Deep-Learning-Based Inertial Odometry for Pedestrian Tracking Using Attention Mechanism and Res2Net Module

Boxuan Chen*, Ruifeng Zhang†, Shaochu Wang‡, Liqiang Zhang†, and Yu Liu†

†School of Microelectronics, Tianjin University, Tianjin 300072, China
‡Tianjin Institute of Surveying, Mapping Co., Ltd., Tianjin 300160, China
*Member, IEEE

Abstract—Pedestrian dead reckoning is a challenging task due to the low-cost inertial sensor error accumulation. Recent research has shown that deep learning methods can achieve impressive performance in handling this issue. In this letter, we propose inertial odometry using a deep-learning-based velocity estimation method. The deep neural network based on Res2Net modules and two convolutional block attention modules is leveraged to restore the potential connection between the horizontal velocity vector and raw inertial data from a smartphone. Our network is trained using only 50% of the public inertial odometry dataset (RoNIN) data. Then, it is validated on the RoNIN testing dataset and another public inertial odometry dataset (OXIOD). Compared with the traditional step-length and heading-system-based algorithm, our approach decreases the absolute translation error (ATE) by 76%–86%. In addition, compared with the state-of-the-art deep learning method (RoNIN), our method improves its ATE by 6%–31.4%.

Index Terms—Sensor applications, attention mechanism, deep learning, inertial navigation, localization.

I. INTRODUCTION

Whether in consumer-level services or professional service scenarios, there are fast and accurate indoor positioning requirements, including shopping guides in shopping malls, parking lot car search, human–computer interaction, and crowd monitoring. Many methods rely on communication networks, for example, a WLAN-based approach is used to statistically analyze the signal strength fingerprint of buildings [1], [2]. However, all these methods rely on external signals, and the results are not stable. While pedestrian dead reckoning (PDR) technology [3] only relies on the inertial measurement unit (IMU), its superior flexibility and portability make it attract increasing attention as a practical method.

The PDR approach commonly includes the inertial navigation system (INS) and step-length and heading system (SHS) methods. INS [4] is based on Newtonian mechanics to determine the position at each moment. However, its errors accumulate very rapidly. By contrast, the SHS algorithm [5] measures the length and heading of each step and then updates the current position. Its error accumulation characteristic is better than that of INS. To mitigate the position drift caused by error accumulation, many works integrate IMUs with other sensors to improve accuracy, such as the visual-inertial sensor [6], the WIFI-integrated inertial odometer, etc. Some studies have reduced drift using step length without the help of other sensors, based on prior knowledge of human walking. One of the methods is zero velocity updates (ZUPT) [4], but ZUPT requires a foot-mounted sensor to achieve high accuracy. Another category is step counting [7], which does not require the sensor to be linked to the foot, but it needs many parameters to be debugged and is not applicable in daily life.

Deep-learning-based methods do not require manual parameters during testing and transform the problem of inertial navigation into a continuous time-series learning task. RIDI [8] first combines inertial navigation with machine learning to estimate position by learning linear acceleration and angular velocity regression velocity vectors. Then, the linear acceleration is corrected by linear least squares. IONet [9] turns the inertial navigation problem into a time-series deep learning problem for the first time. The position is determined directly by the speed and direction of the network regression without using the traditional integration method, and the uncertainty is regressed in their subsequent work [10]. Yan et al. [11] finish an extensive and pose-rich collection of accurate trajectories under natural human motion and propose three new state-of-the-art models (RoNINs) that ultimately regress the position. The present work demonstrates that the deep learning model can infer motion trajectories from original IMU data. IONet, RoNIN, and other networks get better results in inferring trajectories directly, suggesting the possibility of deep learning localization. We continue to study how to improve its accuracy based on the previous work. This letter constructs a deep learning-based inertial odometry for pedestrian tracking, and our contributions can be summarized as the following.

