Deep Learning-Based Multi-Floor Indoor Tracking Scheme Using Smartphone Sensors

CHENXIANG LIN, AND YOAN SHIN, (Senior Member, IEEE)
School of Electronic Engineering, Soongsil University, Seoul 06976, South Korea
Corresponding author: Yoan Shin (yashin@ssu.ac.kr)
This work was supported by the National Research Foundation of Korea (NRF) Grant by the Korean Government through the Ministry of Science and ICT under Grant 2020R1A2C2010006.

ABSTRACT Having recently become an important research topic, indoor tracking in a multi-floor building delivers comprehensive and efficient location-based services. In this paper, we present a deep learning (DL)-based indoor multi-floor tracking scheme, that is independent of infrastructure and only uses the smartphone as a terminal device to measure and analyze the user’s mobility information. Our method detects the floor transition according to changes in barometer readings. We compiled the time-series barometer data to train the DL model, and applied the data augmentation method to avoid overfitting and data imbalances during model training. Furthermore, we developed a floor decision algorithm to process the DL model’s output and generate the floor detection result. In the proposed scheme, the smartphone’s inertial measurement unit sensors are used to measure the user’s mobility information, and pedestrian dead reckoning (PDR) is exploited to update the user’s 2D location. We implemented the multi-floor tracking by combining the floor detection algorithm with PDR. To avoid the accumulated error problem that commonly arises in the infrastructure-free approach, the calibration nodes (CN) were configured in the floor plan to correct the estimated location by matching the possible CNs during the floor transition. We conducted several experiments in multi-floor buildings to evaluate our scheme’s performance, and found that our floor detection method achieves a 99.6% average floor number accuracy, with all floor transition types (i.e., stairs, elevator) being successfully recognized. Furthermore, we compared localization performance with the conventional methods to validate the effectiveness of our approach.

INDEX TERMS Smartphone, indoor tracking, deep learning, barometer, multi-floor localization, inertial measurement unit, floor transition

I. INTRODUCTION

Over the years, localization techniques have been widely applied for location-based services (LBS). In outdoor environments, a pedestrian’s location can be estimated via wireless signals transmitted from a satellite, as seen in the global positioning system. In indoor environments, however, localization generally requires higher accuracy. Meanwhile, the occlusion of a building makes satellite signals inaccurate or unsuitable. To provide better LBS in indoor environments, technologies dedicated to indoor localization are rapidly being developed. Considering increasing smartphone penetration and various micro-electromechanical systems (MEMS) sensors integrated into smartphones, research on smartphone-based indoor localization systems is being actively conducted [1]–[3]. Smartphone-based indoor localization technologies can be broadly classified as infrastructure-dependent or infrastructure-independent technologies [4]. Infrastructure-dependent technologies which provide absolute positional information according to the strength of the signal received from the building infrastructure, include ultra-wideband (UWB) [5]–[7], Wi-Fi [8], [9], Bluetooth low energy (BLE) [10]–[12], and others. These technologies generally face issues such as path loss, shadowing, and fading. Furthermore, because these techniques depend on anchor nodes, they are not always available in specific situations, such as during disasters or power outage. As an infrastructure-independent technology, pedestrian dead reckoning (PDR) is an alternative indoor localization...
method that exploits the user’s mobility information measured by the inertial measurement unit (IMU) sensors inside the smartphone to update the location of each step. The smartphone’s IMU typically consists of an accelerometer sensor, gyroscope sensor, and magnetometer sensor that measures the inertial generated by the user’s movement. As a relative method, PDR derives the user’s current location from the previous estimation; thus, the errors (e.g., missing steps, inaccurate stride length and heading direction estimation) accumulate over time. Due to the limited accuracy of consumer-grade MEMS sensors in smartphones and the inevitable noise present in indoor environments, PDR only achieves high performance within a short tracking trajectory. To improve the performance of smartphone-based PDR methods, fusion algorithms such as Kalman filter [13]–[15], complementary filter [16], [17], and particle filter [18]–[20] have been developed to combine signals from multiple sensors according to certain criteria; thus reducing uncertainty in the results and optimizing the accuracy. Furthermore, several studies focused on components of the PDR method. Zhang et al. improved step detection by filtering and differentiating the acceleration data [21], and Wang et al. used the walking speed as an influencing factor of stride length to achieve a stable dynamic stride length estimation [22]. In [20], a particle filter and an extended Kalman filter were introduced to calibrate the output deviation of magnetometer within soft/hard iron effect environments. Although these approaches mitigate the accumulated error problem in PDR, the absence of estimated location calibration or boundary constraints still results in errors that accumulate over time.

Furthermore, whereas most buildings are multi-layered structures, the calculation result of PDR represents a two-dimensional (2D) location. In an indoor environment, the altitude information can be represented as floor level and determined by detecting floor transition events. Indoor multi-floor localization can be achieved by combining the pedestrian’s estimated altitude and 2D location. The smartphone sensor readings correspond to different patterns as the pedestrian walks in different areas, and these signals can be exploited to achieve floor detection. [23]–[25] localize the floor according to information extracted from acceleration. [23] converts the step counts and traveling time to the number of floors traveled, and [24] proposes acceleration signatures for six types of user movement, which can be used to verify the floor number. Because the signal of height change can be captured by the smartphone’s barometer, this feature was incorporated into a floor detection technique. An approach was proposed in [26] to extract the patterns between the floor levels and barometer readings according to the measurements on different floors at different times. Ye et al. in [27] built the barometer fingerprint of a building by exploiting crowd sourcing which includes the barometric pressure value for each floor, and localized the floor by matching the calibrated barometer readings with the fingerprint. Typical indoor localization scenarios involve medium to large buildings (e.g., malls, school buildings), and the barometric is sensitive to environmental conditions. Consequently, the barometer data collected from different areas may exhibit different characteristics. However, some methods only yield high accuracy in uniform environments such as closed corridors or have a large delay for determining the floor change event.

Recently, deep learning (DL) approaches have been widely applied in LBS. DL methods convert raw input features into abstract, higher-level representations by composing nonlinear modules to learn the relationship between the input and output data [28]. Owning to their powerful learning capabilities, DL methods generate efficient solutions for localization applications. Therefore, they have been utilized to extract features from received signal strength (RSS) datasets [29], [30], optimize step length calculations [31], [32] and heading direction estimation [33]. However, most DL applied in LBS involves supervised learning, which poses a challenge in terms of the costs of training data collection and labeling. Moreover, considering the computational and energy limitations of the mobile operating platform, the DL model should be low complexity and lightweight.

This study proposes an infrastructure-free DL-based indoor multi-floor localization scheme using smartphone sensors for the following purposes.

- Provide a DL-based floor transition detection approach capable of promptly delivering stable and accurate floor detection results in complex indoor environments, as well as distinguish the transition types and direction of usage.
- Realize the multi-floor tracking base on the 2D location and altitude information obtained from PDR and floor detection.
- Correct the estimated location by matching possible calibration nodes (CN) to prevent error accumulation.

