Doctor of Philosophy

Human Movement Measurement using Wearable Inertial Sensors and Deep Learning

The Graduate School of the University of Ulsan

Department of Electrical, Electronic and Computer Engineering

THANH TUAN PHAM

Human Movement Measurement using Wearable Inertial Sensors and Deep Learning

Supervisor: Professor Young Soo Suh

A Dissertation

Submitted to
the Graduate School of the University of Ulsan
In partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

by

Thanh Tuan Pham

Department of Electrical, Electronic and Computer Engineering
University of Ulsan, Korea
August 2022

Human Movement Measurement using Wearable Inertial Sensors and Deep Learning

This certifies that the dissertation of Thanh Tuan Pham is approved by:

[illegible]

Committee Chair: Professor Hee Jun Kang

[illegible]

Committee Member: Professor Kang Hyun Jo

[illegible]

Committee Member: Professor Jaehyun Park

Lee

Committee Member: Professor Hyung Keun Lee

[illegible]

Committee Member: Professor Young Soo Suh

Department of Electrical, Electronic and Computer Engineering

University of Ulsan, Korea

August 2022

Dedicated to my family,

thanks for your love and support

Human Movement Measurement using Wearable Inertial Sensors and Deep Learning

by

Thanh Tuan Pham

Submitted to the Department of Electrical, Electronic and Computer Engineering
on August, 2022, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

With advances in Micro-Electro-Mechanical Systems (MEMS) technology, inertial sensors have become smaller, cheaper, and are commonly integrated into wearable smart devices, in which their applications have many benefits for our daily lives. They are used in a wide range of applications in human gait analysis, sports training, and healthcare applications. Besides, human movement measurement is an interesting and active topic in sports and medical applications, as it can help people to evaluate physical activities and sports in the daily living environment. Therefore, we focus on the methods for human movement measurement using wearable inertial sensors which are attached to the pedestrian's waist.

In this dissertation, the inertial navigation algorithm and deep learning approach for human movement measurement using wearable inertial sensors are proposed to obtain the kinematic and temporal gait parameters. The kinematic parameters, which include the attitude, position, and velocity of the human body, are estimated using the inertial navigation algorithm based on the smoothing algorithm. The temporal gait parameters, such as step length and walking distance, are computed by double integrating acceleration. However, low-cost inertial sensors always have noise and bias that lead to the integration error over time, the algorithm is not possible to estimate the kinematic and temporal gait parameters accurately even for a short distance. Therefore, a known distance straight-line walking trajectory constraint and a constant speed constraint are imposed in the smoothing algorithm. These constraints reduce the accumulation of the integration error even for long walking distances. If the pedestrian walks with a complex walking path or very long distance, the smoothing algorithm needs a very long computation time and large memory requirement to obtain the kinematic and temporal gait parameters, such as the walking trajectory and step length. To tackle this problem, the deep learning-based regression model is proposed to estimate the walking step length and pedestrian traveled distance. The conditional generative adversarial network (CGAN) is used to obtain the step length prediction model, which is trained with a small number of training data. The proposed smoothing algorithm is leveraged to compute the walking step length used as the reference label data in the training stage. Then the step length prediction model is used to estimate walking distance when the testing dataset is obtained.

Through the experiments, the proposed smoothing algorithm and deep learning-based step length prediction model can be applied for human movement measurement in real applications with considerable accuracy.

Thesis Supervisor: Young Soo Suh
Title: Professor

Acknowledgments

This work would not have been possible without the combined support of many people. First, I would like to express my sincere appreciation to my supervisor, Professor Young Soo Suh, for his patient guidance, kindly support, and devotion throughout my period of research time. His encouragement and enthusiastic guidance are a part of my success.

I would like to thank the committee members, Professor Hee Jun Kang, Professor Kang Hyun Jo, Professor Jaehyun Park, and Professor Hyung Keun Lee, for their precious time and helpful feedback to evaluate my work.

My acknowledgment is also sent to my best friends and lab mates. They let me know that a successful way of man is built with heartfelt friends. They are an indispensable thing in my life.

Finally, I would like to especially thank all my family members and my wife, for their unconditional and boundless love and support. Everything they do is for me. Thank God for bringing them into my life.

Contents

Abstract **i**

Acknowledgments **iii**

Contents **iv**

List of Figures **vi**

List of Tables **viii**

List of Algorithms **ix**

1 Introduction **1**

1.1 Motivation 1

1.2 Wearable Inertial Sensors and Applications 2

1.3 Wearable Sensors-based Deep Learning Approaches 4

1.4 Contributions 6

1.5 Thesis Organization 7

2 Human Motion Tracking using Inertial Navigation Algorithms **8**

2.1 Standard Inertial Navigation using an Indirect Kalman Filter 10

2.2 Optimization-based Smoothing for Inertial Navigation Algorithm 12

2.3 Chapter Summary 15

3 Walking Step Length Estimation using Smoothing Algorithm **16**

3.1 Related Work 17

3.2 Problem Formulation 18

3.3 Algorithm Description 20
3.3.1 Proposed Algorithm with a Straight-line Walking Constraint 20
3.3.2 Robust Smoother based on a Constant Speed Constraint 29
3.3.3 Walking Trajectory-based Step Length Estimation 34
3.4 Chapter Summary 36

4 Deep Learning-based Approach for Walking Distance Estimation 37
4.1 Related Work 38
4.2 System Overview 40
4.2.1 System Architecture 40
4.2.2 Sensor System 41
4.3 Algorithm Description 42
4.3.1 Data Preprocessing 42
4.3.2 Regression Approach based on Conditional Generative Adversarial Network (CGAN) 47
4.3.3 Walking Distance Estimation using CGAN-based Regression Model 51
4.4 Chapter Summary 52

5 Experimental and Results 54
5.1 Performance of Smoothing Algorithm-based Human Motion Estimation . . 54
5.1.1 Walking Trajectory Estimation Validation Experiment 54
5.1.2 Walking Step Length Estimation Validation 59
5.2 Performance of Deep Learning-based Walking Distance Estimation 61
5.2.1 Walking Distance Estimation Validation 61
5.2.2 Comparison With Other Conventional Methods 69
5.3 Chapter Summary 72

6 Conclusions and Future Work 74
6.1 Conclusions 74
6.2 Future Work 75

Bibliography 76

Author's Publications 85

List of Figures

1-1 Inertial sensor placements on the human body. 3

3-1 Straight-line walking environment (top view). 19

3-2 Zero velocity and moving intervals. 19

3-3 Line approximation of the walking direction. 23

3-4 Walking trajectory with known total distance constraint. 26

3-5 $s_{heading}$ parameter (left: $s_{heading}$ small, right: $s_{heading}$ large). 28

3-6 A velocity peak detection algorithm for 20 m walking distance at regular speed. Double stance events are identified based on the maximum peaks of the velocity norm. 30

3-7 Relationship of the maximum peaks between the acceleration norm and the difference velocity norm. Double stance events are identified based on the maximum peaks of the velocity norm. 31

3-8 An example of $|f(t_i, t_{i+2})|$ for 20 m walking at the mixed speed levels. . . . 32

3-9 The proposed smoothing algorithm-based walking step length estimation framework. 34

4-1 System architecture of the proposed walking distance estimation method. . 41

4-2 Overview system of the waist-mounted inertial sensor unit. 42

4-3 The filtered acceleration amplitude during normal speed is represented by the solid blue line, walking step time indexes are detected by two consecutive zero-crossing points of the filtered acceleration amplitude. 45

4-4 The acceleration amplitude data of the same subject with different walking speeds. 46

4-5 Framework architecture of the CGAN model. 47

4-6 Architecture and parameters of the generator model. 48
4-7 Architecture and parameters of the discriminator model. 49

5-1 Experiment setup for the walking trajectory estimation validation. 55
5-2 Comparison the estimated positions with the reference ground-truth. 56
5-3 Comparison the estimated velocities with the reference ground-truth. 57
5-4 Performance metrics (MRE, MAE, and RMSE) of the proposed method-based walking step length estimation. 59
5-5 Comparison of walking step length estimation errors using the smoothing algorithm without and with constant speed assumption at different speed levels. 60
5-6 Experimental setup: an inertial sensor module is mounted on the user's waist (left) and an indoor environment is designed to collect the training and testing data (right). 62
5-7 RMSE of the proposed method in different learning rates (top) and decay rates (bottom). The first and second rows of the x-axis data label represent the value of the learning rate (α) and decay rate (λ), respectively, for the generator (G) and discriminator (D). 64
5-8 RMSE of the proposed method in different batch sizes (top) and dimensions of the noise (bottom). Batch size is set to ratio $\{1/10, 1/5, 1/4, 1/3, 1/2, 1\}$ of the training dataset size, which is used with 300 step segments. 64
5-9 Walking distance estimation error using different amounts of training data used. The red squares and length of error bars indicate the mean and standard deviation of the estimated walking distance for all subjects, respectively. 65
5-10 Comparison of RMSE between different regression methods. 67
5-11 Comparison of RE using the universal and personalized models. 68
5-12 Walking along the path of a rectangular on the football field. 69

List of Tables

3.1 Parameters used in the proposed smoothing algorithm. 35

4.1 Xsens MTi-1 sensor specifications. 42

5.1 Lidar-lite v3 specifications. 55

5.2 RMSE of the estimated position and velocity using the proposed algorithm without and with constant speed assumption. 58

5.3 Twenty subjects information. 61

5.4 Descriptions of collected data from twenty subjects. 62

5.5 Hyperparameter tuning for generator and discriminator models. 63

5.6 Performance comparison of walking distance estimation using basic DNN and CGAN models. 67

5.7 Performance comparison of our method with other conventional methods. . 70

5.8 Comparison of computational complexity. 72

List of Algorithms

1 Velocity peak detection algorithm 30

2 Step detection method algorithm 43

3 CGAN-based regression training algorithm 50

Chapter 1

Introduction

1.1 Motivation

Human movement measurement is defined as any procedure for obtaining quantitative information involving the measurement of biomechanical variables, such as spatial-temporal gait, kinematic, and kinetic parameters [1, 2]. These biomechanical parameters are crucial for many applications in human motion analysis such as healthcare applications, human motion monitoring, and activity recognition [3]. Therefore, the measurement of human movement is an interesting and active topic in medicine and sports research [4], as it can help people to evaluate physical activities and sports in their daily lives. There are various methods to perform the human movement measurement. In particular, human movement measurement supported by highly accurate specialized systems, ambulatory systems or wearable sensors has a great potential in the healthcare areas, personal navigation systems, sports, and entertainment.

Specialized systems, including optical measurement systems such as (Vicon Motion Systems Ltd., Oxford, UK) or mat pressure measurement systems such as GAITRite (CIR Systems Inc., NJ, USA), have high accuracy when operating in controlled environments. These systems use several fixed cameras calibrated or pressure sensors and correlated in a specific place to obtain the gait parameters. Ambulatory systems, such as Kinect (Microsoft Corporation, WA, USA) using 3D cameras to capture human motion, are installed in a relatively uncontrolled environment and have limited observation. These systems have a restricted range of operation and are intended for mainly use in indoor environments. In contrast, human movement measurement using wearable sensors has the advantage of being portable and suitable for both indoor and outdoor environments. This presents that it is flexible to measure specific biomechanical variables or movement patterns when the sensor is attached to the human body.

The measurement of human movements in the daily environment provides valuable and complementary information to that obtained in laboratory tests. However, it is extremely difficult to go beyond the laboratory and obtain accurate measurements of human physical activity in the daily living environment [5]. As an alternative to laboratory tests, wearable inertial sensor systems are important to reach a larger population than current systems for movement measurement can do. Therefore, the human movement measurement using wearable inertial sensors has been developed during the last decade, and there are several methods to estimate the gait parameters, such as cadence [6, 7], step speed [8, 9], step length [10–13], and walking distance [14–16]. With the recent advances in the Internet of Things (IoT) and machine learning, wearable inertial sensors-based methods have been proposed by using machine learning and deep learning approaches to provide an accurate and reliable motion estimation for pedestrians [17–22]. Starting from these motivations, this dissertation studies the methods of using wearable inertial sensors and deep learning techniques to obtain biomechanical parameters of the pedestrians in the daily living environment. More precisely, the kinematics of human motion and spatial gait parameters, including step length and walking distance, are specifically considered in this dissertation.

