Vol.2021-UBI-70 No.9
2021/6/3

Preliminary Investigation of Using GPS Information to Improve Indoor Pedestrian Dead Reckoning

Heng Zhou[1] Takuya Maekawa[1]

Abstract: This study presents a method for improving the accuracy of conventional Pedestrian Dead Reckoning (PDR) using GPS satellite information in indoor environments. The accuracy of PDR is limited by the performance of inertial sensors because the errors caused by users' stride prediction and drift of a gyroscope continuously accumulate, resulting in large errors in predicted trajectories. We employ a neural network based PDR which mainly uses accelerometer and gyroscope embedded in a smartphone to predict the user's trajectories. To fix PDR's error on time, we use some landmarks which can be detected by another neural network that leverages GPS satellite information such as S/N ratio and azimuthal angles to predict if the user is close to windows in a building. Then, we fuse these two predictions based on the particle filter to predict a more accurate user's trajectory. We evaluated our framework using data obtained in different buildings in our campus and confirmed the effectiveness of the framework.

Keywords: Indoor localization, Pedestrian Dead Reckoning

1. Introduction

In recent years, with the proliferation of smartphones, a method called Pedestrian Dead Reckoning (PDR) [1], which estimates a walking trajectory of a smartphone user by using the embedded accelerometer and gyroscope, attracts attention of many researchers. In PDR, the accelerometer is used to calculate the walking distance of the user while the gyroscope sensor is used to detect the change in direction of walking. Then combining walking length for each step and direction, the PDR algorithm can estimate the user's walking trajectory. However, conventional PDR has some obvious and serious shortcomings due to the drift of the gyroscope, resulting in accumulated errors in the trajectory estimation with time. To solve this problem, some existing studies use some absolute position information obtained from smartphones to fix the accumulated errors. For example, some methods use Wi-Fi signal information to estimate the user's rough position and then according to this information, the methods eliminate PDR's accumulated error [2][3]. However, such methods require the complete construction of Wi-Fi positioning infrastructure which can cover the whole floor inside the building. In addition, the collection of fingerprints of Wi-Fi signal strength at each absolute coordinates should be conducted in advance and some tiny changes in the original environments may cause recollection of fingerprints, resulting in large installation and maintenance cost [4].

In this research, we propose a new method to solve the existing problems and realize high accuracy PDR regardless of an additional infrastructure. In the proposed method, we use GPS satellites information which can be used by anyone as an existing infrastructure. We have the following observation in our daily lives: in general, we cannot get GPS signal or can get poor GPS signal inside buildings but when approaching the windows or wandering by the window, we can receive GPS signal from the direction of the window. Therefore, by using these features, we can judge whether the user is near the windows and estimate the user in which side of the building using the GPS satellite signal strength information and directional information of observed satellites. Although this method is unlike Wi-Fi fingerprint methods that can provide absolute coordinates, we can combine it with map information so that we can estimate window areas as the relatively accurate coordinates which can help us elimi-

[1] Department of Multimedia Engineering, Graduate School of Information Science and Technology, Osaka University

nate PDR's accumulated errors.

2. Related Work

2.1 Conventional PDR

The PDR system was proposed as a solution to navigation in indoor environments or in weak and unstable GPS signal areas. The main advantage of PDR is to use inertial sensors, which are independent of environment changes. It uses accelerometer signals to detect steps to estimate walking distance and uses magnetometer combined with gyroscope signals to compute the heading direction [1]. Finally, with a given known starting position, it can integrate each step's displacement with direction change to get a full trajectory [5]. The main problem for PDR is the accuracy of heading part that the gyro accuracy and magnetic disturbance will cause errors growing with time. To improve the accuracy of PDR, researchers focus more on reducing errors of heading direction estimation [6], [7].

2.2 Addressing accumulated errors

Another method to improve dead reckoning is using real map information mainly including map matching methods [8], [9], [10] and landmarks based methods [11], [12]. On one hand, in map matching methods, data related to walls or other objects is used so that if an estimated position collides with an object because of drift errors, it will recalibrate the user's trajectory. On the other hand, some researchers use indoor landmarks that represent some special positions obviously different from the other positions. Landmarks can be detected by motion sensors or radio wave sensors like Wi-Fi or Bluetooth embedded in smartphones. For example, if a user approaches an elevator, data from the magnetometer will have obvious changes so that as the user is detected in landmarks but PDR has a drift error, PDR will be fixed immediately by this method.

