is more robust and accurate compared to the conventional schemes.

INDEX TERMS Smartphone based indoor localization, pedestrian dead reckoning, iBeacon placement methodology, Wi-Fi proximity.

I. INTRODUCTION

The development of emerging indoor localization technologies has enabled plenty of applications such as tracking, navigation and pinpointing user's location in real time. By perceiving and collecting data, the aim of tracking task is to identify the location of a moving object continuously by using the hybrid signals generated through wireless sensing [1]–[3]. In order to support indoor user's experiences, different signals such as Wi-Fi, radio-frequency identification (RFID), Bluetooth, ultra wide band (UWB), to name a few [1]–[4], have been utilized. Global positioning system (GPS) available on smartphones is able to determine user location in real time regardless of whether or not it is connected the Internet. Unfortunately, it drains smartphone battery life quickly due to the signaling and computing processes with the GPS satellites. Also, the GPS does not work well in indoor environments because the satellite signals become too weak when they penetrate through indoor building materials. In order to provide accurate locations for indoor users, it requires to develop new indoor localization technologies which reflect user behavior in buildings, while supporting mobile computing and energy-aware localization.

Nowadays, Wi-Fi access points (APs) become more prevalent not only in indoor buildings but also in public common places (e.g., parks, airports, malls, etc.). Wi-Fi-based localization has been used widely in the last decades since it does not require extra infrastructure and specialized hardware. Wi-Fi fingerprinting is one of common techniques [1]–[5]. With this approach, we explore the spatial features of signals at different reference locations and analyze distinctive signal patterns as fingerprints. For better accuracy, fingerprints are collected at several locations and then the locations are determined through the

The associate editor coordinating the review of this manuscript and approving it for publication was Heng Wang.

matching process of the signal patterns. However, Wi-Fi fingerprinting-based approaches require labor effort and maintenance costs. Regarding indoor tracking assisted by smartphones, recent indoor Wi-Fi-based localization schemes have incorporated with the pedestrian dead reckoning (PDR) to provide meter-level tracking accuracy as well as support multiple mobile users [6]–[9]. Those schemes become applicable to modern smartphones because Wi-Fi antenna and motion-sensitive inertial measurement units (IMUs) (e.g., pedometer, accelerometer, gyroscope, etc.) are already equipped on the smartphones. To accomplish a tracking task, a typical hybrid Wi-Fi and PDR based system involves three main components: fingerprint collection, location estimation and location update. During the data collection, fingerprints from mobile user are recorded at several reference points in the area of interest (AoI). For example, we adopt received signal strength indicator (RSSI) to build an offline RSSI database. At each single time, RSSI is measured by the smartphone and used for estimating the user location. When the user is moving, IMU data are collected and fed the PDR to update this location continuously, while utilizing Wi-Fi measurements for correcting the location when a user veers off the correct path. Therefore, the hybrid schemes between Wi-Fi and PDR not only improve the localization accuracy but also have low complexity. While most of the studies [6]–[9] focus on developing localization schemes on popular Android OS, there are not many works on iOS because of its secure platform since iOS 13 updates. For instance, iPhone cannot get raw RSSI from currently-connected Wi-Fi AP because the public application programming interface (API) no longer returns valid Wi-Fi service set identifiers and basic service set identifiers information [10], [11]. Tracking apps running on iPhones become more restricted compares to Android ones. Moreover, the PDR often suffers from large drifts and significant errors over a long walking path. Several works [7]–[9] tried to utilize geometrical constraints by the trajectory to reduce location estimation errors. In [12]–[19], they exploited improved user's mobility factors such as step detection, step length estimation and heading direction to ensure accurate inputs to the PDR.

Currently, Bluetooth low energy (BLE)-based localization approaches are useful since BLE beacons are cheap and energy-efficient, while achieving room-level accuracy. In order to improve accuracy, BLE beacon usually combines with other technologies (e.g., Wi-Fi, PDR). Recently, Apple Inc. releases iBeacon protocol since 2013 [20]. An iBeacon is a BLE proximity beacon and compatible under an Apple licensing program. Unlike GPS, Wi-Fi or conventional Bluetooth, iBeacon offers high-precision location information and saves much more energy. Recent works [21]–[24] demonstrate that the hybrid PDR and Wi-Fi/PDR approaches can benefit from user mobility information and additional measurements from nearby Wi-Fi APs and iBeacons. Specially, Wi-Fi/iBeacon measurements collected by the smartphones can be served as reference data. The collected data may contain two parts: RSSI samples at static state and RSSI readings along mobility data during moving. Then, data received at the smartphone are fed to localization engine to calculate and to calibrate its location.

In this paper, we aim to build a simple self-adaptive pedestrian tracking scheme for current smartphone constraints and Wi-Fi/iBeacon deployment. We try to solve the following issues that researchers have not investigated yet.

(i) **Limited public Wi-Fi information on iPhones:** Regarding currently-connected Wi-Fi AP information, it can be obtained easily on Android smartphones via Google public API but very difficult on iPhones due to newest Apple policy [10], [11]. To exploit nearby Wi-Fi network information on iOS when no public API to access, we build a conversion function to classify Wi-Fi proximity zones between the iPhone and the currently-connected AP. To our best acknowledge, we are the first ones to try and tackle this issue under current iOS circumstances, where we learn how to build localization design specific for iPhones to obtain precise locations in indoor spaces.

(ii) **Compatibility issues with current Wi-Fi infrastructure:** In the aforementioned studies [6]–[9], they assume that the Wi-Fi AP density must cover the entire indoor building to ensure that the collected fingerprint database is reliable to be used in the online phase. If the AP density is low as Fig. 1 illustrates, these studies produce a huge errors in the calculation of the user's location. This is because at the locations that are not covered by Wi-Fi (wall corner, side corridor), only the PDR is deployed. Since the PDR is only effective over a short distance, accumulated errors become larger as the walking distance increases. Either replacing or associating with Wi-Fi, alternative technologies such as BLE should have corresponding deployment costs that are equivalent to currently existing Wi-Fi infrastructure, while are able to support tracking via the PDR over long walking path.

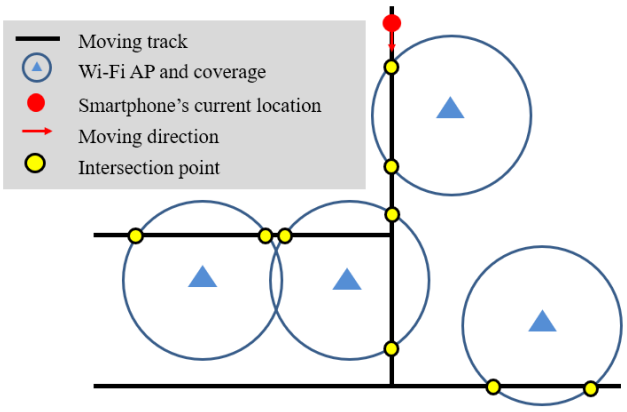


FIGURE 1. An example of low Wi-Fi AP density scenario.

(iii) **Improved PDR:** We observe that the localization accuracy can be greatly improved by reducing the

IMU bias. For instance, instead of using default services that report movement detected by the smartphone's IMUs, we can derive displacement and direction information then applying correction techniques to obtain more robust IMU data. Moreover, with a given floor plan along with Wi-Fi AP/iBeacon placement map, we add a new module to estimate a starting point for the tracking path.

(iv) **Joint location estimation:** In order to handle mobile trajectories, we need to design a localization engine that jointly estimates user's location, while pursuing better accuracy by combining the hybrid signal inputs. Thus, we employ a set of weighted factors of Wi-Fi/iBeacon which iteratively updates depending on how strong the received signals are.

Finally, in order to evaluate the proposed tracking scheme we build an iOS app and conduct several experiments in a typical university building.

The organization of this paper is as follows. Section II presents a background on indoor infrastructure such as PDR, Wi-Fi and iBeacon used in this paper. Section III introduces the related works on hybrid indoor localization algorithms on smartphones, the current issues and the motivation of our paper. In Section IV, we present the proposed system including a function of Wi-Fi proximity conversion, iBeacon measurements and deployment methodology, an improved smartphone-based PDR and a hybrid localization engine. Section V introduces device information and some extensive experiment results that have been carried out in a real indoor environments. Finally, Section VI concludes the paper and provides some future research directions.

Notations: The following notations are used throughout the paper. Matrix and vectors are denoted in bold faces, e.g., matrix $\mathbf{A}$ or vector $\mathbf{a}$. $\mathbf{I}_n$ denotes $n \times n$ square identity matrix. The transpose operator is denoted as $(\cdot)^T$. The distance between two given sets $\mathcal{C}$ and $\mathcal{D}$ is defined as $\text{dist}(\mathcal{C}, \mathcal{D}) = \inf\{||c-d|| \mid c \in \mathcal{C}, d \in \mathcal{D}\}$. The circle with center point $\mathbf{a}$ and radius r is denoted as $C(\mathbf{a}, r)$. We denote g as the acceleration of gravity which has the value of 9.8 m/s^2 on Earth. A random variable X follows the normal distribution with mean μ and variance σ^2, we write $X \sim \mathcal{N}(\mu, \sigma^2)$. The operation of taking the cardinality of a given set is denoted as $\#\{\cdot\}$. The line between two points $\mathbf{A}$ and $\mathbf{B}$ are denoted by $\overrightarrow{\mathbf{AB}}$. $\mathcal{S} \cap \mathcal{T}$ denotes the intersection between two sets $\mathcal{S}$ and $\mathcal{T}$. The big-O notation $\mathcal{O}(\cdot)$ denotes the algorithmic complexity. The floor function of an input x is denoted by $\lfloor x \rfloor$, which gives the greatest integer less than or equal to x.

II. PRELIMINARIES AND MEASUREMENTS

Before going on to the subsequent sections, we aim in this section to help readers to understand the core techniques (e.g., Wi-Fi, BLE, IMU) on building indoor localization system. First, we provide technical background in collecting data on pedestrians' spatial movements towards smartphone inertial sensing and inputting to the PDR. We also present some literature on Wi-Fi infrastructure and typical rules on radio map construction to enable a ubiquitous fingerprint-based solution for practical applications. Regarding BLE beacon capability, we describe iBeacon protocols and how to implement a beacon by an application running on a mobile device.

A. PEDESTRIAN DEAD RECKONING

1) IMU COMPONENTS AND GETTING IMU ON SMARTPHONES

In order to measure the mobility for smartphones, IMU is primarily designed for reading motion data. A typical smartphone-based IMU consists of several sensors such as accelerometer, gyroscope and magnetometer which enable for real-time motion detection, indoor localization and activity recognition. Generally, each mobile platform (i.e., iOS, Android) has different framework support for programming and accessing IMUs data from these sensors. For instance, the Android platform provides motion data through **SensorEvent** framework [25], while in iOS devices the **CoreMotion** framework [26] reports and process the IMU and the environment-related events. Each IMU has different characteristics according its translational function as we presented in Table 1.

