# Anomaly Filtering for Pedestrian Dead Reckoning Using Segment-Based Autoencoders

Mohamad Issam Sayyaf, Ni Zhu, Valerie Renaudin,
AME-GEOLOC, Univ. Gustave Eiffel, F-44344 Bouguenais, France
{ issam.sayyaf | ni.zhu | valerie.renaudin }@univ-eiffel.fr

***Abstract*—Accurate step detection is a cornerstone of Pedestrian Dead Reckoning (PDR), enabling applications such as indoor navigation, fitness monitoring and immersive gaming. However, step detection models often generate false detections during non-walking motions, such as hand gestures or sudden shaking. This paper introduces Robust SmartStep, a hybrid approach combining a reliable step detection algorithm SmartStep with a semi-supervised anomaly filtering framework based on a deep autoencoder. The method starts with the pre-processing of acceleration signals, including the generation of a time–frequency spectrogram and local normalization to enhance gait-specific features. An autoencoder trained on normal gait data identifies anomalous segments based on their higher reconstruction error, allowing the anomaly filter to effectively reject false steps. The experimental results of 14 km in different scenarios (pocket carrying, phone swinging, SMS, tests with visually impaired and sighted people) show that Robust SmartStep reduces over-detection errors by 20% compared to SmartStep and by 15% compared to the Android step counter and improves PDR accuracy. In addition, Robust SmartStep achieves PDR positioning performance that approaches the performance of highly personalized models tailored to the individual and one specific carrying mode of the device without the need for re-training. The method's lightweight design, coupled with open-source availability of datasets, code and an Android application facilitate integration into existing navigation systems. https://gitlab.univ-eiffel.fr/issam/robust-smartstep.**

***Index Terms*—Pedestrian Dead Reckoning, Autoencoder, Anomaly Detection, Step detection, Deep Learning, Inertial Sensors.**

## I. Introduction

From step counting and pedestrian navigation to fitness data tracking and health monitoring, step detection supports a wide range of applications. As part of the broader Human Activity Recognition (HAR) landscape, step detection plays an important role in interpreting and analyzing human movements. In healthcare, for example, accurate step detection with wearable devices (smartphones, smartwatches, etc.) can help monitor a patient's rehabilitation progress [1], [2]. In fall detection systems, step detection can assess mobility degradation [3] and help rapid response and prevention strategies for older adults [4]. In Virtual Reality (VR) and gaming, step detection enables immersive experiences by transforming real-world footfalls into dynamic in-game interactions [5]. When exploring indoor spaces where Global Navigation Satellite System (GNSS) signals are weak and distorted, step detection also assists positioning by estimating the length and direction changes of the steps [6].

Building on these applications, PDR is a particularly important application in scenarios where GNSS signals are weak, distorted, or unavailable [7]. To navigate in large office buildings and shopping malls, up to urban canyons, or to aid the mobility of the visually impaired, PDR uses wearable inertial sensor data for seamless indoor-outdoor navigation [8]. PDR estimates a person's position by tracking their steps, step length, and direction using sensors, often found in smartphones or wearable devices. Artificial Intelligence (AI) is increasingly used in PDR as it considers the different conditions of human gait and the position of wearables on the human body, improving both accuracy and robustness. There are two categories of AI-assisted pedestrian navigation with wearable inertial sensors: Sampling-Frequency-Driven AI Methods and Human-Gait-Driven AI Methods [9]. Accurate step detection is central in PDR positioning, as each detected step triggers the estimation of the pedestrian's trajectory [10], [11]. Performing several incorrect step detections will degrade PDR. Therefore, improving the robustness of the latter can significantly improve the accuracy of PDR and strengthen its central role in next-generation positioning and navigation services.

Over-detection of steps remains common when the smartphone experiences various anomalous movements that do not correspond to the actual steps of the human gait.Activities such as hand gestures, dropping the phone, or abrupt movements while sitting or standing can mimic walking signals (anomalous movements) and lead to false positives in step detection [12]. These false detections accumulate errors in distance and trajectory estimation, degrading navigation performance. Although existing step detection algorithms such as SmartStep have shown promise [13], they often struggle to distinguish between normal steps and these anomalous signals, resulting in lower reliability in real-world scenarios.

Building on previous work [14], this paper investigates how Anomaly Detection Function (ADF) can improve the performance of step detection. We perform a detailed performance evaluation to quantify the benefits of filtering out anomalous motions from the step detection model, and explain in detail the physical parameters that guide our ADF algorithm and their tuning process. By optimizing these parameters, we can effectively reduce false positives and improve the reliability of step detection in different walking scenarios. Furthermore, we show how ADF improves PDR in situations where GNSS signals are weak or unavailable. To evaluate this, we compare two PDR approaches: (1) a traditional PDR with a fixed step length and (2) an AI-driven PDR tailored to a specific wearable carrying mode (e.g., "pocket mode"), called Mobility Adapted Pedestrian Inertial Navigation (MAPIN), which uses a dedicated stride model to improve the positioning accuracy [15].

To advance research in this field, we make all relevant codes and datasets publicly available and also offer an application in the Google Store that allows researchers and practitioners to reproduce, evaluate and further refine our methods.

The remainder of this paper is organized as follows. In Section II a review of existing literature on anomaly detection is conducted. Section III details the proposed anomaly detection framework for robust and accurate step detection. Section IV describes the data sets, performance metrics and evaluation protocols used to assess our method. Section V discusses the open source codes and relevant datasets. Finally, Section VI presents our empirical findings, evaluating how anomaly detection improves step detection accuracy and positioning accuracy in PDR computation schemes. Section VII summarizes the key contributions, addresses the limitations of our study, and offers potential research directions for improving the scalability and reliability in broader navigation scenarios.

## II. Related Work

Anomaly detection in time series data has attracted substantial attention due to its significance in diverse applications such as industrial Internet of Things (IOT), radar systems, and complex multivariate environments. Many existing approaches leverage deep learning, notably Autoencoders (AE), variational autoencoders (VAE), and Generative Adversarial Networks (GAN), to learn robust representations of normal data and flag deviations as anomalies. Lin *et al.* [16] fuse VAE and Long Short-Term Memory (LSTM) architectures to capture both short- and long-term temporal dynamics, while Park *et al.* [17] and Nizam *et al.* [18] combine convolutional filters with recurrent encoders, demonstrating their effectiveness in network and IOT contexts. Other enhancements include frequency-domain analysis, as in Frequency-enhanced Conditional Variational Autoencoder (FCVAE) [19] model, and stochastic Recurrent neural network (RNN), as in OmniAnomaly [20], which can handle noise and complex dependencies in applications like spacecraft telemetry. Meanwhile, Zhang *et al.* [21] use Convolutional neural network (CNN) and ConvLSTMs to learn spatial-temporal correlations, Thill *et al.* [22] adopt temporal convolutions for long-range modeling, and transformer-based techniques [23], [24] employ attention mechanisms to capture extended dependencies and sensor relationships.

