

Article

EDIN: An Enhanced Deep Inertial Navigation Method for Pedestrian Localization

Jin Wu ¹, Gong Cheng ¹ and Jianga Shang ^{1,2,3,*}

¹ School of Computer Science, China University of Geosciences, Wuhan 430078, China; wujineikom@cug.edu.cn (J.W.); 2202410402@cug.edu.cn (G.C.)

² Engineering Research Center of Natural Resource Information Management and Digital Twin Engineering Software, Ministry of Education, Wuhan 430074, China

³ National Engineering Research Center for Geographic Information System, Wuhan 430078, China

* Correspondence: jgshang@cug.edu.cn

Abstract

Indoor pedestrian navigation tasks, as a key part of smart cities and navigation services, face dual challenges of accuracy and cost under complex building environments. Currently, neural inertial navigation is at the vanguard of current research in indoor pedestrian navigation, and existing related studies have achieved positive results. However, the exploration of deep learning solutions is still not sufficient, mainly reflected in the lack of explorations of model training configurations. Based on testing results under different deep learning schemes, this paper proposes EDIN, an enhanced deep inertial navigation approach. This method benefits from a proprietary neural network based on ResNeXt with Convolutional Block Attention Module (CBAM) to predict the relationship between inertial data and motion trajectory. Compared to existing projects, this paper also makes improvements in the model training process, thereby improving the predictive effect of the trained model. Specifically, this paper innovatively uses Logcosh as the loss function and combines data rotation and additional noise as data augment methods. To assess EDIN's performance, extensive tests were conducted using three publicly available datasets: RoNIN, OXIOD, and RIDI. The results clearly indicate EDIN's superior performance relative to other neural inertial navigation systems. Notably, localization accuracy improved significantly, with an average enhancement of 16.06% compared to the RoNIN-ResNet method.

Keywords: indoor localization; inertial navigation; deep learning; pedestrian dead reckoning

1. Introduction

The study of smartphone-based navigation has emerged as a prominent research focus, driven by the escalating demand for navigation services and the pervasive adoption of smartphones. In outdoor environments, smartphones leverage signals from Global Navigation Satellite Systems (GNSSs) to enable precise navigation. However, in urban environments, obstructions often induce multipath interference, compromising the accuracy of GPS-based indoor navigation systems [1]. To enhance navigation precision, integrating supplementary data streams such as WiFi, Bluetooth beacons, visual inputs, and inertial data has become imperative [2]. Existing indoor navigation techniques, however, are typically constrained by environmental dependencies. For instance, WiFi and Bluetooth beacon-based methods are effective only in environments equipped with the requisite infrastructure [3], while visual navigation systems necessitate adequate lighting conditions for clear image capture, limiting their operational applicability [4]. These constraints



Academic Editor: Young-Joo Suh

Received: 22 February 2026

Revised: 14 March 2026

Accepted: 17 March 2026

Published: 20 March 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

pose substantial challenges to the broad deployment of such localization methodologies. In contrast, navigation systems utilizing inertial data circumvent the reliance on images or wireless signals, thereby overcoming limitations related to illumination and beacon infrastructure. Consequently, inertial sensor integration has emerged as a critical enabler for pedestrian navigation, bolstering localization capabilities irrespective of environmental conditions [5].

Traditional inertial navigation approaches primarily rely on the Pedestrian Dead Reckoning (PDR) algorithm [6], which utilizes data from an Inertial Measurement Unit (IMU) to estimate displacement and orientation. PDR algorithms typically employ double-integration and zero-velocity update (ZUPT) techniques [7]. However, the calculation of velocity and position using this methodology results in cumulative navigation errors over time, necessitating high-precision sensors to mitigate inaccuracies [8]. Moreover, ZUPT requires the IMU to be affixed to the user's foot, rendering it impractical for smartphones equipped with an integrated IMU [9].

In addition to PDR, traditional inertial navigation also includes Strapdown Inertial Navigation Systems (SINS) that rely on integration algorithms and filtering techniques (e.g., EKF) [10] for orientation estimation. Derivative algorithms such as ZARU [11] and MARU [12,13] have been developed to reduce error drift [14,15], but they all require strict sensor calibration and foot-mounted IMUs, leading to poor practicality [16]. To address these limitations, early data-driven inertial navigation methods have been proposed, such as Hidden Markov Models (HMMs) [17] and multi-layer perceptrons (MLPs) [18] for motion parameter estimation, and RIDI [19] that combines sensor fusion with data-driven strategies for smartphone-based navigation. However, these early data-driven methods either focus on single-joint movements with limited applicability or still suffer from cumulative errors due to reliance on integration algorithms.

Recent advancements in inertial navigation have increasingly incorporated neural networks, although the domain of neural network-based inertial navigation (NIL) remains relatively nascent [20]. Notwithstanding, neural inertial navigation methodologies have demonstrated promising developments. For example, IONet [21] employs a two-layer Convolutional Neural Network (CNN) in conjunction with a two-layer bidirectional Long Short-Term Memory (LSTM) [22] unit to process raw data from a nine-axis IMU, enabling the estimation of displacement and direction. Building on this foundation, researchers at RoNIN [23] have developed neural network architectures that integrate ResNet, LSTM, and Temporal Convolutional Network (TCN) frameworks. These models process normalized coordinates to generate velocity vector outputs, relying solely on accelerometer and gyroscope data while excluding inclination and azimuth angles. This exclusion effectively mitigates directional drift errors. Furthermore, these models obviate the need for initial coordinate system inputs during trajectory estimation. Despite their theoretical potential, these methods face limitations, including insufficient model testing and inaccuracies in displacement direction estimation, which collectively result in suboptimal navigation performance [24–27]. While neural network technologies have driven significant advancements in other scientific disciplines [28], the field of inertial indoor navigation remains deficient in rigorous experimental validation [29,30]. Although multiple projects have conducted research on deep inertial navigation, there are still parts that have not yet been studied. Research on deep networks is mostly limited to testing the construction of models in order to pursue optimal model solutions. However, due to the variability of sensor noise, device placement, and changes in environmental conditions, inertial measurement has inaccuracies. Therefore, this research approach has limitations on the generalization of deep networks in different conditions and users. Therefore, research should be conducted on

other aspects of deep networks, including innovative training methods, data augmentation strategies, and optimization techniques.

To address the aforementioned limitations, this study takes EDIN (Enhanced Deep Inertial Navigation) as the core research method and designs a dedicated inertial navigation system for smartphone-based indoor pedestrian localization, with all subsequent research and experiments centered on this framework.

In this study, we propose EDIN, a novel method designed to enhance the accuracy of indoor inertial navigation. The main contributions of this work can be summarized as follows:

- A novel neural network architecture is proposed using ResNeXt [31] and the Convolutional Block Attention Module (CBAM) to enhance feature representation, which improves the navigation accuracy.
- An enhanced training pipeline for deep inertial navigation models is proposed through refined data augmentation and loss function, which helps us improve the model robustness.
- Extensive evaluations are conducted on several publicly available inertial navigation datasets, demonstrating superior accuracy compared to existing methods.

The subsequent sections of this study are organized as follows: Section 2 describes the materials and methods. Section 3 describes the architecture of the EDIN system we have developed. Section 4 analyzes navigation performance and presents corresponding results. Section 5 provides a detailed discussion and the conclusions based on our findings.