To meet these goals, the proposed scheme records barometer readings as the user walks across various regions (e.g., floor, stairs), takes the time-series barometric pressure data as input data, and uses the real action of each step as label data for supervised learning. Next, a DL model is built to extract the predictive relationship between the changes in barometer readings and corresponding step actions from the shape of data, and then convert the data to floor transition results using a floor decision algorithm. To improve the performance of the DL model, we propose a data augmentation method that handles the issues of imbalanced data, overfitting, and cost of dataset preparation. According to the floor (transition) detection, we improve the PDR estimation and developed a multi-floor tracking service by combining altitude information with 2D location. Moreover, to prevent the error accumulation over time, the proposed scheme sets up CNs on each floor. Once the floor transition signal is detected, the estimated location can be corrected according to match the CNs. To ensure the accuracy of floor detection, the decision algorithm waits for the DL prediction of several steps whenever the user’s walking state changes. Thus, the proposed scheme is best suited for situations where an output delay is acceptable.
The rest of the paper is organized as follows. In Section II, we present the proposed scheme, which describes each step of the overall multi-floor localization process. The experiments and their results are analyzed in Section III, and the concluding remarks are presented in Section IV.

II. COMPONENTS OF THE PROPOSED TRACKING SCHEME

As illustrated in Figure 1, while the user walking in an indoor environment, our proposed scheme collects data using the IMU and barometer sensors in the smartphone, and feeds them into the localization engine that consists of the following three components: (1) Floor detection is performed first to detect the floor transition and roughly estimate the current user’s location. (2) PDR updates the 2D location according to the mobility measured by the IMU sensors, and the floor detection output is used to improve the PDR estimation. (3) The location correction combines the estimated 2D location with the floor transition signal to calibrate the localization by CN matching, and outputs the current location in the multi-floor building.

A. DL-BASED FLOOR DETECTION

The floor transition detection component in the localization engine operates as shown in Figure 2. In this subsection, the step action prediction scenario and data collection process, the data augmentation and preprocessing to obtain training data, the hyperparameter setting and training of the model, and the floor decision algorithm to process the output from the DL model are described in detail.

1) Step Action Prediction Scenario and Data Collection Procedure

Because atmospheric pressure is inversely proportional to height, the altitude information of the user holding a smartphone can be measured via barometer readings. In an ideal case, the barometer measurements should be similar while the user walking along a flat floor, and change significantly when descending or climbing stairs. Therefore, the action of a step can be recognized by comparing the barometric pressure values of adjacent steps (e.g., difference, slope). However, barometric pressure is easily affected by the environment features, such as changes in building structure during walking. To demonstrate the effect of environmental conditions on barometer measurements, we collected barometric data in two regions: a closed corridor, and a combination of a closed corridor and open corridor, as shown in Figure 3. We shifted the barometric values close to 0 for easy visual comparison. In the first region (blue), the barometer readings did not change significantly on the flat floor, and exhibited a clear altitude correlation; however, the second region (orange) exhibited significant variations in barometric readings even over the flat surface. Therefore, it is difficult to determine whether a change in barometer readings is caused by noise or floor transition solely using the barometric pressure values of adjacent steps. If the prior barometer readings are smooth, we presume that the current barometric fluctuation is caused by changes in altitude; whereas if the prior barometric pressure values are unstable and sloppy, and the current reading manifests a similar instability, the barometric fluctuation is likely caused by noise. In this study, the step action recognition is approached as a classification problem, and the changes in barometric readings were used to generate the probability distribution corresponding to each action.

Because the changes in barometric pressure caused by user movement are largely low-frequency, lowpass filters are generally employed in data collection to eliminate high-frequency noise and provide a better dataset for DL training. We used a method called a weight smoothing to smooth outlier data as follows [34].

$$B^d_t = (1 - \beta) \cdot B^d_{t-1} + \beta \cdot x_t,$$

where $x_t$ is the sensor reading sampled at time $t$, $B^d_t$ is the barometric pressure value after applying a lowpass filter, and $\beta$ is a weight factor to control the effects of $x_t$ on $B^d_t$. The efficiency of denoising depends on the sampling frequency and the magnitude of $\beta$. Although values of $\beta$ closer to 0...
indicate a more significant denoising effect, they also cause considerable delay. During data collection, the corresponding barometer data is stored in the smartphone database whenever a new step is detected. When floor transition occurs, the current number of steps is recorded to label the collected data. For example, if a user goes upstairs at the 50th step and back to a flat floor at the 70th step, the ground-truth label of the 50th to 70th steps is “Go upstairs.” Data were collected in Hyeongnam Engineering Building at Soongsil University.

2) Data Augmentation and Preprocessing
In DL, the reliability and quality of the dataset directly determines the model’s performance. For example, imbalanced data leads to a heavy focus on the majority class, which creates an underfitting problem for the minority class samples. In contrast, insufficient learning data causes an overfitting problem, as it is difficult to procure comprehensive representative information from a limited dataset. These problems are common in DL because data collection is expensive, and the amount and quality of collected data are often lower than expected, which causes machine learning algorithms to underperform. We identify four events in the pedestrian floor transition: (1) from floor to downstairs, (2) from downstairs to floor, (3) from floor to upstairs, and (4) from upstairs to floor. For example, when a user ascends the stairs from the first floor to the second floor, they first transition from floor to upstairs, and then go from upstairs to floor. The number of events for the four types in the dataset should be similar to prevent unbalanced performance in the model (e.g., achieves a good action recognition accuracy for upstairs steps but inaccurate recognition for downstairs steps). Therefore, we utilized a data augmentation method to prevent the deterioration of learning performance caused by insufficient and imbalanced data as well as reduce the cost of data collection.

The international barometric formula, which constructed a relationship between height \( h \) and barometric pressure, is given by [35]

\[
 h(p, p_0) = 44330 \cdot \left(1 - \frac{p}{p_0}\right)^{\frac{1}{5.26}} , 
\]

where \( p \) and \( p_0 \) are the measured pressure from the barometer and reference pressure at sea level in mbar, respectively. Although the calculated absolute altitude is inaccurate owning to atmospheric pressure drift arising from weather, (2) shows that the pressure changes caused by ascending and descending for the same distance are almost identical. According to this symmetric feature, the decreased pressure data caused by going upstairs can be augmented by subtracting the increased pressure caused by going downstairs. Thus, the data augmentation in the proposed scheme was performed as follows.

\[
\delta_i = B^d_{i+1} - B^d_i , 
\]

(3)

\[
B^a_i = B^d_N - \delta_1 , 
\]

(4)

\[
B^a_i = B^a_{i-1} - \delta_i . 
\]

(5)

Here, \( \delta \) is the pressure change between adjacent steps, \( N \) is the total number of collected data, and \( B^a \) is the barometer data obtained by data augmentation. Subsequently, the training dataset \( B \) was obtained by concatenating \( B^d \) and \( B^a \). By performing data augmentation, the sample size of the dataset almost doubled, and the balance of the dataset was maintained.