TABLE 1. Smartphone IMU sensors.

Sensor type	Function
Accelerometer	Measures the acceleration of a motion in meters per second squared (m/s^2). To offer robust motion, it usually performs in three-dimensional information (e.g., X-, Y-, Z-axis acceleration) along with other sensors.
Gyroscope	Measures the orientation of an angular momentum. The smartphone can be rotated about three orthogonal axes that create three rotation values which called Euler angles (yaw, pitch and roll). In order to calculate a rotation angle θ_t, an integration can be utilized as $\theta_t^b = \int_0^t \omega_t^b$, where ω_t^b is the quantity results from the sensor in local reference frame.
Magnetometer	Measures the strength and the direction of magnetic fields.

2) SMARTPHONE ORIENTATION

Since the IMU data is measured in local coordinate system (LCS), i.e., smartphone body frame, in order to obtain the true value in global coordinate system (GCS) we need to apply transformations from LCS to GCS according to the phone orientation. For instance, for given a measured gyroscope data in local frame $\mathbf{g}_{LCS} = [g_x, g_y, g_z]^T$, the value in global frame $\mathbf{g}_{GCS}$ is given by [21].

$$\mathbf{g}_{GCS} = \mathbf{O}_T \mathbf{g}_{LCS}, \tag{1}$$

$$\mathbf{O}_T = \mathbf{O}_x \mathbf{O}_y \mathbf{O}_z, \tag{2}$$

where $\mathbf{O}_x$, $\mathbf{O}_y$ and $\mathbf{O}_z$ are rotation matrices along three frame axes. Here, we present

$$\mathbf{O}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -\cos(g_x) & \sin(g_x) \\ 0 & \sin(g_x) & \cos(g_x) \end{bmatrix}, \tag{3}$$

$$\mathbf{O}_y = \begin{bmatrix} \cos(g_y) & 0 & \sin(g_y) \\ 0 & 1 & 0 \\ -\sin(g_y) & 0 & \cos(g_y) \end{bmatrix}, \quad (4)$$

$$\mathbf{O}_z = \begin{bmatrix} \cos(g_z) & \sin(g_z) & 0 \\ -\sin(g_z) & \cos(g_z) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (5)$$

Similarly, the local acceleration vector $\mathbf{a}_{LCS}$ is converted to the global value $\mathbf{a}_{GCS}$ by subtract the Earth's gravity vector.

$$\mathbf{a}_{GCS} = \mathbf{a}_{LCS} - \mathbf{G}, \quad (6)$$

where $\mathbf{G} = [0\ 0\ g]^T$. To reduce noise effects, an averaging filter can be applied on $\mathbf{a}_{GCS}$ as the authors in [21] suggested. Henceforth, we refer the IMU data as the one in the GCS for further processing.

3) SMARTPHONE DEAD RECKONING INFRASTRUCTURE

PDR is a popular tracking method which offers low complexity and supports user mobility monitoring. With the input data about step, stride, heading direction and other related human mobility information from the IMU, the PDR elaborates them to derive the user position $\mathbf{P}_k(x_k, y_k)$ at each time t_k as

$$\begin{cases} x_k = x_0 + \sum_{m=1}^{k} \lambda_m \cos\alpha_m \\ y_k = y_0 + \sum_{m=1}^{k} \lambda_m \sin\alpha_m. \end{cases} \quad (7)$$

Here, $\mathbf{P}_0(x_0, y_0)$ is the starting location of the track, λ_k and α_k are the corresponding estimated step length and the orientation angle obtained from the IMUs, respectively.

Since the PDR achieves a good accuracy within a short walking distance, various works [7]–[9], [12]–[19], [27] have tried to improve tracking accuracy along long trajectories, which may fall into following categories.

- **Enhance step related data:** To detect a step, any step detection scheme relies on the fact that every step generates a notable impulse in the notable impulse reading along horizontal plane. By capturing such impulse, the authors in [7], [8] defined different acceleration thresholds for step detection. In particular, i) a step is recorded if the magnitude of the corresponding acceleration is larger than the predefined thresholds. Meanwhile, the works [9], [17] defined a step must satisfy (i) and the interval between two adjacent peaks/valleys must be greater than the given threshold value. Concerning step length estimation, there are two approaches to determine the step length: deterministic and dynamic. In the deterministic approach, the step length is fixed according to user height and gender [12], while in the dynamic approach the step length is calculated based on the acceleration readings [13]–[16], [18], [19].
- **Correct heading direction estimation:** To determine the heading direction, many conventional works [18], [19] used the inertial compass to measure the angle between the smartphone and the North of the Earth. However, if the user walks into an indoor building with various structures along the path, there exist inevitable noises integrated with the heading measurements of the walk. To mitigate the influence of such noises, several techniques [17], [27] proposed noise reduction methods for removing noises from the heading direction signals.
- **Provide initial location of tracking path:** To avoid accumulative errors from the start, many studies [7]–[9] assumed the starting location of the track is known in advance. In practice, this assumption is impractical because the user often does not aware his location in the building. Only few works set this point as (i) the first Wi-Fi AP that the smartphone connects with [18], [19], (ii) the strongest iBeacon that the smartphones captures on its scanning list [24], [28], or (iii) a fusion location by weighting factors between (i) and (ii) [21]–[23].

B. WI-FI AP PROCESSING

Conventional Wi-Fi fingerprinting-based localization techniques utilize an RSS profile for location estimation, which typically consists of two phases: offline phase and online phase. During the offline phase, RSSI fingerprint database is collected at several reference points (RPs). For L Wi-Fi APs and N RPs, we denoted the RSSI database as $\{\mathbf{p}_i, \boldsymbol{\psi}_i(1), \cdots, \boldsymbol{\psi}_i(T) \mid i = 1, \cdots, N\}$, where $\mathbf{p}_i$ is the i-th RP coordinates, $\boldsymbol{\psi}_i = [\boldsymbol{\psi}_{i,j}(t)]$ is the RSSI reading vector from the j-th Wi-Fi AP at the i-th RP at time t, T is the number of sample numbers. During the online phase, a smartphone first measures RSSI between detectable APs within its communication range and forms its own RSSI fingerprint $\mathbf{R} = [R_1, \cdots, R_L]$. The Euclidean distance between the online RSSI vector and the stored RSSI in the database is calculated as

$$D(i) = ||\mathbf{R} - \bar{\boldsymbol{\psi}}_i||, \quad (8)$$

where $\bar{\boldsymbol{\psi}}_i = \sum_{\tau=1}^{T} \psi_{i,j}(\tau)$. To compensate for RSSI bias among different devices when it comes online, RSSI deduction [2] can be applied. In particular, let D_{uv} be the Euclidean distance between the two fingerprints collected at two RPs $\mathbf{R}_u = [R_{u1}, \cdots, R_{uL}]^T$ and $\mathbf{R}_v = [R_{v1}, \cdots, R_{vM}]^T$, which is defined as

$$D_{uv} = \sqrt{\sum_{i=1}^{m} |r_{ui} - r_{vi}|^2}. \quad (9)$$

Concerning heterogeneity issue, the RSSIs between two devices A and B can be written as $RSSI_B = \alpha_1\ RSSI_A + \alpha_2$ where α_1 closes to 1 and α_2 is an adjust number. Thus, the optimal RSSI offset is calculated by [1].

$$\mathcal{O}_{uv} = \arg\min_{\alpha_2} D_{uv}, \quad (10)$$

$$D_{uv} = \sqrt{\sum_{i=1}^{m} |r_{ui} - r_{vi} + \mathcal{O}_{uv}|^2}. \quad (11)$$

There are several approaches to utilizing RSSI database to calculate the location such as deterministic methods (e.g. K-NN [29]) or probabilistic methods (e.g., histogram [30], kernel-based [31]). The famous K-nearest neighbors (KNN) method uses weights of K closest RPs to obtain the estimate location $\hat{\mathbf{p}}$ as

$$\bar{\mathbf{p}} = \frac{1}{\sum_{i=1}^{K} \frac{1}{D_i}} \sum_{i=1}^{K} \frac{1}{D_i} \mathbf{p}_i. \tag{12}$$

Generally, fingerprinting technique can obtain several levels of average localization errors from meter to few meters, which depends on the quality of the fingerprinting database. For instance, if there are more number of collected RPs, the resolution of radio map becomes finer, thus allow a better location estimation. However, maintenance of such database and update of RSSI propagation environments over time and different devices cause a big disadvantage of the techniques.

C. iBeacon BACKGROUND

iBeacon is a transmit-only BLE beacon testing under an Apple Inc.'s licensing program. It provides proximity information which indicates the closeness level between two devices. As illustrated in Fig. 2, an iBeacon advertising packet consists of three numerical identifiers: universal unique identifier, major number and minor number. A proximity estimation measures RSSI of a frame at the receiver.

Manufacturer information					iBeacon contents			
Flags	Length	Type	Company ID	Beacon type	Proximity UUID	Major number	Minor number	Measured Power

FIGURE 2. Typical iBeacon advertising packet.

After importing **CoreLocation** [32] and **CoreBluetooth** frameworks [33] into project, it scans nearby iBeacons and obtains beacon distances from the smartphone in meters. The iBeacon major and minor numbers are retrieved from **CLBeacon** object. In general, iOS does not reveal the transmission power of an iBeacon. If the transmission power of iBeacon is known, it can be useful for determining how far away the iBeacon from the iPhone is. Alternatively, an iBeacon's proximity can be approximated by using **CLBeacon.proximity** property. In particular, it calculates the distance from an iPhone to an iBeacon using the RSSI reported at this distance as

$$d \approx 10^{(RSSI_0 - RSSI)/10\eta}. \tag{13}$$

Here, $RSSI$ (dBm) and $RSSI_0$ (dBm) are the reading RSSI at distance d (m) and reference distance $d_0 = 1$ (m), respectively. η is the path loss exponent, typically between 2.7 and 4.3. Due to radio interference and absorption affects from surrounding objects, the default proximity (13) may not be accurate.

III. RELATED WORKS AND MOTIVATION

Several works [21]–[24] adopted fusion algorithms that combine PDR and Wi-Fi/iBeacon to achieve consistently precise localization. In general, given an indoor map which consists of Wi-Fi AP/iBeacon locations and their corresponding offline training database, the key ideas of the fusion algorithms can be summarized as follows.

