Gait-Robust Heading Estimation Using Horizontal Acceleration for Smartphone-based PDR

Kazuma Kano[1], Takuto Yoshida[1], Shin Katayama[1], Kenta Urano[1], Takuro Yonezawa[1] and Nobuo Kawaguchi[1]

[1]*Graduate School of Engineering, Nagoya University, Nagoya, Japan*

Abstract

This study tackles heading estimation for Pedestrian Dead Reckoning (PDR) with smartphones. In dealing with changes in the holding posture of smartphones, it works to consider the relationship between sensor orientation and heading. However, the existing methods lack robustness to various gaits, such as sideways and backward walking. Therefore, we propose a novel method considering various spatiotemporal features of horizontal acceleration with deep learning. The proposed method calculates horizontal acceleration in the global coordinate system from measured acceleration, gravitational acceleration, and rotation vector. Then, it inputs the horizontal acceleration over a certain period into a deep neural network model and predicts the unit vector directed to the mean heading during that period. We created a dataset covering multiple gaits and evaluated the method using four models: Convolutional Neural Network (CNN), Bidirectional Long Short-Term Memory (BiLSTM), DualCNN-LSTM, and DualCNN-Transformer. Consequently, we found that the proposed method was more robust to gaits than the existing methods, with the DualCNN-LSTM and DualCNN-Transformer models achieving the highest accuracy.

Keywords

dataset, deep learning, indoor localizaton, indoor positioning, pedestrian dead reckoning

1. Introduction

Pedestrian Dead Reckoning (PDR) is a positioning technology available indoors and outdoors. It estimates relative trajectory from the start based on measurement data from sensors carried by the target, such as inertial and magnetic sensors. PDR offers advantages such as low power consumption, no need for infrastructure, and frequent estimation, holding promise for application to the positioning of smartphone users. Most PDR methods can be classified into two types: those that estimate 3D trajectories by second-order integrating acceleration and those that estimate 2D trajectories by separately estimating walking speeds and headings[1]. The latter type is commonly used due to poor measurement accuracy when utilizing Micro Electro Mechanical Systems (MEMS) sensors embedded in smartphones.

Proceedings of the Work-in-Progress Papers at the 13th International Conference on Indoor Positioning and Indoor Navigation (IPIN-WiP 2023), September 25–28, 2023, Nuremberg, Germany

kazuma@ucl.nuee.nagoya-u.ac.jp (K. Kano); takuto@ucl.nuee.nagoya-u.ac.jp (T. Yoshida); shinsan@ucl.nuee.nagoya-u.ac.jp (S. Katayama); urano@nagoya-u.jp (K. Urano); takuro@nagoya-u.jp (T. Yonezawa); kawaguti@nagoya-u.jp (N. Kawaguchi)

0000-0002-7514-1655 (K. Kano)

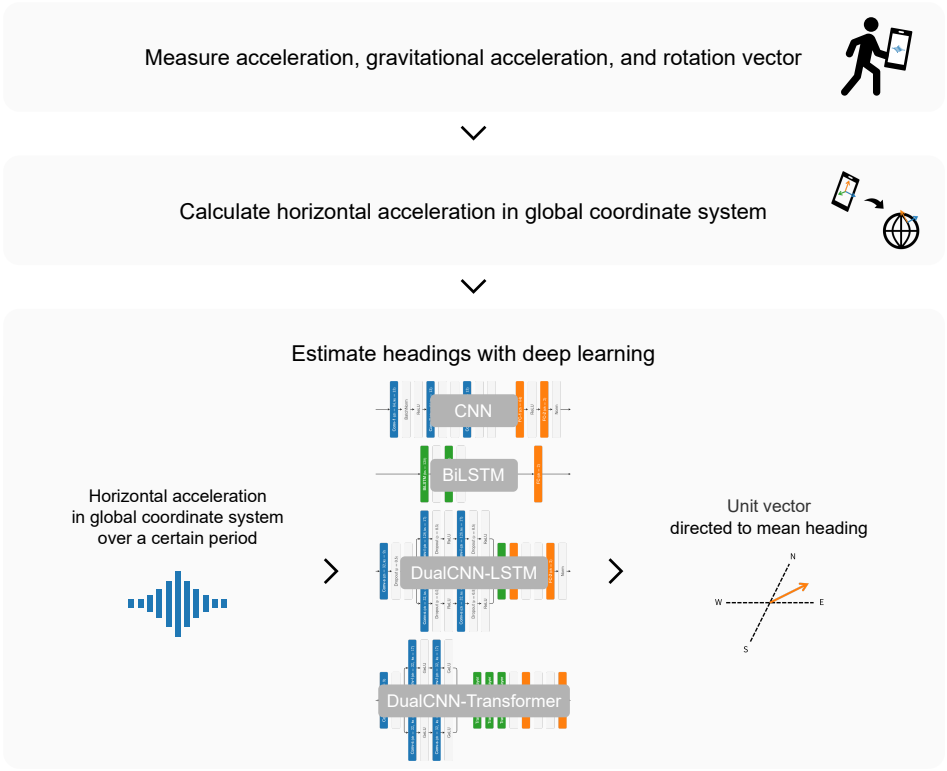


Figure 1: Overview of proposed method.