1) We build a novel deep-learning-based inertial odometry for pedestrian tracking with a velocity regression neural network and integrate velocities to generate positions.
2) Two convolutional block attention modules (CBAMs) and several Res2Net modules compose the velocity regression network to enhance feature extraction and representation with fine granularity.

II. METHOD

We design a velocity estimation network and then integrate velocities to generate positions.
The proposed network consists of three main structures. The first input module contains the parts from a one-dimensional convolutional layer (Conv1d) to a maximum pooling layer (MaxPool). This part aims to extract the features of acceleration $a$ and angular rate $\omega^\alpha$ and reduce the dimension of the features. The second part is the attention mechanism module (CBAM), which consists of the channel attention module (CAM) and the spatial attention module (SAM). Through the attention module, the feature expression can be improved. The third part is LayerX ($X = 1, 2, 3, 4$), composed of several Res2Net modules.

The feature representations learned in the input module move to Layer1 that consists of several Res2Net modules. As shown in Fig. 1, three Res2Net modules are connected in series to form Layer1. Layer2, Layer3, and Layer4 are similar in structure to Layer1. The only difference is the number of Res2Net modules. We follow the original design of ResNet50 to set the number of Res2Net modules in each Layer to 3, 4, 6, and 3. The features pass through a network of three layers in series and then move to a CBAM. Then, the processed features can be projected into the velocity vector after Layer4, a CBAM, and a fully connected (FC) layer. We make use of the mean square error (MSE) as the loss function during training, and the MSE loss is defined as

$$\ell = \frac{1}{n} \sum_{i=1}^{n} (\|\hat{v}_i - \tilde{v}_i\|^2 + \|\hat{\omega}_i - \tilde{\omega}_i\|^2)$$

where $\hat{v}_i$ and $\hat{\omega}_i$ are the output velocities of the network at the $i$th moment, $\tilde{v}_i$ and $\tilde{\omega}_i$ are the corresponding ground truth velocities, and $n$ is the total number of data in the training set.

2) Res2Net Module: The Res2Net module structure is a new backbone architecture [13]. It increases the features dimension and reveals a new dimension, namely scale, which increases the range of perceptual fields for each network layer. In this letter, we replace its two-dimensional convolutions with 1-D convolutions to handle the continuous time-series learning problem. The detailed structure of Res2Net is shown in Fig. 1.

In the Res2Net module structure, features split into four uniform segments, X1, X2, X3, and X4, will pass through a separate convolution layer. The features of the first segment passing through the convolution layer will add to the features of the second segment before entering the second convolution layer. This process can be considered as another residual operation in the residual block. We end up with four segments of features, Y1, Y2, Y3, and Y4, which are fused with the features through a fusion gate. This module allows better extraction of global and local information.

Every X adds up with the previous features before the convolution operation so that all the information of previous features can be utilized. Therefore, the module can get a larger perceptual field. The experiments demonstrate that our idea is practicable.

3) Attention Mechanism Module: The CBAM [14] is a simple and effective feed-forward convolutional neural network attention module that infers the attention map by channel and spatial dimensions. The attention map is multiplied by the features to get the adaptive feature refinement. This module can enhance the practical features and reduce the noise to help the network learn the relationship between IMU data and velocity more effectively. We adapt two CBAMs to obtain the weighted features. The first CBAM is put between Layer3 and Layer4. The influence of the CBAM placement on the network performance is discussed in Section III-C. The second CBAM is placed before the final FC layer to further optimize the processed weighted features.

### B. Position Estimation

The velocity estimation network described in the previous section returns the velocity at each moment, and the displacement can be obtained by integrating the velocity. In the case of the known initial position $(p_{x0}, p_{y0})$, the global position can be written as

$$\begin{align*}
p_{xi} &= p_{x0} + \sum_{i=1}^{n} v_{xi} \Delta t & 0 < i \leq n \\
p_{yi} &= p_{y0} + \sum_{i=1}^{n} v_{yi} \Delta t & 0 < i \leq n
\end{align*}$$

where $v_{xi}$ and $v_{yi}$ are the velocities at the moment $i$ of the velocity estimation network regression. $n$ is the total time of the trajectory, and $\Delta t$ is the sample time interval.

