IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 13 June 2023, accepted 24 June 2023, date of publication 28 June 2023, date of current version 6 July 2023.

*Digital Object Identifier 10.1109/ACCESS.2023.3290167*

APPLIED RESEARCH

# Multi-Floor Indoor Localization Scheme Using a Seq2Seq-Based Floor Detection and Particle Filter With Clustering

**CHENXIANG LIN AND YOAN SHIN, (Senior Member, IEEE)**
School of Electronic Engineering, Soongsil University, Seoul 06978, South Korea

Corresponding author: Yoan Shin (yashin@ssu.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korea Government (MSIT) under Grant RS-2023-00251595.

**ABSTRACT** In this study, we present an infrastructure-independent multi-floor indoor localization scheme that uses a deep learning (DL)-based floor detection method and a particle filter with clustering. To implement localization with limited measurement data, we incorporate the user's vertical motion information to initialize and optimize the system. This method assumes two prerequisites: the capability for rapid floor detection and extraction of vertical motion features. These ensure a correlation between vertical movement and two-dimensional location and can be efficiently integrated with map information. The proposed scheme has several notable features. First, we utilize the strong feature extraction capability of the sequence-to-sequence (Seq2Seq) model for sequential data to implement real-time step action prediction. We also develop a floor decision algorithm to extract vertical movement information from the step action sequence. The proposed floor detection method can track the floor regardless of the user's activities. Second, we configure calibration nodes (CN) on the map based on prior knowledge from the environmental information. By combining CNs with DL-based floor detection, we not only extend the particle filter to three-dimensional applications but also achieve calibration and repair of the localization. Third, we introduce a clustering method to improve localization accuracy and reduce computational complexity in uncertain measurements. The experimental results show that the Seq2Seq model has good robustness to noisy data, the proposed DL-based floor detection achieved an average floor number accuracy of 93.42% without restricting user behavior, and all the floor transitions were successfully recognized. Moreover, under the long path multi-floor scenario, our scheme achieved a localization accuracy of over 96% within a 2m error boundary.

**INDEX TERMS** Smartphone, indoor, deep learning, Seq2Seq, barometer, multi-floor localization, inertial measurement unit, floor transition, particle filter, clustering.

## I. INTRODUCTION

Indoor localization research is being actively pursued due to the growing demand for localization-based services (LBS). Although satellite-based schemes provide a good LBS in outdoor environments, they are inefficient in indoor spaces due to the obstruction of the building. Smartphone-based indoor localization schemes have become prevalent due to the high penetration of smartphones and the various micro-electromechanical systems (MEMS) sensors they carry. Among these, the most direct way to obtain a smartphone user's location is to use received wireless signal information, including Wi-Fi, Bluetooth, ultra-wideband (UWB), and received radio-frequency identification (RFID) [1], [2], [3], [4], [5]. However, these technologies are not always available in specific situations due to dependence on infrastructure (e.g., disasters, power outages), and manual setup and maintenance are time-consuming and labor-intensive.

As another feasible solution in indoor localization systems, pedestrian dead reckoning (PDR) methods are independent of infrastructure. It iteratively updates the user's location using a smartphone equipped with an inertial measurement

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

unit (IMU) as the input device, which includes an accelerometer, magnetometer, and gyroscope. PDR is lightweight and can work in a place without signal coverage. Unfortunately, since PDR's current estimation is derived from previous states, the estimation errors accumulate as the location changes. Thus, conventional PDR is used in conjunction with a dedicated IMU to ensure accuracy in practical applications. Fusion algorithms have been proposed to improve the performance of cheap IMU-based PDR, such as the complementary filter (CF) and Kalman filter (KF) [6], [7], [8], [9], [10]. Moreover, environmental information can be used as prior knowledge for boundary constraints and location calibration. Since the map information can fit conveniently into particle propagation, an efficient way that combines the particle filter (PF) and spatial information was successfully implemented in [11], [12], and [13]. The PF does not require assumptions about whether the problem is Gaussian or linear and can be applied in various situations. However, the PF approach suffers from two notorious problems: multimodality and sample impoverishment [14], [15], [16]. Multimodality can be caused by simple or symmetric structures of the building, which allows for multiple possibilities during propagation and is dispersed into various modes after several iterations. In particular, this problem is more likely to occur when the initial location is not provided [17]. On the other hand, particles tend to collapse at one or a few locations when the PF relies too heavily on measurement data, resulting in a loss of diversity, known as sample impoverishment. In this case, the PF can fail after the particles cross a wall and are eliminated due to inevitable noise in measurements. In addition, resampling also causes the particle distribution to become concentrated [18]. These problems are more likely encountered when measurements are insufficient.

Furthermore, the initial state is crucial for relative methods such as PDR, as it directly determines their subsequent computation. Although the initial location and height can be acquired when the user enters a building, in most cases, the localization process begins indoors where the user's exact location is typically unknown. One advantage of the PF is that it enables the calculation of user location without a given initial state [12]. Some techniques introduce received signal strength (RSS) as one of the inputs for PF localization. Although RSS contains absolute location information, which can assist the particle initialization, wireless signals are not always available. On the other hand, PDR-only PF approaches find the user's location by uniformly distributing particles on a floor plan, updating particle states through PDR estimation, and assigning weights to particles according to predefined rules until the particles gather into a place after several iterations. This approach requires no infrastructure resources but more computation because it has to generate enough particles to cover the interesting state space areas and update them. Furthermore, because IMU data does not contain absolute location information, it requires many iterations for the particles to collapse to the user's location. The user may enter another area before convergence

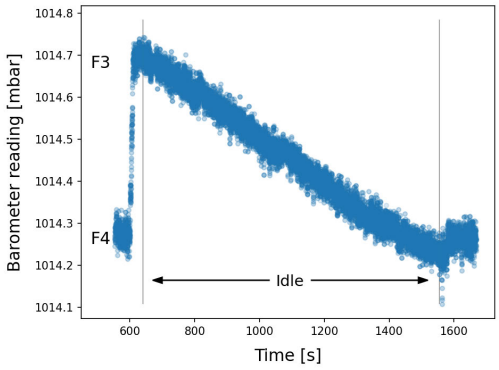



**FIGURE 1.** The barometer reading measured at the same altitude drifted by approximately 3m within 15 minutes.

(e.g., another floor), rendering previous calculations pointless and wasted. The underlying cause of these issues is that the information obtained from PDR is insufficient. Thus the introduction of more user movement information is necessary to aid in the initialization and updating of particles.

Another drawback of PDR is that it only calculates the two-dimensional (2D) location, while most buildings are multi-floor structures. The user's altitude in a multi-floor building can be represented as a floor number, and a three-dimensional (3D) location can be derived by integrating the 2D location with the floor number. There are a few PF approaches assist in determining the user's vertical movement by propagating particles to predefined vertical transition areas [12], [19]. However, particles tend to move to broader areas, while floor transition zones are generally narrow. When a user moves to another floor, only a small number of correct particles enter the transition zone. Altitude calculation based on features extracted from different IMU modes presented during the user's vertical movement was successfully implemented in [20], [21], and [22]. Reference [20] proposed two acceleration integration methods to determine height difference, and [21] formed a mapping table from distinct movement patterns for floor change estimation using travel time and step count. An inherent issue with IMU-based floor detection is that the IMU sensors are sensitive to user behavior, with unpredictable actions from the user severely impacting their measurements. Consequently, these systems typically maintain optimal performance under constant user behavior. The barometer sensor avoids this problem because its measurement is dominated by atmospheric pressure instead of user motion. Numerous floor localization methods based on barometric pressure have been developed [23], [24], [25], [26], which can broadly be classified as reference station-based floor localization methods or pressure difference measurement-based floor localization methods [27]. Since station-based methods require the deployment of infrastructure within the environment, they are not considered in this paper. On the other hand, a relationship

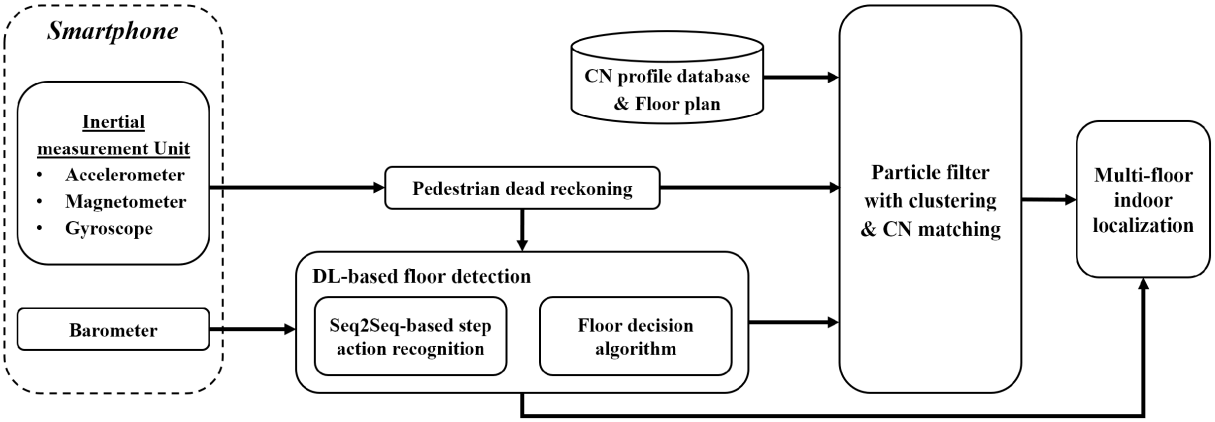



**FIGURE 2.** Architecture of our scheme.

between atmospheric pressure and height $h$ was constructed in [28].

$$h(p, p_0) = 44330 \cdot \left[1 - \left(\frac{p}{p_0}\right)^{\frac{1}{5.255}}\right], \tag{1}$$

where $p$ and $p_0$ are the current barometer reading and standard pressure at sea level in mbar. However, due to atmospheric pressure error caused by weather factors, the altitude calculated by (1) is inaccurate. Figure 1 demonstrates an example of the atmospheric pressure drift effect. Therefore, it's generally accepted to estimate the change in altitude through the pressure difference, instead of directly calculating the altitude. Another challenge of the pressure difference measurement-based method is that barometric measurement is also affected by smartphone usage and environmental factors. Although the pressure distribution of each floor, even containing various noises, has a range in an overall view, waiting for enough data to characterize the pressure distribution in a specific region will result in a significant delay, which is not conducive to cooperating with 2D locations. Moreover, height information can not only be combined with 2D location to provide 3D location but can also be used to optimize 2D localization since both are derived from the user's motion measurements. However, only a few solutions took advantage of this feature to optimize the performance [14], [29]. The reasons could be as follows.

- Delays that result from slow floor transition detection lead to a loss of correlation between height information and 2D location.
- Most floor detection solutions merely compute altitude without the capacity to extract features associated with floor transitions, while altitude alone does not suffice to correct the user's 2D location.

Therefore, the use of height information to improve 2D location requires the ability to detect fast floor transitions and extract vertical motion features. Moreover, despite the wide usage of the PF in indoor localization, handling the state of particles when the user changes floors is a tricky issue, and its optimal strategy for multi-floor scenarios remains an open problem.

In recent years, deep learning (DL) has been widely used in the analysis and processing of sensor data, creating significant advances in data feature engineering and providing many solutions in LBS [30], [31], [32]. An important aspect of the DL scheme in LBS is that, since the platform for LBS is generally a mobile device, the power consumption and computational complexity must be considered.

To address the aforementioned issues, we propose an indoor multi-floor localization scheme that is infrastructure-independent and solely relies on the onboard sensors of a smartphone. Because of the scarcity of wireless signal information, the main challenge for our scheme is determining how to extract as much motion information as possible from the limited sensor data and combine them comprehensively to provide stable, fast, and mobile-friendly indoor multi-floor localization. This research is an extended version of our previous work [33]. We propose a DL-based floor detection that exploits the sequence-to-sequence (Seq2Seq) model to predict the user's step action from time-series barometric data. A floor decision algorithm is developed to not only identify floor transitions and estimate floor numbers from the step action sequence but also extract vertical movement features of a step. Based on the proposed Seq2Seq model's stable performance for noisy data, our floor detection can work regardless of how the smartphone was worn. Next, we design and implement a PF with clustering to fuse sensor data, map information, and floor detection prediction for estimating 2D locations. We introduce mean shift and calibration nodes (CN) matching-based location correction to improve the PF's performance. The mean shift is applied as a clustering algorithm to detect PF divergence and improve location estimation. In addition, according to clustering results, the proposed scheme dynamically adjusts the number of particles to reduce the computational complexity without sacrificing performance. The CN matching-based location correction is used to combine prior knowledge from the map and the vertical user movement obtained from the proposed floor

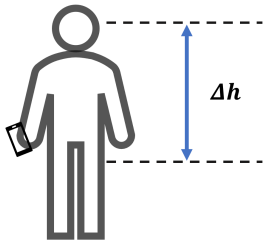



**FIGURE 3.** Height range of the smartphone relative to the body.

detection to accelerate particle convergence, correct the particle state, and provide an effective way to extend the 2D PF to 3D scenarios. The contributions of this paper are summarized as follows:

- We propose a Seq2Seq-based step action recognition and floor decision algorithm that enables fast calculation of step height and vertical movement features, without constraints on user behavior.
- We employ a PF to integrate vertical movement features from floor detection, PDR, clustering, and map information to address the sample impoverishment issue in particle filters and optimize localization performance, as well as reduce the computational complexity without sacrificing performance.
- Our approach is integrated into an infrastructure-independent positioning system, enabling reliable indoor multi-floor localization.