1.2 Wearable Inertial Sensors and Applications

Inertial sensors are composed of accelerometers, gyroscopes, and sometimes magnetometers, which are one of the most important members of the Micro-Electro-Mechanical Systems (MEMS) family and are combined together as an inertial sensor, or inertial measurement unit (IMU). An inertial sensor measures and reports the specific 3D acceleration, angular rate or 3D rate of turn, and magnetic field of an object to which it is attached [23]. It is often integrated into inertial navigation systems (INS) which use the raw or calibrated IMU measurements to calculate the orientation, velocity (direction and speed of movement), and position of a moving object such as an unmanned aerial vehicle (UAV) or autonomous vehicle.

With advances in MEMS technology, an inertial sensor has a small size, lightweight, low power consumption, and low cost which is widely used in many application fields such as drones, mobile robots, and pedestrian navigation systems. Also, it is embedded into wearable smart devices including smartphones, smartwatches, smart bands, smart glasses, and even equipped smart shoes. This presents that it has great potential to conduct the guidance of behavior analysis and monitor human locomotion activities. Among them, the step length estimation (SLE) is an interesting topic that attracted the attention of many researchers since this information is utilized in various applications, such as pedestrian positioning, human gait analysis [24, 25], sports training, and healthcare applications [26, 27]. The usage of IMU can be applied for SLE in various environments without spatial limitations or requiring infrastructure. This is the biggest advantage of wearable inertial sensors as compared with the other tracking systems such as Vicon, Microsoft Kinect, or GAITRite systems.

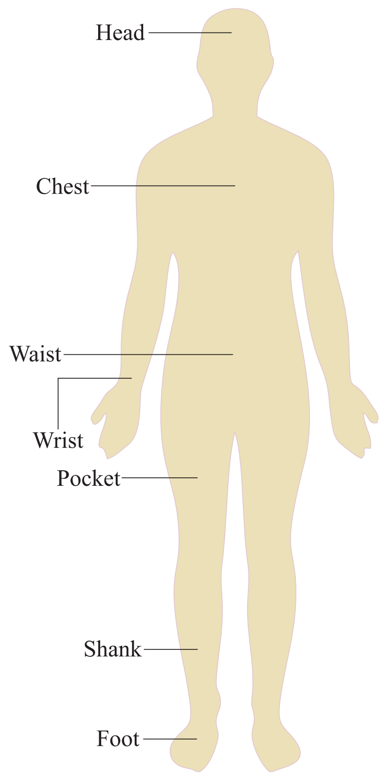


Figure 1-1: Inertial sensor placements on the human body.

SLE methods have been summarized in the review of Diez *et al.* [11] and Vezocnik *et al.* [12]. They analyze the advantages and disadvantages of SLE methods and evaluate each method. In the review of Diez *et al.* [11], they compare different SLE methods depending on the body parts, where a sensor is attached to any body location, such as foot, shank, pocket, wrist, waist, chest, or head (see Fig. 1-1). These SLE methods are divided into two approaches: one is the direct method, where the step length is computed by the integration of acceleration, and the other is the indirect method, where a SLE model is used to determine the step length. The indirect SLE methods have been proposed, mostly based on biomechanical models and statistical regression methods or deep learning methods. Each type of the SLE method has its strengths, weaknesses, and suitable application cases of each technique. The direct SLE methods do not need any models and do not require individual calibration stages. However, the noise and bias of inertial sensors always lead to the integration of error over time that affects the step length errors. Due to this reason, the method is feasible for human gait analysis through short walking distance or needs several assumptions and heuristics, such as zero velocity updating (ZUPT) for a foot-mounted IMU case [28–30], to reduce the errors for longer walking distance. In contrast, the indirect SLE methods are suitable and flexible for many application cases, such as pedestrian dead-reckoning (PDR) [31–33], with different sensor placements, but they require a calibration process or previous training stage with different users and speeds. The purpose of this dissertation is based on the combination of the direct and indirect methods to estimate accurate walking step length for the pedestrians without a laboratory calibration procedure.

1.3 Wearable Sensors-based Deep Learning Approaches

Deep learning is a subset of machine learning techniques, which is based on artificial neural networks (ANN) with feature learning. There are many deep learning architectures, such as deep neural networks (DNN), deep reinforcement learning (RL), recurrent neural networks (RNN), convolutional neural networks (CNN), and generative adversarial networks (GAN). The deep learning approaches have been applied in various field applications including computer vision, prediction, semantic analysis, natural language processing, information retrieval, and customer relationship management [34]. With the development of deep learning applications, the integration of deep learning and wearable inertial sensors is an interesting topic, which has attracted the attention of researchers in sports and medical applications, especially in human movement analysis including activity recognition and pedestrian inertial navigation. There are mainly three categories of deep learning approaches to obtain the biomechanical variables based on the availability of labels: supervised learning [17, 18, 20–22, 35], semi-supervised learning [36–38], and unsupervised learning [39, 40]. In supervised learning methods, the training data is labeled and the response variable is qualitative data for classification tasks or regression tasks. The CNN is applied to recognize different human activities in daily life, which uses a single accelerometer [17] or inertial sensors [18]. In [35], human activity recognition based on bidirectional long short-term memory (BLSTM) is proposed that uses times series collected from a waist-worn smartphone (with the accelerometer and gyroscope embedded on the phone). In [20], a CNN-based speed estimation method is suggested to obtain the speed from the IMU data, which the speed prediction is used in a functioning inertial navigation system (INS) to help constrain the movement. A deep learning-based inertial navigation system is proposed to learn and reconstruct pedestrian trajectories from raw IMU data [21], where the CNN architecture is used with 5 convolutional layers and one fully-connected layer. Another work in [22] presented a deep learning approach for step length estimation. The authors applied a family of deep learning-based approaches to regress the step length, which is used for the pedestrian indoor dead reckoning. Generally, the supervised methods achieve high performance; however, they need a large number of labeled training data which limits their applicability. In contrast, unsupervised learning methods do not require any labeled training data, which are based on variational autoencoders (VAE) and generative adversarial networks (GAN). They are commonly used in various applications including clustering, dimensionality reduction, and anomaly detection. In [39], the VAE is applied to classify human activities using a large number of datasets from wrist-mounted sensor data, where the ground truth of activities has not been used in the training process. Similarly, the GAN approach is proposed for human activity recognition in [40], which has used acceleration recordings from hand, chest, and ankle sensor placements. Semi-supervised learning methods combine supervised and unsupervised learning, which leverage a small amount of labeled data with a large amount of unlabeled data during training to tune the performance of unsupervised learning. In [36], the authors combine the supervised CNN and unsupervised CNN-encoder-decoder to perform semi-supervised learning for human activity recognition. The semi-supervised CNNs learn with limited labeled data and large-scale unlabeled data from the raw sensor data. Likewise, a deep LSTM is presented to recognize accurate human activities with smartphone inertial sensors [37]. On other hand, a deep learning-based pedestrian dead reckoning is proposed in [38]. The authors used the smartphone sensors and WiFi fusion for estimating the relative position of the user, where the CNN is applied to predict the local displacement and user activity, and a semi-supervised learning approach is based on VAE to build an accurate predictor for the WiFi-based positioning. Then, they apply a Kalman filter to correct for the drift of PDR using WiFi that provides a prediction of the user's absolute position each time a WiFi scan is received. However, the above deep learning approaches usually require a large number of training datasets (labeled datasets for supervised learning, unlabeled datasets for unsupervised learning, or both labeled and unlabeled for semi-supervised learning) to train the better fit model. Therefore, this dissertation aims to focus on a step length estimation model that uses supervised learning with a small number of training datasets to obtain accurately walking step length and walking distance.

1.4 Contributions

The dissertation presents the human movement measurement methods using wearable inertial sensors which are attached to the human body, such as the foot or waist. The sensor placement is closely related to the inertial sensor data that exhibits characteristic signal patterns when the pedestrian is moving. Based on the principle of the walking pattern for the waist-mounted inertial sensor cases, we propose the step length estimation methods for human movement measurement that can achieve outstanding performance go beyond the boundaries of the laboratory, and obtain accurate measurements of walking step length and also walking distance. In summary, the contributions of this research are as follows:

- We address the problem of the existing basic inertial navigation algorithms in human motion tracking and identify the motivation of this study.
- We propose a walking step length estimation using the constrained optimization-based smoothing algorithm, which is applicable for various users and walking speed levels. We impose a known distance straight-line walking trajectory constraint and a constant speed constraint to the optimization problem.
- We propose a step detection method that the magnitude of the acceleration data is applied to quickly and accurately detect the number of walking steps and step events when inertial sensors are attached to the waist.

- We propose a new regression approach based on deep learning in which the conditional generative adversarial network (CGAN) is used for estimating the walking step length and walking distance. We leverage the smoothing algorithm to estimate the walking step length used as the reference label data in the training stage.
- We conduct indoor and outdoor experiments to evaluate the performance of the proposed method. In addition to excellent estimation accuracy, the proposed method focuses on supervised learning with reduced training dataset size.

1.5 Thesis Organization

The rest of the dissertation is organized as follows. In Chapter 2, a human motion tracking using a standard smoothing-based inertial navigation algorithm is introduced. This motion tracking method is used to estimate the attitude, position, and velocity of the body system where an inertial sensor is attached. Chapter 3 proposes the smoothing algorithm based on constrained optimization for walking step length estimation. The algorithm uses an inertial sensor attached to the waist of the user and the walking distance is assumed to be known. In Chapter 4, a novel regression approach based on deep learning is proposed to estimate the walking distance using waist-mounted inertial sensors. Chapter 5 shows the experimental results to evaluate the performances of the proposed methods. Finally, Chapter 6 presents the conclusions and future directions for the dissertation.

Chapter 2

Human Motion Tracking using Inertial Navigation Algorithms

An inertial navigation algorithm plays an important role in pedestrian navigation systems. In this chapter, a standard smoothing-based inertial navigation algorithm is described to estimate the attitude (expressed using the quaternion), position, and velocity of the human body from the wearable inertial sensor data (including accelerometers and gyroscopes).

Two coordinate frames are used in this chapter: the body coordinate frame and the navigation coordinate frame. The three axes of the body coordinate frame coincide with the three axes of the IMU. The z axis of the navigation coordinate frame coincides with the local gravitational direction. The choice of the x and y axes of the navigation coordinate frame can be arbitrarily chosen. The notation $[p]_n$ ($[p]_b$) is used to denote that a vector p is represented in the navigation (body) coordinate frame.

The position is defined by $[r]_n \in R^3$, which is the origin of the body coordinate frame expressed in the navigation coordinate frame. Similarly, the velocity and the acceleration are denoted by $[v]_n \in R^3$ and $[a]_n \in R^3$, respectively. The attitude is represented using a quaternion $q \in R^4$, which represents the rotation relationship between the navigation coordinate frame and the body coordinate frame. The directional cosine matrix corresponding to quaternion q is denoted by $C(q) \in SO(3)$.

The accelerometer output $y_a \in R^3$ and the gyroscope output $y_g \in R^3$ are given by

$$\begin{aligned} y_a &= C(q)\tilde{g} + a_b + n_a, \\ y_g &= \omega + n_g, \end{aligned} \tag{2.1}$$

where $\omega \in R^3$ is the angular velocity, $a_b \in R^3$ is the external acceleration (acceleration related to the movement, excluding the gravitational acceleration) expressed in the body coordinate frame, and $n_a \in R^3$ and $n_g \in R^3$ are sensor noises. The vector $\tilde{g}$ is the local gravitational acceleration vector. It is assumed that $\tilde{g}$ is known, which can be computed using the formula in [41]. The sensor biases are assumed to be estimated separately using calibration algorithms [42, 43].

