An Adaptive Sensor Fusion Framework for Pedestrian Indoor Navigation in Dynamic Environments

Gunwoo Lee, Suk-Hoon Jung, and Dongsoo Han, *Member, IEEE*

Abstract—Indoor navigation is a representative application of an indoor positioning system that uses a variety of equipment, including smartphones with various sensors. Many indoor navigation systems utilize Wi-Fi signals, as well as a variety of inertial sensors, such as a 3D accelerometer, digital compass, gyroscope, and barometer, to improve the accuracy of user location tracking. The inertial sensors are vulnerable to changes in the surrounding environments and sensitive to users behavior, but little research has been conducted on sensor fusion under these conditions. In this paper, we propose a dynamic sensor fusion framework (DSFF) that provides accurate user tracking results by dynamically calibrating inertial sensor readings in a sensor fusion process. The proposed method continually learns the errors and biases of each sensor due to the changes in user behavior patterns and surrounding environments. The learned patterns are then dynamically applied to the user tracking process to yield accurate results. The results of experiments conducted in both a single-story and a multi-story building confirm that DSFF provides accurate tracking results. The scalability of the DSFF will enable it to provide more accurate tracking results with various sensors, both existing and under development.

Index Terms—Error correction, indoor navigation, sensor fusion

1 Introduction

The recent evolution of the various sensors installed in smartphones has enabled hybrid indoor positioning that integrates various signals and sensor readings for more accurate user tracking. In a hybrid indoor positioning system, a fundamental issue is how to combine the various sensor data while controlling for the noise contained in those data. In particular, whether to attribute equal or differentiated reliability to each sensor for positioning is open to debate because it is difficult to distinguish the reliability of each sensor precisely. Moreover, the sensor data includes errors and biases caused by the dynamically changing environment, self-contained sensor, etc., further confusing the determination of the weight of each sensor. If the sensor error patterns affected by this dynamic environment were properly interpreted and the reliability of each sensor revealed, a more accurate tracking result could be realized by a more sophisticated fusion of multiple sensors.

Many researchers have already proposed user tracking estimation that focuses on sensor fusion methods [1], [2], [3], [4], [5], [6], [7], in which the sensors independently determine the user's position. One widely used sensor fusion methods is the Kalman filter, which is a typical loosely coupled method [8], [9], [10]. Obviously, there are very sophisticated tightly-coupled [28] and deeply-coupled [29] Kalman filters which can be applied to GNSS. The Kalman filter enables composite user tracking relatively easily using a weighted average of the sensor positioning result. The particle filter [11], [12], [13] and Hidden Markov Model (HMM) [14], [15], [16] are fusion techniques on probability models. The particle filter predicts the current position by a stochastic integration of each weighting function and motion model so that more sophisticated fusion is possible. HMM-based fusion is a tightly coupled fusion technique, where emission and transition probabilities are used to describe the sensor data obtained while a user walks. The signal emission probability is calculated based on the distribution of the RF signals of each location, and transition probabilities are generated from inertial sensor readings. Human movement in a trace has successfully been described by HMM and its variations [17].

The studies above, however, have not properly considered irregular sensor readings due to individual and environmental diversity, especially when handling inertial measurement unit (IMU) sensor readings. Each person has a different height, weight, and movement style, and all of these features can yield different readings from IMU sensors. Environmental factors, such as the material of walls and the size of a space, also affect sensor readings. Research has been conducted on the effects of environmental [33], [43], [44] and individual [34] diversity on sensors. However, it is not easy to calibrate sensor readings to reflect such environmental and user-specific factors because they are so diverse. Therefore, individual and environmental diversity is usually ignored, and the statistically or empirically most-likely values are used to calibrate the sensors. These values are static in most cases, which makes it difficult to integrate sensors into a

• *The authors are with the Department of Computer Science, Korea Advanced Institute of Science and Technology, N1 CS723, 291 Daehak-ro, Yuseng-gu, Daejeon 34141, Republic of Korea. E-mail: {gwleee, sh.jung, dshan}@kaist.ac.kr.*

Manuscript received 5 Jan. 2019; revised 14 June 2019; accepted 7 Oct. 2019. Date of publication 14 Oct. 2019; date of current version 7 Jan. 2021. (Corresponding author: Gunwoo Lee.)
Digital Object Identifier no. 10.1109/TMC.2019.2946809

fusion system with flexible reliability. The error pattern of the IMU sensor varies because of the changes in user movement patterns and their surrounding environment while tracking, so it cannot be properly interpreted in a real-time tracking system without considering its variable reliability. If the error pattern of the IMU sensor is not properly interpreted and corrected, the noise and biases in the sensor readings are used to derive the tracking result, and overlooking these errors eventually lowers the tracking accuracy. Consequently, ignoring individual and environmental diversity and real-time changes leads to inaccurate tracking results. Therefore, there is a need for a model that can calculate reliability more accurately according to individual movement patterns and changes in the surrounding environment. In addition, given the reliability model developed, a fusion policy is needed to accommodate dynamically changing reliabilities.

In this paper, we propose a dynamic sensor fusion framework that can dynamically learn the pattern of errors, bias, and reliability of each sensor at run-time to adapt to a user's tracking environment and individual movement pattern. The proposed framework provides reliable position information using the dynamic sensor fusion (DSF) method. The main idea of this method is that the aggregate result of multiple sensors is more accurate than each result from a single sensor, so the erroneous result of each sensor can be corrected by referring to the aggregate result. By considering the aggregate result as the correct answer, the DSF method learns the error and bias of each sensor, and continuously adapts to their changes. In the early stages of learning, the DSF method assigns high weight to a sensor with known reliability and low weight to a sensor that is not reliable. In the latter case, the unknown reliability and error pattern adjusts the given weight by learning unknown factors as time progresses.

In the proposed framework, we apply an algorithm that adaptively determines the reliability of the latest data of each sensor during user tracking. Because the error patterns and biases change rapidly with the environment according to a user's trajectory, the DSF method needs to distinguish data that has a greater impact on tracking results over time. Through these procedures, the DSF method can accurately determine the reliability and bias of each sensor so that its signals can be combined to obtain a reliable results. In addition, the DSF method is adaptable to environmental changes, environmental conditions, and the typical movement patterns of an individual user.

Moreover, the DSF framework (DSFF) provides a framework in which various sensors can be integrated for user positioning. To clarify the characteristics of the positioning approach and the sensor readings used for positioning, we divide the location information derived from smartphone signal chipsets and sensors into two categories: absolute position change and relative position change. The positioning approaches that use these types of information are general approaches to user location estimation. Absolute positioning is a method that calculates an object's location as fixed coordinates. The absolute position information can be acquired using technologies such as Wi-Fi, a magnetometer, GPS, wireless sensor network, cellular network, Bluetooth, RFID, and LEDs. In contrast, relative positioning is a method that estimates the relative coordinates of present location based on a previous location. The accelerometer, gyroscope, magnetometer, compass, barometer, and other sensors can present the relative position change information. The values of these sensors are interpreted using a stochastic model, and the interpreted result is used for learning and fusion. The DSFF outputs the user tracking result as an alternating stochastic product of absolute and relative position information derived from various sensors. Therefore, a sensor can be integrated into the DSFF by representing the sensors readings or their distributions as a probabilistic model.

2 Related Work

An HMM is a statistical Markov model that has two elements: hidden states and observable outputs. In this model, the Markov process is hidden and only the outputs of the Markov process are available. In an HMM-based framework for location tracking proposed by Xing et al. [13], a graph structure is used to store fusion models generated from multiple sensors in the offline phase, and the data collected is fused using a multimodal particle filter when performing tracking in the online phase. Hoang et al. employed an HMM-based sensor fusion of Received Signal Strength Indicator (RSSI) and inertial sensor information for user tracking. They described a coarse grid of states for Wi-Fi measurements that need to be taken during training and introduce pseudo-states between the regular HMM states to reduce quantization error [14]. Liu et al. presented a smartphone indoor positioning engine that can be easily integrated with a mobile Location Based System (LBS). Two algorithms based on HMM problems, the grid-based filter and the Viterbi algorithm were used as the central processor for data fusion to resolve position estimation [15]. However, the authors did not take into consideration the constantly changing environment nor specify the parameters used for tracking. In other words, they used static values derived from mathematical calculations.

In addition to indoor positioning, HMM-based sensor fusion methods have been studied in various fields. A coupled HMM was used by Kumar et al. to fuse various sensors for sign language recognition [5]. Bernardin et al. fused two sensor inputs, dataglove-measured finger angles and tactile information, in an HMM [6]. Dong et al. developed a statistical methodology for multi-sensor equipment health diagnosis and prognosis utilizing a hidden semi-Markov model, which does not provide a complete observation in the HMM but instead provides a segment of the observations [7]. The Kalman filter, which is a recursive Bayesian estimator, is an algorithm that estimates the state of a linear dynamical system. Chen et al. amalgamated a Wi-Fi-based approach and the Pedestrian Dead Reckoning (PDR) approach with landmarks by applying a Kalman filter [8]. Ligorio et al. used a linear Kalman filter for fusing tri-axial gyroscope and tri-axial accelerometer data [9]. To fuse the various sensors for robot movements, Rigatos et al. used extended Kalman filtering [10]. For better robustness to noise, Assa et al. developed the iterative adaptive unscented Kalman filter, which is an extension of the iterative adaptive extended Kalman filter [18]. However, if the noise does not follow a Gaussian distribution, the Kalman filter performs poorly.

When the distribution of sensor values does not appear to be Gaussian, a particle filter, which is known to be more

flexible than a Kalman filter, is usually applied. A particle filter is also a recursive Bayesian estimator. It samples the possible states of the target system as particles and updates states using observations. For vehicle localization, Suhr et al. used a particle filter to fuse a number of sensors including an IMU, GPS, and a wheel speed sensor [11]. Meanwhile, Wu et al. [12] developed their particle-filtering scheme fusing a fingerprinting method and a PDR algorithm. Zhou et al. proposed a battery state-of-charge estimating algorithm based on particle filter data fusion [13]. Although the schemes are robust to irregular noise, they are not suitable for real-time smartphone tracking because they have a high computational cost.

3 DSFF

The methods to estimate absolute and relative position are classified based on the information in the acquired data of various sensors, as noted above. If the data can be collected in an offline phase and have characteristics that appear only at a specific location, these data can be used for absolute positioning. In contrast, if the sensor data have characteristics that change depending on the user's motion or route, these data can only be used for relative positioning. In the proposed method, both absolute and relative positioning are derived as a probabilistic sensor data fusion.

3.1 Extended Radio Map Construction

The first step in the implementation of the DSF method is to construct an extended radio map (ERM). Typically, a radio map is a model of network characteristics in an area of interest, where the characteristics of a location are called its fingerprint. Because all sensing data for positioning are stored in the form of fingerprints in the DSFF, we expand the concept of a radio map. In other words, the ERM indicates not only radio signal fingerprints but also any kind of sensing data fingerprints that can differentiate a location from others. In the framework proposed in this paper, an offline phase consists of two steps. The site survey step generates a set of training samples for ERM construction. This step involves laborious effort to collect training data and to assign location labels to the measurements. An analysis of the target area precedes the data collection activity so that the survey may be planned. In the calibration step, the ERM is constructed using the set of location-labeled measurements. The ERM construction begins by dividing the area of interest into location cells with the help of a floor plan. Formally, a three-dimensional (3D) area is modeled as a finite location-state space L, which is a set of physical locations with x, y and z coordinates, that is, $L = \{l_1 = (x_1, y_1, z_1), l_2 = (x_2, y_2, z_2), \ldots, l_n = (x_n, y_n, z_n)\}$, where the coordinates denote the center of a location cell. The z coordinate is specifically used for floor detection in our system.

Assume that signals from k access points (APs) can be received in the area. This will result in a one-time measurement of RSSI values transmitted from the APs, which are represented as a k-dimensional vector of RSSI values $w = < rssi_1, rssi_2, \ldots, rssi_k >$, where k is the index of an AP, and $rssi_i$ is the RSSI value of the k th AP. The RSSI value typically ranges from -100 to 0 dBm for the IEEE 802.11 standard. Although this paper uses Wi-Fi fingerprinting as a representative method of absolute positioning, other sensors can be used in the site survey if they are suitable for absolute positioning. In other words, if the sensor can be pre-trained in the offline phase, we can construct an ERM similar to the ERM that stores Wi-Fi data in a fingerprint. When the fingerprint is added to the ERM, one-time measurements of the sensor values transmitted from the devices can be extended as

$$w = < sensor1_1, \ldots, sensor1_i, sensor2_1, \ldots, sensor2_j, \ldots, sensorN_k >, \quad (1)$$

where $sensor1_i$ represents the ith signal of the nth sensor1, $sensorN_k$ represents the kth signal value of the Nth sensor.