Despite their strong performance, most data-centric approaches rely predominantly on end-to-end learned features and do not explicitly incorporate physical insights into sensor signals. This omission can lead to redundant model complexity and potentially overlook domain-specific patterns such as gait regularities in pedestrian motion. Although some works address dimensionality reduction via feature selection [25], they still operate at a generic data level rather than exploiting fundamental physical phenomena. The gap, therefore, lies in bridging data-driven anomaly detection with domain-specific pre-processing that makes use of established physical principles. Nevertheless, these methods often overlook *model uncertainty*, a critical factor in safety-critical or rapidly changing environments, and *computational efficiency*, which becomes paramount when deploying models on resource-constrained devices.

In this work, we address this gap by integrating physically meaningful feature extraction—such as leveraging periodic and quasi-periodic motion characteristics— and power distribution along of spectrum into our pipeline, thus easing the learning task for the model and enhancing interpretability, robustness, and real-world applicability, and study the uncertunity of the model and take into our account the computational efficiency.

## III. Methodology and Implementation

The proposed solution builds on the SmartStep algorithm developed by the GEOLOC lab [13]. SmartStep uses two AI models to detect steps:

- ModelAcc (Acceleration Model): Identifies peaks in the norm of the acceleration signal.
- ModelGyro (Gyroscope Model): Detects valleys in the norm of the angular velocity signal, when the acceleration-based model becomes unreliable due to noise or missing periodicity.

Although SmartStep accurately detects most normal walking steps, it can misclassify abrupt or non-walking device motions (e.g., sudden phone shakes, accidental drops) as steps. To avoid these false positives, we introduce a semi-supervised ADF that verifies each step candidate to see how close it is to the normal walking pattern. Fig. 1 illustrates the overall architecture, where step candidates detected from SmartStep are validated (or rejected) by the ADF before being included in the final step count. This new version is called Robust SmartStep.

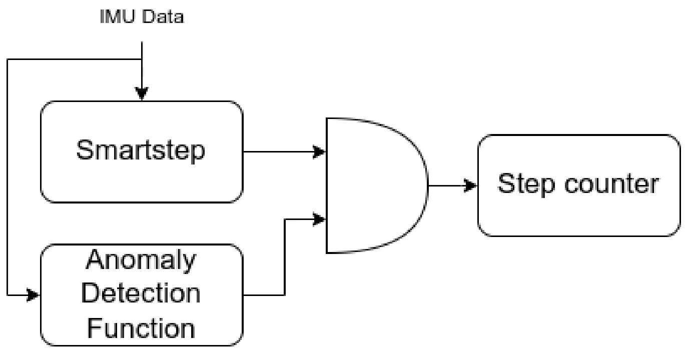


Fig. 1: The Robust SmartStep architecture.

ADF is based on a semi-supervised approach using an AE trained exclusively on normal walking patterns, which assigns higher reconstruction errors to accelerometer signals corresponding to anomaly movements. Since it's impossible to learn what abnormal movements are because their variety

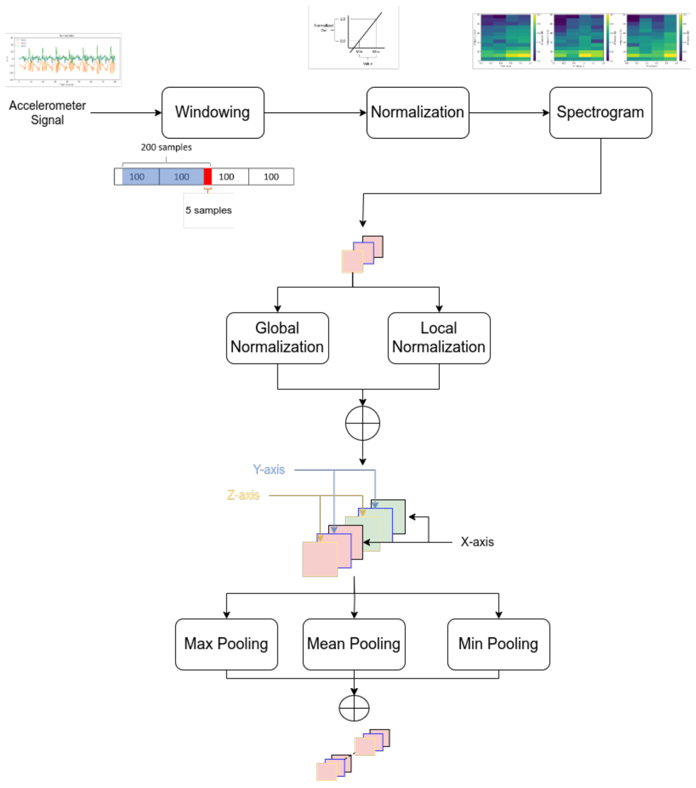


Fig. 2: The preprocessing pipeline.

is infinite, the idea is to build on normality and consider what complements it. Only step events confirmed by both the SmartStep core and the anomaly detector are ultimately counted. The following subsections outline the core design decisions, from preprocessing to model architecture and decision stage.

## A. *Preprocessing and Feature Extraction*

Figure 2 shows the main steps to convert raw accelerometer signals into features suitable for anomaly detection. The accelerometer data is analyzed along all three axes (X, Y and Z) instead of using its norm. Keeping the three axes has a major advantage because during normal walking, one axis typically shows greater periodic fluctuations while the other axes don't fluctuate as much, depending on the carrying modes. This provides an important context for identifying normal patterns, as opposed to abnormal patterns, which show greater randomness between axes and high variation in all axes, which is a very important feature for distinguishing between normal and abnormal patterns. By analyzing each axis individually, the algorithm can capture unique motion features more effectively than by averaging across all axes, which could obscure significant differences.

*1) Windowing:* Pre-processing begins with the segmentation of the accelerometer time series data into windows of fixed duration. Each window has a duration of 2 seconds and is shifted by 200 milliseconds with the following window. This window size was determined empirically to achieve a balance between stability, performance and complexity. It has been shown that longer windows (e.g. 3, 4 or 6 seconds) increase the computational cost and in some cases reduce the accuracy of the algorithm. This is because larger windows can dilute the effects of anomalous segments with normal signal components, resulting in lower reconstruction errors for anomalies.

*2) Normalization:* Following windowing, the data undergoes normalization to ensure consistency across all signals.

*3) Spectrogram Generation:* After normalization, the signals are transformed into the frequency domain using spectrograms. Spectrograms provide a visual representation of the distribution of energy across various frequency bands over time and capture both temporal and spectral features that are essential for detecting anomalies.

The parameters were chosen to generate a spectrogram with 24 frequency bins, each representing a 2 Hz interval, and 4 time slots, each corresponding to approximately 500 milliseconds. A 2 Hz resolution ($\Delta f = 2\,\text{Hz}$) was chosen because it effectively captures the fundamental frequency of walking (1–3 Hz) and its harmonics, resolves important spectral features associated with anomalies, and balances the trade-off between frequency detail and temporal resolution for gait events.