Let T denote the sampling period of a sensor. For a continuous time signal $y(t)$, the discrete value is denoted by $y_k = y(kT)$. The discrete sensor noise $n_{a,k}$ and $n_{g,k}$ are assumed to be white Gaussian sensor noises, whose covariances are given by

$$\begin{aligned} R_a &= \mathrm{E}\left\{n_{a,k} n'_{a,k}\right\} = r_a I_3 \in R^{3\times 3}, \\ R_g &= \mathrm{E}\left\{n_{g,k} n'_{g,k}\right\} = r_g I_3 \in R^{3\times 3}, \end{aligned} \tag{2.2}$$

where $r_a > 0$ and $r_g > 0$ are scalar constants.

Due to the high sensor noise level of low-cost inertial sensors and unknown biases, an inertial navigation algorithm without external reference measurement can estimate the human body's motion only for a few seconds. To overcome this problem, several assumptions and heuristics are used to reduce the accumulation of integration error. For example, the zero velocity updating (ZUPT) has been used for a foot-mounted inertial sensor case: the integration errors are reset when the foot is on the ground during walking. Because there are almost periodic zero velocity intervals as the foot touches the ground. When the sensor is attached to other body parts such as the wrist, pocket, and waist, there are no zero velocity intervals during continuous walking. Therefore, the assumptions and constraints need to apply to the inertial navigation algorithm for these sensor placements. In this chapter, our goal is to apply the standard smoothing-based inertial navigation algorithm for a foot-mounted inertial sensor case or other body positions-attached sensor cases if the walking period is only a few seconds (that is, a few walking steps).

To use the zero velocity updating in the inertial navigation algorithm, a simple zero-velocity detection algorithm in [44] is applied to determine the zero velocity intervals when a person is not moving (that is, a person stands still for a few seconds before and after walking) or the foot is on the ground during walking (only for a foot-mounted sensor case). Let Z_m be a set of all discrete-time indices belonging to the zero velocity intervals. The discrete-time index k is assumed to belong to a zero velocity interval if the following conditions are satisfied:

$$\begin{aligned} \|y_{g,i}\| &\leq B_g, & k - \tfrac{N_g}{2} \leq i \leq k + \tfrac{N_g}{2} \\ \|y_{a,i} - y_{a,i-1}\| &\leq B_a, & k - \tfrac{N_a}{2} \leq i \leq k + \tfrac{N_a}{2} \end{aligned} \tag{2.3}$$

where B_g and B_a are threshold parameters for zero velocity interval detection. N_g and N_a are even integer numbers. These parameters are designed based on the placement of an inertial sensor attached to the body.

2.1 Standard Inertial Navigation using an Indirect Kalman Filter

In this section, q, r, and v are estimated using a standard inertial navigation algorithm with an indirect Kalman filter.

The basic equations for inertial navigation are given as follows [45]:

$$\begin{aligned} \dot{q} &= \tfrac{1}{2}\Omega(\omega)q, \\ \dot{v}_n &= a_n = C'(q)a_b, \\ \dot{r}_n &= v_n, \end{aligned} \tag{2.4}$$

where symbol Ω is defined by

$$\Omega(\omega) = \begin{bmatrix} 0 & \omega' \\ \omega & -[\omega\times] \end{bmatrix},$$

and $[\omega\times] \in R^{3\times3}$ denotes the skew symmetric matrix of ω:

$$[\omega\times] = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}.$$

Let $\hat{q}_{KF,k}$, $\hat{r}_{KF,k}$, and $\hat{v}_{KF,k}$ be the estimated values of q, r and v using (2.4), where ω and a_b are replaced by y_g and $y_a - C(\hat{q})\tilde{g}$:

$$\begin{aligned} \dot{\hat{q}} &= \tfrac{1}{2}\Omega(y_g)\hat{q}, \\ \dot{\hat{v}} &= C'(\hat{q})y_a - \tilde{g}, \\ \dot{\hat{r}} &= \hat{v}. \end{aligned} \tag{2.5}$$

For later use, we define a function f_k, which represents a numerical integration of (2.5) from kT to $(k+1)T$:

$$\begin{bmatrix} \hat{q}_{KF,k+1} \\ \hat{r}_{KF,k+1} \\ \hat{v}_{KF,k+1} \end{bmatrix} = f_k \left(\begin{bmatrix} \hat{q}_{KF,k} \\ \hat{r}_{KF,k} \\ \hat{v}_{KF,k} \end{bmatrix}, \begin{bmatrix} n_{g,k} \\ n_{a,k} \end{bmatrix} \right). \tag{2.6}$$

$\bar{q}_{KF,k}$, $\bar{r}_{KF,k}$ and $\bar{v}_{KF,k}$ denote the estimation errors in $\hat{q}_{KF,k}$, $\hat{r}_{KF,k}$, and $\hat{v}_{KF,k}$, which are defined by

$$\begin{aligned} \bar{q}_{KF,k} &\triangleq [0_{3\times 1} \;\; I_3](\hat{q}^*_{KF,k} \otimes q_k), \\ \bar{r}_{KF,k} &\triangleq r_k - \hat{r}_{KF,k}, \\ \bar{v}_{KF,k} &\triangleq v_k - \hat{v}_{KF,k}, \end{aligned} \tag{2.7}$$

where $\otimes$ is the quaternion multiplication and q^* denotes the quaternion conjugate of a quaternion q.

The state of an indirect Kalman filter is defined by

$$X_{KF,k} \triangleq \begin{bmatrix} \bar{q}_{KF,k} \\ \bar{r}_{KF,k} \\ \bar{v}_{KF,k} \end{bmatrix} \in R^9. \tag{2.8}$$

The dynamic equation of $X_{KF,k}$ is given by (see [44] for details):

$$X_{KF,k+1} = \Phi_k X_{KF,k} + w_k, \tag{2.9}$$

where

$$\Phi_k = e^{A_k T} \approx I_9 + A_k T + \frac{1}{2} A_k^2 T^2, \quad A_k = \begin{bmatrix} [-y_{g,k} \times] & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_3 \\ -2C(\hat{q}_{KF,k})'[y_{a,k}\times] & 0_{3\times3} & 0_{3\times3} \end{bmatrix}. \quad (2.10)$$

The process noise w_k is the zero mean white Gaussian noise with

$$Q_k = \mathrm{E}\{w_k w_k'\} = \int_0^T e^{A_k r} \begin{bmatrix} 0.25R_g & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 10^{-6} I_3 & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & R_a \end{bmatrix} e^{A_k' r} dr.$$

During the zero velocity intervals, the zero velocity updating is applied to reduce the cumulative error and the following measurement equation is used:

$$0_{3\times1} - \hat{v}_{KF,k} = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} & I_3 \end{bmatrix} X_{KF,k} + n_v, \quad (2.11)$$

where n_v is a fictitious measurement noise representing a Gaussian white noise with the noise covariance R_v. The z axis value of $\hat{r}_{KF}$ is almost the same in the zero velocity intervals when a person is not moving or the foot is on the ground (only for a foot-mounted sensor case). Thus, the following z axis measurement equation is also used:

$$0 - \hat{r}_{KF,k,z} = \begin{bmatrix} 0_{1\times3} & 0 & 0 & 1 & 0_{1\times3} \end{bmatrix} X_{KF,k} + n_{r,z}, \quad (2.12)$$

where $n_{r,z}$ is the vertical position measurement noise whose noise covariance is $R_{r,z}$.

2.2 Optimization-based Smoothing for Inertial Navigation Algorithm

To further reduce the estimation errors of the Kalman filter, the smoothing algorithm in [44] is applied in which the estimation problem is expressed as a quadratic optimization. Let Z_m be the set of discrete-time indices when the measurement is available. The measurement equation is given by

$$z_k = H_k X_k + n_k, \quad k \in Z_m, \quad (2.13)$$

where H_k is the measurement matrix, and n_k is the measurement noise whose covariance matrix is R_k.

Let $\hat{q}_{SM,k}$, $\hat{r}_{SM,k}$, and $\hat{v}_{SM,k}$ be the smoothed estimations of the quaternion, position, and velocity, respectively. The estimation errors in $\hat{q}_{SM}$, $\hat{r}_{SM}$, and $\hat{v}_{SM}$ are estimated using the smoothing algorithm. The estimation errors are defined as follows:

$$X_{SM,k} = \begin{bmatrix} \bar{q}_{SM,k} \\ \bar{r}_{SM,k} \\ \bar{v}_{SM,k} \end{bmatrix} = \begin{bmatrix} [0_{3\times1} \;\; I_3](\hat{q}^*_{KF,k} \otimes q_k) \\ r_k - \hat{r}_{KF,k} \\ v_k - \hat{v}_{KF,k} \end{bmatrix} \in R^9, \tag{2.14}$$

where $\bar{q}_{SM,k}$, $\bar{r}_{SM,k}$, and $\bar{v}_{SM,k}$ denote the estimation errors of $q_{SM,k}$, $r_{SM,k}$, and $v_{SM,k}$, respectively.

Since $\bar{q}_{SM,k}$, $\bar{r}_{SM,k}$, and $\bar{v}_{SM,k}$ are time-dependent variables. The equation for these variables is given by

$$\zeta_k + X_{SM,k+1} = \Phi_k X_{SM,k} + w_k, \tag{2.15}$$

where Φ_k is defined in (2.10) and:

$$\zeta_k = \begin{bmatrix} [0_{3\times1} \;\; I_3](\hat{q}^*_{f_k} \otimes \hat{q}_{KF,k+1}) \\ \hat{r}_{KF,k+1} - \hat{r}_{f_k} \\ \hat{v}_{KF,k+1} - \hat{v}_{f_k} \end{bmatrix} \in R^9, \tag{2.16}$$

$$\begin{bmatrix} \hat{q}_{f_k} \\ \hat{r}_{f_k} \\ \hat{v}_{f_k} \end{bmatrix} = f_k \left(\begin{bmatrix} \hat{q}_{KF,k} \\ \hat{r}_{KF,k} \\ \hat{v}_{KF,k} \end{bmatrix}, \begin{bmatrix} n_{g,k} \\ n_{a,k} \end{bmatrix} \right). \tag{2.17}$$

To explain the derivation of (2.15), we compute $\zeta_k + X_{SM,k+1}$ from (2.14), (2.16), and (2.17):

$$\zeta_k + X_{SM,k+1} = \begin{bmatrix} [0_{3\times1} \;\; I_3](\hat{q}^*_{f_k} \otimes q_{k+1}) \\ r_{k+1} - \hat{r}_{f_k} \\ v_{k+1} - \hat{v}_{f_k} \end{bmatrix} \in R^9. \tag{2.18}$$

From (2.18), $\zeta_k + X_{SM,k+1}$ denotes the estimation errors of f_k in (2.17). Thus, (2.15) represents how the estimation error evolves after the integration (2.17).

The smoothing problem to estimate $X_{SM,k}$ can be formulated as a quadratic optimization problem using the method in [46]. Let an optimization variable $\tilde{X}$ is defined by

$$\tilde{X} = \begin{bmatrix} X_{SM,1} \\ X_{SM,2} \\ \vdots \\ X_{SM,N} \end{bmatrix} \in R^{9N}.$$

The smoothing problem can be formulated as the following optimization problem:

$$J(\tilde{X}) = \frac{1}{2}\sum_{k=1}^{N-1} w_k' Q_k^{-1} w_k + \frac{1}{2}\sum_{k\in Z_m} (z_k - H_k X_{SM,k})' R_k^{-1} (z_k - H_k X_{SM,k}) \\ + \frac{1}{2}(X_{SM,1} - X_{init})' P_{X_{init}}^{-1} (X_{SM,1} - X_{init}), \quad (2.19)$$

where X_{init} is the initial estimate of $X_{KF,1}$ and $P_{X_{init}} \in R^{9\times 9} > 0$ is the initial estimation error covariance.