The magnetometer is also used for absolute positioning based on the fingerprinting method [32], [35] in DSFF. Similar to the Wi-Fi ERM data, magnetometer data can be recorded as $m = < i_{norm}, i_{incl} >$, where $< i_{norm}, i_{incl} >$ represent the vector of the magnetic strength norm value and magnetic inclination at each point measured by a three-axis magnetometer. These measurements can be combined as $E_{w,m}$. If there is no disturbance during measurement, magnetic field value ranges from 25 to 65 μT outdoors whereas they range from 0 to 100 μT indoors depending on the distance between a ferromagnetic object and the measurement device.

3.2 Sensor Data Classification and Fusion

3.2.1 Probabilistic ERM Training

Once the ERM has been constructed, Wi-Fi and magnetometer fingerprint training is completed for each geographical coordinate of the target area. Because training data are already labeled in the ERM, the probabilities can be calculated using the respective AP strengths and magnetometer signal strengths measured in the online phase. To calculate a new absolute position in real-time, we need to perform an online measurement OE update formulated as E, such that, $p(w_t = OE_w|L_t)$ and $p(m_t = OE_m|L_t)$, at each location. To calculate $p(w_t|L_t)$ and $(m_t|L_t)$, a Gaussian distribution is assumed for each signal collected $w_{t,i}$ of AP i and m_t at the survey site. Although signal strength noticeably varies with the distance between an AP and a location, a single RSSI value is still not enough to represent the feature of an AP at one location because of the uncertainty in signal propagation. The magnetic field feature has similar characteristics because magnetic distortion is caused by surrounding ferromagnetic objects.

The probabilistic DSF method tries to explain this sort of phenomena by treating RSS and magnetic field measurement as the results of a random process. The aim of the probabilistic absolute position update is to compute the conditional probability $p(l_x|OE)$ of a location l_x given a certain online measurement OE, and find a location l^* that maximizes the posterior probability for positioning. In the online positioning phase, a posterior distribution over all the locations is computed using Bayes rule as follows:

$$p(l_x|OE) = \frac{p(OE|l_x)p(l_x)}{\sum_{l_i \in L} p(OE|l_i)p(l_i)}, \quad (2)$$

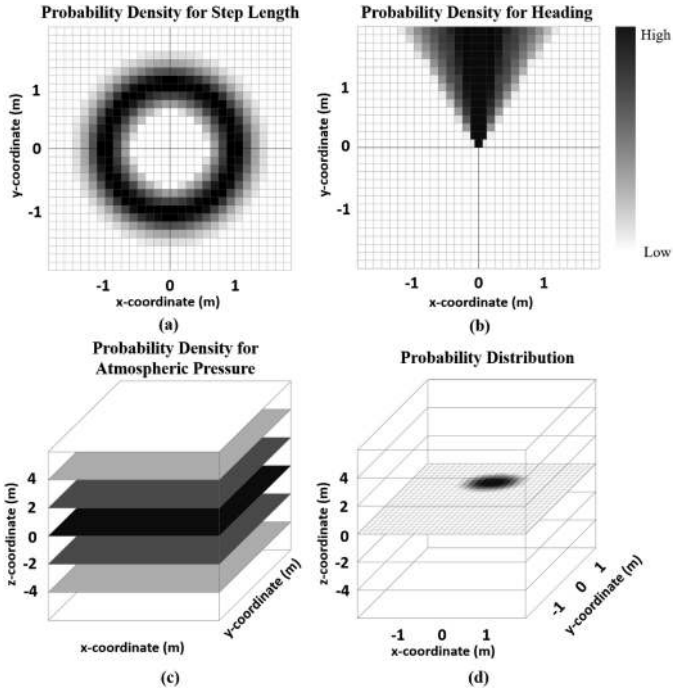

Fig. 1. Conditional probability densities for (a) accelerometer, (b) gyroscope, and (c) barometer. (d) Probability distribution for location.

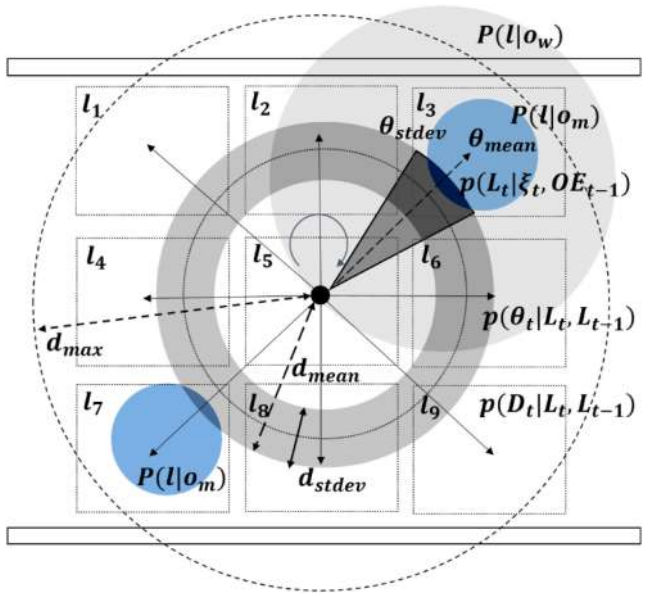

Fig. 2. Probabilistic sensor fusion for relative and absolute position updates. The gray ring and fan-shaped zones indicate the conditional probabilities of the moving distance and orientation, respectively. The probability distribution when considering 2D movement is indicated by dark blue circles.

where $p(l_x)$ is the prior and $p(OE|l_x)$ is the likelihood. The denominator of the equation is a normalization constant related only to the given OE and where a device may be located. A conventional choice for the prior is a uniform distribution that assumes every location is equally likely.

The likelihood function computes the probability of observing that measurement at a certain location. By assuming independence among the signals from different APs, the likelihood $p(OE_w|l_x)$ is computed as follows:

$$p(OE_w|l_x) = \prod_{i=1}^{k} p(rss_i^m|l_x), \tag{3}$$

where rss_i^m is the RSS value of AP i shown in measurement m and $p(rss_i^m|l_x)$ is the likelihood of rss_i^m at location l_x. Similarly, likelihood $p(OE_m l_x)$ is computed as follows:

$$p(OE_m|l_x) = p(i_{norm}, i_{incl}|l_x), \tag{4}$$

where $p(i_{norm}, i_{incl}|l_x)$ is the likelihood at location l_x. The value of OE is calculated by simply multiplying (3) and (4).

3.2.2 Probabilistic Motion Sensor Update

The results for the relative positioning are derived by fusing step length, step detection, the direction of movement, and pressure change with the value of OE. The probability densities for each of these sensor data have a different shape as shown in Fig. 1. The conditional probability of moving distance is derived from accelerometer readings as a ring shape. For moving direction and floor detection, the probability densities from gyroscope and barometer readings form triangle and bar shapes, respectively. Because the probability distributions for relative position changes are distributed differently according to data sensor types, the probability for each of them must be calculated independently.

In the offline phase, the location information of the distance and direction between neighboring cells is determined after dividing the target area into several cells. For instance, the mean value of the real-time gyroscope reading is used to estimate user direction in the online phase. For moving distance, all neighboring cells of a previous location have the same probability. However, if the probability distribution is derived from the conditional probability of the moving distance and the direction simultaneously as shown in Fig. 2, more accurate positioning results can be obtained. Meanwhile, we use barometer signal distribution to determine the floor detection. Because atmospheric pressure changes with altitude, it is possible to distinguish floor levels using this sensor.

To track user movement, we calculate the probability distribution $p(L_t|\xi_t, OE_{t-1})$, where ξ_t denotes the set of movement measurements at time t. In a Markov-based model, user location L_t depends only on the previous location L_{t-1}. Therefore, we can represent moving distance D_t, gyroscope measurement θ_t, and barometer reading B_t as follows:

$$\begin{aligned} p(L_t|\xi_t, OE_{t-1}) = \sum_{L_{t-1}} & p(L_t|L_{t-1}, D_t, \theta_t, B_t) \\ & \cdot p(L_{t-1}|\xi_{t-1}, E_{t-1}). \end{aligned} \tag{5}$$

Given the moving distance, heading measurement and barometer reading can be derived using Bayes law as follows:

$$p(L_t|L_{t-1}, D_t, \theta_t, B_t) = \frac{p(D_t, \theta_t, B_t|L_t, L_{t-1}) \cdot p(L_t|L_{t-1})}{p(D_t, \theta_t, B_t|L_{t-1})}. \tag{6}$$

Because we assume that the moving distance, heading measurement, and barometer reading are independent of each other, (6) yields

$$p(L_t|L_{t-1}, D_t, \theta_t, B_t) = \frac{p(D_t|L_t, L_{t-1}) \cdot p(\theta_t|L_t, L_{t-1}) \cdot p(B_t|L_t, L_{t-1}) \cdot p(L_t|L_{t-1})}{p(D_t) \cdot p(\theta_t) \cdot p(B_t)}. \quad (7)$$

For a transition probability $p(L_t|L_{t-1})$, a user's movement to an adjacent location is more likely than a movement to a distant location. However, the moving distance D_t, gyroscope heading measurements θ_t, and barometer reading B_t are already given reliably; we hence know which location is more likely than the others. By inserting (5) into (7), we obtain

$$\begin{aligned} &p(L_t|\xi_t, OE_{t-1}) \\ &= c \cdot \sum_{L_{t-1}} p(D_t|L_t, L_{t-1}) \cdot p(\theta_t|L_t, L_{t-1}) \cdot p(B_t|L_t, L_{t-1}) \\ &\cdot p(L_{t-1}|\xi_{t-1}, OE_{t-1}). \end{aligned} \quad (8)$$

The constant c in (8) is determined at the end of the motion sensor update for normalization because the summation of all possible locations has to be 1.

As visualized in Fig. 2, for instance, we assume that the movement measurement set ξ_t is taken when a user first reaches the location $l_i(= l_t, i \in k)$. The gray ring and fan-shaped zones denote the conditional probabilities of the moving distance and orientation respectively. By multiplying these conditional probabilities, we can obtain the probability distribution of ξ_t as the dark gray area, which is the intersection of the ring and fan-shape zones at l_3 when $l_{t-1}(= l_5)$ is the previous location (i.e., the center of the coordinate system). Here, we specify the step length of the user as a mean value d_{mean}, the maximum moving distance d_{max} and the standard deviation of the step length d_{stdev}.

Similarly, the heading of the user θ_{mean} and standard deviation of the heading θ_{stdev} are set implicitly. These mean values denote the online phase sensor readings of the user's device. Standard deviations are set empirically at the very beginning. However, as the tracking of a route progresses, these values will be changed based on online phase data. In this case, the probability distribution when considering 2D movement is shown in Fig. 2 as dark blue circles. Similarly, the conditional probability of the barometer reading is also multiplied by the probability distribution of ξ_t when a user is tracked in 3D space.

4 Dynamic Adaptation of Probability Distribution for Sensor Fusion

Generally, a probability distribution for each location is learned through offline training for absolute positioning. The probability of each location can be derived with relatively high reliability by reflecting environmental factors. However, for relative positioning, it is difficult to obtain an error distribution that contains both location and environmental factors. For example, a probability distribution is obtained for the moving distance estimation after the reliability of the method has been calculated using step detection or a similar variable, and the static value is continuously used. Because this approach overlooks the environmental changes and user behavior patterns at each location, it is difficult to obtain a result that is as reliable as absolute positioning.

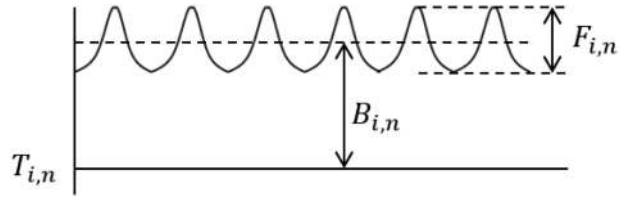

Fig. 3. Error characteristics of the relative sensor.

The framework proposed in this paper solves this problem by continually learning and correcting the error distribution of the sensors that are used for relative positioning by assuming that the position tracking result obtained from the fusion of multiple sensors is closer to the correct answer than the result of a single sensor. The positioning results of each sensor used for absolute positioning are derived using off-line training data and then integrated to determine a single probability through sensor fusion. This integrated absolute positioning result exhibits relatively high accuracy because it is the result of the highest probabilities for each sensor. Therefore, at an early stage of user tracking, we assume that this positioning result is close to the correct answer, and use it to correct sensor error.