Walking speed estimation methods include combining step detection and stride estimation[2, 3] and directly computing walking speeds[4, 5]. These methods have attained adequately high accuracy. On the other hand, heading estimation methods are divided into just regarding sensor rotations around the vertical axis as the heading changes[6] and considering the relationship between the sensor orientations and actual headings. The latter methods can deal with changes in the holding posture of smartphones during positioning. However, the existing methods still have problems, such as a lack of robustness against gait differences.

Therefore, we propose a data-driven approach to estimate headings based on horizontal acceleration in the Global Coordinate System (GCS). We aim to improve gait robustness by considering various spatiotemporal features with deep learning. Fig. 1 illustrates an overview of the proposed method. First, we calculate horizontal acceleration in GCS from acceleration, gravitational acceleration, and rotation vector measured by a smartphone. Then, we input the horizontal acceleration in GCS over a certain period into a deep neural network model to predict the unit vector directed to the mean heading during that period. This study explores four models: Convolutional Neural Network (CNN), Bidirectional Long Short-Term Memory (BiLSTM), DualCNN-LSTM, and DualCNN-Transformer. We created a dataset supporting multiple gaits and evaluated the estimation accuracy. The result showed that the proposed method improved gait robustness compared to the existing and especially achieved the highest accuracy when using the DualCNN-LSTM and DualCNN-Transformer models.

2. Related Work

2.1. Heading Estimation Methods Based on Horizontal Acceleration in Global Coordinate System

Horizontal acceleration tends to spread along heading directions because people repeatedly accelerate and decelerate in walking. Deng et al. proposed Rotation Matrix and Principal Component Analysis (RMPCA), which estimates heading as the first principal component of horizontal acceleration obtained via PCA[7]. Ban et al. estimated heading as the mean vector of horizontal acceleration while the magnitude of acceleration without gravity component exceeded a certain threshold[8]. For convenience, we refer to this method as Horizontal Acceleration Mean (HAM) method. Both methods preprocess acceleration by transforming it into GCS with the origin at the sensor's position. Here, GCS refers to a right-handed coordinate system defined as follows:

- X-axis is horizontal and points east at the origin.
- Y-axis is horizontal and points north at the origin.
- Z-axis is vertical and points upward at the origin.

Transforming into GCS in advance helps identify the problems at sensor orientation estimation and heading estimation. It also enables independent heading estimation at each time step, avoiding the accumulation of heading errors even during prolonged positioning. However, these methods lack gait robustness as they do not account for temporal information or other spatial features.

2.2. Pedestrian Dead Reckoning Methods Using Deep Learning

Deep learning, which can automatically choose and consider various features, seems effective in improving gait robustness. Chen et al. proposed IONet, which uses BiLSTM to estimate trajectories and demonstrated improved positioning accuracy compared to conventional PDR methods[9]. IONet trains the model to output moved distances and heading displacements from acceleration and angular velocity in Sensor Coordinate System (SCS). Then, it sequentially integrates the displacements to estimate the headings. Chen et al. also attempted to reduce the computational complexity by applying WaveNet[10]. Kawaguchi et al. proposed DualCNN-LSTM as a model for walking speed estimation, designed to handle various gaits such as fast walking and stepping in place[4, 5]. DualCNN-LSTM has two paths consisting of convolutional layers with different kernel sizes connected in parallel to an LSTM layer. It is presumed to accommodate various gaits by extracting a wide range of features from short-term to long-term. Although many studies have applied deep learning to PDR, they have not sufficiently discussed the gait robustness of heading estimation. In addition, previous studies usually input sensor measurement data into models before the coordinate transformation, making it impossible to separate errors in sensor orientation estimation and heading estimation.

2.3. Activity Recognition Methods Using Deep Learning

Human Activity Recognition (HAR) is another representative research field involving sensor measurement data analysis. Ha et al. used CNN to recognize actions in car assembly lines and daily life based on acceleration measured by multiple sensors and multimodal data[11]. Zhao et al. applied BiLSTM and improved classification accuracy compared with plain LSTM[12]. Shavit et al. applied Transformer, a state-of-the-art model in research fields such as natural language processing and computer vision, to HAR and improved classification accuracy compared to simple CNN[13].

3. Methodology

We use deep neural network models to estimate headings from horizontal acceleration in GCS. Fig. 2 describes the training and prediction processes of the proposed method. In the offline phase, we train models following the solid blue arrows. First, measure acceleration, gravitational acceleration, and rotation vector with a smartphone while measuring the trajectory with surveying equipment. Next, align their timestamps and resample at 100 (Hz). Then, calculate horizontal acceleration in GCS and denoise them before inputting to the model. Finally, train the model by backpropagating the losses between the estimated headings and ground truth computed from the actual trajectory. In the online phase, we predict headings following the dashed orange arrows.