By inserting (2.15) into (2.19) to remove the variable w_k, we have:

$$J(\tilde{X}) = \frac{1}{2}\sum_{k=1}^{N-1} (\zeta_k + X_{SM,k+1} - \Phi_k X_{SM,k})' Q_k^{-1} (\zeta_k + X_{SM,k+1} - \Phi_k X_{SM,k}) \\ + \frac{1}{2}\sum_{k\in Z_m} (z_k - H_k X_{SM,k})' R_k^{-1} (z_k - H_k X_{SM,k}) \\ + \frac{1}{2}(X_{SM,1} - X_{init})' P_{X_{init}}^{-1} (X_{SM,1} - X_{init}). \quad (2.20)$$

We can easily see that (2.20) is a quadratic function of $\tilde{X} \in R^{9\times N}$, the matrix form of the optimization is given by

$$\min_{\tilde{X}} J(\tilde{X}) = \frac{1}{2}\tilde{X}' M_1 \tilde{X} + M_2' \tilde{X} + M_3, \quad (2.21)$$

where $M_1 \in R^{9N\times 9N}, M_2 \in R^{9N\times 1}$, and $M_3 \in R$ can be computed from (2.20). Since (2.21) is a quadratic function of $\tilde{X}$, it can be computed efficiently using the quadratic optimization method in [47]. Minimizing (2.21) will provide a set of estimation errors. From these values, $\hat{q}_{SM}$, $\hat{r}_{SM}$, and $\hat{v}_{SM}$ are updated using the relationship (2.14).

2.3 Chapter Summary

An inertial navigation algorithm using the Kalman filter is a simple method but still provides a good quality of estimation for human motion tracking. Besides, the Kalman filter is a recursive algorithm that allows its real-time execution without requiring the history of observations or past estimates.

The smoothing-based inertial navigation algorithm uses a different approach for estimating the motion. In the optimization-based smoothing algorithm, all the data are used together to estimate each state at each sampling time. The advantage of this method provides more accurate motion estimation than other filter-based algorithms. However, the smoothing algorithm is more complicated than the Kalman filter and all the data are used, the method requires a high computation cost that is not feasible for real-time applications.

Depending on the purpose of motion estimation, an appropriate inertial navigation algorithm is chosen. Another limitation of the algorithm is the accumulation of integration errors when applying low-cost inertial sensors-based human motion tracking for a longer distance or period. To reduce these errors, several assumptions and heuristics, such as zero velocity updating (ZUPT), have been used for a foot-mounted sensor case. When the foot is on the ground during walking, the ZUPT algorithm is applied to reset the integration errors. Unlike the foot-mounted sensor case, there are no zero velocity intervals during continuous walking when the sensor is placed on the waist, wrist, or pocket area. To solve this issue, the smoothing algorithm based on constrained optimization will be presented in the next chapter.

Chapter 3

Walking Step Length Estimation using Smoothing Algorithm

Accurate walking step length estimation plays a vital role in some application fields such as gait analysis for patients [48] or elderly people [49], and human motion analysis [3]. These applications help predict the quality of life as well as in clinical diagnosis and medical treatment of diseases affecting the ability to walk. The walking speed and step length are the two most popular and mutual influencing parameters to analyze and characterize the human gait. The effects of walking speed on stability control of the step length are investigated [50]. Besides, the walking step length estimation is one of the important components of the pedestrian dead reckoning (PDR) algorithm [31–33], and the accuracy of PDR localization depends on the accurate step length estimation.

This chapter presents a new constrained optimization-based smoothing algorithm for walking step length estimation using waist-mounted inertial sensors, where the total walking distance is known. The walking trajectory is estimated by double integrating acceleration. Due to sensor noises, the walking step length estimation accuracy degrades as the walking distance becomes longer. To tackle this problem, we introduce a known distance straight-line walking trajectory constraint and a constant speed constraint to the smoothing algorithm. These constraints reduce the walking step estimation accuracy degradation even for long walking distance. The proposed smoothing algorithm can be applied to generate training data for walking step length estimation without requiring spatial infrastructure.

3.1 Related Work

Various methods for estimating walking step length can be used depending on the measurement devices. Among them, the visual motion tracking systems such as Vicon [51] are widely used, measure accurately but are costly. The 3D cameras such as Microsoft Kinect can be used as an alternative low-cost motion camera [52, 53]. Mat pressure measurement systems such as the GAITRite provide spatial parameters of gait through a walkway embedded with pressure sensors [54]. However, these systems are unsuitable for a long distance and outside of the laboratory environment due to their high implementation costs and limited walking ranges.

To estimate the walking step length using inertial sensors, there are two kinds of approaches [11]: direct approaches and indirect approaches. Direct approaches use the acceleration double integration, whereas indirect approaches apply the relationship between the step length and gait models or the prediction of statistical methods. In theory, the best method for estimating walking step length is based on the acceleration double integration since it does not need any models and does not require training data or individual calibration stages. However, low-cost inertial sensors always have noise and bias that lead to integration error over time. Due to this reason, several assumptions and heuristics are used to reduce error accumulation. For example, the zero velocity updates (ZUPT) have been used for a foot-mounted sensor case: the integration errors are reset when the foot is on the ground during walking [28–30]. When the sensor is attached to the other body parts (such as the wrist, pocket, and waist), there are no zero velocity intervals during continuous walking. Therefore, indirect approaches are used to replace direct methods for these cases.

Many indirect approaches have been proposed, mainly based on symmetrical gait models, such as empirical relationships [55, 56] and inverted pendulum models [57, 58], or statistical regression methods, such as step frequency-based models [59–62], acceleration-based models [63, 64], angle-based models [65, 66], and multiparameter models [33, 67]. The common point of these methods is a requirement for a calibration process. This process depends on a relationship between gait characteristics or sensor information and the step length as a training stage. Recently, machine learning and deep learning approaches are proposed to estimate the step length accurately [22, 68, 69]. The stacked autoencoders are proposed to estimate the step length based on the smartphone sensors, including accelerometers and gyroscopes [68]. In [69], the authors present a combined long short-term memory (LSTM) and denoising autoencoders to estimate a pedestrian's stride length. Another approach for step length estimation is proposed by applying a one-dimensional convolutional neural network in [22].

The training process always requires the ground truth length of each step. There are two ways to obtain the ground truth. The first way is based on the known total walking distance and then dividing this total distance by the total number of walking steps [59]. Although the number of walking steps is accurate, the walking step length accuracy is not guaranteed. The other way uses an additional device such as an optical motion capture system [65]. This method provides more accurate references. However, this method makes the training process more complicated, and expensive. The purpose of this chapter is to provide a simple method based on the double integration of the acceleration to obtain walking step length, which does not require spatial infrastructure.

In this chapter, we propose a new constrained optimization-based smoothing algorithm for the walking step length estimation using waist-mounted inertial sensors, where the total walking distance is known. The smoothing algorithm is formulated as a quadratic optimization problem, where the constraints are used to improve the walking trajectory accuracy. Two main constraints are used in the proposed algorithm: a known distance straight-line walking trajectory constraint and a constant speed constraint. In our work, we assume that a person walks along a straight path at a constant speed. A velocity peak detection and data rejection algorithm are proposed to determine the maximum peaks of the velocity and suitable walking data. These data are applied to the proposed smoothing algorithm for obtaining the pedestrian trajectory. Then, the step length estimation method is used to estimate walking step length when the step indexes are identified.

3.2 Problem Formulation

In this chapter, we consider a walking scenario, in which a person walks along a straight line (see Fig. 3-1). An inertial sensor unit (including accelerometers and gyroscopes) is attached to the user's waist. The length of the line is assumed to be known (L).

At the start line, a person stands still for a few seconds before starting to walk and then completes the desired straight path. Note that the person stands still once he arrives at

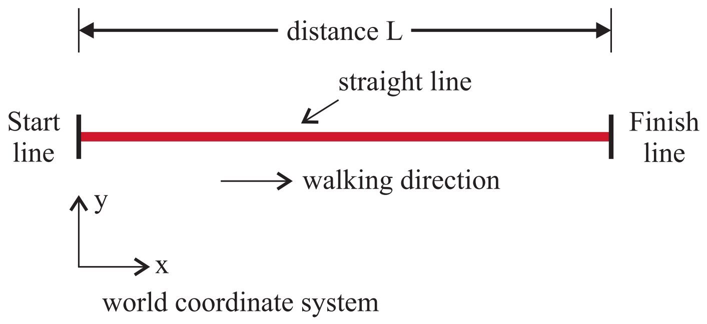


Figure 3-1: Straight-line walking environment (top view).

the finish line. Thus there are zero velocity intervals before and after walking, which makes it easier to separate the walking interval (see Fig. 3-2). The set of discrete-time indices belonging to zero velocity intervals is denoted by $S_{zero} = \{1, \cdots, N_1, N_4, \cdots, N\}$. The set of discrete-time indices belonging to the moving (walking) interval is denoted by $S_{moving} = \{N_2, \cdots, N_3\}$.

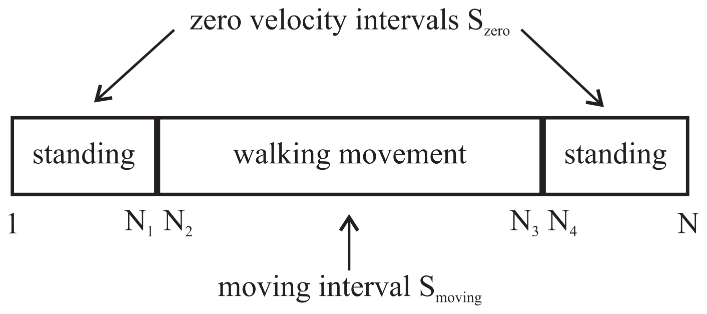

zero velocity intervals S_{zero}
standing
walking movement
standing
1
N_1 N_2
N_3 N_4
N
moving interval S_{moving}

Figure 3-2: Zero velocity and moving intervals.

The goal of this chapter is to estimate the walking step length by double integrating acceleration. Due to sensor noises in low-cost inertial sensors, it is not possible to estimate the walking step length accurately even for a short distance. We propose a walking step length estimation algorithm using the fact that a person is walking in a straight line with a known distance.

Two coordinate systems are used: the body coordinate system and the world coordinate system. The three axes of the body coordinate system coincide with the three axes of the inertial sensor unit. The world coordinate system is a local geographic frame, where the z-axis coincides with the local gravitation direction (up direction). The x-axis of the world coordinate system coincides with the line in Fig. 3-1. In the world coordinate system, the gravitation vector is denoted by $\tilde{g} = \begin{bmatrix} 0 & 0 & g \end{bmatrix}' \in R^3$, where g is the magnitude of gravitation. A rotation matrix C_w^b is used to represent the rotation relationship between the body coordinate system and the world coordinate system.

The sensor outputs (accelerometers $y_a \in R^3$ and gyroscopes $y_g \in R^3$) are modeled as follows:

$$\begin{aligned} y_a &= a_b + C_w^b(t)\tilde{g} + v_a(t), \\ y_g &= \omega(t) + v_g(t), \end{aligned} \tag{3.1}$$

where $a_b \in R^3$ is the external acceleration expressed in the body coordinate frame, $\omega \in R^3$ is the body angular velocity, and $v_a \in R^3$ and $v_g \in R^3$ are white Gaussian sensor noises. The noise covariances are given by

$$\begin{aligned} \mathrm{E}\{v_a(t)v_a(s)'\} &= r_a\delta(t-s)I_3, \\ \mathrm{E}\{v_g(t)v_g(s)'\} &= r_g\delta(t-s)I_3. \end{aligned} \tag{3.2}$$

where $r_a > 0$ and $r_g > 0$ are scalar constants of the sensor noise covariances, and δ is the Dirac's impulse function. Sensor biases are assumed to be removed using a separate calibration algorithm [42, 43].