*4) Normalization of Spectrograms:* Spectrograms play a decisive role in the analysis of signals, especially in the detection of anomalies, as they represent the frequency content of a signal over time.

*a) Characteristics of Normal and Abnormal Signals in Spectrograms:*

- **Normal Signals**: These tend to have most of their power concentrated in the low frequency bands. The distribution is more uniform and consistent because regular movements (such as walking) are more uniform and repeated over time.
- **Abnormal Signals**: These typically show a more random distribution of power across the entire spectrogram range. This randomness is due to irregular or sudden movements that introduce higher frequency components and irregular energy patterns.

*b) Why Normalizing?:* Normalization enhances the ability to distinguish between normal and abnormal signals by adjusting the scale of the spectrogram data.

- **Global Normalization** scales the entire spectrogram uniformly, treating the whole image (or matrix) the same. This method identifies the maximum and minimum values across all possible scenarios or data conditions, using them to normalize the spectrogram. By doing so, the overall power distribution of the signal can be observed consistently, highlighting global trends and patterns, as shown in Fig. 3.
- **Local Normalization** normalizes each section of the spectrogram individually by calculating the maximum and minimum values specific to each local matrix. Local normalization highlights variations in power distribution within specific regions of the spectrogram as shown

in Fig. 4. This approach is key for revealing subtle anomalies, especially where abnormalities manifest.

*c) Importance of Local Normalization:* Local normalization is a critical phase because it emphasizes spatial and intensity relationships between pixels by scaling the pixel values according to their local context. This technique emphasizes contrasts, gradients and transitions and enables the detection of sharp changes (e.g. edges or transients) in the data. In spectrograms, it improves the detection of localized energy bursts or distributed patterns by normalizing pixel intensity relative to its surroundings. This is critical to identify anomalies that exhibit sharp contrasts and to distinguish them from smoother, normal patterns.

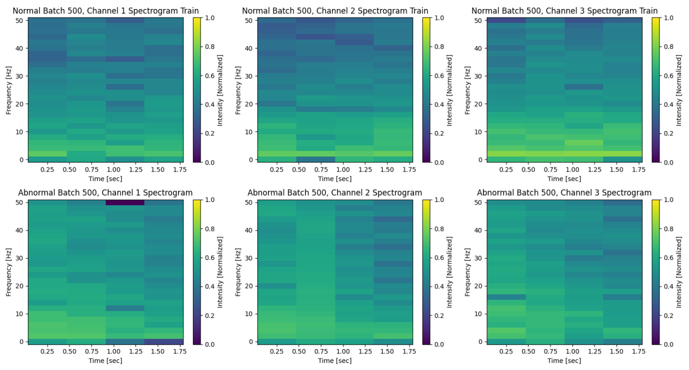


Fig. 3: Global normalization of the spectrogram. The upper plot represents a normal case, while the lower plot represents an anomaly case.

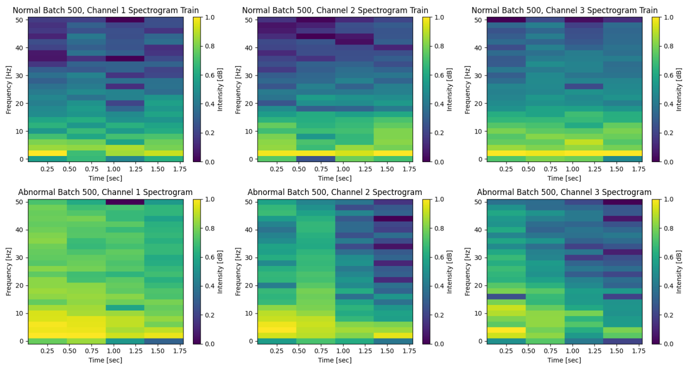


Fig. 4: Local normalization of the spectrogram. The upper plot represents a normal case, while the lower plot represents an anomaly case.

*5) Dimensionality Reduction through Pooling:* Pooling operations (max pooling, mean pooling, and min pooling) serve to summarize and extract essential features from normalized spectrograms while reducing their dimensionality.

## B. Model Architecture and Decision-Making

After preprocessing and feature extraction, the output is used to train the anomaly detection model. An AE architecture is chosen because it is effective in learning representations (latent space). This subsection presents the AE architecture and the decision process based on reconstruction errors.

*1) Autoencoder for Anomaly Detection:* An AE is an unsupervised network designed to learn the latent space of input data. It consists of two main components: an encoder, which compresses the input into a latent space representation, and a decoder, which reconstructs the input from this representation. For anomaly detection, the AE is trained on one type of data, in our case normal data, to reconstruct the input. The reconstruction error will be lower for the input data that follows the normal pattern than for the anomalous data that has a different pattern.

The model architecture consists of a flatten layer that converts the input spectrogram into a one-dimensional vector for processing. The encoder comprises two dense layers with 1024 and 512 units, each followed by dropout layers with a 50 rate to prevent overfitting. The decoder mirrors the encoder structure, featuring dense layers that expand the data back to its original size and dropout layers with a 30 rate for regularization. The final dense layer uses sigmoid activation to reconstruct the original data, which is then reshaped to match the spectrogram format.

The AE is trained using the Adam optimizer with a learning rate that allows efficient convergence. The loss function used is the mean squared error (MSE), which quantifies the difference between the input spectrogram and its reconstruction.

*2) Anomaly Detection via Reconstruction Error:* Once trained, the AE serves as the core component for anomaly detection. The fundamental principle is that the AE should accurately reconstruct inputs similar to the training data (i.e., normal signals) but fail to do so for anomalous inputs, resulting in higher reconstruction errors.

The reconstruction error is calculated as the MSE between the input spectrogram and its reconstruction by

$$\text{Reconstruction Error} = \frac{1}{N}\sum_{i=1}^{N}(S_{\text{input},i} - S_{\text{recon},i})^2 \quad (1)$$

where $S_{\text{input}}$ is the input spectrogram, $S_{\text{recon}}$ is the reconstructed spectrogram, and $N$ is the number of elements in the spectrogram.

*3) Decision-Making:* To improve anomaly detection and reduce short-term noise, the reconstruction errors are averaged over one-second intervals. This design choice is motivated by three main factors. 1) Transitions between normal and anomalous behaviors typically require at least one second. Averaging over this interval thus captures the dominant signal characteristics while preventing rapid oscillations between normal and anomalous states. 2) Short-lived segments of data—whether anomalous segments within a normal interval or normal-like segments within an anomalous interval—can mislead the reconstruction error if treated individually. By integrating over one second, these transient anomalies or normal patches are smoothed out, improving detection robustness. 3) A one-second window offers a practical trade-off between real-time responsiveness and the risk of over-smoothing. During each second, the model runs five inferences (one every 200 ms) and averages their reconstruction errors for final classification. This ensures timely yet accurate detection without imposing excessive computational overhead.