The error probability, which is the basis of the calculation, is corrected for each sensor by training in the online phase. Here, we create a database called an error block (EB) to store the errors for the error correction. Using the tracking result of the optimal trace, the error is continuously accumulated in the EB and re-learned as the environment dynamically changes. Thus, as tracking progresses, the error of each sensor should be corrected.

4.1 System Overview

Fig. 3 shows the error characteristics of the relative sensor used in the DSFF, where the bias is included in the sensor reading and produces an undesirable result. Note that, while there are many ways to express errors in system modeling, we used a way which is to consider fluctuation and bias errors. To this end, variables are defined as follows:

$$R_{i,n} = T_{i,n} + B_{i,n} + F_{i,n}, \quad (9)$$

where, $R_{i,n}$ is a raw value of the sensor reading, i denotes relative sensor type, n represents time, $T_{i,n}$ is the true value, $B_{i,n}$ represents bias error, and $F_{i,n}$ denotes fluctuation error. What we originally expected is that the sensor reading $R_{i,n}$ is equal to the actual value $T_{i,n}$, resulting in an accurate result. However, in reality, it is unlikely for $R_{i,n}$ and $T_{i,n}$ to coincide due to errors arising from bias, fluctuation and other factors. DSFF aims to improve accuracy over time. That is, if $R_{i,n}$ converges to $T_{i,n}$, the desired goal is achived.

The system S is defined as a naive system in which the calibration algorithm of this study is not applied and the relative sensor reading value is simply input and an answer is generated as output. If the true value $T_{i,n}$ is input to the entire system S, the ideal result G_n will be derived, but in reality, the sensor reading $R_{i,n}$ containing the error is input and the estimated result Q_n is derived. This can be expressed as

$$\begin{aligned} S(T_{i,n}) &= G_n, \\ S(R_{i,n}) &= Q_n. \end{aligned} \quad (10)$$

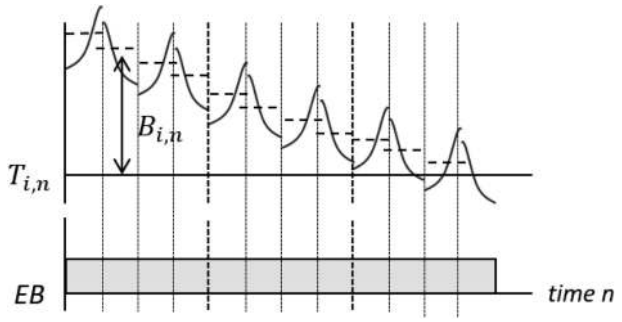

Fig. 4. Bias error correction procedure.

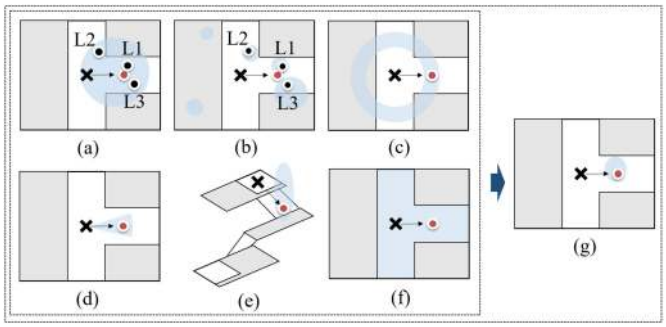

Fig. 5. Probability density of (a) absolute sensor 1, (b) absolute sensor 2, (c) relative sensor 1, (d) relative sensor 2, (e) relative sensor 3; Depiction of (f) map matching and (g) aggregate result.

The main goal of our proposed dynamic sensor fusion technique is to make Q_n as close as possible to G_n. In this case, if filter f is used to receive the sensor reading $R_{i,n}$ as an input and output a correction value for the error, the following filter f output may be used to correct Q_n to be closer to G_n and should be added to G_n. The expression is as follows:

$$correction\ value = f\left(\sum_{r=1}^{n-1} R_{i,r}\right). \tag{11}$$

In previous studies, assuming that the estimated error of the tracking system is obtained as $Q_n - G_n$, the DSF obtains the final revised error as follows:

$$revised\ estimation\ error = S\left(R_{i,n} + \left(f\sum_{r=1}^{n-1} R_{i,r}\right)\right) - G_n. \tag{12}$$

Using this concept, we can intuitively understand the difference between correcting and not correcting inputs. The error factor for the sensor input of the naive system S can be simply expressed as

$$\left|\frac{R_{i,n} - T_{i,n}}{T_{i,n}}\right|. \tag{13}$$

Applying the correction algorithm filter f proposed in this study changes this equation as follows:

$$\left|\frac{R_{i,n} + f\left(\sum_{r=1}^{n-1} R_{i,r}\right) - T_{i,n}}{T_{i,n}}\right|. \tag{14}$$

Since $R_{i,n}$ and $T_{i,n}$ of each relative sensor are not scalar values on the same dimension, this paper will not consider how to calculate these values from the actual value because the calculation depends on the desirable error model for each sensor. As time progresses in the algorithm, since the bias error will be the same in each period for which the environmental factor is homogeneous, the correction value, which is the output of f, will converge to the negative value of the bias value. This can be expressed as follows:

$$\lim_{n\to\infty} f\left(\sum_{r=1}^{n-1} R_{i,r}\right) = -B_{i,n}. \tag{15}$$

We can re-arrange (14) from (15), then all the bias errors are corrected and only the fluctuation error remains. This process is summarized in (16) and is shown in Fig. 4.

$$\left|\frac{R_{i,n} + f\sum_{r=1}^{n-1} - T_{i,n}}{T_{i,n}}\right| = \left|\frac{R_{i,n} - B_{i,n} - T_{i,n}}{T_{i,n}}\right| = \left|\frac{F_{i,n}}{T_{i,n}}\right|. \tag{16}$$

The proposed framework aims to find the right values through online learning, even if the initial probability distribution of error for a particular sensor is not known exactly. Therefore, we assume that the most common Gaussian distribution is the distribution of a particular sensor. In other words, we did not process the probability distribution of each sensor, so the probability distribution of the sensors we used in our experiments is the same as that which is typically applied in other studies [30], [31], [32].

It would be possible to precisely define all the error distribution of each sensor in every building to clearly understand the probability distribution of the error distribution. This would enhance the accuracy of results if the sensor values are set at the beginning. However, our online learning-based error correction technique is not targeted at defining the initial error distribution accurately, However, the aim is,to improve the accuracy of the defined distribution by learning about the error distribution dynamically while tracking. In other words, we set the initial error distribution to Gaussian, and then focus on testing to improve the accuracy over time.

4.1.1 Assumption of the Multiple Sensor Result

As stated above, we assume that the result of multiple sensors is closer to the correct answer than the results of a single sensor. In Fig. 5, (a) and (b) show the probability distributions when absolute positioning based on probability is performed. To this end, we assume that the probability of location L_1 is 0.8, the probability of L_2 is 0.7, and the probability of L_3 is 0.8 in Fig. 5a. We also assume that the probability of location L_1 is 0.7, the probability of L_2 is 0.7, and the probability of L_3 is 0.6 in Fig. 5b. In Fig. 5a, L_1 and L_3 have the same probability of 0.8, and in Fig. 5b the probability of L_1 and L_2 is equal to 0.7. There is ambiguity when we look at them separately because two different locations have the same probability.

This example aims to explain the assumptions underlying the use of multiple sensor results. However, when we consider the probability of the actual positioning results, it

is often similar to 10 decimal places, and any changes are seen only after 10 decimal places. If we multiply the probabilities from the absolute sensors, the probability L_1 is 0.56, L_2 is 0.49, and L_3 is 0.48. Now, we have ascertained that a single location L_1 has the highest probability, simply by multiplication. No complex calculations are necessary. At the same time, we have confirmed the possibility that the accuracy of the defined probability distribution is enhanced when probabilities from multiple sensors are multiplied, than when a single sensor is used.

Using the same method, we can derive the probability of an accurate positioning by calculating the probability distribution for the relative sensors depicted in Figs. 5c, 5d and 5e using (8). Likewise, the probabilities of the additional sensors are multiplied, and the ambiguity disappears. (Note that, because we are describing the concept of multiple sensor results here, we do not consider cases where the error of a single sensor is large. This will be discussed in Section 4.8). If we proceed to map matching using the indoor layout as shown in Fig. 5f for a single probability as derived with this method, we can obtain a more accurate final positioning result as shown in Fig. 5g.

4.1.2 Extension of the DSFF to Other Sensors

DSFF is scalable and new data can be easily fused when new sensors are added to the system. Whether we use the sensors described in Section 1 or other kinds of sensors, we can fuse the values measured by them for absolute positioning by stochastic derivations. Moreover, it is possible to calculate the conditional probability according to the type of sensor or systematically calculate the movement probability through map matching. Assuming these sensors are added to the DSFF and used for tracking, we can obtain their probability distributions as follows.

Because the GPS's Android API provides standard deviations with latitude and longitude coordinates, we can obtain the reliability of the derived coordinates $P(GPS|Loc)$. Bluetooth signal distribution can be trained in advance when the ERM is constructed. The signals collected in the online phase can be used to derive the probability of the location $P(BLE|Loc)$. Locality sensitive hashing [21] can be used to perform image matching for indoor positioning using the probabilistic nearest neighbor search method to calculate $P(IMG|Loc)$.

Wi-Fi modules are often used for absolute positioning, but they can also be used to detect relative position change using signal variation $P(WiFi_{diff}|Loc)$ [22]. Because we use the indoor layout of a building in DSFF, map matching can also be used to calculate the product of the probability. For example, we can set the probability of movement to zero when a wall exists in a movement space or set it to 0.25 at a corridor intersection. Assuming that DSFF tracking is implemented at the intersection, we can derive a probability-based result from the following equation:

$$\begin{aligned} &P(WiFi, BLE, \ldots|Loc) \\ &\quad = \frac{1}{4} \times P(WiFi|Loc) \times P(BLE|Loc) \\ &\quad \times (GPS|Loc) \times (IMG|Loc) \\ &\quad \times (PDR|Loc), \ldots. \end{aligned} \tag{17}$$

In (17), each sensor is independently used to obtain positional probability. Even if error correction is performed for each sensor, the independent final probability value of each sensor is used for tracking estimation. This encapsulates the probability calculation for each sensor. This means that even if a new sensor is added, the tracking result can be derived by simply multiplying the independent probabilities for each sensor.

4.1.3 EB-Applied Relative Sensor Probability Distribution

To calculate the transition probability of relative sensors, i.e., $p(relative\ sensor|L_t, L_{t-1})$, we can derive

$$P(S' = x) = \int_k P(S = k) \cdot P(EB = x - k)dk, \tag{18}$$

where S' denotes the EB-applied sensor signal distribution of each sensor and k represents the sensor reading. Since the distribution of the EB is discrete, it can be re-written as follows:

$$P(S' = x) = \sum_n \int_{x-(n+0.5)w}^{x-(n-0.5)w} P(S = k) \cdot P(EB = nw)dk, \tag{19}$$

where w denotes the width of the EB.

4.2 Dynamic Error Correction for User Direction

The gyroscope is used as a relative positioning method to detect user orientation. Here, the original location is inversely estimated using the gyro effect generated by the rotation of the object and the current direction is inversely calculated. The magnetometer and gyroscope are widely used to determine the direction of a user. The magnetometer uses the azimuth to determine the absolute angle between magnetic north and the moving direction. Because the magnetometer calculates direction with respect to the direction of gravity obtained by the accelerometer and the value of the magnetic field, if ferromagnetic materials that can influence the magnetic field exist in the environment, the measurement error increases. Accordingly, we use the gyroscope sensor to estimate user movement direction. This sensor represents relative angular velocity and is negligibly influenced by external noise. However, gyroscope data are not sufficient for obtaining accurate direction information because errors accumulate as a result of gyro drift.

Fig. 6 illustrates the process of the dynamic error correction focusing on user direction estimation. Suppose a user's movement has been tracked for times T0 to T3. In Fig. 6a, φ represents the direction of the tracking result, θ denotes the reading value of the gyroscope, and $\Delta\varphi$ is the difference between φ and θ. As shown in Fig. 6, the tracking result $\varphi_{T1,T2,T3}$ and the inertial sensor reading $\theta_{T0,T1,T2}$ at each time are not the same. This mismatch should be treated as an error of the inertial sensors and corrected for in the calculation of the next moving probabilities. When a mismatch occurs, the error distribution is changed by storing the EB, as shown, from T1 to T3. After storing the EB, the heading probability distribution is changed depending on its distribution.