2. Materials and Methods

In this section, we present our proposed indoor inertial navigation system, EDIN. Initially, we provide a succinct overview of the system architecture. Subsequently, we detail the core modules comprising the architecture, notably the neural network model and the model training method.

2.1. Inertial Navigation Problem

Inertial navigation involves estimating the user's positional information by interpreting the data obtained from IMU. IMU data consist of a sequence of measurements from accelerometers, gyroscopes, and, optionally, magnetometers, with each sensor providing data along three degrees of freedom (DOF).

Define an IMU sequence as $(\alpha, \omega) = [(\alpha_i, \omega_i), i = 1, 2, \dots, n]$, where α and ω represent the local measurements of the accelerometer and gyroscope, respectively. Orientation information is typically required before using IMU data in a deep network to align it with a common reference system. Define an orientation sequence as $q = [q_i; i = 1, 2, \dots, n]$, where $q = (x, y, z, w)$ represents the rotation vector in quaternion format. The inertial navigation problem can then be addressed by

$$\Delta p_t = f(\alpha_t, \omega_t) \quad (1)$$

$$p_t = p_{t-1} + R(q_{t-1})\Delta p_t. \quad (2)$$

$R(q)$ represents the rotation matrix dependent on q . $f(\cdot)$ is determined by the navigation method.

Subsequently, the user's location is estimated based on the corresponding location sequence $p = [p_i, i = 1, 2, \dots, n]$ derived from (α, ω) . Assuming the ground truth location

of (α, ω) is $\hat{p}$, the primary metrics used to calculate the location estimation error are the absolute trajectory error (ATE) and the relative trajectory error (RTE), as given by

$$\text{ATE} = \sqrt{\frac{1}{n} \sum_{i=1}^n \|p_i - \hat{p}_i\|^2} \quad (3)$$

$$\text{RTE} = \sqrt{\frac{1}{n} \sum_{i=1}^n \|p_{i+\Delta t} - p_i - (\hat{p}_{i+\Delta t} - \hat{p}_i)\|^2}. \quad (4)$$

The ATE quantifies the spatial proximity between the trajectory coordinates generated by the navigation algorithm and the ground truth. Meanwhile, the RTE gauges the local spatial proximity between the trajectory derived from the localization algorithm and the ground truth within a predetermined time interval, commonly set at 1 min.

2.2. Model Network

2.2.1. Model Overview

The proposed EDIN model processes IMU data through a sophisticated deep learning architecture, which is shown in Figure 1. Raw IMU time-series data first undergoes frame randomization and coordinate frame normalization in the input module. The core feature extraction leverages a ResNeXt backbone comprising four stages, each containing multiple Bottleneck Residual Blocks (BRBs) enhanced with CBAM [32]. These CBAM-equipped BRBs sequentially apply channel and spatial attention to refine feature representations. The processed features are then passed through an output module consisting of fully connected (FC) layers with ReLU activations and dropout for regularization, ultimately predicting the location change. This design effectively combines the multi-branch efficiency of ResNeXt with the adaptive feature weighting of CBAM to improve inertial navigation accuracy.

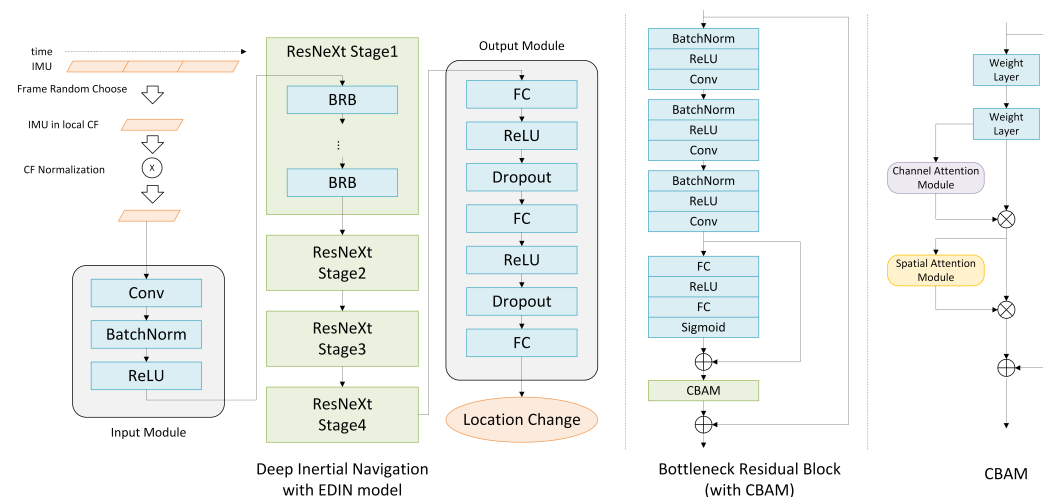


Figure 1. Architecture of the proposed EDIN model, based on ResNeXt with CBAM-integrated Bottleneck Residual Blocks (BRBs) for robust inertial sensor data processing.

The mathematical formulation of the EDIN network is as follows,

$$(\Delta x, \Delta y) = f\left(\left(\omega_{n-N:n}, \alpha_{n-N:n}\right), h_{n-N}\right). \quad (5)$$

where $f(\cdot)$ is the function defined by the neural network. ω and α represent the raw values of gyroscope and accelerometer data, respectively, which are combined as the input to the neural network. h_{n-N} is the hidden state of the GRU unit at the most recent timestamp.

The EDIN model functions by processing continuous IMU data and calculating the corresponding coordinate changes within the geodetic coordinate system. By employing

sliding windows, the model aggregates time-series IMU data, predominantly consisting of acceleration and gyroscope data. Each dataset encompasses 200 frames of IMU data. The navigation trajectory is derived through a recursive process, where coordinate values for each navigation point are sequentially retrieved from the initial coordinates.

2.2.2. Coordinate Systems for Neural Inertial Navigation

When deriving displacement distances from the network using IMU data, the initial velocity within the sensor coordinate system emerges as a potential variable derived from the original IMU data [21]. Consequently, addressing this issue necessitates the inclusion of initial velocity as an input variable. Given the user's coordinate transformation, which can be described as a polar vector denoting displacement and directional changes, the equation for calculating displacement from IMU data is articulated as follows:

$$(\Delta l, \Delta \psi) = f(v, \alpha_{est}^n, \omega_{est}^n). \quad (6)$$

Δl represents the displacement distance. $\Delta \psi$ represents the direction change. v represents the velocity vector at the beginning of navigation.

If using the above output data directly in polar coordinates, the loss function used for training neural network models can be obtained by

$$\mathcal{L} = \sum \left\| \Delta \ell - \widehat{\Delta \ell} \right\|^2 + \kappa \left\| \Delta \psi - \widehat{\Delta \psi} \right\|^2. \quad (7)$$

where $\Delta \ell$ is the estimated displacement distance, $\widehat{\Delta \ell}$ is the ground truth displacement distance, $\Delta \psi$ is the estimated heading angle change, and $\widehat{\Delta \psi}$ is the ground truth heading angle change. κ is a factor determining the weights of $\Delta \ell$ and $\Delta \psi$.