### III. EXPERIMENTS

#### A. Dataset

We evaluate our network on the OXIOD [15], RIDI [8], and RoNIN [11] datasets, which are described as follows.
OXIOD is an inertial odometry dataset with different device placement positions (handheld, in the pocket, in the handbag, and on a trolley). We select three different device placements in pedestrian tracking to evaluate our work.

RIDI is a dataset consisting of IMU sensor measurements and 3-D motion trajectories across multiple human subjects and multiple device placements.

RoNIN is another inertial odometry dataset with more data, more human subjects, and more realistic device placement attitudes than OXIOD. For security reasons, the dataset is only published for 50% of the data, so we use 50% of the data to evaluate our work. We divide 80% of the subjects’ dataset into training, validating, and testing subsets, and the test set in this part is the unseen subjects test set. The remaining 20% is used to test the generalization ability of the model for unknown subjects, the test set in this part is the unseen subjects test set.

B. Metrics Definitions

In order to evaluate the performance of our framework, we apply the following metrics for data of length $n$.

1) Absolute Translation Error (ATE): ATE is defined as the root mean square error between the predicted difference between the whole trajectory and the ground truth trajectory

$$\text{ATE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \| p_i - \hat{p}_i \|^2}. \quad (4)$$

2) Relative Translation Error (RTE): RTE is defined as the root mean square error between the predicted difference of the trajectory and the difference between the ground truth trajectory over a fixed time interval. We set 1 min in our evaluation. The RTE serves to calculate the difference in the amount of position shift and is suitable for estimating the drift of the system

$$\text{RTE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \| p_i + \Delta t - p_i - (\hat{p}_i + \Delta t - \hat{p}_i) \|^2}. \quad (5)$$

C. Training Protocol

1) Model Training: We implement the structure of our model in Pytorch and train using the Adam optimizer [16] on an Nvidia RTX 3070 GPU with 8 GB memory. We use the initial learning rate of 0.001 for training with a batch size of 128, the learning rate decreases by 0.1 for 100 epochs, and the dropout layer is adopted to avoid network overfitting effectively. Considering the computational complexity of our algorithm, the training time cost for training an epoch is 350 s, and the inference time for each model is 1 ms.

D. Evaluations

1) Performance of Different CBAM Placement: Table 1 discusses the effect of the first CBAM placements on positioning performance. p1, p2, p3, and p4 represent different placements of the module. p1 is in front of Layer1, p2 between Layer1 and Layer2, p3 from Layer2 and Layer3, and p4 between Layer3 and Layer4. According to the results, the module performs best at p4, so we place it between Layer3 and Layer4.

2) Ablation Studies: Table 2 shows the ablation study on the RoNIN dataset, demonstrating the effectiveness of the Res2Net and CBAM by toggling these two features. We use the initial ResNet50 without Res2Net and CBAM as the baseline. Our conclusion is that Res2Net and CBAM improve ATE and RTE in general, while the latter seems to have a larger impact. In particular, ATE and RTE have the lowest errors when both features are used simultaneously.

3) Results and Analysis: Table 3 shows our main results. PDR and IONet perform well on the OXIOD dataset compared to other methods. They cannot be accommodated in the RIDI and RoNIN datasets, which have more complex natural motion data for the device. Table 3 shows that our method achieves the best ATE and RTE on most datasets. However, our method is the second best in three cases. RoNIN-ResNet can get a smaller ATE than our method on the OXIOD dataset. RoNIN-TCN achieves the best RTE on the RoNIN unseen dataset and the best ATE on the RIDI dataset. Although our method is the second
In this letter, we propose a Res2Net module and CBAM-based deep learning neural network to regress the 2-D velocity, and then, the position can be estimated. Experimental results show that our method improves the ATE on the RoNIN dataset by 76.0%–86.0% over the traditional PDR algorithm and improves the ATE by 6.5%, 31.4%, and 23.7% compared to the state-of-the-art deep-learning-based inertial odometry (RoNIN-ResNet, RoNIN-LSTM, RoNIN-TCN), respectively.