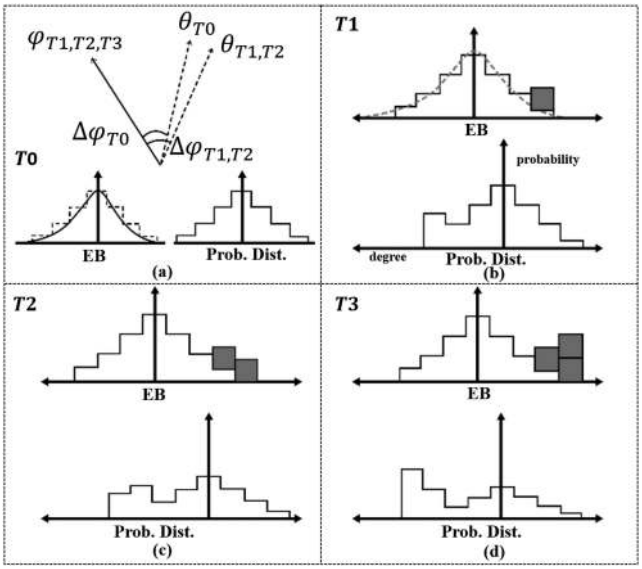

Fig. 6. EB generation for heading error compensation.

At time T0, an initial location is determined from the calculation of E_t, and the EB is initialized for recording the error of the inertial sensors. In Fig. 6a, the EB forms a normal distribution with an initial value under the assumption that the sensor reading is correct. Initially, the EB is formed with a smooth curve, but this can be converted to the shape of a block with the same area, as shown in Fig. 6b.

At time T1, as shown in Fig. 6b, the EB is accumulated at the value corresponding to $\Delta\varphi_{T0}$. By contrast, the opposite value weight of the EB becomes higher in the heading probability distribution. When normalizing the heading probability distribution, the weight of the other heading probability, which is set to the initial value, decreases. In other words, the weight of the new heading bias value relative to the initial error distribution can be adjusted by the ratio of the area of the EB to the area of the initial error distribution.

At time T2, as shown in Fig. 6c, the EB accumulates to the right of the EB values generated at time T1 because $\Delta\varphi_{T1}$ is larger than $\Delta\varphi_{T0}$. This EB construction causes the heading probability distribution to expand to the left; the weight of the correction value rises and the weight of the initial value decreases, as in T1. At time T3, the EB is generated at the same position because $\Delta\varphi_{T2}$ has the same value as $\Delta\varphi_{T1}$. As shown in Fig. 6d, the initial value of the heading probability distribution decreases, the weight of the corrected value increases, and a high peak is formed toward the corrected value. Learning the EBs in this way and applying the correction values to the heading probability distribution corrects the heading bias due to the effect of assigning higher weights to the correction values.

In the EB graph, the y-axis represents the value of the EB and the x-axis is the value of the error that appears at every step. In the heading probability distribution graph, the y-axis represents the probability and the x-axis represents the angle in degrees. For the direction of movement, the relative angular velocity is obtained in rad/s from the gyroscope. To convert radians into degrees, we integrate using $rad_j = \sum_{i=1}^{j}(gyro_i \cdot (time_i - time_{i-1}))$ and find the direction of movement using $\theta_j = rad_j \cdot 180/\pi$.

4.3 Dynamic Error Correction for Moving Distance

Step detection and step length estimation are the basic elements used to calculate the relative movement distance of a user. To detect a user's steps, we use the norm value of the three-axis accelerometer, and calculate steps based on a combination of peak detection and the Fourier transform. The acceleration values obtained for each pedestrian create a waveform with a certain period. The peak detection detects the local maxima of the waveform, which are treated as the steps of the pedestrian.

Peak detection alone has a low recognition rate for steps because there are many local maxima in the acceleration waveform of a pedestrian. Therefore, the frequency components of the waveform are obtained using the Fourier transform and analyzed to estimate whether or not the user is walking. When the transformed data have a large value at a specific frequency, this is defined as the walking frequency. To estimate the number of steps, this frequency component is multiplied by the pedestrians total moving time.

A delay occurs when the Fourier transform method is used to estimate steps because the walking frequency of the pedestrian does not appear until a certain amount of acceleration data has been collected. We use a combination method for relative moving distance, but other step detection algorithms, e.g., zero crossings [19], could be used.

For step length calculation, we use a nonlinear model that represents a relation between acceleration magnitude and step length [20]. The expression is as follows:

$$SL_k = \vartheta \cdot \sqrt[4]{a_{max}(t) - a_{min}(t)}, \tag{20}$$

where SL_k represent the kth step length and $a_{max}(t)$ and $a_{min}(t)$ are respectively the maximum and minimum values of the acceleration reading at time t. Coefficient ϑ is the ratio of the estimated and real distance, which needs to be adjusted for different subjects as follows:

$$\vartheta = \frac{d_{real}}{d_{estimated}}. \tag{21}$$

This approach is easy to implement because of its simplicity, which dynamically calculates the step length during walking. Hence, we adopt this method for the step length estimation of the DSFF.

The relative position of the user is determined from the displacement direction, step detection, and step length as follows:

$$Coor_t = Coor_{t-1} + SL\begin{bmatrix} sin(\theta_t) \\ cos(\theta_t) \end{bmatrix}, \tag{22}$$

where θ_t denotes a corrected heading angle or sensor reading, $Coor_t$ is the (x_t, y_t) coordinate at time t, $Coor_{t-1}$ is the (x_{t-1}, y_{t-1}) coordinate at time $t-1$, and SL_t represents step length. As with the heading EB, the distance EB also generates the EB based on the tracking result of the optimal trace. When tracking results and IMU sensor readings are not synchronized, it does this by dividing by the detected step count in the moving distance of successive tracking results. Consequently, the difference in distance is applied to each step and stored as a probability distribution of errors for the distance EB. The distance error correction employs the same

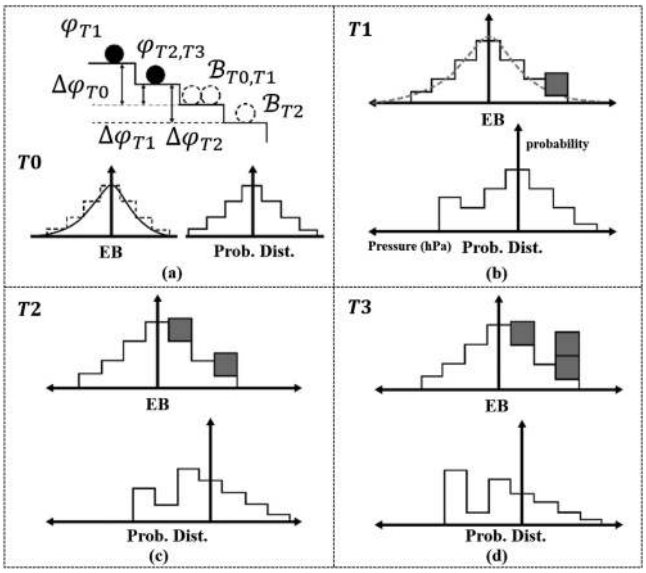

Fig. 7. EB correction of the barometer reading.

concepts and procedures as heading error correction: the x-axis of the EB represents the state in which the tracking result is longer than SL_k, and the y-axis of the EB represents the opposite case. The weight of the distance bias value can also be adjusted based on the ratio of the EB area, just as in the heading error correction.

4.4 Dynamic Error Correction for Floor Detection

To track users' movement between floors, we use a smartphone-embedded barometric pressure sensor that detects pressure change according to altitude. The barometric pressure sensor can be utilized for distinguishing user location in terms of height in a multi-story building. As is generally known, the atmospheric pressure decreases as the altitude increases. The altitude can be calculated using these characteristics and sea level pressure with scale factors [23].

However, the pressure value changes according to the weather or other environmental factors when checked on a daily basis at the same location. If the weather fluctuates suddenly, there may be a larger error. Because of these irregular changes, barometer data is not appropriate for training in the offline phase. It is more helpful to measure the pressure at the point of providing the navigation service and utilize it for movement between floors. Using the barometric formula [24], we can calculate user location in terms of altitude corresponding to barometric pressure changes using

$$p = p_0 \cdot exp\left(-\frac{g \cdot M \cdot H}{R \cdot T}\right), \tag{23}$$

where p denotes the reference barometric pressure at a certain altitude H, p_0 is the mean sea level pressure, g is the gravitational acceleration, M is the molar mass of dry air, R is the universal gas constant of air, and T is the temperature. These scale factors can be obtained from the weather information provided by a weather station or from [25]. We rearrange (23) to obtain altitude as follows:

$$H = -\frac{R \cdot T}{g \cdot M} \cdot ln\left(\frac{p}{p_0}\right). \tag{24}$$

If we calculate p using the height information of the building in (23), it will differ from the barometer reading. This difference then becomes the bias of the barometer, which differs for each device. Although some studies have initially compensated for this bias by calibrating the barometer reading [26], the bias compensation value which is initially set is employed as a static value, which is not consistent with the objective pursued by the DSFF. The International System unit for pressure is the pascal (Pa), but we use hectopascal (hPa) units for the DSFF. In general, hPa or millibar units are used for atmospheric pressure.

The current floor is estimated using (24) as follows:

$$FL = \frac{H_c - H_0}{D}, \tag{25}$$

where FL is the estimated floor level, H_0 denotes the first floor height of the building, H_c denotes the present height of the user location, and D represents inter-floor height of the building.

Fig. 7a shows the error correction process by applying the EB to the barometer reading. In Fig. 7a, B represents barometer reading, φ denotes the tracking result, and $\Delta\varphi$ is the difference between φ and θ. The barometer EB is formed using the vertical moving distance of the tracking result and the change in pressure. After obtaining the approximate floor level at the beginning of the positioning from (25), the tracking result $\Delta\varphi_{T0}$ is obtained using the difference between B_{T0} and the tracking result φ_{T1}. In this case, the EB is generated using the EB probability distribution corresponding to $\Delta\varphi_{T0}$ as shown in Fig. 7b, and the shape of the probability distribution of the barometer value is modified to reflect the EB generated. In the same way, φ_{T1} and φ_{T2} are calculated, and $\Delta\varphi_{T0,T1,T2}$ values are formed by dividing both sides as shown in Figs. 7c and 7d. As the EB is generated, the initial weights of the pressure value probability distribution decrease, and the barometric biases generated by the EB have a higher weight. In this way, even if the barometer readings are different because of the offset of each device, the initial floor level and EB can be used to calibrate for the offset, and error correction can be used to correct the bias of the barometer readings.

4.5 Error Correction Convergence in DSFF

It is assumed that the initial error block is a discretized normal distribution EB, where the probability space is $[-2.5\sigma, 2.5\sigma]$, as depicted in Fig. 8 and, in this example, the width of each error block is set to σ, but it can be changed depending on the user's preference. Further, it is assumed that EB is the initial error block distribution of orientation following $Z(0, 3)$ and that there is a certain orientation bias b, following $B = Z(\mu_b, \sigma_b = 3)$. An error block is then generated at around -30 degrees per each step. Let h be the height of the error block accumulated at each step. The probability density of each error block is $h\sigma$.

The error correction effect can be examined based on the magnitude of $P(EB = -\mu_b)$. If the probability distribution of EB satisfies

$$P(-\mu_b - 2.5\sigma_b \leq EB \leq -\mu_b + 2.5\sigma_b) \geq 0.95, \tag{26}$$

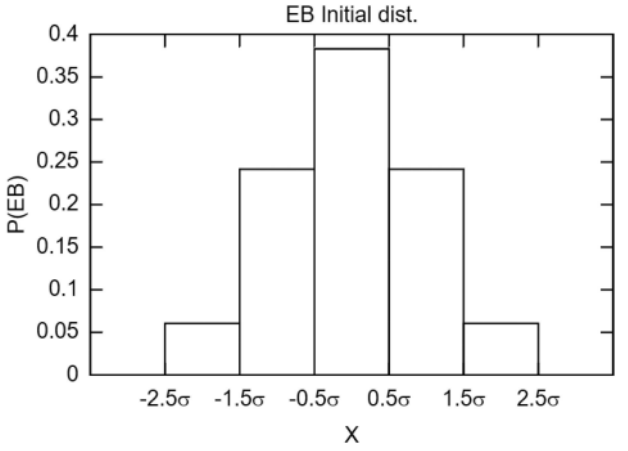

Fig. 8. Initial distribution of error block.