However, a challenge encountered in heading regression tasks in practical applications arises from the inherent ambiguity when the user is stationary. Typically, the ground truth heading angle of the user poses is derived computationally by analyzing the collected trajectory ground truth data. Owing to the potential measurement errors inherent in the ground truth data of the trajectory, considerable inaccuracies may arise in the calculated heading angle derived, particularly when the user is stationary or moving at low speeds. On the other hand, determining the suitable value for the proportion in (7) poses a considerable challenge. The absence of a universally applicable method for discerning this ratio necessitates an extensive array of experiments for validation, entailing a substantial investment of time and effort.

To mitigate this challenge, we use the Cartesian coordinate system scheme in results generated by neural networks. In fact, to obtain the localization trajectory, the coordinate equation can be obtained through Cartesian projection using the initial coordinate data by

$$\begin{cases} L_n^x = L_0^x + \sum_{i=0}^n \Delta L_i^x \\ L_n^y = L_0^y + \sum_{i=0}^n \Delta L_i^y \end{cases}. \quad (8)$$

(L_n^x, L_n^y) represents the current location. n represents the number of positioning times. (L_0^x, L_0^y) represents the initial coordinate. $(\Delta L_n^x, \Delta L_n^y)$ represents the coordinate change, which is expressed as follows:

$$\begin{cases} \Delta L_n^x = \Delta l_n * \cos(\psi_{n-1} + \Delta \psi_n) \\ \Delta L_n^y = \Delta l_n * \sin(\psi_{n-1} + \Delta \psi_n) \end{cases}. \quad (9)$$

Due to the fact that directional information can be obtained from

$$\psi = \tan \frac{v_y}{v_x}. \quad (10)$$

Therefore, (6) can be equivalent to

$$(\Delta L_n^x, \Delta L_n^y) = f(v, \alpha_{est}^n, \omega_{est}^n). \quad (11)$$

According to the above equation, theoretically, the two schemes can be equivalent to each other.

2.2.3. ResNeXt Module

The core feature extraction backbone of the proposed model is constructed upon the ResNeXt architecture, which represents a significant advancement over the conventional ResNet framework. The primary innovation of ResNeXt lies in the introduction of cardinality—the number of parallel transformation paths within a residual block—as an additional dimension of model design, complementing depth and width. In the proposed implementation, each ResNeXt block adopts a split–transform–merge strategy, implemented using a Bottleneck Residual Block (BRB). Specifically, the input feature map is divided into multiple low-dimensional embeddings across a predefined number of groups determined by the cardinality parameter. Each group is processed independently through a dedicated set of convolutional filters, typically implemented using grouped convolutions. The resulting feature maps from all groups are then aggregated through summation and subsequently fused with the shortcut connection to form the final residual output. This architecture enhances the network’s representational capacity and facilitates multi-branch feature learning while maintaining computational efficiency. Consequently, the ResNeXt-based backbone achieves a more favorable trade-off between accuracy and complexity compared to conventional approaches that solely increase network depth or width.

From the perspective of theoretical suitability for IMU time-series feature extraction, the grouped convolution and cardinality design in ResNeXt are highly consistent with the characteristics of 6-axis IMU signals. IMU data include 3-axis acceleration and 3-axis gyroscope, which are multi-channel time series with independent physical attributes but strong temporal correlation. Grouped convolution divides the feature extraction process into multiple parallel branches, which can implicitly decouple the learning of motion-related features and noise-related features, as well as distinguish the heterogeneous characteristics between acceleration and gyroscope. Compared to standard convolution in ResNet, cardinality as an independent design dimension avoids the cross-channel interference caused by single-branch convolution, and is more suitable for capturing multi-modal, low-dimensional, and high-sampling time-series features of IMU. Such a structure can improve feature diversity without significantly increasing computation, which is beneficial for long-sequence inertial trajectory estimation.

2.2.4. Attention Mechanism Module

CBAM [32] is a simple and effective feedforward convolutional neural network attention module that infers attention maps through channel and spatial dimensions. The attention map is multiplied by the features to refine the adaptive features. This module can enhance practical features, reduce noise, and help the network learn the relationship between IMU data and speed more effectively.

In our work, we further enhance the vanilla ResNeXt block by integrating a CBAM with BRB. Specifically, we adapt one CBAM after each ResNeXt module to obtain the weighted features. The CBAM module sequentially infers attention maps along both the

channel and spatial axes, allowing for the network to adaptively emphasize informative features and suppress less useful ones. This integration enables the model to not only capture a rich set of features through multi-branch processing, but also to intelligently refine these features by focusing on critical elements within the inertial data, leading to more robust and accurate navigation estimates.

For low-dimensional 6-axis IMU signals, the rationality of applying CBAM lies in its lightweight sequential attention structure, which avoids excessive freedom and insufficient constraints of attention maps. Unlike high-dimensional image data, IMU signals have limited channel dimension (only 6 channels), so channel attention can effectively learn the importance of acceleration and gyroscope channels without redundant parameters. The spatial attention of CBAM is naturally converted into temporal attention in IMU sequences, which focuses on key time steps such as motion switching, turning, and starting, rather than meaningless spatial regions. Since the attention weights are normalized and bounded, the attention distribution will not be too scattered or out of constraint. Therefore, CBAM is not only compatible with low-dimensional inertial signals, but also enhances the model's ability to suppress sensor noise and focus on effective motion features.

2.3. Enhanced Model Training Method for Deep Indoor Inertial Navigation

2.3.1. Overview of Training Process

For the usual training processes of deep inertial navigation model, the first step is to collect inertial sensors based on smartphone motion and corresponding trajectory data, which are usually integrated into the internal sensors of the smartphone and other external devices. Then, the data need to be preprocessed, including coordinate system transformation, to convert the corresponding coordinate system from the device coordinate system to the navigation coordinate system, as well as data augmentation to expand the data. Finally, the model is trained using the training set and validation set composed of the above data. The model training will consist of multiple rounds, each of which will result in a temporary model and corresponding prediction results from the training set. At this point, it is necessary to calculate the difference between the predicted result and the actual result, which is called the loss value, and use it to adjust the model in the next round of training. The formula for calculating the loss value is called the loss function.

Considering that relevant research has shown that adopting new model training methods can effectively improve model accuracy, we have made improvements in the training process of EDIN, which is shown in Figure 2. Specifically, in the preprocessing section, we added Gaussian-distributed white noise on top of existing data augmentation methods to further simulate real-world data. And we adopted a new Logcosh function for the loss function to further improve the performance of the model.

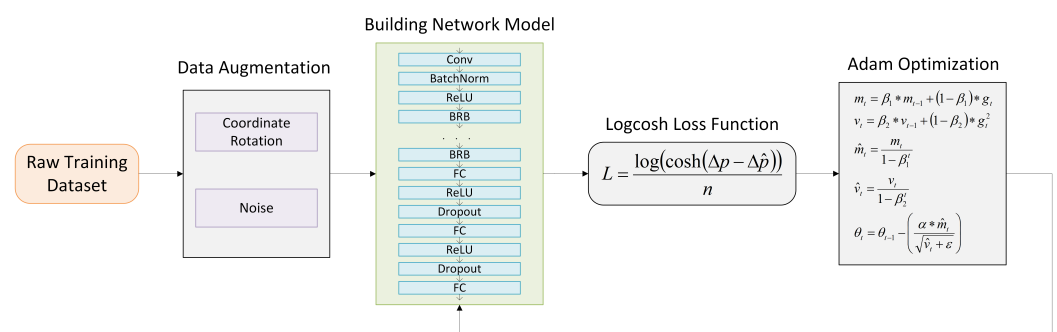


Figure 2. The training pipeline of EDIN, featuring data augmentation with noise addition and loss function with the Logcosh algorithm.