3. Proposed method

3.1 Overview

The proposed method is composed of three main modules as shown in Figure 1. The (i) neural network based PDR module consists of two sub-modules: walking distance predictor and walking direction predictor. The walking distance predictor estimates the user's walking length within specific time from accelerometer data while the walking direction predictor estimates the angle of walking that changes within specific time from gyroscope data. The second module is named (ii) GPS landmark module consisting of two sub-modules. We firstly employ the window proximity detection model to estimate whether the user approaches the window in specific time by time-series information of GPS satellite signal. Then, the direction classification model estimates the direction of the window that we detected in the last model in specific time by using azimuthal angles of satellites. The third module is (iii) trajectory estimation module, based on particle filter, which is responsible for integrating outputs of the above modules to estimate the user's trajectory.

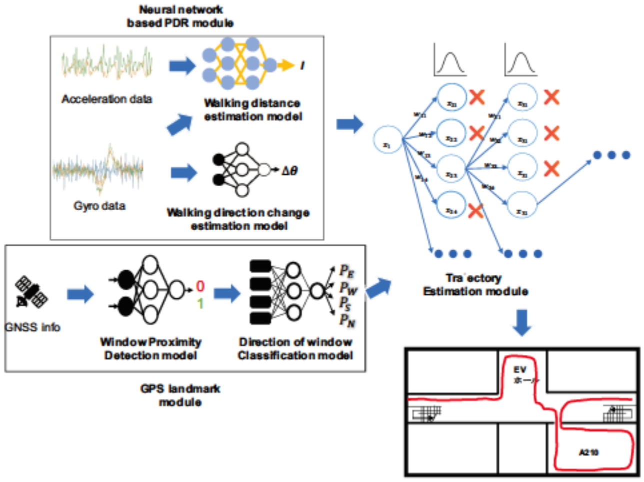


図 1: Overview of the whole system

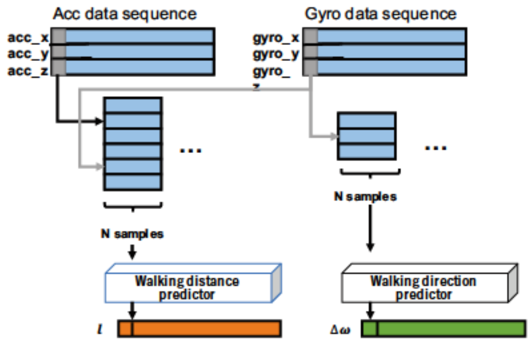


図 2: Data flow of Neural network based PDR Module

3.2 Neural network based PDR Module

We propose a method which is different from conventional PDR that predicts the walking distance based on the step counting and predefined stride length [13]. In our proposed method, we employ machine learning methods relying on inertial measurement units data of accelerometer and gyroscope and 2D ground truth trajectories. We train two different recurrent neural network based submodules [14] for walking distance and direction estimation on these data. Figure 2 shows the data flow of the motion sensor data input and estimated walking length and change of direction output of the two different LSTM models. [15]

3.2.1 Ground Truth Annotation

To acquire ground truth data of two sub-modules, we employ Google ARCore technology which can provide precise relative tri-dimensional positions. ARCore can be applied in all new Android phones and we can use its SDK to track camera poses [16], [17]. In the current method, we only consider two-dimensional positions as (x_t, y_t) at time t in a horizontal floor. Therefore, between time $t-1$ and t, the walking length d_t can be calculated by $d_t = \sqrt{(x_t - x_{t-1})^2 + (y_t - y_{t-1})^2}$ while change in walking direction is $\Delta\theta = atan2(\begin{vmatrix} u \\ v \end{vmatrix}, u \cdot v)$, where $u = (x_t - x_{t-1}, y_t - y_{t-1}), v = (x_{t+1} - x_t, y_{t+1} - y_t)$. We employ these data to train the neural networks.

3.2.2 Walking distance predictor

An input of this model is raw accelerometer data and gyroscope data obtained from a smartphone. After preprocessing these raw data, we feed them into a recurrent neural network to estimate the walking length in each time window.