Owing to the large values and small variation of barometric pressure, the training data need to be converted into forms suitable for learning through preprocessing. (6) and (7) show two feature scaling methods commonly used in data processing, min-max normalization and standardization, respectively. Normalization scales the data into a specific range such as \([0,1]\) according to the maximum and the minimum values of features, while standardization transforms features to a distribution with zero mean and unit standard deviation. Because the collected data contains unavoidable outliers as well as the maximum or minimum values are easily replaced when the floor level updated, normalization is not suitable for this study. Standardization does not influenced by outliers since the data are not restricted into a predefined range, but it changes the magnitude scale of features. We want to retain the unit of features to express the real barometer reading changes, thus mean centering is applied in our work, as shown in (8).

\[
X^\text{norm}_k = \frac{X_k - X^\text{min}}{X^\text{max} - X^\text{min}} , 
\]

(6)

\[
X^\text{std}_k = \frac{X_k - \mu}{\sigma} , 
\]

(7)

\[
CB_k = B_k - \mu . 
\]

(8)

Here, \( \mu \) is the mean of features, and \( CB \) is the centered barometric data. Because the DL model predicts the action of a step based on the changes in barometer readings, we used the time-series pressure data to represent pressure changes.
during walking. A sliding window algorithm was used to obtain subsets $WB_k$ of the dataset as the time-series pressure data.

$$WB_k = \{CB_{k-s+1}, \cdots, CB_k\},$$

where $s$ is the sliding window size. Because the window size should be large enough to contain the pressure change information, we set $s = 15$. $WB_k$ contains the barometric pressure change in $s$ steps, corresponding to the action of $k$-th step. The proposed DL model requires $s$ barometer readings to predict an action, which means it has to wait until $s$ step data is collected at start of tracking. We will discuss how to overcome this drawback in Section II-C2.

3) Model Training and Learning Results

In this study, step action recognition is treated as a classification problem. In time-series analysis tasks, the recurrent neural network (RNN) [36] is widely employed because it includes internal memory, and can therefore consider previous inputs when making a decision. However, the multi-layer perceptron (MLP) [37] is sufficient to learn the relationship between pressure changes and the action of a step through $WB$ to provide accurate step action prediction. Moreover, MLP trains and runs faster than RNN, making it more suitable for mobile platforms. In this study, MLP was used to implement step action recognition. Figure 4 illustrates an overview of the proposed MLP model. The model takes the $WB_k$ as input features, and outputs the probability of the three classes (i.e., “Normal walking”, “Going up”, “Going down”) regarding the $k$-th step after passing through hidden and output layers.

The parameters used in the DL model are listed in Table 1. There were 677 samples collected for model training, and the data size were extended to 1,125 after performing the data augmentation. Because the collected sensor data include outliers, the robustness to perturbations of the DL model should be considered in model training. Therefore, the batch normalization layer was inserted after each hidden layer to maintain neuron activations at unit variance and zero mean by shifting and scaling operations, and the scaled exponential linear unit (SELU) was used instead of rectified linear unit (ReLU) to induce self normalizing properties and prevent the ReLU units be killed off by dying ReLU problem [38]. Moreover, the LeCun normal initializer was used to initialize parameters with a scaled Gaussian distribution, thus improve performance as well as training speed [39].

A simple step action recognition experiment was conducted to demonstrate how the model predicts the action of a step, as shown in Figure 5. The barometer data were collected as the user ascended the stairs from the first floor to the second floor, and then took the elevator back to the first floor. From Figure 5, we noticed that the model detected not only the flat floor and stairs, but also the height change caused by the elevator. In addition, the model is sensitive to sudden changes in barometric pressure, and it was immediately able to detect the first step after floor changes when the user took the elevator. Due to the cost of dataset creation and the model size in the mobile platform, we did not design a complex MLP model with a larger dataset to classify the action more finely, such as by making a distinction between elevators and stairs. Instead, these functions were realized in the floor decision algorithm based on the characteristics of model prediction in stairs and elevators we observed in Figure 5.

4) Floor Decision Algorithm

As shown in Algorithm 1, the floor decision algorithm in this study was developed for the following functions: (a) convert the step action to the floor number, (b) handle any incorrect predictions arising from outlier data, and (c) identify the floor...
Algorithm 1 Floor decision algorithm

**Input:** barometer reading \( B \) and recognized action \( act \) of a detected step

**Output:** floor detection of a step

**Initialization:** \( N_{\text{wait}}, B_{\text{elevator}} \leftarrow 3, 0.3 \)

1. if actions of previous and current step are same & queue is empty then
2. return floor detection result same as previous step
3. else
4. Enqueue current step data into memory
5. Counting number of consecutive occurrences \( N_{\text{con}} \)
6. if first step in this region then
7. Record the barometer reading of this step
8. end if
9. // Decide whether to change walking states
10. if \( N_{\text{con}} > N_{\text{wait}} \) then
11. if \( act = \text{Going up/down} \) then
12. Calculate difference of barometer readings \( B_{\text{delta}} \)
13. // Decide transition type
14. if \( B_{\text{delta}} > B_{\text{elevator}} \) then
15. transition type \( \leftarrow \) elevator
16. else
17. transition type \( \leftarrow \) stairs
18. end if
19. Update steps in queue and clear the queue
20. return transition type and direction
21. else
22. // Calculate floor number when back to floor
23. Update current floor base on transition direction
24. Update steps in queue and clear the queue
25. return current floor
26. end if
27. else
28. Wait for prediction of next step
29. end if
30. end if

Otherwise, the algorithm waits for \( N_{\text{wait}} \) steps and saves the step data in queue to confirm the possible effect of outliers, where the state of the steps in queue is determined following the decision result. A small magnitude of waiting threshold \( N_{\text{wait}} \) implies that the floor detection result can be obtained immediately but is easily affected by incorrect prediction, whereas a large \( N_{\text{wait}} \) means that floor detection is robust to the outlier data but requires a \( N_{\text{wait}} \) step waiting period when the user changes floors. Regarding the transition type, because the model is sensitive to sudden change of the pressure values, the algorithm records the barometer reading of the current step if the predicted action is different from the previous step, and calculates the difference when the state change is confirmed. The transition type can be obtained easily from the difference, since the pressure change caused by the elevator is much larger than that caused by stairs. Therefore, the possible outputs of floor decision algorithm are a floor number, “Stairs up,” “Stairs down,” “Elevator up,” and “Elevator down.” The floor number indicates the current floor where the user is located, and the rest are used to represent floor transition information.