Figure 5 illustrates the overall decision process, showing how the one-second averaging window averages these factors to provide reliable anomaly detection.

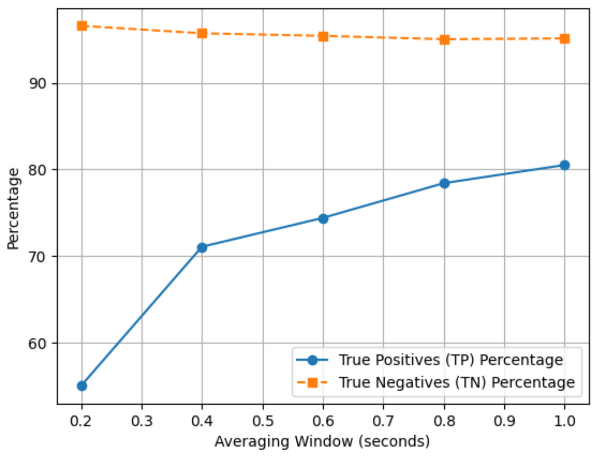


Fig. 5: Decision-Making Process Based on Averaged Reconstruction Error

## IV. Experimental Setup

Several experiments were conducted to evaluate Robust SmartStep at the level of step counting and for PDR based navigation applications.

### A. Step Detection Experiment

We conducted experiments with seven participants in total: five were sighted and two had vision impairment. To simulate real-world conditions, we designed three scenarios specifically for the sighted participants, while the participants with vision impairment simply walked normally. This approach allowed us to test the system's performance across different user profiles and conditions.

*a) Scenario 1:* 1) Walk about 50 meters with the phone in a pocket. 2) Receive a call and take the phone out of the pocket. 3) Walk 25 meters while talking the call. 4) End the call and put the phone back in the pocket. 5) Walk another 25 meters. 6) Stop for 3 minutes at a traffic light while shaking legs nervously. 7) Continue walking for another around 100 meters.

*b) Scenario 2:* 1) Walk about 50 meters while swinging the phone. 2) Stop to tie shoes. 3) Take the phone out of the pocket to check the time. 4) Resume walking while swinging the phone for around 100 meters. 5) Arrive at a coffee shop, sit down, and place the phone on the table. 6) Take the phone to show photos to a friend while sitting. 7) Continue walking a few meters.

*c) Scenario 3:* 1) Walk about 50 meters while texting on the phone. 2) Stop at a bus station. 3) Browse the internet. 4) Talk with a friend while holding the phone. 5) When the bus is delayed, continue walking while texting on WhatsApp for around 50 meters.

### B. PDR Experiment

The aim of the PDR experiment is to test the effect of false step detection due to unusual sensor movements in typical scenarios of smartphone navigation applications. In the evaluation, we study the effect of Robust SmartStep on two PDR algorithms, 1) Traditional PDR using a constant step length estimated from a fixed proportion of the participant's height $L = k.h$, where L is the step length, h is the participant's height, and k is an empirically determined scaling factor) [26]. Heading information is derived from a GNSS-based head-mounted solution. This uses satellite signals to determine direction. This approach is simple and has a low computational cost. In environments with blocked or poor GNSS signals, it is difficult to maintain high navigation accuracy. 2) MAPIN, a customized inertial navigation framework. MAPIN segments accelerations into step units and applies an LSTM-based model to accurately estimate each user's stride length and heading. This approach uses a tailored model and specific movement patterns [15]. MAPIN we need to train the model per person and in a specific carrying mode.

### C. Equipment and Ground Truth

Figure 6 shows the experimental setup.

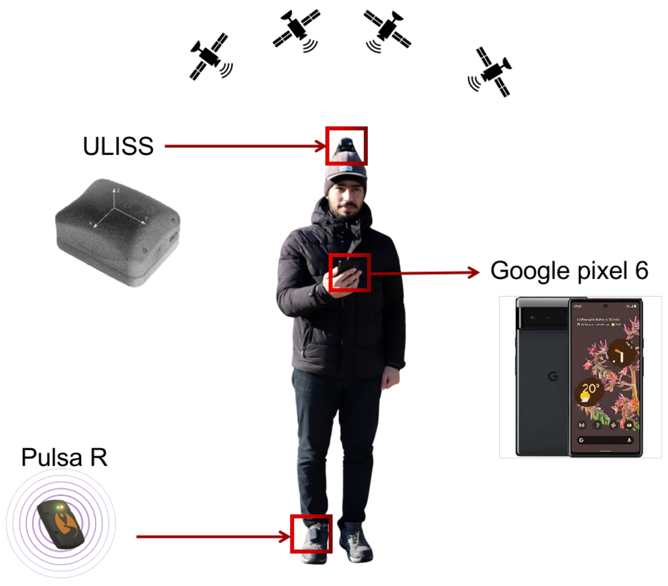


Fig. 6: Experimental hardware setup: ULISS, Smartphone and PulsaR.

*1) Hardware setup:* Tri-axis accelerometer and gyroscope were recorded with a Google Pixel 6 smartphone. PulsaR [27] was used as a reference step counter and attached to the foot. It is a commercial AI-based Zero Velocity Update (ZUPT) positioning solution. The Ubiquitous Localization with Inertial Sensors and Satellite (ULISS) device [28] was mounted on the user's head to collect high-precision GNSS data. ULISS is a state-of-the-art inertial navigation system with an Xsens Mit-7 IMU-Mag sensor, an Ublox ZED-F9P GNSS receiver and a Bosch BMP280 pressure and temperature module that provides triaxial acceleration, gyroscope and magnetometer measurements at 200 Hz, GNSS data at 5 Hz and pressure and temperature measurements at 50 Hz. It also has a Bluetooth communication module for remote control via a dedicated smartphone application that can be used to start and stop data recording. Thanks to the built-in GNSS receiver, all readings are time-stamped with the GPS time of week, facilitating synchronization between multiple devices. Another person accompanied the participant during the experiments and filmed his legs.

*2) Ground Truth:*

*a) The step counter ground truth:* was established using the PulsaR, ULISS and Video Recording. PulsaR provides a reliable reference for step counting, that is based on the detection of Zero Velocity Instants of the foot assisted by AI. But if the user vibrates or moves its foot without actually walking (shaking leg while waiting or searching for an obstacle with the foot as a visually impaired person), PulsaR may register false steps. For this reason, all experiments were also recorded on video to verify step counts and correct any false detection. This supplemental visual data allowed for precise manual verification of each participant's step count and motion patterns, ensuring that the PulsaR system's estimates were accurate.

*b) The pedestrian's positioning ground truth:* was computed with the head mounted ULISS. The GNSS module gives a $1.5\,\mathrm{m}$ CEP Horizontal position accuracy, a $0.05\,\mathrm{m/s}$ Velocity accuracy and a $0.3^\circ$ Dynamic heading accuracy [29]. It is used as a reference to evaluate the PDR positioning accuracy in next sections.