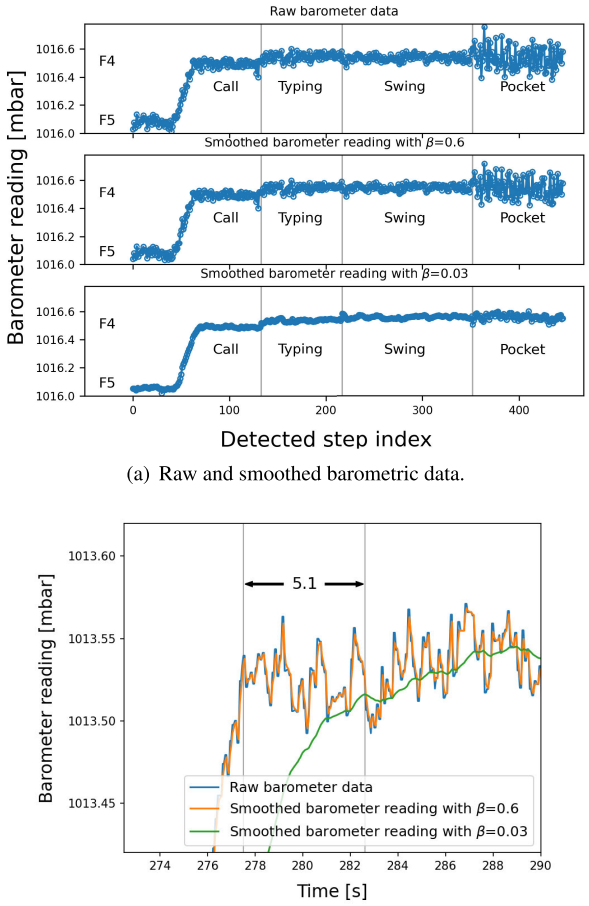
The remainder of this paper is structured as follows. Sections II and III elaborate on the methodology of our proposed indoor multi-floor localization scheme. Section IV is devoted to the evaluation and analytical examination of our scheme, followed by Section V which presents a detailed discussion. Finally, we conclude and outline potential avenues for future work in Sections VI and VII.

## II. SEQ2SEQ-BASED FLOOR DETECTION

Figure 2 illustrates an overview of the proposed scheme, which consists of two modules: DL-based floor detection and PDR-PF with clustering. As illustrated in Figure 2, the proposed scheme reads barometer and IMU sensor data while the user is walking in a building. The DL-based floor detection receives barometric data as the input to perform the floor tracking. Meanwhile, the PF incorporates the PDR estimation based on the IMU data, the prediction from floor detection, and the data from the smartphone database to calculate the 2D location. Finally, the results of floor detection and PF are combined to achieve indoor multi-floor localization. The DL-based floor detection is introduced in this section, which is responsible for floor transition detection and floor number calculation.

### A. USER ACTIVITY ANALYSIS AND FLOOR DETECTION SCENARIO

In this study, we divide multi-floor localization into two stages: the first stage where the user enters the building and moves around freely, and the second stage where the localization begins. To provide the height information required for initializing 2D localization, it is necessary during the first stage to obtain the floor number from the entrance or other technologies (e.g., GPS) [34], [35] and track the floor while the user moves around with the smartphone. The floor number will be used to provide the correct floor plan in stage two. Consequently, the stability and accuracy of the floor detection significantly determine the performance of the entire scheme.




(a) Raw and smoothed barometric data.

(b) Smoothing and delay effect.

**FIGURE 4.** Examples of raw and smoothed barometric data and associated time lag effect.

The barometer measurement is primarily based on altitude. However, it is also affected by short-term noise from user activity and the ambient environment, as well as long-term drift caused by weather in practical applications. We presume that for most cases, the height of the device relative to the user's body is within a specific range, as shown in Figure 3. There are several representative modes of smartphone usage listed: (a) calling, (b) typing, (c) swinging, and (d) pocket [36]. Here, (a) represents the highest case of a smartphone, (c) represents the lowest case, and (d) represents the different surroundings such as when it's in the pocket or bag. The data in Figure 4 shows an example of barometric data collected including the above cases. From Figure 4,

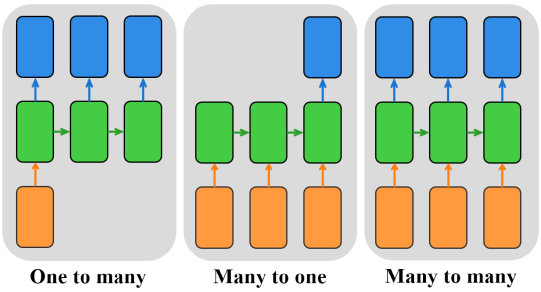



**FIGURE 5.** Various tasks of the RNN model.

we observed that the means and variances of barometric data collected on the same floor differ due to changes in height and environment. In addition, outliers appeared when the user switched smartphone usage cases. We demonstrated the possibility of recognizing step actions through time-series pressure data and proposed an MLP-based step action recognition approach to identify the user's floor transition from noisy pressure data [33]. However, because we assumed in previous work that the user always carries the smartphone in front of their body (i.e., do not change the height of the smartphone with respect to the body), the user behavior is easily recognized as the stair step under the free activity scenarios. The typical approach to smoothing these transient pressure fluctuations is to utilize a lowpass filter, such as simple moving average (SMA) or weight smoothing, as follows [37].

$$x_t^d = \frac{\sum_{i=t-m+1}^{t} x_i}{m}, \tag{2}$$

$$x_t^d = (1-\beta)\cdot x_{t-1}^d + \beta \cdot x_t. \tag{3}$$

Here, $x_t$ and $x_t^d$ indicate the $t$-th sampling data and smoothed sampling data, respectively. $m$ is the size of the average window, $\beta$ is the smooth factor, and they are used to control the smoothing effect. The trade-off between delay and smoothing effect is a known problem with the smoothing algorithm. Figures 4(a) and 4(b) demonstrates the smoothing and delay effect of (3) with different values of $\beta$, and they were smoothed under a sampling frequency of 20Hz. In Figure 4(b), the barometer reading with $\beta = 0.6$ only smoothed out a few severe outliers, and caused a delay of less than 0.05s; while the smoothed barometer reading with $\beta = 0.03$ exhibited a clear height correlation, but the trade-off is causing a delay of 5.1s, which means that the user may take 6–8 steps before the pressure measurement shows the characteristics of the flat floor. These missing steps constitute a significant error in our system because the floor transition signal generated from floor detection is exploited in the PF component to correct the PF's estimation (to be described in Section III-B6). The time difference between the step action and the 2D position should be as small as possible to ensure their correlation. Therefore, we design a Seq2Seq model that can predict the correct step action from barometric data containing noise and outliers instead of heavily relying on the smooth filter. In addition, because a delay of 0.05s is acceptable, we utilize (3) with $\beta = 0.6$ to smooth the data.

**TABLE 1.** Time and weather conditions during training data collection.

| Time | Temperature | Weather | Humidity |
|---|---|---|---|
| 2022-06-16 15:15 | 24 °C | Cloudy | 50% |
| 2022-06-16 21:20 | 18 °C | Clear | 83% |
| 2022-06-17 16:53 | 26 °C | Sunny | 65% |

### B. MODEL SELECTION, DATA PROCESSING, AND TRAINING RESULT

We found the potential of the Seq2Seq model for handling noisy time-series pressure data. The Seq2Seq is an encoder-decoder framework model using recurrent neural network (RNN) [38] and consists of three components: encoder, decoder, and state vector that connects them. The encoder is responsible for compressing the input sequence into a state vector as the initial hidden state of the decoder, and then the decoder predicts the probability of each class from the state vector. The Seq2Seq model can deal with various tasks such as many-to-many, many-to-one, and one-to-many, as shown in Figure 5. In this study, we apply the Seq2Seq model to the many-to-one task. Unlike MLP, the most widely employed DL model, the output of Seq2Seq is determined by both current and previous inputs, thus it is well suited for handling sequences such as time-series data. In this paper, the Seq2Seq model receives time-series pressure data that includes the previous and current barometer readings to confirm whether the current barometric fluctuation is caused by noise or height change.

The training data was collected from Hyeongnam Engineering Building at Soongsil University, with 22 stairs between each floor and a height of 17cm for each stair. Regarding the data collection, a barometer reading is recorded in the smartphone database once a step is detected. The step's label is determined based on the region where the user is located. For example, if a user enters the upstairs at the 30th step and exits the staircase at the 60th step, then the labels for the 30th to the 60th steps would be assigned as "Going up." Moreover, unlike IMU sensors, barometric measurements are primarily driven by changes in altitude. Thus, the impact of individual user characteristics (e.g., weight, gender, height) on barometric measurements is negligible compared to the noise induced by user activity. The primary impact of different users on barometric measurements comes from their walking styles on the staircase, such as some pedestrians taking two stairs in one step. Therefore, we ensure the inclusion of features from various movement patterns in the training data by randomly taking one or two stairs while climbing stairs during data collection. There were 3,726 barometric data collected for model training, which included 14 events of ascending stairs and 14 events of descending stairs. The time and weather conditions during the data collection are shown in Table 1. Next, we applied a data augmentation method to the collected data, as follows.

$$\delta_i = x_{i+1}^d - x_i^d, \tag{4}$$

**TABLE 2. Hyperparameters used in model training.**

| Model parameters | Values |
|---|---|
| Number of trainable parameters | 2,355 |
| Activation function of hidden layers | Tanh |
| Activation function of output layer | Softmax |
| Initializer | Xavier uniform |
| Loss function | Cross entropy |
| Optimizer | Adam |
| Learning rate | 0.001 |
| Number of epochs | 30 |

$$x_1^a = x_{last}^d - \delta_1, \tag{5}$$
$$x_i^a = x_{i-1}^a - \delta_i, \tag{6}$$

where $\delta$ is the pressure difference of adjacent steps, $x_k^d$ is the $k$-th smoothed pressure data, $x_{last}^d$ is the last pressure data of the collected dataset, and $x^a$ is the barometer data generated by data augmentation. Through data augmentation, the ascending and descending pressure data can be mutually transformed. A thorough analysis on the reasoning behind, feasibility, and optimization strategies for ensuring high dataset quality in this data augmentation method can be found in [33]. We concatenated $x^d$ and $x^a$ as training data $x^w$. By performing data augmentation, the size of the training dataset was increased to 7,098, and the number of events for ascending and descending stairs was expanded to 28.

Subsequently, a sliding window method was used to convert the data into learnable forms, as follows.

$$X_k^w = \{x_{k-s+1}^w, \cdots, x_k^w\}, \tag{7}$$

where $s$ stands for the window size. $X_k^w$ is a subset of the dataset, which contains the pressure change from the previous $s$ steps. Its label is determined by the label of the $k$-th step. It is recommended that the value of $s$ be between 10 and 20 to ensure that the barometer sequence of size $s$ is sufficient to represent the pressure change information over a short period, and $s = 15$ in this paper. Next, mean centering is used to shift the feature's center to 0.

$$\begin{aligned} X_k &= X_k^w - \mu_k \\ &= \{x_{k-s+1}, \ldots, x_k\}, \end{aligned} \tag{8}$$

where $\mu_k$ is the mean of $X_k^w$, and $X_k$ is the input of the model. Our model predicts the step action based on the pressure changes over the past $s$ steps. The reason for employing mean centering instead of normalization or standardization lies in our desire to shift the data close to 0 to aid the model training while refraining from scaling operations that modify the data's original units. Furthermore, the main advantage of using fixed-length $X_k$ as the input (i.e., many-to-one) to predict a step action instead of generating the output whenever each input is read (i.e., many-to-many) is that the Seq2Seq model is capable of fitting the trajectories of different lengths well and eliminates the effect of outliers that accumulate over time and the weather factors. The performance of the many-to-many approach becomes unstable in long path scenarios, which results from the accumulation of outliers in previous inputs. Additionally, early barometer data have little correlation with the current step action. In contrast, the fixed-length input means that the model's prediction only depends on past $s$ measurements and has approximate performance for a sequence with arbitrary lengths. Furthermore, since pressure fluctuations typically require tens of minutes to hours to produce a significant altitude drift [39], a pressure sequence with a size of 15, which corresponds to a pressure change over a 10 seconds period, enables the avoidance of long-term errors arising from weather factors.