Preprocessing

To simplify input, we use spline interpolation to unify the intervals and make up for missing data. In addition, since it is possible that frequencies of gyroscope and accelerometer are different, we resampled acceleration data to the same frequency as the gyroscope. We then apply min-max normalization to each of the preprocessed data (three-axis acceleration and gyroscope data). After that, we segment the six-axis time-series into one-second time windows to simply and synchronize the whole system. Note that each segment (time window) is composed of six time-series with the length of L samples.

Neural Network model

Using segments extracted above associated with d_t as labels, we train a LSTM model composed of a single LSTM layer with a dropout layer and one densely connected output layer. Then, we used relu function as activation function and mean squared error as the loss function. An output of this model at time t is estimated walking length within one time window size l_t.

3.2.3 Walking direction predictor

The input of this model is raw gyroscope data collected from a smartphone. After preprocessing these raw data, we feed them into a recurrent neural network to estimate the change of walking direction in each time window.

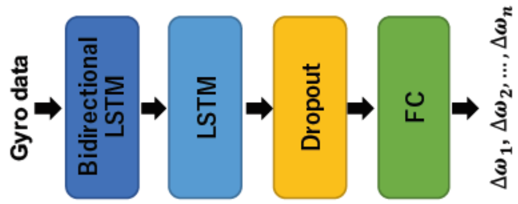


図 3: Neural Network Architecture in walking direction predictor

Preprocessing

The data preprocessing is almost the same as what we mentioned in the walking distance predictor but in the walking direction predictor, we use only three-axis gyroscope data as input. Thus, each one-second time window consists of three time-series with the length of L samples.

Neural Network model

In this part, we use a more complicated LSTM based model which consists of one Bi-directional LSTM [18] and one LSTM as shown in Figure 3. Since it is a regression model, the activation function is linear and the loss function is mean squared error. We chose this complicated model for this task because our preliminary investigation revealed that simpler models failed to capture slight changes in gyroscope data. An output of this model between time $t-1$ and t is the change of the moving direction $\Delta\omega_t$.

3.3 GPS landmark module

3.3.1 Window proximity detection model

GNSS information includes satellites' PRNs (id), S/N ratio (signal strength), azimuthal angles and elevation angles. The latest GNSS-based localization shows more than 10 meters error in indoor environments, which is not reliable. However, our preliminary experiment revealed that smartphones can still receive GNSS signal although they cannot provide an accurate positioning. In addition, when a smartphone approaches a window, it can receive strong GNSS signals, similar to outsides. Therefore, our proposed method leverages this fact to perform window proximity detection [19] to judge whether a smartphone approaches the window proximity area. Based on floormaps of various training environments, we predefined window proximity areas where smartphones can receive strong satellites signals within d meters from windows to collect labeled data for supervised machine learning in this part. In this model, we use processed GNSS signal strengths into a RNN model and compute the probability of window proximity in each time window.

Preprocessing

GNSS information sequences with timestamps are given. In order to synchronize motion data above and GNSS data here, we use the same segmentation size with neural network based PDR module to segment the GNSS time series. Within one window, since some satellites are likely to provide some outlying signals due to surrounding buildings, we delete signals from satellites with low elevation angles. Then we obtain the time series of signal strength within the time window for each satellite. We use min-max normalization to normalize signal strength for each satellite during each collection. Then, within one time window, the number of satellites is limited to 6 and if over 6, we chose only 6 satellites with largest signal strengths and if smaller than 6, zero padding will be performed. And in each time window, the sequences of satellite information are sorted by descending average signal strengths.

Detection

As we mentioned above, we predefined window proximity areas to employ a supervised machine learning. When the smartphone passed through these areas, we labelled feature vectors in the training dataset as 'window-side', and as 'not' otherwise [19]. Therefore, we train a binary classifier with a neural network model composed of a masking layer, a single LSTM layer, a dropout layer and a densely connected output layer on the labeled data. The activation function we used is sigmoid and the loss function is binary-crossentropy. The result obtained is called $P_{window}(t)$ as the probability of window proximity at time t. We set the threshold for $P_{window}(t)$ as 0.4 if over the threshold, we judge the user is near a window at time t and otherwise not near a window.