EB is considered to have converged enough to have the error correction effect. Let EB_k be EB at kth step. Suppose $P(EB_0 = -\mu_b) = 0$. Then

$$\begin{aligned} P(EB_1 = -\mu_b) &= P\left(\left(\frac{\sigma}{2}\right)\left\lfloor\frac{2\mu_b}{\sigma}\right\rfloor \le B \le \left(\frac{\sigma}{2}\right)\left\lfloor\frac{2\mu_b}{\sigma}+2\right\rfloor\right) \\ &\times \left(\frac{h\sigma}{1+h\sigma} + P(EB_0 = -\mu_b)\right) + P\left(\left(\left(\frac{\sigma}{2}\right)\left\lfloor\frac{2\mu_b}{\sigma}\right\rfloor \le B\right.\right. \\ &\left.\left. \le \left(\frac{\sigma}{2}\right)\left\lfloor\frac{2\mu_b}{\sigma}+2\right\rfloor\right)^c\right) \times P(EB_0 = -\mu_0) = KC, \end{aligned} \tag{27}$$

where

$$\begin{aligned} K &= P\left(\left(\frac{\sigma}{2}\right)\left\lfloor\frac{2\mu_b}{\sigma}\right\rfloor \le B \le \left(\frac{\sigma}{2}\right)\left\lfloor\frac{2\mu_b}{\sigma}+2\right\rfloor\right) and \\ C &= \frac{h\sigma}{1+h\sigma}, \end{aligned} \tag{28}$$

K is the probability that b is in the 2σ range, and C is the amount of accumulated error block normalized by $1 + h\sigma$

$$\begin{aligned} &P(EB_2 = -\mu_b) \\ &= K\left(\frac{P(EB_1 = -\mu_b)}{1+c\sigma} + C\right) + (1-K)\frac{P(EB_1 = -\mu_b)}{1+h\sigma} \\ &= \frac{K^2C}{1+h\sigma} + KC + \frac{KC - K^2C}{1+h\sigma} \\ &= KC\left(1 + \frac{1}{1+h\sigma}\right), \\ &P(EB_3 = -\mu_b) \\ &= KC\left(1 + \frac{1}{1+h\sigma} + \frac{1}{(1+h\sigma)^2}\right), \\ &\quad\vdots \end{aligned} \tag{29}$$

From (29), we can derive

$$\begin{aligned} &P(EB_k = -\mu_b) \\ &= KC\left(1 + \frac{1}{1+h\sigma} + \frac{1}{(1+h\sigma)^2} + \cdots + \frac{1}{(1+h\sigma)^k}\right) \\ &= KC\frac{1 - \left(\frac{1}{1+h\sigma}\right)^n}{1 - \frac{1}{1+h\sigma}}. \end{aligned} \tag{30}$$

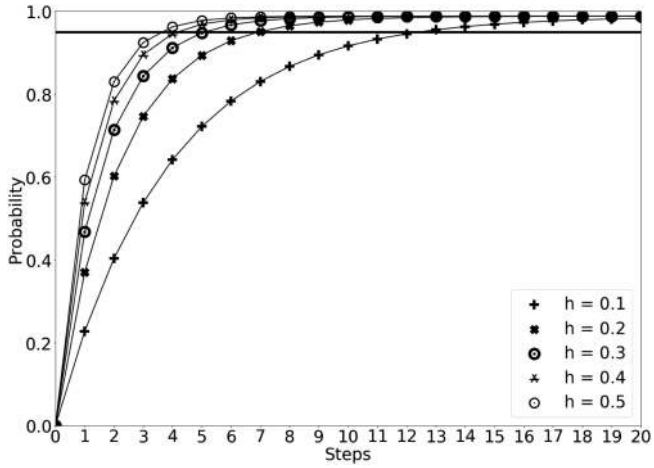

Fig. 9. Error correction convergence with different h values.

Then

$$\begin{aligned} &P(EB_k = -\mu_b + \sigma) = P(EB_k = -\mu_b - \sigma) \\ &= K'C\frac{1 - \left(\frac{1}{1+h\sigma}\right)^n}{1 - \frac{1}{1+h\sigma}}, \\ &P(EB_k = -\mu_b + 2\sigma) = P(EB_k = -\mu_b - 2\sigma) \\ &= K''C\frac{1 - \left(\frac{1}{1+h\sigma}\right)^n}{1 - \frac{1}{1+h\sigma}}, \end{aligned} \tag{31}$$

where

$$\begin{aligned} K' &= P\left(\left(\frac{\sigma}{2}\right)\left\lfloor\frac{2\mu_b}{\sigma}+2\right\rfloor \le B \le \left(\frac{\sigma}{2}\right)\left\lfloor\frac{2\mu_b}{\sigma}+4\right\rfloor\right), \\ K'' &= P\left(\left(\frac{\sigma}{2}\right)\left\lfloor\frac{2\mu_b}{\sigma}+4\right\rfloor \le B \le \left(\frac{\sigma}{2}\right)\left\lfloor\frac{2\mu_b}{\sigma}+6\right\rfloor\right). \end{aligned} \tag{32}$$

Therefore, it holds that

$$\begin{aligned} &P(-\mu_b - 2.5\sigma_b \le EB_k \le -\mu_b + 2.5\sigma_b) \\ &= P(EB_K = -\mu_b) + 2P(EB_K = -\mu_b + \sigma) \\ &\quad + 2P(EB_k = -\mu_b + 2\sigma) \\ &= C(K + K' + K'')\frac{1 - \left(\frac{1}{1+h\sigma}\right)^n}{1 - \frac{1}{1+h\sigma}}. \end{aligned} \tag{33}$$

Fig. 9 shows error correction convergence with respect to different values of h, the height of each error block. To show the error correction convergence of the heading as an example, we use the collected data of the Section 5.2. The width of each error block is fixed as σ. For all cases, the probability of bias increases over time. The error correction effect appears faster as the size of the error block increases. Even with the slowest error block height of 0.1, a bias of 0.95 probability or more appears in the 13th step in Fig. 9. This corresponds to the results in Fig. 13.

4.6 DSFF

Our probabilistic fusion method searches out the most probable location l, given online measurement OE by calculating the posterior priority $P(l|OE)$ for all locations. The proposed DSFF compensates for sensor data errors by analyzing the

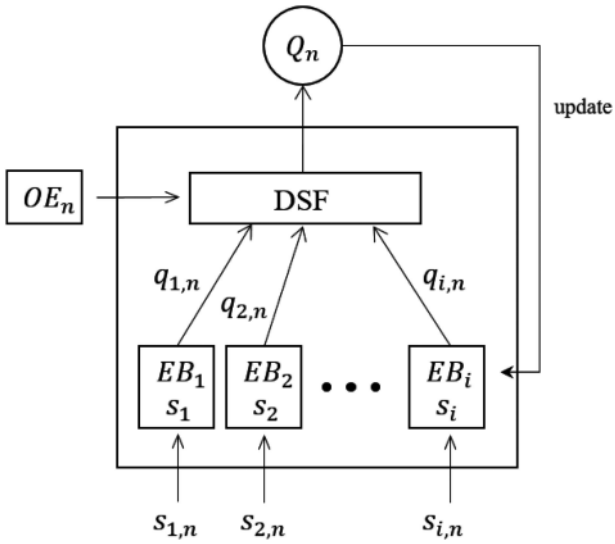

Fig. 10. DSF flow-chart on occurrence of outliers.

error in accumulated sensor data over time as it searches for the optimal trace using the HMM. The HMM is a suitable model for sequential indoor positioning data because it can handle the temporal nature of the data and deduce the desired information from the information. The HMM-based positioning algorithm operates by determining the hidden structure of user traces that fits into the building's inner layout. In general, an HMM estimates a current state at each time point by utilizing observations, HMM properties, and parameters. Although the current state is hidden at each time point, it can be estimated by observations together with the previous state. Indoor localization systems employ HMM by replacing states with locations and observations with online measurements.

The proposed DSFF uses a k-best Viterbi tracking algorithm [36], [37] in the HMM framework. Viterbi tracking is an interpretation framework that efficiently computes position changes using the HMM, which allows historical and dynamic user movement trajectories to be tracked stochastically. Probabilistic Viterbi tracking is performed using absolute and relative positioning. The optimal trace is found by calculating the user position with the highest probability. In other words, the optimal trace is the trace with the highest probability and minimal errors in the various sensor data.

4.7 Environmental Adaptation of the DSF

To determine which data of the corrected probability distribution should be considered for Viterbi tracking, we used modified exponential smoothing (ES), which weights the most recent data while still using all data. When using the EB for tracking, if we use the entire history, it is less reliable because the EB considers the average environment of the entire trajectory rather than considering the most recent information about a constantly changing environment. Conversely, if we only consider recent error distributions, the amount of data is insufficient for error correction, and the tracking accuracy may be low. Therefore, we use a modified ES which we call the aging method to consider the entire history of the EB, but to give a higher weight to the more recent data.

ES is an algorithm for predicting future data from existing data, which gives a higher weight to recent data and applies past historical data to exponential weighting to predict the data average. Because the proposed DSFF does not need to predict the future, only the part that weights the tracking history is used. It is calculated as follows:

$$EB_{aging} = \alpha EB_n + \alpha(1-\alpha)EB_{n-1} + \alpha(1-\alpha)^2 EB_{n-2} + \cdots + \alpha(1-\alpha)^{n-1} EB_{n-m}, \quad (34)$$

where N is the number of EBs, EB_n is an observed EB value at time n, and α is an ES coefficient, which is set 0.2. In general, the ES coefficient is small for a time series with large irregular fluctuations and large for a time series with small irregular fluctuations. A value of 0.2 was chosen by comparing the residual sum of squares between the actual observations and predicted values to obtain the smallest prediction error.

4.8 Effectiveness of Outliers

In the previous sections, we assumed that more sensory data enhances positioning accuracy. At this point, we can consider if only accurate data is present, then extreme sensor noise and outliers may damage the system. This consideration applies to the general fusion method. However, a single sensor which exhibits a large error is not significantly affects to the tracking result of the system because the remaining sensors complement this one sensor in most fusion methods. In other words, if we use sensor fusion, the probability of obtaining inaccurate results is much lower compared to when we use only one sensor. This is because the directionality of each sensor error pattern varies. The error of each sensor is influenced by the inherent nature of the sensor, and the environment that affects each sensor also varies. For these reasons, the sensor errors vary in size and orientation in 2D spaces. For example, in typical fusion, when sensor A shows an error of -5 m on the x-axis and sensor B with an error of +3 m, the sensor fusion result can be -1 m as these two sensors can complement each other. The proposed method can be more robust in a situation where one sensor shows a large error. DSFF continuously learns the error pattern for each sensor and performs sensor fusion with a real-time correction value depending on this pattern. In other words, if sensor A frequently shows minus error on the x axis, it will calibrate the negative sensor reading closer to zero.

The sensor reading $s_{i,n}$ at the nth stage is subjected to an error correction process and estimates the corrected probabilistic distribution $q_{1,n}, q_{2,n}, , q_{i,n}$ as the output in Fig. 10. If an extremely large error Z occurs in a single sensor (i.e., $Z_i > 2\sigma_i$), the sensor reading $s_{i,n}$(=$o_{i,n}$) becomes $s_{i,n} = o_{i,n} + Z$ and there will be no outlier in case of the $Z = 0$. Here, Z denotes the difference between the outlier value and the μ value of the distribution of the respective sensor. These outputs act as input of the DSF along with the probability distribution of absolute sensors, resulting in a corrected result Q_n. In this case, $Q_n - q_{i,n}$ denotes the error of the single sensor at the nth stage, and the error block for each sensor at the next stage is updated through this error. When an outlier occurs in a single sensor, the effect of this single sensor on the result at the nth stage is proportional to Z_i/σ_i. At this time, the degree to which the overall result is affected by this sensor

