GPS-assisted Indoor Pedestrian Dead Reckoning

HENG ZHOU, Osaka University, Japan
TAKUYA MAEKAWA *, Osaka University, Japan

Indoor pedestrian dead reckoning (PDR) using embedded inertial sensors in smartphones has been actively studied in the ubicomp community. However, PDR relying only on inertial sensors suffers from the accumulation of errors from the sensors. Researchers have employed various indoor landmarks detectable by smartphone sensors such as magnetic fingerprints caused by elevators and Bluetooth signals from beacons with known coordinates to compensate for the errors. This study proposes a new type of indoor landmark that does not require additional device installation, e.g., beacons, and training data collection in a target environment, e.g., magnetic fingerprints, unlike existing landmarks. This study proposes the use of GPS signals received by a smartphone to correct the accumulated errors of the PDR. While it is impossible to locate the smartphone indoors using GPS satellites, the smartphone can receive signals at a window-side area through windows from satellites aligned with the orientation of the window normal. Based on this idea, we design a machine-learning-based module for detecting the proximity of a user to a window and the orientation of the window, which enables us to roughly determine the absolute coordinates of the smartphone and to correct the accumulated errors by referring to positions of window-side areas found in the floor plan of the environment. A key technical contribution of this study is designing the module, such that it can be trained based on data from environments other than the target environment yet work in any environment by extracting GPS-related information independent of wall orientation. We evaluated the effectiveness of the proposed method using sensor data collected in real environments.

CCS Concepts: • **Human-centered computing—Ubiquitous and mobile computing systems and tools**;

Additional Key Words and Phrases: Indoor localization system, pedestrian dead reckoning, GPS satellite information

ACM Reference Format:
Heng Zhou and Takuya Maekawa . 2022. GPS-assisted Indoor Pedestrian Dead Reckoning. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 4, Article 166 (December 2022), 36 pages. https://doi.org/10.1145/3569467

166

1 INTRODUCTION

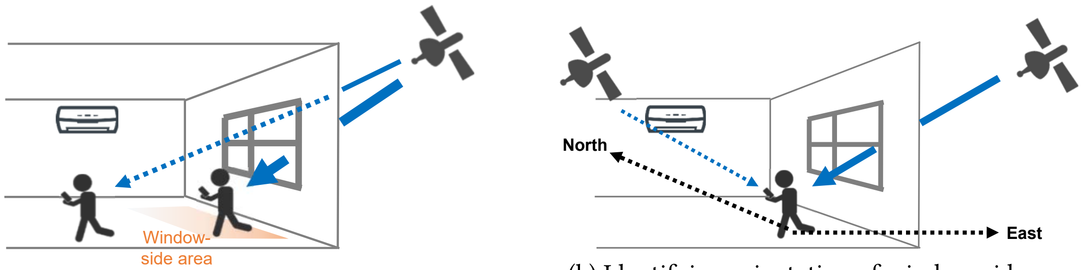
1.1 Background and Problems

To achieve accurate positioning in indoor environments where GPS is not available, ubiquitous and mobile computing researchers have focused on indoor pedestrian dead reckoning (PDR) using embedded inertial sensors in smartphones because PDR has various practical applications, such as indoor navigation, augmented reality, and healthcare monitoring [21]. However, PDR relying only on inertial measurement units (IMUs) has two main drawbacks: i) accumulated errors of the estimated walking trajectory occur owing to the drift of the gyroscope

*This is the corresponding author

Authors' addresses: Heng Zhou, Osaka University, Graduate School of Information Science and Technology, Suita, Osaka, 5650871, Japan; Takuya Maekawa, Osaka University, Graduate School of Information Science and Technology, Suita, Osaka, 5650871, Japan, takuya.maekawa@acm.org,maekawa@ist.osaka-u.ac.jp.

2474-9567/2022/12-ART166 $15.00
https://doi.org/10.1145/3569467



(a) Detecting window-side area

(b) Identifying orientation of window-side area

Fig. 1. Basic idea of using GPS signal information to detect indoor landmarks. (a) When a smartphone is close to a window, it receives strong GPS signals, enabling us to know whether the smartphone is at a window-side area. (b) When a smartphone is close to a window facing the east, it receives strong GPS signals of a satellite from the east, enabling us to know the orientation/direction that the window faces.

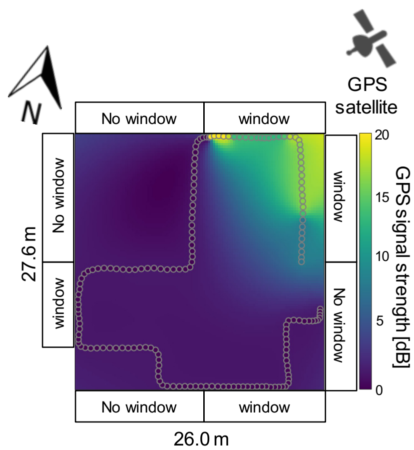


Fig. 2. Heat map in a floor generated using signals from a GPS satellite. Each gray circle indicates an observation point. The azimuth of the satellite was approximately 40 degrees (relative to the north) during the data collection.

and errors in stride prediction, and ii) as the absolute coordinates cannot be estimated by the inertial sensors, information regarding the initial position (and initial moving direction) of the trajectory is required.

To address the first problem, researchers have used indoor "landmarks" that can be detected by various sensors in smartphones to correct the accumulated errors [1]. For example, some researchers used the Bluetooth module in a smartphone to detect nearby Bluetooth Low Energy (BLE) beacons installed in the environment [37, 42]. When the Bluetooth module detects a BLE beacon placed at a certain position, the smartphone's position estimate is corrected with respect to the position of the beacon. In addition, several researchers used magnetic sensors in smartphones to detect the magnetic field emitted from some infrastructure, such as elevators, and set them as landmarks [23, 29]. Furthermore, location-specific activities and actions, such as sleeping in the bedroom [18] or climbing the staircase [1, 15], can be tracked by smartphone sensors and used as landmarks. These methods, when used in conjunction with map matching, enable us to achieve PDR without any information regarding the initial position [3, 41]. Because the above landmark-based methods opportunistically correct the accumulated errors, e.g., errors can be corrected only when a user approaches an elevator, which emits a strong magnetic field, it is important to increase the density of landmarks by fully leveraging the diverse sensors in a smartphone. To efficiently add landmarks in an environment, the cost of installation of extra devices, such as BLE beacons, and the cost of training data collection for each environment to detect landmarks, such as magnetic fingerprints and human activities, need to be considered.

1.2 Approach

In this study, we propose to use landmarks that do not require installation of new devices or pre-acquisition of training data in a target environment. Specifically, this paper proposes to use GPS signals to correct accumulated errors of PDR. GPS satellites are existing infrastructure and do not require extra installation cost. However, as GPS satellites cannot be used to measure the coordinates indoors, they cannot be used to correct the PDR accumulated errors. In this research, we propose to focus on the fact that, at window-side areas inside buildings, we can receive GPS signals from several GPS satellites, although these GPS satellites cannot provide location measurements. Compared to areas far from the windows, the smartphone receives stronger GPS signals at areas near the windows as shown in Fig. 1 (a); therefore, it is possible to estimate the timing when the smartphone is at close vicinity to the window. Fig. 2 shows a heat map in a floor generated from signals from a GPS satellite located at the upper right direction of this figure. The signals were measured by a smartphone during walking. As shown in the figure, when the smartphone was close to window-side areas facing the satellite orientation, it received strong signals from the satellite.

Moreover, the signal from a GPS satellite contains information about the azimuth of the satellite, enabling us to estimate which direction the window is facing. For example, when the smartphone is at the window-side of an outer wall facing east in a building, the signal strength from GPS satellites in the eastern direction will be stronger, as shown in Fig. 1 (b). Therefore, when the signal strength from GPS satellites located at the eastern direction is strong, it can be estimated that the smartphone is located at the window-side of the eastern wall. As a result, we regard these window-side areas detected by GPS satellite information as landmarks to correct PDR accumulated errors. However, we cannot uniquely identify a single landmark, i.e., window-side area, based on GPS information because the target building usually has multiple window-side areas close to walls facing the same orientation. To deal with this problem, we leverage map matching using the floor plan including information about obstacles, i.e., inner walls, of the target environment to eliminate the ambiguity.

In addition, in this study, we explore PDR that does not require initial positions as inputs by leveraging GPS signals. As mentioned above, GPS signals can be used to identify the time t and the orientation o of a window-side area the smartphone is in. In other words, we can generate some candidates of the absolute coordinates of the smartphone at time t within window-side areas facing said orientation o. Then, from the absolute coordinates at time t, the absolute coordinates at time 0 (i.e., initial position) can be estimated by back-tracking the movement using inertial sensors with map matching, as shown in Fig. 3. As shown above, we can implement PDR, which does not require known initial positions. Note that, as mentioned above, when there are multiple window-side areas close to walls facing the same orientation, there exist multiple position candidates at time t. Because the walking direction relative to the window normal at time t is also unknown, it is necessary to generate a candidate moving toward each direction (from 0 to 360 degrees) and then back/forward-track each candidate, which is computationally expensive. Although it is possible to measure the walking orientation at time t through the digital compass in the smartphone, its accuracy is low indoors owing to magnetic interference. Therefore, in this study, a digital compass is used together with GPS signal information to roughly estimate the walking direction. The walking direction is estimated based on the idea that the GPS signal strength gradually increases as the user walks toward a window.

The proposed method is composed of a window-side detector and a window orientation classifier for window-side landmark detection, and then, by incorporating them into the PDR module, the method corrects PDR accumulated errors. In the window-side detector and window orientation classifier, we use time-series GPS signals to estimate the proximity to window-side areas and the orientation of the window normal of a window-side area at each time step. In addition, a walking direction predictor in the proposed method estimates a walking direction relative to the window normal by using GPS signals. Note that, to reduce installation costs of indoor landmarks, this study assumes that training data for detecting window-side areas, window orientation prediction, and walking direction prediction are collected in environments other than the target environment. However, the shapes and orientations of outer walls of different buildings are also different, and positions of GPS satellites change with time (training data collection time vs. test data collection time), making it difficult to detect GPS landmarks independent of environments, which is the technical challenge of this study. To deal with these issues, we develop modules for window-side detection, window orientation prediction, and walking direction prediction that can be used in an environment with any outer wall orientations by extracting GPS-related information independent of the wall orientation and positions of GPS satellites.

1.3 Contributions

Our study provides three main research contributions.

- To the best of our knowledge, this study is the first work that corrects accumulated errors of indoor PDR by using window-side GPS landmarks that do not require additional infrastructure and training data collected in the target environment.
- To achieve GPS-assisted indoor PDR, we design the window-side detector, window orientation classifier, and walking direction predictor, which are incorporated into the PDR method. The technical contributions of this study are i) the designs of these modules that can be trained in environments other than the target environment by extracting environment-independent features of GPS signals, and ii) a novel pipeline of efficiently introducing GPS signal information into inertial sensor data to achieve accurate PDR.

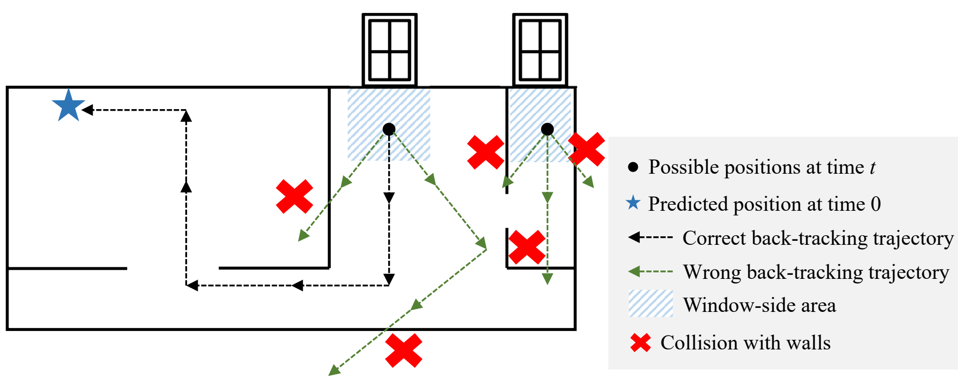


Fig. 3. Back tracking smartphone positions from time t when the smartphone is detected to be near window-side areas facing the upper direction of this figure. Position candidates at time t are generated within multiple window-side areas facing the upper direction because we cannot know which window-side area facing the upper direction the smartphone is in. Then, we back-track the smartphone positions based on inertial sensor data by generating movement paths (trajectories) toward various directions because the movement direction is also unknown. When a movement path collides with a wall, the path is eliminated as a wrong trajectory. Forward-tracking is also conducted in a similar way.

- Extensive experiments on different trajectories collected in real environments showed the feasibility of GPS landmarks.

2 RELATED WORK

2.1 Pedestrian Dead Reckoning (PDR)

2.1.1 Conventional PDR. PDR systems have been developed as a solution to navigation in indoor environments or in areas with weak and unstable GPS signals. The main advantage of PDR relying on inertial sensors is the robustness against environmental differences, as it uses accelerometer readings to detect steps to estimate walking distance and uses gyroscope signals to compute the heading direction [30]. From accelerometer data, the PDR systems detect the user's step. A peak detection algorithm is used that leverages a centered moving average to smoothen the acceleration magnitude and applies windowed peak detection to find peaks related to heel impact [19]. For walking distance estimation, the distance travelled by the user is determined by the cumulative strides. The stride represents the distance travelled by the user within one step. Most step-length estimation methods are based on empirical methods using walk frequency (step frequency) and acceleration variance as main factors [35]. Gyroscope readings are directly used for the heading direction so that angular velocity can be iteratively integrated to generate the relative direction. However, the main problem of PDR is the accuracy of the heading direction part because the drift of the gyroscope will cause errors that accumulate with time. To improve the accuracy of PDR, researchers have focused more on reducing errors of heading direction estimation [2, 28]. Currently, a popular solution, named Zero Velocity Updates (ZUPTs), is generally used that can close the integration loop of angular velocity periodically by applying external constraints to the PDR system to deal with the drift [10, 11]. In such methods, they initially detected the stance phase and applied zero velocity during stance duration. Finally, with a given known starting position, it can integrate each step's displacement with direction change to get a full trajectory [22].