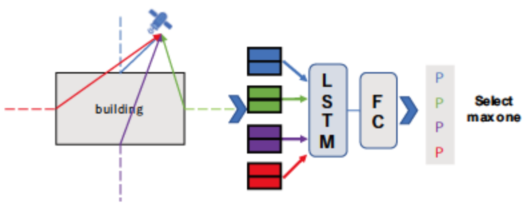


図 4: Data Flow in Direction classification model

3.3.2 Direction classification model

In this part, as we have detected whether a user approaches the window at time t, our goal is to estimate the direction of the external wall where that window is installed. This method is based on previous study of wall orientation classification [19] that makes assumptions that when a smartphone approaches a window, if azimuth of a satellite is closer to the direction of that window, the signal received from this satellite is stronger than signals from other satellites. We use a single binary classifier to determine the user approaches a specific orientation of the building. For each direction, we extract a corresponding feature vector in each time window. For example, if there are four sides in a building, we extract 4 feature vectors totally in a time window. This design allows us to train only one classifier for direction classification and then use it to get the probability for each orientation. Figure 4 shows this model of example with 4 directions that in each window, we can extract 4 feature vectors and through the LSTM model, we can get 4 corresponding probabilities.

Feature Extraction

For each time window, assuming there are m directions of outer walls, we calculate m feature vectors. For each direction, we used 6 satellites in each window as the same as the window proximity detection model, while we chose satellites with smallest angular differences between satellites and direction of wall. For example, for direction i, the normal line angle of λ_i are predefined by floormap information and the azimuthal angle of one satellite chosen above can be viewed as μ_t at time t. Therefore, the angular difference is calculated as $|\mu_t - \lambda_i|$. To normalize data, we use $\cos(|\mu_t - \lambda_i|)$ and $\sin(|\mu_t - \lambda_i|)$ as features. We also used min-max normalization to normalize signal strengths and elevation angle for each satellite. Then we concatenate these 4 values from each of the 6 satellites into a feature vector which is used for a classifier to determine if the direction of the nearest window is direction i. Therefore, the input for LSTM model is a 4 dimension with 6 length feature vector in one time window for direction i. For remaining directions, we used the same method to compute feature vectors.

Classification

When we prepare training data, for each direction, we label each of the feature vectors above as true when the user is really at corresponding window side areas of the building and as false otherwise. Then we train all data of m directions in a LSTM model with a masking layer, a LSTM layer, a batchnormaliztion layer and a fully connected layer. The activation function and loss function are identical to those in the window proximity model. When estimating the window direction, we compute the class probability of true for each orientation and we can get $P_1(t), P_2(t), \ldots, P_m(t)$ for totally m directions at t. Then we chose the largest one as our direction of window prediction at t.

3.4 Trajectory Estimation model

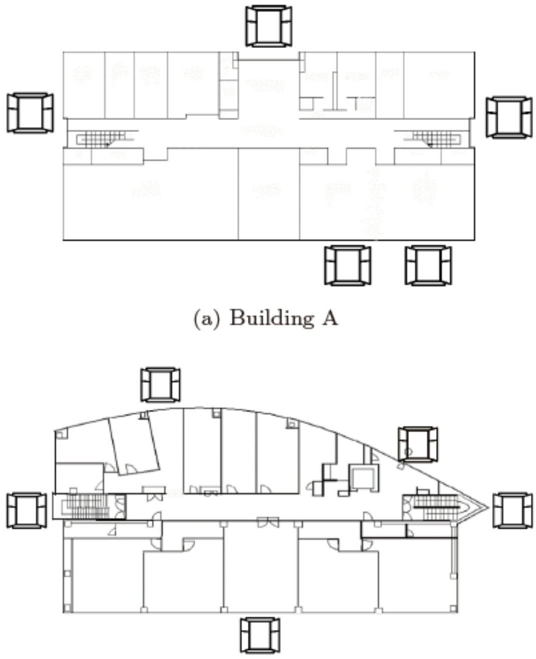
To combine motion results and GPS landmark module results, we choose to use particle filter algorithm [20] which is usually used to estimate the states of non-linear systems. In our particle filter tracking, each particle represents the user's location. The algorithm works by iterating the following steps:

- Predict: For the remaining particles selected from the last iteration, each particle generates N particles. The position of new particle i is calculated by following: $\begin{bmatrix} x_{i_t} \\ y_{i_t} \end{bmatrix} = \begin{bmatrix} x_{i_{t-1}} \\ y_{i_{t-1}} \end{bmatrix} + l_t * \begin{bmatrix} \sin(\Sigma\rho_t) \\ \cos(\Sigma\rho_t) \end{bmatrix}$ where ρ_t is generated randomly by normal distribution with mean $\Delta\omega_t$ and a constant variance σ.
- Update: For each particle which was generated in the Predict step, we reassign weight to it with GPS landmark module results and floormap information in this step. We define the overall weight of particle i combined with two weights following as $w = \alpha_1 w_1 + \alpha_2 w_2$ where α_1 and α_2 are two constants. Firstly, for particle i, it has weight $w_1 = CDF(\rho_t)$ in a normal distribution of $(\Delta\omega_t, \sigma)$ if $\rho_t < \Delta\omega_t$ and $w_1 = 1 - CDF(\rho_t)$ otherwise. We use cumulative distribution function (CDF) here because in this method, it can describe the probability of such a particle in the normal distribution and we set this probability as this particle's self-weight. Then we use GPS landmark module results at time t, if $P_{window}(t)$ is larger than the threshold (our default is 0.4) and (x_{i_t}, y_{i_t}) is in the real corresponding window side area of the predicted direction, the particle i has another weight $w_2 = P_{window}(t)$ and $w_2 = 0$ otherwise. We define the overall weight of particle i combined with two weights mentioned above as $w = \alpha_1 w_1 + \alpha_2 w_2$ where α_1 and α_2 are two constants.
- Resampling: From these new predicted weighted particles, we resample them by their weights. In this study, we will choose top-k weights particles.

4. Evaluation

4.1 Data set

Our dataset consists of GNSS data and motion sensor data collected in two environments (buildings) and there are totally 26 different routes. We used Google Pixel 4 with Android 10 to collect motion and GNSS data with a



(a) Building A

(b) Building B

図 5: Buildings A and B, and locations of windows

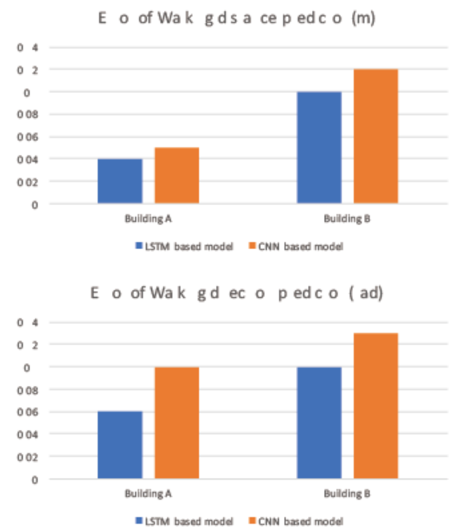
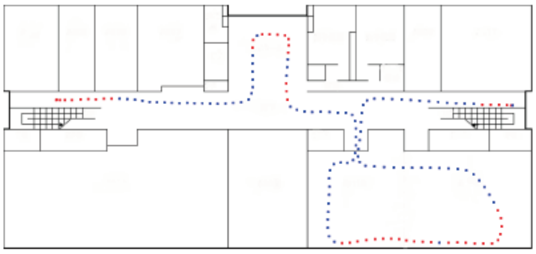
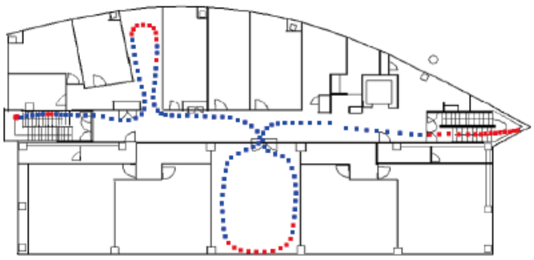