2.3.2. Data Augmentation

Due to the fact that indoor inertial navigation data still relies on manual collection, collecting extensive datasets in this field is often challenging. To address this issue, RIDI proposed the stabilized IMU coordinate system, which is obtained from the device coordinate system by aligning its Y-axis with the negative gravity direction. On the basis of RIDI, RoNIN proposed a heading-agnostic coordinate frame (HACF), which is any coordinate system where the Z-axis is aligned with gravity. In other words, we can choose any such coordinate system as long as we maintain consistency throughout the entire sequence. By using appropriate rotational representations (such as quaternions), transforming coordinates to HACF is not affected by singularities or discontinuities.

In addition to the aforementioned rotation processing, noise addition is also a commonly used data augmentation method. In fact, due to the significant noise accompanying the actual collection of IMU data, it is feasible to enhance the data by adding noise. So, we have added additional data noise based on the above scheme in EDIN.

2.3.3. Logcosh Loss Function

Mean Squared Error (MSE) is the most commonly employed loss function in the training of deep inertial navigation systems, as it computes the squared Euclidean distance between the predicted and ground-truth coordinate changes. However, in the presence of outliers with large deviations, MSE may yield excessively high loss values, potentially causing gradient explosion and degrading model robustness. Considering that outliers frequently occur in inertial data collected from low-cost sensors, it is preferable to address this issue by employing a more effective loss function. The Logcosh loss function provides a smoother alternative. For small prediction errors, it behaves similarly to MSE with quadratic growth, ensuring stable gradient updates, whereas for large errors, it approximates Mean Absolute Error (MAE) with linear growth, thereby reducing sensitivity to outliers. By integrating the advantages of both MSE and MAE, Log-Cosh achieves a balance between smooth optimization, convergence efficiency, and robustness, making it particularly suitable for regression tasks involving dynamically varying error ranges or scenarios requiring both precision and resilience.

3. Evaluations

3.1. Dataset

We conduct comprehensive evaluations on the task of position estimation using the RoNIN, OXIOD, and RIDI datasets for assessment. For each dataset, 80% is allocated for training, 10% for validation, and the remaining 10% for testing.

3.1.1. RoNIN

The RoNIN Dataset is a large-scale benchmark dataset for inertial navigation released by the Stanford University team in 2020. It contains 42.7 h of IMU-motion data collected from 100 human subjects with several Android devices, covering natural human movements such as walking, running, and stair-climbing. A 3D tracking phone (Asus Zenfone AR) was used to provide ground truth trajectories, with location drift less than 0.3 m and orientation drift less than 10 degrees. Based on whether the test sequences belong to the same group as the training set, the RoNIN dataset is divided into two subsets: RoNIN-seen and RoNIN-unseen.

3.1.2. OXIOD

The OXIOD Dataset was released by the University of Oxford team in 2018, focusing on smartphone-based IMU navigation in daily scenarios. It comprises 158 sequences with a

total distance exceeding 42 km of IMU data under various motion modes such as walking, running, and hand-held movements.

3.1.3. RIDI

The RIDI Dataset contains multiple short-duration IMU sequences covering basic motion patterns such as walking and backward walking, with ground truth velocity and position provided by motion capture devices.

3.2. Model Training Details

During the training of models, we utilized PyTorch 2.10.0, torchvision 0.25.0, tensorboardX 2.6.4, and conducted our experiments using an NVIDIA GeForce RTX 4070 with 32GB GPU memory. We used a batch size of 128 and employed an ADAM optimizer with an initial learning rate of 0.0001, reducing the learning rate by a factor of 0.1 if the validation loss did not decrease over ten epochs. The network underwent training for 500 epochs, with each epoch involving a complete pass through all training data. For EDIN, we applied dropout with a keep probability of 0.5.

As described in RoNIN [23], HACF was employed to circumvent singularity and discontinuity issues stemming from coordinate system transformations. This frame features a Z-axis perpendicular to gravity. During training, a random HACF was assigned at each step, defined by randomly rotating the ground truth trajectory within the horizontal plane. IMU data were then transformed to this HACF through device orientation and horizontal rotation. This approach effectively integrated sensor fusion into our data-driven system. During testing, we utilized the coordinate system defined by the orientation of the Android device, aligning its Z-axis with gravity.

Regarding the noise addition, we used Gaussian functions to generate white noise. Specifically, white noise with a variance of 0.1 was added to the acceleration, and white noise with a variance of 0.001 was added to the gyroscope. To further explore the impact of noise intensity on model performance, additional ablation experiments under different noise intensities were conducted. Specifically, five levels of acceleration noise and corresponding gyroscope noise were set, and the R-ResNeXt model was trained and tested under each intensity.

In the evaluations, we tested the MSE and Logcosh loss functions in the model training. We defined Δp as the estimated displacement distance and $\Delta \hat{p}$ as the ground truth displacement distance. n is the length of each set of data; for each set of 200 frames of input data, the value is $n = 200$. The calculation formulas for MSE and Logcosh are as follows:

- MSE:

$$\mathcal{L} = \frac{\sum \|\Delta p - \Delta \hat{p}\|^2}{n} \quad (12)$$

- Logcosh:

$$\mathcal{L} = \frac{\sum \log(\cosh(\Delta p - \Delta \hat{p}))}{n} \quad (13)$$

3.3. Metrics

As shown in (3) and (4), we utilized ATE and RTE as evaluation metrics. In evaluations, an initial orientation was provided by a dataset of orientation data. To mitigate potential biases in trajectory direction and ensure an accurate comparison of model performance, we employed the Iterative Closest Point (ICP) method to align the initial 5 s of the estimated trajectory with the ground truth. Note that ATE and RTE were used as a metric in meters.

3.4. Competing Methods

We selected a series of methods for comparative analysis. In the non-neural network domain, the PDR approach was adopted, while the RIDI method was excluded, as RIDI classifies different smartphone placement states, making it unsuitable for our comparative evaluation. In the neural network domain, we selected RoNIN-ResNet (R-ResNet) from the RoNIN study and its modified version RoNIN-ResNeXt (R-ResNeXt). Other methods, such as IDOL, were excluded due to either negligible differences in network performance or the unavailability of publicly accessible source code, which prevents their inclusion in our comparative evaluation. The following section outlines the characteristics and attributes of these selected comparison methods.

3.4.1. PDR

An inertial navigation system that utilizes a step-counting method. We employ a technique from [33] to detect the steps and determine the heading direction, with the step distance set at a predefined value.

3.4.2. IONet

A neural network-driven inertial navigation approach leveraging accelerometer and gyroscope data within a sliding window framework supported by the LSTM model.

3.4.3. R-ResNet

A robust neural inertial navigation network model utilizing a ResNet-based architecture. For this, only 50% data of RoNIN dataset was publicly available, so we retrained all R-ResNet models on this dataset.

3.4.4. R-ResNeXt

Similarly to the R-ResNet, the only difference was that the original ResNet module was replaced with a ResNeXt module of the same specification.

The exclusion of other neural network-based localization methods is due to minor model differences and undisclosed code. Except for the PDR method, all other models required training. To ensure a fair comparison, these models were trained under identical datasets and conditions to the EDIN model.