3.1. Deriviation of Horizontal Acceleration

Horizontal acceleration in GCS is calculated from acceleration and gravitational acceleration in SCS and rotation vector. Firstly, project the acceleration $\boldsymbol{a}^s$ onto the gravitational acceleration $\boldsymbol{g}^s$ to obtain the vertical component of acceleration $\boldsymbol{a}^s_v$. Subtracting $\boldsymbol{a}^s_v$ from $\boldsymbol{a}^s$ gives horizontal acceleration $\boldsymbol{a}^s_h$. The superscript 's' stands for SCS.

$$\boldsymbol{a}^s_v = \left(\frac{\boldsymbol{a}^s \cdot \boldsymbol{g}^s}{\boldsymbol{g}^s \cdot \boldsymbol{g}^s} \right) \boldsymbol{g}^s \tag{1}$$

$$\boldsymbol{a}^s_h = \boldsymbol{a}^s - \boldsymbol{a}^s_v \tag{2}$$

Secondly, transform the horizontal acceleration $\boldsymbol{a}^s_h$ into GCS using the rotation vector. The rotation vector corresponds to the smartphone orientations in GCS with the origin at the smartphone's position. Applying the rotation operation f represented by the rotation vector to the horizontal acceleration $\boldsymbol{a}^s_h$ in SCS yields horizontal acceleration $\boldsymbol{a}^g_h$ in GCS. The superscript 'g' stands for GCS.

$$\boldsymbol{a}^g_h = f(\boldsymbol{a}^s_h) \tag{3}$$

Finally, apply the Gaussian filter with a standard deviation $\sigma = 1.1$ to remove noises. The filtered horizontal acceleration is calculated as the convolution of the Gaussian kernel and original horizontal acceleration $\boldsymbol{a}^g_h$ along the time dimension. We obtain each component of

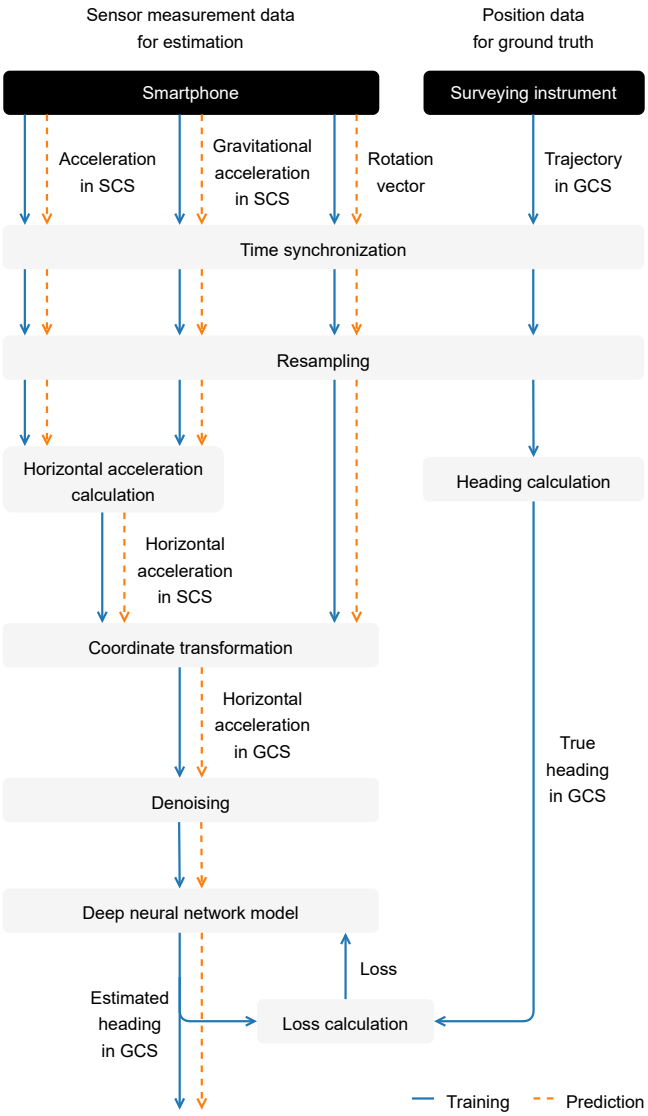


Figure 2: Flows of training models and predicting headings.

the filtered horizontal acceleration at time t

$$\hat{a}_x^{g,t} = \frac{1}{\sqrt{2\pi}\sigma} \sum_{i=-r}^{r} a_x^{g,t-i} \exp\left(-\frac{i^2}{2\sigma^2}\right) \tag{4}$$

$$\hat{a}_y^{g,t} = \frac{1}{\sqrt{2\pi}\sigma} \sum_{i=-r}^{r} a_y^{g,t-i} \exp\left(-\frac{i^2}{2\sigma^2}\right) \tag{5}$$

where $a_x^{g,t-i}$ and $a_y^{g,t-i}$ denote each component of the original horizontal acceleration at time $t-i$. We round the kernel radius r to 3 because the exponential term is negligibly small.