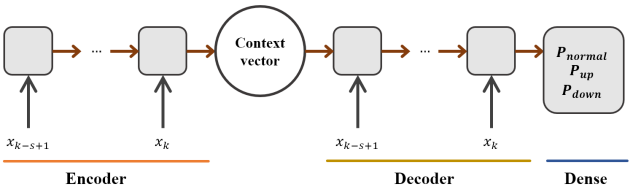



**FIGURE 6. Seq2Seq model overview.**

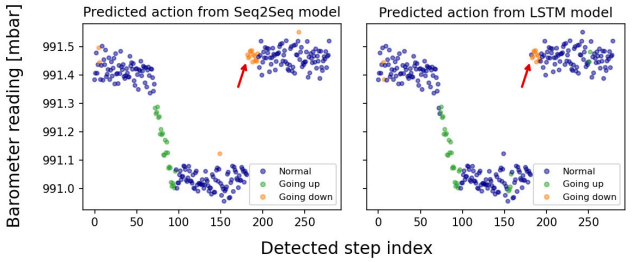



**FIGURE 7. Step action recognition examples.**

Table 2 lists the hyperparameters adopted in the Seq2Seq model. The hyperbolic tangent (Tanh) was used instead of Sigmoid for faster and better training. Xavier uniform was utilized as a weight initializer to make the variance of the output of each layer roughly equal to the variance of its input, to prevent the gradients from becoming too large or too small during training [40]. They are both commonly used hyperparameters in RNN training, and their definitions are given in (9) and (10), where the $fan_{in}$ and $fan_{out}$ indicate the number of input units and output units in the weight tensor, respectively.

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1}, \tag{9}$$

$$W_{i,j} \sim \mathcal{U}\left(-\sqrt{\frac{6}{fan_{in} + fan_{out}}}, \sqrt{\frac{6}{fan_{in} + fan_{out}}}\right). \tag{10}$$

Figure 6 provides an overview of the proposed Seq2Seq model. The decoder and encoder of the model are both composed of an LSTM layer with 16 hidden units [41]. The model is initialized using the uniform Xavier initialization, then sequentially receives past $k$ barometric data. As the

model processes the time-series data, it utilizes the hidden state transitions of the Seq2Seq model to extract the temporal dependencies in the data, thereby effectively extracting the inherent features associated with ascending or descending stairs. Subsequently, the dense (or fully-connected) layer outputs the probabilities of three distinct classes: ''Normal,'' ''Going up,'' and ''Going down.'' The model then updates its weights according to the ground truth. Each of these actions represents a potential pattern of walking behavior. Owing to the lightweight architecture of our Seq2Seq model, it achieves training convergence in less than 30 epochs.

Figure 7 shows simple step action recognition examples. The test data was collected within a building with a floor interval of 3.5m, where the tester ascended the stairs and then returned via the elevator. As can be observed in Figure 7, the barometric data exhibits notable noise even when the tester is walking on a flat floor, making it difficult to differentiate between pressure differences caused by changes in altitude and those caused by noise. For such data, the Seq2Seq model (left in Figure 7) correctly identifies the majority of step actions in real-time. Additionally, the proposed Seq2Seq model exhibits sensitivity to sudden changes in barometric data, which allows to immediately detect the first step after the user takes the elevator, as indicated by the red arrow in Figure 7. This characteristic is crucial for our approach because it enables the immediate detection of an elevator event from the first step through the floor decision algorithm described in the following subsection. Furthermore, we experimented with simple LSTM models, which are commonly used in many-to-one tasks for step action recognition, and found that they lacked the ability to detect a change in step action from the first step of elevator use. The right plot of Figure 7 shows a prediction by an LSTM model, which is constructed with 12,771 trainable parameters and composed of two LSTM layers with a dropout layer of rate 0.2 and 0.1 after each, followed by a dense layer [42]. The red arrow in the figure confirms that the LSTM model did not immediately detect the elevator step. The potential reasons for this discrepancy between the two models could include:

1) Enhanced representation capabilities: The Seq2Seq model offers more robust representational capabilities through its distinct encoder and decoder for handling the input and the output.
2) Better performance with long-term dependencies: The encoder condenses the entire input sequence into a context vector, which the decoder uses to generate predictions. This mechanism aids in capturing long-distance dependencies within the input sequence.

Therefore, Seq2Seq is employed for this study.

### C. FLOOR DECISION ALGORITHM WITH RELATIVE PRESSURE MAP

In this study, the user's vertical movement is represented by step actions instead of directly calculating the altitude. The advantage of this approach is that, due to factors such as pressure drift and user behavior, the barometer reading can vary even if the altitude is the same (i.e., the user is on the same floor). These short-term and long-term noises in barometer measurements cause the height calculation to be inaccurate, such as being identified as another floor level. [24] applied an iterative optimization method to track pressure changes and eliminate the drift effect in real-time, but constant calibration is laborious. The step action sequence is a form of data without atmospheric pressure value, thus avoiding the above problems. The only requirement is to ensure that DL predictions are accurate and robust enough. Based on the step action sequence, we know the exact step of the floor transition that occurred. Therefore, this study estimates the

**Algorithm 1** Floor Decision Algorithm

**Input:** predicted action $act$ of current step
**Output:** floor detection of a step
*Initialization*: $n_{wait}, p_{elevator} \leftarrow 3,\ 0.35$
**if** $act$ is same as previous & queue is empty **then**
  Calculate pressure value $p$ with a lowpass filter
  **return** floor detection result same as previous step
**else**
  Enqueue current IMU and barometer data into queue
  Count number of consecutive occurrences $n_{con}$
  **// Decide whether to change walking state**
  **if** $n_{con} > n_{wait}$ **then**
    Generate a floor transition signal
    Calculate average pressure value $p_a$ of the step data in queue
    Obtain pressure difference by $p_{delta} \leftarrow p - p_a$
    **if** $act \neq$ Normal **then**
      **// Decide transition type**
      **if** $p_{delta} < p_{elevator}$ **then**
        transition type $\leftarrow$ stairs
      **else**
        transition type $\leftarrow$ elevator
      **end if**
      Update and dequeue the steps in queue
      **return** transition type and direction
    **else**
      **// Update floor number when back to floor**
      floor number $\leftarrow$ UpdateFloor()
      Update floor number of current step
      Update and dequeue the steps in queue
      **return** floor number
    **end if**
  **else**
    Wait for prediction of next step
  **end if**
**end if**

**Algorithm 2** UpdateFloor()

**Input:** relative pressure map RM, pivot floor, $p_{delta}$
**Output:** floor number
**// Obtain pressure value of pivot floor from RM**
$p_{pivot} \leftarrow$RM[pivot floor]
**// Calculate new pressure value after floor transition**
$p_{new} \leftarrow p_{pivot} + p_{delta}$
floor number $\leftarrow$ the floor with the closest pressure value to $p_{new}$ in RM.
**return** floor number

height change based on barometric pressure difference only when the region changes are detected.

Algorithm 1 explains the proposed floor decision algorithm. We obtain the step action sequence according to the prediction of the Seq2Seq model, which implies the user's vertical movement. For example, a sequence of ''Going up'' steps means that the user is climbing the stairs. We can update the floor number according to the number of such steps. However, this method has two drawbacks: 1) the number of steps required to walk up a floor varies depending on the user's climbing method, and 2) false floor transitions on a flat floor are also recognized as floor transitions. Therefore, it is necessary to calculate the height difference through the barometer reading. Although the atmospheric pressure drifts due to weather conditions, the pressure difference in a short time interval is credible [43].

Before applying the step action sequence, incorrect DL model predictions need to be eliminated. Outlier data is typically isolated and unordered, whereas height-induced pressure changes are persistent and ordered. Therefore, confirming a floor transition through multiple step actions can eliminate most of the incorrect predictions. When detecting a different step action with previous steps, the proposed method does not immediately confirm the region changed, but instead enqueues the step data in memory and waits for the prediction of new step actions until the queue length exceeds $n_{wait}$. At the point the current region is changed, a floor transition signal is generated.

A method to mitigate pressure drift is to calculate the pressure difference based on the exact steps of entering and exiting the transition zone, ensuring the minimum time interval between them. This can easily be achieved according to the Seq2Seq model's prediction. The floor decision algorithm first calculates the pressure difference $p_{delta}$ when a region change is confirmed. In particular, the pressure value $p$ of the previous floor is calculated through a lowpass filter (line 2 in Algorithm 1), while the current pressure value $p_a$ is obtained from the average of the data in the queue (line 10 in Algorithm 1). This ensures that the time interval in calculating the pressure difference is minimized to avoid the impact of long-term drift errors and smoothens the pressure values of the previous floor and current floor against short-term noises. At line 12 in Algorithm 1, a step action not equal to ''Normal'' indicates that the user moves from the flat floor to an elevator or stairs, and the decision between elevator or staircase is made based on the pressure difference $p_{delta}$. Otherwise, if the step action is ''Normal,'' it means the user is returning to the flat floor, and the floor number must be updated accordingly.

The new floor number is obtained from the UpdateFloor(), as shown in Algorithm 2. UpdateFloor() reads the pressure value of the pivot floor and calculates $p_{new}$ by adding the pressure difference. The floor with the closest pressure value to $p_{new}$ is mapped as the new floor number. Next, the floor number of the current step and steps in the queue is updated. The proposed floor detection can immediately estimate floor number when a step is detected, with only a delay of $n_{wait}$ steps when the region changes. In particular, this delay refers to the delay of output results, rather than the time difference between vertical movement information and 2D position, because sensor data at the same time is saved in the queue when a different step action is detected.

## III. PDR-PF WITH CLUSTERING

The 2D location calculation is introduced in this section, which is activated when a user starts localization. As shown in Figure 8, the PF fuses data from floor detection, smartphone database, and PDR to calculate the 2D location of the user's $k$-th step. The PF first updates the state of particles based on the transition signal from the DL-based floor detection or PDR estimation and then utilizes mean shift to cluster them. Afterward, the location estimation and resampling are performed based on the clustering results.

PDR is described first because it drives the entire scheme. Then, PF will be introduced in detail, including map constraint, clustering, and CN matching-based location correction.

### A. SMARTPHONE SENSOR-BASED PDR

The Android and iOS operating systems respectively provide the SensorEvent and CoreMotion classes to report motion information from the onboard sensors of devices [44], [45], which enables us to estimate the location through PDR. The PDR approach suggests that a step can be expressed as a distance and an angle referring to the previous state, i.e., the current location is determined by the current displacement and previous location. In mathematics, the location of the $k$-th step $P_k(x_k, y_k)$ can be expressed as

$$P_k(x_k, y_k) = \begin{bmatrix} x_k \\ y_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \end{bmatrix} + \lambda_k \begin{bmatrix} \sin(\alpha_k) \\ \cos(\alpha_k) \end{bmatrix}, \quad (11)$$

where $\lambda_k$ and $\alpha_k$ are the stride length and heading direction of the $k$-th step. Next, we introduce the step detection, stride length calculation, and heading direction estimation of PDR.