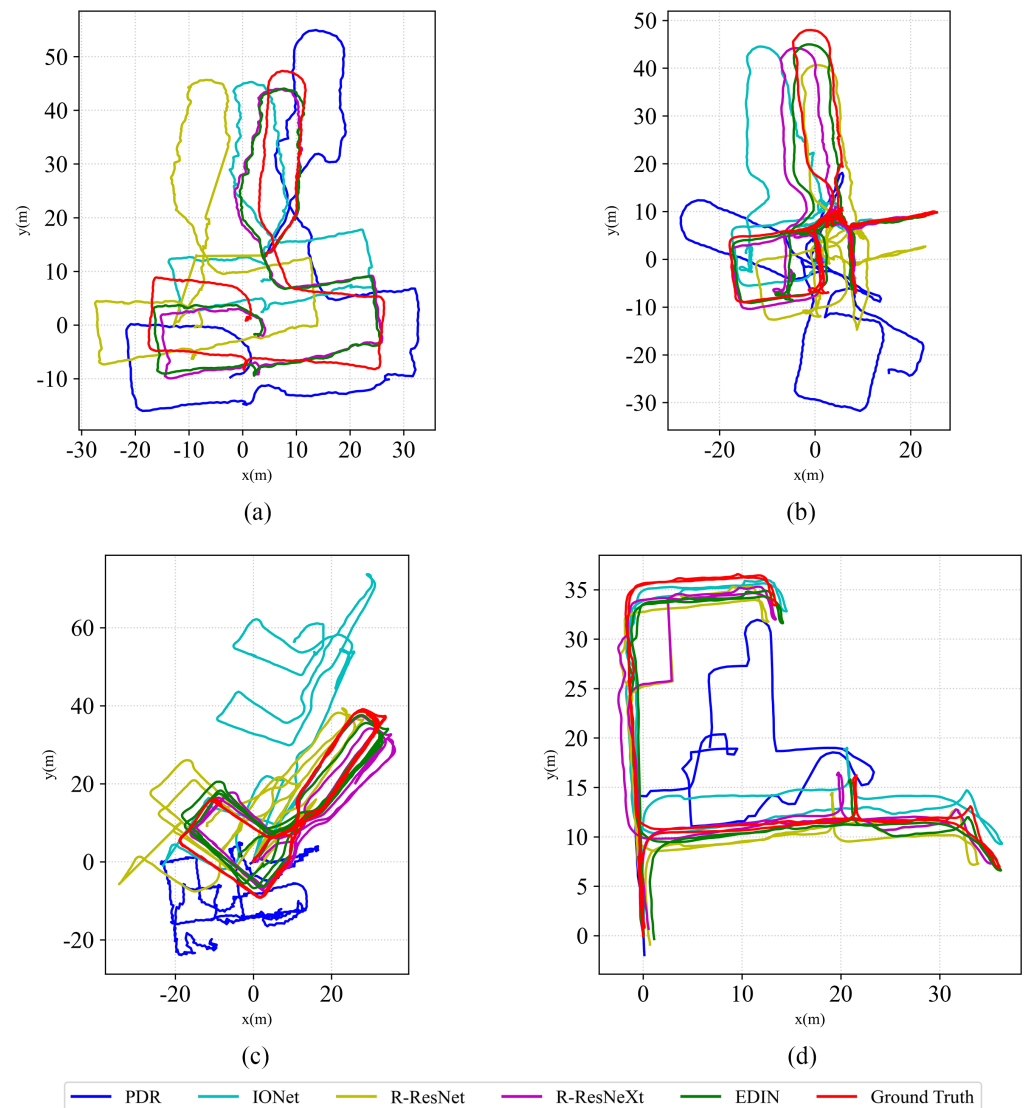
3.5. Results

3.5.1. Position Evaluations

As shown in Table 1, the location evaluation results are partially visualized in Figure 3, revealing different performance trends among evaluation methods. It can be seen that models based on ResNeXt perform better than models based on ResNet on the RoNIN, OXIOD, and RIDI datasets. The performance of the model trained by our proposed EDIN method further improved compared to the method of simply replacing it with ResNeXt. Specifically, EDIN reduces the ATE by 18.78% and RTE by 18.71% compared to R-ResNeXt on the RoNIN-seen dataset; on the more challenging RoNIN-unseen dataset, EDIN still maintains a 13.35% ATE reduction and 11.97% RTE reduction, reflecting its strong adaptability. On all three datasets, the proposed EDIN method achieves the lowest 95% percentile ATE and RTE among all comparison methods, which demonstrates its superior robustness to severe estimation errors. Consistent with the average localization results, the 95% percentile indicators also verify that EDIN can effectively suppress large tracking errors and improve the reliability of inertial navigation, even in rare and difficult scenarios.

Table 1. Localization results with tail risk evaluation (95% quantile) (ATE: m; RTE: m).

	RoNIN-Seen				RoNIN-Unseen				OXIOD		RIDI	
	ATE	RTE	95% ATE	95% RTE	ATE	RTE	95% ATE	95% RTE	ATE	RTE	ATE	RTE
PDR	26.0	23.8	47.86	35.99	23.5	23.1	38.39	38.04	4.66	3.62	11.9	13.1
IONet	7.44	4.26	10.48	5.66	15.4	7.60	27.35	10.46	2.05	2.12	4.84	4.38
R-ResNet	4.89	3.44	7.16	5.11	6.04	4.78	9.10	7.32	1.78	1.96	1.40	1.60
R-ResNeXt	4.74	3.42	7.58	5.62	5.47	4.68	8.75	7.20	1.75	1.99	1.16	1.54
EDIN	3.85	2.78	5.78	4.29	4.74	4.12	6.04	4.39	1.34	1.87	0.83	0.98

**Figure 3.** Visualization of the selected navigation assessment results. (a–c) Navigation trajectory results on the RoNIN dataset. (d) Navigation trajectory result on the RIDI dataset.

In particular, Figure 4 shows the box plots of the evaluation results from the tested deep learning methods on the RoNIN dataset. The proposed EDIN model yields results with a more compact distribution and reduced median and mean values, demonstrating improved stability and superior navigation performance. Nevertheless, the method produces a larger number of outliers with greater deviation, which can be attributed to its limited effectiveness in handling particularly challenging cases.

Figure 5 shows the CDF results on the RoNIN-seen and RoNIN-unseen test dataset. On the RoNIN-seen dataset, the performance difference between EDIN and other RoNIN-series

models is relatively minor. However, on the RoNIN-unseen dataset, EDIN consistently outperforms the RoNIN-series models, indicating its superior robustness and adaptability to diverse environments. Nevertheless, as previously discussed, the figure also reveals that EDIN exhibits limited capability in handling particularly challenging cases. For instance, in the RoNIN-unseen dataset test, the maximum ATE error of EDIN is noticeably higher than that of the RoNIN-series models.