図 6: Comparison of MAEs of two predictors between building A and building B

self-developed App and used another Google Pixel 3a with Android 10 to collect ground truth with ARCore App. During the experiments, a subject held the smartphones by hand. The floormaps of two buildings are shown in Figure 5 and we marked where windows are. Building A is an office floor on the 2nd floor and during the collection, the subject may pass window areas near the south or north staircases or in the west lounge or in the east classroom. There is a neighboring building located in the west of Building A. Building B is also an office floor on the 6th floor. However, the south and north windows are between



(a) Building A



(b) Building B

図 7: Window proximity detection model results in building A and building B

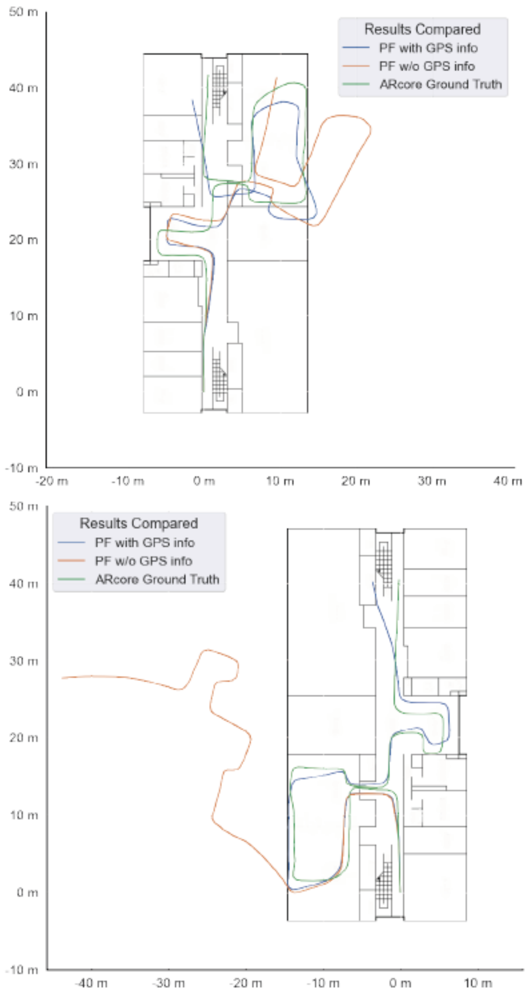


図 8: Comparison of predicted trajectories by different methods

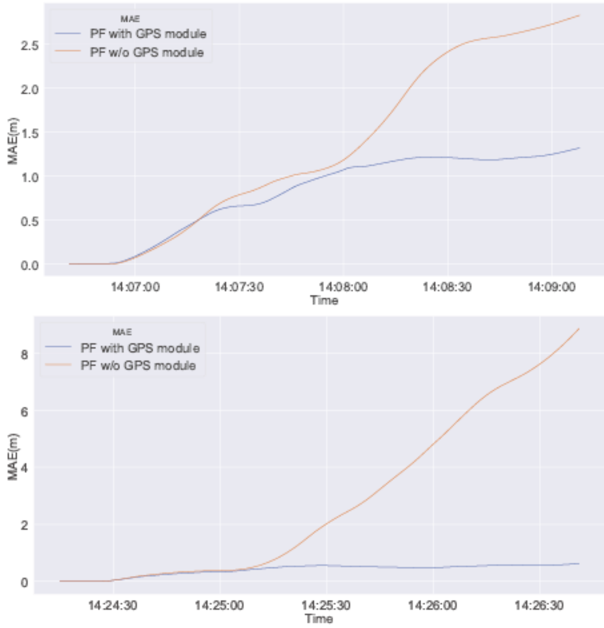


図 9: MAEs changes with time of above two trajectories

	Window proximity detection	Direction Classification
A	82.06%	91.20%
B	81.30%	82.09%
Overall	81.84%	88.03%

表 1: Accuracy of models in GPS landmark module

staircases and there are no neighboring high buildings.

4.2 Evaluation Methodology

The evaluation was conducted using 'leave-one-out' cross validation which means that we iterated each route as the testing dataset with the remaining datasets used as training datasets. In the walking distance predictor and walking direction predictor, we compare the mean absolute errors of LSTM-based models and CNN-based models because we want to test which DNN framework is better for this task. The structures of LSTM-based models are mentioned above while CNN-based models consist of two 1-D convolution layers, one max pooling layer, one flatten layer and one dense layer and same activation function and loss function as LSTM-based models. In addition, the length L of each time window is 50 samples. In the GPS landmark module, we compute the classification accuracy by comparing the predicted results with ground truth. In the trajectory estimation model, we compare our proposed method with the situation if we don't use GPS landmark module results as well as the ground truth trajectories. We use Tensorflow as a deep learning platform.

