TU Delft

Delft University of Technology

Three Symmetries for Data-Driven Pedestrian Inertial Navigation

Wahlstrom, Johan; Kok, Manon

DOI
10.1109/JSEN.2022.3146646

Publication date
2022

Document Version
Final published version

Published in
IEEE Sensors Journal

Citation (APA)
Wahlstrom, J., & Kok, M. (2022). Three Symmetries for Data-Driven Pedestrian Inertial Navigation. *IEEE Sensors Journal*, *22*(6), 5797-5805. https://doi.org/10.1109/JSEN.2022.3146646

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

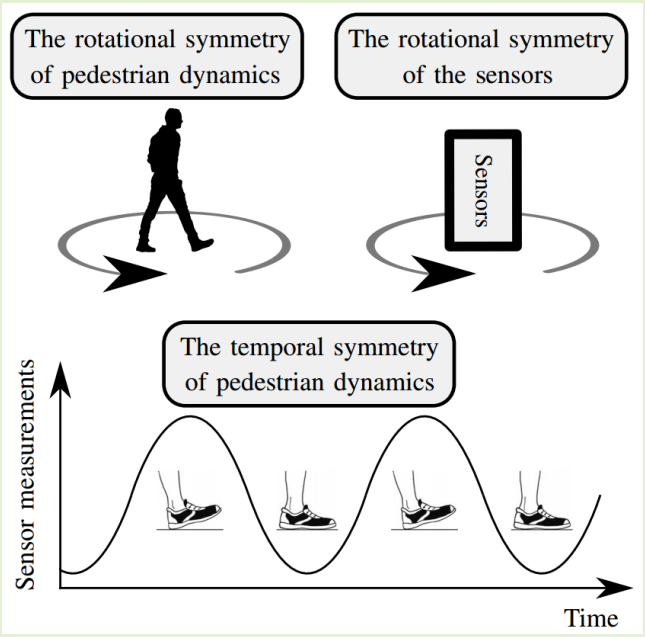
This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Three Symmetries for Data-Driven Pedestrian Inertial Navigation

Johan Wahlström and Manon Kok

***Abstract*—The last years have seen a growing body of literature on data-driven pedestrian inertial navigation. However, despite this, it is still unclear how to efficiently combine classical models and other a priori information with existing machine learning frameworks. In this paper, we first categorize existing approaches to data-driven pedestrian inertial navigation, including approaches where a machine learning algorithm is embedded into an overarching classical framework and purely data-driven frameworks. We then propose an estimation framework where navigation estimates obtained by classical means are fed to a machine learning algorithm which is trained to correct and improve the estimates. Further, we describe three symmetries that can be used to constrain the proposed estimation framework and thereby improve its performance. These are 1) the rotational symmetry of pedestrian dynamics, 2) the rotational symmetry of the sensors, and 3) the temporal symmetry of pedestrian dynamics. To demonstrate the usefulness of the proposed framework, we use data from foot-mounted inertial sensors utilizing zero-velocity updates under mixed walking and running. Machine learning corrections are implemented using both neural networks and Gaussian processes.**

***Index Terms*—Inertial navigation, Gaussian processes, neural networks, pedestrian navigation.**



I. Introduction

TRADITIONALLY, the idea of inertial navigation has been to estimate position by integrating accelerometer and gyroscope measurements twice and thrice, respectively [1]. This idea is formalized by the kinematic inertial navigation equations, which are the foundation of a wide range of systems for pedestrian navigation. Within foot-mounted inertial navigation, the kinematic inertial navigation equations are combined with a zero-velocity model, which adds pseudo measurements of zero velocity whenever the sensor measurements indicate that the foot is stationary [2].

Motivated by the widespread success of machine learning, there have been several attempts at designing data-driven pedestrian inertial navigation systems [3]–[18]. However, despite a considerable amount of literature on the topic, data-driven approaches have not yet outmaneuvered classical algorithms in terms of performance. One potential explanation is that inertial sensors are governed by exact physical laws, described by the kinematic inertial navigation equations. Thus, if the implementation is not sophisticated enough, the main function of the machine learning algorithm will simply be to relearn a model that is already known. In this case, the marginal benefit of utilizing machine learning will be small.

However, there is nothing within the concept of inertial navigation itself that dictates that machine learning algorithms should not be of use. Quite the contrary, as has been shown multiple times, sensor errors and modeling errors lead to systematic estimation errors that cannot be described by standard,

Manuscript received December 31, 2021; accepted January 23, 2022. Date of publication February 10, 2022; date of current version March 14, 2022. This work was supported in part by the Sensor Fusion for Indoor Localization Using the Magnetic Field under Project 18213 of the Research Program Veni funded by the Dutch Research Council [Nederlandse Organisatie voor Wetenschappelijk Onderzoek (NWO)]. The associate editor coordinating the review of this article and approving it for publication was Dr. Yulong Huang. *(Corresponding author: Johan Wahlström.)*

Johan Wahlström is with the Department of Computer Science, University of Exeter, Exeter EX4 4QF, U.K. (e-mail: j.wahlstrom@exeter.ac.uk).

Manon Kok is with the Delft Center for Systems and Control, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: m.kok-1@tudelft.nl).

Digital Object Identifier 10.1109/JSEN.2022.3146646

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

classical estimation frameworks using the kinematic inertial navigation equations, the zero-velocity model, and/or heuristic models for step estimation [19], [20]. Thus, the limited gain of data-driven approaches should not be taken for granted. Rather, as we will argue in the present paper, the lack of progress is a result of applying machine learning algorithms in a way that fails to utilize the distinct symmetries embedded into the pedestrian inertial navigation problem.

To begin with, this paper proposes an estimation framework where machine learning methods are used to correct estimates produced by classical methods for pedestrian inertial navigation. The proposed estimation framework is then used to illustrate three spatial and temporal symmetries that can be exploited to improve the estimation performance. While some of these symmetries have been utilized in previous work, this is the first time that they are formalized and described in a single, unified framework. Finally, the efficiency of the estimation framework is illustrated using mixed walking data from foot-mounted inertial sensors, with the model-based estimation framework implemented as a zero-velocity-aided inertial navigation system.

II. Research Context

In this section, we will first review the main idea of inertial navigation as well as previous work on data-driven pedestrian inertial navigation. Following this, we will motivate our approach to data-driven pedestrian inertial navigation and present the model that will be used in the remainder of the paper.