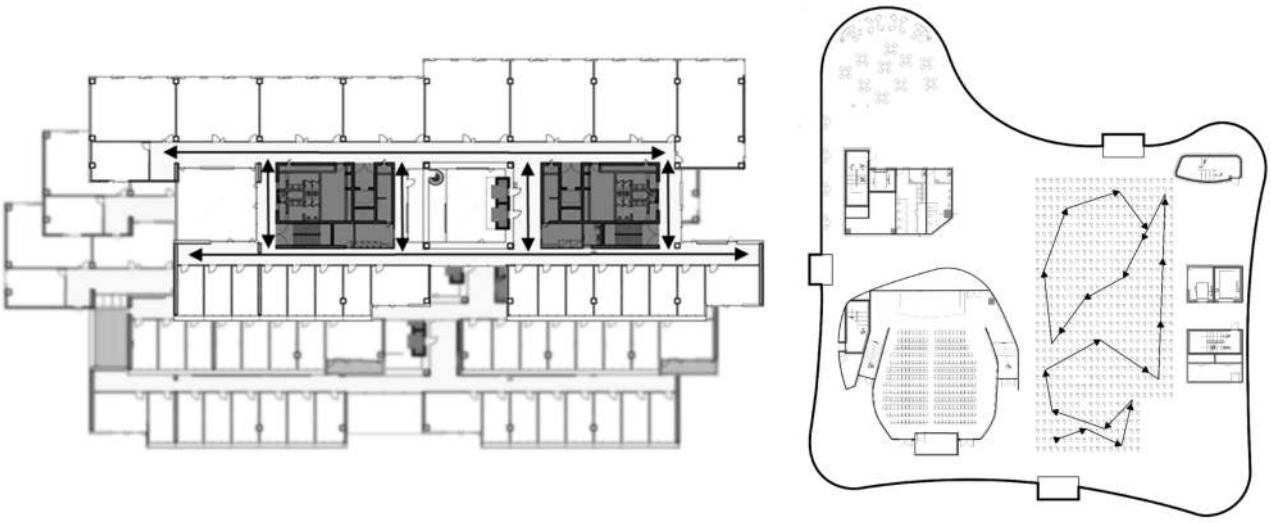

Fig. 11. Experimental area at KAIST, Korea: (a) Testbed N1: Seventh floor of the N1 building and (b) testbed KI: Open space of a lobby.

can be represented by $M = (\frac{Z_i}{\sigma_i}) \times (\frac{\prod \sigma}{\sigma_i})$, and the corrected result value can be expressed as follows:

$$Q'_n = Q_n + M. \tag{35}$$

If the outlier is not included, the effect of the final result at the nth stage on the single-sensor error block update at the $n+1$th stage is as follows:

$$EB_{i,n+1} = (1-\alpha)EB_{i,n} + \alpha(Q_n - q_{i,n}). \tag{36}$$

If the outlier is included

$$EB'_{i,n+1} = (1-\alpha)EB_{i,n} + \alpha(Q'_n - q_{i,n}). \tag{37}$$

On summarizing the effects of including outliers, we can derive (38) from (35) and (37)

$$\begin{aligned} EB'_{i,n+1} &= (1-\alpha)EB'_{i,n} + \alpha(Q'_n - q_{i,n} + M) \\ &= (1-\alpha)EB'_{i,n} + \alpha(Q'_n - q_{i,n}) + \alpha M. \end{aligned} \tag{38}$$

Therefore, deriving from (36) and (38) to (39) shows that when the final result is affected by M due to a single- sensor outlier at the nth stage, the influence of the error block on each single sensor at the $n+1$th stage is proportional to α and M as follows:

$$EB'_{i,n+1} = EB_{i,n+1} + \alpha M. \tag{39}$$

However, the DSF may fail if the sensor outlier continues to appear as an extreme error that deviates from the existing pattern. If outliers occur continuously and (38) repeats repeatedly, M accumulates and Q'_n continuously worsens. In addition, if EBs are accumulated randomly and continuously, the probabilistic distribution of the sensors can form an even distribution, and this situation is the same as when the error is included but not error-corrected.

5 Evaluation

5.1 Experiment Setup and ERM Construction

The experiments were conducted in medium-scale office buildings, N1 and KI, at the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea. We evaluated the performance of the DSFF by dividing the target site into a 2D space and a 3D space with floor movement. Fig. 11a depicts the experimental area in the N1 building, which includes three floors, and Fig. 11b shows the experimental open space in the KI building. The specifications of the testbed and their ERM constructions are listed in Table 1.

Test bed N1, which consists of corridors and rooms, is a typical configuration of an office building. The tracking accuracy test uses Wi-Fi and a magnetometer for absolute positioning and a gyroscope, accelerometer, and barometer for relative positioning. To construct the ERM for the target area, HMM cells of the building shown in Fig. 11 were separated into 1m intervals and training data were collected using a point-by-point collection method. A total of 1,000 fingerprints were collected for 20 Wi-Fi data per cell. In the ERM construction, a fingerprint for each cell was stored after calculating the average measurement, and a total of 500 fingerprints were used in the N1 building.

TABLE 1
Experimental Setup

Testbed	Area	Total length of corridors (m)	#APs detected (used)	Data collection			
				Training data		Test data	
				# traces	# measurements Wi-Fi + Mag.	# traces	# measurements
N1	$(80 \times 32)\ m^2 \times$ 3floors	196	196 (50)	4	10,000 + 250,000	40	7,200
KI	$(18 \times 36)\ m^2 \times$ 1floor	83	206 (all)	1	523 + 8,442	1	621

TABLE 2
Summary of Experiment Methods and Results in 2D Space

Notation	Condition	Mean Err. Dist. (m)
HMM	Tracking without DSF	3.19
HMM+	Tracking with accurate distance data in Fig. 12	3.00
	Tracking with accurate heading data in Fig. 13	2.25
	Tracking with accurate heading and distance data in Fig. 14	2.07
DSF	DSF adaptation to distance data in Fig. 12	3.08
	DSF adaptation to heading data in Fig. 13	2.70
	DSF adaptation to both heading and distance data in Fig. 14	2.52

A total of 196 APs were detected, but 50 APs were used so that a Wi-Fi positioning accuracy of 3.5 m $\sim$ 4 m was obtained. In general, APs are selected to improve performance through dimensional reduction. However, in this experiment, AP selection was performed to more clearly evaluate the effect of DSFF when tracking. About 500 values of magnetometer data were collected in each cell so that the total number of data is 250,000. To construct the ERM for the magnetometer, the norm values of the x, y, and z axes were assigned to the magnet intensity and the averages were used along with the inclination values. The average values of Wi-Fi and magnetic field in each HMM cell of the target site were stored as training data along with the location coordinates during the construction of the ERM. Because DSFF uses probability-based positioning, the signal distribution of the collected training data was analyzed and used in the online phase. For accuracy tests, the test point coordinates were registered in the test data and the accuracy was calculated using the difference in distances between the ground truth and test point. All simulation codes for the evaluation were implemented in Java and run on a 3.30 GHz Intel Core i5-6600 CPU with 16 GB of memory.

5.2 Tracking Accuracy Test

We compared the tracking accuracy achieved by the DSFF with that of a model built in a supervised manner using the ground truth location labels. The average error distances were measured according to the time sequences of the test data. Test data for the Wi-Fi, magnetometer, accelerometer, gyroscope and barometer along with learning data were collected in the N1 building for the tracking accuracy test. Test data were collected for 40 test paths, and about 180 fingerprints were included in each test path. A total of 7,200 fingerprints were used for the test data. In the experiments, the absolute sensor accuracy is in all cases compared to the ground truth. Therefore, there is still an error in all cases even in case of HMM+. The real truth is calculated from the ground truth, and the criterion of correction we mentioned in this paper is calculated using the aggregate result of proposed method. Consecutive position is a consecutive state of HMM. Tracking is performed by continuously calculating the state at t by multiplying the emission probability and transition probability at t-1. The heading computing of the consecutive position can be derived using coordinates between states, and indoor layout with the probabilities.

In this section, we consider the case of tracking DSFF while correcting only heading data, correcting only step data, correcting heading and step data at the same time, and performing no correction. These variations are denoted by DSF and HMM, as per the experimental condition. The abbreviations and mean distance error for the methods in this experiment are summarized in Table 2. The accuracy is compared by evaluating the cases in which where there is no error in the heading data, no error in the step data, and no error in either the heading or step data. The latter case is denoted by HMM+, as per the experimental condition. (Note that the purpose of the experiment is to evaluate how the proposed DSF method is employed in the DSFF so that the sensor values of the HMM are corrected and thus, become similar to the values of the HMM+, which has no relative sensor errors.).

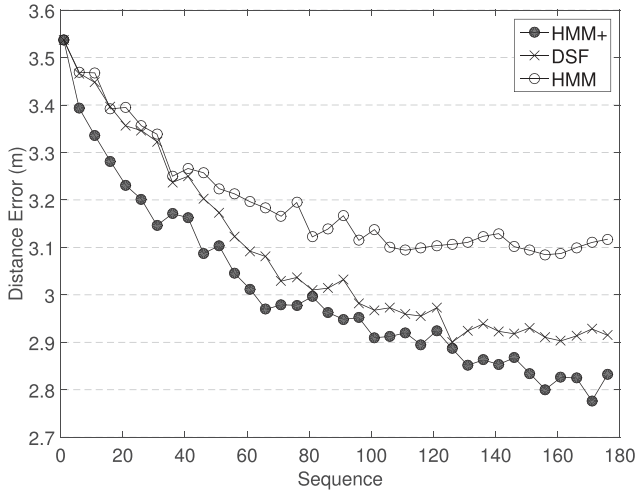

Fig. 12. Comparison of distance-dependent tracking.

Fig. 12 presents the tracking accuracy with respect to the distance data. The tracking accuracy of the HMM converges at 3.1 m, DSF converges at 2.92 m, and HMM+ converges at 2.85 m. DSF achieves an improvement in accuracy of 6.45 percent compared with HMM. Although the accuracy is further improved by HMM+, the difference in accuracy between HMM+ and HMM is only about 0.4 m.

Fig. 13 presents the tracking accuracy with respect to the heading data. In contrast to the results for distance, HMM converges at 3.1 m, DSF converges at 2.4 m, and HMM+ converges at 2 m in Fig. 13. Compared with HMM, DSF converges with a 22.5 percent improvement in accuracy. Comparing HMM+ of Fig. 12 and HMM+ of Fig. 13, HMM+ in Fig. 13 converges with an 0.8 m improvement. DSF of Fig. 12 and DSF of Fig. 13 also show a similar pattern whereby DSF in Fig. 13 converges to an 0.5 m improvement in accuracy.

Because the probability distribution of distance forms a ring shape, as shown in Fig. 1, all HMM cells to which the user can move from a candidate point have the same probability. Therefore, the tracking accuracy of the distance data is lower than that of the heading data. For the heading data, HMM cells with the same probability are limited to the distribution of a fan shape. Consequently, even if the travel distance is not accurately calculated, the traveling direction has a relatively high influence on the tracking result.

DSF in Fig. 12 and DSF in Fig. 13 have similar accuracy to HMM before sequences 40 and 20, respectively. Although the EB is generated for the sensor bias and applied to the

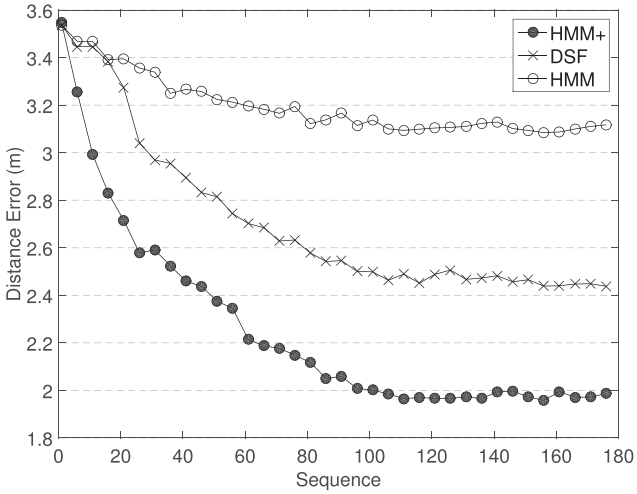

Fig. 13. Comparison of heading-dependent tracking.

probability distribution, it does not substantially improve the accuracy as compared to HMM. However, the accuracy is improved after sequence 40 for DSF in Fig. 12 and sequence 20 for DSF in Fig. 13. This means that the effect of error correction begins to appear near these sequences (40 and 20, respectively), and we can see that accuracy continues to improve up to sequence 140 and 100, respectively, where the improvement appears to reach its maximum. In the case of DSF in Fig. 12, the improvement in accuracy with respect to DSF in Fig. 13 is less than 0.2 m. This is because the deviation in the step length is not large, even if error correction is applied. By contrast, the accuracy is highly improved from sequence 20 in the case of DSF in Fig. 13 because the weight is accumulated in the value obtained by applying EB to the heading distribution. A comparison of the results in Figs. 12 and 13 shows that the accuracy improves and degrades intermittently because the methods shown in Fig. 12 correct the distance error but not the heading error. This indicates that the bias of the heading data degrades the tracking result relatively strongly.

