Deep Learning-based PDR Scheme that Fuses Smartphone Sensors and GPS Location Changes

KWAN-SOO KIM (Student Member, IEEE), and YOAN SHIN† (Senior Member, IEEE)
School of Electronic Engineering, Soongsil University, Seoul 06978, Korea
†Corresponding author: Yoan Shin (yashin@ssu.ac.kr)
This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (2020R1A2C2010006).

ABSTRACT Pedestrian dead reckoning (PDR), a sensor-based localization method using a smartphone, combines multi-sensor data from an inertial measurement unit (IMU) generated by the movement of pedestrians and calculates the amount of movement change from a previous location using fusion of sensor data. In this study, we propose a method to improve the efficiency of a deep learning (DL)-based PDR scheme to solve problems associated with the existing PDR method. The proposed DL-PDR scheme solves the movement change of smartphone users as a regression problem by combining IMU and global positioning system (GPS) data. In this paper, we (1) describe the existing PDR methods and problems, describe the proposed DL-PDR scheme and the data collection process of the input sensor data and output GPS used for deep learning, (2) correlate the collected I/O data and conduct preprocessing to make the data suitable for learning, (3) apply data refining and data augmentation methods to provide efficient learning and prevent overfitting, and (4) Verify the performance of the proposed scheme. The localization performance between the proposed scheme and existing methods is compared in various buildings where continuous localization is possible owing to connected indoor/outdoor spaces.

INDEX TERMS smartphone; in-and-outdoor localization; pedestrian dead reckoning; deep learning; supervised learning; inertial measurement unit; global positioning system

I. INTRODUCTION
Research on efficient and accurate localization regarding indoor/outdoor environments using communication technology and multi-sensors in smartphones is being actively conducted [1]–[4]. Broadly, the smartphone-based localization method can be represented as shown in Fig. 1. Among the localization methods that use wireless communication signals, which are external resources in outdoor environments, the user’s current location is estimated using a signal transmitted from a global positioning system (GPS) satellite. In an indoor environment, the users’ location is estimated according to the signal strength of the Wi-Fi access point or Bluetooth low energy beacon. The localization method using external resources causes problems such as path loss, multipath signals, and shadowing, as shown in Fig. 2. Therefore, for accurate localization, a line-of-sight (LoS) condition must be guaranteed [5]–[8]. In addition, pedestrian dead reckoning (PDR) methods with high accuracy are required because it is difficult to use external communication equipment in situations such as during disasters.

PDR methods use data from an inertial measurement unit (IMU), an internal resource of a smartphone, which uses accelerometer, magnetometer, and gyroscope sensor values that change with the user’s movement to calculate the user’s step length and movement orientation to track a user’s location. The step length of smartphone users, one of the main elements of PDR methods, is calculated using accelerometer sensor values and correction factors. The correction factor should be empirically set considering the physical characteristics of the pedestrian (for example, whether the user is tall or short, or their average walking speed is fast or slow), and an incorrect factor setting has a significant impact on stride calculations [9]–[11]. Another major factor, calculating the orientation of movement, can be determined from accelerometer, magnetometer, and gyroscope sensors. Typically, the data from three sensors are fused to determine the orientation.
Current PDR methods have the following problems depending on the surrounding environment, equipment, and sensors.

- Micro electro mechanical systems (MEMS) sensors in smartphones are designed to conserve power rather than provide high accuracy.
- Parameters that can be set according to the walking environment and the physical characteristics of pedestrians are needed to improve localization accuracy.
- Various problems are associated with the measurement method employed by the IMU, such as drift problems in the gyroscope sensor.

To solve the problems associated with the smartphone IMU-based PDR method, after sampling sensor data over some time, the Kalman filter and the particle filter methods are used to smooth outlier data by using statistical indicators of the sampled dataset [12]–[14]. Recently, the characteristics of pedestrians or sensors have been studied using artificial intelligence (AI) methods to smooth noisy sensor data, classify pedestrians’ smartphone positions (e.g., walking, texting, calling, in pocket, etc.), and to calculate pedestrian step lengths. However, most of the current AI-based localization methods must directly set the ground truth used for learning, so the objectivity and accuracy of the data decrease [15]–[17].

In this paper, we compare the method for increasing the efficiency of deep learning (DL)-PDR schemes [18] to solve the problems arising from the aforementioned existing PDR methods and their localization performance. To increase the efficiency of the proposed scheme, we propose data refinement and augmentation methods and discuss overfitting prevention methods for deep neural network (DNN) learning. The proposed DL-PDR scheme uses sensor data that a smartphone user can collect while walking in an outdoor environment as input data and the amount of change in GPS location data between steps as label data for supervised learning. This information is used to predict the amount of change in the users’ movement based on the IMU sensor data by approaching the IMU-based PDR method as a regression problem in which deep learning applies an approximation function to solve a problem.

The remainder of this paper is organized as follows. Section 2 introduces the overall localization process of the PDR method operating in a smartphone and the problems associated with PDR methods. Section 3 discusses the DL-PDR scheme and proposes efficiency improvement methods, and Section 4 introduces the experimental environment and the actual localization results, followed by concluding remarks in Section 5.

II. IMU-BASED PDR & PROBLEMS

This section introduces the overall localization process of the smartphone sensor-based PDR method. It describes the overall content of tracking the movement path by using the sensor values of the smartphone generated while walking. In addition, problems caused by sensor measurement methods and the surrounding environment are described.

A. TRADITIONAL PDR-BASED LOCALIZATION

Various sensors exist in the MEMS of a smartphone, and the sensor group that measures the inertia generated by the movement of a user is called an IMU, which consists of an accelerometer sensor [scale: m/sec²], a magnetometer sensor [scale: µT], and gyroscope sensor [scale: rad/sec]. The factors that affect each sensor are different because the measurement methods used to detect changes in the sensors are different.

The accelerometer is affected by the swing caused by the user rather than by the surrounding environment or walking path. Under this measurement method, the fluctuation of noise measured by the accelerometer is not large, and it has high stability compared to other IMU sensors. When tracking the location of a pedestrian, whether the user is walking or not is determined according to (1) and (2) [19].

\[
A(t) = \sqrt{Acc_x^2(t) + Acc_y^2(t) + Acc_z^2(t) - g}, \quad (1)
\]

\[\{A(t) \geq Acc_{upper}, A(t + \Delta t) \leq Acc_{lower}, 0 < \Delta t < 1\}, \quad (2)\]

\[\lambda_{Kim} = \tau_{Kim} \cdot \left(\sum_{i=0}^{N} \|A(i)\|/N\right)^{2}, \quad (3)\]

\[\lambda_{Weingberg} = \tau_{Weingberg} \cdot (A(\cdot)_{max} - A(\cdot)_{min})^{2}. \quad (4)\]

In the equations, \([Acc]_{i=x,y,z}\) is the accelerometer sensor data measured on each of the three axes, which are used as the basis for the local coordinate system (LCS) set as the reference point for the smartphone, and \(g\) is the acceleration...
due to gravity. The sensitivity involving step detection is determined by how Acc_upper and Acc_lower are set. Existing PDR methods use data collected from \( \Delta t \) intervals, where walking is detected to determine the length and orientation of the user’s step, and the information is stored in the smartphone. In addition, (3) calculates the step length using the average value of the accelerometer values occurring between the time \( \Delta t \) used to detect a step, and (4) calculates the step length of the user using the maximum and minimum values \( A(\cdot)_{\text{max}} \) and \( A(\cdot)_{\text{min}} \) over \( \Delta t \). These two methods are representative methods for calculating the step length [9], [10], and the higher the sampling rate of the sensor values, the higher the accuracy. In the Android operating system used for our experiments, the sampling rate can be selected from among SENSOR_DELAY_NORMAL: 200,000microsec, SENSOR_DELAY_UI: 60,000microsec, SENSOR_DELAY_GAME: 20,000microsec, and SENSOR_DELAY_FASTEST: 0microsec [20]. When using multiple sensors in combination, this parameter should be set considering the synchronization of each sensor.