#### 1) STEP DETECTION

When a pedestrian walks, the vertical acceleration presents periodic sine waves, with each step represented by a local peak or valley in the acceleration. This pattern enables step detection by recognizing these peaks and valleys in vertical acceleration. To counteract the impact of device tilts on sensor measurements, rotation transformation needs to be performed to convert the accelerometer readings from the local coordinate system (LCS) to the global coordinate system (GCS). The rotation matrix $R$ can be calculated through several methods, including quaternions and sensor fusion [46], [47]. In this paper, we utilized getRotationMatrix() function in SensorManager class to compute the rotation matrix by cross-product of accelerometer and magnetometer measurements. The acceleration vector in the GCS $A_t^G$ can then be

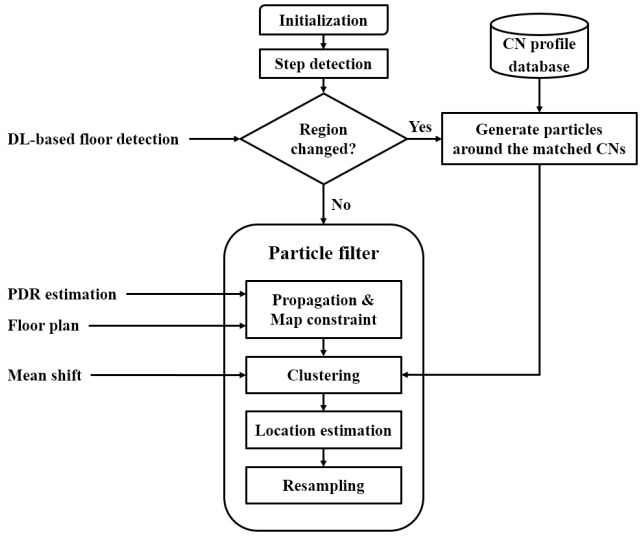



**FIGURE 8.** Flowchart of the PF used in our scheme.

determined as

$$A_t^G = R_t \cdot A_t^L, \tag{12}$$

where $A_t^L$ are the acceleration vectors in LCS. Subsequently, a valid step is defined as

$$\{a_t > a^{upper},\ a_{t+\Delta t} < a^{lower},\ 0.15\text{s} < \Delta t < 0.6\text{s}\}, \tag{13}$$

where $a_t$ is the vertical acceleration, and $\Delta t$ is the time interval between the peak and valley. We established the amplitude thresholds $a^{upper} = 1.0\text{m/s}^2$ and $a^{lower} = -0.8\text{m/s}^2$ in this study.

#### 2) STRIDE LENGTH ESTIMATION

Theoretically, displacement can be calculated by integrating acceleration. However, due to the limited accuracy of the onboard accelerometer, the step length is typically derived based on features of the acceleration data, such as peaks/valleys and variance [48], [49], [50]. We use [51] to estimate the stride of a step, which establishes a relationship between vertical acceleration and step length, as follows.

$$\lambda_k = \tau_k \cdot \sqrt[4]{a_{max,k} - a_{min,k}}, \tag{14}$$

where $a_{max,k}$ and $a_{min,k}$ denote the maximum and minimum vertical acceleration value during $k$-th step, and $\tau_k$ is the coefficient that can be specified for different subjects. Due to the influence of gravity, the variance of acceleration for steps taken in stairs is typically greater than that for steps taken on flat ground, and causes errors in the stride estimation within stairs. To improve the step length calculation, we adjust the value of $\tau_k$ based on the region where the user is located using (15). The region information is obtained through the proposed floor detection method.

$$\tau_k = \begin{cases} \rho \cdot \tau, & \text{if stairs} \\ \tau, & \text{otherwise.} \end{cases} \tag{15}$$

Here, $\rho$ denotes a scale factor used to compensate for step length calculation in stairs. It is recommended that the value of $\rho$ ranges from 0.5 to 0.8, and we empirically set $\tau = 0.43$ and $\rho = 0.6$ in this paper. In fact, the exact value of $\rho$ is not strictly required in our scheme, since a correction will be made both when detecting entering and exiting stairs (to be described in Section III-B6).

#### 3) HEADING DIRECTION ESTIMATION

The accuracy of heading direction in PDR is crucial because the main source of error comes from distortions of direction. There are two main methods for the heading direction estimation through IMU sensors: 1. calculate the change in angle over a period of time by integrating the gyroscope reading $\omega$, and 2. determine the absolute orientation $\alpha^m$ relative to the north through the accelerometer and magnetometer readings [46]. In this paper, $\alpha^m$ is obtained from getOrientation() in SensorManager class. For the iOS platform, $\alpha^m$ can be retrieved from the CLHeading class in the CoreLocation framework [52], [53]. The orientation calculated by integrating the angular velocity tends to slowly drift away from the actual orientation, while the orientation derived from the accelerometer/magnetometer can be easily distorted by surrounding electronic devices. Thus, the common practice is to fuse these two measurements based on certain criteria rather than relying on a single angle source. A typical orientation fusion can be expressed as follows [54]:

$$\begin{aligned} \alpha_k &= \gamma \cdot (\alpha_{k-1} + \omega_k \cdot \Delta t) + (1-\gamma) \cdot \alpha_k^m \\ &= [\alpha_k^x\ \alpha_k^y\ \alpha_k^z]^T, \end{aligned} \tag{16}$$

where $\gamma$ is a coefficient that determines the fusion proportion, with its value ranging between 0 and 1. A larger value of $\gamma$ (e.g., $\gamma = 0.99$) indicates a stronger influence of the angle calculated by the gyroscope in the direction update. Furthermore, since the tilts of device affect the direction estimation, the orientation is generally transformed into the GCS through a rotation matrix. However, in this study, we assume that the user's smartphone points forward during the localization stage, and thus we directly utilize the calculated azimuth (i.e., the z-axis angle) as the heading direction.

### B. PF WITH CLUSTERING AND CORRECTION

The PF is a sequential importance sampling (SIS) method used to estimate a system's state from noisy and incomplete measurements. A PF represents the distribution of possible states of the system using a set of random particles. These particles are propagated through the state space based on measurement data, and the weights of the particles are updated each iteration [55]. The weights reflect the likelihood that each particle represents the true state of the system, given the measurement data. For more description of the PF in general, see [18], [56].

As mentioned above, to improve the performance of PF in inadequate measurements, more user mobility information must be combined. Therefore, two techniques are proposed in this study: mean shift and CN matching-based location correction. Mean shift is used to cluster particles based on spatial

distance and select the main particle cluster to calculate the location, thereby improving location estimation performance. Meanwhile, CN matching-based location correction enables the PF to utilize the floor transition signal obtained from the floor decision algorithm and prior knowledge from the floor plan to accelerate particle convergence, correct the particle's state, and optimize particle updates in subsequent tracking. These two technologies are integrated into 2D localization to stabilize and sustain the proposed PF. Next, we introduce the components of the PF as depicted in Figure 8.

#### 1) PARTICLE INITIALIZATION

The PF is activated when the user starts localization (e.g., press the localization button). Since the user's location is not given, the PF first acquires the current floor plan based on our floor detection, and then $N_0$ particles are uniformly dispersed on the entire map. The attribute of the $i$-th particle at the $k$-th step, including 2D coordinates, heading, weight, and cluster number, is as follows.

$$\boldsymbol{A}_k^{(i)} = [P_k^{(i)}, \theta_k^{(i)}, w_k^{(i)}, c_k^{(i)}], \tag{17}$$

where $P_k^{(i)} = (x_k^{(i)}, y_k^{(i)})$ denotes the 2D location and $\theta_k^{(i)}$ denotes the heading direction. We assume that the orientation measured by accelerometer/magnetometer sensors is Gaussian distributed around the true orientation and thus generates $\theta_0^{(i)} \sim \mathcal{N}(\alpha^m, \sigma^{ori})$ as the initial heading of the $i$-th particle. $w_k^{(i)}$ and $c_k^{(i)}$ stand for the particle weight and cluster label, respectively, and they are initialized as $1/N_0$ and $-1$.

#### 2) PROPAGATION, UPDATE, AND MAP CONSTRAINT

In this subsection, the location $P^{(i)}$, heading $\theta^{(i)}$, and weight $w^{(i)}$ of particles are updated. Whenever a step is detected, the step length and direction are calculated by the PDR and fed into the PF. The particles propagate based on the current state and PDR estimation. In addition, Gaussian errors with zero mean and standard deviation $\sigma^l$ and $\sigma^o$ are respectively added to step length and heading direction update to simulate the effect of the uncertainty and noises of measurements, as well as avoid the loss of diversity among the particles.

The weight of the particles is determined by the system evaluation function, with the commonly used evaluation parameters including direction or distance [57], [58]. We apply Gaussian distribution to calculate the weight of the particles, as follows [59].

$$w_k^{(i)} = w_{k-1}^{(i)} \cdot \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{\left[\frac{(\Delta x_k - \Delta x_k^{(i)})^2 + (\Delta y_k - \Delta y_k^{(i)})^2}{-2\sigma^2}\right]}, \tag{18}$$

where $\Delta$ represents the displacement. Furthermore, map information is used to detect collisions with walls and eliminate invalid particles to guide the propagation of particles. In this study, the digital map is derived from the floor plan and comprises blocks and lines. The blocks represent unreachable areas and are used to eliminate impossible particles during particle generation (e.g., initialization and resampling). The lines represent the walls and are used to kill particles that cross them. Whenever the particle state is updated, the collision detection algorithm checks whether the particle crosses a line or falls within a block based on its location at the previous and current time steps. The weight of invalid particles is set as zero, while the valid particles are retained. After the map constraint, the weights are normalized to ensure their sum is one.

#### 3) CLUSTERING USING MEAN SHIFT

In this study, clustering is used to group the surviving particles and find the centroid of each cluster. With the clustering results, we 1) confirm the convergence of particles, 2) optimize the location estimation from multiple modes of particle distribution (to be described in the location estimation subsection), and 3) adjust the number of particles dynamically subject to reduce computational burden without sacrificing performance (to be described in resampling subsection). Before explaining these, we introduce the clustering algorithm.

We utilized mean shift as the clustering algorithm [60]. Mean shift is a non-parametric and centroid-based technique that defines a region around each data point and moves the center (or mean) of that region toward the densest part of the region until it converges to the local maximum. The K-means is the most widely used clustering algorithm, in which the main parameter is the number of clusters $K$. However, K-means cannot delineate non-convex clusters. In addition, we want to count the number of clusters instead of manually providing this parameter, thus it is unsuitable for our scheme. Reference [14] used Density-Based Spatial Clustering for Applications with Noise (DBSCAN) to group the particles and detect outliers according to their density, where the outliers are annexed to a new cluster. Although mean shift does not identify outliers, having the clustering algorithm recognize and operate on these outliers is not desired in our scheme. A surviving particle contains relative information iterated from the PDR update and may be correct. Therefore, these ''outliers'' remain until they are killed by a collision detection algorithm or grow up to a larger cluster. Furthermore, we will introduce how to exclude the effects of these ''outliers'' in the location estimation subsection. The motivation for using mean shift is that it is simple, fast, and can delineate arbitrarily shaped clusters and count the number of clusters automatically, which is well suited for the dynamic and irregularly shaped particle cloud. In the proposed scheme, mean shift evaluates the similarity between particles based on their location on the orthogonal coordinate system. In particular, normalization is not performed as we want to retain the unit of features to express the real distance. The main parameter of mean shift is the bandwidth $B$, which is set as 3m in our scheme. This is a relatively large value, which implies that the spatial separation between particles may need to approximate the distance across a room for them to be classified into distinct clusters. The purpose of utilizing mean shift is to describe particle distribution and discover

dispersed particle clusters rather than dividing a converged particle cloud into several clusters. Therefore, a slightly larger value of $B$ is recommended.

In the initialization phase, particles are evenly distributed across the state space area. As the new step is detected, the particle cloud converges to where the user might be. The early particles are meaningless until the filter gathers over several iterations to represent the user's possible location. The clustering results can be exploited to explain the current particle distribution: the more dispersed the particles are, the more clusters and the more distant the centroid are from each other. Therefore, we assume that the PF has converged enough to provide valid location information when only one cluster exists. However, particles do not easily cluster in one place, and waiting for the number of clusters to decrease to one requires a long time. Thus, it is also considered to have converged when the largest cluster's weight exceeds 80% of the total weight. The location estimation starts after convergence.

#### 4) LOCALIZATION ESTIMATION

In PF, the estimate of user location is obtained by taking the weighted average of the surviving particle's location. Mathematically, they can be expressed as follows.

$$\begin{bmatrix} x_k \\ y_k \end{bmatrix} = \begin{bmatrix} \sum_i^{n^s} w_k^{(i)} x_k^{(i)} \\ \sum_i^{n^s} w_k^{(i)} y_k^{(i)} \end{bmatrix}, \tag{19}$$

where $n^s$ indicates the number of surviving particles. As we mentioned before, due to the building structure and uncertainty of the measurement, the particles may become spread out over multiple modes during propagation. Furthermore, particles can be regenerated in one or a few locations based on the user's motion information and CN profile in our scheme (to be discussed in the location correction subsection). As a result, it is difficult to accurately estimate the user's location using the entire particle set. Therefore, we calculate the location from the selected particles through the results of clustering. First, we ignore tiny clusters whose weight is less than 5-10% of the total weight. Second, when the weight of the largest cluster exceeds 70% of the total weight, the location is calculated only using that cluster. Otherwise, all the particles are used to estimate location.