2.1.2 Machine Learning-based PDR. Unlike conventional PDR systems with step displacement segmentation and stride estimation, state-of-the-art neural network-based PDR systems are based on batch processing of a raw inertial data window to estimate the formula of displacement and angular change [4, 40]. Although raw inputs (acceleration and angular velocity) are independent of continuous IMU measurement, they have strong temporal dependencies and represent walking activities. Therefore, a deep recurrent neural network (RNN), especially long-short term memory (LSTM), can capture these dependencies and can take advantage of them to restore potential connections between data features and walking characteristics. To employ neural network-based PDR in low-end devices like smartphones, Chen et al. [5] designed a PDR method based on convolution neural networks. In this study, we leverage state-of-the-art neural-network-based PDR as the basis and integrate our GPS landmark module to correct accumulated errors of the PDR method.

2.2 Landmark-assisted PDR

As PDR systems relying only on inertial sensors suffer from accumulated errors, mobile and ubiquitous computing researchers have developed PDR systems that correct accumulated errors by leveraging various smartphone sensors to detect indoor landmarks. These systems employ sensors in smartphones to detect some "unusual" positions and mark them as landmarks. When a smartphone user approaches or passes by a landmark, the PDR systems correct the position estimate to the position of the landmark. Currently, there are many methods to detect indoor landmarks. Abdelnasser et al. [1] classified landmarks that can be used in indoor environments into two categories by data features: Seed Landmarks (SLMs) and Organic Landmarks (OLMs). SLMs are positions known priori that exhibit obvious changes in sensors in smartphones, such as elevators and staircases [15, 16], whereas OLMs are positions that can be not physical but have obvious signatures by clustering data from sensors,

Table 1. Comparison of different types of landmarks

Landmark type	Training data collected in advance in target environment	Installation of additional infrastructure	Spatial ambiguity of landmarks	Density of landmarks
Magnetic fingerprinting	Yes	No	Low	High
Visible light communication	No	Yes	Low	Depends on # devices
Wi-Fi fingerprinting	Yes	No	Middle	High
Bluetooth fingerprinting	Yes	Yes	Middle	High
Bluetooth proximity detection	No	Yes	Low	Depends on # devices
Acceleration-based stair detection	No	No	High	Low
Activity recognition	Yes	No	Middle	Middle/High
GPS (proposed approach)	No	No	Middle/High	Middle

such as Wi-Fi or GSM signals [6, 39]. For example, positions without any Wi-Fi and GSM signals in a building can be used as OLMs.

Wireless signalling devices have been widely used as indoor landmarks, e.g., BLE beacons for proximity detection and Wi-Fi APs for creating Wi-Fi fingerprints [7, 44]. Abnormal magnetic fields detected in indoor environments have also been used as indoor landmarks [23, 29]. Lee et al. [29] used a robot to collect magnetic fields in all areas in the target building and then picked areas with abnormal magnetic fields. After labelling them and training a convolutional neural network for detecting the magnetic landmarks, these landmarks could be detected by smartphones. In addition, visible light has been used as indoor landmarks [20, 26]. Sakshi et al. [20] used visible light communication-based indoor positioning where light emitted by dedicated bulbs is used to send position signals for identification. Therefore, a smartphone can know its position after receiving the signals from LED lamps by obtaining information from a database storing LED lamp identifiers and their positions. Moreover, location-specific activities and actions, e.g., ascending/descending a stair, have been used as landmarks [18, 43]. A method called ActionSlam aims to build landmarks by activity recognition to improve PDR [18]. In their work, they used different sensors to record some daily activities, such as eating or opening/closing doors, and then created a ground truth map with action landmarks marked at positions where they expected the corresponding actions to happen. However, many of these methods require substantial costs regarding equipment installation and/or training data collection. We summarize features of these methods and our proposed GPS landmarks in Table 1. While the proposed GPS landmarks have significant advantages on the costs pertaining to device installation and training data collection, the spatial ambiguity of the landmarks is high when a target environment has multiple window-side areas facing the same orientation. Therefore, this study leverages map matching and GPS-based walking direction prediction to reduce the ambiguity. While the acceleration-based landmarks using staircases also do not require device installation and training data collection, the acceleration-based landmarks are sparse in many cases.

Note that we do not believe that error correction using GPS landmarks will replace other types of indoor landmarks. As smartphones have various sensors, landmark-based error correction should be implemented by using various types of landmarks to increase the density of landmarks located in a target environment. We believe that the proposed GPS landmark is one of the landmarks with the lowest installation costs because it does not require extra infrastructure or training data in the target environment.

2.3 Indoor Positioning with GPS

Few studies have investigated indoor positioning using GPS. Kjærgaard et al. [25] investigated the positioning performance of a dedicated GPS receiver and mobile phones in indoor environments. While the positioning error of the dedicated receiver was below 10 meters, the error of GPS receivers embedded in mobile phones was considerably higher. Ochiai et al. [12, 32] achieved indoor positioning using GPS signals based on a fingerprinting approach, which required a site survey to be conducted in the target environment. In contrast, we have developed environment-independent models for window-side detection and wall orientation prediction based on GPS signals. Fujiwara et al. [13] employed GPS signals to locate indoor static air conditioners to provide location-aware air conditioning services. In contrast, this study is the first study that locates moving smartphones indoors by integrating GPS landmarks into PDR systems.

2.4 Multi-modal Indoor Localization

With the development of smartphones and small sensing devices, various sensors can be equipped in a single device. Fusion of the multi-modal sensors has become a trend in the studies on indoor localization. These studies correct the deviations in walking direction by using various sensors such as Wi-Fi modules, magnetometers, and cameras [27, 31, 38]. For example, Venkatnarayan et al. [38] proposed a Wi-Fi based indoor localization method without fingerprints. In their study, they used Wi-Fi signal to calculate the travelled distance by a subject as well as an accelerometer to predict the distance, and then used the Kalman filter to correct the wrong walking direction by fusing the distance estimates. In addition, several studies [8, 36] predict semantics of a user's current location, e.g., living room and kitchen, by using multi-modal sensor data such as acceleration data and sound data. For example, Dissanayake et al. [8] extracted location-specific sensor data motifs from acceleration data based on GINI impurity. Tachikawa et al. [36] used a magnetometer, barometer, and microphone to extract location-specific features, and then estimated room-level location semantics by using modified decision trees. In our study, we incorporate information about GPS satellites into IMU-based PDR.

3 PROPOSED METHOD

3.1 Preliminaries

We assume that a pedestrian holds a smartphone with an accelerometer, gyroscope, digital compass, and GPS module. From these sensors, we obtain time-series sensor data $\mathcal{S} = \{\mathcal{S}^i, \mathcal{S}^c, \mathcal{S}^g\}$, where $\mathcal{S}^i$ is the time-series of 6-axis inertial measurement data composed of accelerometer and gyroscope data, $\mathcal{S}^c$ represents the time-series of orientation (relative to the north), and $\mathcal{S}^g$ represents the time-series of GPS satellite data. In addition, s^i_t is IMU data at time t, s^c_t is orientation data at time t, and s^g_t is GPS information containing the elevation angle, azimuth, and signal strength of visible satellites at time t.

Moreover, we assume that the floor plan information (like Fig. 3) of a target floor is given. The floor plan information contains the outer shape and orientation of each wall. In this study, we define the orientation of a wall as the angle (relative to north) of the line that runs perpendicular to the surface of that wall. The floor plan information also contains positional information of each window, i.e., which wall the window is installed on. The window-side area is defined as an area within d_w m from the window, where d_w refers to the distance from the window. Furthermore, the floor plan information contains positional information about inner walls in the floor, which is used for map matching.

3.2 Overview

3.2.1 Modules. Figure 4 shows an overview of the proposed method, which consists of the following three main modules.

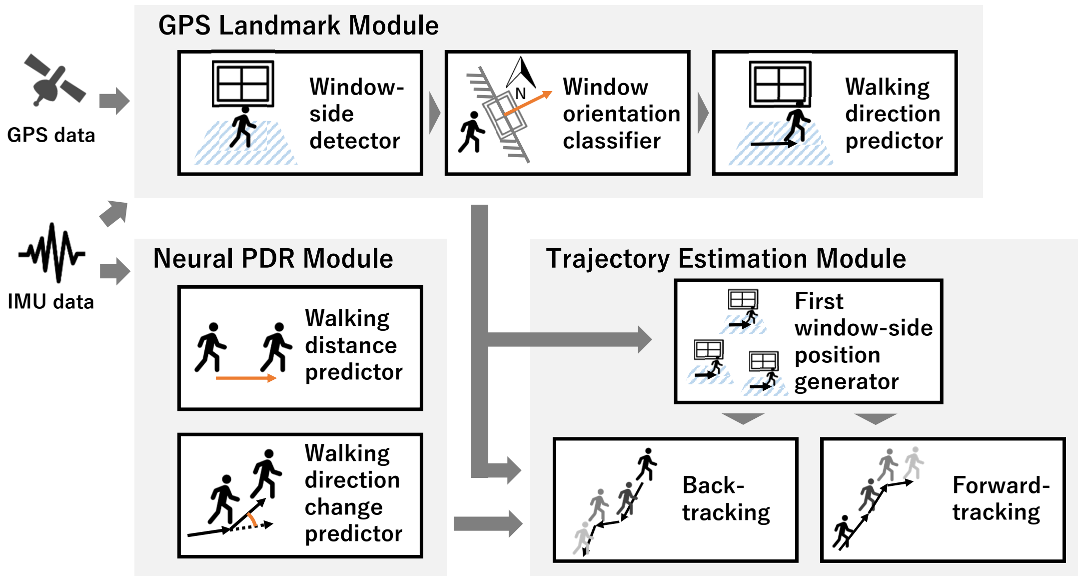


Fig. 4. Overview of the proposed method

- **Neural PDR module**: This is composed of the walking distance predictor and walking direction change predictor, which predict the walking distance and relative change of walking direction within a time window, respectively, by employing accelerometer and gyroscope data.
- **GPS landmark module**: This is used to find GPS landmarks. It is composed of three sub-modules: window-side detector, window orientation classifier, and walking direction predictor. The window-side detector employs GPS information to estimate whether the smartphone user is within a window-side area at each time t. When the user is estimated to be in a window-side area at time t, the window orientation classifier estimates the orientation of that window. The walking direction predictor predicts the walking direction relative to the window normal when the user is in a window-side area, which is used to reduce the computation cost in the trajectory estimation module.
- **Trajectory estimation module**: This is based on a particle filter and fuses outputs of the above modules to predict a walking trajectory. It is composed of three sub-modules: first window-side position generator, forward-tracking, and backward-tracking. When the GPS landmark module first detects the user in a window-side area facing orientation o at time t_i, the first window-side position generator generates random candidate positions in window-side areas facing o with walking direction predicted from the walking direction predictor. Then, forward-tracking is performed by employing results from the neural PDR module and map matching to reproduce the trajectory from time t_i to subsequent times. Note that the outputs of the window-side detector and window orientation classifier at each time step are used to correct accumulated errors of the neural PDR module. Meanwhile, backward-tracking is performed to recover the trajectory before the user first approaches the window-side area from time t_i to time 0.

3.2.2 Procedures. The procedures of these modules are explained as follows. (i) First, we suppose that the user starts walking from an initial position at time 0. At that time, it is impossible to determine the user's trajectory because the absolute initial position is unknown. (ii) Then, we assume that the user passes through a window-side area. If the window-side detector and window orientation classifier detect the user entering a window-side area facing orientation o at time t_i with high confidence, the first window-side position generator of the trajectory estimation module generates candidates of position estimates at time t_i within window-side areas facing the estimated orientation o. At that time, the walking directions of the candidates at time t_i are determined based on an estimate by the walking direction predictor. (iii) Subsequently, on the one hand, based on the output of the neural PDR module, the trajectory estimation module uses map matching to back-track the trajectory from time t_i to time 0. (iv) On the other hand, the trajectory after time t_i is also forward-tracked with the output of the neural PDR module. In the forward- and back-tracking, according to predictions by the window-side detector and the window orientation classifier, the accumulated errors of the trajectory are corrected.

3.3 Neural PDR Module

In this module, we estimate the walking distance and walking direction change in a W-sec time window ($W = 1$ in this study) to reconstruct a trajectory. Fig. 5 shows the relationship between the user coordinates and estimates by this module (i.e., walking distance and walking direction change). For example, the walking distance at time t, i.e., $d_{(t)}$, corresponds to the distance between the position at time t and that at time $t-1$. In addition, the walking direction change at time t, i.e., $\theta_{(t)}$, corresponds to the angle between the two movement vectors: a vector whose source and destination are coordinates at times $t-2$ and $t-1$, respectively, and a vector whose source and destination are coordinates at times $t-1$ and t, respectively. Because it is implemented using state-of-the-art neural network-based methods [4, 9, 40], we explain this module briefly. We will explain the way of acquiring ground truth, i.e., $d_{(t)}$ and $\theta_{(t)}$ in Fig. 5, later. Note that the time unit of t in $d_{(t)}$ is "seconds." Therefore, time $t-1$ in $d_{(t-1)}$ denotes 1 second before time t in $d_{(t)}$.

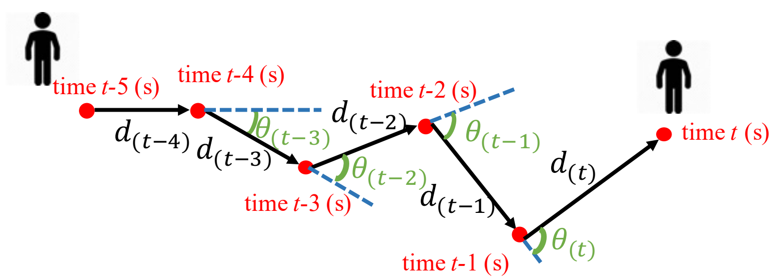


Fig. 5. Displacement d_t and relative walking direction change θ_t at each time step. In our implementation, we predict $d_{(t)}$ and $\theta_{(t)}$ for each second by using sensor data between time $t-1$ (s) and t, and re-construct a movement path by using the predicted values.

3.3.1 Walking Distance Predictor. The inputs for the walking distance predictor is a 1-s window of accelerometer and gyroscope data $\mathbf{s}^i_t = \{s^i_{t-W_i}, ..., s^i_{t-1}, s^i_t\}$, where W_i corresponds to the 1-s window size. The input is fed into a neural network for distance prediction composed of an LSTM layer containing 32 units with the Rectified linear unit (ReLU) activation function, a dropout layer, and an output layer with the linear activation function (Fig. 6). The output of this network is the displacement $d_{(t)}$ between time $t-1$ (s) and time t (s), as shown in Fig 5. That is, when we predict $d_{(t)}$, we employ acceleration data collected between time $t-1$ (s) and time t (s). The model training is done by using the Adam optimizer [24] to minimize the mean squared error (MSE) between the estimate and ground truth.