To convert data obtained from smartphone sensors into a orientation for localization, data measured in the LCS must be converted to a global coordinate system (GCS) using the Euler angle and rotation matrix. GCS consists of \( x \): Pitch\( \sim\phi \), \( y \): Roll\( \sim\theta \), \( z \): Yaw\( \sim\psi \), and in most smartphone-based localization methods, the orientation of the \( z \)-axis, which is the main rotation axis when the smartphone screen faces the sky while lying on a flat surface, is used as the moving orientation. The moving orientation is calculated by combining the accelerometer and magnetometer sensors, or by measuring the magnitude and time of the rotation angle on the gyroscope. First, we discuss the formula combining the accelerometer and magnetometer data used to calculate the orientation at each step \( k \). The formula is as follows [21], [22].

\[
\phi^k = \tan^{-1}\left(\frac{-\text{Acc}_y(t)}{\text{Acc}_z(t)}\right),
\]

\[
\theta^k = \tan^{-1}\left(\frac{-\text{Acc}_x(t)}{\text{Acc}_y(t) \sin(\phi) - \text{Acc}_z(t) \cos(\phi)}\right),
\]

\[
\psi^k = \tan^{-1}\left(\frac{-\text{Mag}_y(t)}{\text{Mag}_z(t)}\right),
\]

\[
\text{Ori}^k_{\text{AM}} = [\phi^k \ \theta^k \ \psi^k]^T.
\]

The accelerometer sensor can calculate only the pitch and roll angle of movement, and the yaw angle is derived by calibrating the magnetometer value with the pitch and roll values obtained from the accelerometer.

Another sensor that can calculate the orientation, the gyroscope, detects the smartphone user’s rotation, and measures the corresponding angular velocity. To detect the users’ rotation, the Coriolis effect at the gyroscope is used to measure the angular velocity and its integral is used to calculate the rotation angle about the smartphone based on its 3-axis coordinate system [23], [24].

\[
F_{\text{Coriolis}}(t)_i = 2m \cdot (\nu_i(t) \cdot \omega_i(t))|_{i=x, y, z},
\]

\[
\omega_i = \int_0^{\Delta t} \omega_i(\tau) d\tau,
\]

\[
\text{Ori}^k_{\text{Gyro}} = \text{Ori}^1_{\text{Gyro}} + [\omega^k_x \ \omega^k_y \ \omega^k_z]^T, \ k \geq 2.
\]

In Equation (9), \( m \) is the mass of an object, \( \nu \) is the velocity, and \( \omega \) is the angular velocity. Because the gyroscope can measure only the rotation angle, the initial orientation is set as shown in (11) using the orientation obtained from the accelerometer and magnetometer sensors. Usually, the orientation of the smartphone is determined by the fusion of \( \text{Ori}^k_{\text{AM}} \) and \( \text{Ori}^k_{\text{Gyro}} \) as in (13) to compensate for the disadvantages of each sensor [25], [26], and the final user’s next estimated location \( \hat{L}_k(x_k, y_k) \) is calculated by (14).

\[
\text{Ori}^k_{\text{Fusion}} = \beta \cdot (\text{Ori}^{k-1}_{\text{Fusion}} + \omega^k) + (1 - \beta) \cdot \text{Ori}^k_{\text{AM}} = [\text{Ori}^k_{F_x} \ \text{Ori}^k_{F_y} \ \text{Ori}^k_{F_z}]^T.
\]

\[
\hat{L}_k = \hat{L}_{k-1} + \lambda^k \cdot [\sin(Ori^k_{F_z}) \ \cos(Ori^k_{F_z})]^T.
\]

B. EXISTING PDR METHOD PROBLEMS

In the PDR method described in Section II-A, various complex problems occur depending on the smartphone users’ unique physical characteristics and walking patterns and the features of each sensor of the IMU. This section describes various difficulties that occur in the traditional smartphone-based PDR method. Finally, parameters should be adjusted considering the physical characteristics of the smartphone user.

First, to calculate the stride length for PDR purposes, it is necessary to set the appropriate correction factors in (3) and (4) according to the user’s physical characteristics and behavioral habits. For example, Figure 3 shows two smartphone users with contrasting physical characteristics. The user’s actual mean step length is determined by the height of the user, but the mean step length determined with the smartphone is based on the magnitude of the acceleration.
Among the sensors in the IMU, the magnetometer is the most affected by the external environment. Magnetometer sensors perceive the highest magnetic force in the environment as the North Magnetic Pole, and the sensor value is distorted when a magnetic material emits a larger magnetic force in the vicinity of the smartphone. This phenomenon is divided into a hard iron effect and a soft iron effect. The hard iron effect is a distortion phenomenon caused by a magnetic field generated from a magnetic material such as the permanent magnet of a speaker present inside a smartphone and is calibrated by measuring and subtracting the offset when designing the smartphone [27]. The soft iron effect is a distortion caused by magnetic fields outside the smartphone such as steel structures, and can be temporarily reduced by a correction algorithm such as shaking 8 characters or as described in [28], but it is difficult to calibrate in situations where the surrounding magnetic fields continue to change, especially in indoor environments. In this situation, the yaw angle $\psi$ in (8) becomes very unstable.

Calculation problems mainly arise with the gyroscope sensors. For example, when the user rotates while moving, the gyroscope sensor detects the rotation based on the Coriolis effect in (9) and the angular velocity is calculated using (10). The problem is that in the gyroscope, which only needs to detect the user’s rotation, residual sensor values are generated owing to any shaking of the device or noise generated by the device itself while walking, and these residual sensor values are included in (10), resulting in cumulative errors.

To date, methods that have been proposed to solve the problems associated with IMU sensors attempt to solve the problems by applying sampling-based filtering methods [12]–[14] or various DNNs [29], [30] to measure the PDR stride and orientation. However, these solutions were not considered in this study because of the high computational costs or the limitations regarding direct measurement of ground-truth data.

III. PROPOSED DL-PDR SCHEME

DL-PDR has been proposed to operate as shown in Figure 5 and to address various problems arising from the traditional PDR method. DL-PDR has been designed for the following purposes.

- Provide a localization model that does not require setting different user characteristics and simply requires data collection and preprocessing.
- Improve localization performance by compensating for problems occurring in existing sensors by using GPS data.
- Objective and simple data generation: the amount of movement change of the smartphone user, which is the ground truth of the label data, is measured with a GPS signal to configure the training data.

In this section, the performance improvement scenario of the proposed DL-PDR method, the correlation between GPS location data and IMU sensor data, the construction and collection process to obtain training data, the data preprocessing...
and refinement process to prevent overfitting, and the model setting and learning method are discussed in detail.