(i) *Initialization:* In practice, the starting location $\mathbf{P}_0$ of the tracking path is often unknown (i.e., the user does not know where he/she is in the indoor building). We denote N_{Dev} as the total number of detectable Wi-Fi APs and iBeacons, the location of $\mathbf{P}_0(x_0, y_0)$ is estimated by combining Wi-Fi/iBeacon as

$$\mathbf{P}_0(x_0, y_0) = \sum_{i=1}^{N_{Dev}} \beta_i \mathbf{a}_i, \tag{14}$$

where $\mathbf{a}_i$ is the locations of the available Wi-Fi APs and iBeacons on the smartphone scanning list, and β_i indicate the weight factors indicating how strong the received signals are.

(ii) *Location updates:* At each step k, we denote λ_k as the user's step length and α_k as the heading direction angle, which are obtained from the IMU measurements. The position of the smartphone $\mathbf{P}_k(x_k, y_k)$ is updated by the PDR as

$$\mathbf{P}_k^{pdr} = \mathbf{P}_{k-1}^{pdr} + \lambda_k [\sin\alpha_k \ \cos\alpha_k]^T. \tag{15}$$

(iii) *Position calibration:* Particle filter (PF) is a common approach to reduce the cumulative errors in the PDR. With N_{PF} particles, each particle $\mathbf{x}_k^i = [x_k, y_i]$ around $\mathbf{P}_k(x_k, y_k)$ is predicted according to the movement state and the observed heading direction as

$$\begin{cases} x_k^i = x_{k-1}^i + \lambda_k \sin(\alpha_k) \\ y_k^i = y_{k-1}^i + \lambda_k \cos(\alpha_k). \end{cases} \tag{16}$$

Thus, the smartphone location is re-calculated based on the particles set as

$$\begin{cases} x_k = \sum_{i=1}^{N_{PF}} \bar{w}_k^i x_k^i \\ y_k = \sum_{i=1}^{N_{PF}} \bar{w}_k^i y_k^i. \end{cases} \tag{17}$$

Here, the weights of particles are calculated as $w_k^i = w_{k-1}^i \times p(x_k^i | m_{ML})$, then normalized as $\bar{w}_k^i = w_k^i / \sum_{j=1}^{N_{PF}} w_k^j$, where m_{ML} is maximum a posterior estimation. We note that the localization accuracy depends on particle selection. That is, if the number of particles is not enough to accurately represent the distribution, the PF is depleted to track user trajectory. Meanwhile, increasing the number of particles N_{PF} may increase computational cost and delay on calculating user location.

Unfortunately, the following problems have not been solved yet in the aforementioned works.

1) LIMITED PUBLIC WI-FI INFORMATION ON iOS DEVICES

To retrieve Wi-Fi information in iOS devices, an iOS app requests the user's authorization and enables the access of Wi-Fi information capability. In particular, it uses the **CNCopyCurrentNetworkInfo** API to configure the current Wi-Fi network for a given network interface. However, according to Apple Inc.'s newest update, calling this function returns NULL since iOS 12 or later. From iOS 13 updates, Apple Inc. is tighter on providing public application programming interface since they no longer return valid Wi-Fi service set identifier (SSID) and basic service set identifier information (BSSID). For example, the requesting app is only able to access the signal strength information of a single Wi-Fi AP in status bar instead of raw RSSI in dBm [10]. Thus, the PDRSC scheme [18] and its enhanced version [19] cannot function well without raw RSSI readings from the currently-connected Wi-Fi AP. Alternatively, to perform localization task on iPhone it must be done with a single Wi-Fi AP so that using Wi-Fi information and extracting them are very limited. As a result, iPhones-based Wi-Fi/PDR should be combined with other alternative techniques to improve indoor localization performance. In fact, if we can resolve the tracking problem under iOS limitations, we can get better results on the Android smartphones because they provide more information (e.g., Wi-Fi scan list, BLE channel selection, etc.) compared to the iOS counterparts.

2) BLE DEPLOYMENT METHODOLOGY

The applicability of any hybrid Wi-Fi/PDR scheme may fail in the indoor environments due to low Wi-Fi AP density. This problem can happen in practice because the Wi-Fi APs may be pre-established in the AoI before designing a tracking application. A low Wi-Fi AP density may contain two scenarios: (a) APs do not cover all the AoI. (b) Their cover ranges do not overlap enough to execute the tracking purpose. Towards applications of associating IoT, BLE technology enables low power wireless communication that can be used over a short distance. To integrate BLE to smartphone-based localization system, it can be installed in one of two ways:

- **Co-exist with Wi-Fi AP:** Since both Wi-Fi and BLE techniques share the 2.4 GHz spectrum, there exists interference among these devices. In order to reduce such interference, the BLE utilizes frequency hopping spread spectrum [20]. By deploying BLE beacons beside Wi-Fi APs, it helps to cover all locations that Wi-Fi is not accessible (e.g., corridors, office rooms, staircases, etc.). At this point, the problem is how to design the tracking algorithm using the aggregated information from IMU, BLE and Wi-Fi signals.
- **BLE beacons only:** If only BLE beacon are deployed, the tracking app needs to able to control the information flow from multiple beacons and gather BLE readings for localization purpose. In this case, the BLE deployment must satisfy following two requirements. First, the number of BLE beacons must be sufficient to reach currently-existing Wi-Fi coverage. Second, BLE beacons must be placed in separate areas to avoid interference with other nearby iBeacons.

Depending on the applications, we need to choose whether deploying BLE with or without Wi-Fi in order to balance the trade-off between the localization accuracy and the deployment cost.

3) IMU INPUTS AND IMPROVED PDR OVER LONG TRAJECTORIES

In order to achieve high localization accuracy over long trajectories via PDR, we must prevent the biases from the IMU components. Instead of using the default hardware-generated data, we need to modify on the PDR inputs according to raw IMU readings on devices. In particular, each component of the IMU data (e.g., step detection, step length estimation and heading direction estimation) needs to be precisely quantified before inputting to the PDR. Thus, in this work we aim to perform self-tracking with current smartphone constraints, while utilizing an adjustment method to reduce the misalignment between the desired path and the estimated one.

IV. PROPOSED HYBRID PDR/iBeacon LOCALIZATION AND TRACKING

Different from aforementioned works [7]–[9], [12]–[19], [21]–[24], [27], we identify several practical issues when applying PDR and Wi-Fi/iBeacon for mobile smartphones, especially for iPhones. In particular, we resolve the traditional Wi-Fi/iBeacon fingerprint problem in Section II-B without time-consuming and labor-intensive site survey, which has not been considered in the aforementioned works. On the other hand, the iBeacon has some attractive features which can be designed to support self-localization on the smartphones. More importantly, the beacons should be organized to accomplish their tasks without human supervision. Despite the fact that several works have tried to improve the PDR tracking accuracy along trajectories, there are still many aspects that have not been fully exploited (e.g. remove bias from the IMU components). Aiming at a simple and easy adaptive tracking system, we follow a typical Wi-Fi/iBeacon infrastructure with little efforts and try to incorporate with an improved PDR version to make it a promising hybrid scheme for practical applications.

In this section, we present an enery-aware hybrid tracking scheme based on PDR and Wi-Fi/iBeacon, which satisfies the following properties. First, it is able to extract AP information when scanning nearby Wi-Fi, thus fills the gap between the Android and the iOS platforms. Instead of maintaining a radio map, we only use Wi-Fi proximity range and Wi-Fi AP location map for estimating the user location. Second, it can work in general office buildings without changing current existing Wi-Fi deployment over the area. Moreover, in order to install and find an optimal placement for iBeacons, general rules on iBeacon deployment spacing must be established. During a training phase, RSSI readings from different iBeacons are collected by the smartphones at known positions (e.g., reference points) to determine an appropriate iBeacon deployment spacing and to classify

iBeacon proximity zones based on distance between the smartphone and the iBeacon. Third, it provides a good accuracy with current Wi-Fi/BLE constraints when performing real-time tracking on the smartphones. The IMU readings provide the heading of the mobile smartphone and avoid the complexity of radio map maintenance, while Wi-Fi/iBeacon readings choose the relevant regions that the smartphone is inside and fix its location during moving. In order to evaluate the efficiency of the proposed tracking system, we built an iOS app, deployed iBeacon beacons and conducted several experiments in a typical office building.

A. WI-FI INFRASTRUCTURE

Regarding Android devices, an app requests a **starScan()** command in **WifiManager** API [34] to scan nearby Wi-Fi AP signals. Once the scan has completed, the app can obtain the scanning results by commanding a **getScanResults()**. It also can return the RSSI in dBm. In order to convert a real-time RSSI value to a related distance D, we build a mapping between them as

$$RSSI_{in} = RSSI_{reg} + RSSI_{offset}, \tag{18}$$

$$D = 10^{-\frac{RSSI_{in}+RSSI_0}{\beta}}. \tag{19}$$

Here, $RSSI_0$ is the RSSI at a reference distance $d_0 = 1$m, β is the path loss exponent (typically between 2.7 and 4.3.) and $RSSI_{offset}$ is a constant bias value that reflects device dependency.

On the other hand, in order to retrieve public Wi-Fi AP information, iPhone needs to turn on the function "*Auto connecting Wi-Fi without asking*" in the "*Settings*". According to the newest documents that summarize current issues with Wi-Fi APIs available on iOS [10], [11], an iOS app only receives a specific Wi-Fi AP information (i.e., currently-connected Wi-Fi AP) instead of the user's list of all available Wi-Fi APs in the area. Besides, it no longer obtains RSSI measurements of currently-connected AP since iOS 13 updates. In fact, the module **CNCopyCurrentNetworkInfo** was used to return the information about the currently-connected AP for a given network interface. However, with the latest iOS Apple Inc. has changed permission to access location, where the public API does not bring back valid Wi-Fi SSID and BSSID information. Instead, the information is returned by a Boolean value, which indicates Wi-Fi status bar. Using this value, we classify proximity zones into four classes as Table 2. We would like to note that an iPhone carried by the user collects Wi-Fi AP status at various locations along the moving path, and converted distances are also recorded. The distance D may vary depending on Wi-Fi AP installed in the AoI.

In our approach, an iPhone only detects its proximity zone with Wi-Fi AP based on its connected status value. The advantage of Table 2 is simple implementation since there is no computation involved in. Scanning this value on iPhone is very quick, thus it is suitable for tracking a fast moving user.

TABLE 2. Classification of proximity zone based on distance between the user and the connected Wi-Fi AP.

Wi-Fi zone	Status value s	Distance D
Immediate	3	<5 m
Near	2	5-10 m
Far	1	10-15 m
Unknown	0	>15 m

One big disadvantage is that it is only sufficient for coarse localization stage due to limited accuracy.

B. iBeacon MEASUREMENTS AND DEPLOYMENT PRINCIPLES

We use a simple path loss model to describe the relationship between an RSSI sample readings $rssi_i$ and a related distance d_i as

$$rssi_i = rssi_0 - 10\beta \log\left(\frac{d_i}{d_0}\right) + X_\sigma, \tag{20}$$

where $rssi_0$ [dBm] is the reference RSSI at the distance $d_0 = 1$ [m]. In Fig. 3 shows real-time RSSI readings of a RedBear iBeacon measured by iPhone 7.