### B. PDR METHOD USING SMARTPHONE SENSORS

In this subsection, we introduce the smartphone IMU-based PDR used in this study. The proposed scheme assumes that the smartphone always points forward, which means the device heading and walking direction are identical; thus, they are both referred to heading direction in throughout the paper. The user mobility information can be obtained using the SensorEvent and SensorManager classes in Android OS [40], [41], or CoreMotion and CLHeading classes in iOS [42], [43]. A typical IMU-based PDR method composes of step detection, stride length estimation, and heading direction estimation, as well as provides the polar vectors such as \{step length, heading direction\} that can be summed to estimate the location of each step. Next, we describe how the three basic functions of the PDR method work in practical localization.

1) Step Detection

Step detection is typically implemented by detecting the peaks and valleys of vertical acceleration generated by feet periodically touching the ground during walking [46]. To obtain the vertical acceleration, the raw acceleration values measured in the local coordinate system (LCS) need to be transferred to the global coordinate system (GCS) through the rotation transformation. The rotation matrix \( R \) can be obtained by the cross-product of a gravity vector from the accelerometer and a magnetic vector from the magnetometer. We used getRotationMatrix function provided by SensorManager to perform this calculation. Subsequently, the acceleration in GCS \( A^g \) can be computed from the acceleration in LCS \( A \) as

\[
A^g = R \cdot A. \tag{10}
\]
Assuming that $a_z$ is the vertical acceleration in $A^0$, step detection can be expressed as
\[
\{a_k^z > a_{upper,k}, a_{upper,k} + p < a_{lower,k}, p_{min} < p < p_{max}\}. \tag{11}
\]
Here, $a_{upper}$ and $a_{lower}$ are the positive and negative threshold, respectively, and we set $a_{upper} = 0.9m/s^2$ and $a_{lower} = -0.8m/s^2$. $p_{min}$ and $p_{max}$ are the minimum and the maximum duration of one step, respectively, and we set $p_{min} = 3$ and $p_{max} = 12$, which correspond to 0.15s and 0.6s under 20Hz of sampling frequency. Once a step is detected, PDR calculates the corresponding stride length and heading direction.

2) Stride length estimation
A commonly used approach that describes the relationship between stride length and vertical acceleration is given in [44].
\[
\lambda_k = \tau_k \cdot \sqrt{a_{max,k}^z - a_{min,k}^z}, \tag{12}
\]
where $\lambda_k$ is the stride length of the $k$-th step, $a_{max,k}^z$ and $a_{min,k}^z$ are the maximum and the minimum vertical acceleration during the $k$-th step, and $\tau_k$ is the coefficient for adjusting the magnitude of the $\lambda_k$.

When a user takes stairs, the variance in vertical acceleration is generally greater than that during normal walking due to gravity. Therefore, by calculating (12), the estimated step length when taking the stairs is usually slightly longer than that during normal walking, whereas the real step length is typically similar to or even shorter than that during normal walking owing to the length limitation by the stairs. To obtain an accurate step length, we need to compensate for the stride length estimation by adjusting the parameter $\tau$ in (12). According to the floor transition detection method presented in Section II-A, we know the region (e.g., a certain floor, stairs) where a detected step is located. Therefore, $\tau_1$ and $\tau_2$ are given to compensate for the step length calculation in different regions. In this study, we set step length coefficient $\tau_k$ of the $k$-th step as
\[
\tau_k = \begin{cases} 
\tau_1, & \text{if stairs} \\
\tau_2, & \text{otherwise} 
\end{cases}, \tag{13}
\]
where $\tau_1 = 0.7\tau_2$, and $\tau_1$ and $\tau_2$ can be determined empirically or from the building information. By distinguishing between stairs and normal walking, the stride length estimation in multi-floor tracking is improved.

The accelerometer reading is mainly affected by the force generated by user movement rather than environmental factors; thus the performance of step detection and stride length estimation in an indoor environment is almost identical to that in an outdoor environment.

3) Heading Direction Estimation
The traditional way to estimate the orientation of a device is to compute the pitch and roll angle from the accelerometer reading and the azimuth angle from the magnetometer reading, and then combining them to obtain a vector that represents 3-axis orientation [34], [45]. Another way to update the orientation is by integrating the angular velocity measured by the gyroscope sensor to obtain the 3-axis rotation angle during a sampling time interval. We denote $\alpha_k^{zm}$ as the orientation calculated from accelerometer and magnetometer sensors, and $\omega_k$ as the gyroscope measurement at the $k$-th step. $\alpha_k^{zm}$ provides the absolute heading angle relative to the GCS, which is prone to errors from magnetic interference, whereas $\omega_k$ is characterized by short-term accuracy but suffers from the drift problem. In this study, we estimated the orientation $\alpha_k^f$ by fusing the $\alpha_k^{zm}$ and $\omega_k$ as follows to compensate for their characteristic disadvantages.
\[
\alpha_k^f = \beta \cdot (\alpha_{k-1}^f + \omega_k \cdot \Delta t) + (1 - \beta) \cdot \alpha_k^{zm}. \tag{14}
\]
Here, $\Delta t$ is the sampling time interval and $\beta$ is the parameter that determines fusion proportion. We set $\beta = 0.98$, which means that the heading update was dominated by gyroscope readings, and the accelerometer/magnetometer was used to compensate for the gyroscope drift over time.

Note that although there are many approaches to implement the components of PDR, the optimal solution is still an open problem. We did not focus on providing the best PDR strategy, and the appropriate implementation of a PDR component can be chosen depending on practical use. After the step length calculation and heading direction estimation, the location at the $k$-th step $P_k(x_k, y_k)$ is updated as
\[
P_k(x_k, y_k) = [x_k \ y_k]^{T} = P_{k-1}(x_{k-1}, y_{k-1}) + \Delta P_k = [x_{k-1} \ y_{k-1}]^{T} + \lambda_k [\sin(\alpha_k^{fz}) \cos(\alpha_k^{fz})]^{T}. \tag{15}
\]

C. LOCATION CORRECTION AND TRACKING PROCEDURE
In this subsection, we introduce location correction to prevent the accumulation of estimated error, and the complete indoor multi-floor tracking procedure. From the DL-based floor transition detection method introduced in Section II-A, we know whether a detected step serves as a transition. The proposed scheme uses these transition signals to correct the estimated location, and then combines the altitude information with PDR tracking to generate the multi-floor localization result.

1) Calibration Node Matching-Based Location Correction
We observed that the design of the common facilities distribution in the building always considers the balance between convenience and cost due to limited resources. Therefore, common facilities (e.g., toilets, stairs) that perform the same function on each floor are distributed as evenly as possible over a wide area. When a pedestrian changes floors, we know that they are near a specific regions, such as stairs or an
From this observation, we create CNs near stairs and elevators. Once a floor transition signal is detected, the proposed scheme retrieves the possible CNs and calibrates the estimated location through the closest CN’s information.