A. Inertial Navigation

Inertial navigation benefits from a kinematic equation that directly relates the sensor measurements to the navigation quantities of interest. In discretized and linearized form, this equation can be written as $\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}_k$, where $\mathbf{x}$ denotes the navigation state (comprising three-dimensional position $\mathbf{p}$, velocity $\mathbf{v}$, and orientation $\mathbf{R}$), $\mathbf{u}$ denotes the inertial measurements, and $\mathbf{w}$ denotes process noise. Refer to [21] for more details on the kinematic inertial navigation equations $\mathbf{f}$. Due to the threefold gyroscope integration, standalone navigation based on the kinematic inertial navigation equations will lead to a cubic position error growth [22]. Therefore, the equations are typically complemented with a stabilizing measurement model $\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{e}_k$, based on motion models, map information, or measurements from other sensors, thereby resulting in the state-space model

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}_k, \tag{1a}$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{e}_k, \tag{1b}$$

where $\mathbf{y}$ and $\mathbf{e}$ denote measurements and measurement noise, respectively.[1] Given initialization parameters, noise parameters, and measurements $\{\mathbf{u}_k\}_{k=1}^N$ and $\{\mathbf{y}_k\}_{k=1}^N$, it is possible to obtain state estimates $\{\hat{\mathbf{x}}_k\}_{k=1}^N$ by applying a nonlinear filter or smoother to (1); see Fig. 1 (a). However, inertial navigation systems of this form are often plagued by modeling errors and systematic sensor errors (including biases and scale factor errors) [20]. In addition, there may exist motion patterns that are not accounted for in the motion models in $\mathbf{h}(\cdot)$. Next, we will describe how these problems have been tackled using machine learning.

[1] In foot-mounted inertial navigation, a zero-velocity detector produces a binary output which indicates whether the foot is stationary at a given sampling instance. At sampling instances k where the foot is presumed to be stationary, pseudo measurements of zero velocity are added, so that $\mathbf{y}_k = \mathbf{0}$ and $\mathbf{h}(\mathbf{x}_k) = \mathbf{v}_k$.

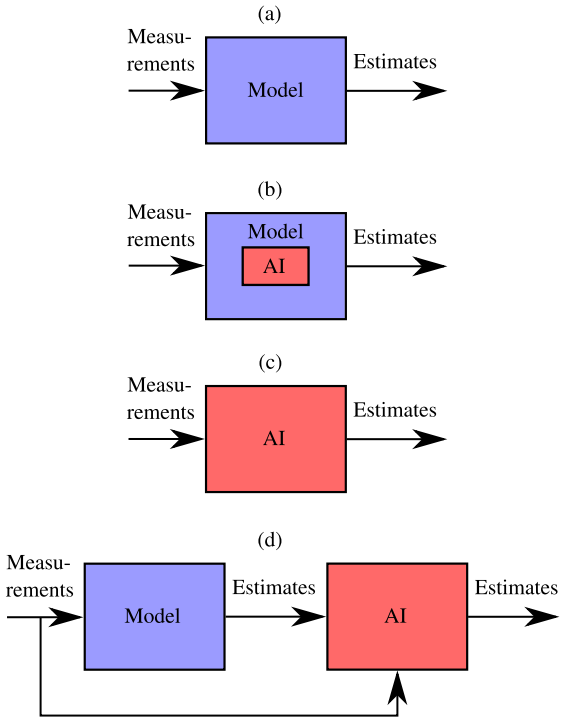


Fig. 1. Process diagrams illustrating the information flow within (a) classical pedestrian inertial navigation (Section II-A), (b) data-driven pedestrian inertial navigation where machine learning is used to solve a problem within an overarching classical framework (Section II-B), (c) data-driven pedestrian inertial navigation which discards classical models (Section II-B), and (d) our approach to data-driven pedestrian inertial navigation (Section II-C).

B. Previous Research on Data-Driven Pedestrian Inertial Navigation

Roughly speaking, there have been two lines of research within data-driven pedestrian inertial navigation. The first approach is to use machine learning to solve a clearly specified problem within an overarching classical framework; see Fig. 1 (b). We will give two examples of this approach. The first example is the use of data-driven zero-velocity detectors [3]–[10], whose output is used as input to the nonlinear filter or smoother that is used to solve (1). The second example is to replace $\mathbf{u}_k$ in (1a) with a function learned from data [18]. Conceptually, this can be described as first correcting inertial measurements using data-driven inference, and then applying the standard inertial navigation equations on these corrected measurements.

The downside of confining a machine learning algorithm within the bounds of a classical algorithm is that the resulting navigation system still will be limited by the constraints

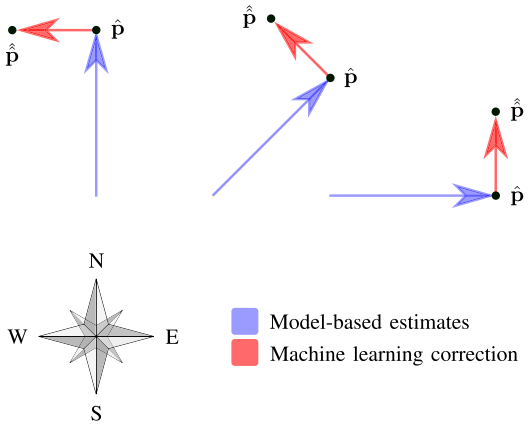


Fig. 2. By utilizing the rotational symmetry of pedestrian dynamics we ensure that the machine learning algorithm produces the same corrections, as seen from the pedestrian, regardless of the walking direction. The blue arrows represent the classically estimated change in position over a specific time interval, and the red arrows represent the associated machine learning correction.

of the classical algorithm. Specifically, in our first example with data-driven zero-velocity detectors, the navigation system will, regardless of how good the detector is, still be limited by the shortcomings of the zero-velocity model (non-zero mean of ZUPT errors, temporal correlation of ZUPT errors, etc.) [20]. Similarly, the downside of replacing $\mathbf{u}_k$ in (1a) with a function learned from data is that this is an unnatural approach for breaking the cubic position error drift resulting from navigation based only on (1a), or for learning motion models related to the navigation state $\mathbf{x}$, for example, motion models that constrain the speed to be within a specific interval.

The second approach to data-driven pedestrian inertial navigation is to discard all applicable models, including the kinematic inertial navigation equations, and force a machine learning algorithm to learn everything from scratch; see Fig. 1 (c). This approach has been taken in [12]–[15], where raw inertial measurements $\mathbf{u}$ or standard statistical features based on these measurements were fed to a deep neural network. The downside here obviously is that the inference framework has to relearn the already well-known kinematic inertial navigation equations (as well as other applicable models) from data. Thereby, we are wasting data and computational power that otherwise could have been spent more wisely.

C. Our Approach to Data-Driven Pedestrian Inertial Navigation

Intuitively, the best way to combine machine learning and classical algorithms is to keep all applicable models and extend them with some structure that is learned from data. The argument for not "throwing away" applicable models should be especially clear as it pertains to the kinematic inertial navigation equations (1a), which are valid in all inertial sensing, regardless of the motion dynamics.[2] The efficiency of incorporating the kinematic inertial navigation equations into a deep learning framework has been demonstrated in [11]. Specifically, it was shown that by feeding a neural network with navigation estimates $\hat{\mathbf{x}}$, instead of with raw inertial measurements $\mathbf{u}$, it is possible to both increase the estimation performance and reduce the computational complexity.