3.3 Algorithm Description

In this section, the proposed algorithm with a straight-line walking constraint is presented. A robust smoother based on a constant speed constraint is then explained, and finally, the step length estimation method is performed to compute the walking step length.

3.3.1 Proposed Algorithm with a Straight-line Walking Constraint

In this subsection, the proposed algorithm consists of three steps. The first step is to detect zero velocity intervals (standing still events before and after walking). The second step is to generate an initial walking trajectory based on a smoothing algorithm with zero velocity constraints. Then, the third step is to apply the constraint of known walking distance for a smoothing algorithm to reduce further the estimation errors of the previous method.

Zero velocity detection

When a person stands still for a few seconds before and after walking, the zero velocity intervals are easily detected. There are many zero velocity interval detection algorithms [70].

Among them, the zero velocity detection algorithm is applied based on the accelerometer magnitude and gyroscope norm. The discrete index k is determined to belong to the zero velocity interval if the following condition is satisfied:

$$\alpha_a(\|y_{a,i}\| - g)^2 + \alpha_g \|y_{g,i}\|^2 \leq \alpha_{zero}, \quad k - M_{zero} \leq i \leq k + M_{zero}, \tag{3.3}$$

where α_a and α_g are the weighting factors of the acceleration and gyroscope norms, respectively. α_{zero} is a threshold parameter and M_{zero} is a specified window length for zero velocity interval detection.

Smoothing Algorithm with Zero Velocity Constraints

In the second step, a smoother with zero velocity constraint is used to generate a walking trajectory. Recall that the x-axis of the world coordinate system coincides with the walking direction (the straight line in Fig. 3-1). Since the initial walking direction cannot be known without a heading sensor, an initial walking trajectory is generated with an arbitrary initial heading. Any smoothing algorithms can be used in this step to generate an initial trajectory [44, 71, 72], where the estimated position, velocity, and quaternion are denoted by $\hat{r}_{initial,k} \in R^3$, $\hat{v}_{initial,k} \in R^3$ and $\hat{q}_{initial,k} \in R^4$, respectively.

From the fact that the origin of the world coordinate system coincides with the starting point of the person, the initial condition constraint at $k = 1$ is given by

$$\begin{bmatrix} \hat{q}_{initial,1} \\ \hat{r}_{initial,1} \\ \hat{v}_{initial,1} \end{bmatrix} = \begin{bmatrix} \hat{q}_{triad} \\ 0_{3\times 1} \\ 0_{3\times 1} \end{bmatrix}, \tag{3.4}$$

where the initial attitude $\hat{q}_{triad}$ is computed from the accelerometer data using the TRIAD algorithm [73] for the following two vector equations:

$$y_{a,1} = C(\hat{q}_{triad}) \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}, \quad \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = C(\hat{q}_{triad}) \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}. \tag{3.5}$$

Note that the heading is arbitrarily chosen in the second equation of (3.5): the world coordinate x-axis is chosen in the direction of the body coordinate x-axis. The heading is later adjusted so that the world coordinate x direction coincides with the walking direction.

The initial estimation error covariance P_1 is given by

$$P_1 = \begin{bmatrix} P_{attitude} & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & s_{init,r} I_3 & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} & s_{init,v} I_3 \end{bmatrix},$$

where the initial attitude error covariance $P_{attitude} \in R^{3\times 3}$ is computed by the method in [74, Chapter 5.5] for two equations of (3.5). We note that the multiplicative quaternion error model is used [75], where the quaternion error is represented by a three-dimensional vector. The covariances $s_{init,r} I_3$ and $s_{init,v} I_3$ can be considered as design parameters, whose $s_{init,r}$ and $s_{init,v}$ are the initial noise values of the position and velocity, respectively (see Table 3.1).

The inverse of initial attitude error covariance $P^{-1}_{attitude}$ is used in the smoothing algorithm and computed by the TRIAD information matrix as follows [74, Chapter 5.5]:

$$P^{-1}_{attitude} = \frac{1}{r_a}(I_3 - b_1 b_1') + \frac{1}{s_{heading}}(b_2 \times b_\times)(b_2 \times b_\times)',$$

$$b_1 = C(\hat{q}_{triad}) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \; b_2 = C(\hat{q}_{triad}) \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix},$$

$$b_\times = \frac{b_1 \times b_2}{\|b_1 \times b_2\|}.$$

Normally $s_{heading} > 0$ denotes the heading noise covariance, which represents how accurate the heading sensor. In this initial estimation, $s_{heading}$ does not play such a role; $s_{heading}$ is a design parameter in Section 3.3.1.

The constraint for the first-zero velocity interval at $k = \{2, \cdots, N_1\} \in S_{zero}$ is defined by

$$v_k = 0_{3\times 1} + n_{init2,k}, \tag{3.6}$$

$$\mathrm{E}\{n_{init2,k} n'_{init2,k}\} = s_{init2,v} I_3,$$

where $s_{init2,v}$ is the velocity measurement noise at the first zero-velocity interval.

The last zero-velocity interval ($N_4 \leq k \leq N-1$) is constrained as follows:

$$\begin{bmatrix} r_{k,z} \\ v_k \end{bmatrix} = \begin{bmatrix} 0 \\ 0_{3\times1} \end{bmatrix} + n_{final,k}, \tag{3.7}$$

$$\mathrm{E}\{n_{final,k} n'_{final,k}\} = \begin{bmatrix} s_{final,r_z} & 0_{1\times3} \\ 0_{3\times1} & s_{final,v} I_3 \end{bmatrix},$$

where s_{final,r_z} and $s_{final,v}$ are the measurement noises of the vertical position and velocity at the final zero-velocity interval, respectively.

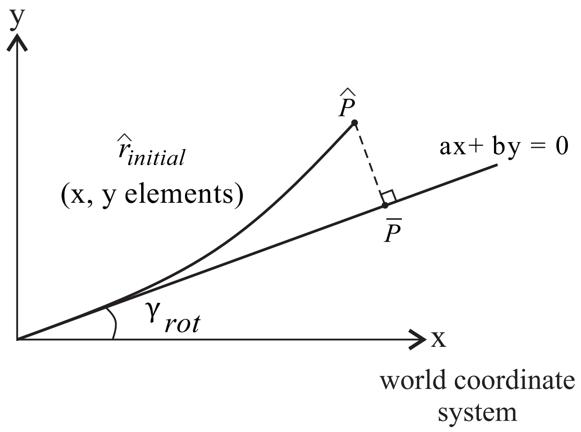


Figure 3-3: Line approximation of the walking direction.

Assume that the initial walking trajectory is obtained by using the smoothing algorithm with constraints (3.4), (3.6), and (3.7). Then the initial yaw angle is adjusted so that the world coordinate x-axis coincides with the walking direction. The initial trajectory is approximated by a line (see Fig. 3-3), where the following line equation is used:

$$ax + by = 0 \tag{3.8}$$

subject to

$$a^2 + b^2 = 1. \tag{3.9}$$

Since the position accuracy degrades as the time index k increases, the initial $M_{initial}$ estimated positions are used to find the line parameters a and b as the following:

$$\min_{a,b} \sum_{i=2}^{M_{initial}} \left\| w_i (a\hat{r}_{initial,x,i} + b\hat{r}_{initial,y,i}) \right\|^2 \tag{3.10}$$

subject to the constraint (3.9). The weighting factor w_i is simply chosen $w_i = M_{initial} - i$ so that more weights are given on smaller k reflecting the fact that the position accuracy degrades as k increases.

Let $D \in R^{(M_{initial}-1)\times 2}$ be defined by

$$D = \begin{bmatrix} w_2\hat{r}_{initial,x,2} & w_2\hat{r}_{initial,y,2} \\ w_3\hat{r}_{initial,x,3} & w_2\hat{r}_{initial,y,3} \\ \vdots & \vdots \\ w_{M_{initial}}\hat{r}_{initial,x,M_{initial}} & w_2\hat{r}_{initial,y,M_{initial}} \end{bmatrix},$$

then (3.10) can be expressed as follows:

$$\min_{a,b} ||D \begin{bmatrix} a \\ b \end{bmatrix} ||^2. \tag{3.11}$$

It is known that the minimizing solution to (3.11) is given by

$$\begin{bmatrix} a \\ b \end{bmatrix} = \text{the last column vector of } V, \tag{3.12}$$

where $D = U\Sigma V'$ is the singular value decomposition of D.

Since the line equation provides two directions (both ends of line from the origin), the walking direction is chosen from $\hat{P} \in R^2$, which is x and y elements of $\hat{r}_{initial,M_{initial}}$. Let $\bar{P} \in R^2$ be the orthogonal projection point to the line from $\hat{P}$ (see Fig. 3-3). Because the orthogonal direction of the line is $\begin{bmatrix} a & b \end{bmatrix}'$, $\bar{P}$ is given by

$$\bar{P} = \hat{P} + c \begin{bmatrix} a \\ b \end{bmatrix}. \tag{3.13}$$

The constant c can be determined from the fact $\bar{P}$ should be on the line:

$$\begin{bmatrix} a & b \end{bmatrix} \bar{P} = 0. \tag{3.14}$$

From (3.13) and (3.14), c is given by

$$c = -\begin{bmatrix} a & b \end{bmatrix} \hat{P}. \tag{3.15}$$

Let γ_{rot} be defined by

$$\gamma_{rot} = \text{atan2}\left(\bar{P}_x, \bar{P}_y\right). \tag{3.16}$$

Now the line becomes the x-axis of the new world coordinate system: if we rotate the new world coordinate system by γ_{rot} along the z-axis, we obtain the old world coordinate system. Let $q_{\gamma_{rot}}$ and $C_{\gamma_{rot}}$ be defined by

$$q_{\gamma_{rot}} = \begin{bmatrix} \cos\frac{\gamma_{rot}}{2} \\ 0 \\ 0 \\ -\sin\frac{\gamma_{rot}}{2} \end{bmatrix}, \quad C_{\gamma_{rot}} = C(q_{\gamma_{rot}}) = \begin{bmatrix} \cos\gamma_{rot} & -\sin\gamma_{rot} & 0 \\ \sin\gamma_{rot} & \cos\gamma_{rot} & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Let $\hat{r}_{rotated}, \hat{v}_{rotated}$ and $\hat{q}_{rotated}$ denote the position, velocity and quaternion in the new world coordinate. They can be computed as follows:

$$\begin{aligned} \hat{r}_{rotated} &= C_{\gamma_{rot}} \hat{r}_{initial}, \\ \hat{v}_{rotated} &= C_{\gamma_{rot}} \hat{v}_{initial}, \\ \hat{q}_{rotated} &= q_{\gamma_{rot}} \otimes \hat{q}_{initial}, \end{aligned} \tag{3.17}$$

where $\otimes$ denotes the quaternion multiplication.

Smoothing Algorithm with Known Distance Straight-line Walking Trajectory Constraint

After the rotation (3.17), the walking direction approximately coincides with the world x-axis (see Fig. 3-3). The next step is to apply the known distance straight-line constraint so that the walking trajectory becomes like the right-hand side of Fig. 3-4.