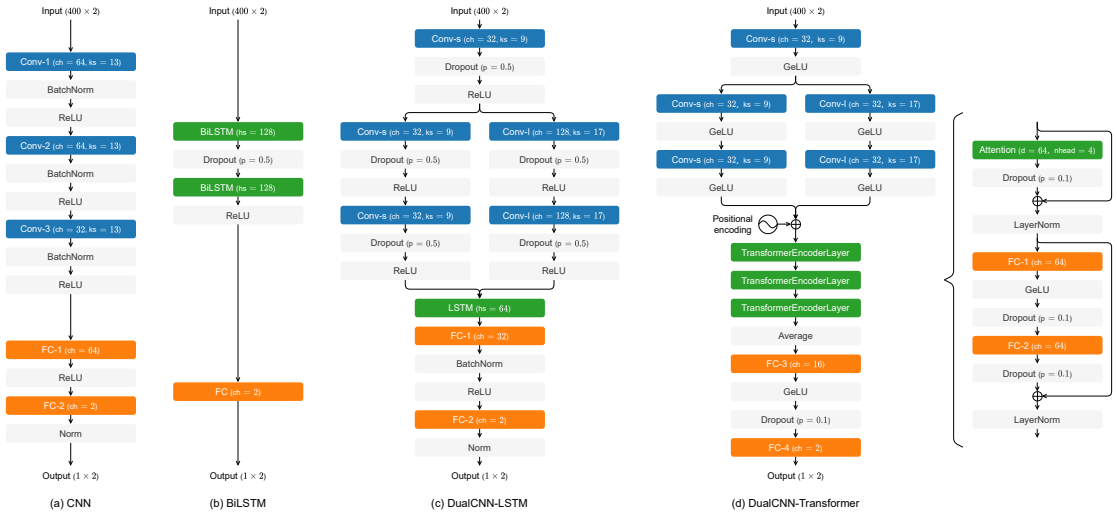


Figure 3: Network architectures.

3.2. Heading Estimation Model

3.2.1. Input and Output

The model receives the horizontal acceleration in GCS over a certain period and outputs the unit vector directed to the mean heading during that period. Note that the trajectories measured by surveying equipment and the headings computed from them may contain errors. We intend to alleviate their influence by taking the average. Additionally, estimating the heading as a vector rather than an angle helps eliminate discontinuity at the ± 180 (deg) boundary and stabilize the learning process[14]. The x and y components of the vector correspond to the cosine and sine of the angle, respectively. In this paper, we empirically set the input length to four seconds. Since resampled at 100 (Hz), the input size is 400×2, and the output size is 1×2.

3.2.2. Network Architecture

This study considers four models: CNN, BiLSTM, DualCNN-LSTM, and DualCNN-Transformer. Fig. 3 displays the model architectures. The CNN model has three stacked convolutional layers, extracting short-term features at the shallow layers and long-term at the deep. The BiLSTM model extracts features across the entire data in the BiLSTM layers. The concatenated forward and backward hidden states of the second BiLSTM layer are fed into the fully connected layer. The DualCNN-LSTM model extracts various-timescale features in two different-sized convolutional paths and captures how those features lie throughout the data in the LSTM layer. The inputs of two paths are shared, and their outputs are concatenated and passed to the LSTM layer. The DualCNN-Transformer model extracts features in two different-sized convolutional paths and considers the relationship among them by self-attention. The concatenated outputs of two paths with learnable positional encoding added are passed to the Transformer encoder. The output of the Transformer encoder is averaged over the time dimension and then fed into

Table 1
Hyperparameters of CNN model

Conv-1	out channels	32, **64**, 128
	kernel size	3, 7, **13**
	stride	**1**
	padding	**0**
Conv-2	out channels	32, **64**, 128
	kernel size	3, 7, **13**
	stride	**1**
	padding	**0**
Conv-3	out channels	**32**, 64, 128
	kernel size	3, 7, **13**
	stride	**1**
	padding	**0**
FC-1	out channels	32, **64**, 128

Table 2
Hyperparameters of BiLSTM model

BiLSTM	hidden size	32, 64, **128**
	num layers	**2**, 3, 4
	dropout	0.125, 0.25, **0.5**

Table 3
Hyperparameters of DualCNN-LSTM model

Conv-s	out channels	**32**, 64, 128
	kernel size	3, 5, **9** *
	stride	**1**
	padding	**0**
	dropout	0.125, 0.25, **0.5** *
Conv-l	out channels	32, 64, **128**
	kernel size	5, 9, **17** *
	stride	**1**
	padding	**0**
	dropout	0.125, 0.25, **0.5** *
LSTM	hidden size	32, **64**, 128
FC-1	out channels	**32**, 64, 128

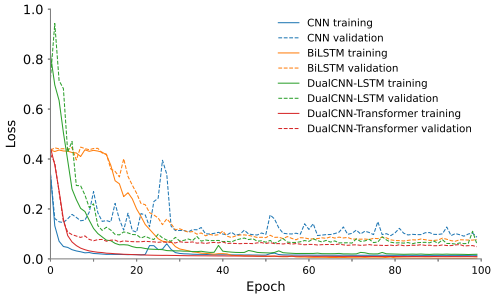


Figure 4
Loss transition during training.

the fully connected layer. We normalize the final output to have a norm of 1 for the CNN and DualCNN-LSTM models. On the other hand, we do not normalize for the BiLSTM and DualCNN-Transformer models because it hinders their learning processes.