To formalize this discussion, imagine that we first solve the state-space model (1) by classical means (that is, by applying a nonlinear filter or smoother), and then use the resulting navigation estimates $\{\hat{\mathbf{x}}_k\}_{k=1}^N$, possibly in combination with the inertial measurements $\{\mathbf{u}_k\}_{k=1}^N$, as input to a data-driven function $\mathbf{g}$ which outputs improved navigation estimates $\hat{\hat{\mathbf{x}}}$. Thus, when estimating $\mathbf{x}_k$, we arrive at the inference model

$$\hat{\hat{\mathbf{x}}}_k = \mathbf{g}(\hat{\mathbf{x}}_{k-l(k)+1:k}, \mathbf{u}_{k-l(k)+1:k}). \tag{2}$$

Here, $\mathbf{g}$ denotes some chosen data-driven inference algorithm. It should be noted that $\mathbf{g}$ may include a feature extraction step; we don't make the assumption that the arguments $\hat{\mathbf{x}}_{k-l(k)+1:k}$ and $\mathbf{u}_{k-l(k)+1:k}$ are sent directly to a machine learning algorithm.

For simplicity, we have constrained the inference model in (2) to only use estimates and measurements over a sampling window of length $l(k)$, which ends at k. Further, the inference model is assumed to be applied for all $k \in \mathcal{K}$, where $\mathcal{K}$ is some chosen set of sampling instances. It is assumed that $l(k)$ and $\mathcal{K}$ are chosen so that $\mathbf{g}$ uses sampling instances in between two sequential elements in $\mathcal{K}$. For example, if $k_1 \in \mathcal{K}$ and $k_2 \in \mathcal{K}$ while $k \notin \mathcal{K}$ for all $k_1 < k < k_2$, then $l(k_2) = k_2 - k_1$ and $\hat{\hat{\mathbf{x}}}_{k_2} = \mathbf{g}(\hat{\mathbf{x}}_{k_1+1:k_2}, \mathbf{u}_{k_1+1:k_2})$. The discussion of how to choose $l(k)$ is deferred to Section III-C.

The estimation procedure is illustrated in Fig. 1 (d) and summarized as follows.

Estimation framework

1) A nonlinear filter or smoother is applied to the state-space model (1).
Input: Measurements $\{\mathbf{u}_k\}_{k=1}^N$ and $\{\mathbf{y}_k\}_{k=1}^N$
Output: Estimates $\{\hat{\mathbf{x}}_k\}_{k=1}^N$

2) Data-driven inference is performed based on the model (2).
Input: Measurements $\{\mathbf{u}_k\}_{k=1}^N$ and estimates $\{\hat{\mathbf{x}}_k\}_{k=1}^N$
Output: Estimates $\hat{\hat{\mathbf{x}}}_k$ for all $k \in \mathcal{K}$

III. Three Symmetries for Data-Driven Pedestrian Inertial Navigation

In this section, we will use the estimation framework presented in Section II-C to describe three symmetries that may be used in the data-driven inference.

A. The Rotational Symmetry of Pedestrian Dynamics

Pedestrian dynamics is symmetric in the sense that, disregarding the presence of local obstacles such as walls and buildings, a given movement is equally likely to happen in any of the directions in the horizontal plane. Thus, a person's movement characteristics do not differ depending on whether he or she is walking south or north. Likewise, the corrections

[2]Note that although the state-space model (1) describes a navigation system including the kinematic inertial navigation equations, the argument for incorporating applicable models in data-driven inference is also valid for navigation systems based on empirical step length estimation models [23].

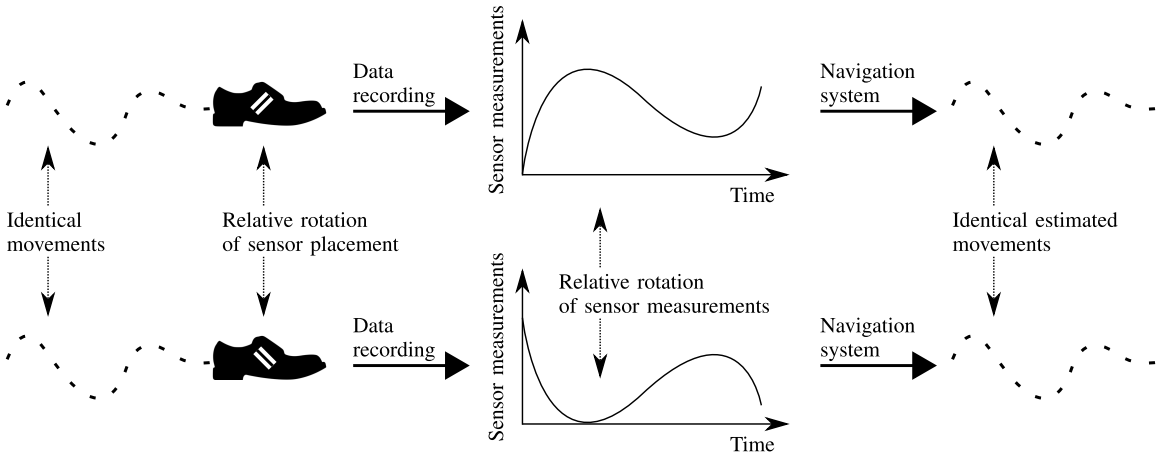


Fig. 3. By utilizing the rotational symmetry of the sensors we ensure that the developed navigation system produces the same estimates under all possible rotations of the inertial measurements. The top row illustrates how a set of pedestrian movements together with a specific sensor placement produces sensor measurements which are used to estimate the pedestrian movements. The bottom row illustrates how the same movements together with a rotated sensor placement will produce rotated sensors measurements which, assuming that we utilize the rotational symmetry of the sensors, will produce the same estimated movements. In this way, the navigation system becomes invariant to the relative orientation of the sensors and the user.

$\hat{\hat{\mathbf{x}}} - \hat{\mathbf{x}}$ should be identical, as seen from the pedestrian, regardless of the walking direction. For example, if $\hat{\mathbf{x}}$ underestimates the stride length, the function $\mathbf{g}$ should adjust the step length estimates accordingly, and do so in the same way for all steps, regardless of the horizontal direction in which the step was taken. Since inertial measurements are independent of the initial horizontal alignment of the sensor unit, this symmetry is automatically embedded into data-driven inference algorithms that only use raw inertial measurements as input. However, when also using navigation estimates as input, as in the model (2), more care must be taken.

To illustrate how to utilize the rotational symmetry of pedestrian dynamics, let's assume that we follow the estimation scheme outlined in Section II-C. Thus, we have obtained the estimates $\{\hat{\mathbf{x}}_k\}_{k=1}^N$ by applying a nonlinear filter or smoother to (1), and then wish to improve upon these estimates by learning some data-driven function $\mathbf{g}$, modeled as in (2). To start with, we will consider a machine learning correction $\hat{\hat{\mathbf{x}}} - \hat{\mathbf{x}}$ only dependent on the velocity $\hat{\mathbf{v}}$ and the rotation $\hat{\mathbf{R}}$. Moreover, we will only consider the output of $\mathbf{g}$ mapped to the position estimates $\hat{\hat{\mathbf{p}}}$.[3] This will be denoted $\hat{\hat{\mathbf{p}}} = \mathbf{g_p}(\hat{\mathbf{x}})$. We then incorporate the rotational symmetry of pedestrian dynamics by constraining this function to be of the form

$$\mathbf{g_p}(\hat{\mathbf{x}}) = \hat{\mathbf{R}}^\mathsf{T}\mathbf{h_p}(\hat{\mathbf{R}}\hat{\mathbf{v}}) + \hat{\mathbf{p}}. \tag{3}$$

In other words, we learn $\mathbf{g_p}(\hat{\mathbf{x}})$ by learning $\mathbf{h_p}(\hat{\mathbf{R}}\hat{\mathbf{v}})$. Here, $\hat{\mathbf{R}}$ is the 3D rotation matrix from the navigation frame to the sensor frame.[4] The output from $\mathbf{h_p}(\hat{\mathbf{R}}\hat{\mathbf{v}})$ is the positional correction $\hat{\hat{\mathbf{p}}} - \hat{\mathbf{p}}$ expressed in the sensor frame. Therefore, $\mathbf{h_p}(\hat{\mathbf{R}}\hat{\mathbf{v}})$ is multiplied by $\hat{\mathbf{R}}$ to rotate the correction to the navigation frame. The formulation in (3) is illustrated in Fig. 2.