A. LOCALIZATION PERFORMANCE IMPROVEMENT SCENARIO AND DATA COLLECTION PROCESS

In general, a given problem in DL can be approached as a function approximation problem, that is, a regression or classification problem. In this study, the amount of movement change in IMU-based sensors is treated as a regression problem. The performance improvement of the proposed DL-PDR scheme is shown in Figure 6. In the Figure, a walking situation is depicted in an outdoor environment where a smartphone user can receive GPS signals, and when the user moves along the navy (blue) path, the existing PDR method estimates the red path owing to drift or magnetic field variations. However, the location measured by the GPS signal is different from the actual location (indicated by the green path), but the user’s moving orientation and distance are measured similarly to the blue path. The DL-PDR corrects the error by learning the sensor data measured in the red path as relatively normal GPS data affected by a sensor problem. In addition, because there are various routes outdoors, various rotations can occur, as shown in Figure 7, and we propose to improve localization accuracy by storing the sensor data and GPS data together.

GPS is an outdoor localization system developed in the United States among various global navigation satellite systems, and localization can be used by anyone who has equipment that can receive GPS signals in an outdoor environment. The GPS positioning method is divided into the stand-alone (S) type, which processes GPS satellite signals directly from the device, and the assisted (A) type, which is assisted by an Internet network and a GPS signal calculation server. Most smartphones use the A-GPS method in consideration of positioning speed, performance, and battery efficiency. User location using GPS signals is calculated according to the following [31].

\[
\hat{r}_j^k = \{x_{u, j}^k, y_{u, j}^k, z_{u, j}^k\},
\]
\[
r_j^k = \{T_{s, j}^k, [x_{s, j}^k, y_{s, j}^k, z_{s, j}^k]\}, \ldots\}
\]
\[
c \cdot (T_{u, j}^k - T_{s, j}^k) = \sqrt{(x_{s, j}^k - x_{u, j}^k)^2 + (y_{s, j}^k - y_{u, j}^k)^2 + (z_{s, j}^k - z_{u, j}^k)^2},
\]
\[
\hat{r}_j^k = [x_{u, j}^k, y_{u, j}^k, z_{u, j}^k] + c^k.
\]

Here, \(\hat{r}_j^k\) is the user’s location vector whose origin is the center of the earth, and \(r_j^k\) is the signal received from the \(j\)-th satellite at the \(k\)-th step of the smartphone. The received GPS signal includes information such as the location vector \([x_j^k, y_j^k, z_j^k]\) of each satellite and the transmission time \(T_{s, j}^k\). The user’s location is estimated using the information received from four or more GPS satellites, the speed of light \(c\), and receipt time by the user \(T_{u, j}^k\) through the distance
relationship given by (17). An error \( \epsilon^k \) occurs in the estimated user position vector owing to the communication channel environment, satellite clock error, and other factors. Finally, the estimated user location vector is used to calculate latitude and longitude. If the communication channel environment and satellite clock error of the GPS signal received at steps \( k \) and \( k + 1 \) are constant, the error can be assumed to be \( \epsilon^{k+1} \approx \epsilon^k \), and the difference between the user’s location vectors can be expressed as the actual user’s moving orientation and distance, as follows.

\[
\Delta \vec{u}_k = \vec{u}^{k+1} - \vec{u}^k - \epsilon^{k+1} + \epsilon^k = [x_u^{k+1} - x_u^k, y_u^{k+1} - y_u^k, z_u^{k+1} - z_u^k].
\]

(19)

Figure 8 shows a comparison of the actual moving path and the estimated location based on the GPS signal received from the smartphone to confirm the aforementioned assumption, and the storage process of the sensor data and GPS data for steps \( k \) and \( k + 1 \) used for DL-PDR learning. As mentioned in the performance improvement scenario, the green path, which is the measurement result of the actual GPS, is different from the red path, which is the actual moving path, but it can be confirmed that the user’s moving orientation and distance are measured fairly accurately. In addition, the data collection process stores the GPS latitude and longitude data, and the smartphone’s 3-axis sensor data (accelerometer, magnetometer, and gyroscope) in the smartphone database (DB) generated whenever the user walks. Data were collected from Gwangjin-gu, Dongjak-gu, Gwanak-gu in Seoul, and from Yeonje-gu, Busanjin-gu in Busan, Korea.

**B. TRAINING DATA ANALYSIS**

In DL, the output of the model depends on the learning method and data used. In other words, if there is no correlation between the data used for training, it is difficult to obtain the desired result regardless of the complexity of the model or whether a good optimization method is selected. DL methods are classified into supervised learning, unsupervised learning, and reinforcement learning according to the nature of the training data. Supervised learning is used in the DL-PDR model. Supervised learning is a learning method that is possible when the input data are used to train a model to produce output data that are composed of label data. Input data for the proposed DL-PDR method consist of multi-sensor values from the accelerometer, magnetometer, and gyroscope sensors that users can obtain when walking with smartphones. Output data mapped to input data is constructed the location variations computed by GPS satellite signals when walking outdoors.

In this section, we examine the data distributions of the input data and output data and the correlation between these data, and we describe in detail the conversion process into forms suitable for learning through preprocessing. Raw data obtained through the data collection process consists of IMU sensor data measured on the 3-axes of the smartphone that can determine the user’s walking pattern information are used as input data and raw GPS latitude and longitude coordinates consisting of 13 decimal places represent the desired output. These raw data are converted into input data and output data through preprocessing.

\[
\text{Raw data}^k = \{\text{Acc}^k, \text{Mag}^k, \text{Gyro}^k, \text{Ori}^k, \text{Long}^k, \text{Lat}^k\}_{i=x, y, z}.
\]

(20)

1) Output Data Configuration

In supervised learning, the weight of each hidden layer is updated according to label data to solve a problem (classification or regression) [32]. In particular, data generation costs and objectivity are determined by how the ground-truth of label data is measured. DL-PDR stores label data \( Y^k \) as a GPS signal received whenever a smartphone user walks outdoors (where GPS satellite signals can be received), and the location of each step is stored. The stored longitude and latitude position data, in degrees [°], minutes [′], and seconds...
\[ 1'_\text{Long} = 2\pi R \cos(\phi)/360^2, \quad 1''_\text{Lat} = 2\pi R/360^2, \quad (21) \]
\[ \Delta \text{Long}^k = (\text{Long}^{k+1} - \text{Long}^k) \cdot 1'_\text{Long}, \quad (22) \]
\[ \Delta \text{Lat}^k = (\text{Lat}^{k+1} - \text{Lat}^k) \cdot 1''_\text{Lat}, \quad (23) \]
\[ y^k = \{\Delta \text{Long}^k, \Delta \text{Lat}^k\}. \quad (24) \]

where \( R \) is a radius of approximately 6,378 km of the earth, and \( \phi \) is the latitude degree at the measured GPS location. In (21), the interval of the latitude is not related to the location, but in the case of longitude, the interval changes according to the user’s location and must be multiplied by \( \cos(\phi) \). This ground-truth measurement method produces label data \( y^k \), which are calculated as objective GPS location signals. Data can be generated by simply walking outside, allowing data collection in daily life and inexpensive data generation. In general, the degrees and minutes of the latitude and longitude measured by GPS at the \( k \) and \( k + 1 \) steps do not change, and only the second changes, so the output data \( y \) consists only of variance of seconds.