3.3.2 Walking Direction Change Predictor. The inputs for the walking direction change predictor is a 1-s window of gyroscope data. A neural network for directional change prediction is composed of a bi-directional LSTM layer

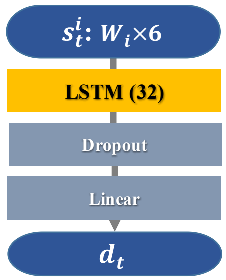


Fig. 6. Architecture of walking distance predictor. s_t^i refers to time-series data of accelerometer and gyroscope

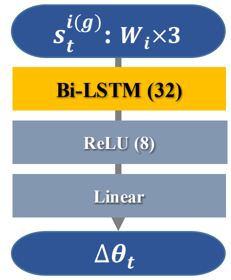


Fig. 7. Architecture of walking direction change predictor. $s_t^{i(g)}$ refers to time-series data of gyroscope

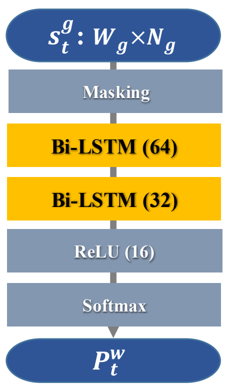


Fig. 8. Architecture of window-side detector. The output P_t^W refers to the probability of the subject in the window-side at time t

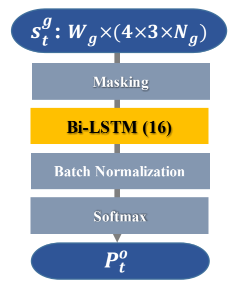


Fig. 9. Architecture of window orientation classifier. The output P_t^o refers to the probability of user in the window-side area whose orientation faces o at time t. Note that the probabilities of all other orientations like $o - 90$ and $o + 90$ are predicted by the same classifier.

containing 32 units with the ReLU activation function, a fully connected layer containing 8 units with the ReLU activation function, and an output layer with the linear activation function (Fig. 7). The output of the walking direction change predictor is the angular change in walking direction $\Delta\theta_{(t)}$ between time $t - 1$ (s) and time t (s), as shown in Fig 5. The model training is also done by using the Adam optimizer to minimize the MSE between the estimate and ground truth.

3.4 GPS Landmark Module

3.4.1 Window-side Detector. The window-side detector estimates whether the user is in a window-side area or not at each time step. In this study, a window-side area refers to an area within d_w m from a window. Fig. 10 shows a time-series of signal strengths of GPS satellites when a smartphone user approached a window-side area from a position away from the window side and then moved away. We can observe that as the user walks closer to the window-side, some GPS signals become stronger. Based on this observation, we use the time-series of signal strengths from top-N_G satellites in terms of their signal strengths as inputs of the window-side detector, where N_G refers to a hyperparameter regarding the number of satellites.

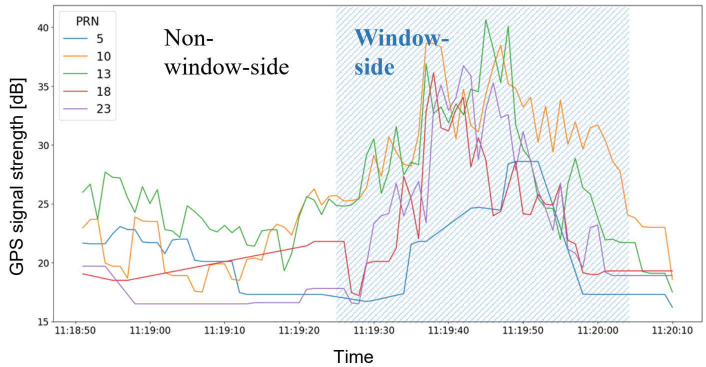


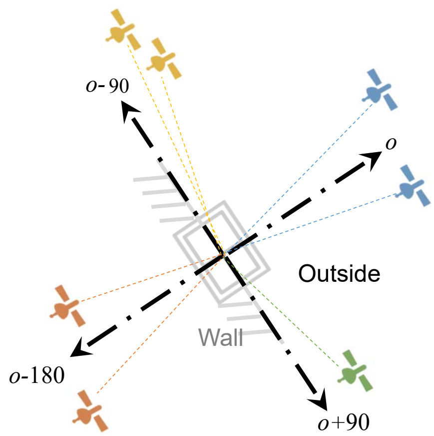
Fig. 10. Time-series of signal strengths from GPS satellites when a smartphone user approaches a window-side area and moves away from the area. PRN refers to a pseudo random noise code of a satellite signal, which can be used to identify the satellite.

The inputs of the window-side detector are extracted from a time window of GPS information $\mathbf{s}_t^g = \{s_{t-W_g}^g, ..., s_{t-1}^g, s_t^g\}$, where W_g is the window size. The time-series of the signal strengths of top-N_G satellites, which have top-N_G largest mean signal strength within $\mathbf{s}_t^g$, are chosen as inputs for the window-side detector (Fig. 8). In other words, the input is N_G-dimensional time-series with the length of W_g, whereas N_G satellites are sorted in the descending order of the mean signal strength. By using the input, we can judge if the smartphone is in a window-side area independent of environments and positions of GPS satellites. The output of the window-side detector is a binary value at time t, i.e., "True" or "False," which indicates whether the user is in a window-side area or not at time t.

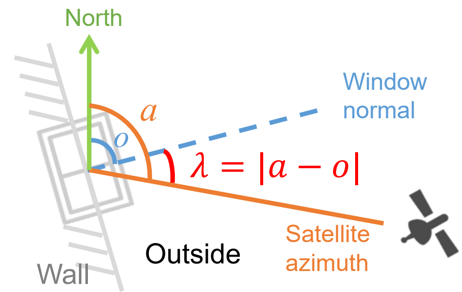
The window-side detector is a binary classifier consisting of two bi-directional LSTM layers with 64 and 32 nodes using the ReLU activation function, a densely connected layer with 16 nodes using the ReLU activation function, and an output layer with the softmax activation function. The network is trained to minimize the binary cross entropy loss by using the Adam optimizer.

3.4.2 Window Orientation Classifier. When the window-side detector determines that the user is in a window-side area, the window orientation classifier will estimate the orientation of the window, i.e., window normal, which is identical to that of the outer wall onto which that window is installed. We design the window orientation classifier based on our idea that when a smartphone is located close to a window, the signals arriving from satellites whose azimuth aligns with the orientation of that window are stronger than those from other satellites, as shown in Fig. 1 (b).

Note that the window orientation classifier (and the other machine learning models) is trained in environments where the orientation of a wall can be different from that of the target environment. Therefore, the window orientation classifier should be designed so that it can deal with a wall facing any orientation. To achieve this, we design the window orientation classifier so that it predicts whether the orientation of the window of interest is facing a certain candidate orientation or not, i.e., binary classification—"True" vs. "False"—by mainly using information about GPS satellites whose azimuth aligns with that orientation. Provided that the floor plan of the target environment is available, orientation candidates can be obtained from the floor plan (e.g., north, west, east, south candidates for a building consisting of four walls: north, west, east, south). That is, the window orientation classifier determines whether the window of interest faces each candidate orientation, and the orientation candidate with the highest probability of "True" is regarded as the orientation that the window of



(a) Selecting GPS satellites positioned at four different orientations; o, $o-90$, $o-180$, $o+90$.



(b) Angle between window normal and satellite azimuth

Fig. 11. Selecting GPS satellites used for inputs of the window orientation classifier. (a) We select satellites close to each of the four different orientations: o (window normal), $o-90$, $o-180$, $o+90$. Signals from satellites at o are expected to be strong when a smartphone is close to the window. (b) Top-N_G satellites with smallest λ are selected for each orientation.

interest faces. With this design, we can determine whether a window of interest faces any orientation, achieving an environment-independent classifier.

The inputs of the window orientation classifier are also extracted from a time window of GPS information $\mathbf{s}_t^g = \{s_{t-W_g}^g, ..., s_{t-1}^g, s_t^g\}$. To determine whether the orientation of the window of interest aligns with the orientation candidate o, the input of the window orientation classifier is calculated using information of GPS satellites existing in four orientations (o, $o+90$, $o-90$, $o-180$), as shown in Fig. 11 (a). When the window faces o, signals from satellites positioned at o are expected to be stronger than those from the other orientations. As for o, we find top-N_G satellites whose angle between its azimuth and o is small, as shown in Fig. 11 (b). Then, we use the time-series of signal strength, absolute angle λ between the window normal and azimuth of the satellite, and elevation angle of each top-N_G satellite as the inputs. Note that these satellites are sorted in ascending order of λ. For the other orientations, i.e., $o+90$, $o-90$, and $o-180$, we use the same procedure to extract the time-series of each orientation. As a result, the inputs of the window orientation classifier are $4 \times 3 \times N_G$ dimension time-series with the length of W_g. Note that the 4 refers to the four orientations and the 3 to the signal strength, absolute angle, and azimuth time-series. Here we selected 4 because many buildings has walls with four different orientations.

The output of the window orientation classifier is a binary value, i.e., "True" vs. "False," which indicates whether the window orientation is o or not at time t.

The neural network for the window orientation classifier is a binary classifier consisting of one bi-directional LSTM layer containing 16 units with the ReLU activation function, a Batch-Normalization layer, and an output layer with the softmax activation function (Fig. 9). The network is trained to minimize the binary cross entropy loss by using the Adam optimizer.

As mentioned above, the window orientation classifier is trained in environments where the walls can be oriented differently from the target environment. Therefore, using a four-class classifier to determine the window orientation would result in a limited performance. For instance, if the training environment is composed of four walls facing north, east, south, and west, the trained classifier will not be able to deal with a test environment

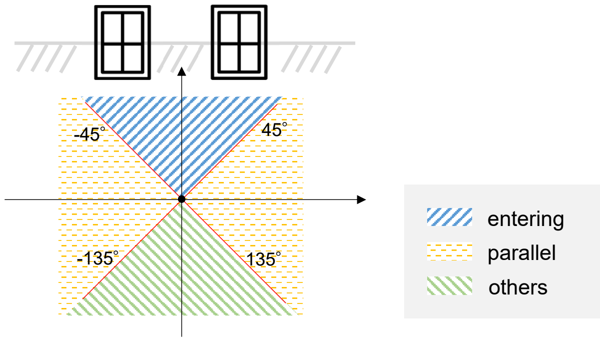


Fig. 12. Three classes of walking direction. When a smartphone user walks toward the window (wall), the direction is classified as "entering." When the user walks parallel to the wall surface, it is classified as "parallel." Otherwise, it is classified as "others."

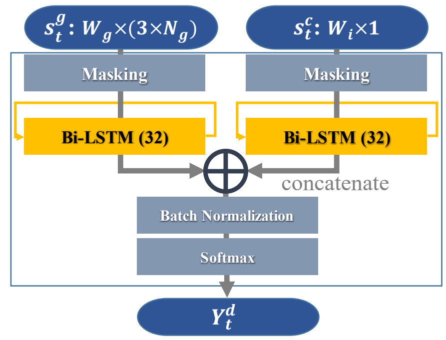


Fig. 13. Network architecture of the walking direction predictor. s_t^g and s_t^c show input time-series of GPS and digital compass data, respectively. Y_t^d shows the predicted walking direction.

also comprising four walls but which face northwest, southwest, southeast, and northeast. Similarly, the trained classifier will be unable to handle a test environment if it is composed of three or five walls. In contrast, a binary classifier like ours outputs a binary value which indicates the probability that the orientation of a window that the user is close to is identical to orientation o. Thus, regardless of the number of walls in the test environment, we can predict the binary value for each wall by setting the normal of the wall as o. For example, if the first wall faces the northeast, we calculate inputs of the orientation classifier by using satellites located around the northeast direction and predict a binary value (probability) by using these inputs. Likewise, if the second wall faces north, we calculate inputs of the orientation classifier by using satellites located around the north direction and predict a binary value by using those inputs. After performing the procedure for each wall, we take the wall with the highest probability and output the estimated wall orientation.

3.4.3 Walking Direction Predictor. The walking direction predictor employs digital compass and GPS data to estimate the rough direction of walking when the user is estimated as having entered a window-side area. The trajectory estimation module, which is described below, is designed to estimate the trajectories before and after the time when the user initially enters the window-side area, based on the predicted walking direction. The output of the walking direction predictor is either of three classes: entering, parallel, and others, as shown in Fig. 12. "Entering" means that the user walks toward the nearby window, and we define "entering" as the range of angles between the walking direction and the wall normal line from -45 degrees to 45 degrees. "Parallel" means that the user walks parallel to the wall (window) surface. It is not defined as strictly parallel but as the range of angles between the user's forward direction and the wall normal line from 45 degrees to 135 degrees or from -45 degrees to -135 degrees. "Others" means that the user leaves the window-area, and we define "others" as the range of angles between the walking direction and the wall normal line from 135 degrees to 225 (-135) degrees.

The inputs of the walking direction predictor are extracted from a time window of digital compass data $\mathbf{s}_t^c = \{s_{t-W_c}^c, ..., s_{t-1}^c, s_t^c\}$, where W_c is window size, and a time window of GPS information $\mathbf{s}_t^g = \{s_{t-W_g}^g, ..., s_{t-1}^g, s_t^g\}$. Assume a window of interest facing orientation o. Because the digital compass outputs the azimuth that is relative to the north, we convert the orientation outputs to relative angles toward o, and employ the time-series of the relative angles as the input. In addition, like the window orientation classifier, we find the N_G satellites that exist in the orientation o, and employ those elevation, relative angle (angle between satellite azimuth and orientation o), and signal strength time-series data as the input of the walking direction predictor.

As shown in Fig. 13, the walking direction predictor has a bi-directional LSTM layer with 32 nodes and the tanh activation function that processes orientation data, and another bi-directional LSTM layer with 32 nodes that processes GPS data. The network is trained to minimize the categorical cross-entropy loss by using the Adam optimizer.

3.5 Trajectory Estimation Module

The proposed method employs a particle filter [17], which is usually used to estimate inner states of a non-linear system and also used in map matching-based human tracking [33, 34], for user tracking because it enables us to easily combine outputs from the GPS landmark module and neural PDR module. In the particle filter, a particle at time t is one of the position candidates of the user at time t. As shown in Fig. 3, the first window-side position generator generates user position candidates (particles) in window-side areas facing direction o_i that the user is predicted to position at time t_i when the user is detected initially entering a window-side area. When there are multiple window-side areas facing o_i, the user's position at time t_i is ambiguous. Then, the trajectory before and after time t_i is estimated by the particle filter with map matching. By map matching, the estimated candidates (particles) that collide with walls are discarded to eliminate the ambiguity of the user position as shown in Fig. 3.