## V. Open-Source Resources

This paper provides all resources as open source in the repository. The repository is structured into several main folders. https://gitlab.univ-eiffel.fr/issam/robust-smartstep

### *A. Anomaly Detection Codes*

This folder contains the code developed in C++ for deployment on embedded devices, as C++ is well-suited for such applications. It also includes Python code used for training the anomaly detection model. The workflow involves using the outputs generated by the C++ implementation to train the model in Python, and then transfer the trained model to Open Neural Network Exchange (ONNX) format for compatibility with embedded devices. In practical terms, ONNX [30] supports techniques like quantization, which reduces the precision of weights and activations (e.g., 32-bit to 8-bit), thereby compressing the model size and reducing memory usage. This also accelerates inference by enabling efficient use of hardware accelerators optimized for lower precision, making it highly suitable for resource-constrained embedded systems.

### *B. Robust SmartStep Codes*

This folder includes the Robust SmartStep system, which integrates SmartStep with the ADF. This system features a Python-based script called run designed for user-friendly operation (not the developer version). Upon execution, the script prompts the user to add the desired data to the input folder. It then processes the data and saves the results in the output folder. The output folder also contains another script for detailed signal analysis. Users can run Robust SmartStep and achieve results identical to those obtained by walking, ensuring practical and reliable performance.

### *C. Training Dataset for Anomaly Detection in Step Counter*

The folder includes a dataset of Accelerometer signals, representing 30 km of walking data collected in texting, swinging, and pocket carrying modes for training, and 10 km for testing across all handling modes. The dataset consists of data from eight participants. For training, data from one participant was used, while data from the other seven participants was reserved for testing.

### *D. Evaluation Dataset*

It is the dataset for the evaluation of the performance of the robust SmartStep system according to the previously described scenarios  IV-A. The dataset contains time series data with fields including time, accelerometer (Acc), gyroscope (Gyro) and magnetometer (Mag). It was collected from six participants. Four were sighted and two were visually impaired. This diverse data set was specifically designed to evaluate the robustness, accuracy, and inclusivity of the system under a variety of real-world conditions.

### *E. Android App: Geoloc Smart Step*

In addition, we developed an Android application for Robust SmartStep, available for download on the Google Play Store called "GEOLOC Smart Step" [31]. This app allows users to record data from the accelerometer, gyroscope, and magnetometer, along with the number of steps taken. The recorded data can then be saved and shared for further analysis.

## VI. Performance Evaluation and Discussion

This section evaluates the performance of Robust SmartStep both at the step counting and the pedestrian positioning system levels.

*1) Performance of Robust Smartstep:* As shown in Fig. 8, Robust SmartStep effectively reduces over-detection errors by filtering out abnormal movements, like phone vibrations or static handling. It achieves as step counting average error of $7.25\% \pm 6.36\%$, compared to $28.20\% \pm 11.43\%$ for SmartStep. The robust step count significantly improves the estimates with a lower variance. The commercial step counter provided by the Android Operating System [32] gives an average error of $23.14\% \pm 11.17\%$. Again Robust SmartStep better performances both in accuracy and variance.

The residual errors observed in the Robust SmartStep system can be attributed, in part, to the inherent limitations in the design of the ADF [14]. A part of the normal distribution overlaps with the anomaly distribution as shown in Fig. 7, making it harder to distinguish normal signals from abnormal ones in this overlap. This leads to higher False Positive Rate (FPR), which is the proportion of normal signals incorrectly flagged as anomalies, and False Negative Rate (FNR), which is the proportion of anomalies incorrectly flagged as normal. Consequently, the True Negative Rate (TNR), which is the proportion of normal signals correctly identified, and the True Positive Rate (TPR), which is the proportion of anomalies correctly identified, both decrease.

In particular, the "pocket mode" produces normal signals that resemble abnormal signals. By excluding the pocket mode from the training of the ADF, it is possible to significantly improve the performance of the pedometer. This exclusion is useful for various wearables, such as smartwatches, which usually have a more predictable usage mode. Table I shows the performance of ADF with and without pocket mode. Overall, an improvement in ADF leads to an improvement in the performance of Robust Smarstep.

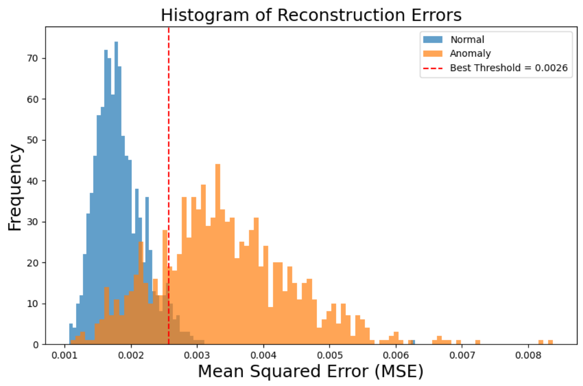


Fig. 7: Histogram of RMSE values with overlap between normal and anomaly signals.

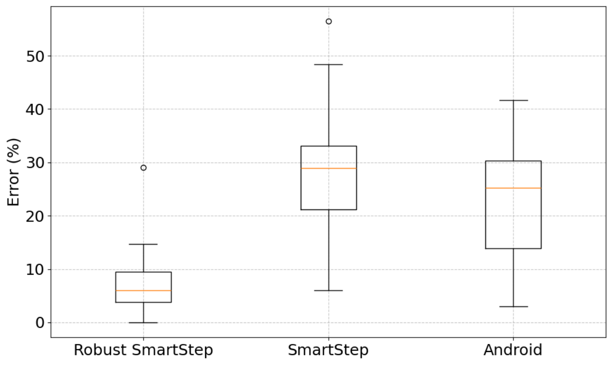


Fig. 8: Comparison of the error distributions for SmartStep, Robust SmartStep and Android Step Counter.

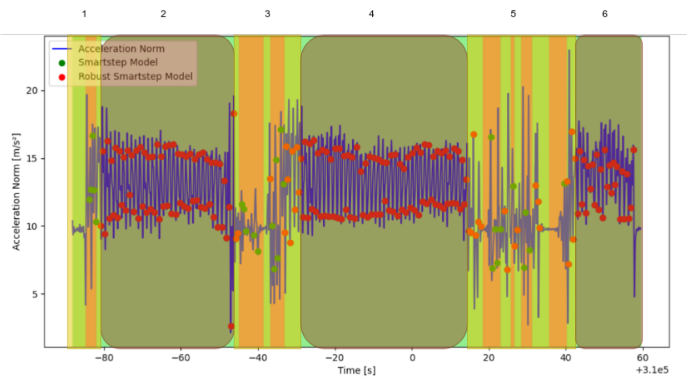


Fig. 9: Accelerometer signal in Scenario 2, showing how Robust SmartStep filters out anomalous intervals (orange zones) while preserving normal activity (green zones).