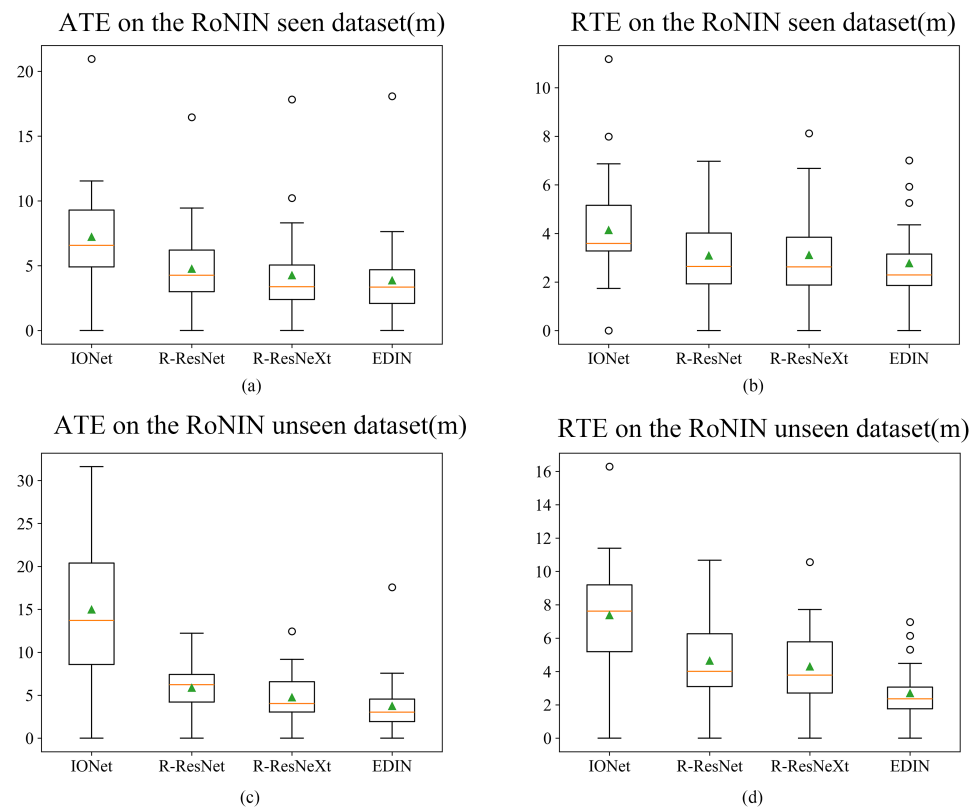


Figure 4. Box plots of the evaluation results from the tested deep learning methods on the RoNIN dataset. (a) ATE on the RoNIN seen dataset(m); (b) RTE on the RoNIN seen dataset (m); (c) ATE on the RoNIN unseen dataset (m); (d) RTE on the RoNIN unseen dataset (m). The box represents the interquartile range (IQR, Q1 to Q3), containing the middle 50% of the data, with the orange horizontal line inside denoting the median (Q2). The green triangle marker indicates the mean value. Whiskers extend to the minimum and maximum values within $1.5 \times \text{IQR}$. Open circles represent outliers, defined as values exceeding $Q3 + 1.5 \times \text{IQR}$ or below $Q1 - 1.5 \times \text{IQR}$. Different colors are used to distinguish different comparison methods for clear visualization.

3.5.2. Ablation Study

The improvements of our proposed EDIN method compared to the R-ResNeXt model can be summarized in three aspects: adding noise in the data augmentation part, adding CBAM to the ResNeXt module, and replacing MSE with Logcosh loss function (see Table 2). To verify the effectiveness of the above changes, we conducted additional ablation experiments. In this section, we train the R-ResNeXt model separately using one of the improved methods mentioned above. The test results of training the model using this method are shown in Table 3. To further validate the superiority of the Logcosh loss function over other mainstream robust loss functions and provide empirical evidence for its gradient explosion mitigation ability, we conduct a dedicated comparative experiment on the R-ResNeXt baseline with Huber, Tukey, MAE and Logcosh loss functions (see Table 2). The results show that the Logcosh loss achieves the optimal ATE and RTE on both RoNIN-seen and RoNIN-unseen datasets, with 4.32 m/2.93 m and 4.95 m/4.32 m respectively, which outperforms other loss functions significantly.

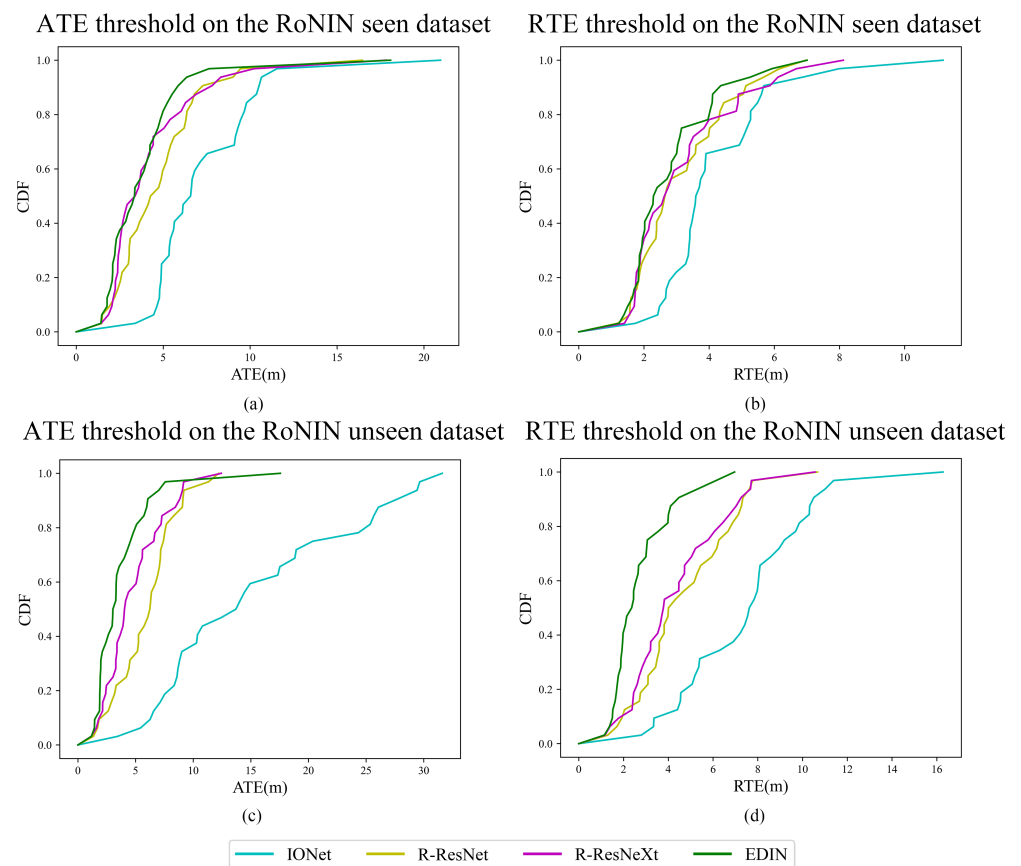


Figure 5. The ratio of RoNIN-seen and RoNIN-unseen test dataset under different thresholds for ATE and RTE metrics. (a) ATE threshold on the RoNIN seen dataset; (b) RTE threshold on the RoNIN seen dataset; (c) ATE threshold on the RoNIN unseen dataset; (d) RTE threshold on the RoNIN unseen dataset.

Table 2. Performance comparison of different loss functions based on R-ResNeXt (ATE: m; RTE: m).