#### 5) RESAMPLING

One drawback of the SIS method is the degeneracy of weight, where the importance weights concentrate on a few particles while the majority of particles have weights close to 0 after multiple iterations. Resampling is a common solution to handle this issue which ignores the particles with low weights and multiplies the particles with high weights. However, resampling causes the particles to lose diversity, resulting in sample impoverishment [11], [18], [61]. A typical approach to handle this issue is to implement resampling only at certain iterations [29]. Hence, instead of resampling every iteration, we only perform it when 1) five iterations have passed since the last resampling, or 2) the number of surviving particles is less than $N_p/5$, where $N_p$ is the current maximum number of particles, and its value is dynamically adjusted based on the clustering result.

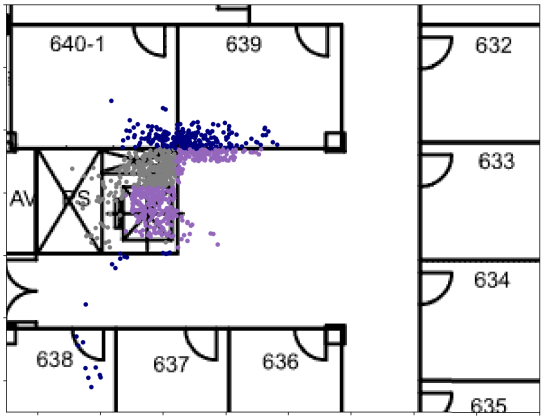



**FIGURE 9.** After the user took the elevator, some particles appeared on the other side of the wall. The gray particles were eliminated due to falling within the block, the purple particles remained valid, and the navy particles were not eliminated but were incorrect.

Theoretically, a large number of particles can reduce the variance of the estimated posterior distribution, leading to more accurate state estimation. However, the PF is a computationally intensive method, and increasing the number of particles increases the computational cost of the algorithm. This problem is magnified in real-time tasks and on the mobile platform. Therefore, balancing the number of particles with computational efficiency is important for optimizing the performance of a PF. We assume that fewer particles can still achieve good localization results when the particles are concentrated in one area, such as a closed corridor, while if the particle distribution is dispersed, more particles are needed to explore the feasible paths. Thus, the proposed scheme dynamically adjusts the number of particles $N_p$ based on the number of clusters to achieve good performance with lower overhead, as follows.

$$N_p = \min(15, n_{cluster}) \times N_c, \tag{20}$$

where $n_{cluster}$ is the number of clusters, and $N_c$ stands for the number of particles assigned for one cluster. In the initialization phase, there can be many clusters due to the dispersion of particles. To prevent generating too many particles, the maximum value of $N_p$ is $15 \times N_c$. A large number of particles $N_0$ are only used at the first iteration since PF has to cover the interesting state space areas. Then, $N_p$ is determined by (20).

#### 6) CN MATCHING-BASED LOCATION CORRECTION

The performance of PF is improved according to the map constraint. However, there are still 3 practical problems that have to be considered:

- In the initialization phase, the convergence of PF is slow due to inadequate measurement, which can take over a hundred steps in large buildings.

- When a user changes the floor, the particle's state may not be available on the new map because they were iteratively updated on the previous floor plan. For example, particles cannot reach a room through the wall due to collision detection algorithms, but the floor change caused the particles to appear in this room because there was no wall in the previous floor plan. Figure 9 illustrates this example. It is an important issue to efficiently extend PF to 3D scenarios.
- Due to the absence of absolute location information (e.g., RSS), the PF cannot correct itself when particles converge to the wrong location.

To solve these problems, we present CN matching-based location correction, which combines the floor transition signal from the floor decision algorithm and prior knowledge from the map. Whenever a step is detected, the floor decision algorithm is first checked for the presence of a floor transition signal. If a transition signal is detected, particles are corrected based on matched CNs. Otherwise, particles are updated through PDR estimation.

Before the implementation of location correction, the CN profile needs to be established for searching and matching. When the user changes floors, the proposed floor decision algorithm outputs a transition signal that includes the vertical motion feature. This vertical motion feature is valuable for the probabilistic approach such as PF, and it is used to match the CNs on the floor plan to narrow down the possible area where the user is located to one or few places. CNs are established around vertical transition facilities (e.g., elevator, stairwell, etc.) based on their usage and location. The CN's profile is designed as follows.

$$CN_{set} = \{f_j, P_j^{cn}, Type_j, D_j, \theta_j^{cn}, j = 1, \ldots, n^{cn}\}. \quad (21)$$

Here, $f_j$ and $P_j^{cn}$ respectively indicate the floor level and location of the CN, $Type_j$ is the floor transition type (i.e., stairs or elevator), $D_j$ is the transition direction, $\theta_j^{cn}$ stands for the possible direction range, and $n^{cn}$ is the total number of CNs. Because the structure of the facilities restricts the direction of user movement (e.g., an elevator having only one exit direction), each CN is assigned a $\theta^{cn}$ that is used for initializing the heading of the regenerated particles.

There is an example of CN matching-based location correction. A user takes an elevator from the first floor to the third floor, and the floor decision algorithm returns the first floor as the previous region and the third floor as the current region, with the mode of vertical transportation being recognized as the elevator. Therefore, the CNs that $f = 3$, $Type =$ elevator, and $D =$ ascending are matched for location correction. Location correction is performed according to the following criteria.

- If the PF has not converged, then $N_c$ particles are generated around each matched CN. Note that we established CNs near each vertical transportation, thus there is at least one CN that exists for matching.
- If the PF has converged, the closest CN is treated as the main CN based on the estimated user's location, and then $n_{cluster} \times N_c$ particles are generated near this CN. The remaining CNs are considered as sub-CNs, and $(n_{cluster} - 1) \times N_{sub}$ particles are generated near each sub-CN. In this study, we set $N_c = 150$ and $N_{sub} = 15$; thus sub-CNs will be identified as tiny clusters and do not affect the location estimation.

In this way, we not only extend PF to 3D scenarios but also correct particle states using information from matched CNs. Additionally, this method accelerates particle convergence, because the collision detection algorithm can easily eliminate incorrect clusters due to the narrow transition zone. Furthermore, small clusters generated based on the sub-CN provide an opportunity for rectification when the PF converges to the incorrect place: the correct cluster can grow after the other cluster disappears by colliding with the wall.

Note that while there are variants of the PF to improve the performance (e.g., the backtracking PF can optimize the localization results under multimodal particle distribution [14], [62]), the optimal strategy remains an open question. Our focus in this paper is not on providing the most accurate localization solution, but rather on offering sustainable and reliable long path tracking services with minimal human effort and limited measurement data. Finally, the 2D location and the floor number obtained from the floor decision algorithm are combined to represent the user's location in a multi-floor scenario.

## IV. EXPERIMENT RESULTS

In this section, multiple experiments were conducted to validate the performance of the proposed scheme. The data was collected using a smartphone app we built on Android OS. The experimental device was a Samsung Note 10+ smartphone equipped with a barometer and triaxial IMU sensors, with all sensors having a sampling rate of 20Hz. Whenever a step is detected by the PDR, our app stores the sensor data for location estimation based on Python 3.7.7. The creation of the Seq2Seq model, data augmentation and preprocessing, model training, and floor decision algorithm was performed using TensorFlow running on an Nvidia RTX 3070. In this paper, we consider indoor multi-floor localization as a two-stage process: 1) entering the building and moving, and 2) starting localization. In the first stage, the initial floor level is determined when entering the building, utilizing either the entrance information or alternative technologies, and the proposed DL-based floor detection tracks the user's floor without constraining user activity. In the second stage, when users start localization, we assume that the smartphone is pointed forward and remains stationary relative to the user's body. Furthermore, to reflect the performance under real-world usage, the tester exhibited complex mobility patterns during the experiments, such as varying walking speeds and occasionally taking one or two stairs in a single step when navigating staircases. The following subsections describe the experimental results, including the proposed DL-based

**TABLE 3.** Time and weather conditions during test data collection.

| Building | Time | Temperature | Weather | Humidity |
|---|---|---|---|---|
| Sung-deok Hall | 2022-08-12 16:14 | 31°C | Cloudy | 62% |
| Sung-deok Hall | 2022-10-18 21:00 | 8°C | Clear | 57% |
| Jilli Hall | 2022-10-13 20:45 | 14°C | Clear | 82% |
| Jilli Hall | 2022-10-14 15:04 | 22°C | Cloudy | 47% |
| Cho Man-sik Memorial Hall | 2022-10-13 14:52 | 23°C | Sunny | 38% |
| Cho Man-sik Memorial Hall | 2022-10-20 20:33 | 12°C | Clear | 82% |

**TABLE 4.** Paths and activities during experiments.

| # | Building | Paths and activities |
|---|---|---|
| 1 | Sung-deok Hall | F1 (P-T-C) $\xrightarrow{E}$ F2 (C-S) $\xrightarrow{S}$ F3 (S-T) $\xrightarrow{E}$ F6 (T-S) $\xrightarrow{E}$ F2 (S-P) $\xrightarrow{S}$ F1 (P) |
| 2 | Sung-deok Hall | B2 (T) $\xrightarrow{E}$ F2 (T-P) $\xrightarrow{S}$ F6 (P-C) $\xrightarrow{E}$ F3 (C-T) $\xrightarrow{S}$ F1 (T) |
| 3 | Jilli Hall | F1 (P) $\xrightarrow{E}$ F5 (P-C) $\xrightarrow{S}$ F2 (C-T) $\xrightarrow{E}$ F1 (T-P) $\xrightarrow{S}$ F3 (P) |
| 4 | Jilli Hall | F3 (T) $\xrightarrow{S}$ F1 (T-C) $\xrightarrow{S}$ F4 (C-P) $\xrightarrow{E}$ F5 (P-T) $\xrightarrow{E}$ F1 (T) |
| 5 | Cho Man-sik Memorial Hall | F1 (T) $\xrightarrow{E}$ F7 (T-C) $\xrightarrow{S}$ F4 (C-P) $\xrightarrow{E}$ F2 (P-T) $\xrightarrow{S}$ F1 (T) |
| 6 | Cho Man-sik Memorial Hall | F4 (P) $\xrightarrow{S}$ F7 (P) $\xrightarrow{E}$ F1 (P-C) $\xrightarrow{S}$ F3 (C-T) $\xrightarrow{E}$ F7 (T) |

floor detection and indoor localization under a multi-floor scenario. The results of the experiment were analyzed to assess the performance of the proposed methods. In addition, for ease of expression, F# represents the number of floors above ground, while B# represents the number of floors below ground in this section. For example, F5 means the fifth floor.

### A. DL-BASED FLOOR DETECTION

The proposed DL-based floor detection aims to track the user's floor without limiting user activity, and its evaluation results are presented in this section. When a step is detected, the Seq2Seq model first predicts the step action, and then the floor decision algorithm calculates the floor number and user's vertical movement information based on the step action, barometer reading, and relative pressure map. The floor decision algorithm describes a step using one of the following classes: a specific floor, "Stairs up," "Stairs down," "Elevator up," or "Elevator down." A flat step is referred to as a step on a flat floor in this section.

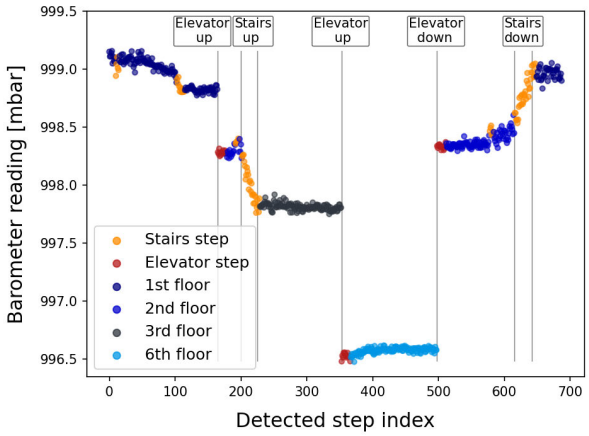



**FIGURE 10.** Floor detection result in Sung-deok Hall.

The accuracy rate (AR) is adopted to evaluate the accuracy of floor number calculation as follows.

$$AR^{FN} = \frac{\#\{\hat{f}_i \mid \hat{f}_i = f_i\}}{n_{floor}} \ (\times 100\%), \tag{22}$$

where $\hat{f}$ and $f$ are the predicted floor number and actual floor number, respectively. $n_{normal}$ indicates the total number of steps whose actual label is "Normal." This means the stairs and elevator steps are skipped in $AR^{FN}$ computation because they do not represent a floor number.