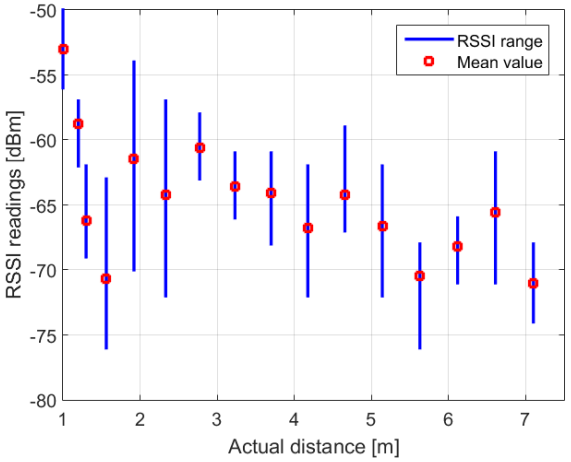


FIGURE 3. RSSI sample readings of a RedBear iBeacon.

In practice, the path loss (20) also depends on the transmit power Tx, thus the distance conversion can be given by

$$d_i \approx d_0 \exp\left(\frac{rssi_0 - rssi_i - Tx}{10\beta}\right). \tag{21}$$

In our previous work [28], we conducted several measurements for RedBear iBeacons. As a result, we classify iBeacon measurements into four zones as in Table 3.

TABLE 3. Classification of iBeacon proximity zones.

iBeacon zone	RSSI range [dBm]	Ground distance [m]	Confidence [%]
Immediate	(-65, -56)	<0.5	98.0
Near	(-76, -66)	0.5-1.5	95.0
Far	(-94, -77)	1.5- 4.5	83.7
Unknown	<-95	>4.5	78.0

This table is used to select the zones with similar RSSI readings, so that the localization system is able to quickly

look up a feasible region where the smartphone may belong to. The reasons we do not divide the zones greater than four are two-folds. First, it is easy to implement on mobile devices. Second, from our practical experiments we recognize that it is difficult to distinguish two neighbor RPs which have the same RSSI signals. This observation can be clearly seen in the below experiment. In order to overcome this problem, if the RSSI reading indicates that the current user's location is in the same zone as its previous one, it is still updated via the PDR until the user walks into a different signal zone.

In order to find an appropriate iBeacon deployment spacing, we performed several experiments to evaluate iBeacon ranging. The experiment setting is illustrated in Fig. 4(a). Our idea is that an iPhone collects the sampled RSSI from nearby iBeacons. Suppose there are m iBeacons in a given area. For each RP, the RSS fingerprint at this location is denoted as $\mathbf{r} = [r_1, \cdots, r_m]^T$, where r_i is the RSSI of the i-th iBeacon. If the smartphone does not detect this iBeacon, we set $r_i = 0$. For given two fingerprints $\mathbf{r}_i$ and $\mathbf{r}_j$, we define the RSSI difference function between them as

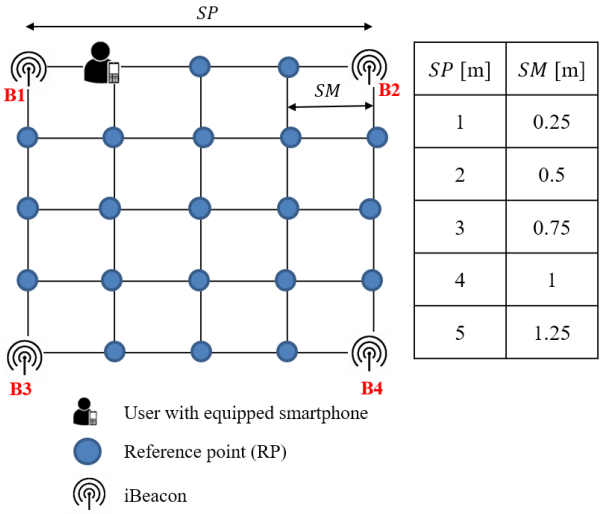
$$\delta_{ij}(\mathbf{r}_i, \mathbf{r}_j) = ||\mathbf{r}_i - \mathbf{r}_j||_1 = \sum_{k=1}^{m} |r_{ik} - r_{jk}|. \tag{22}$$

If the RSSI difference δ_{ij} is smaller than a predefined threshold (i.e., $\delta_{ij} < \epsilon$), we treat $\mathbf{r}_i$ and $\mathbf{r}_j$ as a same point in the fingerprint space. Otherwise, we indicate them as two different points. After data collection, we set ϵ as the average maximum dissimilarity of fingerprints from distinct RPs. In order to test this scheme, we use four RedBear iBeacons (i.e., $m = 4$) that are operated under BLE 4.0 specification (details in Section V). At a static reference position, the iPhone 7 used as a BLE scanner measures real RSSI signals during two minutes. With each spacing SP, we define the distinguished fingerprint percentage $P_{df}(SP)$ as the percent that two RSSI samples are identified as different fingerprints. Mathematically, we write

$$P_{df}(SP) = \frac{\#\{\delta_{ij} : \delta_{ij} \geq \epsilon\}}{\text{Number of RPs}} \times 100\ (\%). \tag{23}$$

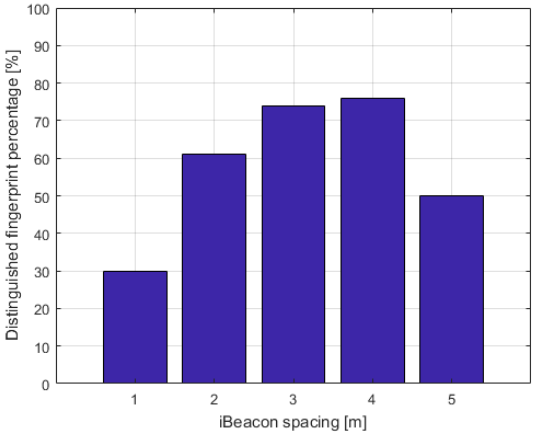
Fig. 4(b) shows the distinguished fingerprint percentage with different spacing when the number of RPs is 25. We observe that the spacing between two iBeacons should be set as $3 \sim 4$m to achieve a distinguished fingerprint percentage of 70% or more. From this result, we set the minimum spacing between two iBeacons as $SP_{\min} = 4$.

Regarding iBeacon deployment with current existing Wi-Fi infrastructure, if the Wi-Fi AP density in the area is low as in Fig. 1, the iBeacon positions are expected to cover the non-overlapping areas between any two adjacent Wi-Fi APs. We assume that the current Wi-Fi deployment over the area is not changeable. For each $\mathbf{AP}_i$, all intersection points (yellow points in Fig.1) form a set $\mathbf{S}_i = \{\mathbf{s}_{i1}, \cdots, \mathbf{s}_{im_i}\}$, where m_i is the corresponding number of intersection points, i.e., $m_i = \#\{\mathbf{S}_i\}$. The iBeacon deployment principles can be mainly summarized by two following points:



SP [m]	SM [m]
1	0.25
2	0.5
3	0.75
4	1
5	1.25

(a) Experiment setting: Number of RPs = 25.



(b) Distinguished fingerprint percentage versus iBeacon spacing.

FIGURE 4. iBeacon spacing experiment and result.

(**P1**) Ensuring appropriate coverage for the area between two adjacent Wi-Fi APs: Let $\kappa_i = \text{dist}(\mathbf{S}_i, \mathbf{S}_{i+1})$, the corresponding closest pair of points between the two sets are denoted as $\mathbf{u} \in \mathbf{S}_i$ and $\mathbf{v} \in \mathbf{S}_{i+1}$. If $\kappa_i \geq SP_{\min}$, we put an iBeacon $\mathbf{a}$ between $\mathbf{u}$ and $\mathbf{v}$ satisfying $\frac{SP_{\min}}{2}$ away from $\mathbf{u}$ along the line through $\mathbf{u}$ and $\mathbf{v}$.

$$\mathbf{a} = \mathbf{u} + \frac{SP_{\min}}{2}(\mathbf{u} - \mathbf{v}). \tag{24}$$

(**P2**) Check iBeacon coverage consistency: Let $\mathbf{C} = C(\mathbf{p}, \frac{SP_{\min}}{2})$, we recalculate $\kappa_i = \text{dist}(\mathbf{C}, \mathbf{S}_{i+1})$. If $\kappa_i \leq 4$, it means that the last iBeacon location has been covered enough. Otherwise, we repeat (**P1**) with $\mathbf{C}$ and $\mathbf{S}_{i+1}$ until the area between two intersection points is filled up from the installed iBeacons.

C. IMPROVED SMARTPHONE-BASED PDR

After installing our localization app, iPhone collects the readings of IMU sensors. With a simple Xcode project, we implement IMU data collection for iPhone 7. A sample of the IMU can be found in our previous paper [19]. We choose to pursue better accuracy of IMU inputs to the PDR by modifying step

detection, step length estimation and heading direction. After a step is recognized, it will be added each time to the user walks. Denoting the total number of steps as n_k, the distance of walking path is calculated as follows.

$$d_{pdr} = \sum_{i=1}^{n_k} \lambda_i. \tag{25}$$

The procedure of step detection is described as follows. A step is recorded if it satisfies the following two rules: (i) The corresponding acceleration measurements form a peak followed by a valley. The peak acceleration value a_t must be higher than an upper bound threshold a_{upper} and the valley acceleration value must be lower than a lower threshold a_{lower}. Mathematically, we write

$$\{a_t \geq a_{upper}, a_{t+\omega} \leq a_{lower}, 0 < \omega < 1\}. \tag{26}$$

To find the best thresholds a_{upper} and a_{lower}, an user walks a certain number of steps (e.g., 10 steps) to learn the changes in vertical acceleration data. In practice, we set $a_{upper} = 1$ [m/s^2] and $a_{lower} = -1$ [m/s^2] under the settings in Section V. (ii) The time spacing between two adjacent peaks/valleys must be greater than a given threshold value. It is usually set as two third of the walking step period. For instance, assuming that a human typically takes two steps per second, we set this value around 0.33s in this paper.