Let $\hat{r}_{SM,k} \in R^3, \hat{v}_{SM,k} \in R^3$ and $\hat{q}_{SM,k} \in R^4$ denote the known distance straight-line constraint imposed estimations of the position, velocity, and quaternion, respectively. These values are computed from $\hat{r}_{rotated}, \hat{v}_{rotated}$ and $\hat{q}_{rotated}$ by imposing the known distance straight-line constraint. The estimation errors in $\hat{q}_{SM}, \hat{r}_{SM}$ and $\hat{v}_{SM}$ are estimated using

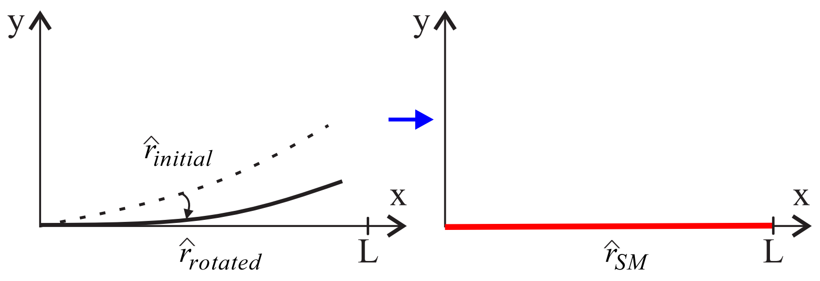


Figure 3-4: Walking trajectory with known total distance constraint.

the proposed smoothing algorithm. The estimation errors are defined as follows:

$$X_k = \begin{bmatrix} \bar{q}_k \\ \bar{r}_k \\ \bar{v}_k \end{bmatrix} = \begin{bmatrix} [0_{3\times1} \;\; I_3](\hat{q}^*_{rotated,k} \otimes q_k) \\ r_k - \hat{r}_{rotated,k} \\ v_k - \hat{v}_{rotated,k} \end{bmatrix} \in R^9, \tag{3.18}$$

where $\bar{q}_k \in R^3, \bar{r}_k \in R^3$ and $\bar{v}_k \in R^3$ denote the estimation errors of $q_{SM,k}, r_{SM,k}$ and $v_{SM,k}$, respectively. In (3.18), q^* denotes the quaternion conjugate of a quaternion q.

Let the function f_k be the numerical integration of the quaternion, position, and velocity equations for inertial navigation from kT to $(k+1)T$, where T is the sensor sampling period. The function f_k is defined as follows [44]:

$$\begin{bmatrix} \hat{q}_{f_k} \\ \hat{r}_{f_k} \\ \hat{v}_{f_k} \end{bmatrix} = f_k\left(\begin{bmatrix} \hat{q}_{rotated,k} \\ \hat{r}_{rotated,k} \\ \hat{v}_{rotated,k} \end{bmatrix}, \begin{bmatrix} v_{g,k} \\ v_{a,k} \end{bmatrix} \right). \tag{3.19}$$

Since $\bar{q}_k, \bar{r}_k$ and $\bar{v}_k$ are time-dependent variables. The equation for these variables is given by

$$\zeta_k + X_{k+1} = \Phi_k X_k + w_k, \tag{3.20}$$

where ζ_k is defined by

$$\zeta_k = \begin{bmatrix} [0_{3\times1} \;\; I_3](\hat{q}^*_{f_k} \otimes \hat{q}_{rotated,k+1}) \\ \hat{r}_{rotated,k+1} - \hat{r}_{f_k} \\ \hat{v}_{rotated,k+1} - \hat{v}_{f_k} \end{bmatrix} \in R^9, \tag{3.21}$$

and Φ_k are given by [44]:

$$\Phi_k = e^{A_k T} \approx I_9 + A_k T + \tfrac{1}{2!}A_k^2 T^2, \quad A_k = \begin{bmatrix} [-y_{g,k}\times] & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_3 \\ -2C(\hat{q}_{rotated,k})'[y_{a,k}\times] & 0_{3\times3} & 0_{3\times3} \end{bmatrix}.$$

The process noise w_k is the zero mean white Gaussian noise with

$$Q_k = \mathrm{E}\{w_k w_k'\} = \int_0^T e^{A_k r} \begin{bmatrix} 0.25 r_g I_3 & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 10^{-6} I_3 & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & r_a I_3 \end{bmatrix} e^{A_k' r} dr.$$

To explain the derivation of (3.20), we compute $\zeta_k + X_{k+1}$ from (3.18), (3.19) and (3.21):

$$\zeta_k + X_{k+1} = \begin{bmatrix} [0_{3\times1} \;\; I_3](\hat{q}_{f_k}^* \otimes q_{k+1}) \\ r_{k+1} - \hat{r}_{f_k} \\ v_{k+1} - \hat{v}_{f_k} \end{bmatrix} \in R^9. \tag{3.22}$$

From (3.22), $\zeta_k + X_{k+1}$ denotes the estimation errors of the function f_k in (3.19). Thus, (3.20) represents how the estimation error evolves after the integration (3.19).

The smoothing problem to estimate X_k can be formulated as a quadratic optimization problem using the method in [44, 46]. Let the optimization variable $\tilde{X}$ be defined by

$$\tilde{X} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_N \end{bmatrix} \in R^{9\times N}. \tag{3.23}$$

The smoothing problem is formulated as the following optimization problem (see [44, 46] for details):

$$J(\tilde{X}) = \frac{1}{2}\sum_{k=1}^{N-1} (\zeta_k + X_{k+1} - \Phi_k X_k)' Q_k^{-1} (\zeta_k + X_{k+1} - \Phi_k X_k) + (\text{constraint terms}). \tag{3.24}$$

The main advantage of formulating the smoothing problem as an optimization problem is that constraints are easily included in the optimization problem. See [44] as to how the constraint terms (such as (3.4), (3.6) and (3.7)) are formulated in (3.24). The following constraints are used in the smoothing algorithm. The first and second constraints are applied in the same way as the conditions of (3.4) and (3.6). Except that the initial quaternion used in this algorithm is $\hat{q}_{SM,1} = \hat{q}_{rotated,1}$. Also, the inverse of initial attitude error covariance $P^{-1}_{attitude}$ is computed by the information matrix with $\hat{q}_{rotated,1}$ and $s_{heading}$, where $s_{heading}$ is a parameter determining how the whole trajectory is rotated during the optimization. If $s_{heading}$ is small, the straight-line constraint is satisfied without rotating (see the left of Fig. 3-5). If $s_{heading}$ is large, the walking trajectory ($\hat{r}_{rotated}$) is rotated during the optimization (see the right of Fig. 3-5).

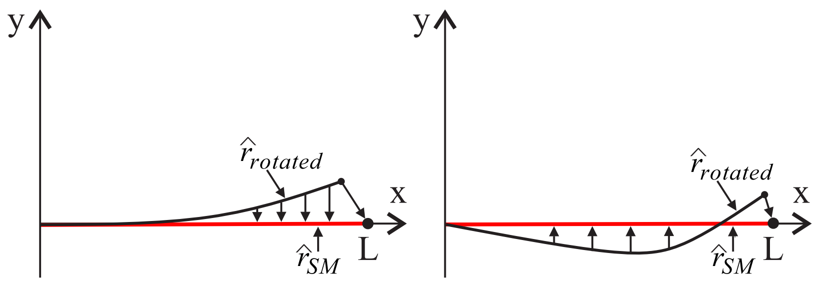


Figure 3-5: $s_{heading}$ parameter (left: $s_{heading}$ small, right: $s_{heading}$ large).

Since the walking trajectory is assumed as the straight line and in the direction of the x-axis. Thus the y-axis position is almost zero during the moving interval. The third constraint for this interval ($k \in S_{moving}$) is used as the following constraint:

$$r_{k,y} = 0 + n_{moving,k}, \tag{3.25}$$

$$\mathrm{E}\{n_{moving,k} n'_{moving,k}\} = s_{moving,r_y},$$

where s_{moving,r_y} is an optimization parameter. If s_{moving,r_y} is very small, the estimated walking trajectory becomes an exact line, while in practice there is a small y-axis fluctuation during walking. If s_{moving,r_y} is very large, there could be a large fluctuation in the estimated trajectory, which is contradictory to the assumption that a person is walking along a line.

We note that a person walks along a straight path with a known distance. Therefore, we use the known walking distance (L) for the final zero-velocity interval. The last condition is imposed as an additional constraint of (3.7) and defined as follows:

$$\begin{bmatrix} r_{k,x} \\ r_{k,y} \end{bmatrix} = \begin{bmatrix} L \\ 0 \end{bmatrix} + n_{final2,k}, \tag{3.26}$$

$$\mathrm{E}\{n_{final2,k} n'_{final2,k}\} = \begin{bmatrix} s_{final,r_x} & 0 \\ 0 & s_{final,r_y} \end{bmatrix},$$

where s_{final,r_x} and s_{final,r_y} is the x-axis and y-axis position measurement noises at the final zero-velocity interval, respectively.

After computing the optimization problem of $\tilde{X}$ using the smoothing algorithm combined with the constraint of a known distance straight-line walking trajectory, the estimation values of $\hat{q}_{SM}, \hat{r}_{SM}$, and $\hat{v}_{SM}$ are updated.

3.3.2 Robust Smoother based on a Constant Speed Constraint

For a short walking distance, the optimization algorithm in Section 3.3.1 provides a relatively accurate estimation. For a long walking distance, the estimation accuracy rapidly degrades due to sensor noises. To reduce this performance degradation, a new constraint is introduced in this subsection. During walking, a person is assumed to walk at a constant speed. One difficulty in using this assumption is that a human walking speed is like a sinusoidal form (see Fig. 3-6) even during constant walking speed. Thus, we cannot simply use the constraint that v_{r_x} = constant. In this subsection, we use the constraint that the walking speed is constant at the double stance phase of each gait cycle.

Velocity Peak Detection

Based on the principles of a walking model for waist-mounted inertial sensors [58, 76], the double stance phase occurs when the waist position reaches the lowest point. In a gait cycle, the bottom position of the waist corresponds to the highest peak of the velocity norm, which corresponds to a double stance phase. In this chapter, the velocity norm is computed from three axes of the estimated velocity ($\hat{v}_{SM}$), which is obtained by using the smoothing algorithm in Section 3.3.1. The maximum peak at the discrete-time k is determined using Algorithm 1.

An example of a velocity peak detection algorithm for walking along a straight-line

Algorithm 1 Velocity peak detection algorithm

```
Input: Velocity norm $V_{norm}$, threshold value $\delta_{th}$, window size parameters $M_{size}$ and $W_{size}$.
Output: Maximum peak index $p$.
 1: Compute the length $N$ from the length of $V_{norm}$.
 2: for $m = M_{size} + 1$ to $N - M_{size}$ do
 3:     Compute the average filter on the velocity norm
            $$\bar{V}_{norm,m} = \frac{1}{2M_{size}+1} \sum_{k=m-M_{size}}^{m+M_{size}} V_{norm,k}.$$
 4: end for
 5: Compute $\bar{y} = V_{norm} - \bar{V}_{norm}$.
 6: for discrete-time $k$ from $k = W_{size} + 1$ to $N - W_{size}$ do
 7:     if $(\bar{y}_k > \delta_{th})$ and $\bar{y}_k \geq \max(\bar{y}_{k-W_{size}} : \bar{y}_{k-1})$ and $\bar{y}_k \geq \max(\bar{y}_{k+1} : \bar{y}_{k+W_{size}})$ then
 8:         $p = p \cup \{k\}$.
 9:     end if
10: end for
11: return $p$
```

corridor (20 meters) at regular speed is shown in Fig. 3-6. The figure above shows the standard velocity norm and the filtered average velocity norm. The difference between these velocities is described with the solid blue line in the figure below, while the red circles represent the maximum peaks. Furthermore, Fig. 3-7 illustrates that the highest peaks of the difference velocity norm correspond to the maximum peaks of the acceleration norm. These peaks represent the double stance phase during walking.

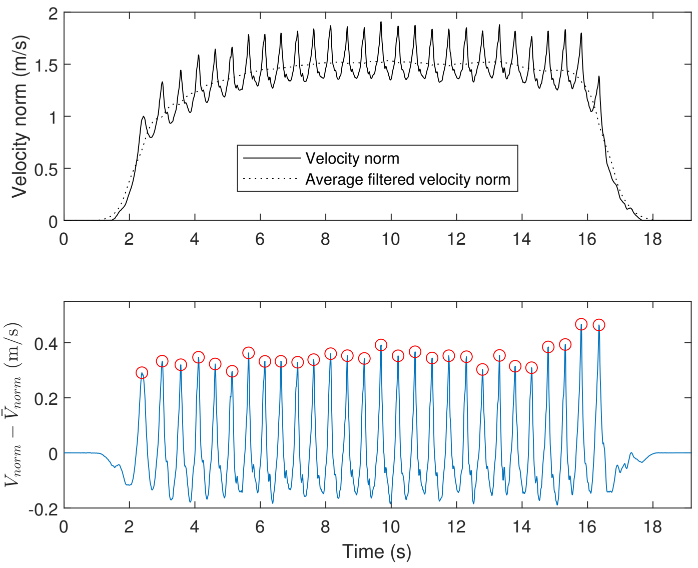


Figure 3-6: A velocity peak detection algorithm for 20 m walking distance at regular speed. Double stance events are identified based on the maximum peaks of the velocity norm.