There were 9 floor detection experiments conducted in Sung-deok Hall, Jilli Hall, and Cho Man-sik Memorial Hall at Soongsil University. Each floor of the buildings had a gap of about 3.0–3.5m and was equipped with both elevators and stairs. With three experiments per building, we utilized 13 elevators, 12 sets of stairs, and moved across a total of 69 floors. The time and weather conditions during the test data collection are shown in Table 3. Table 4 provides a detailed description of the paths and activities performed during the experiments, where C, T, S, and P stand for calling, typing, swinging, and pocket cases, respectively, while $\xrightarrow{E}$ and $\xrightarrow{S}$ represent elevators and stairs. For example, F1 (C) $\xrightarrow{E}$ F2 (C-P) means the user goes upstairs from F1 to F2 by elevator, during which he finishes a phone call and puts his smartphone in his pocket. A floor detection result in Sung-deok Hall is shown in Figure 10, where both "Stairs up" and "Stairs down" steps are denoted as "Stairs step," and both "Elevator up" and "Elevator down" steps are denoted as "Elevator step." The walking path and activity executed in Figure 10 correspond to the first entry in Table 4. From Figure 10, it can be observed that the barometer data collected on the same floor exhibits drift and fluctuation due to changes in the environment and user behavior, resulting in unclear boundaries of atmospheric pressure differentiation between floors, while the proposed DL-based floor detection correctly calculates floor numbers. However, four false floor transition detections occurred on the first and second

**TABLE 5.** Confusion matrix for floor transition detection.

| | | Ground truth action | | |
|---|---|---|---|---|
| | | Stairs | Elevator | Floor |
| **Recognized action** | Stairs | **97.08%** | 0% | 6.58% |
| | Elevator | 0% | **100%** | 0% |
| | Floor | 2.92% | 0% | **93.42%** |

**TABLE 6.** AR for floor detection.

| Building | Sung-deok Hall | Jilli Hall | Cho Man-sik Memorial Hall | Total |
|---|---|---|---|---|
| $AR^{FN}$ | 93.81% | 92.56% | 93.77% | 93.42% |

floors when the user entered the buildings and changed the device carrying case. These false detections did not cause incorrect floor number calculations through the floor decision algorithm. Particularly, calculating the atmospheric pressure difference by using the mean of the data in the queue, rather than merely considering the pressure difference between two adjacent steps, provides a more accurate representation of the true pressure changes. In addition, the four false floor transition detections in Figure 10 can be reduced to one by raising the region transition threshold $N_{wait}$ to 6 in the floor decision algorithm. Since these false transition detections do not result in incorrect floor number calculation, a smaller $N_{wait}$ is chosen to minimize waiting time. Furthermore, Tables 5 and 6 present the evaluation confusion matrix and $AR^{FN}$ scores for floor detection. From Table 5, all elevator steps are accurately recognized, while some stairs and flat floor steps are misclassified. Table 6 shows $AR^{FN}$ scores of the floor calculation in the three experimental buildings. The accuracy of over 90% for all experiments indicates that the majority of estimated floor numbers are consistent with the actual floor numbers under conditions of complex user activity. Although some errors exist, they primarily occur when the user enters and exits the transition zone or during changes in activity, as shown in Figure 10. These false positive and negative errors result in a delay of $n_{wait}$ steps but do not cause inaccuracies in the computation of the floor level. In a scenario of unrestricted user activity, our goal is not to guarantee perfect accuracy in step action detection, but rather to prevent these potential errors from leading to incorrect floor number calculations.

Furthermore, we assess the accuracy of our Seq2Seq model against noisy data using raw barometer data. For comparison purposes, we incorporate the scores of the MLP model proposed in our previous work. The comparative data was collected in Jilli Hall's F1 to F5, and the collection paths are presented in Table 7. The dataset size is 4,704, including 2,352 raw barometer readings and 2,352 barometer readings smoothed by (3) with $\beta$ of 0.03. The smartphone was held

**TABLE 7.** Comparative data collection paths in Jilli Hall.

| # | Paths |
|---|---|
| 1 | F3 $\xrightarrow{S}$ F4 $\xrightarrow{S}$ F2 $\xrightarrow{S}$ F5 $\xrightarrow{S}$ F4 $\xrightarrow{S}$ F1 $\xrightarrow{S}$ F3 |
| 2 | F3 $\xrightarrow{S}$ F2 $\xrightarrow{S}$ F4 $\xrightarrow{S}$ F1 $\xrightarrow{S}$ F2 $\xrightarrow{S}$ F5 $\xrightarrow{S}$ F3 |
| 3 | F3 $\xrightarrow{E}$ F1 $\xrightarrow{S}$ F3 $\xrightarrow{E}$ F5 $\xrightarrow{S}$ F3 $\xrightarrow{E}$ F2 |
| 4 | F1 $\xrightarrow{S}$ F3 $\xrightarrow{E}$ F5 $\xrightarrow{S}$ F3 $\xrightarrow{E}$ F1 $\xrightarrow{S}$ F2 $\xrightarrow{E}$ F4 $\xrightarrow{E}$ F2 $\xrightarrow{S}$ F1 $\xrightarrow{E}$ F5 $\xrightarrow{E}$ F1 $\xrightarrow{E}$ F4 $\xrightarrow{E}$ F1 |

**TABLE 8.** Step action recognition accuracy for two models.

| Model | Smoothed data score | Raw data score |
|---|---|---|
| MLP | 91.8% | 76.1% |
| Seq2Seq | 90.4% | 87.0% |

in front of the body throughout data collection. Table 8 lists the scores of two models. Based on Table 8, we observed that the MLP model achieved accuracies of 91.8% for smoothed data and 76.1% for noisy raw data, which means it performs well for smoothed data, but its accuracy rapidly decreases for raw data. In contrast, the Seq2Seq model demonstrates a similar performance for both smoothed and raw data, yielding accuracies of 90.4% for smoothed data and 87.0% for noisy raw data, resulting in a mere 3.4%p decrease in accuracy compared to the smoothed data. This implies that the proposed Seq2Seq model demonstrates significantly better stability when dealing with noisy data compared to the MLP model. As aforementioned in Section II-A, to reduce the time difference between altitude information and 2D location to enhance the performance of the multi-floor localization scheme, the model should not rely on the filter. In addition, robustness to noisy data is more suitable for free activity scenarios. Therefore, the Seq2Seq is more appropriate for this scheme.

### B. INDOOR MULTI-FLOOR LOCALIZATION

To assess the performance of the proposed indoor multi-floor localization approach, we conducted long path tracking experiments on the third floor to sixth floor of Cho Man-sik Memorial Hall. Each floor of the building is composed of two sections, measuring 75m × 18m and 18m × 42m respectively. These areas encompass an open study space, and various corridors and rooms. Additionally, each floor is equipped with a staircase and a stairwell, as well as three elevators, two of which are adjacent to each other. We employ the 2D location's AR to compute the proportion of localization errors that fall below a particular threshold $\epsilon$, as follows [63].

$$d_k = \sqrt{(P_k - \hat{P}_k)^2}, \tag{23}$$

$$AR^{Loc} = \frac{\#\{d_k \mid d_k \leq \epsilon\}}{n} \times (100\%), \tag{24}$$

where $P_k = (x_k, y_k)$ and $\hat{P}_k = (\hat{x}_k, \hat{y}_k)$ indicate the estimated location and actual location of $k$-th step, respectively, and $d_k$ indicates the Euclidean distance between them. Furthermore,

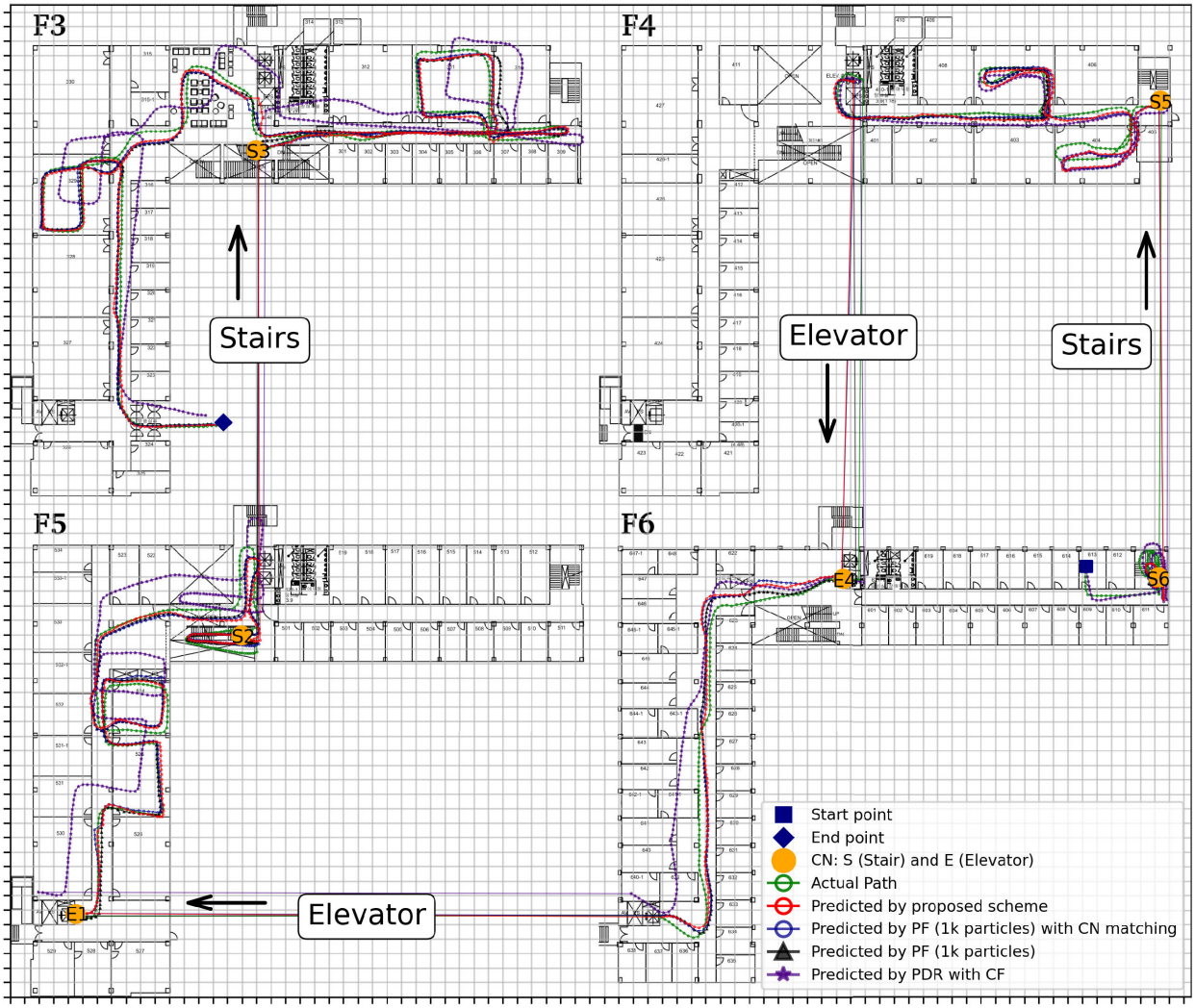



**FIGURE 11.** Results of long path localization experiment.

root-mean-square error (RMSE) was employed to compute the localization loss, as follows.

$$\text{RMSE} = \sqrt{\frac{\sum_{k=1}^{n}(P_k - \hat{P}_k)^2}{n}} = \sqrt{\frac{\sum_{k=1}^{n}(x_k - \hat{x}_k)^2 + (y_k - \hat{y}_k)^2}{n}}. \quad (25)$$

We collected 1,000 step data for the experiment. To demonstrate the performance of our scheme, we performed six approaches to calculate the location, which are: (a) the proposed scheme, (b) PF (1k particles) with CN matching, (c) PF (1k particles), (d) PDR with CF, (e) calibrated PDR with CF, and (f) PDR with Acc & Mag. The setup of each approach is described as follows.

(a) The PF with CN matching-based location correction and dynamic adjustment of particle numbers via (20).
(b) The PF with CN matching-based location correction uses a fixed number of 1,000 (1k) particles generated in the resampling phase, instead of using (20).
(c) Conventional PF generates a fixed number of 1k particles in the resampling phase. When a floor transition occurs, the particle information from the previous region is used directly.
(d) Step locations are calculated with (11). The step length is calculated by (14) and (15), and the heading direction is calculated using (16), where $\gamma = 0.99$.
(e) Step locations are calculated in the same way as (d). In addition, whenever a floor transition is detected, the location of matched CN is set as the current location to correct the location [33].
(f) Step locations are calculated using (11). The step length is calculated by (14) and (15), and the heading direction is calculated from the accelerometer and magnetometer sensors, i.e., $\alpha^m$.