Inspired by the step length estimation model proposed in [19] and by suggestions from [35], in order to address the relationship between walking frequency and velocity, we use the following regression model to estimate λ_i in (7) under the condition that no Wi-Fi AP/iBeacon was detected in the walking area.

$$\lambda = \pi_0 + \pi_1 \times f + \pi_2 \times |\bar{\mathbf{a}}_{\max}|. \tag{27}$$

Here, π_0, π_1 and π_2 are regression coefficients which can be obtained through training, $\bar{\mathbf{a}}_{\max}$ is the maximum acceleration magnitude after applying a threshold on the variance of acceleration over a sliding window φ. For instance, with the user holding an iPhone 7 in Section V, we obtain those parameters as $\pi_0 = 0.39, \pi_1 = -0.10$ and $\pi_2 = 0.02$, where the walking frequency follows a uniform distribution, i.e., $f \sim \mathcal{U}[1.8\ 2.2]$ Hz and the slide window $\varphi = 5$. Through several experiments in [19], we observe that either too low or too high acceleration sensitivity may cause large errors in step length estimation because the errors accumulate rapidly along the walking path. The step length estimation based on a linear model should be taken in priority due to its simplicity and efficient computation. It has been shown in [8] that when tracking a real-time user moving in a long path, a modified linear model as (27) is more stable and has lower estimation error compared to nonlinear ones in [13]–[15].

For iOS app development, the heading direction can be directly read from **CLHeading** class of **CoreLocation** framework [32]. To obtain the heading direction, there are two choices: gyroscope-based method (reading user's orientation change using the gyroscope) and compass-based method (using compass reading directly). As shown by the authors in [18], [19], compass-based method is stable in a long walk but easily affected by metal and conducting material in the area, while the gyroscope-based [36] method remains unaffected by magnetic fields but suffers from accumulated errors. In this section, we denote α_g and α_c as the gyroscope-based reading and the compass-based reading, respectively. The value of α is calculated by

$$\alpha = \alpha_c + \delta t \times \alpha_g + \delta_\alpha, \tag{28}$$

where δt is the time interval and δ_α is assumed to be a Gaussian error with zero mean and variance σ^2. To reduce the heading estimation bias, we use the best linear unbiased estimator (BLUE) [37] for the Cartesian coordinates of current heading. The idea here is to avoid the ambiguity problem associated with averaging angular measurements, while taking advantage of sample variances from the measurements collected in a certain window size. Mathematically, denoting M as the time window size of the heading measurements, we assume that the measured heading is $\hat{\alpha}_i = \alpha_i^{true} + \nu_i$, where α_i^{true} is the true heading and $\nu_i \sim \mathcal{N}(0, \sigma_i^2)$ is the measurement noise. Then, we let $\rho_{xi} = \sin\hat{\alpha}_i$ and $\rho_{yi} = \cos\hat{\alpha}_i$ for $i = 1, \cdots, M$, where M is the number of heading samples. In practice, we set $M = 5$, that is, only the last five heading samples will be used to correct the current heading direction. It is a recursive procedure with a given window size M. By setting the variances of $\boldsymbol{\rho}_x$ and $\boldsymbol{\rho}_x$ as weight factors, we perform the BLUEs for both $\boldsymbol{\rho}_x$ and $\boldsymbol{\rho}_y$ as follows.

$$\hat{\boldsymbol{\rho}}_x = \frac{1}{\sum_{i=1}^{M} \sigma_i^2} \sum_{i=1}^{M} \rho_{xi}/\sigma_i^2, \quad \hat{\boldsymbol{\rho}}_y = \frac{1}{\sum_{i=1}^{M} \sigma_i^2} \sum_{i=1}^{M} \rho_{yi}/\sigma_i^2. \tag{29}$$

Thus, a new heading estimation is given by

$$\hat{h}_i = \tan^{-1}\left(\frac{\hat{\rho}_{xi}}{\hat{\rho}_{yi}}\right). \tag{30}$$

D. HYBRID LOCALIZATION ENGINE

Our proposed hybrid PDR/iBeacon tracking system is divided into three components: iBeacon/Wi-Fi profile module, smartphone IMU module and a localization engine as shown in Fig. 5. After the app installed on user iPhone, it collects the readings from IMUs and associates with Wi-Fi/iBeacon readings to perform self-localization and tracking. We would like to take a few notes on the working flow.

1) INITIAL LOCATION ESTIMATION

In many studies [7]–[9], the starting location of the tracking path is assumed to be known, which is an impractical assumption. In [18], [19], we suggest $\mathbf{P}_0$ as the strongest Wi-Fi AP that connects to the smartphone. Under the iPhone's constraints, if the Wi-Fi status bar is at 1 (weak connection) or 2 (medium connection), the use of this Wi-Fi AP as the starting point of the track causes a huge error in localization accuracy. In this work, we set the initial point as the output of a fusion algorithm between iBeacon and Wi-Fi AP locations

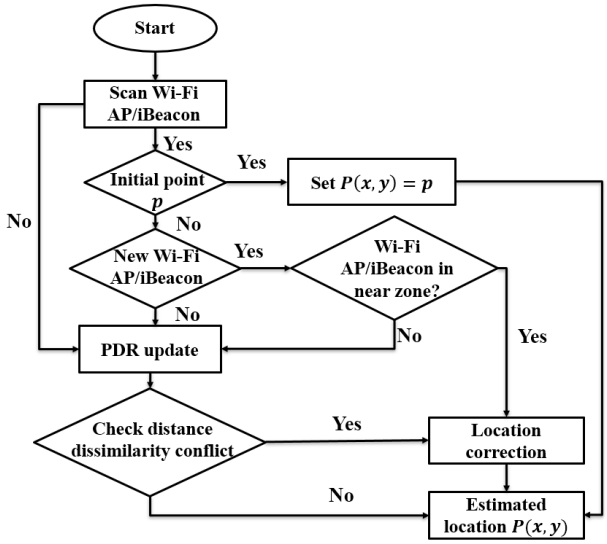


FIGURE 5. Working flow of the proposed hybrid localization scheme.

based on RSSIs. Let $\mathbf{AP}_1$ and $\mathbf{a}_1$ are the strongest Wi-Fi AP and the strongest iBeacon on the scanning list. Mathematically, we write

$$\mathbf{AP}_1 = \{\mathbf{A}_j : \ i = \arg\max_{j=1,\cdots,N_{AP}} RSSI_j\}. \quad (31)$$

$$\mathbf{a}_1 = \{\mathbf{a}_i : \ i = \arg\max_{i=1,\cdots,N_{iBeacon}} RSSI_i\}. \quad (32)$$

To obtain the starting point of the tracking path, we follow two simple rules:

(i) If either the status bar of the Wi-Fi AP $\mathbf{AP}_1$ is at $s = 3$ (strong connection) or the iBeacon $\mathbf{a}_1$ is in the near/immediate zone, we set $\mathbf{P}_0(x_0, y_0)$ at this Wi-Fi AP or this iBeacon.

(ii) Otherwise, we use a weighted centroid algorithm to determine $\mathbf{P}_0(x_0, y_0)$ as

$$\mathbf{P}_0(x_0, y_0) = \sum_{i=1}^{N_{iBeacon}} w_i \mathbf{a}_i + \sum_{j=1}^{N_{AP}} W_j \mathbf{AP}_j. \quad (33)$$

Here, $N_{iBeacon}$ and N_{AP} are the number of the available iBeacons $\{\mathbf{a}_i, i = 1, \cdots, N_{iBeacon}\}$ and Wi-Fi APs $\{\mathbf{AP}_j, j = 1, \cdots, N_{AP}\}$ which appear on the smartphone's scanning list. We note that, unlike the Android operating system, an iPhone can only get information of a single AP instead of all available APs in the area, i.e., $N_{AP} = 1$. The weight factors in (33) are assigned according to the strongness levels of Wi-Fi AP/iBeacon signals. Mathematically, we write

$$w_i = \frac{r_i - r_{\min}}{r_{\max} - r_{\min}}, \quad W_j = \frac{(1/D_j)}{\sum_{j=1}^{N_{AP}} (1/D_j)}. \quad (34)$$

The value r_i is the recorded RSSI at the i-th iBeacon, $r_{\min}$ and $r_{\min}$ are the minimum and the maximum iBeacon RSSI readings on the scanning list. The values of D_i are obtained from Table 2. Apparently, the proposed hybrid localization scheme requires no prior knowledge of starting point in the PDR.

2) CORRECTION MATCHING

Beside the use of the online RSSI readings to choose the relevant Wi-Fi AP/iBeacon positions, we can use a possible range of the user's current position. Using the fact that a normal person cannot walk far away within a short period of time, it is reasonable to limit a feasible region that the user steps in based on the PDR walking range. At step k, we denote $D^k_{AP_j}$, d^k_i and d^k_{pdr} as the corresponding Wi-Fi range, iBeacon range and the walking distance (25) via the PDR, respectively. We define the dissimilarities among them as

$$\Upsilon_k = \sum_{j=1}^{N_{AP}} W_i |D_{AP_j} - d^k_{pdr}|, \quad (35)$$

$$\upsilon_k = \sum_{i=1}^{N_{iBeacon}} w_i |d^k_i - d^k_{pdr}|. \quad (36)$$

We employ a calibration process that jointly estimates the smartphone's previous location and Wi-Fi/iBeacon gains according to a specific iBeacon placement method as follows.

- If $\mathbf{AP}_1$ is in immediate Wi-Fi zone or $\mathbf{a}_1$ is within iBeacon near zone, the smartphone takes the location of this Wi-Fi AP/iBeacon as its corrected location.
- If either $\Upsilon_k \leq \Delta_{th}$ or $\upsilon_k \leq \Delta_{th}$ (in practice, we set $\Delta_{th} = 3$m), the smartphone location keeps its current location estimation.
- If $\Upsilon_k > \Delta_{th}$ and $\mathbf{AP}_1$ is in near Wi-Fi zone, the smartphone location is corrected as the intersection of the circle made by $\mathbf{AP}_1$ with radius D_1 and the line segment made by $\mathbf{P}_{k-1}$ and $\mathbf{P}_k$ as

$$\mathbf{P}(x, y) = C(\mathbf{AP}_1, D_1) \bigcap \overrightarrow{\mathbf{P}_{k-1}\mathbf{P}_k}. \quad (37)$$

Otherwise, the smartphone location is updated according to $\mathbf{a}_1$ as

$$\mathbf{P}(x, y) = C(\mathbf{a}_1, d_1) \bigcap \overrightarrow{\mathbf{P}_{k-1}\mathbf{P}_k}. \quad (38)$$

Similarly, if only iBeacons are installed and $\upsilon_k > \Delta_{th}$, the smartphone location is calibrated as (38).

E. OPEN DISCUSSIONS

In this section, we discuss the system performance in several aspects such as complexity, power consumption analysis, and alternative approaches for designing localization algorithms.