Figure 9 shows the norm of the raw acceleration signals (blue line) over time in scenario 2. The model classifies regions as anomalous (orange zones) or normal (green zones). All detected steps are discarded in the anomalous zones and retained in the normal (green) zones.

Rectangular markers labeled (2, 4, 6) indicate the true walking steps recorded in "swinging mode." In contrast, the markers labeled (1, 3, 5) highlight periods of anomalous activity, which include; 1): The initial phase of pressing the record button and laying the phone down in swinging mode, 2): Setting the phone on the ground to tie shoe laces, then picking it up again, 3): Stopping abruptly, placing the phone on a table, then briefly showing photos to a friend and handing it over for further viewing. This figure demonstrates how Robust SmartStep effectively filters out these anomalous intervals (orange zones), preventing them from being misidentified as steps.

*2) Statistical Analysis:* To determine if Robust Smartstep leads to a significant reduction in error rates compared to the original SmartStep, we conducted an independent two-sample t-test. We make 2 hypotheses.

- **Null Hypothesis** ($H_0$): There is no difference in error rates between Robust SmartStep and SmartStep.
- **Alternative Hypothesis** ($H_1$): Robust SmartStep has lower error rates than SmartStep.

The results of our statistical tests are summarized in Table II. As shown, The negative t-statistic indicates that Robust SmartStep generally has lower error rates compared to SmartStep, and Android. The extremely low p-value ($p < 0.001$) suggests that the difference in performance is statistically significant. Therefore, we reject the null hypothesis and conclude that Robust SmartStep significantly outperforms SmartStep in reducing error rates. The statistical analysis confirms that Robust SmartStep provides a significant reduction in error rates, resulting in an overall error decrease from approximately 28% to under 7%. By filtering out non-walking events (e.g., vibrating legs or interactions with the phone), Robust SmartStep demonstrates both lower variance and higher accuracy across diverse

TABLE I: Performance of the ADF with/without pocket mode.

| **Case** | **TNR** (% ± std) | **TPR** (% ± std) | **FPR** (% ± std) | **FNR** (% ± std) |
|---|---|---|---|---|
| ADF with Pocket | 82.50 ± 1.25% | 95.20 ± 0.70% | 17.50 ± 1.25% | 4.80 ± 0.70% |
| ADF without Pocket | 96.72 ± 0.95% | 96.05 ± 0.62% | 3.28 ± 0.95% | 3.95 ± 0.62% |

conditions. These findings underscore the practical advantages of using Robust SmartStep for reliable step detection in real-world scenario.

TABLE II: t-test results comparing Robust SmartStep error rates against SmartStep and Android

| **Comparison** | **t-Statistic** | **p-Value** |
|---|---|---|
| Robust vs. SmartStep | -7.1 | $7.2 \times 10^{-9}$ |
| Robust vs. Android | -5.5 | $2.5 \times 10^{-6}$ |

## A. *Pedestrian Dead Reckoning*

The PDR experiment assesses the system's ability to accurately estimate a pedestrian's position during abnormal movements. We compared the performance of the traditional PDR algorithm with the AI-based MAPIN. This study includes 15 trajectories for two sighted and two visually impaired people, covering a total distance of 6.7 $km$.

*1) Performance evaluation metrics:* The following metrics are used to assess the positioning performances.

**Root Mean Square Error (RMSE):**

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(e_i)^2}, \tag{2}$$

where $e_i$ is the error at the $i$-th point, and $n$ is the total number of points.

**Drift:** It is given in (%) by

$$\text{Drift} = \frac{L_{\text{est}} - L_{\text{gt}}}{L_{\text{gt}}} \times 100, \tag{3}$$

where $L_{\text{est}}$ and $L_{\text{gt}}$ are the estimated and ground truth total traveled length walked by the pedestrian.

*2) Positioning with classic PDR Algorithm:* Positioning is assessed with two kinds of step detection algorithms: SmartStep, we call it Normal PDR (N-PDR) and Robust SmartStep, called Robust PDR (R-PDR). The details of the classic PDR algorithm are described in Section IV-B.

TABLE III: Comparison of R-PDR vs N-PDR for 15 trajectories.

| **Metric** | **R-PDR** | **N-PDR** |
|---|---|---|
| RMSE | $20.68 \pm 14.01$ | $49.64 \pm 25.86$ |
| Drift | $9.61 \pm 7.08$ | $35.52 \pm 10.70$ |

Table III gives the statistical performances for R-PDR and N-PDR averaged over 15 trajectories, covering a total distance of 6.7 $km$. The R-PDR consistently outperforms the N-PDR on every metric. RMSE drops from approximately 49.64 $m$ N-PDR to 20.68 $m$ R-PDR. Additionally, the drift for R-PDR was reduced to 9.61%, compared to 35.52% for N-PDR, demonstrating improved alignment with the true path length.

The t-test shows statistically significant differences (p-values well below 0.05). Moreover, the estimated length is substantially closer to the true path for R-PDR.

To analyze more, we present one of the trajectory for a visually impaired participant, covering 553.98 $m$ with N-PDR and R-PDR. Fig. 10 shows the trajectory of R-PDR, that is closer to the ground truth, while N-PDR has large deviations, which shows the effectiveness of the Robust SmartStep in reducing position errors. The RMSE for R-PDR is 18.95 m, whereas the RMSE for N-PDR is 58.71 m. This significant reduction in RMSE demonstrates the superior performance of the R-PDR system in accurately estimating the trajectory. The cumulative distribution function (CDF) of the positioning error, that related with the same experiment is shown in Fig. 11 for the R-PDR and N-PDR. In the case of R-PDR, 90% of the errors are below 27.5 $m$ corresponding to 4.49%. The steep curve and tight error distribution indicate better robustness and reliability.

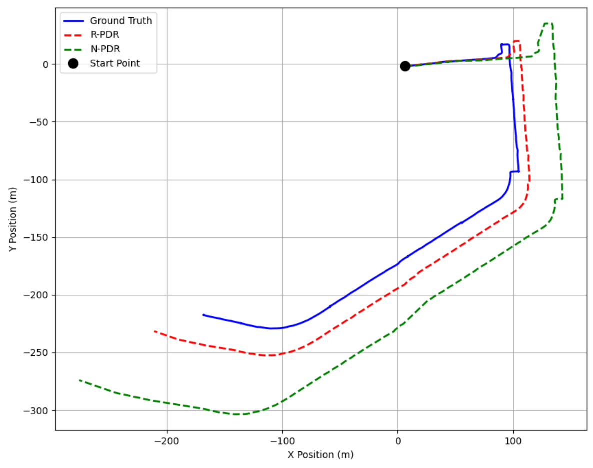


Fig. 10: Comparison of trajectories for N-PDR and R-PDR against the ground truth

Whereas for the N-PDR, 90% of errors are below 96 $m$ corresponding to 17.31%. The flatter curve reflects higher variability and larger errors. The drift for R-PDR is lower at 11.11%, compared to 42.68% for N-PDR.