4.3 Performance of neural network based PDR module

Figure 6 shows the comparison of the MAEs (mean absolute errors) for all trajectories collected in building A and in building B between LSTM-based model and CNN-based model for walking distance predictor and walking direction predictor respectively. No matter walking distance predictor or walking direction predictor, in both environments, the average MAEs of LSTM-based models are smaller than that of CNN-models. The results in building B are not very good because we consider that it includes much data about going downstairs or going upstairs which is not a horizontal motion.

4.4 Performance of GPS landmark module

Table 1 shows the accuracies of window proximity detection model and direction classification model for all trajectories respectively in building A and B. Also, in Figure 7, we showed the results of two randomly selected trajectories associated with the outputs of the window proximity detection model respectively in building A and B. These dots represent positions during movement at different times. Red dots represent positions which are detected as window areas while blue dots are not. We can find that there are some outliers but these results do not affect finding out window areas.

4.5 Performance of Trajectory Estimation module

We randomly selected two trajectories in building A and Figure 8 shows the comparison of predicted trajectories by the proposed method, predicted trajectories by the proposed method without the GPS landmark module and ground truth trajectories by ARCore. Figure 9 shows the transitions of the MAEs with time. Our proposed method can achieve our goal that it can eliminate gyro's accumulated errors in window areas while without the GPS landmark module, the errors caused by the gyro would continuously increase.

5. Discussion

5.1 GNSS data collected in movement

In the previous study, we investigated the relationship between GNSS data and indoor positions. We collected GNSS data in some specific areas indoor and kept standing for long time [19]. However, in this study, we collected GNSS data during movement and since our window areas are not large enough, in general, we passed through a window area no more than 10 seconds. As a result, it is not as accurate as data collected during standing. Therefore, sometimes signal strength may keep unchanged or collapse and they are outliers.

5.2 Limitations

All of our data collection environments in this study are in concrete buildings, which interfere with GNSS signals. In addition, it may not work effectively in buildings with few windows and in buildings where windows are in the corners.

5.3 Future work

In the future, we still have some problems to solve. We will improve the accuracy of the motion system and a more intelligent method to detect outliers in the two GPS related models to improve versatility of the trajectory prediction model. In addition, to improve the current system, we plan to use more map information such as walls and in such a way, starting position information cannot be required, which will be an important contribution for pedestrian dead reckoning. Furthermore, we will try to explore a tri-dimension motion system including going upstairs or downstairs in the future.

6. Conclusion

In this research, we propose a new PDR method which is combined with GPS information. Our proposed method can reduce accumulated errors caused by gyro shift. We investigated the performance of the proposed method with different walking routes. The models we used in the neural network based PDR module are better than other DNN models. The accuracy of the window proximity detection module is 81.84% and the accuracy of direction classification module is 88.03%. The mean absolute errors of predicted trajectories can be controlled into 2 meters by our proposed method.

Acknowledgments

This work is partially supported by JST CREST JP-MJCR15E2, JSPS KAKENHI Grant Number JP16H06539 and JP17H04679.

参考文献

[1] Mezentsev, O., Lachapelle, G. and Collin, J.: Pedstrian Dead Reckoning—A solution to navigation in GPS signal degraded areas?, *GEOMATICA*, Vol. 59, No. 2, pp. 175–182 (2005).

[2] Zhuang, Y. and El-Sheimy, N.: Tightly-Coupled Integra-

tion of WiFi and MEMS Sensors on Handheld Devices for Indoor Pedestrian Navigation, *IEEE Sensors Journal*, Vol. 16, No. 1, pp. 224–234 (2016).

[3] Chen, Z., Zou, H., Jiang, H., Zhu, Q., Soh, Y. C. and Xie, L.: Fusion of WiFi, Smartphone Sensors and Landmarks Using the Kalman Filter for Indoor Localization, *Sensors*, Vol. 15, No. 1, pp. 715–732 (2015).