Remark 1 (Computational Complexity): In each step k, the complexity of the proposed hybrid localization algorithm in Fig. 5 is divided into two parts: (i) sort the available Wi-Fi APs and iBeacons in the area according to their RSSI readings, (ii) check the distance dissimilarity conflicts and adjust the estimated location according to the correction matching procedure in Section IV-D2. The first part (i) requires a complexity of $\mathcal{O}(\max\{N_{AP}, N_{iBeacon}\})$. The procedure of calculating the dissimilarities (35) and (36) has a linear complexity in terms of N_{AP} and $N_{iBeacon}$, thus it can be easily implemented in real-time applications. A worst-case includes the computation from finding an intersection of two objects $C(\mathbf{AP}_1, D_1)$ (or $C(\mathbf{a}_1, d_1)$) and $\overrightarrow{\mathbf{P}_{k-1}\mathbf{P}_k}$ that takes $\mathcal{O}(1)$. Afterwards, the corresponding complexity of the second part (ii)

is $\mathcal{O}(\max\{N_{AP}, N_{iBeacon}\})$. Since mobile iPhone can obtain the information from a single currently-connected AP only, instead of a full list of currently-available APs in the AoI (i.e., $N_{AP} = 1$), the total complexity of hybrid localization at k-th step is $\mathcal{O}(\max\{1, N_{iBeacon}\})$.

Remark 2 (Deployment Cost and Power Consumption Analyses): As we have introduced in Section I, since Wi-Fi deployment becomes popular in indoor environments and off-the-shelf accessible on smartphone, the total deployment cost of the hybrid system mostly depends on the total number of iBeacons, $TN_{iBeacon}$, which has been used in the AoI. For a single iBeacon with average price c, we can expect to pay around $c \times TN_{iBeacon}$. In order to choose the right number of $TN_{iBeacon}$, we must consider the app features as well as the acquired localization accuracy. Generally, iBeacon installation layout varies across different indoor environments. Recently, best practice guidelines [38], [39] show that BLE positions should be installed above user head height to avoid interference from human body (up to 1.5m) and above 2.5m from floor level. Assuming that $SP_{\min} = 4$m and $SP_{\max} = 10$m, a rough number of $TN_{iBeacon}$ can be approximated as

$$\left\lfloor \frac{S(\text{AoI})}{SP_{\min}^2} \right\rfloor \leq TN_{iBeacon} \leq \left\lfloor \frac{S(\text{AoI})}{SP_{\max}^2} \right\rfloor, \tag{39}$$

where $S(\text{AoI})$ is the total area of the AoI.

Regarding power consumption, since the whole system is running on the smartphone, it depends on how much power has been consumed in each Wi-Fi/iBeacon scanning, which consists of three main steps: register a broadcast listener, request a scan and get scan results. For a certain scanning activity, the average power for each Wi-Fi/iBeacon scanning depends on the scanning intervals and its corresponding current. According to [40], an approximation for the power consumption of scanning activity is given by

$$PC = \frac{1}{\text{Period}} \sum_i (Intv_i \times I_i), \tag{40}$$

where $Intv_i$ is the time interval with its corresponding current I_i, respectively. For our experiments, given Wi-Fi scanning interval of $100 \sim 300$ ms, average current consumption[1] is around $100 \sim 120$mA for continuously Wi-Fi scanning and $25 \sim 47$mA for iBeacon scanning in one hour recording. For an iPhone 7 (i.e., battery capacity of 1960mAh), our localization app consumes about $5.1 \sim 6.12\%$ of battery for Wi-Fi scanning and $1.27 \sim 2.4\%$ per hour for iBeacon scanning. We note that other apps have been disabled from running in the background until the scanning schedule finished.

Remark 3 (Alternative Design Choices to Improve Tracking Performance): First, in this paper we more focus on client-based localization approach, i.e., the smartphone carried by a user collects the RSSI measurements of detectable Wi-Fi APs and iBeacons for self-localization. Alternatively, the computation workload handled by the mobile device should be reasonable to give real-time updates for the user. Existing approaches such as particle filters (PFs) [5], [12], Kalman filters (KFs) and their variants [9], [13], [17] offer high resolution accuracy, but require more expensive computation, power consumption and proper planning strategy. For example, the smartphone location is updated according to PF weights as (16), where the PF weights indicate the signal variability from nearby Wi-Fi APs or iBeacons. Regarding Table 2, there are not much changes in the PF weights within a certain Wi-Fi zone because the particle locations have equal weights for the same observation window. Thus, it further causes time delay in computing outputs' weighted sum when tracking the user in real-time. On the other hand, the application of the KFs and their variants need to be learned from experiments to set the optimal parameters on each step based on indoor map information. Second, the client-based approaches are flexible when it comes to optimization strategy and can adapt quickly to user's behavior. Those approaches have their limitations and in some cases they need to adopt a server-side to engage integrated information such as clustering trajectories for tracking multiple users and combining with context landmarks. These opportunistic suggestions are suitable for server-based approaches rather than client-based ones, that leads to other research directions. In summary, depending on target application, we may incorporate our current proposed scheme with other enhanced techniques, which we consider as our future work.

[1] Power consumption measured in mW is the operating voltage multiplied by the current, but it is more difficult when using discharge batteries and the voltage changes overtime and loading conditions. In terms of low power application, current consumption measured in mA is often used.

V. EXPERIMENTAL RESULTS

In this section, we present how we collect data and conduct several practical experiments to validate our proposed tracking system. To accomplish the above tasks, we build an iOS app to implement three main modules in Section IV. Moreover, a graphic user interface is designed to visualize the tracking results.

A. ASSUMPTIONS

Before going through the experiment results, we make the following three assumptions.

First, the starting point of the tracking path is not given. That means, before the user starts using our localization app, he does not know where he is in the testing area. According to Section IV-D, the starting point of the tracking path is set as a fusion point of nearby Wi-Fi APs and iBeacons to which smartphone detects. If he cannot find any Wi-Fi AP/iBeacon on the scanning list, the app forces him to walk around until he has found the nearest one. For example, Fig. 6 illustrates how the starting point has been selected. The actual user location is not necessary near or underneath Wi-Fi AP/iBeacon location. The first estimated point of the tracking path (e.g., red point) is only recorded by (33) when the smartphone connects to any nearby Wi-Fi AP/iBeacon.

Second, the user may show complex human mobility information. For example, he may walk at different speeds or motion patterns even on the same path. At each time, he holds

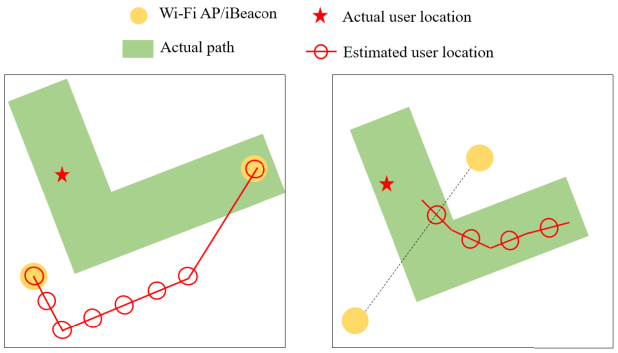


FIGURE 6. Illustration of the estimated starting point.

his smartphone in hand and naturally walks along the path during the data collection.

Third, the accuracy rate is leveraged after few times the user finishes the desired path. In our experiments, the recorded data are collected at three different times (morning, afternoon, night) during a day. Thus, there exist inevitable noises caused by other occasionally pedestrians or various obstacles along the path. We collect the statistics of localization results under those conditions. Concerning the design of walking path, we only have rights to access certain public areas (e.g., public corridor and lobby) for conducting the experiments. They consist of accessing Wi-Fi AP, deploying iBeacons and performing localization testing. Consequently, the corresponding waking path is meant for the pedestrians in public area only.

B. EXPERIMENT DESIGN AND DEVICE INFORMATION

We performed several experiments at the third floor of Huyngnam Engineering Building in Soongsil University. The testing area of our experiments consists of two parts: one part of 39m × 17m and another one of 86m × 24m. A user is 1.75m in height and holds the phone face up when he is measuring iBeacon signals and walking in a given track. We use an iPhone 7 with iOS 14.3 and 256GB storage for testing. All steps along the tracking path are recorded and saved in the device storage. Since we do not have any local server, all floor plan map, Wi-Fi AP/iBeacon related information (e.g., AP/iBeacon list, AP/iBeacon locations, etc.) and user tracking history are stored in the smartphone. He holds the iPhone in hand for data collection and tracking. The smartphone is placed in front of him, where the iPhone's screen is pointed up that let the user view its screen. Moreover, the designed mobile app is specifically implemented for iOS devices, thus it makes the Wi-Fi proximity conversion table (i.e., Table 2) meaningful.

In the testing area, there are four Alcatel Lucent IAP-305 2x/3x 11ac Wi-Fi APs deployed over the area. Each Wi-Fi AP is attached to the ceiling of 2.5m height at various locations in the building. Detailed description of the specifications of this AP can be found at [41]. The coverage of this Wi-Fi AP is expected about 45m indoors and about 91m in open areas. However, it may be much shorter in practice due to obstacles like walls, human bodies, various steel structures, etc. Throughout our experiments, we found practical radius of this Wi-Fi AP is about 20m.

Regarding iBeacon deployment, we use RedBear iBeacons that are operated under BLE 4.0 specification [42]. The dimension is (W)25mm × (H)70mm × (D)18.3mm as illustrated in Fig. 7. It uses two tripple-A batteries for power supply. The average battery life is about 1 year. As we mentioned in [18], the trusted-range of a RedBear iBeacon is within 4.5m. According to Section III-2, we install the iBeacons using the following two settings.

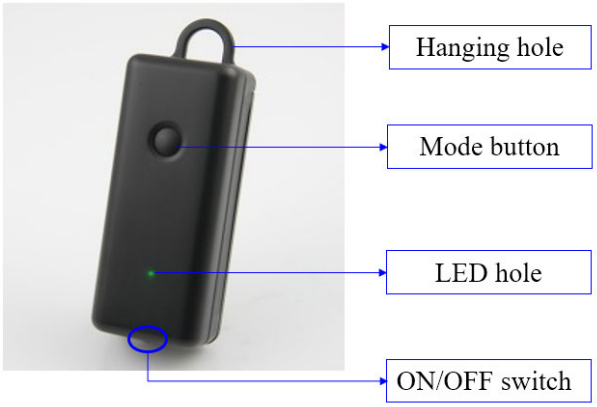


FIGURE 7. RedBear BLE iBeacon.

- (CH1) Wi-Fi APs and iBeacons are co-existed: we install iBeacons to cover the area that two adjacent Wi-Fi AP radii do not overlap according to the principles in Section IV-B. For example, we can put an iBeacon in the area between WiFi2 and WiFi3 as illustrated in Fig. 8.
- (CH2) iBeacon-only: Since the BLE signal is typically weaker than the Wi-Fi signal, the iBeacon density must be at approximately equaled to or greater than a current deployment of existing Wi-Fi infrastructure. In the areas where the iBeacon signals do not reach to the iPhone, our app uses the PDR-only approach to continuously track the iPhone location.