Loss Function	Test Set	ATE	RTE
Huber	RoNIN-seen	4.31	3.11
	RoNIN-unseen	5.40	4.33
Tukey	RoNIN-seen	5.50	3.38
	RoNIN-unseen	5.51	4.48
MAE	RoNIN-seen	4.68	3.10
	RoNIN-unseen	6.11	4.60
Logcosh (Ours)	RoNIN-seen	4.32	2.93
	RoNIN-unseen	4.95	4.32

Table 3. Ablation study results (ATE: m; RTE: m).

Feature				RoNIN		OXIOD		RID1	
				Seen		Unseen			
Model Block	Loss Function	Data Augmentation		ATE	RTE	ATE	RTE	ATE	RTE
R-ResNeXt	ResNeXt	MSE	Rotation	4.89	3.44	6.04	4.78	1.75	1.99
	ResNeXt-CBAM	MSE	Rotation	4.10	2.99	5.23	4.28	1.48	1.88
	ResNeXt	Logcosh	Rotation	4.32	2.93	4.95	4.32	1.55	1.92
	ResNeXt	MSE	Rotation + Noise	4.25	2.90	5.48	4.31	1.52	1.94
EDIN	ResNeXt-CBAM	Logcosh	Rotation + Noise	3.85	2.78	4.74	4.12	1.34	1.87

It can be observed that, in comparison with the baseline R-ResNeXt model, all three proposed improvement strategies generally contribute to performance enhancement. Specifically, the integration of the CBAM effectively strengthens feature representation by selectively emphasizing informative channels and spatial locations, which in turn leads to higher localization accuracy. The adoption of the log-cosh loss function provides a more suitable optimization criterion for navigation-related tasks, as it mitigates the influence of outliers while maintaining sensitivity to small errors. Additionally, the incorporation of noise-based data augmentation increases the diversity of the training set, and to further explore the impact of noise intensity on model performance, additional ablation experiments under different noise intensities were conducted, with the detailed results presented in Table 4. Five levels of acceleration noise (0.05, 0.1, 0.2, 0.5, 1.0 m/s²) and corresponding gyroscope noise (0.0005, 0.001, 0.002, 0.005, 0.01 rad/s) were set, and the R-ResNeXt model was trained and tested under each intensity. The results indicate that appropriate noise intensity is conducive to enhancing the model's robustness against sensor noise. Thus, such a data augmentation strategy helps us to improve the model's generalization capability across different datasets, with particularly noticeable improvements in reducing translational errors.

Table 4. Ablation study on different noise intensities based on R-ResNeXt (Acc Noise: m/s²; Gyro Noise: rad/s; ATE: m; RTE: m).

Algorithm	Test Set	Acc Noise	Gyro Noise	ATE	RTE
R-ResNeXt	RoNIN-seen	0.05	0.0005	4.3974	3.1297
	RoNIN-unseen	0.05	0.0005	5.3888	4.5169
	RoNIN-seen	0.1	0.001	4.2500	2.9000
	RoNIN-unseen	0.1	0.001	5.4800	4.3100
	RoNIN-seen	0.2	0.002	4.6742	3.1466
	RoNIN-unseen	0.2	0.002	5.5829	4.6069
	RoNIN-seen	0.5	0.005	4.7626	3.1575
	RoNIN-unseen	0.5	0.005	5.2521	4.5088
	RoNIN-seen	1.0	0.01	4.6247	3.1275
	RoNIN-unseen	1.0	0.01	5.4548	4.5908

Despite these overall benefits, it is noteworthy that in certain scenarios, the use of these enhancements can result in decreased performance on specific evaluation metrics. Such degradation is likely attributable to variations in data characteristics among datasets, including differences in motion patterns, sensor noise profiles, and device placements, which may interact differently with the respective improvement strategies.

4. Discussion

In this study, we demonstrate that the proposed EDIN model achieves superior performance compared to other state-of-the-art approaches, particularly in terms of positioning accuracy and robustness. For most test sequences characterized by relatively small localization errors, EDIN is capable of producing trajectories that closely align with the ground truth, thereby reflecting its high precision in stable motion conditions. However, in a subset of anomalous sequences with substantially larger errors, the performance improvement becomes limited, and, in some cases, EDIN even underperforms relative to existing RoNIN-based methods. This degradation can primarily be attributed to the use of the log-cosh loss function during model training, which inherently suppresses the influence of outliers by assigning lower gradient weights to samples with large residuals. Consequently, the model tends to neglect anomalous sequences, leading to suboptimal generalization in

these scenarios. Addressing this limitation—specifically, how to effectively identify and adaptively handle anomalous sequences to further enhance system robustness—constitutes a key direction for our future research efforts.

In addition to the issues observed during testing, future work will focus on further improving the training methodology for deep inertial navigation. Specifically, we aim to investigate the integration of existing deep learning model optimization techniques to identify and refine optimal model architectures. This includes the application of automated hyperparameter optimization, neural architecture search, and other algorithmic strategies that have demonstrated effectiveness in enhancing model performance. Furthermore, we plan to explore alternative data augmentation techniques and loss functions tailored to inertial navigation tasks, with the goal of increasing model generalization and robustness across diverse motion scenarios. The effectiveness of these approaches will be rigorously evaluated through systematic training and testing on relevant inertial navigation datasets, thereby providing a foundation for more accurate and reliable deep inertial navigation systems.

5. Conclusions

This study introduces a new method for estimating user displacement through inertial sensors in daily navigation. We developed a deep inertial navigation model based on the ResNet model and CBAM module and overcame the previous training difficulties by improving the training method. To evaluate the effectiveness of this method, we conducted experiments using various representative user motion datasets. Our results indicate that this neural network model surpasses other existing neural network-based technologies. Future work will focus on expanding the research scope to enhance the robustness of the system, including its adaptability to different motion states and the ability to effectively manage IMU measurements characterized by high noise and bias.

Author Contributions: Conceptualization, J.W. and J.S.; Methodology, J.W. and G.C.; Software, J.W.; Validation, J.W. and G.C.; Formal analysis, J.W.; Investigation, J.W. and G.C.; Resources, J.S.; Data curation, J.W. and G.C.; Writing—original draft preparation, J.W.; Writing—review and editing, J.S. and G.C.; Visualization, J.W. and G.C.; Supervision, J.S.; Project administration, J.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in RoNIN, OXIOD and RIDI Public Repositories. These data were derived from the following resources available in the public domain: RoNIN Dataset (URL: <https://doi.org/10.20383/102.0543> (accessed on 2 October 2023)), OXIOD Dataset (URL: <http://deepio.cs.ox.ac.uk/> (accessed on 2 October 2023)), RIDI Dataset (URL: https://www.dropbox.com/scl/fi/39vw9jd491j68o7nbal7t/ridi_data_publish_v2.zip?rlkey=u8f1btq7vfc5vsovteb9bad7&e=1&dl=0 (accessed on 2 October 2023)).