[4] Yang, Z., Wu, C. and Liu, Y.: Locating in fingerprint space: Wireless indoor localization with little human intervention, *the Annual International Conference on Mobile Computing and Networking, MOBICOM* (2012).

[5] Kang, W. and Han, Y.: SmartPDR: Smartphone-Based Pedestrian Dead Reckoning for Indoor Localization, *IEEE Sensors Journal*, Vol. 15, No. 5, pp. 2906–2916 (2015).

[6] Ladetto, Q. and Merminod, B.: An alternative approach to vision techniques-pedestrian navigation system based on digital magnetic compass and gyroscope integration, *6th World Multiconference on Systemics, Cybernetics and Information, Orlando, USA*, No. CONF (2002).

[7] Abyarjoo, F., Barreto, A., Cofino, J. and Ortega, F. R.: Implementing a Sensor Fusion Algorithm for 3D Orientation Detection with Inertial/Magnetic Sensors, *Innovations and Advances in Computing, Informatics, Systems Sciences, Networking and Engineering* (Sobh, T. and Elleithy, K., eds.), Vol. 305–310, Cham, Springer International Publishing (2015).

[8] Yoshimi, S., Kanagu, K., Mochizuki, M., Murao, K. and Nishio, N.: PDR Trajectory Estimation Using Pedestrian-Space Constraints: Real World Evaluations, *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, New York, NY, USA, Association for Computing Machinery, pp. 1499–1508 (2015).

[9] Ascher, C., Kessler, C., Wankerl, M. and Trommer, G.: Dual IMU Indoor Navigation with particle filter based map-matching on a smartphone, *2010 International Conference on Indoor Positioning and Indoor Navigation*, pp. 1–5 (2010).

[10] Zhu, X., Li, Q. and Chen, G.: APT: Accurate outdoor pedestrian tracking with smartphones, *2013 Proceedings IEEE INFOCOM*, pp. 2508–2516 (2013).

[11] Abdelnasser, H., Mohamed, R., Elgohary, A., Farid, M., Wang, H., Sen, S., Choudhury, R. and Youssef, M.: SemanticSLAM: Using Environment Landmarks for Unsupervised Indoor Localization, *IEEE Transactions on Mobile Computing*, Vol. 15, pp. 1–1 (2015).

[12] Wang, H., Sen, S., Elgohary, A., Farid, M., Youssef, M. and Choudhury, R.: No need to war-drive: Unsupervised indoor localization, *10th International Conference on Mobile Systems, Applications, and Services* (2012).

[13] Kawaguchi, N.: Sampling rate dependency in pedestrian walking speed estimation using DualCNN-LSTM (2019).

[14] Jain, L. C. and Medsker, L. R.: *Recurrent Neural Networks: Design and Applications*, CRC Press, Inc., USA, 1st edition (1999).

[15] Hochreiter, S. and Schmidhuber, J.: Long Short-Term Memory, *Neural Computation*, Vol. 9, No. 8, pp. 1735–1780 (1997).

[16] Google: ARCore, https://developers.google.com/ar/.

[17] Teng, X., Xu, P., Guo, D., Guo, Y., Hu, R., Chai, H. and Chuxing, D.: ARPDR: An Accurate and Robust Pedestrian Dead Reckoning System for Indoor Localization on Handheld Smartphones, *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10888–10893 (2020).

[18] Schuster, M. and Paliwal, K.: Bidirectional Recurrent Neural Networks, *Trans. Sig. Proc.*, Vol. 45, No. 11 (1997).

[19] Fujiwara, M., Nakatani, T., Tian, Y., Korpela, J., Maekawa, T. and Hara, T.: Smartphone-Assisted Automatic Indoor Localization of BLE-Enabled Appliances Using BLE and GNSS Signals, *the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, BuildSys '20, New York, NY, USA, Association for Computing Machinery, pp. 80–89 (2020).

[20] Gustafsson, F., Gunnarsson, F., Bergman, N., Forssell, U., Jansson, J., Karlsson, R. and Nordlund, P.-J.: Particle Filters for Positioning, Navigation, and Tracking, *Trans. Sig. Proc.*, Vol. 50, No. 2, pp. 425–437 (2002).