In summary, it is possible to see that the formulation in (3) means that the positional correction, as seen by the pedestrian, that is, $\hat{\mathbf{R}}(\hat{\hat{\mathbf{p}}} - \hat{\mathbf{p}}) = \mathbf{h_p}(\hat{\mathbf{R}}\hat{\mathbf{v}})$, will

1) be independent of the walking direction θ.
2) depend on the velocity in the sensor frame $\hat{\mathbf{R}}\hat{\mathbf{v}}$, not on the velocity in the navigation frame $\hat{\mathbf{v}}$.

The velocity output $\hat{\hat{\mathbf{v}}} = \mathbf{g_v}$ is treated in a completely analogous manner. Rotation corrections $\hat{\hat{\mathbf{R}}} = \mathbf{g_R}$ are conceptually analogous, but somewhat more complicated due to the nonlinear algebra of three-dimensional rotations. For brevity, we will not discuss in detail how to utilize the rotational symmetry of pedestrian dynamics for rotation corrections.

B. The Rotational Symmetry of the Sensors

Consider a pedestrian equipped with a body-worn inertial measurement unit. Further, assume that the pedestrian is engaged in some specific motions resulting in the inertial measurements $\{\mathbf{u}_k\}_{k=1}^N$, where the inertial measurements $\mathbf{u}_k = [\mathbf{a}_k^\mathsf{T}\ \boldsymbol{\omega}_k^\mathsf{T}]^\mathsf{T}$ consist of both accelerometer measurements $\mathbf{a}$ and gyroscope measurements $\boldsymbol{\omega}$. Now, consider a separate scenario where the sensor unit is rotated, with respect to the pedestrian, based on the rotation matrix $\mathbf{R}^\star$, but still kept at the same position on the pedestrian's body. In this case, assuming that there are no sensor errors, the same pedestrian motions will result in the inertial measurements $\{\mathbf{u}_k^\star\}_{k=1}^N$, where $\mathbf{u}_k^\star = [(\mathbf{R}^\star\mathbf{a}_k)^\mathsf{T}\ (\mathbf{R}^\star\boldsymbol{\omega}_k)^\mathsf{T}]^\mathsf{T}$ for all k. In other words, a rotation of the sensor unit placement leads to a rotation of the resulting inertial measurements.

As described in the above paragraph, a given set of movements can lead to different measurements, as dependent on the rotation of the sensors with respect to the user. Ideally, all these different measurements should produce the same

[3]Note that the position estimates $\hat{\mathbf{p}}$ may not be consistent with the speed estimates $\hat{\mathbf{v}}$.

[4]One alternative is to use a rotation matrix that only performs a yaw rotation, since pedestrian dynamics only is invariant with respect to rotations around the vertical axis. However, based on our experiments, this does not lead to any performance improvement.

navigation estimates (that is, the correct ones). Therefore, it is intuitive to constrain the machine learning algorithm to produce the same output under all possible rotations of the inertial measurements. Applying this to the estimation scheme presented in Section II-C, we arrive at the constraint

$$\mathbf{g}(\hat{\mathbf{x}}_{k-l(k)+1:k}, \mathbf{u}_{k-l(k)+1:k}) = \mathbf{g}(\hat{\mathbf{x}}_{k-l(k)+1:k}, \mathbf{u}^{\star}_{k-l(k)+1:k}) \quad (4)$$

for all $k \in \mathcal{K}$, where

$$\mathbf{u}^{\star}_k = [(\mathbf{R}^{\star}\mathbf{a}_k)^{\mathsf{T}} \ (\mathbf{R}^{\star}\boldsymbol{\omega}_k)^{\mathsf{T}}]^{\mathsf{T}} \quad (5)$$

for all k and any rotation matrix $\mathbf{R}^{\star}$ that is independent of k. The constraint is illustrated in Fig. 3. One way to implement this constraint is by feeding the machine learning algorithm with lots of examples where the constraint is true. Specifically, given a training example $\{\hat{\mathbf{x}}^{\diamond}_k, \hat{\mathbf{x}}^{\diamond}_{k-l(k)+1:k}, \mathbf{u}^{\diamond}_{k-l(k)+1:k}\}$ where $\mathbf{u}^{\diamond}_k = [(\mathbf{a}^{\diamond}_k)^{\mathsf{T}} \ (\boldsymbol{\omega}^{\diamond}_k)^{\mathsf{T}}]^{\mathsf{T}}$, it is possible to create a new training example $\{\hat{\mathbf{x}}^{\diamond}_k, \hat{\mathbf{x}}^{\diamond}_{k-l(k)+1:k}, \mathbf{u}^{\diamond\star}_{k-l(k)+1:k}\}$, where $\mathbf{u}^{\diamond\star}_k = [(\mathbf{R}^{\star}\mathbf{a}^{\diamond}_k)^{\mathsf{T}} \ (\mathbf{R}^{\star}\boldsymbol{\omega}^{\diamond}_k)^{\mathsf{T}}]^{\mathsf{T}}$ for all k. This method has previously been used in [5]. A more rigorous way of enforcing the constraint (4) could be to integrate it into a constrained neural network [24].

By utilizing the constraint (4), the inference will become invariant to the relative orientation of the sensor unit and the user. This is particularly useful in scenarios that permit the sensor unit to be positioned at an arbitrary orientation with respect to the user. However, the drawback of the constraint is that it is derived under the assumption of small sensor errors,[5] and it may therefore be unsuitable in situations where this assumption is not true. In Section IV-D, we investigate the marginal benefit of utilizing the rotational symmetry of the sensors in a foot-mounted inertial navigation system.

C. The Temporal Symmetry of Pedestrian Dynamics

We will now discuss how to set the window length $l(k)$ over which to perform the data-driven inference (see Section II-C). Several publications have used sampling windows of equal time length, that is, $l(k)$ is set as a constant l. Typical time lengths are one second [15], [18] and two seconds [11], [14]. However, when the window length is set as a constant, you have no control over what part of the gait cycle that is included in a given sampling window. In practice, this means that you force the machine learning algorithm to learn by itself, from data, how to interpret the context of a given set of measurements.