3.5.1 First Window-side Position Generator. As mentioned above, at each time step the window-side detector estimates whether or not the user is in a window-side area. When N_c consecutive detections of the user in a window-side area occur after the user starts walking, where N_c is a hyperparameter regarding the number of window-side detections, the user is determined to have entered the window-side area. Using the orientation o predicted by the window orientation classifier at the first detection time t_i, we generate candidates for the user's absolute coordinates at time t_i, as shown in Fig. 3. Specifically, N_i particles are randomly generated in the window-side areas facing o, where N_i is a hyperparameter regarding the number of generated particles. Then, we assign an initial rough walking direction to each particle according to the estimated probabilities of the three walking direction classes. Therefore, we make a weighted random selection for the walking direction class of the particle, such that the choice is weighted in favor of classes with greater probability, because our preliminary investigation revealed that the classification accuracy of the walking direction predictor is not perfect. When "entering" is selected, the initial walking direction of a candidate at time t_i is randomly selected from $o - 45$ to $o + 45$, as shown in Fig. 12. In the case of "parallel," the walking direction is randomly chosen from $o + 45$ to $o + 135$ or $o - 135$ to $o + 45$. Because the "others" class can be interpreted as the user not entering a window-side area, in this case the particle is discarded.

In addition, to reduce computational cost, we discard initial particles that may have incorrect positions or walking directions. Assume that t_i is the first window-side detection time and t_{i+N_c} is the last time of the first consecutive window-side detection. We calculate the trajectory between time t_i and time t_{i+N_c} for each initial particle using the outputs of the neural PDR module. When all predicted positions within the trajectory are in a window-side area, the particle is used in forward-tracking/backward-tracking. Otherwise, the particle is discarded.

3.5.2 Forward-tracking. The user's trajectory after time t_i is estimated by the particle filter using outputs of the neural PDR module, as shown in Fig. 14. In the particle filter, a particle p_i at time t represents a candidate of the user's position at time t, containing information regarding the indoor coordinates $x^t_{p_i}$, walking orientation $o^t_{p_i}$, and weights $w^t_{p_i}$. The output of the GPS landmark module is used to correct the accumulated errors in the trajectory. For example, when the GPS landmark module estimates that the user is in window-side areas facing a certain orientation, a low weight is assigned to particles outside these areas. In addition, the weights of particles that collide with the wall are all set to 0.

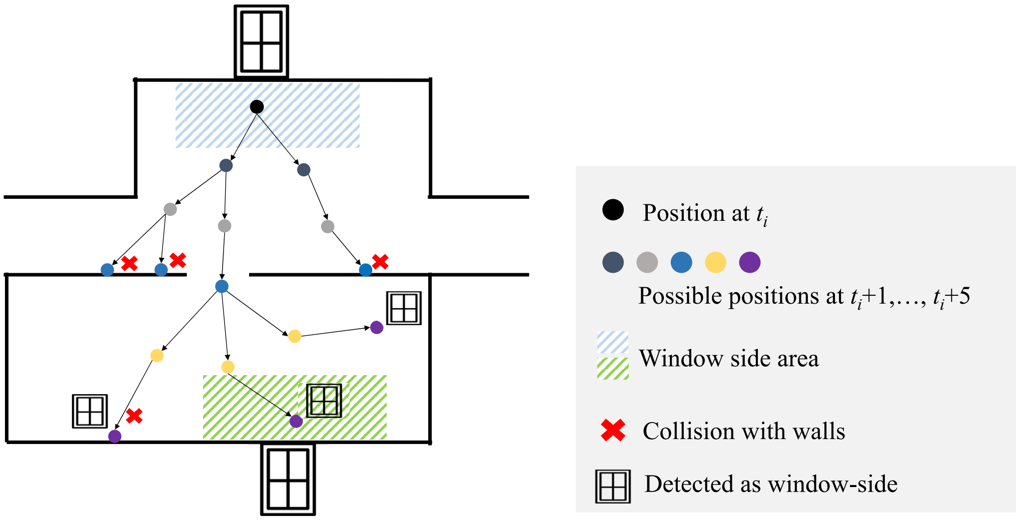


Fig. 14. Tracking a smartphone user with the particle filter from time t_i at which the user is detected to be on the blue window-side area. Several movement paths were generated by the particle filter by varying the displacement and movement orientation, and then the paths colliding with the walls were deleted. The user is detected as being in the green window-side area at time t_{i+5}. Therefore, the position estimate at time t_{i+5} within the green area can be estimated as the true user position at time t_{i+5}.

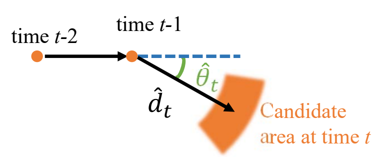


Fig. 15. Position candidates, i.e., particles, at time t are sampled from the candidate area, which is determined based on $\hat{d}_t$ and $\hat{\theta}_t$

The tracking algorithm based on the particle filter works in iterations of a three-step process: sampling, weight calculation, and re-sampling. In the sampling process, new particles at time t are generated from each particle at the previous time step (i.e., time $t-1$) based on a motion model. The generated particles show prior estimates of the user location at time t. In the weight calculation process, the particle weights are computed based on outputs of the GPS landmark module and the floor plan. In the re-sampling process, particles are re-sampled according to their weights.

[Sampling] In the sampling process, we sample the coordinates of the i-th particle at time t from a candidate area as shown in Fig. 15, which is determined based on the coordinates at time $t-1$ and the predicted displacement and change of direction at time t by the neural PDR module. The motion model to sample $x^t_{p_i}$ is defined as follows:

$$x^t_{p_i} = x^{t-1}_{p_j} + \mathcal{N}(\hat{d}_t, \sigma_d)\vec{v}^t_{p_i},$$

where $\mathcal{N}(\mu, \sigma)$ is the normal distribution with the mean μ and standard deviation σ, $\hat{d}_t$ is an estimate of displacement at time t by the walking distance predictor, σ_d is a hyper-parameter, and p_j represents the j-th particle, which is a position estimate at time $t-1$. Assume that N_p particles at time t are generated from p_j at time $t-1$

and p_i is one of them. $\vec{v}^t_{p_i}$ is a unit vector that indicates the walking direction of p_i at time t, and the direction $\angle\vec{v}^t_{p_i}$ is determined by the following formula:

$$\angle\vec{v}^t_{p_i} = o^{t-1}_{p_j} + \mathcal{N}(\hat{\theta}_t, \sigma_o)$$

where $\hat{\theta}_t$ is an estimate of the change of walking direction at time t by the walking direction change predictor and σ_o is a hyper-parameter. Then, $o^t_{p_i}$ is updated as follows.

$$o^t_{p_i} = \angle\vec{v}^t_{p_i}$$

[Weight calculation] For each time step t, we calculate initial weights of sampled particles so that a particle located at the center of the candidate area shown in Fig. 15 at time t has high weight. The weight for p_i is calculated as $w^t_{p_i} = w_d \cdot w_o$, where

$$w_d = \begin{cases} \mathrm{CDF}(d_{p_i}, \mathcal{N}(\hat{d}_t, \sigma_d)), & \text{if } d_{p_i} \leqq \hat{d}_t \\ 1 - \mathrm{CDF}(d_{p_i}, \mathcal{N}(\hat{d}_t, \sigma_d)), & \text{otherwise} \end{cases} \quad \text{and} \quad w_o = \begin{cases} \mathrm{CDF}(\theta_{p_i}, \mathcal{N}(\hat{\theta}_t, \sigma_o)), & \text{if } \theta_{p_i} \leqq \hat{\theta}_t \\ 1 - \mathrm{CDF}(\theta_{p_i}, \mathcal{N}(\hat{\theta}_t, \sigma_o)), & \text{otherwise} \end{cases}$$

In addition, d_{p_i} is a sampled displacement of p_i, θ_{p_i} is a sampled direction change of p_i, and CDF() shows the cumulative distribution function. As shown in the above equations, when value of d_{p_i} or θ_{p_i} is closer to $\hat{d}_t$ or $\hat{\theta}_t$, respectively, the weight of p_i becomes larger. We then update the particle weights by using the floor plan and outputs of the GPS landmark module. First, when a particle collides with an inner or outer wall defined in the floor plan, its weight is set to 0. Then, we update their weights only when the window-side detector output is True by using the following equation.

$$w^t_{p_i} = \begin{cases} \alpha \cdot w_d \cdot w_o + P^w_t + P^{o_c}_t, & \text{if the particle in a window-side area facing orientation } o_c \\ \alpha \cdot w_d \cdot w_o, & \text{otherwise} \end{cases}$$

where α is a hyper-parameter, P^w_t is the probability of "True" class by the window-side detector at time t, and $P^{o_c}_t$ is the probability for orientation candidate o_c by the window orientation classifier. By this equation, the weight of a particle in window-side areas facing the estimated orientation o by the window orientation classifier becomes large.

[Re-sampling] From the weighted samples, we re-sample N_r particles according to their weights. Here, the probability with which a particle is re-sampled is proportional to its weight. The posterior estimate of the user's position at time t is the weighted average position of the N_r particles.

3.5.3 Back-tracking. We also back-track the user's trajectory before time t_i by the particle filter. The basic procedure is almost identical to the procedure of forward-tracking. Note that it back-tracks the user's trajectory from the position at time t_i to that at time 0 by reversing movement vectors.

4 EVALUATION

4.1 Experimental Settings

Because the proposed method is independent of the PDR models, we employed both a user-dependent and a user-independent PDR model to evaluate our method.

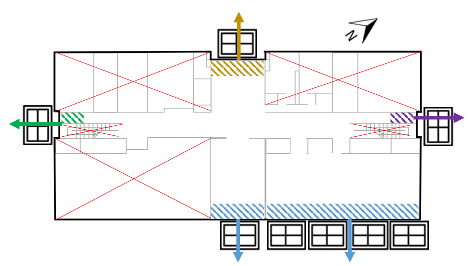
4.1.1 Environments. We collected sensor data at 21 different experimental environments, divided into two datasets. For the first dataset, we asked a participant to collect data at our four main environments (buildings) to investigate the user-dependent PDR model. For the second dataset, we collected data both at the four main environments and at the seventeen additional environments from five different participants per environment. Details of the four main environments are shown in Fig. 16 and Table 2. Environment A is an office floor on

Table 2. Experimental environments used in this study. Float shows float glass. ALC stands for autoclaved lightweight aerated concrete.

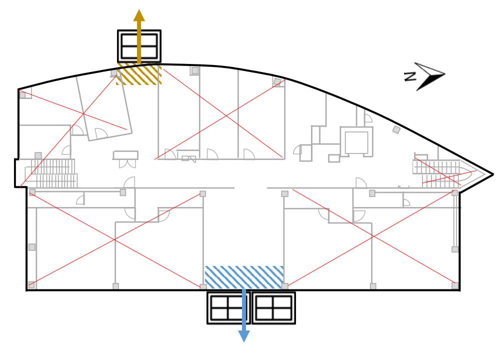
Env.	Floor	Size	Type	Window/wall
A	2F	22.9m x 43.2m	Office	Float/ALC
B	6F	21.7m x 41.9m	Office	Float/ALC
C	3F	22.2m x 26.2m	Multipurpose	Float/ALC
D	3F	35.4m x 49.2m	Library	Float/Concrete

the 2nd floor and there are four window-side areas, which are located near the north and south staircases, in the west open space, and in the east conference rooms. There is one short neighboring building and one tall building located to the west and south of Environment A, respectively. Environment B is also an office floor on the 6th floor. There are two window-side areas, located in the west room and in the east room. The curved walls in Environment B are approximated as straight lines in a floor map used in our method. As Environment B is on the 6th floor, there are no neighboring buildings to the west or east. Environment C is a multipurpose room on the 3rd floor, where metallic barriers are installed on the exterior walls. These metallic barriers interfere with GPS signals. In addition, there are tall buildings located on both sides of the building (left and right in the picture). Environment D is a library environment on the 3rd floor, where walls are very thick. Although there are windows on all sides, there are many desks on the south and north window-side areas. There is one neighboring tall building (located to the right in the figure). Refer to Table 26 in the appendix for detailed information about the additional environments.

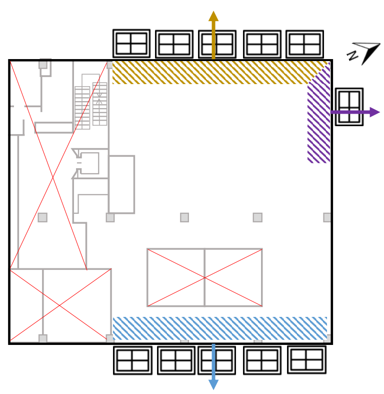
4.1.2 Data Collection. We used a Google Pixel 4 with Android 10 to collect IMU, orientation, and GPS data with a self-developed application and used another Google Pixel 3a with Android 10 to collect ground truth with ARCore [14] simultaneously. We employed ARCore to acquire the user's coordinates at 1 Hz, and then we calculated $d_{(t)}$ and $\theta_{(t)}$, i.e., the ground truth of the neural PDR module, from the coordinates. Referring to information regarding positions of window-side areas in the floor map, the ground truth of the GPS landmark module is also calculated from the positions by ARCore. In this study, all the sensor data were collected at the default sampling rates of the smartphone. The sampling rates of the accelerometer, gyroscope, digital compass, and GPS receiver are 100 Hz, 50 Hz, 50 Hz, and 1 Hz, respectively. Then, the acceleration data were downsampled to 50 Hz. GPS data comprise the elevation angle, azimuth angle, signal strength, and PRN (identifier) of each visible satellite. When we collected each walking trajectory in an environment; a participant holding a smartphone randomly walked in the environment so that the participant passed a window-side area at least once. Table 3 lists the number of trajectories and total duration of the data recorded at each environment for the first dataset, which was collected by one participant. For additional experiments, which will be introduced later, Motorola G pro and Sharp Aquos Sense4 Plus smartphones were used. Refer to Table 26 in the appendix for the number of trajectories and total duration of the data recorded in each environment for the second dataset. When collecting the second dataset, we asked participants to carry two smartphones: one in a hand and the other in a shirt pocket. We also asked the participants to change the walking speed to simulate different walking patterns, such as uninterrupted walking and strolling. When the participant carrying the smartphones passed by a window-side area, we asked additional subjects to stand between the participant and the window. We recorded approximately 15 trajectories from each participant at each environment by locating 0, 1, or 2 people blocking the GPS signals and collecting five trajectories per condition. Note that the obstacle participants were located at one window-side area in each environment.