Acknowledgments: During the preparation of this manuscript, the authors used ChatGPT 4.0 Free for the purposes of polishing the manuscript language. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

IMU	Inertial Measurement Unit
EDIN	Enhanced Deep Inertial Navigation
ResNeXt	Residual Next
CBAM	Convolutional Block Attention Module
ATE	Absolute Trajectory Error
RTE	Relative Trajectory Error
SINS	Strapdown Inertial Navigation System

References

- Swathi, N.; Dutt, V.I.; Rao, G.S. An adaptive filter approach for gps multipath error estimation and mitigation. In *Microelectronics, Electromagnetics and Telecommunications: Proceedings of ICMEET 2015*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 539–546.
- Zafari, F.; Gkelias, A.; Leung, K.K. A survey of indoor localization systems and technologies. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 2568–2599. [[CrossRef](#)]
- Zhuang, Y.; Yang, J.; Qi, L.; Li, Y.; Cao, Y.; El-Sheimy, N. A pervasive integration platform of low-cost mems sensors and wireless signals for indoor localization. *IEEE Internet Things J.* **2017**, *5*, 4616–4631. [[CrossRef](#)]
- Fan, B.; Yang, Y.; Feng, W.; Wu, F.; Lu, J.; Liu, H. Seeing through darkness: Visual localization at night via weakly supervised learning of domain invariant features. *IEEE Trans. Multimed.* **2022**, *25*, 1713–1726. [[CrossRef](#)]
- Diaz, E.M.; Ahmed, D.B.; Kaiser, S. A review of indoor localization methods based on inertial sensors. In *Geographical and Fingerprinting Data to Create Systems for Indoor Positioning and Indoor/Outdoor Navigation*; Academic Press: Cambridge, MA, USA, 2019; pp. 311–333.
- Harle, R. A survey of indoor inertial positioning systems for pedestrians. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 281–1293. [[CrossRef](#)]
- Foxlin, E. Pedestrian tracking with shoe-mounted inertial sensors. *IEEE Comput. Graph. Appl.* **2005**, *25*, 38–46. [[CrossRef](#)] [[PubMed](#)]
- Hou, X.; Bergmann, J. Pedestrian dead reckoning with wearable sensors: A systematic review. *IEEE Sens. J.* **2020**, *21*, 143–152. [[CrossRef](#)]
- Ahmetovic, D.; Gleason, C.; Ruan, C.; Kitani, K.; Takagi, H.; Asakawa, C. NavCog: A navigational cognitive assistant for the blind. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, Florence, Italy, 6–9 September 2016; pp. 90–99.
- Simon, D. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*; John Wiley & Sons: Hoboken, NJ, USA, 2006.
- Rajagopal, S. Personal Dead Reckoning System with Shoe Mounted Inertial Sensors. Master's Thesis, KTH Electrical Engineering, Stockholm, Sweden, 2008.
- Zampella, F.; Khider, M.; Robertson, P.; Jiménez, A. Unscented kalman filter and magnetic angular rate update (maru) for an improved pedestrian dead-reckoning. In *Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium*; IEEE: New York, NY, USA, 2012; pp. 129–139.
- Borenstein, J.; Ojeda, L.; Kwanmuang, S. Heuristic reduction of gyro drift in imu-based personnel tracking systems. In *Optics and Photonics in Global Homeland Security V and Biometric Technology for Human Identification VI*; SPIE: Bellingham, WA, USA, 2009; Volume 7306, pp. 244–254.
- Mahony, R.; Hamel, T.; Pflimlin, J.M. Nonlinear complementary filters on the special orthogonal group. *IEEE Trans. Autom. Control* **2008**, *53*, 1203–1218. [[CrossRef](#)]
- Madgwick, S.O.H.; Harrison, A.J.L.; Vaidyanathan, R. Estimation of imu and marg orientation using a gradient descent algorithm. In *2011 IEEE International Conference on Rehabilitation Robotics*; IEEE: New York, NY, USA, 2011; pp. 1–7.
- Chen, X.; Zhang, X.; Zhu, M.; Lv, C.; Xu, Y.; Guo, H. A novel calibration method for tri-axial magnetometers based on an expanded error model and a two-step total least square algorithm. *Mob. Netw. Appl.* **2022**, *27*, 794–805. [[CrossRef](#)]
- Mannini, A.; Sabatini, A.M. Walking speed estimation using foot-mounted inertial sensors: Comparing machine learning and strap-down integration methods. *Med. Eng. Phys.* **2014**, *36*, 1312–1321. [[CrossRef](#)] [[PubMed](#)]
- Xiao, X.; Zarar, S. Machine learning for placement-insensitive inertial motion capture. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*; IEEE: New York, NY, USA, 2018; pp. 6716–6721.
- Yan, H.; Shan, Q.; Furukawa, Y.; Ridi: Robust imu double integration. In *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, Germany, 8–14 September 2018; pp. 621–636.

20. Golroudbari, A.A.; Sabour, M.H. Recent advancements in deep learning applications and methods for autonomous navigation—A comprehensive review. *arXiv* **2023**, arXiv:2302.11089.
21. Chen, C.; Lu, X.; Markham, A.; Trigoni, N. Ionet: Learning to cure the curse of drift in inertial odometry. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; p. 32.
22. Greff, K.; Srivastava, R.K.; Koutník, J.; Steunebrink, B.R.; Schmidhuber, J. Lstm: A search space odyssey. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *28*, 2222–2232. [[CrossRef](#)] [[PubMed](#)]
23. Herath, S.; Yan, H.; Furukawa, Y. Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*; IEEE: New York, NY, USA, 2020; pp. 3146–3152.
24. Shen, S.; Gowda, M.; Choudhury, R.R. Closing the gaps in inertial motion tracking. In Proceedings of the 24th Annual International Conference on Mobile Computing and Networking, New Delhi, India, 29 October–2 November 2018; pp. 429–444.
25. Liu, W.; Caruso, D.; Ilg, E.; Dong, J.; Mourikis, A.I.; Daniilidis, K.; Kumar, V.; Engel, J. Tlio: Tight learned inertial odometry. *IEEE Robot. Autom. Lett.* **2020**, *5*, 5653–5660. [[CrossRef](#)]
26. Sun, S.; Melamed, D.; Kitani, K. Idol: Inertial deep orientation-estimation and localization. *Proc. Aaai Conf. Artif. Intell.* **2021**, *35*, 6128–6137. [[CrossRef](#)]
27. Wang, Q.; Luo, H.; Men, A.; Zhao, F.; Huang, Y. An infrastructure-free indoor localization algorithm for smartphones. *Sensors* **2018**, *18*, 3317. [[CrossRef](#)] [[PubMed](#)]
28. Fu, R.; Zhang, Z.; Li, L. Using lstm and gru neural network methods for traffic flow prediction. In *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*; IEEE: New York, NY, USA, 2016; pp. 324–328.
29. Cioffi, G.; Bauersfeld, L.; Kaufmann, E.; Scaramuzza, D. Learned inertial odometry for autonomous drone racing. *IEEE Robot. Autom. Lett.* **2023**, *8*, 2684–2691. [[CrossRef](#)]
30. Zhang, K.; Jiang, C.; Li, J.; Yang, S.; Ma, T.; Xu, C.; Gao, F. DIDO: Deep inertial quadrotor dynamical odometry. *IEEE Robot. Autom. Lett.* **2022**, *7*, 9083–9090. [[CrossRef](#)]
31. Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; He, K. Aggregated residual transformations for deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–27 July 2017; pp. 1492–1500.
32. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
33. Tian, Q.; Salicic, Z.; Kevin, I.; Wang, K.; Pan, Y. An enhanced pedestrian dead reckoning approach for pedestrian tracking using smartphones. In *2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*; IEEE: New York, NY, USA, 2015; pp. 1–6.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.