A more natural approach is to adapt the window length to the temporal length of a step [13]. In this way, we can improve the estimation performance by exploiting the fact that pedestrians move about by taking a large number of individual walking steps with similar gait characteristics. To implement this approach, we first need to use a step detector to find the sampling instances where each individual step begins and ends. We then set $l(k)$ equal to the length of the step (measured in sampling instances) that ends at sampling instance k. The difference between a fixed window length and a window length adapted to the temporal length of a step is illustrated in Fig. 4. In Section IV-E, we compare the performance of fixed and adaptive window lengths within foot-mounted inertial navigation.

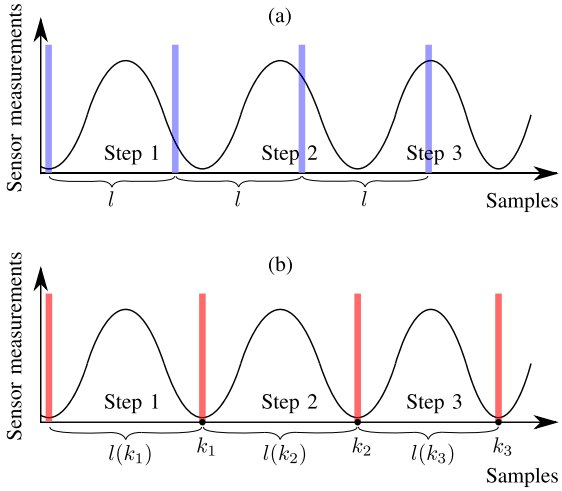


Fig. 4. The window length $l(k)$ over which to perform the data-driven inference can either (a) be set as a constant $l(k) = l$, or (b) be adapted to the temporal length of a step.

IV. Numerical Results

This section demonstrates the performance of the estimation framework described in Section II-C when utilizing the symmetries presented in Section III. The code and the data is available at https://github.com/johanwahlst/Three-Symmetries. The performance evaluation was based on the data set described in [25]. This data set includes measurements from a foot-mounted inertial measurement unit[6] (Xsens MTx-28A53G25, temperature-compensated internally by the device) as well as ground truth position and orientation data from a high-accuracy camera tracking system (Bonita by Vicon) using eight infrared (IR) cameras and strobes. All data had a sampling rate of 100 [Hz]. Refer to [25] for more details on the experimental setup.

The considered trajectory consisted of about five minutes of mixed walking and running in a lab area of about 20 [m^2] and is illustrated in Fig. 5. The model-based estimation forming the first half of the inference framework described in Section II-C (the blue box in Fig. 1 (d)) was a foot-mounted inertial navigation system utilizing zero-velocity updates [2]. The estimates were computed using an extended Kalman smoother [26] implemented with the SHOE detector [27]. The position, velocity, and orientation estimates were initialized with a standard deviation of 10^{-5} [m], 10^{-5} [m/s], 100 [°]

[5] Assume that there is additive accelerometer noise, so that $\mathbf{a}_k = \mathbf{a}^{\text{true}}_k + \delta\mathbf{a}_k$, where $\mathbf{a}^{\text{true}}_k$ represents the true accelerometer dynamics and $\delta\mathbf{a}_k$ is the accelerometer noise. Following the rotation $\mathbf{R}^{\star}$, we will obtain the accelerometer measurements $\mathbf{a}^{\star}_k = \mathbf{R}^{\star}\mathbf{a}^{\text{true}}_k + \delta\mathbf{a}_k$. In this case, the only way in which we can have $\mathbf{a}^{\star}_k = \mathbf{R}^{\star}\mathbf{a}_k$ for all rotation matrices $\mathbf{R}^{\star}$ is if $\delta\mathbf{a}_k = \mathbf{0}$. The analogous argument can be made for the gyroscope measurements.

[6] In this paper, we have chosen to use data from foot-mounted inertial sensors since this is the sensor placement producing the most accurate pedestrian inertial odometry. However, note that the three symmetries presented in Section III are equally applicable to data from other sensor placements, including measurements from smartphone-embedded sensors.

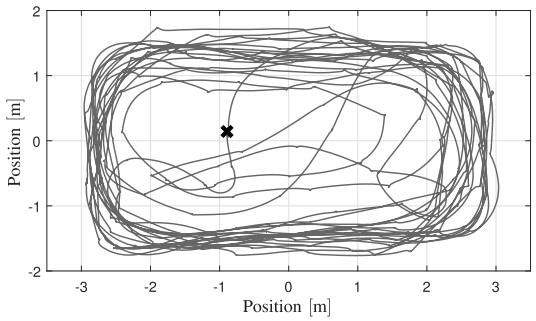


Fig. 5. Ground truth positions for the trajectory used in the experiments. The starting position is marked with 'X'. The data was taken from ID 15 in [25].

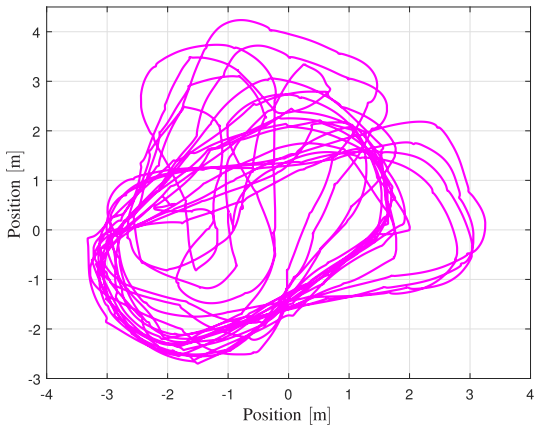


Fig. 6. Position estimates from foot-mounted inertial navigation system.

(roll and pitch angles), and 0.1 [°] (yaw angle), respectively. The first sampling instance used in the smoother was chosen to be a sampling instance with a detected zero-velocity event, and the initial velocity estimates were therefore set to zero. Further, the position and yaw estimates were initialized to align with the ground truth data,[7] and the roll and pitch angles were estimated based on the 20 first accelerometer readings (all during standstill). The resulting position estimates are shown in Fig. 6.

In the remainder of this section, we will discuss how to implement a machine learning correction (the red box in Fig. 1 (d)) to the inertial odometry in Fig. 6 with the aim of getting as close as possible to the ground truth in Fig. 5. The general setup is described in Sections IV-A and IV-B. The main results are presented in Section IV-C, while Sections IV-D and IV-E detail further investigations of the symmetries discussed in Sections III-B and III-C, respectively. The relation between Section III and the numerical results presented in this section is detailed at the end of Section IV-A.

A. Feature Extraction

This subsection describes how the features that were used as input to the machine learning algorithm were computed from the estimates provided by the foot-mounted inertial navigation system. A step segmentation algorithm based on detected zero-velocity events was first used to identify sampling instances that separate the individual steps. These sampling instances formed the set $\mathcal{K}$ at which to apply machine learning corrections (see Section II-C). Thus, if we assume that $\mathcal{K} = \{k_1, k_2, k_3, \dots\}$, where all sampling instances in $\mathcal{K}$ correspond to detected zero-velocity events, the first step occurred between sampling instances k_1 and k_2, the second step occurred between sampling instances k_2 and k_3, etc.