3.2.3. Training and Hyperparameters

We perform estimation and update the weights at 10 (Hz) during training. In other words, the window to slice input data slides by ten samples at a time. We employ Mean Squared Error (MSE) as a loss function and Adam with a learning rate of 0.001 as an optimizer. The batch size is set to 512. We train models for 100 epochs and use the weights at the epoch with the smallest validation loss. We determined the hyperparameters by grid search from the candidate values listed in Tables 1, 2, 3, and 4. Parameters written in bold indicate what resulted in the smallest validation loss. Note that we constrained some parameter combinations due to the large search space. For the DualCNN-LSTM model, the kernel sizes ks_{cs} and ks_{cl}, the dropout

Table 4
Hyperparameters of DualCNN-Transformer model

Conv-s	out channels	16, **32**, 64 *
	kernel size	3, 5, **9** *
	stride	**1**
	padding	**0**
Conv-l	out channels	16, **32**, 64 *
	kernel size	5, 9, **17** *
	stride	**1**
	padding	**0**
Transformer encoder	num layers	1, 2, **3**
	Attention dim	32, **64**, 128 *
	Attention num heads	**4**
	Feed-Forward dim	32, **64**, 128
	dropout	**0.1**
FC-3	out channels	8, **16**, 32 *
	dropout	**0.1**

proportions p_{cs} and p_{cl}, for Conv-s and Conv-l, respectively, satisfy the following relationship:

$$2ks_{cs} - 1 = ks_{cl} \tag{6}$$

$$p_{cs} = p_{cl} \tag{7}$$

For the DualCNN-Transformer model as well, the numbers of out channels ch_{cs} of Conv-s and ch_{cl} of Conv-l, the dimension d_{at} of Attention, the number of out channels ch_{fc} of FC-3, and the kernel sizes ks_{cs} of Conv-s and ks_{cl} of Conv-l satisfy the following relationship:

$$2ch_{cs} = 2ch_{cl} = d_{at} = 4ch_{fc} \tag{8}$$

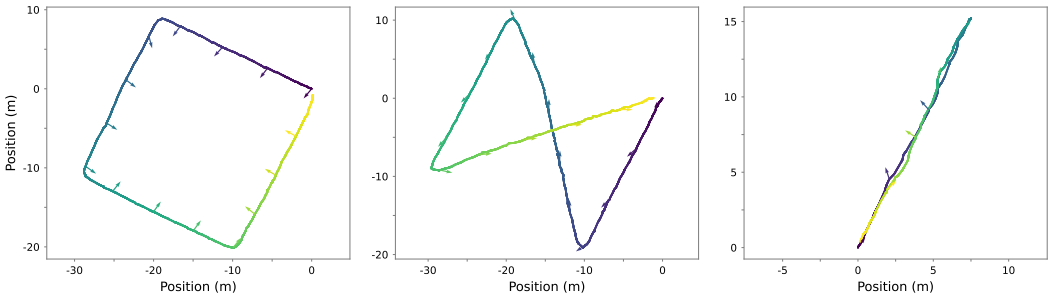
$$2ks_{cs} - 1 = ks_{cl} \tag{9}$$

Fig. 5 shows the loss transition with the best hyperparameters.

4. Evaluation

4.1. Dataset

We created a dataset supporting multiple gaits for evaluation. We collected acceleration, gravitational acceleration, rotation vector, angular velocity, and geomagnetic field with a smartphone (Google Pixel 4, Android 10). These sensor measurement data can be retrieved via Android Sensor Framework API and are recorded in SCS. The gravitational acceleration and rotation vector are internally computed from the acceleration, angular velocity, and geomagnetic field. We simultaneously measured 3D position data in GCS using a laser surveying instrument (TOPCON GT-1205). We instructed the subjects to hold the smartphone in front of their chest and walk along markers we placed beforehand.



(a) Rectangle (b) Hourglass (c) Straight line

Figure 5: Instances of walking patterns.

The subjects are eight males aged 21 to 23 years. The gaits consist of forwards, right sideways, left sideways, and backwards. The walking courses have three shapes: rectangle, hourglass, and straight line. Fig. 5 exhibits the instances of trajectories and walking postures. Their colors indicate the temporal transition, meaning the subjects moved from purple to yellow. The arrows represent the smartphone orientations at each time computed from the rotation vector. The side lengths of the rectangular and hourglass-shaped courses range from 20 to 31 (m). The subjects circulate the route by a single gait in the rectangular and hourglass-shaped courses. These data include cornering movements (i.e., heading changes during walking). On the other hand, the subjects change their gaits every 5 (m) and change their headings by 180 (deg) every 15 (m) in the straight-line courses. These data include gait changes and heading changes by 180 (deg). Total walking distance and recording time are 8543 (m) and 120 minutes, respectively.

4.2. Experimental Conditions

We evaluate the heading estimation accuracy of the proposed method for each model. We also evaluate and compare RMPCA[7] and HAM[8], based on horizontal acceleration in GCS, and IONet[9], based on deep learning. First of all, we split the dataset into three subsets for training, validation, and testing. However, we allocated all data of the hourglass-shaped courses for training due to limited data available. Then, we conducted grid search using the training and validation subsets to determine the hyperparameters. Finally, we performed estimation at 100 (Hz) on the testing subset and evaluated the results.

The proposed method trains the model to output the mean heading over a certain period. However, we treat it as the estimated heading at the central time during evaluation. We used our implementation of IONet based on the original paper because it is not publicly available. IONet sequentially calculates headings by integrating estimated heading displacements, so we provided the ground truth as the initial heading. The evaluation metrics are MSE of heading cosine and sine. The MSE value e takes a minimum of 0 when the estimated headings $\hat{\theta}^i$ and the ground truth θ^i coincide for all time $1 \leq i \leq N$ and a maximum of 2 when they differ by