Moreover, iBeacons are placed on the ceiling as the Wi-Fi AP in order to reduce the disturbance when people are walking by. We transform a three-dimensional iBeacon range d_m to two-dimensional space value d_g as

$$d_g = \sqrt{d_m^2 - h_d^2}. \tag{41}$$

Here, d_m is the actual iBeacon range which is calculated by (21), h_d is the height difference between iBeacon and user's smartphone. In practice, we set h_d as one third of user height.

C. LOCALIZATION METRIC

We use the accuracy rate (AR) as a metric to evaluate the localization performance. Given a localization error bound ϵ, the AR is defined as the percentage of number of estimate

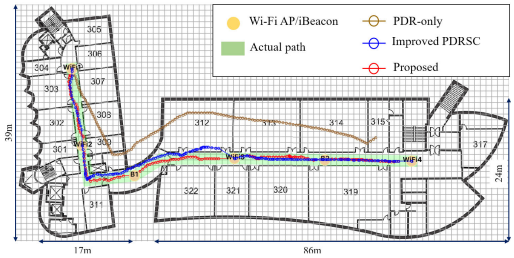


FIGURE 8. CH1: Tracking results.

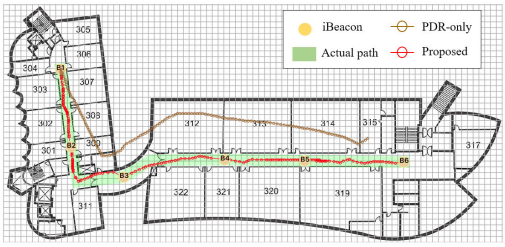


FIGURE 9. CH2: Tracking results.

points that fall inside the path of thickness ϵ. Mathematically, we write

$$AR(\epsilon) = \frac{\#\{\mathbf{P}_k \ : \ \mathbf{P}_k \subset Path(\epsilon)\}}{N_{step}} \times 100(\%), \qquad (42)$$

where N_{step} is the actual number of steps that the user has finished a given path $Path(\epsilon)$ with the thickness ϵ.

D. TRACKING RESULTS

Fig. 8 shows the tracking results of three schemes: the conventional PDR-only, the improved pedestrian dead reckoning with step compensation (PDRSC) [19] and the proposed hybrid scheme in the given testing area in case of CH1. The ground-truth resolution of the grid cells is 1 meter on the map display. In the figure, we plot the actual path in green color and Wi-Fi AP/iBeacon locations in bright yellow circle. The thickness of this actual path is set to $\epsilon = 2$ m in this experiment. The footsteps estimated by the PDR-only, the improved PDRSC [19] and the proposed scheme are marked in brown, blue and red, respectively. As we notice from the figure, the conventional PDR-only scheme suffers from large drifts and significant accumulative errors over the walking path. With the improved PDRSC [19], this drift has been corrected by integrating Wi-Fi information and the improved version of the PDR. From the figure, we observe that due to location correction process, it produces a large jump in the footstep records, especially near WiFi3 and WiFi4. This is because the Wi-Fi signals are strong at those points, so it sets the Wi-Fi AP location as the current iPhone location. On the other hand, with the help of iBeacon, our proposed approach produces a smaller jump because those iBeacons cover the areas of weak Wi-Fi signals. Thus, our approach helps to improve the accuracy of the localization system.

Similarly, Fig. 9 shows the tracking results of two schemes: the conventional PDR-only and the proposed hybrid scheme in the given testing area in case of CH2. The results show that about 81% of testing cases are successfully localized within 2m error bound via our proposed approach, which is sufficiently accurate and promising.

Regarding the proposed-hybrid only scheme only with iBeacons, Fig. 10 plots the AR versus error bound with different iBeacon spacing. From the results of Section IV-B, we start with $SP = SP_{\min}$ and increase the value of SP until it almost reaches the current Wi-Fi AP deployment spacing in the testing area. Since Beacon signals are weaker than Wi-Fi signals, the iBeacon density cannot be sparse than the

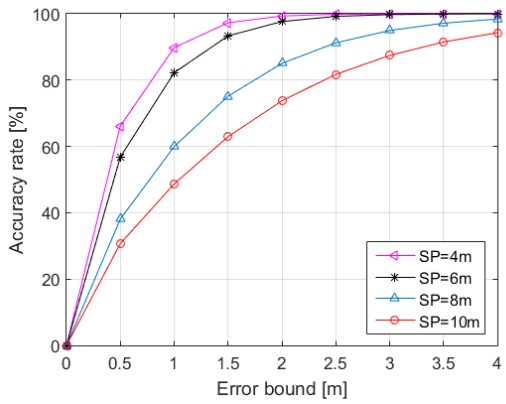


FIGURE 10. Accuracy rate versus iBeacon spacing.

Wi-Fi density. Note that we only evaluate the localization performance of the proposed approach with different iBeacon spacing where iBeacons were installed only. We observe that the localization AR decreases as the iBeacon spacing increases. However, this decrement gradually changes because of the PDR support. In particular, even when $SP = 10$m, the PDR still keeps updating the user location with IMU readings until the smartphone reaches another nearby iBeacon.

Fig. 11 shows the average localization AR with different error bound in which a user walks three times on the same path. In order to make a fair comparison among three tracking schemes (PDR-only, improved PDRSC [19] and the proposed schemes in CH1 and CH2 cases), we examine the these schemes for the same walking path, record the tracking results and leverage them under same settings. For the proposed hybrid scheme only with iBeacon (i.e., CH2), we use six iBeacons and installed them as shown in Fig. 9. We observe that the PDR-only does not perform well as the three other schemes. At 2m error bound, the improved PDRSC and our proposed schemes achieve AR of 76%. We also observe that the proposed-WiFi/PDR achieves the best AR, while remaining a stable performance over error bound. This shows that if we know how to properly deploy the Wi-Fi APs and iBeacons, we can achieve a meter-level localization accuracy.

Remark 4 (Effectiveness of the Proposed System and Other Suggestions): We would like to note that the location of the iPhone at each step was estimated by (7). To accomplish the task of tracking, its location is recursively updated by the PDR and calibrated according the workflow

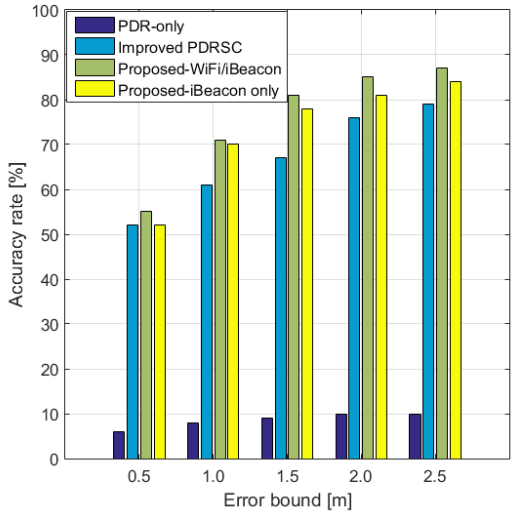


FIGURE 11. Accuracy rate versus error bound.

in Fig. 5. In summary, our proposed localization scheme achieves a good performance in mobile iPhone cases with a reasonable Wi-Fi AP/iBeacon density. For overall accuracy performance, we observe that the proposed localization scheme achieves an average accuracy rate of above 70% with 2.0m location error, which is a competitive accuracy level. Although latest iOS updates do not support much on raw RSSI readings from Wi-Fi AP as well as BLE channel selection for iPhones, our proposed tracking system overcomes these issues and requires no prior knowledge of starting point for selecting a conservative starting point for the PDR.

Our experiments show that the proposed tracking system is good enough within the testing area such as long corridor, which is a common practice in indoor environments. Due to various reasons, however, the tracking results within rooms have not been demonstrated yet. In order to determine the iPhone location at a room level with sufficient reliability, we provide some suggestions to accomplish this goal. First, with a typical floor construction above, we can place iBeacons in two categories: office room and office corridor. Since the iBeacon's major and minor values in Fig. 2 can be modified, we can mark an iBeacon as either office room or office corridor according to its placement within the floor plan. During online phase an iPhone is able to separate iBeacons scanning list according to their types of locations. Then, the tracking app can be incorporated with the series of iBeacons to help the iPhone to accurately place itself on a map. Second, because an office room is typically much smaller than the floor area, room-positioning would make more sense if we perform the localization task on a small object such as BLE tag rather than smartphones. This is because, with only one BLE beacons in the certain office room, a user can pinpoint exactly where he is in the room without using the tracking app.

VI. CONCLUSION

In this paper, we propose a hybrid Wi-Fi/iBeacon indoor tracking system for smartphones. The proposed algorithm is simple, but surpasses the conventional approaches when tracking an iPhone location in a typical office building. We tackle the indoor localization problem under several issues such as limited Wi-Fi AP information, smartphone characteristics, iBeacon placement methodology and PDR-based displacement mitigation. Smartphones collect the IMU readings which provide basic information on user motion and inputs to the improved PDR. Then, we extract related WiFi/iBeacon information to fix accumulated errors, while automatically self-updating user locations. Experiment results show that the localization accuracy is greatly improved by the proposed schemes.

Through practical experiments, we discuss some future works to adopt this technique to various indoor environments. In order to enable meter-level accuracy assisted by smartphones, we may construct user trajectories with precisely identifying indoor pinpoints and floor constraints. Regarding our proposed localization scheme, even with limit information on Wi-Fi ranging we still explore alternatives solutions to estimate the smartphone location with mobility information. Depending on the applications, we need to choose whether applying filters (e.g., particle filters, Kalman filters and their variants) or not in order to balance the trade-off between the localization accuracy and the computation cost. Thus, a framework of optimization strategy can be pursued as a separate line of research as a future work.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflict of interest.

ACKNOWLEDGMENT

(Tuan D. Vy, Thu L. N. Nguyen, and Yoan Shin contributed equally to this work.)