When a new step is detected, the proposed scheme not only calculates the step length and direction, but also estimates altitude information, represents the rough region of the user’s location. By comparing the region of the current step to that of the previous step, we can confirm whether the user has transitioned between regions. The following information can be obtained when a signal of region change is detected: (i) the region where the previous step is located, (ii) the direction of stairs/elevator usage, and (iii) the transition type. This information limits the possible locations of the user to a few places. Figure 6 shows an example of the user changing floors. Assuming that this floor has only one staircase and elevator, we know that the user was previously on the second floor according to (i), is heading downstairs according to (ii), and is taking the stairs rather than the elevator according to (iii). With the presence of a nearby CN, the estimated location can be directly calibrated based on the CN’s information.

Before using the information, the CN profile needs to be defined for matching. We generated CN profiles according to facility usage and location, such as

\[ CN_{set} = \{ P_{cn}^j, Feature_j, j = 1, \ldots, n_{cn} \} \]  

where \( P_{cn}^j = (x_{cn}^j, y_{cn}^j) \) is the location of the CN, \( Feature_j = \langle \text{previous region, current region} \rangle \) is the CN characteristic, and \( n_{cn} \) presents the total number of CNs. Here, the previous region in \( Feature_j \) corresponds to the (i), and the current region corresponds to (ii) and (iii), as the height information obtained from floor detection denotes both the transition direction and type. Thus, there is at least one CN for matching and correction when the user changes regions. Next, the location correction is performed according to the following rules:

- Once a signal of region change is detected, we only retrieve the possible CNs at the first step of the new region, and then generate a list that includes the possible CNs.
- We compute the Euclidean distances between the current location and the CNs in the list by (17). If the length of the list is 1 (i.e., only one possible CN), then match this CN directly; otherwise, match the closest CN.

\[
d(P, P_{cn}) = \sqrt{(x - x_{cn})^2 + (y - y_{cn})^2}. \tag{17}
\]

- We denote the distance of the closest CN from the current location as \( d_c \). If \( d_c \geq 1.5 \text{m} \), the proposed scheme sets the location of the closest CN as the calibrated location. Otherwise, the current location would not be corrected.

2) Indoor Multi-Floor Tracking Procedures

In this subsection, we describe how the proposed scheme performs multi-floor tracking in an indoor environment. Our scheme consists of a smartphone and a CN profile database, which is applicable in areas that cannot be covered by infrastructure-dependent approaches. As the user carrying a smartphone walks inside in a building, the scheme employs the various sensors in the smartphone to measure mobility information over time and update the location of each step, as well as corrects the location by matching it to a CN.

**Application initialization**: Regarding the floor transition detection presented in Section II-A, there are two problems that need to be considered first in the online phase, as follows.

(a) The DL-based step action recognition is performed by receiving \( s \) (e.g., \( s = 15 \)) barometer readings as input features, and predicts the action of the current step, which implies that the DL model has to wait until \( s \) step data are collected. This is a severe delay, since the user may have already changed floors once during those \( s \) steps.

(b) In the data preprocessing stage, we execute the mean centering operation to the training data by subtracting the mean of the barometer data. This is impractical in the online phase since the mean of the measured pressure data is unknown in advance. We need an alternative to shifting the barometer readings close to 0.

In order to overcome the problem described in (a), our scheme generates \( s - 1 \) synthetic barometer data based on the reading of the first detected step to simulate barometer data before the first step. Since the identical pressure data causes sensitivity to pressure change, random variables are generated based on the normal distribution. Assuming that the barometer reading of the first detected step is \( B_1 \), we generate \( s - 1 \) barometer data \( B_{gen} \sim \mathcal{N}(B_1, 0.15) \) as the simulated pressure data. Subsequently, we concatenate \( B_{gen} \) and \( B_1 \) to obtain the \( \{ B_{gen}, B_{gen}, \ldots, B_{gen}, B_1 \} \) as the first input data of DL model. The simulated pressure data could be gradually replaced by the measured data as the tracking progresses until all input features are measured pressure values. Regarding the problem described in (b), because the mean centering is unusable in the online phase, we center the barometer readings by subtracting the first barometric value. The pressure value used for centering can then be updated after a certain tracking path length to avoid the effect of atmospheric pressure drift.
Multi-floor tracking: As illustrated in Figure 7, the proposed scheme executes as follows.

1. The sensors in the smartphone continuously measure and analyze mobility data during user movement. When a step is detected, the DL model predicts the step action and delivers its height attribute through the floor decision algorithm. The height information of a detected step is represented as a rough region, such as a certain floor, “Stairs up,” etc.

2. After the height information has been determined, we detect the region change by comparing adjacent steps. If the region has changed, the scheme retrieves the possible CNs from the database and corrects the user’s location by matching it to a CN according to the location correction rules described in Section II-C1. In particular, if the floor transition type is stairs, the scheme performs twice the location calibration to eliminate the localization error generated in the flat floor and stairs. If the floor transition type is elevator, the scheme only performs once the location calibration, since the elevator transports the user vertically without 2D location changes.

3. If the region has not changed, the scheme estimates the stride length $\lambda$ and heading direction $\alpha$ of this step according to the smartphone-based PDR method. The step length calculation uses the two gain coefficients $\tau_1$ and $\tau_2$, corresponding to stairs and other cases respectively, to improve the accuracy of stride length estimation. Afterwards, PDR updates the current location according to (15).

4. Finally, the scheme combines the height information obtained from DL-based floor detection with the 2D location obtained from PDR, to generate the multi-floor tracking result.

III. EXPERIMENT RESULTS

In this section, we present several experimental results to validate the proposed scheme. The smartphone used in tracking was a Samsung Note 10+ with Android 11, 256 GB storage, a barometer and triaxial IMU sensors. We built a data collection application on the smartphone to measure and save the sensor data. Once a step was detected, the application stored the sensor data for location estimation based on Python 3. In addition, development, training, and prediction of the DL model were conducted using the machine learning library TensorFlow. We assumed that the user carries the smartphone in front of their body during tracking, and walks in the middle of the corridor if there are no other pedestrians or obstacles along the path.

A. DL-BASED FLOOR TRANSITION DETECTION

The accuracy of floor transition detection determines the scheme’s performance, as it not only provides the altitude information but also impacts the PDR estimation and location correction. To demonstrate the effectiveness of the proposed method, we assessed the accuracy of the DL-based step action recognition and floor decision algorithm. In this study, the step action is represented as “Normal walking,” “Going up,” or “Going down.” The floor decision algorithm receives the step action and outputs the altitude information for each step, which is represented as one of five classes: a certain floor, “Stairs up,” “Stairs down,” “Elevator up,” “Elevator down.”