The vertical movement and altitude information were obtained from the floor detection. Because traditional PDR can not function without an initial state, the start location of (d), (e), and (f) was manually annotated. Furthermore, the probabilistic nature of the PF, slight variations can occur in its

computational results each time it is run. Hence, the scores of (a), (b), and (c) are obtained from the average of ten separate computations.

Figure 11 presents the results of the long path localization. Before going through the analysis of the experimental results, we provided some explanations regarding the visual representation of the trajectory. Because the pedestrian's altitude information is expressed in terms of the floor number, we plotted the estimated step location on the corresponding floor plan to represent the specifics of user activities and trajectory. The start and end points are respectively indicated by a navy square and diamond. The CN is denoted by an orange circle, with S indicating a CN located at the stairs, and E indicating a CN at the elevator. Only CNs identified as main are depicted. Each grid on the map represents a space of 2m. Text blocks indicate the type of floor transition, and arrows signify the direction of these transitions. To facilitate a clear representation, only results from (a), (b), (c), and (d) were illustrated in Figure 11. Moreover, since the estimated locations before particle convergence are meaningless and could potentially hinder visual comprehension, the results for PF methods are plotted after convergence is confirmed.

In Figure 11, we consider a realistic trajectory consisting of a sequence of movements through various complex sections within the building, as follows.

- F6 activities: Start from room 613 on F6 ≫ exit the room and go downstairs through the right stairwells. We matched three CNs during the floor transition. Since convergence had not yet been reached, $N_p$ particles are generated around each CN.
- F4 activities: Arrive at F4 and exit the stairwells ≫ enter rooms 404 and 407 consecutively ≫ exit the rooms and move along the corridor ≫ back to F6 through an elevator. In this segment, S5 was matched as the main CN.
- F6 activities: Arrive at F6 and exit the elevator ≫ move along the corridor ≫ enter the elevator to go downstairs. In this segment, E4 was matched as the main CN.
- F5 activities: Arrive at F5 and exit the elevator ≫ move along the corridor ≫ enter rooms 525 and 524 consecutively ≫ move to the elevators near the open corridor ≫ go downstairs through the staircase. In this segment, E1 and S2 were matched as the main CNs.
- F3 activities: Arrive at F3 and exit the staircase ≫ move along the right side corridor ≫ enter room 311 and exit the room ≫ move to the open study space along the corridor ≫ enter room 329 and exit the room ≫ move to the exit along the corridor ≫ detected as leaving the building at the exit. In this segment, S3 was matched as the main CN.

Note that the user's walking trajectory may vary between rooms due to the presence of obstacles (e.g., tables). The first floor transition occurred at the 32nd step, and convergence was confirmed at the 42nd step for the PF with CN matching and at the 150th step for the PF without CN matching.

**TABLE 9. Evaluation results for each approaches.**

| | AR $\epsilon$=0.5m | AR $\epsilon$=1.0m | AR $\epsilon$=1.5m | AR $\epsilon$=2.0m | RMSE [m] |
|---|---|---|---|---|---|
| Proposed scheme | 17.6% | 58.9% | 86.6% | 96.7% | 1.1 |
| PF (1k particles) with CN matching | 16.7% | 61.5% | 87.1% | 97.8% | 1.1 |
| PF (1k particles) | 16.6% | 46.1% | 68.8% | 86.9% | 1.2 |
| PDR with CF | 3.2% | 8.5% | 19.7% | 27.3% | 3.8 |
| Calibrated PDR with CF | 11.3% | 24.1% | 31.6% | 41.2% | 3.8 |
| PDR with Acc & Mag | 0.7% | 2.0% | 3.9% | 8.1% | 11.2 |

From Figure 11, PDR with CF demonstrates good performance in the early stages of tracking with a given start point, but it increasingly deviates from the actual path as the tracking progresses. PF (1k particles) offered an enhanced result by eliminating cumulative errors in long path tracking through boundary constraints. However, several challenges were encountered during its implementation. 1. Prone to failure: Out of the ten PF experiments conducted, six resulted in localization failure due to particle convergence to incorrect locations. 2. Slow convergence: In the remaining four successful instances, convergence to the correct location was achieved after approximately 150 steps. 3. Inaccurate localization: When entering the first room on F3 (i.e., room 311), there was an incorrect entry into the adjacent room (i.e., room 310). These issues render the conventional PF approach inaccurate or impractical in real applications. Fortunately, they are overcome by our solutions. All ten instances of PF (1k particles) with CN matching successfully located the user. Particularly, there were two instances where convergence to incorrect locations was observed, and both were rectified using our CN matching-based location correction. Additionally, it detected all rooms correctly. Moreover, the proposed scheme demonstrated performance comparable to that of PF (1k particles) with CN matching, i.e., achieving successful tracking throughout, convergence at approximately the 42nd step, and correct detection of all rooms.

Table 9 shows the AR and RMSE values for approaches. Regarding the details of the calculation, since we conducted multiple computations for PF approaches, values in Table 9 were obtained from the average of separate computations. Furthermore, because the convergence speed of PF depends on factors such as activity and surrounding environment, it is unstable and not related to the accuracy of the estimated location. Therefore, the AR score and RMSE loss of PF approaches were calculated when the convergence is confirmed, i.e., at the 42nd step. This implies that slower convergence in the case of PF without CN matching would yield a lower AR score. Furthermore, because the estimated locations before convergence are random and would distort

the informative value of the RMSE loss, the RMSE loss for PF without CN matching is calculated from the 150th step.

In Table 9, PDR with Acc & Mag achieved an AR value of 8.1% within the error boundary $\epsilon = 2.0$m, which demonstrates the inaccuracies introduced when utilizing consumer-grade processors in the conventional PDR method. PDR with CF and its calibrated version provided improved results by fusing the orientations, but the performance remained insufficient due to the inherent issue of path deviation when IMU data is used over long trajectories. On the other hand, the PF methods show a better performance. Among them, PF (1k particles) without CN matching obtained low AR scores due to slow convergence. By comparing the RMSE values, it can be observed that even excluding the factor of convergence speed, the PF without CN matching still underperforms compared to the PF with CN matching. Moreover, the performance of the proposed scheme is similar to that of PF (1k particles) with CN matching, which obtained an AR score of 96.7% within the error boundary $\epsilon = 2.0$m. Furthermore, the average number of particles to be calculated in each iteration $N_{average}$ can be computed as

$$N_{average} = \frac{\sum_{i=42}^{1000} N_{p,i}}{n_{conv}}, \tag{26}$$

where $N_{p,i}$ is the particle number of the $i$-th step and $n_{conv}$ is the number of steps after convergence. By computing (26), we obtain a $N_{average}$ of 198.7. This means that by dynamically adjusting the number of particles, our scheme achieved performance comparable to the PF using 1k particles with less than 1/5 of the particle number. Overall, the experimental results demonstrate the benefits of our proposed floor detection method for 2D localization systems in multi-floor scenarios and show the superiority of the proposed scheme in terms of both performance and computational efficiency.

## V. DISCUSSION

In this paper, we contend that the merit of a localization system should not be solely evaluated based on location accuracy, and a robust system should exhibit long-term stability and the capability to efficiently process and operate on a limited amount of measurement data.

The proposed DL-based floor detection not only tracks the floor level but also extracts the vertical movement information of a step. The floor level can be used to extend a 2D localization to the 3D application, and the vertical movement features are particularly useful for probabilistic methods such as the PF. Furthermore, the proposed CN matching-based location correction also holds value within some infrastructure-dependent systems. For instance, CN is well-suited to serve as a substitute for anchor nodes in areas such as stairwells that lack adequate signal coverage to optimize localization performance and reduce costs.

## VI. CONCLUSION

We propose an infrastructure-free indoor multi-floor localization scheme that leverages only a smartphone's IMU and barometer sensors. Our scheme consists of two components: DL-based floor detection and PF with clustering. Our scheme is designed to facilitate indoor localization without relying on the infrastructure and calculate the user's location without a given initial state. To provide height information when the user starts localization, our floor detection must track the user's floor level with unrestricted user activity. Additionally, the proposed floor detection should be able to detect floor transitions quickly and stably in order to calibrate the 2D localization. Therefore, we present a Seq2Seq model to predict in real-time the step action from the noisy barometer data. Subsequently, we developed a floor decision algorithm to update the floor level based on the Seq2Seq model's prediction and extract vertical movement features which are used to improve the 2D localization. Furthermore, we applied a PF assisted by map constraints. Directly applying the PF in a multi-floor scenario can cause severe errors, thus we combined it with our floor detection and CNs established around vertical transportation to extend the PF to 3D scenarios. In addition, we introduced a clustering algorithm based on the mean shift method to improve the PF estimation and reduce the computational cost. Finally, the floor number and 2D location are combined to form the user's 3D location in multi-floor buildings. We conducted multiple experiments in typical university buildings to evaluate the proposed floor detection and indoor multi-floor localization. The experimental results show the promising performance of our scheme. The DL-based floor detection accurately tracked the floor number and efficiently extracted vertical movement information under a variety of user activities. The indoor multi-floor long path localization scheme achieved an average localization accuracy of over 96% within a 2m error boundary with a limited number of particles in the PF.

## VII. FUTURE WORKS

Although the proposed scheme works well in typical medium-sized buildings, its performance may be challenged in some large, open spaces (e.g., airports) due to the lack of map constraints. We believe that magnetic field information is a good alternative, which can provide absolute location information and is available everywhere, and some research demonstrates the potential of magnetic measurements as a data source [32], [64], [65]. In addition, in future work, we aim to optimize the design of DL models and floor decision algorithm to extract more information on vertical movements. Finally, we will accommodate various ways of carrying smartphones in 2D localization to make our scheme applicable to more scenarios.

## 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. 4554–4573, Oct. 2019.

[2] Z. Chen, Q. Zhu, and Y. C. Soh, "Smartphone inertial sensor-based indoor localization and tracking with iBeacon corrections," *IEEE Trans. Ind. Informat.*, vol. 12, no. 4, pp. 1540–1549, Aug. 2016.

[3] T. L. N. Nguyen, T. D. Vy, K. Kim, C. Lin, and Y. Shin, "Smartphone-based indoor tracking in multiple-floor scenarios," *IEEE Access*, vol. 9, pp. 141048–141063, 2021.

[4] T. Brovko, A. Chugunov, A. Malyshev, I. Korogodin, N. Petukhov, and O. Glukhov, "Complex Kalman filter algorithm for smartphone-based indoor UWB/INS navigation systems," in *Proc. Ural Symp. Biomed. Eng., Radioelectron. Inf. Technol. (USBEREIT)*, May 2021, pp. 0280–0284.

[5] P. Tan, T. H. Tsinakwadi, Z. Xu, and H. Xu, "Sing-Ant: RFID indoor positioning system using single antenna with multiple beams based on LANDMARC algorithm," *Appl. Sci.*, vol. 12, no. 13, pp. 6751–6766, Jul. 2022.

[6] 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, Jun. 2008.

[7] L. Xie, J. Tian, G. Ding, and Q. Zhao, "Holding-manner-free heading change estimation for smartphone-based indoor positioning," in *Proc. IEEE 86th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2017, pp. 1–5.

[8] 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. 7th Workshop Positioning, Navigat. Commun.*, Mar. 2010, pp. 135–143.

[9] C. Wang, H. Liang, X. Geng, and M. Zhu, "Multi-sensor fusion method using Kalman filter to improve localization accuracy based on Android smart phone," in *Proc. IEEE Int. Conf. Veh. Electron. Saf.*, Dec. 2014, pp. 180–184.

[10] C. Jiawei, Z. Wenchao, W. Dongyan, and S. Xiaofeng, "Research on indoor constraint location method of mobile phone aided by magnetic features," in *Proc. IEEE 12th Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2022, pp. 1–7.

[11] J. Racko, P. Brida, A. Perttula, J. Parviainen, and J. Collin, "Pedestrian dead reckoning with particle filter for handheld smartphone," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Oct. 2016, pp. 1–7.

[12] T. Fetzer, F. Ebner, M. Bullmann, F. Deinzer, and M. Grzegorzek, "Smartphone-based indoor localization within a 13th century historic building," *Sensors*, vol. 18, no. 12, pp. 4095–4127, Nov. 2018.

[13] G. Pipelidis, N. Tsiamitros, C. Gentner, D. B. Ahmed, and C. Prehofer, "A novel lightweight particle filter for indoor localization," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2019, pp. 1–8.