We then computed the change in position, as seen from the sensor frame, over each individual step. For example, assume that two sequential steps start at sampling instances k_1 and k_2. The three-dimensional position differential over the corresponding step was then computed as

$$\mathbf{s}_{k_1} = \hat{\mathbf{R}}_{k_1}(\hat{\mathbf{p}}_{k_2} - \hat{\mathbf{p}}_{k_1}) \tag{6}$$

where $\hat{\mathbf{p}}_{k_1}$ and $\hat{\mathbf{p}}_{k_2}$ are the positions at the start and end of the step, respectively, while $\hat{\mathbf{R}}_{k_1}$ is the matrix describing rotations from the navigation frame to the sensor frame at sampling instance k_1. Fig. 7 illustrates the first two (horizontal) elements in the position differentials as computed from the foot-mounted inertial navigation system and the ground truth data over the full trajectory. In a few instances, ground truth data was missing at sampling instances in $\mathcal{K}$. The corresponding steps were simply excluded from the training and testing phases. We experimented with including raw inertial measurements or simple statistical quantities based on these measurements as features, however, this was not found to improve performance.

Looking back at the symmetries described in Section III it can be seen that

- The rotational symmetry of pedestrian dynamics was utilized by expressing the features in the sensor frame rather than in the navigation frame. That is, we used $\hat{\mathbf{R}}_{k_1}(\hat{\mathbf{p}}_{k_2} - \hat{\mathbf{p}}_{k_1})$ rather than $\hat{\mathbf{p}}_{k_2} - \hat{\mathbf{p}}_{k_1}$.[8]
- The temporal symmetry of pedestrian dynamics was utilized by computing features based on the dynamics over a given step.

The rotational symmetry of the sensors could not be utilized with the features described in this section. Instead, this symmetry was utilized in the experiments described in Section IV-D.

B. Training and Testing

The machine learning algorithms were used to correct both the estimated position $\hat{\mathbf{p}}$ and the estimated yaw angle $\hat{\theta}$.[9] Thus, during the training phase, a training example was created by pairing up the feature $\mathbf{s}_{k_1}$, computed using estimates from the foot-mounted inertial navigation system, with the outputs $\mathbf{p}_{k_2} - \mathbf{p}_{k_1}$ and $\theta_{k_2} - \theta_{k_1}$, computed using ground truth data. In order to utilize the entire trajectory for both training and testing, we employed 10-fold cross-validation.

[7] Before we could start training and testing machine learning algorithms, it was necessary to align the position and orientation estimates obtained from the zero-velocity-aided inertial navigation system with the ground truth data (Fig. 6 shows the position estimates after alignment). This process is described in the appendix.

[8] Conceptually, using the change in position over a small time interval as a feature is similar to using the speed; compare with the discussion in Section III-A.

[9] These are the only navigation quantities whose estimation errors will grow with time in a foot-mounted inertial navigation system with intermittent zero-velocity updates.

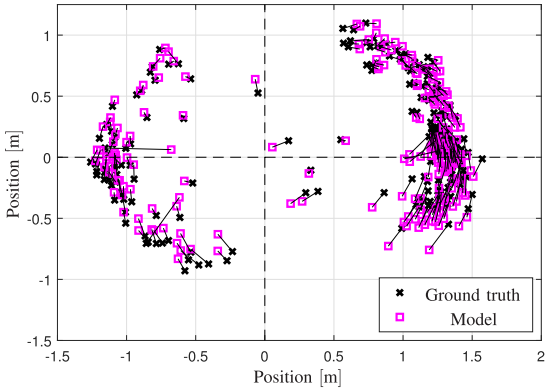


Fig. 7. The change in horizontal position over steps as measured by ground truth data and as estimated by the model. Markers corresponding to the same step are connected by a black line. All steps are shown in the coordinate frame of the pedestrian; markers to the left and right represent backwards and forwards steps, respectively.

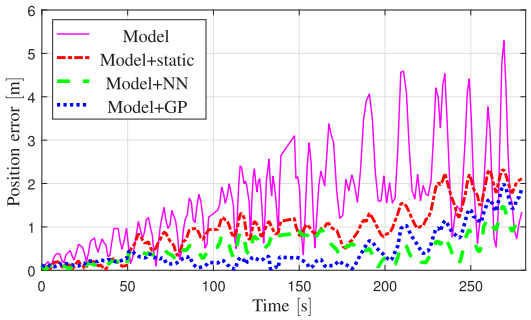


Fig. 8. The horizontal position error as dependent on time.

Thus, within each iteration, 90% of the steps were used for training and validation, and the remaining 10% were used for testing. In this way, each iteration produced slightly different machine learning corrections. By iterating over all available data, we eventually obtained corrections for the whole data set, and obtained improved position estimates by sequentially correcting the position and yaw estimates after each step.

Two machine learning algorithms were compared: a neural network (NN) and a Gaussian process (GP). Both the NN and the GP were trained separately for each output dimension. The NN was a feedforward network with a single hidden layer, two neurons, and a hyperbolic tangent activation function. The NN was trained using the Levenberg-Marquardt algorithm to minimize the mean-square error over the validation set. $0.75 \cdot 90\%$ of the data was used for training, and $0.25 \cdot 90\%$ of the data was used for validation. The GP used a squared exponential covariance function. The kernel parameters and the noise variance were estimated by means of marginal likelihood maximization; thus, the GP did not use a dedicated validation data set. The NN and the GP were deliberately chosen to be relatively simple, to illustrate that the proposed estimation framework does not require intricate parameter tuning and is easy to reproduce with standard algorithms. The NN and the GP were benchmarked against a static correction which, after each step, subtracted the step-wise mean position and yaw error from the position and yaw estimates, respectively.

TABLE I
POSITION RMSE OF THE CONSIDERED INFERENCE FRAMEWORKS

Inference framework	Position RMSE [m]
Model	1.94
Model+static	1.11
Model+NN	0.59
Model+GP	0.64

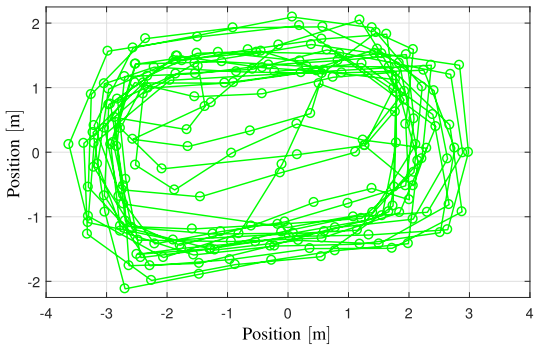


Fig. 9. Position estimates after neural network correction.

C. Results

Fig. 8 compares the horizontal position error of the stand-alone foot-mounted inertial navigation system with the navigation systems utilizing static corrections, NN corrections, and GP corrections. While the error growth was significantly reduced with a simple static correction, the performance was enhanced even further with NN and GP corrections. As demonstrated in Table I, the NN and GP managed to reduce the position root-mean-square error (RMSE) computed over the full data set by more than a factor of three. The periodic pattern seen in Fig. 8 is caused by the yaw drift of the foot-mounted inertial navigation system. This yaw drift is also evident when comparing Fig. 5 and Fig. 6, or when studying Fig. 7. As seen in Fig. 9, the NN correction produces a much slower yaw drift.