2) Input Data Characteristics and Configuration

To obtain a generalized DL model using data containing considerable noise, it is necessary to simplify the model or reduce the number of input features to reduce the complexity of the network [33]. As mentioned in the Introduction, the sensor data used as input data for DL-PDR represent a performance limitation of MEMS, designed to consume low power, and sensor values contain significant noise, so proper features need to be combined. In addition, the input data should be composed of components capable of expressing the movement of the smartphone user in two dimensions (14) and should include data generated when the smartphone user walks as well as the pedestrian’s characteristics, as shown in Figure 3. Finally, because the output data are calculated as the difference between the measured locations at steps \( k \) and \( k + 1 \), the input characteristics should also characterize the values at steps \( k \) and \( k + 1 \) appropriately.

The input data consist of three components capable of expressing a step length, a rotation amount, and a moving orientation, and a normalization process is required for the different physical quantities of each sensor. First, accelerometer sensor data measure the amount of acceleration generated according to the movement speed of the smartphone user, and (3) and (4) use it to calculate the users’ step length. In other words, the magnitude of the sensor value of the accelerometer is a major factor that enables the estimation of the users’ step length, and in DL-PDR, it is used as input data related to the step length using the intermediate value of the 3-axis accelerometer for each set of \( k \) and \( k + 1 \) steps. Second, the gyroscope sensor returns the angular speed according to the rotation of the smartphone user and calculates the amount of rotation, as shown in (10). Finally, the moving orientation in (13) is used to preprocess the input data as follows:

\[ \frac{\text{Acc}_{xz}^k}{\text{Acc}_{yz}^k + \text{Acc}_{xz}^k} - 2 - \{\text{Acc}_{xy}^2\} \mu, \quad (25) \]
\[ N \text{ Acc}^k = \frac{\{\text{Acc}_{xz}^k + \text{Acc}_{zy}^k + \text{Acc}_{xy}^k\}}{\{\text{Acc}_{xy}^2\}} \mu, \quad (26) \]
\[ \text{Gyro}_{xyz}^k = \frac{\{\text{Gyro}_{xyz}^k + \text{Gyro}_{xyz}^k \}}{\{\text{Gyro}_{xyz}^2\}}, \quad (27) \]
\[ N \text{ Gyro}^k = \frac{\{\text{Gyro}_{xyz}^k \}}{\{\text{Gyro}_{xyz}^2\}} \mu, \quad (28) \]
\[ \Delta \text{ Ori}^k = \|\text{Ori}^{k+1} - \text{Ori}^k\|, \quad (29) \]
\[ N \text{ Ori}^k = \{\begin{cases} \text{Case 1} : \|\text{Ori}^{k+1} + 1 - \text{Ori}^k\|/2 \cdot 360, \\
\text{Case 2} : \|\text{Ori}^{k+1} + 1 - \text{Ori}^k\|/360, \\
\text{if} \Delta \text{ Ori}^{k+1} > 3 \cdot \{\Delta \text{ Ori}^k\}_\sigma, \\
\text{if} \Delta \text{ Ori}^{k+1} > 3 \cdot \{\Delta \text{ Ori}^k\}_\sigma, \end{cases} \quad (30) \]

\[
\begin{array}{c|c|c|c}
\text{Case} & \text{Acc}_{xyz} & \text{Gyro}_{xyz} & \Delta \text{ Ori}^k \\
\hline
\mu & 12.14 & -0.22 & 2.95 \\
\sigma & 0.81 & 0.21 & 4.91 \\
\end{array}
\]

Here, \{\cdot\}_\mu and \{\cdot\}_\sigma are the mean and standard deviation values of the collected dataset, respectively. Accelerometer and gyroscope sensors have average values according to user moving patterns and are preprocessed in the standardization method in consideration of these characteristics, whereas the orientation of movement is normalized to a maximum size of 360 because it occurs randomly in the range of \( 0 \sim 360^\circ \). At this time, \( \{\Delta \text{ Ori}^k\}_\sigma \), used as a reference, was assumed and used as an error caused by noise in the surrounding environment or sensor when walking in a straight line, and if the difference in the orientation calculated at step \( k \) and \( k + 1 \) was \( 3 \cdot \{\Delta \text{ Ori}^k\}_\sigma \) or more, it was used as an indicator to determine whether the actual movement orientation of the user has changed. The orientation data are preprocessed as shown in (30). The average values and standard deviations obtained from approximately 110,000 data points collected are shown in Table 1.

The input data configured as described above are represented by a two-dimensional polar coordinate system of the user’s movement change, and the output data are represented by a two-dimensional orthogonal coordinate system to confirm the relationship between the input and output data.

C. DATA REFINING AND AUGMENTATION

In Section III-B, the features were compressed by preprocessing the learning data for the generalized performance of the DL model. However, even when the features are compressed through the preprocessing process, severe noise may still...
remain in the data, and refining such data is used as a means of improving the learning results of the model and preventing overfitting. This section describes the refining criteria for each feature and corresponding changes in the learning data, and proposes data augmentation methods to improve learning results and localization performance when the quantity of data is limited.

1) Data Refinement

Data refinement is performed according to the following criteria.

1) GPS signals are attenuated owing to the problem shown in Figure 2, and the assumption $\epsilon_k \approx \epsilon_{k+1}$, is not well-established. We checked this problem as shown in Figure 9. Figure 9a shows the distribution of step lengths measured by GPS according to each smartphone IMU azimuth and the average and standard deviation values for step lengths. When such a problem occurs, the calculated step length differs from the actual step length of the smartphone user. To reduce this effect, only the data corresponding to the ±0.5 standard difference ($\sigma=24$cm) based on the mean ($\mu=77$cm) were used as learning data, and the refinement results are shown in Figure 9b.

2) When the smartphone is motionless on a flat surface, the magnitude of the measured acceleration is measured to be approximately $9.8 \text{m/sec}^2$, which is the Earth’s gravitational acceleration. In general, data lower than $9.8 \text{m/sec}^2$ are excluded from learning because the acceleration magnitude measured during movement is higher than the Earth’s gravitational acceleration.

3) Figure 10 shows a comparison of the orientation calculated by the smartphone IMU $N Ori_k \cdot 360 \text{ [scale: color]}$ and the orientation $Ori_{GPS}^k$ calculated by the GPS location change amount. In Figure 10a, it can be seen that many outliers of IMU data are included, and if the data are included in the training data, they can affect the positioning performance. To remove such data, values were used as learning data only if the difference between $N Ori_k \cdot 360$ and $Ori_{GPS}^k$ was less than $3 \cdot \{\Delta Ori_F\}_\sigma$, which allows $99.7\%$
of the data involving general orientation errors while excluding most of the outliers. $O_{GPS}^k$ is calculated as follows, and the data from which outliers are removed are shown in Figure 10b.

$$O_{GPS}^k = \tan^{-1}\left(\frac{\Delta Lat^k}{\Delta Long^k}\right) \cdot \frac{180}{\pi}.$$  (32)

From a total of 110,150 data values collected for learning, if data are refined according to the above criteria, 54,987 values are removed under criteria 1, 13,308 under criteria 2, and 23,360 under criteria 3, such that 18,495 values remain.

2) Data Augmentation

Data augmentation methods are used to enhance the actual performance of the model or reduce deteriorating learning performance caused by insufficient learning data. For image data, data are augmented by rotating the image or cropping a certain area, and data that are difficult to simply rotate or crop are augmented by learning the distribution of data using a generative adversarial network (GAN) [34], [35]. To obtain a good generator using GAN, a sufficient (large) amount of data is required to learn the distribution of that dataset.