[14] C. De Cock, W. Joseph, L. Martens, J. Trogh, and D. Plets, "Multi-floor indoor pedestrian dead reckoning with a backtracking particle filter and Viterbi-based floor number detection," *Sensors*, vol. 21, no. 13, pp. 4565–4594, Jul. 2021.

[15] X. Wang, T. Li, S. Sun, and J. Corchado, "A survey of recent advances in particle filters and remaining challenges for multitarget tracking," *Sensors*, vol. 17, no. 12, pp. 2707–2728, Nov. 2017.

[16] J. Qian, L. Pei, J. Ma, R. Ying, and P. Liu, "Vector graph assisted pedestrian dead reckoning using an unconstrained smartphone," *Sensors*, vol. 15, no. 3, pp. 5032–5057, Mar. 2015.

[17] Y. Wu, H.-B. Zhu, Q.-X. Du, and S.-M. Tang, "A survey of the research status of pedestrian dead reckoning systems based on inertial sensors," *Int. J. Autom. Comput.*, vol. 16, no. 1, pp. 65–83, Feb. 2019.

[18] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter, Particle Filters for Tracking Applications*. Norwood, MA, USA: Artech House, 2003, ch. 3.

[19] W. Jaworski, P. Wilk, P. Zborowski, W. Chmielowiec, A. Y. Lee, and A. Kumar, "Real-time 3D indoor localization," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2017, pp. 1–8.

[20] S. Boim, G. Even-Tzur, and I. Klein, "Height difference determination using smartphones based accelerometers," *IEEE Sensors J.*, vol. 22, no. 6, pp. 4908–4915, Mar. 2022.

[21] H. Ye, T. Gu, X. Zhu, J. Xu, X. Tao, J. Lu, and N. Jin, "FTrack: infrastructure-free floor localization via mobile phone sensing," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun.*, Mar. 2012, pp. 2–10.

[22] K. Itzik and L. Yaakov, "Step-length estimation during movement on stairs," in *Proc. 27th Medit. Conf. Control Autom. (MED)*, Jul. 2019, pp. 518–523.

[23] H. Ye, T. Gu, X. Tao, and J. Lu, "B-loc: Scalable floor localization using barometer on smartphone," in *Proc. IEEE 11th Int. Conf. Mobile Ad Hoc Sensor Syst.*, Oct. 2014, pp. 127–135.

[24] C. Yi, W. Choi, Y. Jeon, and L. Liu, "Pressure-pair-based floor localization system using barometric sensors on smartphones," *Sensors*, vol. 19, no. 16, pp. 3622–3640, Aug. 2019.

[25] R. Ichikari, L. C. M. Ruiz, M. Kourogi, T. Kurata, T. Kitagawa, and S. Yoshii, "Indoor floor-level detection by collectively decomposing factors of atmospheric pressure," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Oct. 2015, pp. 1–11.

[26] H.-B. Ye, T. Gu, X.-P. Tao, and J. Lv, "Infrastructure-free floor localization through crowdsourcing," *J. Comput. Sci. Technol.*, vol. 30, no. 6, pp. 1249–1273, Nov. 2015.

[27] Q. Wang, M. Fu, J. Wang, H. Luo, L. Sun, Z. Ma, W. Li, C. Zhang, R. Huang, X. Li, Z. Jiang, Y. Huang, and M. Xia, "Recent advances in floor positioning based on smartphone," *Measurement*, vol. 214, pp. 112813–112836, Apr. 2023.

[28] T. Willemsen, F. Keller, and H. Sternberg, "Concept for building a MEMS based indoor localization system," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Oct. 2014, pp. 1–10.

[29] H. Nurminen, A. Ristimäki, S. Ali-Löytty, and R. Piché, "Particle filter and smoother for indoor localization," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat.*, Oct. 2013, pp. 1–10.

[30] B. Gao, F. Yang, N. Cui, K. Xiong, Y. Lu, and Y. Wang, "A federated learning framework for fingerprinting-based indoor localization in multi-building and multifloor environments," *IEEE Internet Things J.*, vol. 10, no. 3, pp. 2615–2629, Feb. 2023.

[31] Nader. G. Rihan, M. Abdelaziz, and S. S. Soliman, "A hybrid deep-learning/fingerprinting for indoor positioning based on IEEE P802.11az," in *Proc. 5th Int. Conf. Commun., Signal Process., Their Appl. (ICCSPA)*, Dec. 2022, pp. 1–6.

[32] M. Abid, P. Compagnon, and G. Lefebvre, "Improved CNN-based magnetic indoor positioning system using attention mechanism," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Nov. 2021, pp. 1–8.

[33] C. Lin and Y. Shin, "Deep learning-based multifloor indoor tracking scheme using smartphone sensors," *IEEE Access*, vol. 10, pp. 63049–63062, 2022.

[34] Y. Kim, S. Lee, S. Lee, and H. Cha, "A GPS sensing strategy for accurate and energy-efficient outdoor-to-indoor handover in seamless localization systems," *Mobile Inf. Syst.*, vol. 8, no. 4, pp. 315–332, 2012.

[35] M. Yu, F. Xue, C. Ruan, and H. Guo, "Floor positioning method indoors with smartphone's barometer," *Geo-Spatial Inf. Sci.*, vol. 22, no. 2, pp. 138–148, Apr. 2019.

[36] L. Wang, "A robust context-based heading estimation algorithm for pedestrian using a smartphone," in *Proc. 28th Int. Tech. Meeting Satell. Division Inst. Navigat. (ION GNSS)*, Tampa, FL, USA, Sep. 2015, pp. 2493–2500.

[37] G. Milette and A. Stroud, Eds., *Prof. Android Sensor Program.*. Hoboken, NJ, USA: Wiley, 2012, ch. 6.

[38] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," 2014, *arXiv:1409.3215*.

[39] M. Tanigawa, H. Luinge, L. Schipper, and P. Slycke, "Drift-free dynamic height sensor using MEMS IMU aided by MEMS pressure sensor," in *Proc. 5th Workshop Positioning, Navigat. Commun.*, Mar. 2008, pp. 191–196.

[40] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *J. Mach. Learn. Res.*, vol. 9, pp. 249–256, May 2010.

[41] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, Jun. 2014.

[43] K. Muralidharan, A. J. Khan, A. Misra, R. K. Balan, and S. Agarwal, "Barometric phone sensors: More hype than hope!" in *Proc. 15th Workshop Mobile Comput. Syst. Appl.*, Feb. 2014, p. 12.

[44] Android Developer. *SensorEvent*. Accessed: May 12, 2023. [Online]. Available: https://developer.android.com/reference/android/hardware/SensorEvent

[45] Apple Developer. *CoreMotion*. Accessed: May 12, 2023. [Online]. Available: https://developer.apple.com/documentation/coremotion

[46] W. Kang and Y. Han, "SmartPDR: smartphone-based pedestrian dead reckoning for indoor localization," *IEEE Sensors J.*, vol. 15, no. 5, pp. 2906–2916, May 2015.

[47] R. Valenti, I. Dryanovski, and J. Xiao, "Keeping a good attitude: A quaternion-based orientation filter for IMUs and MARGs," *Sensors*, vol. 15, no. 8, pp. 19302–19330, Aug. 2015.

[48] J. H. Lee, B. Shin, S. L. J. H. Kim, C. Kim, T. Lee, and J. Park, "Motion based adaptive step length estimation using smartphone," in *Proc. 18th IEEE Int. Symp. Consum. Electron. (ISCE)*, Jun. 2014, pp. 1–2.

[49] S. H. Shin, C. G. Park, J. W. Kim, H. S. Hong, and J. M. Lee, "Adaptive step length estimation algorithm using low-cost MEMS inertial sensors," in *Proc. IEEE Sensors Appl. Symp.*, Feb. 2007, pp. 1–5.
[50] A. Abadleh, E. Al-Hawari, E. Alkafaween, and H. Al-Sawalqah, "Step detection algorithm for accurate distance estimation using dynamic step length," in *Proc. 18th IEEE Int. Conf. Mobile Data Manage. (MDM)*, May 2017, pp. 324–327.
[51] H. Weinberg, "Using the ADXL202 in pedometer and personal navigation applications," Analog Devices, Wilmington, MA, USA, Tech. Rep. AN-602, 2002. [Online]. Available: https://www.analog.com/media/en/technical-documentation/application-notes/513772624AN602.pdf
[52] Apple Developer. *CoreLocation*. Accessed: May 12, 2023. [Online]. Available: https://developer.apple.com/documentation/corelocation
[53] A. Poulose, B. Senouci, and D. S. Han, "Performance analysis of sensor fusion techniques for heading estimation using smartphone sensors," *IEEE Sensors J.*, vol. 19, no. 24, pp. 12369–12380, Dec. 2019.
[54] S. O. H. Madgwick, A. J. L. Harrison, and R. Vaidyanathan, "Estimation of IMU and MARG orientation using a gradient descent algorithm," in *Proc. IEEE Int. Conf. Rehabil. Robot.*, Jun. 2011, pp. 1–7.
[55] G. Kitagawa, "A Monte Carlo filtering and smoothing method for non-Gaussian nonlinear state space models," in *Proc. U.S.-Japan Joint Seminar Stat. Time Ser. Anal.*, vol. 2, Jan. 1993, pp. 110–131.
[56] A. Doucet and A. M. Johansen, "A tutorial on particle filtering and smoothing: Fifteen years later," in *The Oxford Handbook of Nonlinear Filtering*, D. Crisan and B. Rozovskii, Eds. London, U.K.: Oxford Univ. Press, 2009, pp. 656–704.
[57] D. A. Medina, M. Schwaab, D. Plaia, M. Romanovas, S. Meckler, M. Traechtler, and Y. Manoli, "A foot-mounted pedestrian localization system with map motion constraints," in *Proc. 12th Workshop Positioning, Navigat. Commun.*, Dresden, Germany, Jan. 2015, pp. 1–6.
[58] A. Perttula, H. Leppäkoski, M. Kirkko-Jaakkola, P. Davidson, J. Collin, and J. Takala, "Distributed indoor positioning system with inertial measurements and map matching," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 11, pp. 2682–2695, Nov. 2014.
[59] H. Wang, H. Lenz, A. Szabo, J. Bamberger, and U. D. Hanebeck, "WLAN-based pedestrian tracking using particle filters and low-cost MEMS sensors," in *Proc. 4th Workshop Positioning, Navigat. Commun.*, Mar. 2007, pp. 1–7.
[60] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.
[61] J. Elfring, E. Torta, and R. van de Molengraft, "Particle filters: A hands-on tutorial," *Sensors*, vol. 21, no. 2, pp. 438–465, Jan. 2021.
[62] M. Klepal and S. Beauregard, "A backtracking particle filter for fusing building plans with PDR displacement estimates," in *Proc. 5th Workshop Positioning, Navigat. Commun.*, Mar. 2008, pp. 207–212.
[63] T. D. Vy, T. L. N. Nguyen, and Y. Shin, "A precise tracking algorithm using PDR and Wi-Fi/iBeacon corrections for smartphones," *IEEE Access*, vol. 9, pp. 49522–49536, 2021.
[64] M. Frassl, M. Angermann, M. Lichtenstern, P. Robertson, B. J. Julian, and M. Doniec, "Magnetic maps of indoor environments for precise localization of legged and non-legged locomotion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 913–920.
[65] L. Antsfeld and B. Chidlovskii, "Magnetic field sensing for pedestrian and robot indoor positioning," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Nov. 2021, pp. 1–8.

**CHENXIANG LIN** received the B.S. degree in computer science and engineering from Korea University, South Korea, in 2021. He is currently pursuing the M.S. degree in information and telecommunication engineering with Soongsil University, South Korea. His research interests include indoor localization, sensor fusion, and deep learning.

**YOAN SHIN** (Senior Member, IEEE) received the B.S. and M.S. degrees in electronics engineering from Seoul National University, South Korea, in 1987 and 1989, respectively, and the Ph.D. degree in electrical and computer engineering from The University of Texas at Austin, USA, in 1992. From 1992 to 1994, he was a member of the Technical Staff with Microelectronics and Computer Technology Corporation (MCC), Austin, TX, USA. Since 1994, he has been with the School of Electronic Engineering, Soongsil University, South Korea, where he is currently a Professor. From 2009 to 2010, he was a Visiting Professor with the Department of Electrical and Computer Engineering, The University of British Columbia, Canada. He was the Dean of the School of Electronic Engineering, the Vice Dean of the College of IT, the Dean of Office of Research, and the Dean of Office of Planning. He is currently the Provost and the Senior Vice President for Academic Affairs of Soongsil University. His research interests include mobile communications and intelligent signal processing. 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 was the President of the Korean Institute of Communications and Information Sciences (KICS), in 2022.