D. The Rotational Symmetry of the Sensors

To investigate the utility of the rotational symmetry of the sensors we had to use features that were dependent on the rotation of the sensors with respect to the pedestrian. For this purpose, we discarded the features described in Section IV-A, and instead considered a three-dimensional feature set consisting of the variance of the gyroscope measurements over the considered step, computed along each spatial dimension. Further, the training data set was augmented by generating a number of randomized rotations, and then, for each such rotation, using this to rotate the gyroscope measurements before re-computing the features and adding them to the training data set (this was previously described in Section III-B). In this way, we simulated multiple sensor-pedestrian orientations in the training data set. Each example in the original training data set was used both in its original form and in combination with all randomized rotations. Thus, using n randomized rotations meant that the size of the training data set was increased by a

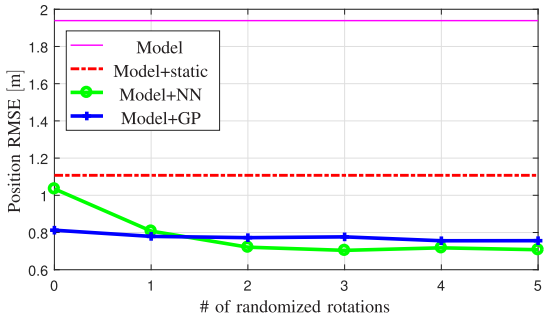


Fig. 10. The horizontal position error as dependent on to what extent the estimates utilize the rotational symmetry of the sensors.

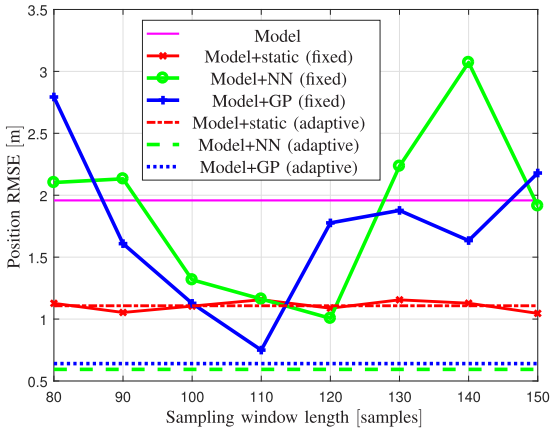


Fig. 11. The position root-mean-square error obtained with static and adaptive sampling window lengths.

factor of $n+1$. Fig. 10 shows the resulting position RMSEs when using up to five randomized orientations. The results were averaged over 10 simulations. As can be seen, the NN obtains a significant performance improvement as a result of utilizing this symmetry, while the performance improvement of the GP is marginal.

E. The Temporal Symmetry of Pedestrian Dynamics

To investigate the utility of an adaptive sampling window length, we compared the position RMSE obtained with adaptive and fixed sampling window lengths. Fixed sampling window lengths of $80, 90, \ldots, 160$ samples were considered. For comparison, the adaptive sampling window lengths ranged from 73 to 180 samples, with a median length of 118 samples. As seen in Fig. 11, the adaptive sampling window length outperformed all fixed sampling window lengths for both the NN and the GP. With a static correction, the adaptive and fixed sampling window lengths had roughly the same performance. One explanation for why an adaptive sampling window length is particularly useful in *foot-mounted* inertial navigation, is that it enables the training data to be computed using only estimates from detected zero-velocity events. Generally, the estimation uncertainty of a foot-mounted inertial navigation system will be lower in the temporal vicinity of zero-velocity updates [26]. Therefore, the uncertainty of the position differential $\mathbf{s}_{k_1}$ defined in (6) will be lower if the indices k_1 and k_2 are taken from detected zero-velocity events. In addition, the ground truth data will be less variable in the temporal vicinity of zero-velocity updates, which means that the output data will be more robust against eventual timestamp errors at these sampling instances. In summary, an adaptive sampling window length can be said to improve the accuracy of both the feature extraction and the output data. Theoretical and experimental support of the usefulness of adaptive sampling window length in data-driven foot-mounted inertial navigation is provided by the reasoning above and the results shown in Fig. 11, respectively. However, further research is needed to validate these results in more diverse scenarios and to study the performance of adaptive and fixed sampling window length for other sensor placements.

One benefit of the GP in situations with large errors in the feature extraction is that the estimated noise variance can provide information about the quality of the training data. In comparison to the NN and the GP, the static correction is both less flexible and less sensitive to outliers in the training data, which also explains its stable performance in Fig. 11.

V. Summary

This article has proposed an inference framework for pedestrian inertial navigation where machine learning algorithms are used to correct estimates produced by classical methods. Further, we described three symmetries that can be integrated into the inference to improve the estimation performance. These are the rotational symmetry of pedestrian dynamics, the rotational symmetry of the sensors, and the temporal symmetry of pedestrian dynamics. Finally, data from a foot-mounted inertial navigation system was used to demonstrate the performance of the proposed inference framework and the benefit of the symmetries. In particular, it was shown that an adaptive sampling window length clearly outperforms all fixed sampling window lengths.

Appendix

Four different alignments were made.

1) Rotational alignment of the position estimates.
2) Translational alignment of the position estimates.
3) Temporal alignment.
4) Alignment of the rotation estimates.

The first two alignments were made by minimizing the position RMSE of the foot-mounted inertial navigation system over the first three seconds of data (there is no point in using all available data since a foot-mounted inertial navigation system always drifts with time, even with perfect alignment). The first alignment was used to ensure that the initial walking direction was the same for the two trajectories. The second alignment was used to ensure that the two trajectories started at the same position.

The third and fourth alignments were made by minimizing the root-mean-square error of the roll and pitch angles (the roll and pitch estimates do not drift with time and it was therefore possible to use all available data for the alignment). The third alignment was used to remove any eventual constant offset in the timestamps. The fourth alignment rotated the orientation estimates to align these with the ground truth orientation data.

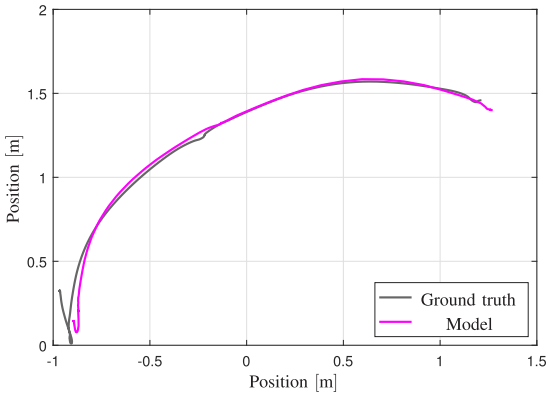


Fig. 12. Ground truth position and position estimates from the foot-mounted inertial navigation system over the first three seconds of data.

The optimization problems were solved using standard nonlinear least-squares solvers [28]. The solutions were then validated by visual inspection. As an example, Fig. 12 shows the position estimates over the first three seconds of data.

REFERENCES

[1] M. Kok, J. D. Hol, and T. B. Schön, "Using inertial sensors for position and orientation estimation," *Found. Trends Signal Process.*, vol. 11, nos. 1–2, pp. 1–153, 2017.