Such data augmentation using a GAN model is not suitable because the augmentation method used by DL-PDR requires methods to solve problems using low amounts of data.

The data augmentation method used in the DL-PDR scheme assumes a case in which the data are insufficient, as shown in Figure 11. Figure 11a shows a case in which there is a lack of data in the overall orientation, and Figure 11b shows data that are collected only on fixed routes, such as when commuting; therefore, the data quantity and orientation information are insufficient. In this case, the data augmentation method shown in Figure 12 can be used to improve the learning results and the performance of the model. The augmentation method is divided into data movement and duplication steps, and the following augmentation process is described from the perspective of Data1 in Figure 12.

1) The amount of change in the latitude and longitude of GPS values can be expressed in two-dimensional orthogonal coordinate system, and the position of the data can be rotated using a rotation matrix $R_i$. When the rotation range (RRA) $\theta_{RRA}$ of the data is determined, the number of rotation regions is determined, and the orientation value $N \cdot O_{Data1} \cdot 360$ and the GPS change amount rotate together as follows.

$$K = \frac{360}{\theta_{RRA}}, \quad 0 < i \leq K \ (i \in \mathbb{N}),$$  (33)

$$R_i = \begin{bmatrix} \cos(\theta_{RRA} \cdot i) & -\sin(\theta_{RRA} \cdot i) \\ \sin(\theta_{RRA} \cdot i) & \cos(\theta_{RRA} \cdot i) \end{bmatrix},$$  (34)

$$Y^R_{Data1} = [\Delta Long_{Data1} \cdot \Delta Lat_{Data1}] \cdot R_i,$$  (35)

$$N \cdot O_{Data1} = (N \cdot O_{Data1} \cdot 360 + i \cdot \theta_{RRA}) \mod 360.$$  (36)

2) Input features $N \cdot Acc$ and $N \cdot Gyro$ can be viewed as data representing events that can occur regardless of orientation, so the remaining input components $N \cdot Acc_{Data1}$ and $N \cdot Gyro_{Data1}$ of the rotated data are
D. MODEL TRAINING AND LEARNING RESULTS

DL solves the problem by first defining the model and a loss function to be used and then approximating the function using the defined model to solve the regression or classification problem. DL-PDR solves IMU-based localization as a regression problem. The model used in DL-PDR is a multi-layer perceptron (MLP) model. In the DL-PDR localization schemes, attempting to use time-series data is inadvisable because a large number of data sections are removed during the refining process to eliminate severe data noise, and therefore a recurrent neural network (RNN) model was not considered as a useful model because it is not suitable for learning discontinuous data.

1) DL-PDR Model Overview and Training Parameter Setting

An overview of the DL-PDR MLP model is presented in Figure 14. The MLP model uses sensor data preprocessed in the offline stage as input data, and the label data for supervised learning uses the amount of change in latitude and longitude to train the model. In the online stage, the input preprocessed sensor data are used to predict the amount of change in the location of the smartphone user. To identify the appropriate network size of the DL-PDR model and compare the generalization performance according to complexity, we used two models (simple and complex), as shown in Table 2, and created models considering appropriate parameters for preprocessed sensor data. Because the input characteristics $N_{Acc}^k$ and $N_{Gyro}^k$ of DL-PDR have standardized values, the values are distributed in the negative and positive regions. The exponential linear unit (Elu) activation function used in the MLP model allows the MLP model to have nonlinearity, and unlike the rectified linear unit (ReLu) activation function, it is possible to update all weights because nodes do not die and can even differentiate negative input values [36]. The root mean square error (RMSE) as the loss function is used to solve the regression problem by updating weights to reduce the difference between the model’s predicted value and the desired label value [37]. The optimization adaptive moment estimation (Adam) algorithm estimates the gradient’s 1st moment $m_t$ and the 2nd moment $v_t$ as learning progresses to update weights and use the learning rate decay parameters provided by the TensorFlow platform to prevent overfitting [38]. The activation function $Elu(\cdot)$, loss function $L(\cdot)$, and the gradient $\theta_t$ for optimizing the objective function $J(\cdot)$ are substituted as input components of Data1 as follows.

\begin{align}
N_{Acc}^{R_{Data1}} &= N_{Acc}^{Data1}, \\
N_{Gyro}^{R_{Data1}} &= N_{Gyro}^{Data1}.
\end{align}

Figure 13 is an example of data augmentation and shows the results of 6,000 data augmentations, a K-times ($K = 12$) increase from 500 existing data using the augmentation algorithm after setting the rotational area to $30^\circ$.

### TABLE 2. Parameters used in DL-PDR model.

| Model Parameters | Simple Model | Complex Model |
|------------------|--------------|---------------|
| Number of hidden layer | 2 | 7 |
| Number of weights | 1,330 | 11,154 |
| Activation function | Elu | |
| Loss function | RMSE | |
| Optimizer | Adam | |
| Learning rate | 0.001 | |
| Epochs | 100 | |
TABLE 3. Learning results according to the type of data used.

| Model Type          | Augmentation Case | Orientation Range [°] | Rotation Range [°] | Augmentation Data | Final Loss (RMSE) [cm] |
|---------------------|-------------------|-----------------------|--------------------|------------------|------------------------|
| Non Refine          | Simple            | ·                     | ·                  | Non Augmentation, Training Data: 110,150 | 67.23                  |
| Original Data       | Complex           | ·                     | ·                  | Non Augmentation, Training Data: 18,495  | 8.37                   |
| Refine              | Simple            | ·                     | ·                  | Non Augmentation, Training Data: 100     | 44.70                  |
| Original Data       | Complex           | ·                     | ·                  | Non Augmentation, Training Data: 1000     | 10.83                  |
| Refine              | Total Data 100    | Case 1 0~360          | ·                  | Non Augmentation, Training Data: 500      | 13.12                  |
| Total Data 500      | Simple            | ·                     | ·                  | Non Augmentation, Training Data: 18,000   | 8.43                   |
| Limited Ori and     | Simple            | 90~120, 270~300       | ·                  | Non Augmentation, Training Data: 1000     | 10.44                  |
| Refine Data 500     | Complex           | 30                    | 500⇒6,000          | 8.25              |
| Limited Ori and     | Simple            | 90~120, 270~300       | ·                  | Non Augmentation, Training Data: 1000     | 7.88                   |
| Refine Data 1000    | Complex           | 30                    | 1,000⇒6,000        | 8.52              |
| Limited Ori and     | Simple            | 90~120, 270~300       | ·                  | Non Augmentation, Training Data: 2,000    | 8.97                   |
| Refine Data 2000    | Complex           | 30                    | 2,000⇒24,000       | 8.44              |
| Limited Ori 10       | Simple            | 100~110, 280~290      | ·                  | Non Augmentation, Training Data: 1,898    | 8.38                   |
| and Refine Data     | Complex           | 10                    | 1,898⇒68,328       | 8.07              |
| Limited Ori 20       | Simple            | 95~115, 275~295       | ·                  | Non Augmentation, Training Data: 3,340    | 9.05                   |
| and Refine Data     | Complex           | 20                    | 3,340⇒60,120       | 8.47              |

defined as follows.