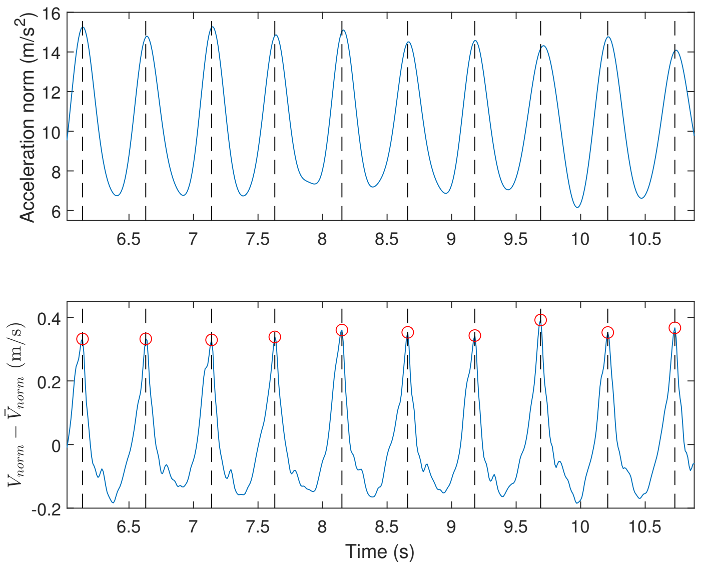


Figure 3-7: Relationship of the maximum peaks between the acceleration norm and the difference velocity norm. Double stance events are identified based on the maximum peaks of the velocity norm.

Data Rejection Algorithm

Before applying a constant speed constraint, we need some guarantee that a person is walking at a constant speed. For example, a person may not walk at a constant speed even if he is instructed to walk at a constant speed.

To reject the walking data with varying walking speeds, we propose a simple criterion to determine whether a person is walking at a constant speed. The velocity equation for inertial navigation is given as follows (sensor noises are omitted):

$$v_{t_{i+1}} \approx v_{t_i} + \int_{t_i}^{t_{i+1}} (C_b^w(t) y_a - \tilde{g}) dt, \tag{3.27}$$

where t_i $(1 \leq i \leq N_p)$ denotes the time index of the maximum peak and N_p is the number of detected peaks.

Since $\tilde{g}$ is a constant, the following equation is obtained:

$$v_{t_{i+1}} - v_{t_i} \approx \Delta v_{t_{i+1}}^{t_i} - \tilde{g}(t_{i+1} - t_i), \tag{3.28}$$

where

$$\Delta v_{t_{i+1}}^{t_i} = \int_{t_i}^{t_{i+1}} C_b^w(t) y_a dt.$$

The rotation matrix C_b^w is computed from the quaternion $\hat{q}_{SM}$, which is estimated using the smoothing algorithm in 3.3.1. Since the moving speed between the left and right steps (that is, two consecutive maximum peaks of the velocity norm) may not be the same, the velocity difference at t_i and t_{i+2} is used to detect whether the walking speed is constant or not. From (3.28), $\Delta v_{t_{i+2}}^{t_i} - \tilde{g}(t_{i+2} - t_i) \approx 0$ implies $v_{t_{i+2}} - v_{t_i} \approx 0$. That is, $\Delta v_{t_{i+2}}^{t_i} - \tilde{g}(t_{i+2} - t_i)$ value can be used to determine whether a person is walking at a constant speed. However, the computation of $\Delta v_{t_{i+2}}^{t_i}$ requires $C_b^w(t)$ whose accurate computation may not be possible. Thus, the following function is used instead:

$$f(t_i, t_{i+2}) = ||\Delta v_{t_{i+2}}^{t_i}|| - g(t_{i+2} - t_i), \tag{3.29}$$

where the exact initial attitude is not required for the computation of $||\Delta v_{t_{i+2}}^{t_i}||$.

In this chapter, a constant walking speed assumption is invalidated if

$$|f(t_i, t_{i+2})| > \varepsilon_{th}, \tag{3.30}$$

where ε_{th} is a small threshold value.

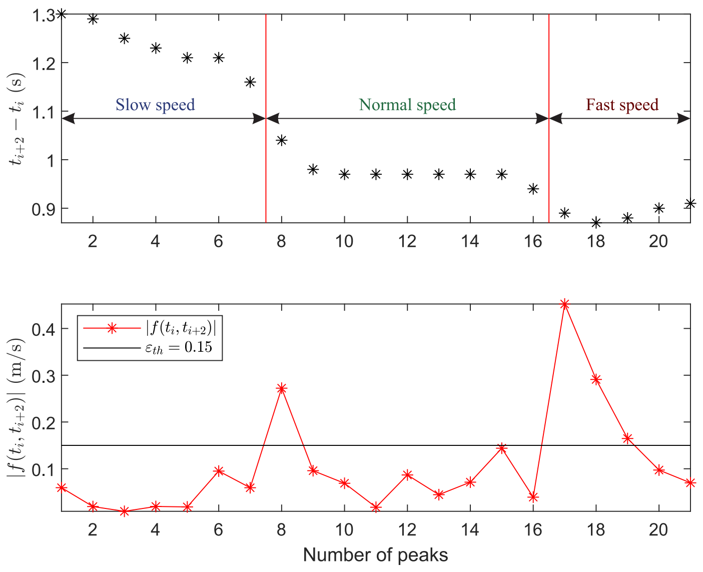


Figure 3-8: An example of $|f(t_i, t_{i+2})|$ for 20 m walking at the mixed speed levels.

An example of $|f(t_i, t_{i+2})|$ when a person walks along a 20 m straight line at the mixed walking speeds as shown in Fig. 3-8. We can see that $|f(t_i, t_{i+2})|$ is larger than a threshold value if the walking speed changes. Thus, the data rejection algorithm is used to obtain the suitable walking data for the proposed smoothing algorithm.

Constant Speed Constraint

Once we have the walking data at a constant speed, the next step is to apply the constant speed constraint to optimize the smoothing problem. We assume that a person is advised to walk at three different self-paced constant walking speeds: slow, normal, and fast.

Using the peak detection method in Section 3.3.2, we obtain the maximum peaks of the velocity norm. Since the walking speed is not constant during initial accelerating and final decelerating intervals, two maximum peaks of these steps are not used for the constant speed constraint. Let p_i $(2 \leq i \leq N_p - 1)$ be indices of maximum peaks. When the maximum peaks p_i are detected, the additional constraints are imposed as follows:

$$v_{p_i,x} = v_{moving} + n_{peak,p_i}, \tag{3.31}$$

$$\mathrm{E}\{n_{peak,p_i} n'_{peak,p_i}\} = s_{peak},$$

where v_{moving} is an average speed of all maximum peaks corresponding to x-axis velocity, which is estimated by the constrained optimization based on the constant speed assumption. s_{peak} is a design parameter for the proposed algorithm. If the s_{peak} value is large, the difference between two consecutive maximum peaks of x-axis velocity is allowed to be large. Conversely, this difference is almost zero when the s_{peak} value is very small.

In the smoothing problem, v_{moving} is added in the optimization variable in (3.23). Thus, a new optimization variable $\tilde{X}_{opt}$ has become as follows:

$$\tilde{X}_{opt} = \begin{bmatrix} \tilde{X} \\ v_{moving} \end{bmatrix} \in R^{9 \times N + 1}. \tag{3.32}$$

Also, the following term is added in (3.24):

$$\frac{1}{s_{peak}} \left(v_{p_i,x} - v_{moving} \right)^2. \tag{3.33}$$

The final walking trajectory $\hat{r}_{final}$ is obtained by imposing the constraint (3.31) (in addition to the existing constraints) to the rotated initial estimations of $\hat{q}_{rotated}, \hat{r}_{rotated}$, and $\hat{v}_{rotated}$, which are estimated by the smoothing algorithm in Section 3.3.1.

3.3.3 Walking Trajectory-based Step Length Estimation

After obtaining the final walking trajectory, a step length estimation method is applied to compute the walking step length when the step indexes are detected. Each walking step is identified by two consecutive minimum peaks of the filtered acceleration norm. These minimum peaks at the discrete-time k can be determined if the following condition is satisfied:

$$(\bar{y}_{a,k} < \delta_{min}) \text{ and } \bar{y}_{a,k} \leq \min\left(\bar{y}_{a,k-w_{size}} : \bar{y}_{a,k-1}\right) \text{ and } \bar{y}_{a,k} \leq \min\left(\bar{y}_{a,k+1} : \bar{y}_{a,k+w_{size}}\right), \quad (3.34)$$

where $\bar{y}_a$ is the filtered acceleration norm using the zero-phase filter with a normalized cutoff frequency of 0.2π radians/sample, δ_{min} is the minimum threshold value, and w_{size} is the window length.

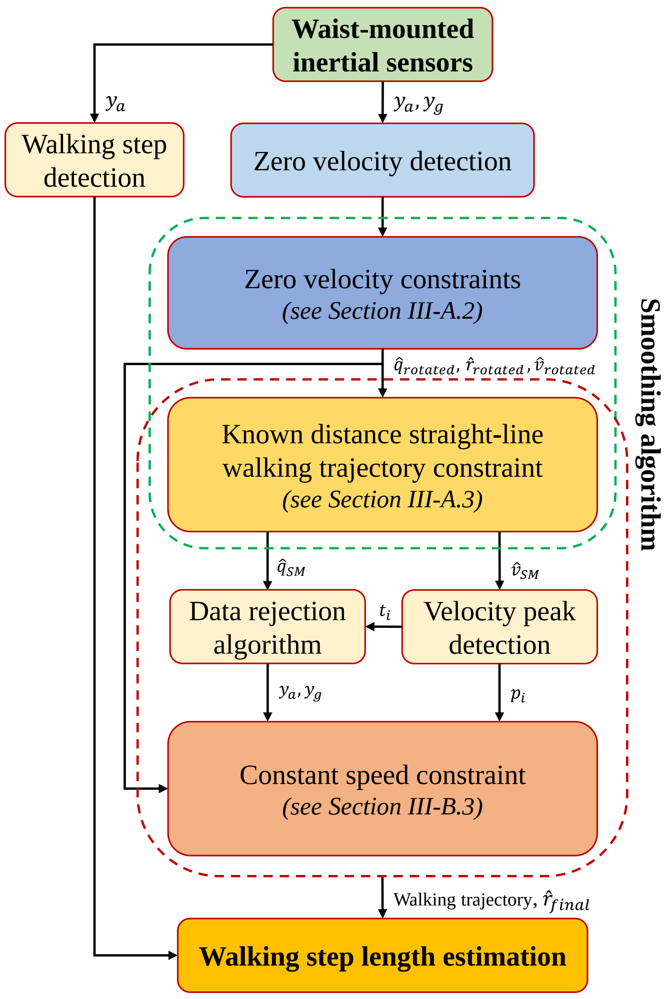

Waist-mounted inertial sensors
y_a
y_a, y_g
Walking step detection
Zero velocity detection
Zero velocity constraints
(see Section III-A.2)
$\hat{q}_{rotated}, \hat{r}_{rotated}, \hat{v}_{rotated}$
Smoothing algorithm
Known distance straight-line walking trajectory constraint
(see Section III-A.3)
$\hat{q}_{SM}$
$\hat{v}_{SM}$
Data rejection algorithm
t_i
Velocity peak detection
y_a, y_g
p_i
Constant speed constraint
(see Section III-B.3)
Walking trajectory, $\hat{r}_{final}$
Walking step length estimation

Figure 3-9: The proposed smoothing algorithm-based walking step length estimation framework.