Overall, the R-PDR approach significantly improves trajectory estimation compared to the N-PDR approach. R-PDR achieves lower RMSE, and effectively prevents overestimation of path length. Therefore, R-PDR provides a more reliable, and accurate solution for pedestrian navigation.

*3) AI-Based PDR with MAPIN:* The purpose of this comparison is to evaluate the performance of R-PDR against one

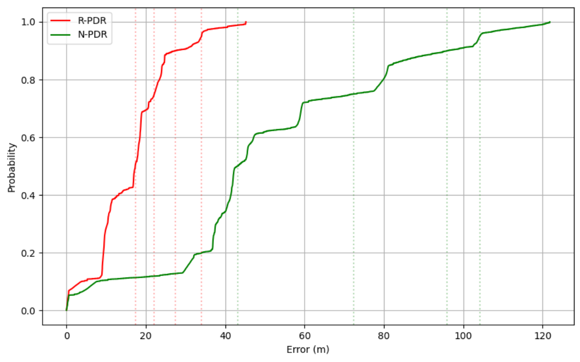


Fig. 11: Cumulative Distribution Function (CDF) of positioning errors for R-PDR and N-PDR.

of the cutting-edge PDR algorithms with AI, MAPIN. This human gait-driven PDR builds a model tailored to the individual fingerprint of the human gait, including the location of the wearable (e.g. carrying a phone in a pocket) [15]. Through this customization, MAPIN can effectively filter out unexpected movements (anomalous movements), ensuring high accuracy in certain scenarios. For this study, we retrained the MAPIN model for each participant using at least 3 km of walking data in pocket mode.

The box plot in Fig. 12 compares the performance of the two algorithms using data from 15 trajectories. MAPIN demonstrated an RMSE of $11.78 \pm 6.44,\text{m}$ and a drift of $6.09 \pm 8.68$ %. By contrast, R-PDR, as shown in Table III, exhibited slightly higher errors. However, this difference is not substantial, with R-PDR showing an RMSE approximately 2% higher than MAPIN.

While MAPIN achieves better results, this is largely due to its reliance on a customized model tailored to a specific person and phone-carrying mode. This specialization is a strength when the application involves specific carrying mode and demands high accuracy. However, it also represents a limitation, as the algorithm requires retraining for each individual for specific carrying modes.

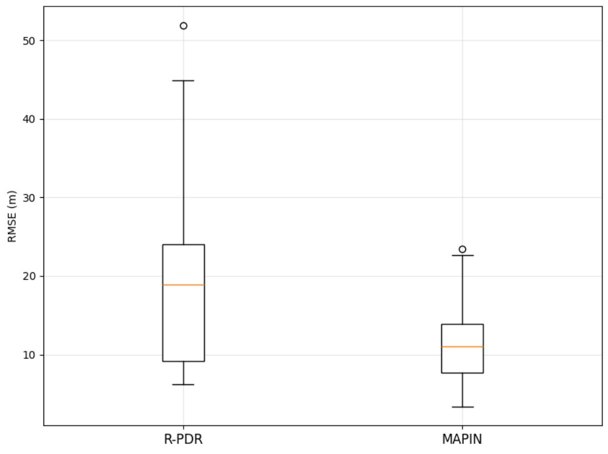
On the other hand, R-PDR offers a simpler, lightweight, and generalizable solution. Unlike MAPIN, R-PDR is designed to work across different phone-carrying modes, including those not explicitly trained. Its accuracy and general applicability could make it suitable for scenarios where a universal solution is needed, even if it tolerates slightly higher errors compared to MAPIN.

Overall, there is a trade-off between user-specific precision and broader adaptability. MAPIN is well-suited for consistent, single-user use under a specfic carrying mode, while more general approaches the R-PDR are preferable in multi-user or rapidly changing environments, where adaptability is key.

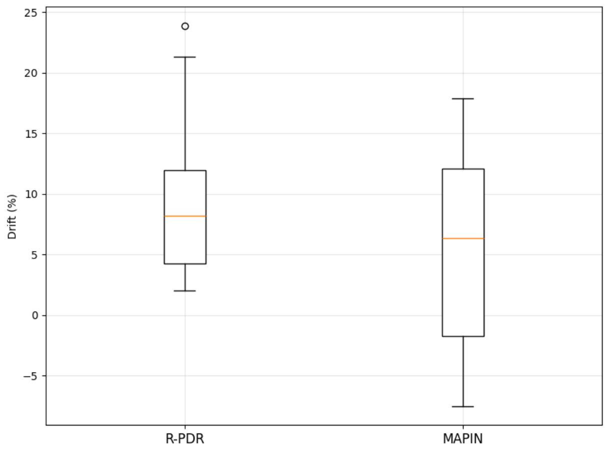
## VII. Conclusion and Future Work

This study introduces an anomaly-filtering framework designed to enhance step detection accuracy by minimizing false-positive step detections caused by phone movements not related to walking. Central to this framework is a semi-supervised AE model, trained exclusively on data representing normal walking patterns. The AE identifies anomalies by detecting elevated reconstruction errors, which indicate deviations from normal walking pattern. When integrated with the existing SmartStep algorithm, Robust SmartStep effectively filters out spurious step detections in real-time, thereby improving the reliability of step counting.

We evaluated Robust SmartStep in a range of real-world scenarios, including phone-carrying modes such as pocket, swinging, and texting, as well as with visually impaired participants. It reduced false step detections by 20%, while retaining nearly all real steps. In addition Robust SmartStep shows higher performance comparing with Android step counter algorithm by more than 15%. When employed within a PDR framework, we obtained R-PDR, which has also notably increased navigation accuracy compared to N-PDR. R-PDR achieves accuracy close to that of specialized user-



(a) RMSE



(b) Drift

Fig. 12: Comparison of MAPIN and R-PDR metrics: (a) RMSE (m) and (b) Drift (%).

specific AI algorithms without requiring retraining, and it works effectively across different carrying modes and users. Interestingly, training the ADF exclusively with data from texting and swinging modes further enhanced its performance to filter anomaly motions, suggesting that Robust SmartStep may be particularly well-suited for other applications with specific carry mode like smartwatch.

Beyond its strong performance across diverse conditions, Robust SmartStep is notable for its lightweight design and easy integration into existing navigation systems. All relevant datasets, source code, and an Android application have been made publicly available to encourage further research, reproducibility, and real-world applications. Future work will focus on expanding applicability to broader domains, such as healthcare and wearable technologies.

## References

[1] T. Fokkema, T. J. M. Kooiman, W. P. Krijnen, C. P. van der Schans, and M. de Groot, "Reliability and validity of ten consumer activity trackers depend on walking speed," *Medicine and Science in Sports and Exercise*, vol. 49, no. 4, pp. 793–800, 2017.