**ACKNOWLEDGMENT**

This work was supported in part by the Yunnan Major Science and Technology Special Plan under Grant 202002AD008001 and in part by the National Natural Science Foundation of China under Grant 61771338.

**REFERENCES**

[1] Z. Yang, C. Wu, and Y. Liu, “Locating in fingerprint space: Wireless indoor localization with little human intervention,” in Proc. 18th Annu. Int. Conf. Mobile Comput. Netw., 2012, pp. 269–280.

[2] W. S. A. Au et al., “Indoor tracking and navigation using received signal strength and compressive sensing on a mobile device,” IEEE Trans. Mobile Comput., vol. 12, no. 10, pp. 2050–2062, Oct. 2013.

[3] R. K. Harle, “A survey of indoor inertial positioning systems for pedestrians,” IEEE Commun. Surv. Tut., vol. 15, no. 3, pp. 1281–1293, Jul.–Sep. 2013.

[4] E. Foslin, “Pedestrian tracking with shoe-mounted inertial sensors,” IEEE Comput. Graph. Appl., vol. 25, no. 6, pp. 38–46, Nov./Dec. 2005.

[5] Q. Tian, Z. A. Salcic, K.-I. K. Wang, and Y. Pan, “A multi-mode dead reckoning system for pedestrian tracking using smartphones,” IEEE Sensors J., vol. 16, no. 7, pp. 2079–2093, Apr. 2016.

[6] S. Leutenegger, S. Lynen, M. Bosse, R. Y. Siegwart, and P. T. Furgale, “Keyframe-based visual–inertial odometry using nonlinear optimization,” Int. J. Robot. Res., vol. 34, pp. 314–334, 2015.

[7] A. Brajdic and R. K. Harle, “Walk detection and step counting on unconstrained smartphones,” in Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput., 2013, pp. 225–234.

[8] H. Yan, Q. Shan, and Y. Furukawa, “RIDI: Robust IMU double integration,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 621–636.

[9] C. Chen, C. X. Lu, A. Markham, and A. Trigoni, “IONet: Learning to cure the curse of drift in inertial odometry,” in Proc. AAAI Conf. Artif. Intell., 2018, pp. 6468–6476.

[10] C. Chen, X. Lu, J. Wahlstrom, A. Markham, and N. Trigoni, “Deep neural network based inertial odometry using low-cost inertial measurement units,” IEEE Trans. Mobile Comput., vol. 20, no. 4, pp. 1351–1364, Apr. 2021.

[11] H. Yan, S. Herath, and Y. Furukawa, “Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 3146–3152.

[12] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 770–778.

[13] S. Gao, M.-M. Cheng, K. Zhao, X. Zhang, M.-H. Yang, and P. H. S. Torr, “Res2Net: A new multi-scale backbone architecture,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 2, pp. 652–662, Feb. 2021.

[14] S. Woo, J. Park, J.-Y. Lee, and I.-S. Kweon, “CBAM: Convolutional block attention module,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 3–19.

[15] C. Chen, P. Zhao, C. X. Lu, W. Wang, A. Markham, and A. Trigoni, “Deep-learning-based pedestrian inertial navigation: Methods, data set, and on-device inference,” IEEE Internet Things J., vol. 7, pp. 4431–4441, 2020.

[16] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” CoRR, vol. abs/1412.6980, 2015.

[17] Q. Tian, Z. A. Salcic, K.-I. K. Wang, and Y. Pan, “An enhanced pedestrian dead reckoning approach for pedestrian tracking using smartphones,” in Proc. IEEE 10th Int. Conf. Intell. Sensors, Sensor Netw., Inf. Process., 2015, pp. 1–6.