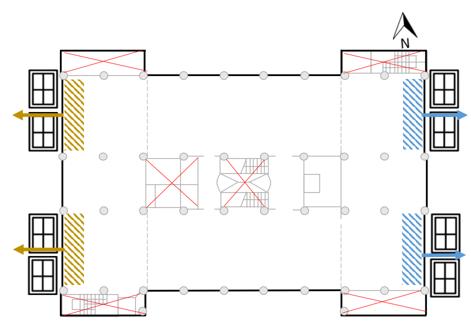
(a) Environment A



(b) Environment B



(c) Environment C



(d) Environment D

Fig. 16. Floor plans of the experimental environments. Window icons represent locations of windows installed. The colored rectangles with diagonal stripes represent corresponding window-side areas inside the building and different colors correspond to different orientations. In addition, a colored arrow shows the normal of the corresponding window. The red crosses mark private areas in these environments, which we were unable to enter.

4.1.3 Evaluation Methodology. For the window-side detector, which is a binary classifier, we used classification accuracy (ratio of correctly classified instances) and confusion matrices to evaluate it. For the window orientation classifier and window direction predictor, we also used classification accuracy to evaluate them. For the trajectory estimation module, we calculated the mean absolute error (MAE) between the ground truth coordinates and the weighted average of posterior estimates of particles in survived trajectories for each time step for an evaluation metric. Note that when all particles collide with the wall at time t, the method cannot output results. In such a case, it ignores walls only at time t.

In the evaluation of the trajectory estimation module, we compared the proposed method with other methods, listed in the following:

Table 3. Overview of dataset in each environment

Env.	# of trajectories	Total data duration
A (Pixel)	22	53 min 41 sec
A (Aquos)	13	40 min 05 sec
A (Moto)	14	42 min 29 sec
B (Pixel)	6	13 min 29 sec
C (Pixel)	20	38 min 31 sec
D (Pixel)	10	46 min 44 sec

- **neural PDR**: This is regarded as a state-of-the-art neural network-based method. Only map matching based on the particle filter and PDR results by the neural PDR module were used. In addition, initial position and direction were given.
- **W/o WOC**: This is a variant of the proposed method that does not use results from the window orientation classifier. Because the window orientation classifier is not included in this method, the amount of information regarding a GPS landmark is limited. This method is used to investigate the effect of the information. Thanks to the window-side detector, this method is able to recognize when the user is at a window-side area. However, it cannot identify the orientation of the window. Therefore, the first window-side position generator produces particles in all the window sides. In the re-sampling process, only the results of the window-side detector are used to calculate particle weights. That is, when we calculate the particle weight, $P_t^{o_c} = 0$. Note that the number of initial particles generated is the same as in the proposed method.
- **W/o WDP**: This is a variant of the proposed method that does not use results from the walking direction predictor. That is, this method cannot identify the walking direction at the first window-side position. Therefore, the walking directions of the particles at the first window-side position are randomly sampled from all directions (0–360 degrees). This method is used to investigate the effect of the initialization. Note that the number of first window-side particles generated is the same as in the proposed method.

In addition, to investigate the effectiveness of the modules in the proposed method, we also applied the following simple methods for window-side detection and window orientation classification:

- **Thresholding**: This is a threshold-based window-side detector that uses a decision stump, i.e., a one-level decision tree, to determine if a user is close to a window-side based on the overall signal strength. The input feature for the detector is the average strength of the signals from all the visible satellites computed within a time window starting at time t. The output is a binary value, "True" or "False," which indicates whether the user is in a window-side area at time t.
- **Sudden-change**: This method determines the window orientation based on the sudden increase in the strength of the signals from the GPS satellites. When a user is detected in a window-side area at time t, we analyze the strength time-series of the GPS satellite signals within the current and previous time windows (i.e., those starting at t and $t - W$, respectively), each containing W_g samples, where W is 1 s. When we detect a sudden increase in a GPS satellite signal strength at time t, we search the wall (window orientation class) whose normal is the closest to the azimuth of each of the satellites. Then we employ the majority voting to select a window orientation class, which is an estimate of the window orientation classification. When we do not detect sudden increases in the current time window, we analyze the previous windows, i.e., t vs. $t - W$, t vs. $t - 2W$, ... and so on. Note that, when we detect a sudden increase, we calculate the average satellite signal strength within the current and previous windows. When the average strength at t is c_s times larger than that at $t - W$, we assume that the sudden change occurs in the satellite at time t.

Table 4. Experimental parameters used in this experiment

Parameter	Value	Description
W_i	50	window size in neural PDR module
N_G	6	top-N_G satellites are used as inputs
W_g	5	window size in GPS landmark module
W_c	5	window size in walking direction predictor
σ_d	0.05	standard deviation of normal distribution when sampling displacement
σ_o	20°	standard deviation of normal distribution when sampling direction change
N_c	5	N_c consecutive window-side detections are used in the first window-side position generator
N_i	5000	N_i particles generated by the first window-side position generator
N_p	50	N_p particles generated from each particle from each time step in sampling
N_r	20	re-sampled N_r particles after weight calculation

- **Angle-regression**: In our method, we employ a binary window orientation classifier. In contrast, the angle regression method directly outputs the orientation of the window. The inputs for the method are signal strength, azimuth, and elevation angle time-series from each of the N_G satellites with the highest average signal strength. The network structure of this method is identical to our method except for the output layer, which employs a linear activation function that produces a numeric value corresponding to the orientation. The network is trained to minimize the mean absolute error between an estimated value and the ground truth of the orientation of the closest window, using the Adam optimizer. After obtaining an output from this method, we select the wall whose normal is closest to the estimated window orientation.

The user-dependent models were evaluated by running the first dataset through a "leave-one-environment-out" cross-validation, where one environment was used as a testing environment while the others were used to train the window-side detector, window orientation classifier, and PDR module. Considering that, unlike the user-independent setting, the user-dependent PDR model was expected to achieve a precise prediction of the walking distance and direction change, we evaluated the main components of the proposed method, namely the window-side detector and window orientation classifier, in detail when a PDR module yielded optimal accuracy. Note that our method is independent of a PDR module. Therefore, we can replace the current neural PDR module with any state-of-the-art method that is eventually proposed. Note that, in this main experiment, we used only the data from Google Pixel.

To evaluate the user-independent models using the second dataset, we split the environments into five training environments (Environments A, B, E, H, and K) and sixteen test environments. Note that, when a participant was included in a test environment, we did not use the training data from the same participant in the training environments. In addition, we trained a window-side detector, window orientation classifier, and walking direction predictor for both the smartphone in the hand and the one in the pocket. Furthermore, we investigated the effect of various environments and the user-independent PDR model, which is greatly affected by the difference in walking patterns among participants, on the performance of the proposed method.

Table 4 shows experimental parameters used in this study.

4.2 Results of 1st Dataset

4.2.1 Performance of the Window-side Detector. Table 5 shows the classification accuracy of the window-side detector in each environment. Note that we calculate the accuracy only for positions where GPS signals are available. When a smartphone is far from window-side areas and it does not receive any GPS signals, we can

Table 5. Classification accuracy of the window-side detector for each environment

	Env. A	Env. B	Env. C	Env. D
$d_w = 2$	82.3%	70.7%	70.9%	68.1%
$d_w = 3$	81.5%	71.4%	74.9%	76.0%
$d_w = 4$	80.6%	68.0%	72.0%	76.7%

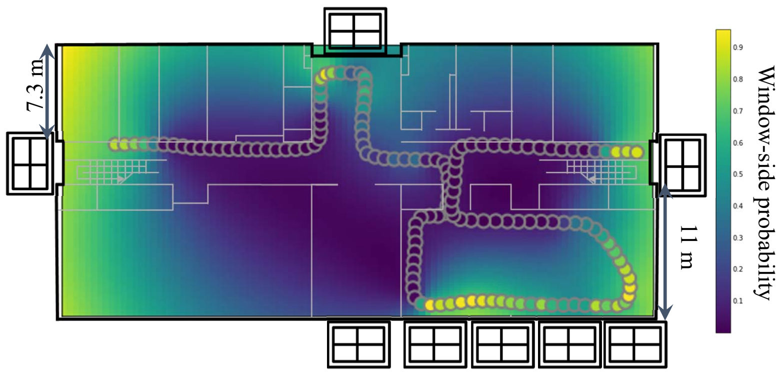
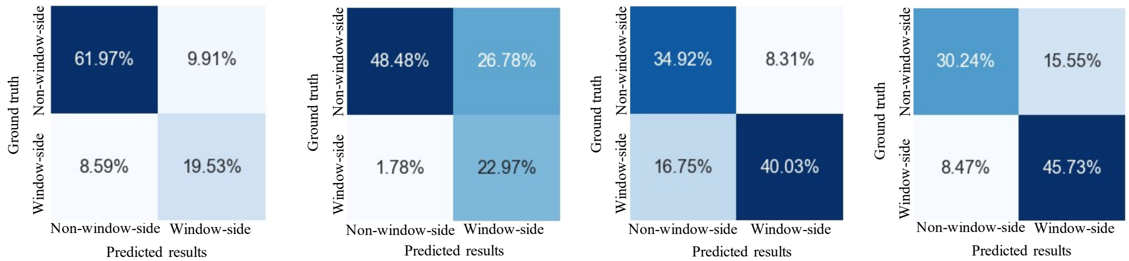


Fig. 17. Heat map of window-side probabilities for a trajectory in Environment A. The gray circles indicate observation points.

easily assume that the smartphone is not in a window-side area. As shown in the table, the window-side detector achieved relatively better accuracy when $d_w = 3$, which was used in our method, compared to the other settings. However, the classification results for these three settings are not very different. Fig. 17 shows a heat map of class probabilities of the "window-side" for a trajectory in Environment A. As shown in the figure, it is difficult to detect strong GPS signals when positioned more than 5 m away from a window. The figure also indicates that the window-side detector yields false detections in areas approximately five meters from a window. In addition, as shown in Table 5, the classification accuracy in Environment B is poorer than that in the other environments. This could be because the window-side areas in that environment are small, as shown in Fig. 16. Because the total stay duration in the window-side areas was short, the classification accuracy was greatly affected by detection errors just after entering/leaving the window-side areas.

Fig. 18 shows the confusion matrix for $d_w = 3$ in each environment. In addition, Table 6 lists the sensitivity and specificity of the window-side detector ($d_w = 3$). As can be observed, many window-side instances in Environments A and C were mistakenly classified as non-window-side cases. As shown in the lower right corner of Fig. 17, even when a smartphone is close to a window, the window-side classifier outputs low window-side probabilities. This can be caused by the presence of pillars at window-side areas. Similarly, as shown in Fig. 18 and Table 6, many non-window-side instances in Environments B and D were mistakenly classified as window-side cases. This is because of a short delay in the reception of GPS signals by the smartphones, which greatly affects the classification accuracy because window-side areas in these environments are small.

Table 7 shows the classification accuracy of the Thresholding method. As indicated by the results, the accuracy of this simple method, which employs the average strength of the signals from all the visible satellites within a time window, is comparatively low. The results in Environment B are particularly poor compared with those in other environments. This can be because overall signal strengths in Environment B were weaker than those in



(a) Environment A (b) Environment B (c) Environment C (d) Environment D

Fig. 18. Confusion matrices of window-side detector results for $d_w = 3$ in each environment

Table 6. Sensitivity and specificity of the window-side detector for $d_w = 3$ (window-side corresponds to the positive class)

	Env. A	Env. B	Env. C	Env. D
sensitivity	69.4%	92.8%	70.5%	84.3%
specificity	86.2%	64.4%	80.8%	66.0%

Table 7. Classification accuracy of window-side detection for the Thresholding method

	Env. A	Env. B	Env. C	Env. D
Thresholding	59.7%	28.5%	49.9%	61.2%
Proposed $d_w = 3$	81.5%	71.4%	74.9%	76.0%

other environments. Because the Thresholding method learns a threshold for the overall signal strength based on training data from other environments, the environmental differences significantly affected it. In contrast, the use of a recurrent neural network enables the proposed method to individually analyze signal strength time-series from each satellite.

4.2.2 Performance of the Window Orientation Classifier. Table 8 shows the classification accuracy of the window orientation classifier in each environment. As shown in the results, the window orientation classifier achieved very high accuracy in Environments A, B, and D. The reason behind the poor accuracy in Environment C compared to the other environments may be difficulties in orientation classification at the top-right corner of the environment, as shown in Fig. 16 (c). Because a smartphone receives signals from two directions at a corner, the classes for both directions have high probabilities. However, incorrect classification at the corners does not affect the final PDR performance significantly. In addition, Table 9 and Fig. 19 show the detailed classification results of the proposed window orientation classifier. The accuracy of the 1st and 3rd walls in Environment A was low, resulting in the low macro-averaged accuracy in that environment. However, because the window-side areas corresponding to the walls (windows) are small, only a few data instances collected.

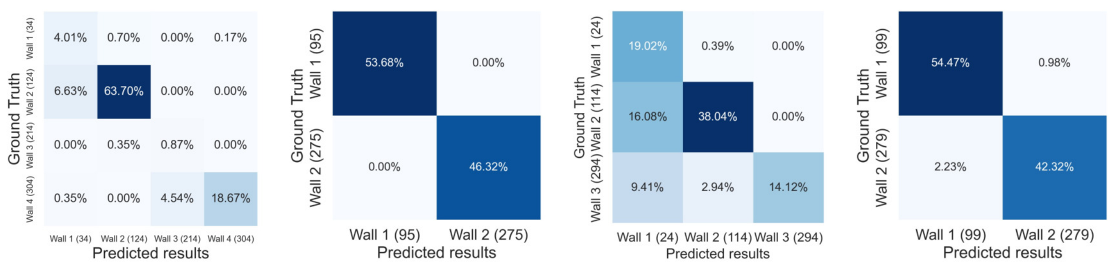
Table 10 shows the accuracy of the window orientation classifier based on the sudden signal strength change when the c_s threshold increases from 1.1 to 1.3. When c_s was larger than 1.3, we could hardly detect any sudden change except in Environment A. As indicated by the results, the orientation classification accuracy using the sudden change method was considerably lower than that using the proposed method, which is due to the signals

Table 8. Overall classification accuracy of the window orientation classifier for each environment

	Env. A	Env. B	Env. C	Env. D
$d_w = 2$	89.5%	100%	59.5%	96.7%
$d_w = 3$	87.3%	100%	69.9%	96.7%
$d_w = 4$	81.5%	100%	64.7%	96.4%

Table 9. Detailed classification accuracy of window orientation classifier and macro average of accuracy ($d_w = 3$)

	Env. A	Env. B	Env. C	Env. D
1st wall	36.5%	100%	42.7%	96.1%
2nd wall	98.4%	100%	91.9%	97.7%
3rd wall	16.1%	n/a	100%	n/a
4th wall	99.1%	n/a	n/a	n/a
macro average	62.5%	100%	78.2%	96.9%



(a) Environment A (b) Environment B (c) Environment C (d) Environment D

Fig. 19. Confusion matrices of window orientation classifier results for $d_w = 3$ in each environment, with the number in parentheses showing the wall angle relative to the north

received from satellites that are not oriented perpendicular to the window close to the user. We found that, while the signals that a smartphone receives from such satellites are weak, their strengths sometimes suddenly increase even when the satellites are not oriented in the direction of the window normal. However, the classification accuracy was significantly higher in Environment D than in other environments. This can be because the walls of this building were made of thick concrete. Thus, in many cases in which the smartphone was close to a window, the signals from satellites not oriented in the direction of the window normal were blocked by the thick walls. Therefore, we can easily identify the directions of the environment's windows through this simple method.