Fig. 14 shows the tracking accuracy of the simultaneous correction of step and heading sensor data. DSF converges at 2.2 m and HMM+ converges at 1.8 m. Here, DSF improved

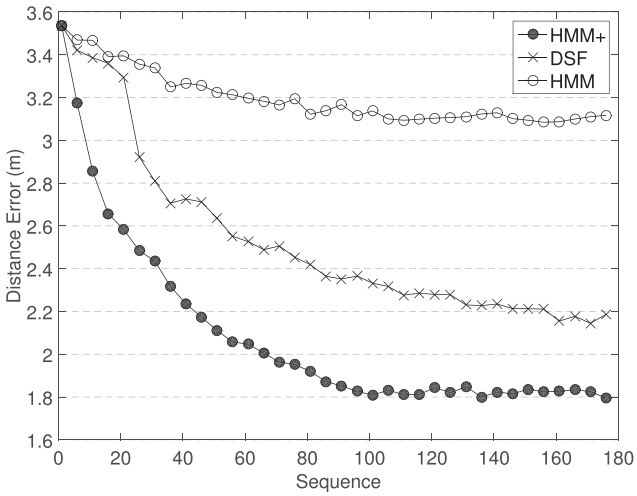

Fig. 14. Comparison of a distance- and heading-dependent tracking.

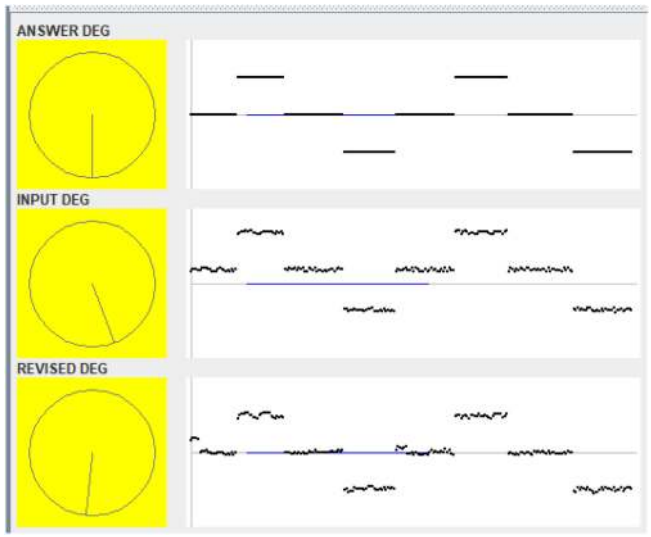

Fig. 15. This figure shows the influence of the heading error correction. Here, "ANSWER DEG" denotes HMM+, "INPUT DEG" denotes HMM, and "REVISED DEG" denotes DSF.

accuracy by about 29 percent with respect to HMM. The accuracy begins to improve as DSF moves past sequence 20, which is similar to the results for DSF in Fig. 13. Fig. 15 shows how the heading errors are corrected in the test trace, where "ANSWER DEG" denotes HMM+, "INPUT DEG" denotes HMM, and "REVISED DEG" denotes DSF. Compared to HMM, all values of HMM are biased to positive values, indicating that gyro drift is occurring. On the other hand, DSF is not corrected in the early stage, but the correction rapidly occurs at a certain time point. The points at which the accuracy improves are similar to those in Fig. 14. Although it is not completely error-corrected, the DSF values are corrected to a value similar to the correct answer as a whole, compared to those of HMM, and the accuracy is improved, as shown in Fig. 14, through error correction. Even in HMM(+), where there is no relative sensor error in the sensor data, HMM+ in Fig. 14 is improved by 35.71 and 10 percent compared to HMM+ in Fig. 12 and HMM+ in Fig. 13, respectively. The relatively low improvement rate of HMM+ in Fig. 13 with respect to HMM+ in Fig. 12 confirms that the moving direction of a pedestrian has a higher effect on the tracking result. In this regard, if the conditional probability of the moving direction and the moving distance is applied, the accuracy of DSF as well as that of HMM+ in Fig. 14 improves.

5.3 Floor Detection Evaluation

The test path for the floor detection experiment moves between three levels in the N1 building, and the test data was collected four times between the highest level and the lowest level. In the floor detection test, the plane detection space was set to about 30 percent, and the floor detection space was set to about 70 percent of the whole test space, so that the effect of the barometer on the floor classification can be clearly seen in the DSFF. A total of 800 test data were used, including about 80 fingerprints per 10 test paths.

Fig. 16 compares the cases in which the accuracy varies depending on the barometer data. In the figure, HMM converges at 3.4 m, DSF converges at 3.1 m, and HMM+ converges at 2.8 m. Compared to the case of tracking without

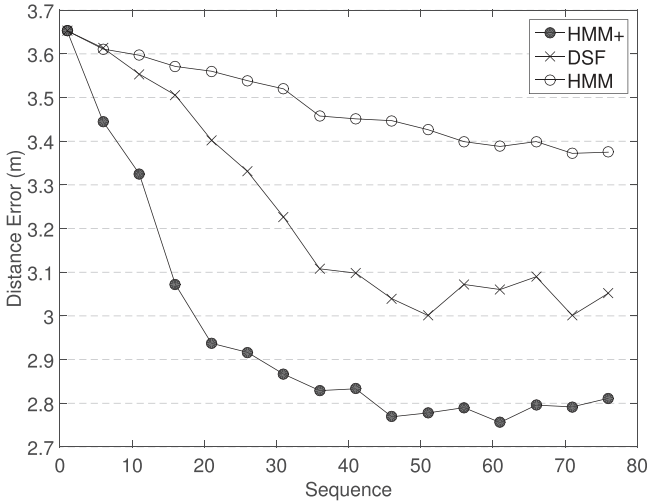

Fig. 16. Comparison of a barometer-dependent tracking.

error compensation, DSF improves accuracy by 6.45 percent. Also, although there was an improvement, the difference in accuracy between HMM+ and HMM was not substantial (about 0.4 m). Fig. 17 shows the result of tracking using heading, distance, and pressure data, and the average accuracy of each case is shown in Table 3. In Figs. 16 and 17, HMM+ converges at 2.8 and 1.7 m, respectively. This indicates that even if the altitude value is fused correctly using DSF, the tracking accuracy is not substantially improved unless the direction and distance values are correct. This is because the probability distribution of barometric pressure only varies over the z coordinate, and the x and y coordinates depend on the positioning results of the other sensors.

By contrast, DSF improves accuracy from sequence 10 in Fig. 16. In the case of the floor displacement test, it is possible to perform relatively accurate calculations with absolute positioning using (25). Hence, if error correction is applied to the probability distribution of the barometer data, the improvement may be relatively faster than for other sensors. As shown in Fig. 17, the rapid accuracy improvement occurs near sequence 20 of DSF. Similar to Figs. 13 and 14, this phenomenon is due to the heading error correction.

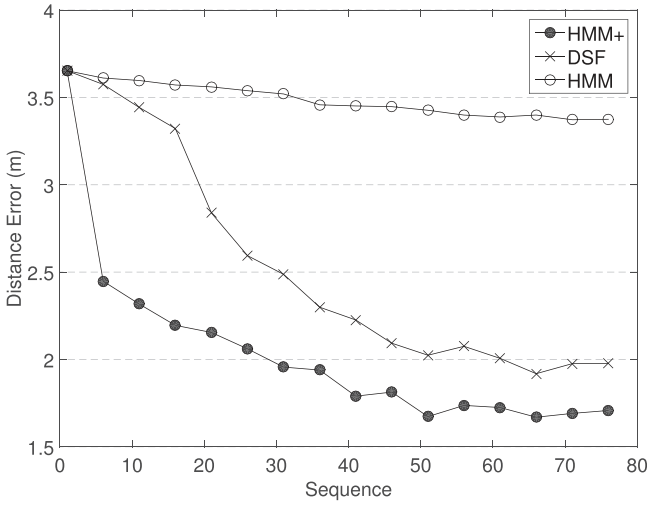

Fig. 17. Comparison of a distance-, heading- and barometer-dependent tracking.

TABLE 3
Summary of Experiment Methods and Results in 3D Space

Notation	Condition	Mean Err. Dist. (m)
HMM	Tracking without DSF	3.48
HMM+	Tracking with accurate barometer data in Fig. 16	2.95
	Tracking with accurate data for all sensors in Fig. 17	1.99
DSF	DSF adaptation to barometer data in Fig. 16	3.23
	DSF adaptation to all sensor data in Fig. 17	2.45

5.4 Tracking Accuracy Test for Open Space

The open space experiment was conducted in the first floor of the KI building, to evaluate the tracking accuracy of the DSFF and HMM. The performance was measured for a random walk path in the 18 × 36 m lobby of the KI building, where the average error distance is 4.34 m when Wi-Fi alone is used for positioning. Moreover, the accuracy is not greatly improved even if the IMU sensor is used.

Fig. 18 shows the random walk path using blue points, and the red points show the estimated points. Fig. 18a shows HMM tracking and Fig. 18b shows DSF using distance and heading data tracking. The mean errors for these methods are 4.73 and 4.29 m, respectively. In this space, it is difficult to improve the accuracy using the IMU sensors when error correction is not performed. Even if error correction is performed, the accuracy is not dramatically improved. Nonetheless, Fig. 18 shows that DSF can also affect the user tracking accuracy, even in an open space. As shown in Fig. 18a, the estimated points are scattered without any pattern in the black solid circle. In Fig. 18b, the estimated points at the lower part of the solid circle are spread out to the left and right, and the estimated points are approximately linear in the upper part. Although they differ from the ground truth, the estimated points more closely follow the actual moving pattern when DSF is used.

In the dotted circle in Fig. 18a, the heading is shifted continuously to the right because of the accumulated error, but

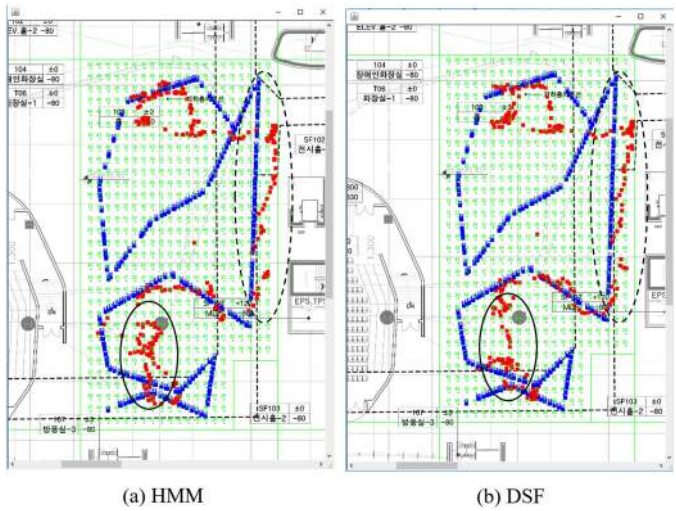

Fig. 18. Comparison of tracking (a) without and (b) with DSF in an open space. Blue and red points indicate the ground truth and estimated points, respectively.

the DSF results are closer to the ground truth in Fig. 18b. In other words, even if it is difficult to expect a good positioning result due to a decrease in positioning accuracy when using a radio signal in an open space, accuracy can be improved by performing error correction using an aggregated result.

6 Discussion

The main purpose of the framework proposed in this paper is to compensate for the degradation of tracking accuracy by providing a fusion mode that can be used as a universal fusion system. This aim was accomplished by developing a real-time error-learning model and adaptive fusion framework. The proposed error-learning model suggests that it is possible to correct the errors of sensor data by reflecting the characteristics of a continuously changing environment in real time. In contrast, other fusion methods require complicated calculations and unfamiliar fusion models. The proposed framework is more robust when fusing new sensors in indoor positioning. We have developed a framework that can provide a navigation service without complicated calculations or processing by easily and simply combining new sensors using a probability-based fusion model. In other words, the framework developed in this study enables a navigation service to be used more accurately in various environments.

The experiments conducted under a changing environment revealed that the proposed DSFF can provide a precise fusion model without a static or empirical setting. In an office building, the tracking accuracy was improved by at least 3.45 percent and up to 28.95 percent in the 2D space during tracking. In the floor displacement experiment, the tracking accuracy was improved by at least 7.18 percent and up to 29.59 percent. These results imply that an indoor navigation service would be possible through dynamic sensor fusion, even under the continuously changing indoor environments that exist in most spaces of a building.

In contrast, in an open space, using a diverging signal as a characteristic for a specific location in indoor positioning using Wi-Fi remains difficult. As with HMM-based positioning, the proposed method also performs poorly if an indoor layout is not used. Although these limitations exist, we have confirmed that the proposed method can be calibrated along the ground truth in the open space. This indicates that the proposed method can provide guidance to positioning studies in open space in the future. Moreover, there is a trade-off between the amount of data accumulated and the agility of the error correction adaptation, as mentioned in Section 4.7. Although ES applies temporal significance to Viterbi tracking data, additional research is needed to determine the importance of data over time and thus, optimize the DSFF developed here.