\[ Elu(x) = \begin{cases} 
  a \cdot (e^x - 1), & \text{if } x < 0 \\
  x, & \text{if } x \geq 0
\end{cases} \]

\[ L(y, \hat{y}) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}, \quad (40) \]

\[ g_t = \nabla_{\theta_t} J(\theta_t), \quad (41) \]

\[ m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t, \quad (42) \]

\[ v_t = \beta_1 v_{t-1} + (1 - \beta_2) g_t^2, \quad (43) \]

\[ \hat{m}_t = \frac{m_t}{1 - \beta_1}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2}, \quad (44) \]

\[ \theta_{t+1} = \theta_t - lr \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}, \quad (45) \]

Here, \( lr \) is the learning rate, and the model was trained by setting \( \beta_1, \beta_2, \) and \( \epsilon \) of the optimization algorithm to 0.9, 0.999, and \( 10^{-8} \), respectively. Finally, the estimated location using the trained model \( M(\cdot) \) is determined as follows.

\[ \hat{L}_k(x_k, y_k) = \hat{L}_{k-1}(x_{k-1}, y_{k-1}) + M(X^k) \]

\[ = \hat{L}_{k-1}(x_{k-1}, y_{k-1}) + [\Delta \text{Long}^k \Delta \text{Lat}^{k,T}] \]

2) Training Results

DL-PDR training was conducted by randomly selecting 10% of the total data used for learning as the model’s validation dataset. Weights were stored whenever the loss function value for the validation data decreased using the checkpoint method provided by TensorFlow. Table 3 shows the results of the validation using different models, data quantities, and preprocessing and non preprocessing data. Based on the results presented in the table, the following can be confirmed.

- Owing to excessive noise in the unrefined data, there was a significant difference in learning outcomes compared to the model using the refined data.
- In situations where the number of data was insufficient (less than 500), the learning results using data augmentation methods showed significant differences in loss function values for the simple model compared to the complex model.
- In Fig. 11, the difference in the values of the loss function was low in the model trained before and after the augmentation of the data accumulated in the orientation-limited case (Case 2). This learning results is judged the amount of data is sufficient for learning just that limited orientation. However, in this case, it is necessary to compare the actual performance by checking the localization results of the augmented data application model and the nonaugmented data application model.
IV. EXPERIMENTS AND RESULTS

To implement the proposed DL-PDR and augmentation methods, a data collection application was built on the Android OS, and the entire experiment was conducted using a Samsung Galaxy S8 smartphone. In addition, positioning model creation, data preprocessing and augmentation, and model training were conducted using Python 3 and Google’s open framework TensorFlow, which is easy to transplant to smartphones, and the model was trained using NVIDIA’s GTX 3090 GPU. To verify the performance of the proposed localization model (sensor-based IMU), the experiment was conducted in various environments that allowed continuous indoor and outdoor localization. In addition, the mean square error (MSE) and accuracy rate (AR) were used as performance indicators to evaluate the accuracy of the localization method. The MSE returns the distance between the actual moving point \( L_k(x_k, y_k) \) and the predicted localization point \( \hat{L}_k(x_k, y_k) \) to calculate the amount of error that occurs on average, and the AR returns the localization accuracy by calculating whether there is a prediction point within the error boundary \( \zeta \), set based on \( L_k \), as follows [39].

\[
MSE(L_k, \hat{L}_k) = \frac{1}{N} \sum_{k=1}^{N} (L_k - \hat{L}_k)^2, \quad (47)
\]
\[
\delta_k = \sqrt{(L_k - \hat{L}_k)^2}, \quad 0 < k \leq N, \quad (48)
\]
\[
AR = \frac{\#\{\delta_k | \delta_k \leq \zeta\}}{N} \cdot 100\%.
\]

A. SHORT PATH EXPERIMENT: HYUNGNAM MEMORIAL ENGINEERING BUILDING 2ND FLOOR

The first experiment was conducted on the second floor of the Hyungnam Memorial Engineering Building of Soongsil University. There is an entrance to the building on the second floor of the Hyungnam Engineering Building, so the interior and exterior are connected, and consists of a large hall containing various pieces of furniture. Therefore, various disturbances are present in the area. Approximately 110 to 120 steps (103m) were taken to obtain sensor data during the experiment, and the performance of the model and augmentation method for DL-PDR were evaluated by comparing the actual path to the model prediction using this test data.

Figure 15 shows the results of localization using sensor data generated according to entering the actual Hyungnam Engineering Building from outside. The results lead to the following observations for models with and without enhancement in Table 3 with respect to the conventional PDR method and the proposed DL-PDR model.

- In the conventional PDR method, differences from the actual path occurred (orange path) owing to the unstable orientation caused by the unstable geomagnetic sensor (navy path) in the indoor space when the correction coefficient \( \tau \) of (3) was 0.5.
- The localization result (red path) differed from the actual path owing to the noisy learning data in the model without refinement, and in contrast, in the learning model using refined data, accurate localization results were verified. Additionally, it was confirmed that the DL-PDR estimated the step length and orientation well...
TABLE 4. First experiment: localization results according to the data used.

| Augmentation        | Model Type | AR [≤0.1m] | AR [≤1m] | AR [≤1.5m] | AR [≤2m] | AR [≥2.5m] | MSE [m] |
|---------------------|------------|------------|----------|------------|----------|------------|---------|
| Non Refine Original Data | No         | Simple     | 7.4%     | 13.2%      | 18.2%    | 22.3%      | 26.4%   | 128.2   |
|                     |            | Complex    | 3.3%     | 24.8%      | 42.1%    | 43.0%      | 47.1%   | 311.1   |
| Refine Original Data | No         | Simple     | 41.3%    | 65.3%      | 76.9%    | 87.6%      | 99.2%   | 13.0    |
|                     |            | Complex    | 34.7%    | 69.9%      | 87.6%    | 98.3%      | 100%    | 8.5     |
| Refine Total Data 100 | No         | Simple     | 2.5%     | 5.0%       | 7.4%     | 10.7%      | 13.2%   | 2062.2  |
|                     |            | Complex    | 8.3%     | 13.2%      | 16.5%    | 20.7%      | 50.4%   | 67.5    |
|                     | Yes        | Simple     | 93.9%    | 5.1%       | 43.0%    | 82.6%      | 100.0%  | 273.4   |
|                     |            | Complex    | 19.0%    | 44.6%      | 65.3%    | 68.6%      | 123.6%  | 510.9   |
| Refine Total Data 500 | No         | Simple     | 6.6%     | 12.4%      | 19.0%    | 26.4%      | 38.8%   | 467.6   |
|                     |            | Complex    | 10.7%    | 19.0%      | 52.9%    | 63.6%      | 75.2%   | 42.2    |
|                     | Yes        | Simple     | 27.3%    | 66.9%      | 97.4%    | 100%       | 100%    | 8.0     |
| Limited Ori and Refine Data 500 | No        | Simple     | 9.9%     | 14.9%      | 19.0%    | 22.3%      | 23.1%   | 1973.1  |
|                     |            | Complex    | 19.8%    | 24.0%      | 24.8%    | 25.6%      | 26.3%   | 175.6   |
|                     | Yes        | Simple     | 35.1%    | 43.0%      | 56.2%    | 66.1%      | 75.2%   | 24.3    |
|                     |            | Complex    | 42.1%    | 47.9%      | 64.5%    | 73.6%      | 84.3%   | 24.7    |
| Limited Ori and Refine Data 1000 | No        | Simple     | 11.6%    | 18.2%      | 22.3%    | 24.0%      | 24.8%   | 1205.4  |
|                     |            | Complex    | 10.0%    | 24.8%      | 25.6%    | 26.4%      | 26.3%   | 1575.2  |
|                     | Yes        | Simple     | 14.0%    | 38.0%      | 51.2%    | 66.9%      | 78.5%   | 31.2    |
|                     |            | Complex    | 23.3%    | 43.8%      | 68.6%    | 80.0%      | 100%    | 17.5    |
| Limited Ori and Refine Data 2000 | No        | Simple     | 24.8%    | 25.6%      | 27.3%    | 28.1%      | 28.9%   | 572.9   |
|                     |            | Complex    | 24.3%    | 16.3%      | 24.0%    | 24.8%      | 25.0%   | 173.8   |
|                     | Yes        | Simple     | 29.8%    | 43.0%      | 64.5%    | 87.6%      | 100%    | 17.3    |
|                     |            | Complex    | 42.1%    | 48.8%      | 64.5%    | 71.9%      | 85.1%   | 23.3    |
| Limited Ori 10 and Refined Data | No        | Simple     | 9.1%     | 14.0%      | 18.2%    | 21.5%      | 23.1%   | 1880.7  |
|                     |            | Complex    | 14.0%    | 21.5%      | 23.1%    | 24.0%      | 24.8%   | 1772.9  |
|                     | Yes        | Simple     | 28.9%    | 42.1%      | 57.0%    | 68.6%      | 83.5%   | 25.9    |
|                     |            | Complex    | 10.9%    | 54.3%      | 98.3%    | 100%       | 100%    | 91.1    |
| Limited Ori 20 and Refined Data | No        | Simple     | 24.8%    | 25.6%      | 26.4%    | 28.1%      | 28.9%   | 590.8   |
|                     |            | Complex    | 21.5%    | 24.8%      | 25.6%    | 26.4%      | 27.3%   | 1970.2  |
|                     | Yes        | Simple     | 40.5%    | 52.9%      | 66.1%    | 80.2%      | 98.3%   | 17.9    |
|                     |            | Complex    | 43.0%    | 48.8%      | 69.4%    | 78.5%      | 86.0%   | 22.7    |
| Traditional PDR Acc & Mag | -          | 3.3%       | 5.0%     | 6.6%       | 7.4%     | 9.1%       | 336.9   |
| Traditional PDR Fusion Orientation | -          | 19.0%      | 40.5%    | 46.3%      | 52.1%    | 59.5%      | 51.0    |