Table 5
Mean squared errors of heading cosine and sine

		Forwards	Rectangle Sideways	Backwards	Straight line	All
Proposed	CNN	**0.08**	0.21	0.32	0.09	0.11
	BiLSTM	0.24	0.14	0.34	0.09	0.12
	DualCNN-LSTM	0.21	**0.10**	0.17	**0.07**	**0.09**
	DualCNN-Transformer	0.15	0.12	**0.14**	0.08	**0.09**
RMPCA		0.18	0.93	1.56	0.67	0.69
HAM		0.24	0.23	1.52	0.33	0.37
IONet		1.05	0.82	1.30	1.06	1.04

180 (deg).

$$e = \frac{1}{2N}\sum_{i=1}^{N}\left(\left(\cos(\hat{\theta}^i) - \cos(\theta^i)\right)^2 + \left(\sin(\hat{\theta}^i) - \sin(\theta^i)\right)^2\right) \tag{10}$$

4.3. Results and Discussions

Table 5 summarizes MSE for each course shape and gait. Fig. 6 presents the estimated headings for some test data by the proposed method on the left and the comparative methods on the right. Fig. 7 shows the corresponding trajectories based on the actual walking speeds computed from the ground truth trajectories. To begin with, we discuss the effects of model structure on estimation accuracy and gait robustness on the basis of the evaluation results of the proposed method. The CNN model was very accurate for forward walking but not stable for other gaits. It seems overly optimized for forwards, the most common gait in the dataset. Besides, feature timescales should differ depending on gaits, and the present CNN model may not adequately perceive differences in gaits. The receptive field of the third convolutional layer is 37, indicating that this model estimates headings by combining approximately 0.4 seconds of features. Extending the receptive field by increasing the kernel strides or the number of layers may improve accuracy for various gaits. The BiLSTM model could estimate heading correctly to some extent for all gaits but had lower accuracy compared to other models. A possible reason is that this model cannot sufficiently grasp local features due to the absence of feature extraction at convolutional layers.

The DualCNN-LSTM model achieved high accuracy overall and estimated headings robustly for every gait. It seems to successfully consider a wide range of features through the different-sized convolutional layers and the LSTM layer. The DualCNN-Transformer model achieved high accuracy as well. Incidentally, the quantity of its weight parameters is less than half of that in the DualCNN-LSTM model due to the lower number of channels in convolutional layers. From these observations, we have concluded that extracting local features, in which models like CNN excel, is crucial for boosting estimation accuracy. At the same time, extracting long-term features, for which models like LSTM and Transformer are suitable, improves robustness to