[2] C. Tudor-Locke, C. L. Craig, Y. Aoyagi *et al.*, "How many steps/day are enough? for older adults and special populations," *Intl. J. of Behavioral Nutrition and Physical Activity*, vol. 8, p. 80, 2011.

[3] N. A. Abiad, E. Houdry, C. El Khoury, V. Renaudin, and T. Robert, "A method for calculating fall risk parameters from discrete stride time series regardless of sensor placement," *Gait & Posture*, vol. 111, pp. 182–184, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0966636224001310

[4] F. Bagala, C. Becker, A. Cappello, L. Chiari, K. Aminian, and J. M. Hausdorff, "Evaluation of accelerometer-based fall detection algorithms on real-world falls," *PLoS ONE*, vol. 7, no. 5, p. e37062, 2012.

[5] P. Caserman, P. Krabbe, J. Wojtusch, and O. von Stryk, "Real-time step detection using the integrated sensors of a head-mounted display," in *IEEE Intl. Conf. on Systems, Man, and Cybernetics (SMC)*, 2016, pp. 003 510–003 515.

[6] A. Poulose, O. S. Eyobu, and D. S. Han, "An indoor position-estimation algorithm using smartphone imu sensor data," *IEEE Access*, vol. 7, pp. 11 165–11 177, 2019.

[7] R. Harle, "A survey of indoor inertial positioning systems for pedestrians," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1281–1293, 2013.

[8] S. Bai, W. Wen, Y. Li, C. Shi, and L.-T. Hsu, "Toward persistent spatial awareness: A review of pedestrian dead reckoning-centric indoor positioning with smartphones," *IEEE Transactions on Instrumentation and Measurement*, vol. 73, pp. 1–28, 2024.

[9] H. Fu, V. Renaudin, Y. Kone, and N. Zhu, "Analysis of the recent ai for pedestrian navigation with wearable inertial sensors," *IEEE Journal of Indoor and Seamless Positioning and Navigation*, vol. 1, pp. 26–38, 2023.

[10] S. Tiwari and V. K. Jain, "A novel step detection technique for pedestrian dead reckoning based navigation," *ICT Express*, vol. 9, no. 1, pp. 16–21, 2023.

[11] S. Beauregard and H. Haas, "Pedestrian dead reckoning: A basis for personal positioning," *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication (WPNC'06)*, 01 2006.

[12] Z. Chen, Q. Zhu, and Y. C. Soh, "Smartphone inertial sensor-based indoor localization and tracking with ibeacon corrections," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 4, pp. 1540–1549, 2016.

[13] N. A. Abiad, Y. Kone, V. Renaudin, and T. Robert, "Smartstep: A robust step detection method based on smartphone inertial signals driven by gait learning," *IEEE Sensors Journal*, vol. 22, no. 12, pp. 12 288–12 297, 2022.

[14] M. I. Sayyaf, N. Zhu, V. Renaudin, and T. Feigl, "Step Detection Enhanced by Anomaly Filtering," in *Proc. Intl. IEEE Applied Sensing Conference (APSCON)*, Jan 2025, pp. 1–4.

[15] H. Fu, V. Renaudin, T. Bonis, and N. Zhu, "Mapin: Mobility adapted pedestrian inertial navigation using smartphones for enhanced travel of the visually impaired," in *2024 14th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2024, pp. 1–6.

[16] S. Lin, R. Clark, R. Birke, S. Schönborn, N. Trigoni, and S. Roberts, "Anomaly detection for time series using vae-lstm hybrid model," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 4322–4326.

[17] W. Park, N. Ferland, and W. Sun, "Autoencoder for network anomaly detection," in *2022 IEEE International Symposium on Measurements & Networking (M&N)*, 2022, pp. 1–6.

[18] H. Nizam, S. Zafar, Z. Lv, F. Wang, and X. Hu, "Real-time deep anomaly detection framework for multivariate time-series data in industrial iot," *IEEE Sensors Journal*, vol. 22, no. 23, pp. 22 836–22 849, 2022.

[19] Z. Wang, C. Pei, M. Ma, X. Wang, Z. Li, D. Pei, S. Rajmohan, D. Zhang, Q. Lin, H. Zhang, J. Li, and G. Xie, "Revisiting vae for unsupervised time series anomaly detection: A frequency perspective," in *Proceedings of the ACM Web Conference 2024*, ser. WWW '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 3096–3105. [Online]. Available: https://doi.org/10.1145/3589334.3645710

[20] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2828–2837.

[21] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla, "A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 1409–1416.

[22] M. Thill, W. Konen, H. Wang, and T. Bäck, "Temporal convolutional autoencoder for unsupervised anomaly detection in time series," *Applied Soft Computing*, vol. 112, p. 107751, 2021.

[23] Z. Chen, D. Chen, X. Zhang, Z. Yuan, and X. Cheng, "Learning graph structures with transformer for multivariate time-series anomaly detection in iot," *IEEE Internet of Things Journal*, vol. 9, no. 12, pp. 9179–9189, 2021.

[24] S. Ma, S. Guan, Z. He, J. Nie, and M. Gao, "Tpad: temporal pattern based neural network model for anomaly detection in multivariate time series," *IEEE Sensors Journal*, 2023.

[25] H. Y. Teh, K. I.-K. Wang, and A. W. Kempa-Liehr, "Expect the unexpected: Unsupervised feature selection for automated sensor anomaly detection," *IEEE Sensors Journal*, vol. 21, no. 16, pp. 18 033–18 046, 2021.

[26] L. E. Díez, A. Bahillo, J. Otegui, and T. Otim, "Step length estimation methods based on inertial sensors: A review," *IEEE Sensors Journal*, vol. 18, no. 17, pp. 6908–6926, 2018.

[27] Nav4You, "Pulse r, indoor and outdoor navigation solution," https://www.nav4you.fr/en/home/#oursolution, accessed: 2025-01-13.

[28] M. Ortiz, M. De Sousa, and V. Renaudin, "A New PDR Navigation Device for Challenging Urban Environments," *Journal of Sensors*, vol. 2017, pp. 1–11, 2017.

[29] u-blox AG, *ZED-F9P-04B Data Sheet*, 2021, accessed: 2025-01-20. [Online]. Available: https://content.u-blox.com/sites/default/files/ZED-F9P-04B_DataSheet_UBX-21044850.pdf

[30] J. Bai, F. Lu, K. Zhang *et al.*, "Onnx: Open neural network exchange," https://github.com/onnx/onnx, 2019.

[31] G. LAB, "Geolocimu," https://play.google.com/store/apps/details?id=fr.univeiffel.geolocimu&hl=en, 2025, accessed: January 27, 2025.

[32] A. Developers, "Read step count data," https://developer.android.com/health-and-fitness/guides/basic-fitness-app/read-step-count-data, [Online; accessed 27-Jan-2025].