without having to estimate and set different parameters separately, unlike conventional PDR methods.

- In the learning results in Table 3, the localization results of the model that conducted learning with nonaugmented data (i.e., Limited Ori and Refine Data 500, 1000, 2000) is the yellow path. The yellow path is only learned at a limited orientation; therefore, the localization result is accurate only for the orientation included during learning, and in contrast, relatively accurate localization results can be confirmed in the model that learned using augmented data.

- The model that learned using limited data (i.e., Refine Total Data 100, 500) underestimated the stride owing to lack of data, so there was a significant difference from the actual endpoint, and in contrast use of augmented data resulted in relatively accurate localization results.

We confirmed the performance of the proposed DL-PDR model and the model applying the proposed augmented method through experiments on the second floor of the Hyungnam Engineering Building, and the MSE and AR of the models used are shown in Table 4.

B. LONG PATH EXPERIMENT

The experiment on the second floor of the Hyungnam Engineering Building confirmed the accurate performance of DL-PDR using sensor data generated while walking a simple, short path. We conducted a second localization experiment using sensor data obtained for 729 steps (approximately 560 m) that occurred while walking around buildings, as shown in Figure 16 to confirm the stability and accuracy of each model identified in Table 3 on a more complex and longer path. Figure 16, shows the localization results for the long path, indicating that the sky-blue and navy paths, which are the localization results of the existing PDR method, differed significantly from the actual walking path owing to sensor problems. However, for DL-PDR trained using the original data, localization results similar to the actual walking path could be confirmed, and the localization results for the second experiment are shown in Table 5. In the localization results, only the models using augmented data that achieved high performance in the first experiment were compared with the original model. In addition to the MSE and AR indicators, to compare the long path localization performance the same starting point \( L_1(x_1, y_1) \) and ending point \( L_N(x_N, y_N) \) used in the second experiment, and the distance between the
two points was added as an assistance accuracy indicator as follows.

\[ D_{SE} = \sqrt{(L_1 - L_N)^2}. \]  

From Table 5, which indicates the long path localization accuracy, we observed the following.

- The difference in localization performance between the trained model using the original data and the model using refined data was more clearly observed, and in particular, the complex model learning refined data obtained an AR value within the error boundary \( \zeta = 5 \) m of 99.3\%, and a \( D_{SE} \) of 1.1m. It is judged that the model using more parameters can more accurately predict the amount of change in the user’s movement.
- Most models using augmented data derived more accurate localization results than conventional PDR models, and complex models were able to obtain better localization results than relatively simple models. However, compared to the model referred to as (Using Refined Original Data) using all refined data, there was a significant difference in accuracy compared to the results between the simple path and complex path, which highlights the limitations of models using limited data.

V. CONCLUSIONS AND DISCUSSION

The localization performance was degraded owing to noise in sensor data after setting the orientation angle error and user stride correction coefficient when using existing PDR methods. To solve this problem, in this study, the model was trained using data that combines smartphone sensor data with GPS location changes as MLP learning data.

The label data needed for supervised learning is designed as a system that can be obtained by simply walking outside, where GPS signals can be received, reducing the costs of generating learning data. In addition, a network setting to improve the localization performance of the model and prevent overfitting in the learning phase, a refining process to improve the quality of the training data, and a data augmentation method to achieve efficient performance with fewer data were
proposed. To validate this approach, localization experiments were conducted in two experimental environments, and the orientation and step length prediction performance of the proposed scheme was significantly more accurate than existing PDR methods, as confirmed by the experimental results.

In future work, to improve the performance of DL-PDR, we will conduct additional studies on refining and augmentation methods and expand the PDR model, which currently derives two-dimensional positioning results, to three dimensions (adding an elevation change prediction).

**CONFLICT OF INTEREST STATEMENT**
The authors declare no conflict of interest.

**ACKNOWLEDGMENT**
(Kwan-soo Kim, and Yoan Shin contributed equally to this work.)

**REFERENCES**

[1] S. Subedi, D.-H. Kim, B.-H. Kim, and J.-Y. Pyun, “Improved Smartphone-Based Indoor Localization System Using Lightweight Fingerprinting and Inertial Sensors,” *IEEE Access*, vol. 9, pp. 53343 - 53357, Apr. 2021.

[2] R. Elbakly, H. Aly, and M. Youssif, “TrueStory: Accurate and robust RF-based floor estimation for challenging indoor environments,” *IEEE Sensors Jour.*, vol. 18, no. 24, pp. 10115-10124, Dec. 2018.

[3] P. Li, X. Yang, Y. Yin, S. Gao, and Q. Niu, “Smartphone-Based Indoor Localization With Integrated Fingerprint Signal,” *IEEE Access*, vol. 8, pp. 33178 - 33187, Feb. 2020.

[4] T. L. N. Nguyen, T. D. Vy, K.-S. Kim, C. Lin, and Y. Shin, “Smartphone-Based Indoor Tracking in Multiple-Floor Scenarios,” *IEEE Access*, 2021, to appear.