REFERENCES

[1] F. B. Elbahhar and A. Rivenq, Eds., *New Approach of Indoor and Outdoor Localization*. Rijeka, Croatia: InTech, 2012, Sec. 1.
[2] Y. Liu, Z. Yang, X. Wang, and L. Jian, "Location, localization, and localizability," *J. Comput. Sci. Technol.*, vol. 25, no. 2, pp. 274–297, Mar. 2010.
[3] D. Lymberopoulos, J. Liu, X. Yang, R. R. Choudhury, V. Handziski, and S. Sen, "A realistic evaluation and comparison of indoor location technologies: Experiences and lessons learned," in *Proc. 14th Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, Seatle, WA, USA, Apr. 2015, pp. 178–189.
[4] H. Shin, Y. Chon, Y. Kim, and H. Cha, "A participatory service platform for indoor location-based services," *IEEE Pervas. Comput.*, vol. 14, no. 1, pp. 62–69, Jan. 2015.
[5] S. Bak, S. Jeon, Y. Suh, C. Yu, and D. Han, "Characteristics of a large-scale WiFi radiomap and their implications in indoor localization," in *Proc. IoF*, Pohang, South Korea, Oct. 2013, pp. 1–5.
[6] H.-H. Hsu, J.-K. Chang, W.-J. Peng, T. K. Shih, T.-W. Pai, and K. L. Man, "Indoor localization and navigation using smartphone sensory data," *Ann. Oper. Res.*, vol. 265, no. 2, pp. 187–204, Jan. 2017.
[7] J. Liu, R. Chen, L. Pei, R. Guinness, and H. Kuusniemi, "A hybrid smartphone indoor positioning solution for mobile LBS," *Sensors*, vol. 12, no. 12, pp. 17208–17233, Dec. 2012.
[8] A. Poulose, O. S. Eyobu, and D. S. Han, "A combined PDR and Wi-Fi trilateration algorithm for indoor localization," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Okinawa, Japan, Feb. 2019, pp. 72–77.
[9] Q. Lu, X. Liao, S. Xu, and W. Zhu, "A hybrid indoor positioning algorithm based on WiFi fingerprinting and pedestrian dead reckoning," in *Proc. IEEE 27th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Valencia, Spain, Sep. 2016, pp. 1–6.
[10] Apple Developer. *CNCopyCurrentNetworkInfo*. Accessed: Sep. 12, 2020. [Online]. Available: https://developer.apple.com/documentation/systemconfiguration/1614126-cncopycurrentnetworkinfo?language=occ

[11] Apple Developer. *iOS Wi-Fi management APIs*. Accessed: Sep. 12, 2020. [Online]. Available: https://developer.apple.com/library/archive/qa/qa1942/_index.html
[12] S. Traini, L. Sciullo, A. Trotta, and M. Di Felice, "Practical indoor localization via smartphone sensor data fusion techniques: A performance study," in *Proc. 16th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Las Vegas, NV, USA, Jan. 2019, pp. 1–7.
[13] M. Kourogi, T. Ishikawa, and T. Kurata, "A method of pedestrian dead reckoning using action recognition," in *Proc. IEEE/ION Position, Location Navigat. Symp.*, Indian Wells, CA, USA, May 2010, pp. 85–89.
[14] H. Weinberg, "Using the ADXL202 in pedometer and personal navigation applications," Analog Devices, Cambridge, MA, USA, Appl. Note AN-602, 2002, pp. 1–6.
[15] J. W. Kim, H. J. Jang, D.-H. Hwang, and C. Park, "A step, stride and heading determination for the pedestrian navigation system," *J. Global Positioning Syst.*, vol. 3, nos. 1–2, pp. 273–279, Dec. 2004.
[16] J. Scarlett. *Enhancing the Performance of Pedometers Using a Single Accelerometer*. Accessed: Sep. 12, 2020. [Online]. Available: https://www.analog.com/en/analog-dialogue/articles/enhancing-pedometers-using-single-accelerometer.html
[17] J. Hannink, T. Kautz, C. F. Pasluosta, J. Barth, S. Schülein, K.-G. Gaßmann, J. Klucken, and B. M. Eskofier, "Mobile stride length estimation with deep convolutional neural networks," *IEEE J. Biomed. Health Informat.*, vol. 22, no. 2, pp. 354–362, Mar. 2018.
[18] T. D. Vy, T. L. N. Nguyen, and Y. Shin, "A smartphone indoor localization using inertial sensors and single Wi-Fi access point," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Pisa, Italy, Sep. 2019, pp. 1–7.
[19] T. D. Vy, T. L. N. Nguyen, and Y. Shin, "Inertial sensor-based indoor pedestrian localization for iPhones," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Jeju Island, South Korea, Oct. 2019, pp. 200–203.
[20] Apple Developer. *Getting Started With iBeacon*. Accessed: Nov. 12, 2020. [Online]. Available: https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf
[21] Z. Chen, Q. Zhu, and Y. C. Soh, "Smartphone inertial sensor-based indoor localization and tracking with iBeacon corrections," *IEEE Trans. Ind. Informat.*, vol. 12, no. 4, pp. 1540–1549, Aug. 2016.
[22] P. Spachos and K. N. Plataniotis, "BLE beacons for indoor positioning at an interactive IoT-based smart museum," *IEEE Syst. J.*, vol. 14, no. 3, pp. 3483–3493, Sep. 2020.
[23] H. Zou, Z. Chen, H. Jiang, L. Xie, and C. Spanos, "Accurate indoor localization and tracking using mobile phone inertial sensors, WiFi and iBeacon," in *Proc. IEEE Int. Symp. Inertial Sensors Syst. (INERTIAL)*, Kauai, HI, USA, Mar. 2017, pp. 1–4.
[24] T.-M.-T. Dinh, N.-S. Duong, and K. Sandrasegaran, "Smartphone-based indoor positioning using BLE iBeacon and reliable lightweight fingerprint map," *IEEE Sensors J.*, vol. 20, no. 17, pp. 10283–10294, Sep. 2020.
[25] Android Developer. *Sensor Event*. Accessed: Nov. 12, 2020. [Online]. Available: https://developer.android.com/reference/android/hardware/SensorEvent
[26] Apple Developer. *Core Motion*. Accessed: Nov. 12, 2020. [Online]. Available: https://developer.apple.com/documentation/coremotion
[27] Q. Wang, H. Luo, L. Ye, A. Men, F. Zhao, Y. Huang, and C. Ou, "Pedestrian heading estimation based on spatial transformer networks and hierarchical LSTM," *IEEE Access*, vol. 7, pp. 162309–162322, 2019.
[28] T. D. Vy and Y. Shin, "iBeacon indoor localization using trusted-ranges model," *Int. J. Distrib. Sensor Netw.*, vol. 15, no. 1, pp. 1–13, Jan. 2019.
[29] D. Li, B. Zhang, and C. Li, "A feature-scaling-based k-nearest neighbor algorithm for indoor positioning systems," *IEEE Internet Things J.*, vol. 3, no. 4, pp. 590–597, Aug. 2016.
[30] L. Mengual, O. Marbán, and S. Eibe, "Clustering-based location in wireless networks," *Expert Syst. Appl.*, vol. 37, no. 9, pp. 6165–6175, Sep. 2010.
[31] A. Kushki, K. N. Plataniotis, and A. N. Venetsanopoulos, "Kernel-based positioning in wireless local area networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 6, pp. 689–705, Jun. 2007.
[32] Apple Developer. *Core Location*. Accessed: Nov. 12, 2020. [Online]. Available: https://developer.apple.com/documentation/corelocation
[33] Apple Developer. *Core Bluetooth*. Accessed: Sep. 12, 2020. [Online]. Available: https://developer.apple.com/documentation/corebluetooth
[34] Android Developer. *Wi-Fi Manager*. Accessed: Nov. 12, 2020. [Online]. Available: https://developer.android.com/reference/android/net/wifi/WifiManager
[35] A. Pachi and T. Ji, "Frequency and velocity of people walking," *Struct. Eng.*, vol. 84, no. 3, pp. 36–40, Feb. 2005.
[36] W. Kang and Y. Han, "SmartPDR: Smartphone-based pedestrian dead reckoning for indoor localization," *IEEE Sensors J.*, vol. 15, no. 5, pp. 2906–2916, May 2015.
[37] S. M. Kay, *Fundamentals of Statistical Signal Processing, Estimation Theory*, vol. 1. Upper Saddle River, NJ, USA: Prentice-Hall, 1993, ch. 6.
[38] S. Kilburn. *Location Beacon Deployment Guide*. Accessed: Feb. 12, 2020. [Online]. Available: https://docs.meridianapps.com/hc/en-us/articles/360040136073-Location-Beacon-Deployment-Guide
[39] Aruba Networks Community. *Best Practices in Deploying and Managing Aruba Beacons, Sensors and APs for Location-Based Services*. Accessed: Sep. 12, 2020. [Online]. Available: https://community.arubanetworks.com/browse/articles/blogviewer?blogkey=74896720-ed59-449d-b6a7-e5c72ec6dd63
[40] E. Dekel. *Low-Power Internet Connectivity Over Wi-Fi*. Accessed: Feb. 12, 2021. [Online]. Available: https://www.ti.com/lit/wp/swry019a/swry019a.pdf?ts=1614047031874
[41] Alcetel Lucent Enterprise. *Alcatel-Lucent OmniAccess 320 Series Access Points: Service Multi-Tenant Network Management*. Accessed: Oct. 20, 2020. [Online]. Available: https://www.al-enterprise.com/-/media/assets/internet/documents/omniaccess-ap320-series-access-points-datasheet-en.pdf
[42] FCC ID Database. *Beacon B1 Instruction*. Accessed: Oct. 20, 2020. [Online]. Available: https://fccid.io/2ABXJ-B-B1/User-Manual/User-Manual-2223286.pdf

TUAN D. VY (Student Member, IEEE) received the B.S. degree in mathematics and computer science from the University of Science, Ho Chi Minh, Vietnam, in 2011, and the M.S. degree in information and telecommunication from Soongsil University, South Korea, in 2016, where he is currently pursuing the Ph.D. degree. His research interests include Bluetooth low energy, indoor localization, and location-based services.

THU L. N. NGUYEN received the B.S. degree in mathematics and computer science from the University of Science, Ho Chi Minh, Vietnam, in 2011, and the M.S. and Ph.D. degrees in information and telecommunication from Soongsil University, Seoul, South Korea, in 2014 and 2019, respectively. She is currently working at the Communication and Intelligent Laboratory, as a Postdoctoral Researcher. Her research interests include wireless communication, indoor/outdoor localization, and Bluetooth low energy.

YOAN SHIN (Senior Member, IEEE) received the B.S. and M.S. degrees in electronics engineering from Seoul National University, Seoul, South Korea, in 1987 and 1989, respectively, and the Ph.D. degree in electrical and computer engineering from The University of Texas at Austin, in 1992. From 1992 to 1994, he was with Microelectronics and Computer Technology Corporation (MCC), Austin, TX, as a Technical Staff Member. Since 1994, he has been with the School of Electronic Engineering, Soongsil University, Seoul, where he is currently a Professor. From September 2009 to August 2010, he was a Visiting Professor with the Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, Canada. His research interests include localization and mobile communications. He has been serving as an Organizing/a Technical Committee Member for various prominent international conferences, including VTC 2003-Spring, ISITA 2006, ISPLC 2008, APCC 2008, ISIT 2009, APWCS 2009, APWCS 2010, APCC 2010, ICTC 2010, APWCS 2011, ICTC 2011, ISAP 2011, APWCS 2012, APCC 2012, and WCNC 2020. In particular, he was the Technical Program Committee Chair of APWCS 2013, the General Co-Chair of APWCS 2014 and APWCS 2015, and the General Vice Chair of ICC2022.