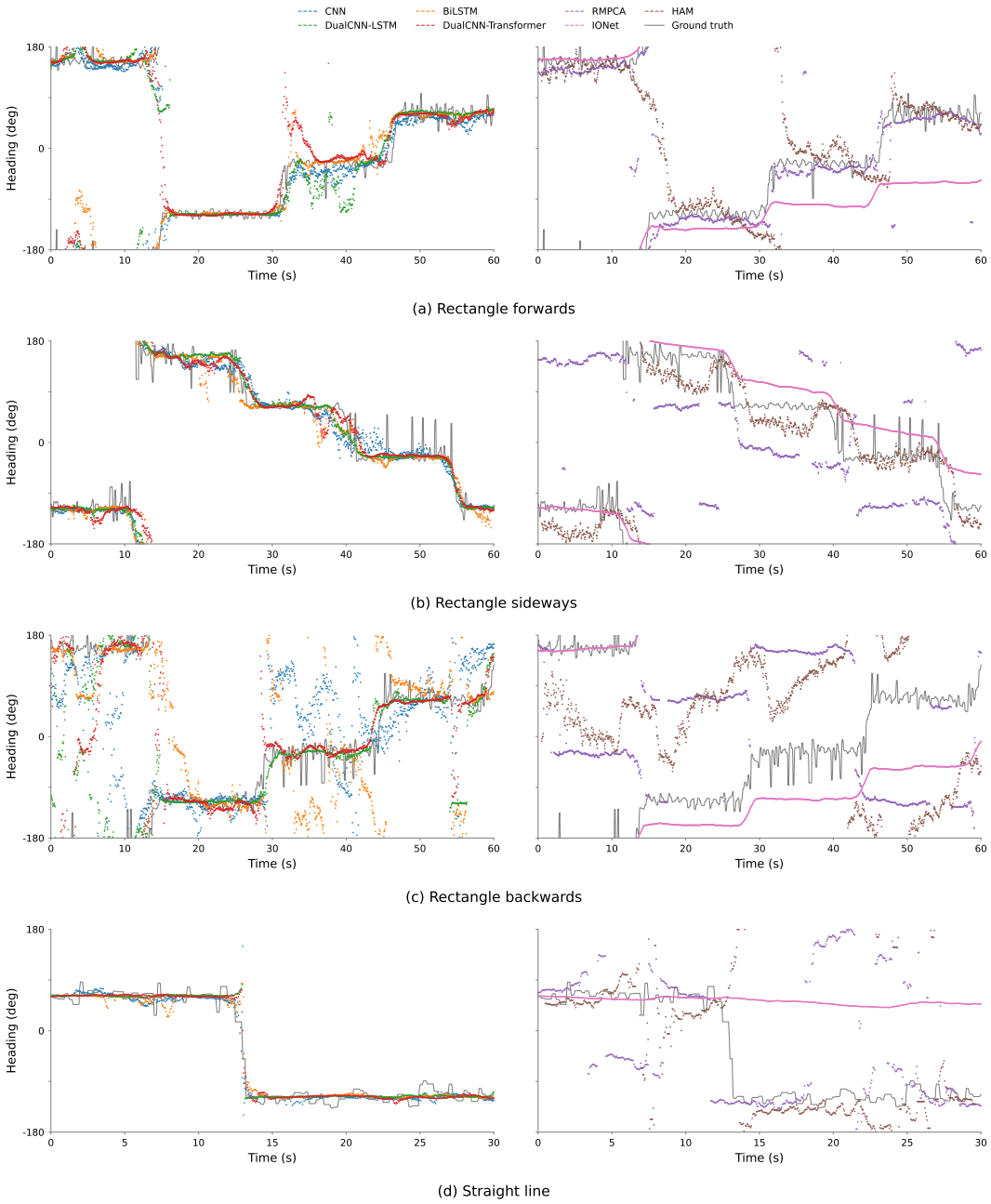


Figure 6: Instances of estimated headings.

different gaits.

Next, we examine each comparative method. RMPCA was satisfactory accurate for forward walking but experienced a significant decrease in accuracy for sideways and backwards. It tends to cause errors of approximately ± 90 (deg) for sideways and ± 180 (deg) for backwards, as

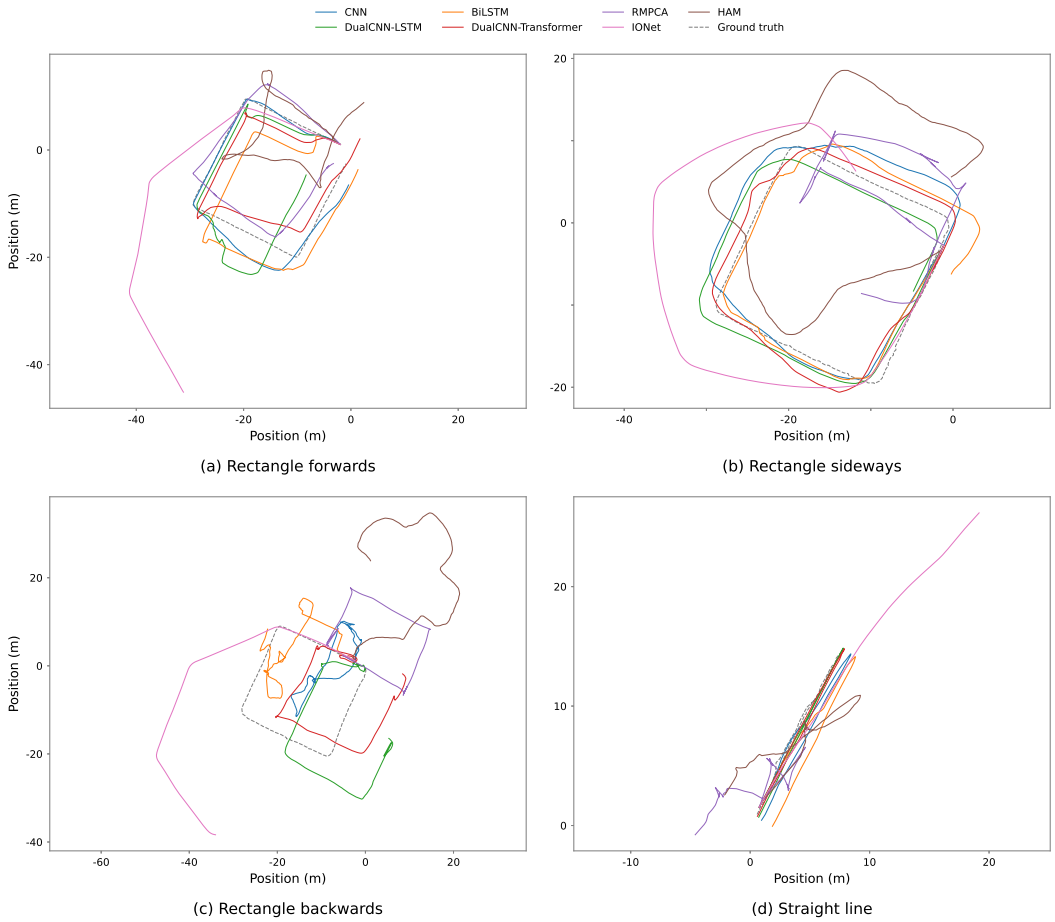


Figure 7: Instances of estimated trajectories.

seen in Fig. 6(b)(c). The principal component of horizontal acceleration is seemingly associated with the anterior-posterior direction of the body rather than the heading direction. HAM lost accuracy in backward walking. Additionally, focusing on the cornering movements, HAM occasionally estimated them as the opposite turns, shown around 15 seconds in Fig. 6(a) and Fig. 7(a). It is presumed that HAM cannot recognize heading changes since it does not consider the temporal order of input data. IONet precisely tracked headings during the spans with minimal heading changes, but the errors increased at the cornering movements, as shown in Fig. 6 and 7. This behavior is attributed to its sequentially heading estimation, accumulating errors over time. The model tends to estimate heading displacements smaller, so increasing the proportion of data with heading changes may improve accuracy. Moreover, IONet often failed to detect the heading changes of approximately 180 (deg) on the straight-line courses. This problem could be solved by training the model to output heading displacements as vectors, similar to the proposed method.