The action and region of each step were recorded during tracking and used to evaluate the floor transition detection. In particular, if the floor transition type is stairs, there are approximately 20 steps to be taken during floor transition; therefore, we can easily assess the floor transition accuracy for steps in stairs by comparing the predicted labels for each step to the ground-truth labels. However, as the floor change through the elevator is sudden, there is no step to be taken during floor transition; therefore, alternative evaluation criteria for steps regarding the elevator are required. We determined that the prediction regarding “Elevator up/down” is correct when there are more than $N_{wait}$ steps continuously classified as “Elevator up/down” after the user takes the elevator.

The accuracy rate (AR) of floor detection was employed to evaluate the accuracy by calculating whether the estimated floor is consistent with the true floor for a step, as follows.

$$AR^{FD} = \frac{\#\{\hat{f}_i \mid \hat{f}_i = f_i\}}{N_{normal}} \times 100\%,$$ (18)

where $AR^{FD}$ is the AR for floor detection, and $f_i$ and $\hat{f}_i$ are the ground-truth number and detected floor number, respectively. $N_{normal}$ indicates the total number of steps whose ground-truth label is “Normal walking” since the steps in the transition zone (i.e., stairs or elevator) are not calculated for estimation.

We conducted several experiments in Cho Man-sik Memorial Hall, Han Kyung-chik Memorial Hall, Jilli Hall, and...
Sung-deok Hall at Soongsil University. With three experiments per building, we used 13 elevators and 16 sets of stairs. The floor height of each building is approximately 3.0–3.5m. Figure 8(a) shows one result for step action recognition in Cho Man-sik Memorial Hall, where the barometer readings have been denoised by (1). Figure 8(b) shows the results for floor detection after utilizing the floor decision algorithm. Due to significant noise during tracking, the barometer readings exhibited fluctuations even during denoising. In Figure 8(a), it can be seen that there are several incorrectly identified actions when the region changed, while in Figure 8(b) they are ignored as errors by the floor detection algorithm, and do not affect floor detection accuracy. Table 2 and Table 3 show the AR values of floor detection in each building, and the confusion matrix of the results for the floor decision algorithm, respectively. Based on Table 3, we observe that the detection accuracies of elevator cases are 100%, which implies that all elevator usage was detected. For “Stairs up” and “Stairs down,” the detection accuracies are close to 90% because data augmentation eliminated the imbalanced data problem. Most false-negative errors were the “Stairs up/down” steps detected as floor steps at the start of stair transitions as shown in Figure 8(b). This occurred because the DL-model requires 2–5 barometric reading changes to confirm a floor transition. Although these errors did not affect the accuracy of floor detection, which is corroborated by the over 99% total accuracy shown in Table 2, they caused a delay of several steps when the user entered and exited the stairs, which could lead to inaccurate location corrections.

Because we know the direction of stairs usage, the CNs can be set a little distance in front of the joint area of the stairs and the floor to compensate for the delay.

### B. INDOOR MULTI-FLOOR LOCALIZATION

To verify the performance of our indoor multi-floor tracking approach, we conducted localization experiments on two floors (third and fourth floors) of Cho Man-sik Memorial Hall at Soongsil University. Each floor consists of two segments with dimensions of 18mx42m and 75mx18m, and includes a corridor, rooms, open study regions, and other subdivisions. There are two open staircases and one separate stairwell on each floor: one staircase has 22 stairs, and the other is not open for use, while the stairwell has 20 stairs. In addition, each floor has three elevators, two of which are next to each other: one elevator from the second underground floor to the first floor, and another from the first floor to the seventh floor. We set one CN at each available elevator, and two CNs at each joint area of stairs and floor base in the walking direction to avoid confusion in CN matching and compensate for the transition delay. In terms of the performance metrics, the AR of localization was computed to represent the probability of estimated steps within the error boundary $\epsilon$, as follows [47]:

$$AR_{loc} = \frac{\#\{P_k(x_k, y_k) | P_k \subset P(\epsilon)\}}{N_{step}} \times 100\%,$$

where $P(\epsilon)$ indicates the desired path with thickness $\epsilon$, and $N_{step}$ denotes the total number of detected steps.

Before demonstrating the experimental results, we made several descriptions for the visualization of the tracking trajectory. Because the altitude information of the pedestrian is represented as a floor level, we plotted the 2D location of each detected step on the floor plan that corresponds to the floor number where the user is located to show more details.

#### TABLE 2. AR for floor detection.

| Building   | Cho Man-sik | Han Kyung-chik | Jilli | Sung-deok | Total |
|------------|-------------|----------------|-------|-----------|-------|
| $AR^{FD}$  | 99.35%      | 99.87%         | 99.33%| 99.83%    | 99.62%|

#### TABLE 3. Confusion matrix for floor transition detection.

| Recognized action | Stairs down | Elevator down | Floor | Stairs up | Elevator up |
|-------------------|-------------|---------------|-------|-----------|-------------|
| Stairs down       | 0%          | 0%            | 100%  | 0%        | 0%          |
| Elevator down     | 0%          | 90%           | 10%   | 0%        | 0%          |
| Floor             | 99.6%       | 0%            | 10%   | 0%        | 0%          |
| Stairs up         | 0%          | 0%            | 0%    | 0%        | 0%          |
| Elevator up       | 0%          | 0%            | 0%    | 0%        | 100%        |
of the user’s activities on each floor. When the user detected to walk through the transition zone, the floor number was updated at the first “Elevator up/down” step or the last “Stairs up/down” step (i.e., exiting the elevator or stairs).

We conducted a tracking experiment using sensor data collected for 483 steps during walking. To illustrate the performance of the CN matching-based location correction algorithm, the experimental results are given in Figures 9 and 10, which demonstrate two demo localization trajectories: indoor multi-floor tracking without location correction, and indoor multi-floor tracking with CN matching-based location correction. The user holding the smartphone walked along the desired path plotted in green. Each grid in the map represents 2m of the distance. Each empty circle in the figure is a detected step where the red circle indicates a step recognized as “Normal walking,” a purple circle indicates “Stairs up/down,” and a blue circle indicates “Elevator up/down.” We set the location of the CNs 1–3m in front of the transition area of the stairs and the floor to avoid the delay problem generated when entering and exiting the stairs. Only the CNs that we used were plotted in the figures. The start and end points are indicated by a square and a diamond. The activities during walking are described below, which include all transition cases: go upstairs/downstairs by elevator and stairs.

- 3rd floor activities (F3): Start from the 3rd floor » walk along F3 corridor until the user enter the separate stairwells » go upstairs to 4th floor. The estimated location could be corrected via S3 and S6 during the floor transition.
- 4th floor activities (F4): Exit the stairwells » move along F4 corridor » enter room 404 and exit room » move along F4 corridor and enter the elevator to go downstairs to the 3rd floor. The estimated location could
be corrected via E2 when taking the elevator.