[5] H. Liu, H. Darabi, P. Banerjee, and J. Liu, “Survey of wireless indoor positioning techniques and systems,” *IEEE Trans. Syst. Man Cybern. C Appl. Rev.*, vol. 37, no. 6, pp. 1067-1080, Nov. 2007.

[6] N. Kim and Y. Kim, “A novel RSS-ratio position estimation scheme for Wi-Fi networks,” *Proc. ICEEM 2015*, Phuket, Thailand, pp. 61-64, Dec. 2015.

[7] J. Zuo, S. Liu, H. Xia, and Y. Qiao, “Multi-phase fingerprint map based on interpolation for indoor localization using iBeacons,” *IEEE Sensors Jour.*, vol. 18, no. 8, pp. 3351-3359, Apr. 2018.

[8] A. Zayets, C. Gennert, and E. Steinbach, “High-Precision Multipath-Based Indoor Localization Scheme With User Privacy Protection for Dynamic NLoS Environments,” *IEEE Access*, vol. 9, pp. 116033-116049, Aug. 2021.

[9] J. W. Kim, H. J. Jang, D. H. Hwang, and C. Park, “A step, stride and heading determination for the pedestrian navigation system,” *Jour. Global Positioning Syst.*, vol. 3, no. 1-2, pp. 273-279, 2004.

[10] H. Weinberg, “AN-602: Using the ADXL202 in pedestrian and personal navigation applications,” *Analog Devices Tech. Rep.*, 2002. [Online]. Available: http://www.analog.com/static/imported-files/application_notes/513772624AN602.pdf.

[11] J. Scarlet, “Enhancing the performance of pedometers using a single accelerometer,” *Analog Devices Tech. Rep.*, 2007. [Online]. Available: http://www.sgbotic.com/products/appnotes/sensors/47970549270982182018.pdf.

[12] L. Pei, D. Liu, D. Zou, R. L. F. Choy, Y. Chen, and Z. He, “Optimal heading estimation based multidimensional particle filter for pedestrian indoor positioning,” *IEEE Access*, vol. 6, pp. 49705-49720, Sept. 2018.

[13] F. Zampella, M. Khider, P. Robertson, and A. Jimenez, “Unscented Kalman filter and magnetic angular rate update (MARU) for an improved pedestrian dead-reckoning,” *Proc. IEEE/ION PLANES 2012*, Myrtle Beach, USA, pp. 129-139, Apr. 2012.

[14] H. Li, H. Guo, Y. Qi, L. Deng, and M. Yu, “Research on multi-sensor pedestrian dead reckoning method with UKF algorithm,” *Measurement*, vol. 169, pp. 108524, Feb. 2021.

[15] Y. Wang, Z. Li, J. Gau, and L. Zhao, “Deep neural network-based Wi-Fi/pedestrian dead reckoning indoor positioning system using adaptive robust factor graph model,” *IET Radar, Sonar & Nav.*, vol. 14, no. 1, pp. 30-41, Jan. 2019.

[16] C. Chen, P. Zhao, C. X. Lu, W. Wang, A. Markham, and N. Trigoni, “Deep-learning-based pedestrian inertial navigation: Methods, data set, and on-device inference,” *IEEE IoT Jour.*, vol. 7, no. 5, pp. 4431-4441, May 2018.

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

[18] K.-S. Kim and Y. Shin, “Deep learning-based PDR scheme for predicting movement changes for smartphone users,” *Jour. KICS*, vol. 46, no. 11, pp. 36-47, Nov. 2021, to appear.

[19] S. Traini, L. Sciculo, A. Trotta, and M. Di Felice, “Practical indoor localization via smartphone sensor data fusion techniques: A performance study” *Proc. IEEE CCNC 2019*, pp. 1-7, Las Vegas, USA, Feb. 2019.

[20] Android Developer, “Sensors overview,” available at https://developer.
Kim et al.: Deep Learning-based PDR Scheme that Fuses Smartphone Sensors and GPS Location Changes

android.com/guide/topics/sensors/sensors_overview  [Accessed: 29-Aug-2021]

[21] G. Milette and A. Stroud (Eds.), Professional Android Sensor Programming, Ch. 6, John Wiley & Sons, 2012.

[22] J. W. Lee, Android Sensor Story, Chs. 4-6, Freelec, 2014.

[23] A. D. King, “Inertial navigation – Forty years of evolution,” GEC Review, vol. 13, no. 3, pp. 140-149, 1998.

[24] V. Marotto, A. Serra, D. Carbouni, M. Sole, T. Dessi, and A. Manchini, “Orientation analysis through a gyroscope sensor for indoor navigation systems,” Proc. IFSA Sensordevice 2013, Barcelona, Spain, pp. 85-90, Aug. 2013.

[25] A. -J. Baerveldt and R. Klang, “A low-cost and low-weight attitude estimation system for an autonomous helicopter,” Proc. IEEE IECS 1997, Budapest, Hungary, pp. 391-395, Sept. 1997.

[26] T. S. Yoo, S. K. Hong, H. M. Yoon, and S. Park, “Gain-scheduled complementary filter design for a MEMS based attitude and heading reference system,” Sensors, vol. 11, no. 4, pp. 3816-3830, Mar. 2011.

[27] L. Árvai and S. Homolya, “Filtering methods for indoor magnetic compass,” Proc. ICC 2020, Tatras, Slovakia, pp. 1-5, Oct. 2020.

[28] M. Kok and T. B. Schön, “Magnetometer Calibration Using Inertial Sensors,” IEEE Sensors Journal, vol. 16, no. 14, pp. 5679-5689, July, 2016.

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

[30] J. Hannink, T. Kautz, C. F. Pasluosta, J. Barth, S. Schulein, K.-G. Gallmann, J. Kluecken, and B. M. Eskofier, “Mobile stride length estimation with deep convolutional neural networks,” IEEE Jour. Bio. & Health Inform., vol. 22, no. 2, pp. 354-362, Mar. 2018.

[31] C. J. Hegarty, Understanding GPS/GNSS: Principles and Applications, Ch. 2, Artech House, 2019.

[32] J. Schmidhuber, “Deep learning in neural networks: An overview,” Neural networks, vol. 61, pp. 85-117 Jan. 2015.

[33] A. Gérondakis, Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, Chs. 1-2, O’Reilly Media, 2019.

[34] Howard, A. G. “Some improvements on deep convolutional neural network based image classification,” arXiv, pp. arXiv:1312.5402, Dec. 2013.

[35] Antoniou, A., Storkey, A., and Edwards, H., “Data augmentation generative adversarial networks.” arXiv, pp. arXiv:1711.04340, Nov. 2017.

[36] D. A. Clevert, T. Unterthiner, and S. Hochreiter, “Fast and accurate deep network learning by exponential linear units (elus),” arXiv, pp. arXiv:1511.07289, Nov. 2015.

[37] C. J. Willmott and K. Matsuura, “Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance,” Climate Research, vol. 30, no. 1, pp. 79-82, Dec. 2005.

[38] D. P. Kingma, and J. Ba, “Adam: A method for stochastic optimization,” arXiv, pp. arXiv:1412.6980, Dec. 2014.

[39] T.D. Vy, T. L. N. Nguyen, and Y. Shin, “A Precise Tracking Algorithm Using PDR and Wi-Fi/Beacon Corrections for Smartphones,” IEEE Access, vol. 9, pp. 49522-49536, Mar. 2021.