Through this experiment, we confirmed that the proposed method improved gait robustness compared to RMPCA and HAM. We also found the DualCNN-LSTM and DualCNN-Transformer models, which consider features with a wide range of timescales, were especially effective. Furthermore, we observed the advantages of the proposed method over IONet not only in the separation of orientation estimation and heading estimation and no need for initial heading but also suitability for long-duration positioning and robustness to heading changes of 180 (deg). However, there is room for improvement in estimation accuracy for forward walking, which was not much different from the existing methods.

5. Summary

This paper focused on heading estimation for PDR with smartphones. The existing methods have limitations regarding gait robustness and calculation interpretability. Therefore, we proposed an approach that inputs horizontal acceleration in GCS, calculated from acceleration, gravitational acceleration, and rotation vector, into a deep neural network model to estimate headings. Additionally, we created a dataset consisting of forward, sideways, and backward walking to evaluate gait robustness. We prepared four models: CNN, BiLSTM, DualCNN-LSTM, and DualCNN-Transformer, and evaluated the heading estimation accuracy in comparison with RMPCA, HAM, and IONet. As a result, the proposed method outperformed in gait robustness, especially when using the DualCNN-LSTM and DualCNN-Transformer models. Further improvement in estimation accuracy and expansion of the dataset are future challenges.

Acknowledgments

This work is partially supported by JSPS KAKENHI (JP22K18422), NEDO (JPNP23003), NICT (22609), and TRUSCO Nakayama Corporation.

References

[1] R. Harle, A survey of indoor inertial positioning systems for pedestrians, IEEE Communications Surveys & Tutorials 15 (2013) 1281–1293. doi:10.1109/SURV.2012.121912.00075.

[2] M. Alzantot, M. Youssef, Uptime: Ubiquitous pedestrian tracking using mobile phones, in: 2012 IEEE Wireless Communications and Networking Conference (WCNC), 2012, pp. 3204–3209. doi:10.1109/WCNC.2012.6214359.

[3] I. Klein, O. Asraf, Stepnet-deep learning approaches for step length estimation, IEEE Access 8 (2020) 85706–85713. doi:10.1109/ACCESS.2020.2993534.

[4] T. Yoshida, J. Nozaki, K. Urano, K. Hiroi, T. Yonezawa, N. Kawaguchi, Gait dependency of smartphone walking speed estimation using deep learning (poster), in: Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys '19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 641–642. URL: https://doi.org/10.1145/3307334.3328667. doi:10.1145/3307334.3328667.

[5] N. Kawaguchi, J. Nozaki, T. Yoshida, K. Hiroi, T. Yonezawa, K. Kaji, End-to-end walking speed estimation method for smartphone pdr using dualcnn-lstm., in: IPIN (Short Papers/Work-in-Progress Papers), 2019, pp. 463–470.

[6] W. Kang, Y. Han, Smartpdr: Smartphone-based pedestrian dead reckoning for indoor localization, IEEE Sensors Journal 15 (2015) 2906–2916. doi:10.1109/JSEN.2014.2382568.

[7] Z.-A. Deng, G. Wang, Y. Hu, D. Wu, Heading estimation for indoor pedestrian navigation using a smartphone in the pocket, Sensors 15 (2015) 21518–21536. URL: https://www.mdpi.com/1424-8220/15/9/21518. doi:10.3390/s150921518.

[8] R. Ban, K. Kaji, K. Hiroi, N. Kawaguchi, Indoor positioning method integrating pedestrian dead reckoning with magnetic field and wifi fingerprints, in: 2015 Eighth International Conference on Mobile Computing and Ubiquitous Networking (ICMU), 2015, pp. 167–172. doi:10.1109/ICMU.2015.7061061.

[9] C. Chen, X. Lu, A. Markham, N. Trigoni, Ionet: Learning to cure the curse of drift in inertial odometry, Proceedings of the AAAI Conference on Artificial Intelligence 32 (2018). URL: https://ojs.aaai.org/index.php/AAAI/article/view/12102. doi:10.1609/aaai.v32i1.12102.

[10] C. Chen, P. Zhao, C. X. Lu, W. Wang, A. Markham, N. Trigoni, Deep-learning-based pedestrian inertial navigation: Methods, data set, and on-device inference, IEEE Internet of Things Journal 7 (2020) 4431–4441. doi:10.1109/JIOT.2020.2966773.

[11] S. Ha, J.-M. Yun, S. Choi, Multi-modal convolutional neural networks for activity recognition, in: 2015 IEEE International Conference on Systems, Man, and Cybernetics, 2015, pp. 3017–3022. doi:10.1109/SMC.2015.525.

[12] Y. Zhao, R. Yang, G. Chevalier, X. Xu, Z. Zhang, Deep residual bidir-lstm for human activity recognition using wearable sensors, Mathematical Problems in Engineering 2018 (2018) 7316954.

[13] Y. Shavit, I. Klein, Boosting inertial-based human activity recognition with transformers, IEEE Access 9 (2021) 53540–53547. doi:10.1109/ACCESS.2021.3070646.

[14] Q. Wang, H. Luo, L. Ye, A. Men, F. Zhao, Y. Huang, C. Ou, Pedestrian heading estimation based on spatial transformer networks and hierarchical lstm, IEEE Access 7 (2019) 162309–162322. doi:10.1109/ACCESS.2019.2950728.