Table 11 shows the classification accuracy and mean absolute error (MAE) based on the angle regression. As with our proposed method, in Environments A and C, the classification accuracy is lower and the MAE is higher than in other environments. However, the classification accuracy based on angle regression is slightly lower than that obtained with our proposed approach, which may be attributed to the limited variation in ground truth labels, i.e., the angle relative to the north. Assuming a training environment with four walls, only four values (angles) are possible for the ground truth labels, which are not sufficient for the regression task.

Table 10. Classification accuracy of orientation classification for the Sudden-change method

	Env. A	Env. B	Env. C	Env. D
Sudden-change ($c_s = 1.1$)	31.8%	42.3%	39.6%	84.1%
Sudden-change ($c_s = 1.2$)	28.6%	35.9%	38.8%	76.4%
Sudden-change ($c_s = 1.3$)	24.3%	64.7%	45.3%	73.3%
Proposed	87.3%	100%	69.9%	96.7%

Table 11. Classification accuracy and mean absolute error (MAE) of orientation classification for the Angle-regression method

	Env. A	Env. B	Env. C	Env. D
classification accuracy	67.30%	97.50%	54.40%	89.10%
MAE in degree	41.56	20.37	70.64	21.98

Table 12. Classification accuracy when smoothing esimates of window orientation classifier

	Env. A	Env. B	Env. C	Env. D
Proposed	87.3%	100%	69.9%	96.7%
Smooth (3 windows)	87.6%	100%	71.2%	96.8%
Smooth (5 windows)	87.9%	100%	75.1%	96.9%

The window orientation classifier outputs an estimate of the orientation class for each time step. We can remove sporadic errors of these estimates by using a moving average filter, although this approach causes a loss of real-time trajectory prediction. Note that, because estimates of the window-side detector frequently change in time due to the existence of window-side obstacles such as pillars, this approach may not work well in the window-side detector. Table 12 shows the results of applying this smoothing approach to window orientation classification. For example, the "Smooth (3 windows)" row shows the classification accuracy of the moving average filter with a window size of three. That is, we calculate the average classification probability for each direction averaging the previous, current, and next prediction, and take the orientation class with the highest average probability. As indicated by the results, the moving average filter improved the classification accuracy. Specifically, the improvement in Environment C is significant, indicating that sporadic classification errors frequently occur in this environment. As mentioned above, the classification errors occurred at the corner of the environment. Because the orientation of the window-side area where the user is does not change frequently, this approach is effective for the window orientation classifier.

Here we also investigate a mathematical method to determine the window orientation at each time step by calculating a weight score of each window orientation as follows.

$$W_o = \frac{\sum_{k=1}^{n_{st}} (1 - \frac{\lambda_k}{180}) \times SS_k}{n_{st}},$$

where n_{st} is the number of visible satellites, λ_k represents the absolute angle between the window normal and azimuth of the satellite k as mentioned above and SS_k stands for signal strength of k. Then, we compared W_o of each window orientation and select the largest one as an estimate at each time step. The classification accuracy based on this method in Environments A, B, C, and D was 50.2%, 100%, 50.1% and 96.6%, respectively. For Environments B and D, the mathematical method can also achieve high accuracy, similar to our proposed

Table 13. Classification accuracy of the walking direction predictor for each environment by three methods

	Env. A	Env. B	Env. C	Env. D
GPS+magnetic	55.9%	51.4%	48.0%	66.2%
only magnetic	50.2%	55.0%	36.3%	60.4%
only GPS	53.9%	48.6%	48.0%	48.8%

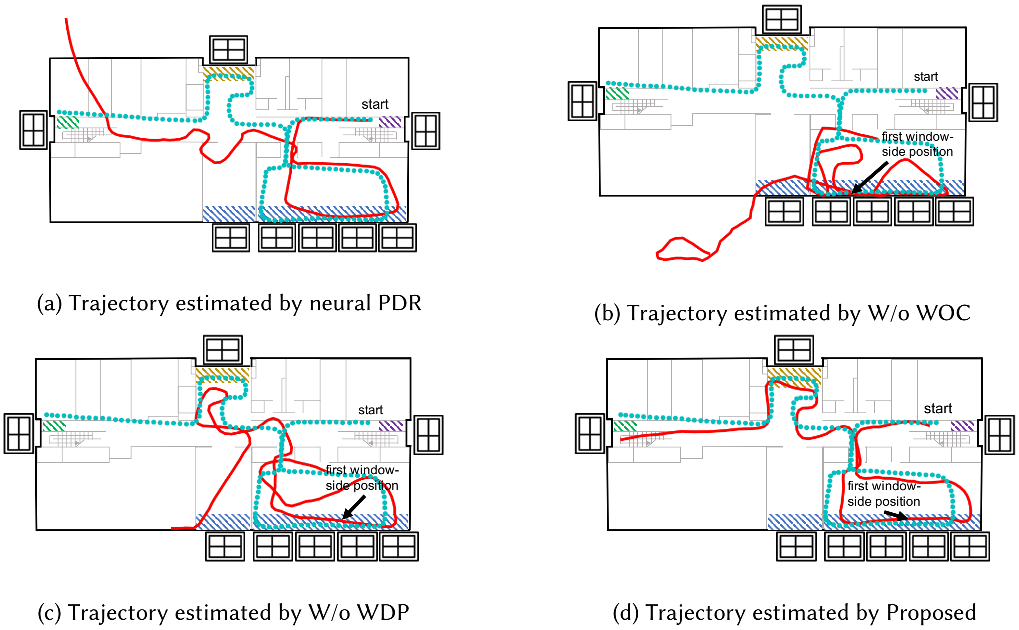
Table 14. Positioning errors of the four methods in each environment. Note that neural PDR requires information about the initial position and initial walking direction.

Positioning error (m)	Env. A	Env. B	Env. C	Env. D
neural PDR	2.595	1.791	2.180	3.978
W/o WOC	6.380	2.762	7.321	12.741
W/o WDP	2.995	2.558	3.247	5.528
Proposed	1.819	1.316	1.267	2.672

method. However, in the other environments, this method cannot get ideal results, much lower than results by the proposed method. This can be because, Environments B and D have only two window orientations, i.e., binary classification, and the window orientations of the two window side areas are opposite, making it easier to distinguish between these areas. However, the other environments have three or four window orientations, and the smartphone can receive some signals from the same satellites even when the smartphone is in window-side areas with different window orientations. This factor may confound the judgment of this mathematical method. In contrast, our method based on machine learning performed well in different environments.

4.2.3 Performance of the Walking Direction Predictor. Table 13 shows the classification accuracy of the walking direction predictor. As shown in the table, while the classification accuracy of the proposed method (GPS+magnetic) is not perfect, it outperforms methods that employ a single sensor. In addition, because the accuracy of these methods is higher than the random guess ratio (i.e., 33%), they can be useful for generating the walking directions of particles produced by the first window-side position generator. The classification accuracy for 'only magnetic' is poor in Environment C. This might be because of magnetic fluctuations in the environment. In addition, the classification accuracy for 'only GPS' is poor in Environment D. This can be attributed to the thick walls in Environment D that block GPS signals. While the walking direction predictor requires GPS time-series as its inputs, the walls prevented the collection of long time-series, which can be necessary for achieving precise classification. However, by combining GPS and magnetic readings, the proposed method (GPS+magnetic) can achieve stable performance in the experimental environments.

4.2.4 Performance of Trajectory Prediction. Table 14 shows the positioning errors of the PDR methods in each environment. As shown in the table, the proposed method significantly outperformed the state-of-the-art neural PDR even though the proposed method does not use information about the initial position and initial walking direction. When the duration of a trajectory is long, all the particles sometimes collide with the inner and outer walls owing to accumulated errors in neural PDR. In such cases, because the method ignored the wall only at that time (i.e., when all the particles were eliminated by map matching), the positioning errors of the neural PDR method increased. The positioning errors in Environment D were larger than those in the other environments because the size of Environment D is large and the window-side areas are sparse, resulting in large accumulated errors.



(a) Trajectory estimated by neural PDR

(b) Trajectory estimated by W/o WOC

(c) Trajectory estimated by W/o WDP

(d) Trajectory estimated by Proposed

Fig. 20. Examples of estimated trajectories in Environment A. The red line represents the predicted trajectory with the methods and blue dotted line represents the ground truth. The first window-side position is detected as the first N_c consecutive detections of 'window-side.'

W/o WOC (window orientation classifier) sometimes failed to generate good particles when the subjects were first detected in window-side areas. Because W/o WOC generates particles at all window-side areas, while the number of generated particles is identical to that of the proposed method, the probability with which particles are generated at actual positions is lower than that of the proposed method, resulting in the poor performance of W/o WOC. The performance of W/o WDP (walking direction predictor) was also poorer than that of the proposed method. Although the performance of the walking direction predictor was not very high, the positioning error of the proposed method was much smaller than that of W/o WDP, which randomly determines the walking direction when a user first enters a window-side area. The impact of the walking direction predictor was higher than we had expected.

Fig. 20 shows examples of trajectories estimated by the methods associated with ground truth trajectories. As mentioned above, because all particles are sometimes eliminated by map matching, the methods sometimes generate trajectories that pass through the walls. However, the proposed method based on GPS information can correct the accumulated errors of the neural PDR module. Specifically, the window orientation classifier enables us to narrow down the candidates for window-side positions. As shown in Fig. 20 (b), W/o WOC usually failed to generate good first window-side positions. W/o WDP also sometimes failed to generate good walking directions for first window-side positions. In this example, W/o WDP could not find a path that starts with the right window-side area (purple area). In contrast, because the proposed method generated many particles with correct walking directions at the first window-side position, the proposed method could find a path that starts with the right window-side area.

Table 15. Prediction results of each module for Aquos phone and Motorola phone

	Window-side	Window orientation	Walking direction	Positioning error (m)
Aquos	85.80%	90.70%	69.80%	1.721
Moto	87.90%	88.80%	70.30%	1.776

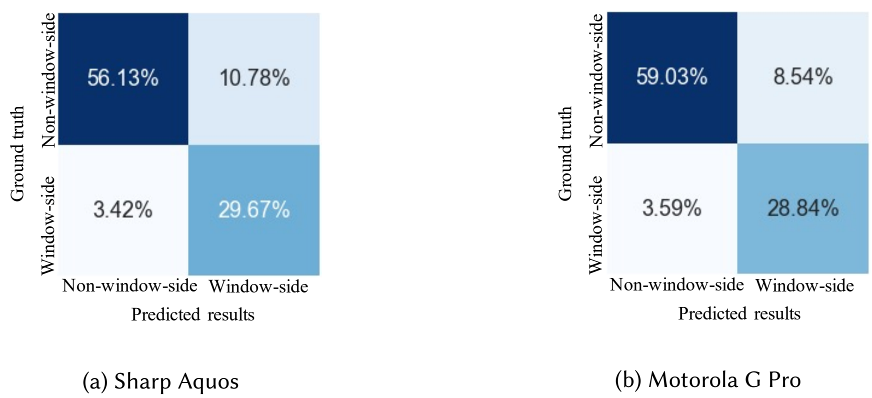


Fig. 21. Confusion matrices of window-side detector results for $d_w = 3$ for different devices

4.2.5 Device Heterogeneity. The main evaluation in this study employed sensor data collected from a Pixel 4 smartphone. However, sensor sensitivity can depend on smartphone products. Here, we tested our method trained on sensor data from the Pixel 4 in Environments B-D by employing sensor data from the Motorola and Sharp Aquos smartphones (Table 3) as test data. Table 15 and Fig. 21 show the results of this experiment. As shown in the table and confusion matrices, the window-side classification performance for these smartphones are very high, indicating the limited effect of the device difference. Similarly, the performance of the window orientation classifier and walking direction predictor was not significantly different from that of the Pixel 4. In addition, the positioning errors for these smartphone are small in this experiment. Based on these results, we were unable to identify the effects of device differences on the proposed method.

4.2.6 Number of Generated Particles by First Window-side Position Generator. According to the estimated walking direction, the first window-side position generator samples the walking direction of each first window-side particle at time t_i as described in Section 3.5.1. In contrast, the W/o WDP method randomly selects the walking directions of these particles from all directions (0–360°). Therefore, the performance of W/o WDP degrades when the number of first window-side particles N_i is small, which is related to the computational costs of W/o WDP. Here, we investigate the effect of N_i on the PDR performance. Table 16 shows the positioning errors of this experiment. As shown in the results, when we reduce N_i, the positioning error for W/o WDP significantly increases. Although the error for the proposed method also increases as N_i decreases, the error for the proposed method is much smaller than that for W/o WDP. As mentioned above, the walking direction predictor can suppress performance degradation when we use a small N_i to reduce the computational cost.

4.2.7 Effect of Wall Materials. This study assumes that GPS signals are blocked by walls. Here we investigate the effect of the wall material on the received signal strength. Fig. 22 shows heat maps of class probabilities for "window-side" when a smartphone was close to a wall without windows. As shown in the figure, when the

Table 16. Positioning errors [m] when N_i is varied for W/o WDP and the proposed method in Env. B

	100%	80%	50%
W/o WDP	2.995	3.788	4.730
Proposed	1.819	2.171	2.905

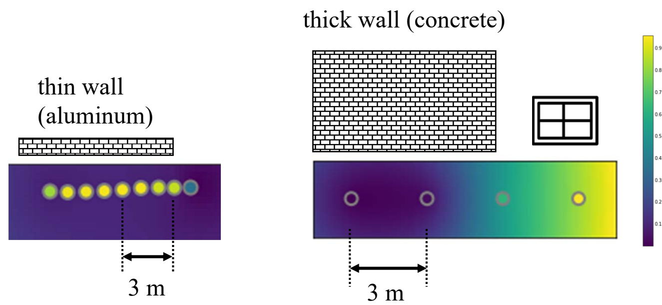


Fig. 22. Heat map of window-side probabilities when a smartphone was closed to a thin/thick wall. The gray circles indicate observation points.