7 Conclusion

This paper presents a dynamic sensor fusion framework to combine various sensors and provides a fusion model for fusion-based indoor navigation services. Although it is not possible to fully remove the error and bias characteristics of each sensor, it has been confirmed that a few sensors such as Wi-Fi and magnetometer can compensate for the errors or biases of various other sensors by complementing their functionality. Although the characteristics of each sensor are different because of the variety of sensors and vendors, the proposed method enable the fusion of sensor data in real time. The proposed fusion system will become a critical technique for accurate user tracking. We are considering integrating the proposed system into the KAIST Indoor Locating System (KAILOS) [27], and providing the sensor fusion framework and maintenance service to the public.

Meanwhile, reducing site survey effort has long been an issue in indoor positioning. The data used for offline training can cause different signal measurements, even at the same location, due to the addition or removal of infrastructure over time. In the dynamic sensor fusion method, which is relatively dependent on the accuracy of absolute sensor data being trained offline, this problem becomes even more critical. To reduce this effort, methods of crowdsourcing [38], automatic updating of training data [39], [40], and user feedback [41] have been studied. Another typical solution for reducing site survey effort is to use additional sensors to capture signal changes [42]. Future research could focus on finding a suitable method for reducing site survey effort and a more sophisticated method for aging that works well with the proposed fusion framework.

Acknowledgments

This work was partially supported by the National Research Foundation of Korea (NRF) Grant funded by the Korea government (MSIP) under Grants 2018R1A2A1A05078018 and No. 2018X1A3A1068603, and this work was supported by the research 'Development of Location-based Blockchain Sytstem' under Grant 20180002380021001 funded by the Institute for Information communication Technology Planning & evaluation (IITP).

References

[1] C. Yang, T. Nguyen, and E. Blasch, "Mobile positioning via fusion of mixed signals of opportunity," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 29, no. 4, pp. 34–46, Apr. 2014.

[2] T. Gaedeke, M. Johnson, M. Hedley, and W. Stork, "Fusion of wireless ranging and inertial sensors for precise and scalable indoor localization," in *Proc. IEEE Int. Conf. Commun. Workshops*, 2014, pp. 138–143.

[3] A. Colombo, D. Fontanelli, D. Macii, and L. Palopoli, "Flexible indoor localization and tracking based on a wearable platform and sensor data fusion," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 4, pp. 864–876, Apr. 2014.

[4] S. Kumar and R. M. Hegde, "Multi-sensor data fusion for indoor localization under collinear ambiguity," *Pervaive Mobile Comput.*, vol. 30, pp. 18–31, 2016.

[5] P. Kumar, H. Gauba, P. P. Roy, and D. P. Dogra, "Coupled HMM-based multi-sensor data fusion for sign language recognition," *Pattern Recognit. Lett.*, vol. 86, pp. 1–8, 2017.

[6] K. Bernardin, K. Ogawara, K. Ikeuchi, and R. Dillmann, "A sensor fusion approach for recognizing continuous human grasping sequences using hidden Markov models," *IEEE Trans. Robot.*, vol. 21, no. 1, pp. 47–57, Feb. 2005.

[7] M. Dong and D. He, "Hidden semi-Markov model-based meth-odology for multi-sensor equipment health diagnosis and prognosis," *Eur. J. Oper. Res.*, vol. 178, pp. 858–878, 2007.

[8] Z. Chen, H. Zou, H. Jiang, Q. Zhu, Y. Chai Soh, and L. Xie, "Fusion of WiFi, smartphone sensors and landmarks using the Kalman filter for indoor localization," *Sensors*, vol. 15, pp. 715–732, 2015.

[9] G. Ligorio and A. M. Sabatini, "A novel Kalman filter for human motion tracking with an inertial-based dynamic inclinometer," *IEEE Trans. Biomed. Eng.*, vol. 62, no. 8, pp. 2033–2043, Aug. 2015.

[10] G. Rigatos and S. Tzafestas, "Extended Kalman filtering for fuzzy modeling and multi-sensor fusion," *Math. Comput. Model. Dynamical Syst.*, vol. 13, no. 3, pp. 251–266, 2007.
[11] J. K. Suhr, J. Jang, D. Min, and H. G. Jung, "Sensor fusion-based low-cost vehicle localization system for complex urban environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 5, pp. 1078–1086, May 2017.
[12] Z. Wu, E. Jedari, R. Muscedere, and R. Rashidzadeh, "Improved particle filter based on WLAN RSSI fingerprinting and smart sensors for indoor localization," *Comput. Commun.*, vol. 83, no. 1, pp. 64–71, 2016.
[13] D. Zhou, K. Zhang, A. Ravey, F. Gao, and A. Miraoui, "Online estimation of lithium polymer batteries state-of-charge using particle filter-based data fusion with multimodels approach," *IEEE Trans. Ind. Appl.*, vol. 52, no. 3, pp. 2582–2595, May/Jun. 2016.
[14] X. He, D. N. Aloi, and J. Li, "Probabilistic multi-sensor fusion based indoor positioning system on a mobile device," *Sensors*, vol. 15, pp. 31464–31481, 2015.
[15] M. K. Hoang, J. Schmalenstroeer, C. Drueke, D. H. Tran Vu, and R. Haeb-Umbach, "A hidden Markov model for indoor user tracking based on WiFi fingerprintig and step detection," in *Proc 21st Eur. Singnal Process. Conf.*, 2013, pp. 1–5.
[16] J. Liu, R. Chen, L. Pei, R. Guinness, and H. Kuusniemi, "A hybrid smartphone indoor positioning solution for mobile LBS," *IEEE Sensors*, vol. 12, no. 12, pp. 17208–17233, Dec. 2012.
[17] A. M. Ladd, K. E. Bekris, A. P. Rudys, D. S. Wallach, and E. E Kavraki, "On the feasibility of using wireless ethernet for indoor localization," *IEEE Trans. Robot. Autom.*, vol. 20, no. 3, pp. 555–559, Jun. 2004.
[18] A. Akbar and J. S. Farrokh, "A Kalman filter-based framework for enhanced sensor fusion," *IEEE Sensors*, vol. 15, no. 6, pp. 3281–3292, Jun. 2015.
[19] D. Gusenbauer, C. Isert, and J. Kroche, "Self-contained indoor positioning on off-the-shelf mobile devices," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat.*, 2010, pp. 1–9.
[20] Y. Jin, H. S. Toh, W. S. Soh, and W. C. Wong, "A robust dead-reckoning pedestrian tracking system with low cost sensors," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun.*, 2011, pp. 222–230.
[21] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proc. 20th Annu. Simp. Comput. Geom.*, 2004, pp. 253–262.
[22] Y. Ni, J. Liu, S. Liu, and Y. Bai, "An indoor pedestrian positioning method using HMM with a fuzzy pattern recognition algorithm in a WLAN fingerprint system," *IEEE Sensors*, vol. 16, no. 9, Sep. 2016, Art. no. 1447.
[23] L. Binghao, B. Harvey, and T. Gallagher, "Using barometers to determine the height for indoor positioning," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat.*, 2013, pp. 23–31.
[24] M. N. Berberan, E. N. Bodunov, and L. Pogliani, "On the barometric formula," *Amer. J. Physics*, vol. 65, pp. 404–412, 1997.
[25] NASA, U.S. "Standard Atmosphere 1976," D.C., U.S. Government Printing Office, 1976, pp. 1–243. [Online]. Available: https://ntrs.nasa.gov/search.jsp?R=19770009539
[26] H. Xia, X. Wang, Y. Qiao, J. Jian, and Y. Chang, "Using multiple barometers to detect the floor location of smartphones with built-in barometric sensors for indoor positioning," *Sensors*, vol. 15, pp. 7857–7877, 2015.
[27] D. Han, S. Lee, and S. Kim, "KAILOS: KAIST indoor locating system," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat.*, 2014, pp. 615–619.
[28] J. Wang, Y. Gao, Z. K. Li, X. L. Meng, and C. M. Hancock, "A tightly-coupled GPS/INS/UWB cooperative positioning sensors system supported by V2I communication," *Sensors*, vol. 16, 2016, Art. no. 944.
[29] M. Y. Wu, J. C. Ding, L. Zhao, Y. Y. Kang, and Z. B. Luo, "An adaptive deeply-coupled GNSS/INS navigation system with hybrid pre-filters processing," *Meas. Sci. Technol.*, vol. 29, 2018, Art. no. 025103.
[30] J. Song, B. Huang, and R. Yang, "An experimental study on measurement errors of inertial sensors in smartphones," in *Proc. Chin. Control Decision Conf.*, 2018, pp. 4819–4824.
[31] M. Alfakih, M. Keche, and H. Benoudnine, "Gaussian mixture modeling for indoor positioning WiFi systems," in *Proc. 3rd Int. Conf. Control Eng. Inf. Technol.*, 2015, pp. 1–5.
[32] W. Shao, F. Zhao, C. Wang, H. Luo, T. M. Zahid, Q. Wang, and D. Li, "Location fingerprint extraction for magnetic field magnitude based indoor positioning," *J. Sensors*, vol. 2016, 2016, Art. no. 1945695.
[33] J. Rantanen, L. Ruotsalainen, M. Kirkko-Jaakkola, and M. Makela, "Height measurement in seamless indoor/outdoor infrastructure-free navigation," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 4, pp. 1199–1209, Apr. 2019.
[34] C. Combettes and V. Renaudin, "Walking direction estimation based on statistical modeling of human gait features with handheld MIMU," *IEEE/ASME Trans. Mechatronics*, vol. 22, no. 6, pp. 2502–2511, Dec. 2017.
[35] K. P. Subbu, B. Gozick, and R. Dantu, "LocateMe: Magnetic-fields-based indoor localization using smartphones," *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 4, pp. 73:1–73:27, 2013.
[36] S. H. Jung, G. Lee, and D. Han, "Method and tools to construct a global indoor positioning system," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 6, pp. 906–919, Jun. 2018.
[37] D. G. Brown and D. Golod, "Decoding HMMs using the k best paths: Algorithms and applications," *BMC Bioinf.*, vol. 11, no. 1, 2010, Art. no. S28.
[38] S. H. Jung and D. Han, "Unsupervised learning for crowdsourced indoor localization in wireless networks," *IEEE Trans. Mobile Comput.*, vol. 15, no. 11, pp. 2892–2906, Nov. 2016.
[39] K. M. Chang and D. Han, "Crowdsourcing-based radio map update automation for Wi-Fi positioning systems," in *Proc. 3rd ACM SIGSPATIAL Int Workshop Crowdsourced Volunteered Geographic Inf.*, 2014, pp. 24–31.
[40] S. H. Jung and D. Han, "Automated construction and maintenance of Wi-Fi radio maps for crowdsourcing-based indoor positioning systems," *IEEE Access*, vol. 6, pp. 1764–1777, 2018.
[41] J. S. Lim, W. H. Jang, G. W. Yoon, and D. Han, "Radio map update automation for WiFi positioning systems," *IEEE Commun. Lett.*, vol. 17, no. 4, pp. 693–696, Apr. 2013.
[42] Y.-C. Chen, J.-R. Chiang, H.-H. Chu, P. Huang, and A. W. Tsui, "Sensor-assisted Wi-Fi indoor location system for adapting to environmental dynamics," in *Proc. 8th ACM Int. Symp. Model. Anal. Simul. Wireless Mobile Syst.*, 2005, pp. 118–125.
[43] P. Zhou, M. Li, and G. Shen, "Use it free: Instantly knowing your phone attitude," in *Proc. 20th Annu. Int. Conf. Mobile Comput. Netw.*, 2014, pp. 605–616.
[44] S. Shen, M. Gowda, and R. R. Choudhury, "Closing the gaps in inertial motion tracking," in *Proc. 24th Annu. Int. Conf. Mobile Comput. Netw.*, 2018, pp. 429–444.

Gunwoo Lee received the MS degree in computer science and engineering from Chung-Ang University, Seoul, South Korea in 2012. He is currently working toward the PhD degree in computer science at the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea. His research interests include computer networks, indoor positioning, and machine learning and their applications to pervasive computing.

Suk-Hoon Jung received the PhD degree in information and communications engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2015, where he is currently a researcher of computer science. His research interests include machine learning, data analytics, and their application to mobile and pervasive computing.

Dongsoo Han (M'13) received the PhD degree in information science from Kyoto University, Kyoto, Japan, in 1996. He is currently a professor of computer science with the Korea Advanced Institute of Science and Technology, Daejeon, South Korea. His research interests include indoor positioning, pervasive computing, and location-based mobile applications. He is a member of the IEEE.