- 3rd floor activities: Exit the elevator » go through the open study region and the F3 corridor to the elevator at the lower left of the floor plan » go upstairs to the next floor. The estimated location could be corrected via E4.
- 4th floor activities: Exit the elevator » move along F4 corridor to the open staircase » go downstairs through the staircase to the 3rd floor. The estimated location could be corrected via S5 and S1 during the floor transition.
- 3rd floor activities: Back on F3 corridor and move to end point.

These floor transition detection results are presented in Figure 8(b). Base on the floor transition detection results, all transition types and the directions were successfully detected, and the fluctuation in barometer readings caused by several complex regions in each floor did not affect the floor detection accuracy. These floor detection results provide stable altitude information for the estimated 2D location. From Figures 9 and 10, the stride length estimation of PDR improved according to the building’s floor plan. When the user changes floors, the scheme matches the possible CN base on the location and mobility of the user, and corrects the estimated location using the CN information to eliminate the accumulated error during tracking. To evaluate the scheme’s performance, we conducted 12 experiments in four buildings and compared the accuracy of multi-floor tracking with and without location correction. The experimental results show that DL-based floor detection delivers stable and accurate tracking results in varying areas, and the localization performance in long path tracking is significantly improved by location correction.

The CN presented in the scheme not only can be used in infrastructure-independent approaches, but also is useful for the infrastructure-dependent approaches, to reduce the requirements of the anchor nodes. In future, we will optimize the DL model size for mobile platform friendly, introduce additional feature data to improve the prediction of the DL

IV. CONCLUSION AND DISCUSSION

In this study, we presented an indoor multi-floor tracking scheme that functions without infrastructure. The proposed scheme consists of three components: DL-based floor transition detection, IMU-based PDR, and CN matching-based location correction. Considering the limitations of the mobile platform, we designed a lightweight MLP model and train it using time-series pressure data collected from the barometer sensor. In addition, we presented a data augmentation method to solve the overfitting and imbalanced performance issues caused by insufficient and imbalanced train data, as well as to reduce the cost of dataset preparation. A floor decision algorithm was developed to obtain a robust prediction of each step in complex areas, and to identify the floor transition direction and type. The 2D location update was implemented by the PDR method, and the stride length estimation of PDR was improved by the floor detection information. The proposed scheme realizes multi-floor tracking by combining floor detection and PDR. To avoid the error accumulation of PDR estimation, we implemented CNs near transition zones according to the building’s floor plan. When the user changes floors, the scheme matches the possible CN base on the location and mobility of the user, and corrects the estimated location using the CN information to eliminate the accumulated error during tracking. To evaluate the scheme’s performance, we conducted 12 experiments in four buildings and compared the accuracy of multi-floor tracking with and without location correction. The experimental results show that DL-based floor detection delivers stable and accurate tracking results in varying areas, and the localization performance in long path tracking is significantly improved by location correction.

The CN presented in the scheme not only can be used in infrastructure-independent approaches, but also is useful for the infrastructure-dependent approaches, to reduce the requirements of the anchor nodes. In future, we will optimize the DL model size for mobile platform friendly, introduce additional feature data to improve the prediction of the DL
model such as acceleration and gyroscope data, and further optimize 2D location estimation by combining it with floor detection.

REFERENCES

[1] H. Zhao, W. Cheng, N. Yang, S. Qiu, Z. Wang, and J. Wang, “Smartphone-based 3D indoor pedestrian positioning through multi-modal data fusion,” Sensors, vol. 19, no. 20, pp. 5454-5473, Oct. 2019.

[2] T. D. V. T. L. N. Nguyen, and Y. Shin, “A smartphone indoor localization using inertial sensors and single Wi-Fi access point,” in Proc. Int. Conf. Indoor Pos. & Indoor Nav., Pisa, Italy, Sept. 2019, pp. 1-7.

[3] H. Zou, Z. Chen, H. Jiang, L. Xie, and C. Spanos, “Accurate indoor localization and tracking using mobile phone inertial sensors, Wi-Fi and iBeacon,” in Proc. IEEE Int. Symp. Inertial Sens. & Syst., Kaui, USA, Mar. 2017, pp. 1-4.

[4] A. Alarifi, A. Al-Salman, M. Alsaleh, A. Alnaefesah, S. Al-Hadhrami, M. A. Al-Ammar, and H. S. Al-Khalifa, “Ultra wideband indoor positioning technologies: analysis and recent advances,” Sensors, vol. 16, no. 5, pp. 707-743, May 2016.

[5] A. Poulose and D. S. Han, “UWB indoor localization using deep learning LSTM networks,” Appl. Sci., vol. 10, no. 18, pp. 6290-6312, Sept. 2020.

[6] T. Brovko, A. Chugunov, and A. Malyshev, “Positioning algorithm for smartphone based staff tracking,” in Proc. Int. Russian Automation Conf., Sochi, Russian Federation, Sept. 2021, pp. 464-468.

[7] G. Cheng, “Accurate TOA-based UWB localization system in coal mine based on WSN,” Phys. Proceedia, vol. 24, pp. 534-540, Jan. 2012.

[8] T. D. V. T. L. N. Nguyen, and Y. Shin, “Pedestrian indoor localization and tracking using hybrid Wi-Fi/PDR for iPhones,” in Proc. IEEE Veh. Technol. Conf., Helsinki, Finland, Apr. 2021, pp. 1-7.

[9] J.B. Xu, H.Q. Zhang, and J.L. Zhang, “Self-adapting multi-fingerprints joint indoor positioning algorithm in WLAN based on database of AP ID,” in Proc. Chinese Control Conf., Nanjing, China, July 2014, pp. 534-538.

[10] P. Spachos and K. N. Plataniotis, “BLE beacons for indoor positioning at an interactive IoT-based smart museum,” IEEE Syst. J., vol. 14, no. 3, pp. 3483-3497, Sept. 2020.

[11] Z. Chen, Q. Zhu, and Y. C. Soh, “Smartphone inertial sensor-based indoor localization and tracking with iBeacon corrections,” IEEE Trans. Industrial Inform., vol. 12, no. 4, pp. 1540-1549, Aug. 2016.

[12] P. Baronti, P. Barsocchi, S. Chessa, F. Mavilia, and F. Palumbo, “Indoor Bluetooth low energy dataset for localization, tracking, occupancy, and social interaction,” Sensors, vol. 18, no. 12, pp. 4462-4479, 2018.

[13] S. Romanuk and Z. Gosiewski, “Kalman filter realization for orientation and position estimation on dedicated processor,” Acta Mechanica et Automatica, vol. 8, no. 2, pp. 89-94, 2014.

[14] A. R. Jiménez, F. Seco, J. C. Prieto, and J. Guevara, “Indoor pedestrian navigation using an INS/EKF framework for yaw drift reduction and a foot-mounted IMU,” in Proc. Workshop Pos. Navig. Commun., Dresden, Germany, Mar. 2010, pp. 135-143.