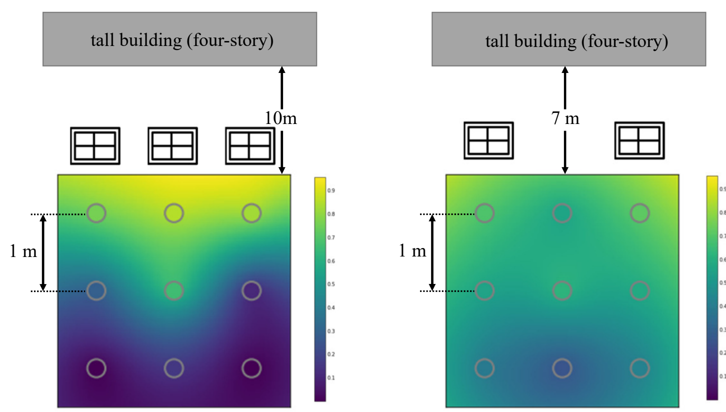


Fig. 23. Heat map of window-side probabilities when a smartphone was close to a thin or thick wall. These environments are the first floors of buildings. The gray circles indicate observation points.

smartphone was close to the concrete wall, GPS signals were completely blocked by the wall. However, when the smartphone was close to the thin wall, the window-side detector mistakenly classified the observation points into the window-side class. The experiment revealed that when we created a floor map of a building with thin walls, the areas around these walls should be defined as window-side areas.

4.2.8 Effect of Nearby Tall Buildings. Because our method relies on GPS signals observed at window-side areas, it is affected by nearby tall buildings that can interfere with the signals. We collected GPS signals from window-side areas in buildings with nearby tall buildings. The distance between the window-side area in the first environment and its nearby building is approximately 10 m. The distance between the window-side area in the second area and its nearby building is approximately 7 m. Fig. 23 shows heat maps of class probabilities for "window-side."

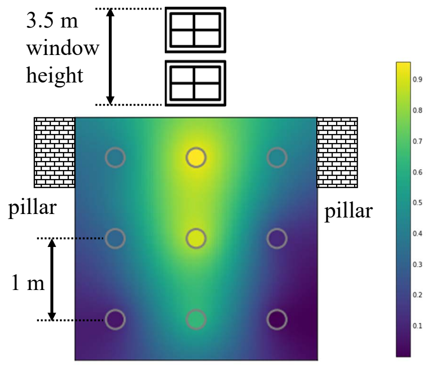


Fig. 24. Heat map of window-side probabilities when the smartphone was close to a tall window. The gray circles indicate observation points.

The figure indicates that it is possible to detect window-side areas, even when there are tall buildings close to an environment of interest. The figure also shows that the signal strength for the 10-meter nearby building was stronger than that for the 7-meter nearby building.

4.2.9 Effect of Window Height . Our experimental environments (A–D) are standard floors with a floor-to-ceiling height of approximately 3.5 m. Here, we collected GPS data in an environment with a tall window (approximately 3.5 m). There are no tall buildings around this environment.

Fig. 24 shows a heat map of class probabilities for "window-side." As shown in the figure, the smartphone received strong signals even when the distance between the smartphone and the window was approximately 2 m. However, the detection range of "window-side" was not significantly different from that in our experimental environments.

4.3 Results of 2nd Dataset

4.3.1 Performance of Walking Distance and Walking Direction Change Prediction. Table 17 summarizes the MAEs obtained from the first and second datasets for the prediction of the walking distance and direction change. As shown in the results, the performance of the walking direction change predictor was not very different between the two datasets. In contrast, the MAE of the walking distance prediction with the second dataset is larger than that with the first dataset, although the error was only 0.08 meters. This performance difference may be caused by the use of various subjects to train the walking direction change predictor.

4.3.2 Performance of Trajectory Prediction . Table 18 presents the positioning errors obtained when running the second dataset through the four methods when participants held a smartphone in their hand. As with the first dataset, in this case, the proposed method also achieved a very high positioning performance in a variety of environments, demonstrating the robustness of the proposed method. Although the performance of the neural PDR module was slightly poorer than that in the first dataset, it was not significantly affected by the positioning performance. However, the MAEs in Environments D, J, O, R for the proposed method were somewhat larger than those in the other environments. The MAE in Environment D was also large in the user independent model because the window-side areas are sparse in the environment. The performance of the window-side detector was poor in Environments J, O, and R, as explained below, resulting in the large MAE in the environments.

Table 19 presents the positioning performance of the proposed method when the smartphone was in a shirt pocket. The performance in this case is much poorer than that when the smartphone was held in the hand. This is

Table 17. MAEs of walking distance and walking direction change prediction

	walking distance (m)	walking direction change (rad)
1st dataset	0.05	0.08
2nd dataset	0.08	0.09

Table 18. Positioning errors (m) of the user-independent models when participants held a smartphone

Env.	Neural PDR	W/o WOC	W/o WDP	Proposed
C	1.780	7.552	2.503	1.157
D	2.268	10.595	5.074	1.703
F	2.585	6.697	2.059	0.974
G	1.681	8.544	3.261	1.516
I	2.828	1.969	1.471	1.099
J	2.018	5.526	4.185	1.698
L	3.458	5.664	2.031	1.480
M	2.121	1.739	1.731	0.931
N	1.195	2.298	2.474	1.076
O	2.639	3.809	2.768	1.590
P	2.742	1.767	2.321	1.068
Q	2.241	1.510	2.241	0.990
R	2.509	3.830	2.354	1.525
S	1.801	2.079	1.367	1.205
T	2.475	3.133	2.512	0.973
U	2.096	6.308	3.633	1.090

because the performance of the window-side detector and window orientation classifier is very poor, as explained next.

Table 20 lists the positioning errors of the proposed method for different numbers of obstacle participants standing close to the window in each environment. As shown in the results, the number of obstacle participants did not greatly affect the positioning performance. This can be because, in general, the size of a window is still sufficiently large to capture GPS signals even when people are standing between the smartphone user and the window when the user is close to the window.

4.3.3 Performance of the Window-side Detector. Table 21 presents the classification results of the user-independent window-side detector when the smartphone was in the hand of the participant. As indicated by the results, the performance of the window-side detector is similar to that under the user-dependent setting in many environments, thus confirming the robustness of the window-side detector. However, the classification accuracy in Environments J, O, Q, and R was poorer than that in the remaining environments. This can be explained by the presence of tall buildings close to one side of Environments J, O, and Q. In addition, there are large eaves at one side of Environment R, which seem to block GPS signals in the corresponding window-side areas. As mentioned above, the positioning performance in Environments J, O, and R was poor, indicating that the classification performance of the window-side detector significantly affects the positioning performance of our method. The positioning performance in Environment Q was high even though the classification performance

Table 19. Positioning errors (m) of the user-independent models for the proposed method when participants held a smartphone or inserted a smartphone in a shirt pocket

Env.	hand	pocket
C	1.157	2.257
D	1.703	2.214
F	0.974	1.411
G	1.516	1.954
I	1.099	1.622
J	1.698	2.041
L	1.480	2.138
M	0.931	1.393

Env.	hand	pocket
N	1.076	1.573
O	1.590	2.180
P	1.068	1.499
Q	0.990	1.570
R	1.525	1.386
S	1.205	1.611
T	0.973	1.505
U	1.090	1.591

Table 20. Positioning errors (m) of the user-independent models when the number of obstacle participants standing close to a window is different (0, 1, 2)

Env.	0	1	2
C	1.136	1.115	1.191
D	1.767	1.770	1.585
F	0.825	1.192	0.957
G	1.365	1.631	1.545
I	1.041	1.086	1.184
J	1.761	1.537	1.800
L	1.507	1.306	1.632
M	0.949	1.004	0.838

Env.	0	1	2
N	1.071	1.129	1.043
O	1.826	1.618	1.109
P	1.127	1.004	1.072
Q	0.955	0.948	1.035
R	1.579	1.491	1.505
S	1.082	1.342	1.329
T	1.091	0.956	0.853
U	0.817	1.266	1.204

Table 21. Classification results of user-independent window-side detector when smartphones were in the hands

Env.	Accuracy	Sensitivity	Specificity
C	71.9%	87.1%	26.9%
D	81.2%	81.8%	80.7%
F	83.4%	80.9%	85.9%
G	76.2%	79.4%	68.4%
I	84.1%	72.1%	87.8%
J	73.0%	69.3%	75.8%
L	76.0%	79.9%	72.2%
M	76.8%	71.5%	80.5%

Env.	Accuracy	Sensitivity	Specificity
N	72.9%	76.2%	70.8%
O	70.1%	56.4%	81.0%
P	77.4%	74.3%	79.2%
Q	69.5%	65.1%	76.5%
R	69.6%	40.1%	90.2%
S	89.1%	83.3%	92.5%
T	91.5%	83.5%	93.7%
U	74.0%	62.9%	84.9%

was poor because the environment has many walls/obstacles and the map matching-based method could correct trajectories based on the information.

As shown in Table 22, the classification accuracy when the smartphone was in the pocket was poorer than that when it was in the hand. While the smartphone can receive strong/stable GPS signals in the pocket when a participant is facing the window, the signals were blocked by the participant's own body when facing the direction opposite to the window.

Table 22. Classification results of user-independent window-side detector when smartphones were in the pockets

Env.	Accuracy	Sensitivity	Specificity
C	61.8%	66.3%	48.6%
D	75.8%	66.4%	83.9%
F	72.4%	48.7%	96.3%
G	52.7%	39.7%	84.9%
I	87.4%	57.0%	96.9%
J	67.5%	58.0%	74.7%
L	64.4%	45.3%	82.5%
M	70.9%	47.2%	87.4%

Env.	Accuracy	Sensitivity	Specificity
N	62.7%	59.1%	65.1%
O	64.3%	64.4%	64.2%
P	76.7%	79.9%	74.9%
Q	60.2%	41.5%	90.5%
R	68.3%	46.7%	83.4%
S	81.3%	64.6%	91.2%
T	86.0%	76.7%	88.7%
U	59.8%	29.3%	89.5%

Table 23. Classification results of user-independent window orientation classifier when smartphones were in the hands

Env.	Overall accuracy	Macro accuracy
C	67.4%	78.5%
D	100%	100%
F	99.6%	96.1%
G	97.8%	96.9%
I	96.3%	92.5%
J	95.4%	94.0%
L	79.2%	80.8%
M	100%	100%

Env.	Overall accuracy	Macro accuracy
N	83.2%	53.3%
O	83.9%	68.0%
P	70.1%	83.0%
Q	88.7%	82.8%
R	71.8%	68.3%
S	99.7%	96.9%
T	83.1%	84.9%
U	77.1%	66.6%

Table 24. Classification results of user-independent window orientation classifier when smartphones were in the pockets

Env.	Overall accuracy	Macro accuracy
C	65.5%	73.8%
D	99.7%	99.7%
F	95.9%	83.8%
G	78.7%	83.5%
I	99.0%	99.0%
J	81.4%	81.1%
L	66.0%	64.3%
M	99.0%	99.2%

Env.	Overall accuracy	Macro accuracy
N	68.6%	50.7%
O	68.0%	70.5%
P	48.8%	74.1%
Q	82.9%	69.1%
R	54.3%	55.5%
S	72.6%	75.4%
T	85.9%	85.9%
U	69.1%	65.5%

4.3.4 Performance of the Window Orientation Classifier. Table 23 introduces the classification results of the user-independent window orientation classifier when the participants held the smartphone in their hands. Overall, the classification accuracy under this setting was very high in most of the environments, even though the classifier was trained on data from other participants. The classification accuracy in Environment C was poorer than that in other environments. As in the case of the first dataset, the classification results at the corners were poor. However, as mentioned above, the influence of the corners when determining the position was limited.

As shown in Table 24, the classification performance of the user-independent window orientation classifier was also deficient when the participants carried the smartphone in the shirt pocket, which can also be attributed to the signal blocking caused by the participant's body.

Table 25. The proportion of each kind of failed trajectories in the total number of trajectories in the four environments

	Category 1	Category 2	Category 3
Ratio	6%	8%	4%

4.4 Discussion

4.4.1 Failed Trajectories. We investigated the reasons for failed trajectory predictions made by the proposed method using the first dataset, assuming that a trajectory with an MAE greater than 3 meters is a failed one. As a result, the failed trajectories were categorized into three types. Table 25 shows the proportion of the total number of trajectories in the four environments corresponding to each type.

(1) **False positive predictions by the window-side detector near the window-side areas.** A false positive prediction by the window-side detector denotes a case where a user, despite not being in a window-side area, was predicted as being there at time t. When we identify the prediction at time t as a false positive prediction but the subject was in the window-side area at $t-1$, the trajectory estimation module was likely to predict that the possible position of the user at time t was still in the window-side area, which may cause an accumulated trajectory error after time t. This issue was observed when a subject is at a boundary of a window-side and non window-side areas.
(2) **All possible particles traversing the nearby wall.** Corresponds to the case where the subject is near a wall (i.e., that the distance between the subject and wall is extremely small) at time $t-1$ and turns around that wall at time t, resulting in the prediction that all possible particles collide with that wall at time t. As a consequence, a null weight is assigned to all particles and the trajectory estimation module randomly chooses the particles. This problem is caused by two factors: a large error in the predicted change of walking direction, and a small σ_o (hyperparameter of the particle filter regarding the walking direction variance).
(3) **Symmetric building structure & lack of information on walls or other obstacles.** There is a special condition in Environment D: the floor plan is symmetric but wall information is insufficient. Therefore, it is likely that the map matching-based trajectory estimation module predicted trajectories whose shape was similar to the ground truth, but each location was different (translation in 2D geometry).

We found that the second category of failed trajectories is the most frequent, posing a challenging problem for us. A simple solution would be to increase the value of σ_o, but this approach requires that we generate more particles in each time step to ensure better accuracy, increasing the computation cost. Another simple approach would be that the system generates other random particles which do not collide with the wall at time t. In future studies, we plan to focus on this problem. In addition, we plan to address the first and third types of failed trajectories by improving the accuracy of the window-side detector and adding more obstacle information.