[2] J. Nilsson, A. K. Gupta, and P. Händel, "Foot-mounted inertial navigation made easy," in *Proc. IEEE Int. Conf. Indoor Positioning Indoor Navigat.*, Oct. 2014, pp. 24–29.

[3] S. Y. Park, H. Ju, and C. G. Park, "Stance phase detection of multiple actions for military drill using foot-mounted IMU," in *Proc. IEEE Int. Conf. Indoor Positioning Indoor Navigat.*, Alcala de Henares, Spain, Oct. 2016, p. 16.

[4] J. Wahlström, I. Skog, F. Gustafsson, A. Markham, and N. Trigoni, "Zero-velocity detection—A Bayesian approach to adaptive thresholding," *IEEE Sensors Lett.*, vol. 3, no. 6, pp. 1–4, Jun. 2019.

[5] B. Wagstaff and J. Kelly, "LSTM-based zero-velocity detection for robust inertial navigation," in *Proc. IEEE Int. Conf. Indoor Positioning Indoor Navigat.*, Nantes, France, Sep. 2018, pp. 1–8.

[6] H. Zhao *et al.*, "Adaptive gait detection based on foot-mounted inertial sensors and multi-sensor fusion," *Inf. Fusion*, vol. 52, pp. 157–166, Dec. 2019.

[7] X. Yu *et al.*, "AZUPT: Adaptive zero velocity update based on neural networks for pedestrian tracking," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Waikoloa, HI, USA, Dec. 2019, pp. 1–6.

[8] T. Zhu, X. Pan, and S. Zhang, "A zero velocity detection method for soldier navigation based on deep learning," *J. Phys., Conf. Ser.*, vol. 1345, no. 3, Nov. 2019, Art. no. 032018.

[9] Y. Kone, N. Zhu, V. Renaudin, and M. Ortiz, "Machine learning based zero-velocity detection for inertial pedestrian navigation," *IEEE Sensors J.*, vol. 20, no. 20, pp. 12343–12353, Oct. 2020.

[10] J. Wahlström, A. Markham, and N. Trigoni, "FootSLAM meets adaptive thresholding," *IEEE Sensors J.*, vol. 20, no. 16, pp. 9351–9358, Dec. 2020.

[11] R. Khorrambakht, H. Damirchi, and H. D. Taghirad, "Preintegrated IMU features for efficient deep inertial odometry," 2020, *arXiv:2007.02929*.

[12] T. Feigl, S. Kram, P. Woller, R. H. Siddiqui, M. Philippsen, and C. Mutschler, "A bidirectional LSTM for estimating dynamic human velocities from a single IMU," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Pisa, Italy, Sep. 2019, pp. 1–8.

[13] I. Klein and O. Asraf, "StepNet-deep learning approaches for step length estimation," *IEEE Access*, vol. 8, pp. 85706–85713, 2020.

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

[15] O. Asraf, F. Shama, and I. Klein, "PDRNet: A deep-learning pedestrian dead reckoning framework," *IEEE Sensors J.*, early access, Mar. 17, 2021, doi: 10.1109/JSEN.2021.3066840.

[16] Y. Li *et al.*, "Inertial sensing meets artificial intelligence: Opportunity or challenge?" 2020, *arXiv:2007.06727*.

[17] S. Cortés, A. Solin, and J. Kannala, "Deep learning based speed estimation for constraining strapdown inertial navigation on smartphones," in *Proc. IEEE Int. Conf. Mach. Learn. Signal Process. (MLSP)*, Aalborg, Denmark, Sep. 2018, pp. 1–6.

[18] H. Yan, Q. Shan, and Y. Furukawa, "RIDI: Robust IMU double integration," in *Proc. Eur. Conf. Comput. Vis.*, Munich, Germany, Sep. 2018, pp. 621–636.

[19] J.-O. Nilsson, I. Skog, and P. Händel, "A note on the limitations of ZUPTs and the implications on sensor error modeling," in *Proc. IEEE Int. Conf. Indoor Positioning Indoor Navigat.*, Sydney, NSW, Australia, Nov. 2012, pp. 1–4.

[20] J. Wahlström and I. Skog, "Fifteen years of progress at zero velocity: A review," *IEEE Sensors J.*, vol. 21, no. 2, pp. 1139–1151, Jan. 2021.

[21] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, Norwood, MA, USA: Artech House, 2008.

[22] Y. Wang, A. Chernyshoff, and A. M. Shkel, "Error analysis of ZUPT-aided pedestrian inertial navigation," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2018, pp. 206–212.

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

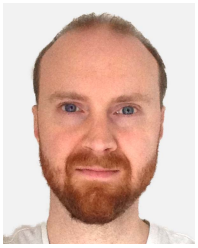
[24] J. Hendriks, C. Jidling, A. Wills, and T. Schön, "Linearly constrained neural networks," 2020, *arXiv:2002.01600*.

[25] M. Angermann, P. Robertson, T. Kemptner, and M. Khider, "A high precision reference data set for pedestrian navigation using foot-mounted inertial sensors," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat.*, Zurich, Switzerland, Sep. 2010, pp. 1–6.

[26] D. S. Colomar, J. Nilsson, and P. Händel, "Smoothing for ZUPT-aided INSs," in *Proc. IEEE Int. Conf. Indoor Positioning Indoor Navigat.*, Sydney, NSW, Australia, Nov. 2012, pp. 1–5.

[27] I. Skog, P. Handel, J. O. Nilsson, and J. Rantakokko, "Zero-velocity detection—An algorithm evaluation," *IEEE Trans. Biomed. Eng.*, vol. 57, no. 11, pp. 2657–2666, Nov. 2010.

[28] H. Mohammad, M. Y. Waziri, and S. A. Santos, "A brief survey of methods for solving nonlinear least-squares problems," *Numer. Algebra, Control Optim.*, vol. 9, no. 1, p. 1, 2019.



Johan Wahlström received the B.Sc., M.Sc., and Ph.D. degrees from the KTH Royal Institute of Technology, Stockholm, Sweden, in 2013, 2014, and 2017, respectively. Between 2018 and 2020, he was a Postdoctoral Researcher with the University of Oxford. He is currently a Lecturer of Data Science with the University of Exeter. In 2015, he spent one month with the University of Porto and six months at Washington University in St. Louis as a Visiting Ph.D. student. In 2016, he spent two months at MIT spin-off Cambridge Mobile Telematics. He was accepted into the program of excellence in electrical engineering at KTH in 2014, and in 2015. He was the youngest recipient of the Sweden-America Foundation's Research Scholarship.

Manon Kok received the M.Sc. degrees in applied physics and in philosophy of science, technology and society from the University of Twente, Enschede, The Netherlands, in 2007 and 2009, respectively, and the Ph.D. degree in automatic control from Linköping University, Linköping, Sweden, in 2017. From 2009 to 2011, she was a Research Engineer with Xsens Technologies. From 2017 to 2018 she was a Postdoctoral Researcher with the Computational and Biological Learning Laboratory, Machine Learning Group, University of Cambridge, Cambridge, U.K. She is currently an Assistant Professor with the Delft Center for Systems and Control, Delft University of Technology, The Netherlands. Her research interests include the fields of probabilistic inference for sensor fusion, signal processing, and machine learning.