[15] M. El-Diasty, “An accurate heading solution using MEMS-based gyroscope and magnetometer integrated system (Preliminary results),” ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci., vol. 2, pp. 75-78, Oct. 2014.

[16] R. Mahony, T. Hamel, and J.-M. Pflimlin, “Nonlinear complementary filters on the special orthogonal group,” IEEE Trans. Autom. Control, vol. 53, no. 5, pp. 1203-1218, June 2008.

[17] A. Wang, X. Ou, and B. Wang, “Improved step detection and step length estimation based on pedestrian dead reckoning,” in Proc. IEEE Int. Symp. Electromagn. Comput., Nanjing, China, Nov. 2019, pp. 1-4.

[18] H. Ye, T. Gu, X. Zhu, J. Xu, X. Tao, and N. Jin, “FTrack: infrastructure-free floor localization via mobile phone sensing,” in Proc. IEEE Int. Conf. Pervasive Comp. Commun., Lugano, Switzerland, Mar. 2012, pp. 2-10.

[19] K. V. Ramana, N. Jianwei, M. A. A. Aziz, and M. Y. Umar, “A robust multi-cue blending-based approach for floor detection,” in Proc. Int. Blurban Conf. Appli. Sci. & Technol., Islamabad, Pakistan, Jan. 2016, pp. 64-69.

[20] T. L. N. Nguyen, T. D. Vy, K.-S. Kim, C. Lin, and Y. Shin, “Smartphone-based indoor tracking in multiple-floor scenarios,” IEEE Access, vol. 9, pp. 141048-141063, Oct. 2021.

[21] B. Siğmşek and O. N. Güneş, “Indoor floor change detection using barometer,” in Proc. Signal Proc. & Commun. Appli. Conf., Sivas, Turkey, Apr. 2019, pp. 1-4.

[22] Y. Le Cun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436-444, May 2015.

[23] K. Y. Seok and J. H. Lee, “Deep learning based fingerprinting scheme for wireless positioning,” in Proc. Int. Conf. Artifical Inell. In Inform. & Commun., Fukuoka, Japan, Feb. 2020, pp. 312-314.

[24] F. Mahdavi, H. Zavary, and R. Rajabi, “RSS localization using an optimized fusion of two deep neural networks,” IEEE Sensors Lett., vol. 5, no. 12, pp. 1-4, Dec. 2021.

[25] I. Klein and O. Asraf, “StepNet—Deep learning approaches for step length estimation,” IEEE Access, vol. 8, pp. 85706-85713, May 2020.

[26] J. Hannink, T. Kautz, C. Pasluosta, J. Barth, S. Schülein, K.-G. Gabmann, J. Kluken, and B. M. Eskofier, “Mobile stride length estimation with deep convolutional neural networks,” IEEE J. Biomed. & Health Inform., vol. 22, no. 2, pp. 354-362, Mar. 2018.

[27] Q. Wang, H. Luo, L. Ye, A. Men, F. Zhao, Y. Huang, and C. Ou, “Pedestrian heading estimation based on spatial transformer networks and hierarchical LSTM,” IEEE Access, vol. 7, pp. 162309-162322, Oct. 2019.

[28] G. Milette and A. Stroud, Eds., Professional Android Sensor Programming, Hoboken, NJ, USA: Wiley, 2012, ch. 6.

[29] T. Willemsen, F. Keller, and H. Sternberg, “Concept for building a MEMS based indoor localization system,” in Proc. Int. Conf. Indoor Pos. & Indoor NAV., Busan, Korea, Oct. 2014, pp. 1-10.

[30] A. Sherstinsky, “Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network,” Mar. 2020, arXiv:1808.03314.

[31] M. C. Popescu, V. E. Balas, L. Peresu-Popescu and N. Mastarakis, “Multilayer perceptron and neural networks,” WSEAS Trans. Circ. & Syst., vol. 7, no. 3, pp. 579-588, July 2009.

[32] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, “Self-normalizing neural networks,” June 2017, arXiv:1706.02515.

[33] B. Hanin and D. Rolnick, “How to start training: the effect of initialization and architecture,” Mar. 2018, arXiv:1803.01719.

[34] Android Developer. SensorEvent. Accessed: Sept. 12, 2021. [Online]. Available: https://developer.android.com/reference/android/hardware/SensorEvent

[35] Apple Developer. Core Motion. Accessed: Nov. 12, 2020. [Online]. Available: https://developer.apple.com/documentation/coremotion

[36] Apple Developer. Core Location. Accessed: Nov. 12, 2020. [Online]. Available: https://developer.apple.com/documentation/corelocation

[37] H. Weinberg, “Using the ADXL202 in pedometer and personal navigation applications,” Analog Devices, Tech. Rep. AN-602, 2002. [Online]. Available: https://developer.apple.com/documentation/coremotion/applereference/application_notes/p513772624AN0602.pdf

[38] J. W. Lee, Android Sensor Story. Los Angeles, CA, USA: Freetee, 2014, chs. 4-6.

[39] 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. Syst., Seattle, USA, Mar. 2011, pp. 222-230. [Online]. Available: http://www.analog.com/static/imported-files/application_notes/p513772624AN0602.pdf

This article has been accepted for publication in IEEE Access. This is the author’s version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3183072
CHENXIANG LIN received the B.S. degree in computer science and engineering from Korea University, Korea, in 2021. Currently, he is pursuing his M.S. degree in information and telecommunication engineering at Soongsil University, Korea. His research interests include indoor localization and deep learning.

YOAN SHIN (Senior Member, IEEE) received the B.S. and the M.S. degrees in electronics engineering from Seoul National University, Korea, in 1987 and 1989, respectively, and the Ph.D. degree in electrical and computer engineering from The University of Texas in Austin, USA, in 1992. From 1992 to 1994, he was a member of the technical staff with Microelectronics and Computer Technology Corporation (MCC), Austin, USA. Since 1994, he has been with School of Electronic Engineering, Soongsil University, Korea, where he is currently a professor. From 2009 to 2010, he was a visiting professor with Department of Electrical and Computer Engineering, The University of British Columbia, Canada. He was the Dean of School of Electronic Engineering, the Vice Dean of College of IT, the Dean of Office of Research, and is currently the Dean of Office of Planning of Soongsil University. He has served as an Organizing/Technical Committee Member for various prominent international conferences, including IEEE VTC 2003-Spring, ISITA 2006, IEEE ISPLC 2008, APCC 2008, IEEE ISIT 2009, APWCS 2009, APWCS 2010, APCC 2010, APWCS 2011, ISAP 2011, APWCS 2012, APCC 2012, and IEEE WCNC 2020. In particular, he was the Technical Program Committee Chair of APWCS 2013, the General Co-Chair of APWCS 2014 and APWCS 2015, and the General Vice Chair of IEEE ICC 2022. He is the President of year 2022 of the Korean Institute of Communications and Information Sciences (KICS). His research interests include mobile communications and intelligent signal processing.