4.4.2 Detecting Corners in a Floor. There is an environment that has windows on both sides of a corner area, where a smartphone can receive signals from the direction of each of the two windows. If we can detect when a user is at a corner, we can precisely predict the user's position. However, because the availability of environments with windows on both sides of a corner area is limited, we plan to implement a corner detection module that can be trained with limited data.

4.4.3 Limitations. Because our proposed GPS landmarks rely on GPS satellites, they cannot be used in underground environments. In addition, GPS landmarks cannot be used to distinguish between multiple floors in a multi-floor environment. However, as mentioned in the Introduction, because smartphones are equipped with various sensors, GPS landmarks can be used with other types of landmarks that can distinguish between multiple floors. We believe that the advantage of GPS landmarks is their ease of installation, as shown in Table 1. In addition, this is the first study to demonstrate the feasibility of using GPS landmarks.

5 CONCLUSION

This paper presented a new type of indoor landmark based on GPS signals that can be used to correct accumulated errors of IMU-based PDR. GPS landmarks can be observed without installing a new signalling infrastructure. In addition, we designed the GPS landmark module for detecting GPS landmarks so that it is trained on sensor data collected in environments other than the target environment. We evaluated the PDR system equipped with the GPS landmark module on sensor data collected from real environments. The experiment revealed that the GPS landmark module provides indoor contexts useful for correcting accumulated errors of PDR. As a part of our future work, we plan to explorer other applications of indoor contextual information acquired by GPS satellites.

ACKNOWLEDGMENTS

This work is partially supported by JSPS KAKENHI Grant Number JP21H03428, JP21H05299, and JP21J10059.

REFERENCES

[1] Heba Abdelnasser, Reham Mohamed, Ahmed Elgohary, Moustafa Farid Alzantot, He Wang, Souvik Sen, Romit Roy Choudhury, and Moustafa Youssef. 2015. SemanticSLAM: Using environment landmarks for unsupervised indoor localization. *IEEE Transactions on Mobile Computing* 15, 7 (2015), 1770–1782.

[2] Fatemeh Abyarjoo, Armando Barreto, Jonathan Cofino, and Francisco R. Ortega. 2015. Implementing a sensor fusion algorithm for 3D orientation detection with inertial/magnetic Sensors. In *Innovations and Advances in Computing, Informatics, Systems Sciences, Networking and Engineering*, Vol. 305–310. Springer International Publishing, Cham.

[3] Christian Ascher, Christoph Kessler, Matthias Wankerl, and Gert F. Trommer. 2010. Dual IMU indoor navigation with particle filter based map-matching on a smartphone. In *2010 International Conference on Indoor Positioning and Indoor Navigation*. 1–5.

[4] Changhao Chen, Xiaoxuan Lu, Andrew Markham, and Niki Trigoni. 2018. Ionet: Learning to cure the curse of drift in inertial odometry. In *the AAAI Conference on Artificial Intelligence*, Vol. 32.

[5] Changhao Chen, Peijun Zhao, Chris Xiaoxuan Lu, Wei Wang, Andrew Markham, and Niki Trigoni. 2020. Deep-learning-based pedestrian inertial navigation: Methods, data set, and on-device inference. *IEEE Internet of Things Journal* 7, 5 (2020), 4431–4441.

[6] Guoliang Chen, Xiaolin Meng, Yunjia Wang, Yanzhe Zhang, Peng Tian, and Huachao Yang. 2015. Integrated WiFi/PDR/Smartphone using an unscented Kalman Filter algorithm for 3D indoor localization. *Sensors* 15, 9 (2015), 24595–24614.

[7] Zhenghua Chen, Han Zou, Hao Jiang, Qingchang Zhu, Yeng Chai Soh, and Lihua Xie. 2015. Fusion of WiFi, smartphone sensors and landmarks using the Kalman filter for indoor localization. *Sensors* 15, 1 (2015), 715–732.

[8] Thilina Dissanayake, Takuya Maekawa, Takahiro Hara, Taiki Miyanishi, and Motoaki Kawanabe. 2022. IndoLabel: Predicting Indoor Location Class by Discovering Location-Specific Sensor Data Motifs. *IEEE Sensors Journal* 22, 6 (2022), 5372–5385. https://doi.org/10.1109/JSEN.2021.3102916

[9] Marcus Edel and Enrico Köppe. 2015. An advanced method for pedestrian dead reckoning using BLSTM-RNNs. In *2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 1–6.

[10] Raul Feliz, Eduardo Zalama, and Jaime Gómez-García-Bermejo. 2009. Pedestrian tracking using inertial sensors. *Journal of Physical Agents* (01 2009).

[11] Eric Foxlin. 2005. Pedestrian tracking with shoe-mounted inertial sensors. *IEEE Computer Graphics and Applications* 6 (2005), 38–46.

[12] Masahiro Fujii and Yoshiaki Mori. 2016. A study on indoor positioning systems based on SkyPlot mask. In *2016 IEEE 5th Global Conference on Consumer Electronics*. IEEE, 1–2.

[13] Masayuki Fujiwara, Tomoya Nakatani, Yiming Tian, Joseph Korpela, Takuya Maekawa, and Takahiro Hara. 2020. Smartphone-assisted automatic indoor localization of BLE-enabled appliances using BLE and GNSS Signals. In *the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys '20)*. Association for Computing Machinery, New York, NY, USA, 80–89.

[14] Google. 2018. ARCore. https://developers.google.com/ar/.

[15] Fuqiang Gu, Shahrokh Valaee, Kourosh Khoshelham, Jianga Shang, and Rui Zhang. 2020. Landmark graph-based indoor localization. *IEEE Internet of Things Journal* 7, 9 (2020), 8343–8355.

[16] Sheng Guo, Xiong Hanjiang, Xianwei Zheng, and Yan Zhou. 2017. Activity recognition and semantic description for indoor mobile localization. *Sensors* 17 (03 2017), 649.

[17] Fredrik Gustafsson, Fredrik Gunnarsson, Niclas Bergman, Urban Forssell, Jonas Jansson, Rickard Karlsson, and Per-Johan Nordlund. 2002. Particle filters for positioning, navigation, and tracking. *Trans. Sig. Proc.* 50 (2 2002), 425–437.

[18] Michael Hardegger, Daniel Roggen, Sinziana Mazilu, and Gerhard Tröster. 2012. ActionSLAM: Using location-related actions as landmarks in pedestrian SLAM. In *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 1–10.

[19] Yan Huang, Huiru Zheng, Chris Nugent, Paul McCullagh, Suzanne M McDonough, Mark A Tully, and Sean O Connor. 2010. Activity monitoring using an intelligent mobile phone: a validation study. In *the 3rd International Conference on Pervasive Technologies Related to Assistive Environments*. 1–6.

[20] Sakshi Juneja and Sharda Vashisth. 2017. Indoor positioning system using visible light communication. In *2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN)*. 79–83.

[21] Daisuke Kamisaka, Shigeki Muramatsu, Takeshi Iwamoto, and Hiroyuki Yokoyama. 2011. Design and implementation of pedestrian dead reckoning system on a mobile phone. *IEICE Trans. Inf. Syst.* 94-D (2011), 1137–1146.

[22] Wonho Kang and Youngnam Han. 2015. SmartPDR: Smartphone-based pedestrian dead reckoning for indoor localization. *IEEE Sensors Journal* 15, 5 (2015), 2906–2916.

[23] Yong Hun Kim, Min Jun Choi, Eung Ju Kim, and Jin Woo Song. 2019. Magnetic-map-matching-aided pedestrian navigation using outlier mitigation based on multiple sensors and roughness weighting. *Sensors* 19, 21 (2019).

[24] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[25] Mikkel Baun Kjærgaard, Henrik Blunck, Torben Godsk, Thomas Toftkjær, Dan Lund Christensen, and Kaj Grønbæk. 2010. Indoor positioning using GPS revisited. In *International Conference on Pervasive Computing*. Springer, 38–56.

[26] Ye-Sheng Kuo, Pat Pannuto, Ko-Jen Hsiao, and Prabal Dutta. 2014. Luxapose: Indoor positioning with mobile phones and visible light. In *the 20th annual international conference on Mobile computing and networking*. 447–458.

[27] Myeongcheol Kwak, Youngmong Park, Junyoung Kim, Jinyoung Han, and Taekyoung Kwon. 2018. An energy-efficient and lightweight indoor localization system for Internet-of-Things (IoT) environments. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* (2018). https://doi.org/10.1145/3191749

[28] Quentin Ladetto and Bertrand Merminod. 2002. An alternative approach to vision techniques-pedestrian navigation system based on digital magnetic compass and gyroscope integration. In *6th World Multiconference on Systemics, Cybernetics and Information, Orlando, USA*.

[29] Namkyoung Lee and Dongsoo Han. 2017. Magnetic indoor positioning system using deep neural network. In *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 1–8.

[30] Oleg Mezentsev, Gerard Lachapelle, and Jussi Collin. 2005. Pedstrian Dead Reckoning—A solution to navigation in GPS signal degraded areas? *GEOMATICA* 59 (2005), 175–182.

[31] Qun Niu, Ning Liu, Jianjun Huang, Yangze Luo, Suining He, Tao He, S.-H. Gary Chan, and Xiaonan Luo. 2019. DeepNavi: A deep signal-fusion framework for accurate and applicable indoor navigation. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* (2019). https://doi.org/10.1145/3351257

[32] Masayuki Ochiai, Masahiro Fujii, Atsushi Ito, Yu Watanabe, and Hiroyuki Hatano. 2014. A study on indoor position estimation based on fingerprinting using GPS signals. In *2014 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE, 727–728.

[33] Kazuya Ohara, Takuya Maekawa, Yasue Kishino, Yoshinari Shirai, and Futoshi Naya. 2015. Transferring positioning model for device-free passive indoor localization. In *the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*. 885–896.

[34] Hiroaki Santo, Takuya Maekawa, and Yasuyuki Matsushita. 2017. Device-free and privacy preserving indoor positioning using infrared retro-reflection imaging. In *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 141–152.

[35] Jim Scarlett. 2007. Enhancing the performance of pedometers using a single accelerometer. *Application Note, Analog Devices* (2007).

[36] Masaya Tachikawa, Takuya Maekawa, and Yasuyuki Matsushita. 2016. Predicting Location Semantics Combining Active and Passive Sensing with Environment-Independent Classifier. In *the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Heidelberg, Germany) *(UbiComp '16)*. Association for Computing Machinery, New York, NY, USA, 220–231. https://doi.org/10.1145/2971648.2971684

[37] Daisuke Taniuchi and Takuya Maekawa. 2015. Automatic update of indoor location fingerprints with pedestrian dead reckoning. *ACM Transactions on Embedded Computing Systems (TECS)* 14, 2 (2015), 1–23.

[38] Raghav H. Venkatnarayan and Muhammad Shahzad. 2019. Enhancing indoor inertial odometry with WiFi. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 2 (2019). https://doi.org/10.1145/3328918

[39] He Wang, Souvik Sen, Ahmed Elgohary, Moustafa Farid, Moustafa Youssef, and Romit Roy Choudhury. 2012. No need to war-drive: Unsupervised indoor localization. In *the 10th International Conference on Mobile systems, Applications, and Services*. 197–210.

[40] Takuto Yoshida, Junto Nozaki, Kenta Urano, Kei Hiroi, Katsuhiko Kaji, Takuro Yonezawa, and Nobuo Kawaguchi. 2019. Sampling rate dependency in pedestrian walking speed estimation using DualCNN-LSTM. In *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers*.

862–868.
[41] Shun Yoshimi, Kohei Kanagu, Masahiro Mochizuki, Kazuya Murao, and Nobuhiko Nishio. 2015. PDR trajectory estimation using pedestrian-space constraints: real world evaluations. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers (UbiComp/ISWC'15 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 1499–1508.
[42] Ning Yu, Xiaohong Zhan, Shengnan Zhao, Yinfeng Wu, and Renjian Feng. 2018. A precise dead reckoning algorithm based on Bluetooth and multiple sensors. *IEEE Internet of Things Journal* 5 (2018), 336–351.
[43] Baoding Zhou, Qingquan Li, Qingzhou Mao, Wei Tu, Xing Zhang, and Long Chen. 2015. ALIMC: Activity landmark-based indoor mapping via crowdsourcing. *IEEE Transactions on Intelligent Transportation Systems* 16, 5 (2015), 2774–2785.
[44] Yuan Zhuang, Jun Yang, You Li, Longning Qi, and Naser El-Sheimy. 2016. Smartphone-based indoor localization with bluetooth low energy beacons. *Sensors* 16, 5 (2016), 596.

A APPENDIX

Table 26. Overview of the 2nd dataset. The data were collected from five subjects in each environment.

Env.	Floor	Size	Type	# of trajectories	Total data duration
A	2F	22.9m x 43.2m	Office	47	1 hr 28 min 25 sec
B	6F	21.7m x 41.9m	Office	73	1 hr 24 min 15 sec
C	3F	22.2m x 26.2m	Multipurpose	70	1 hr 10 min 31 sec
D	3F	35.4m x 49.2m	Library	74	1 hr 31 min 20 sec
E	5F	20.9m x 46.5m	Office	65	1 hr 22 min 19 sec
F	5F	28.0m x 28.0m	Office	63	1 hr 09 min 50 sec
G	3F	27.6m x 48.0m	Convention center	68	1 hr 22 min 51 sec
H	2F	12.6m x 20.0m	Multipurpose	68	1 hr 29 min 25 sec
I	2F	22.0m x 28.0m	Community center	59	1 hr 17 min 28 sec
J	2F	22.5m x 33.4m	Community center	75	1 hr 12 min 31 sec
K	1F	30.0m x 26.7m	Multipurpose	75	1 hr 22 min 10 sec
L	3F	37.4m x 17.5m	Conference center	74	1 hr 45 min 43 sec
M	2F	28.5m x 17.5m	Workshop	62	1 hr 12 min 11 sec
N	1F	24.0m x 31.5m	Multipurpose	65	1 hr 21 min 49 sec
O	2F	21.8m x 15.2m	Conference center	74	1 hr 25 min 33 sec
P	4F	23.7m x 28.3m	Seminar rooms	75	1 hr 32 min 04 sec
Q	2F	18.6m x 62.0m	Conference center	72	1 hr 21 min 04 sec
R	2F	22.6m x 13.6m	Multipurpose	75	1 hr 38 min 20 sec
S	3F	37.0m x 40.0m	Cultural hall	72	1 hr 35 min 04 sec
T	4F	6.2m x 30.0m	Office	56	1 hr 31 min 52 sec
U	9F	15.3m x 33.0m	Office	45	1 hr 02 min 24 sec