Let $\hat{r} \in R^2$ be the x-axis and y-axis positions of the final walking trajectory, which is estimated by the proposed smoothing algorithm. Since the z-axis position is not used in the walking step length estimation, this information is omitted. The walking step length SL_s is computed as follows:

$$\mathrm{SL}_s = \left\| \hat{r}_{End_s} - \hat{r}_{Start_s} \right\|, \tag{3.35}$$

where $Start_s$ and End_s are the starting and ending indices of the s-th walking step, respectively.

An overview of the proposed smoothing algorithm-based walking step length estimation is shown in Fig. 3-9. Also, the parameters used in the proposed algorithm are given in Table 3.1.

Table 3.1: Parameters used in the proposed smoothing algorithm.

Parameters	**Symbol**	**Values**	**Related Equations**
Velocity peak detection	δ_{th}	0.05	Algorithm 1
	M_{size}	50	
	W_{size}	25	
Sensor noise	r_a	0.005 m/s^2	Equation (3.2)
	r_g	0.001 rad/s	
Zero velocity detection	α_a, α_g	1	Equation (3.3)
	α_{zero}	0.1	
	M_{zero}	40	
Step detection	δ_{min}	9.65 m/s^2	Equation (3.34)
	w_{size}	30	
Measurement noise	$s_{heading}$	0.0076 rad	
	$s_{init,r}$	0.0001 m	
	$s_{init,v}$	0.0001 m/s	
	$s_{init,2,v}$	0.001 m/s	
	s_{moving,r_y}	0.01 m	
	s_{final,r_x}	0.0004 m	
	s_{final,r_y}	0.0025 m	
	s_{final,r_z}	0.0009 m	
	$s_{final,v}$	0.0001 m/s	
	s_{peak}	0.04 m/s	

3.4 Chapter Summary

This chapter has proposed a smoothing algorithm-based approach to estimate the walking step length using waist-mounted inertial sensors. In our method, the walking trajectory was estimated by using the smoothing algorithm with known total walking distance. The constrained optimization problems including a known distance straight-line walking trajectory constraint and a constant speed constraint are used for the smoothing problem. Then the walking step length was computed based on the pedestrian trajectory and the detected walking step indexes.

Walking speed and step length estimations are important in the health care context and the personal navigation system. Using the proposed smoothing algorithm, a walking speed can be estimated without requiring a sophisticated measurement device. For example, elderly people can measure their walking speed accurately without visiting the clinic. Since the walking speed is closely related to the health condition, the walking speed information can be used to remotely evaluate the health condition. In the personal navigation system with a wearable inertial sensor, a person-dependent calibration procedure is required to relate the inertial sensor output and walking step length. The proposed smoothing algorithm can be used in this calibration procedure by providing accurate walking step length. In the next chapter, the deep learning-based regression model is applied for estimating the pedestrian traveled distance, where the proposed smoothing algorithm is used to obtain the walking step length as the reference label data in the training stage.

Chapter 4

Deep Learning-based Approach for Walking Distance Estimation

Wearable inertial sensors-based walking distance estimation is one of the important parameters in various areas such as pedestrian navigation systems [77], healthcare context [26], and sports training [78]. Besides, walking step length is a vital component in these applications, and plays a key role in walking distance estimation accuracy. Besides, the accurate walking distance and average speed can be used to calculate the total energy expenditure in our daily life [79]. It can help to evaluate the health condition as well as avoid overtraining that affects the ability to exercise. Various approaches with additional equipment such as pedometers or global positioning systems (GPS) have been used to estimate the walking distance. These methods only provide the total distance traveled and average walking speed without step parameters. Moreover, the GPS only works in an outdoor environment. On the other hand, an approach using inertial sensors mounted on the pedestrian's body provides step information and also the walking distance. This approach has flexible usage and does not require special infrastructure.

This chapter introduces a novel regression approach based on deep learning for estimating the walking distance using inertial sensors attached to the pedestrian's waist. Walking step length can be estimated by using supervised learning. However, supervised learning commonly requires a large amount of labeled training data to achieve better performance. To tackle this issue, we propose the walking step length estimation method based on conditional generative adversarial network (CGAN) used as a regression model. The CGAN-based regression model consists of a generator model for a step length regression task and a discriminator model for a classification task. Step segmentation is performed to extract acceleration amplitude data into step segments. These data are applied as additional input for both generator and discriminator. The generator model aims to generate walking step length as a label, while the discriminator model aims to classify an input label as either real or generated. Then, the step length prediction model using the CGAN-based regression approach is applied to calculate the walking distance.

4.1 Related Work

In the literature on the step length estimation method, several approaches using inertial sensors have been presented [11, 12]. These step length methods are divided into two ways: direct methods using the acceleration double integration, and indirect methods using gait walking models or statistical prediction techniques. In theory, the direct method is one of the best methods for walking step length estimation. Because this method does not require any gait model and also does not need personalized calibration or training procedures. Nonetheless, the noise and bias in the low-cost inertial sensors lead to the accumulation of integration errors over time. Thus, to reduce this accumulation error, several assumptions and heuristics have been used in different body-worn sensor cases [80–82]. For a foot-mounted inertial sensor case [80, 81], zero velocity updates (ZUPT) are used to reset integration errors when the foot is on the ground during movement. In contrast, there are almost no zero velocity intervals in one continuous movement when the sensor is mounted on other parts of the body like the pocket, waist, and wrist. Therefore, indirect methods are used to solve problems in these cases when direct methods are not suitable.

Many indirect methods that are classified according to the gait walking model or statistical regression method, have been proposed for walking step length estimation. The method based on the gait model is to define walking step length expressions using biomechanical models [83, 84], empirical relationships [55, 56, 85], and inverted pendulum models [58, 86, 87]. Another approach is the statistical regression method that uses the relationship between step length and sensor information. There are two main ways of regression methods: parametric and non-parametric models. The most common and straightforward way for parametric models is the linear regression method. This method is used to describe the linear regression between the step length and walking characteristic features. These features are obtained from the inertial sensor data, which is divided into four groups according to their walking information: step frequency [59, 61], acceleration-based feature [63, 64], angle-based feature [65, 66], and multiparameter [14, 88]. Non-parametric models are flexible methods that provide a good fit for the training process. These models generally achieve more accurate predictions than parametric ones [11], but they require a large number of labeled training datasets and need to find the best configuration of hyperparameters for models.

Recently, various methods based on non-parametric models have been proposed to obtain accurate step length estimation using deep learning approach [22, 68, 89–92]. Hannink *et al.* [89] proposed the stride length estimation using deep convolutional neural networks (CNN) for foot-mounted inertial sensor data. Diaz *et al.* [90] proposed a deep neural network (DNN) to estimate step length and step width using six features, which are extracted from the inertial sensors attached to five positions of the body. This method has comparable results with the CNN model used in [89], but it needs fewer features and parameters than the CNN model in the training stage. In [68], the stacked autoencoders were presented to the step length estimation using accelerometers and gyroscopes from the smartphone sensors. Similarly, in [91], the long short-term memory (LSTM) network and denoising autoencoders were proposed to measure stride length. These methods based on the deep learning approach are called supervised learning, which trains a model from training datasets with their step lengths used as labels. Besides, Sui *et al.* [92] proposed self-supervised learning using the CNN model to predict the stride length of running and walking. This method trains a pretext task for a large unlabeled dataset using self-supervised learning and then uses supervised learning to train a downstream task with small labeled datasets. However, the above supervised and self-supervised learning methods usually require a large amount of training data (labeled dataset for supervised learning, both labeled and unlabeled datasets for self-supervised learning) to train the better fit model. The aim of this chapter is to provide a step length prediction model that uses supervised deep learning with a small number of training data to estimate the walking distance accurately.

In this chapter, we propose a regression method based on a conditional generative adversarial network (CGAN) for walking distance estimation. Mirza *et al.* [93] proposed the CGAN, which is extended version of generative adversarial network (GAN) in [94]. The

CGAN takes auxiliary information as additional input that is fed to both the generator and discriminator of the CGAN model. The CGAN has been applied in various fields, including image-to-image translation [95], face aging [96], clinical diagnosis [97], text recognition [98], and sensor data generation [99, 100]. In [101], the step length and direction are estimated using GAN, where domain (sensor position) invariant features and transformation of sensor data to other domains are proposed. Compared with [101], the proposed method focuses on accurate step length estimation with reduced training dataset size. Aggarwal *et al.* [102] used the CGAN as a regression model to synthetic datasets sampled from heteroscedastic and multi-modal distributions, and then compared this model with conventional regression models on some real-world datasets. They have shown that this model can be competitive with other regression models. From this motivation, we develop a CGAN-based regression method that aims to generate walking step length and then compute pedestrian distance traveled. To the best of our knowledge, a CGAN-based regression approach to obtain the step length prediction model is a novel approach. In our work, we leverage the smoothing algorithm based on a constrained optimization in the previous chapter to estimate walking step length at various speed levels using inertial sensors attached to the waist. The constrained optimization-based smoothing algorithm is only applied for the training process, in which the total distance traveled is known. Thus, the estimated step lengths are collected as labels for the training datasets. The CGAN-based regression model is built using a deep neural network (DNN), which includes a generator model for a step length regression task and a discriminator model for a classification task. The acceleration amplitude data is extracted from the accelerometer output into each step segment, which is used as additional input for both the generator and discriminator. The step length prediction model is then used to compute walking distance when the testing dataset is obtained.

4.2 System Overview

4.2.1 System Architecture

The pedestrian uses the inertial sensor unit consisting of accelerometers and gyroscopes attached to the waist. Fig. 4-1 illustrates the system architecture of our method, mainly including three parts: data preprocessing, conditional generative adversarial network (CGAN), and walking distance estimation.

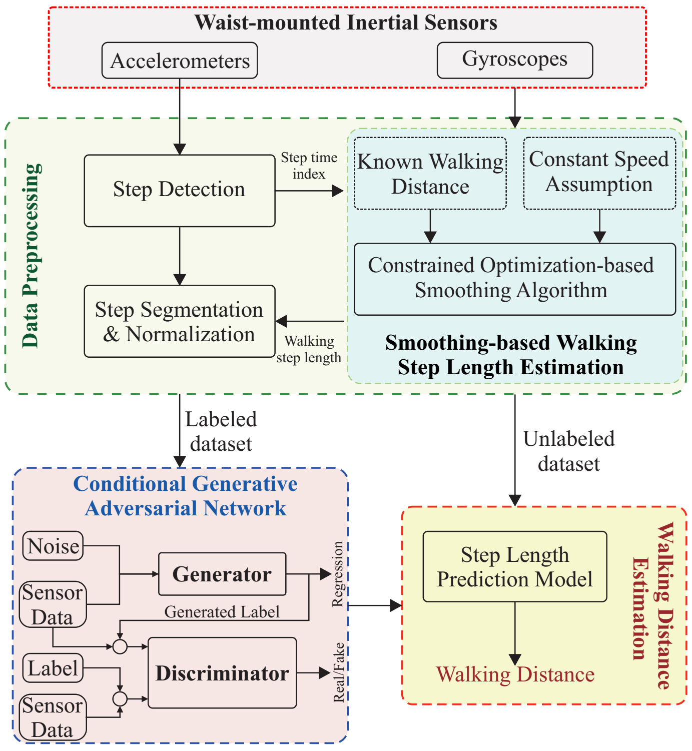

Waist-mounted Inertial Sensors
Accelerometers
Gyroscopes
Data Preprocessing
Step Detection
Step time index
Known Walking Distance
Constant Speed Assumption
Constrained Optimization-based Smoothing Algorithm
Step Segmentation & Normalization
Walking step length
Smoothing-based Walking Step Length Estimation
Labeled dataset
Unlabeled dataset
Conditional Generative Adversarial Network
Noise
Sensor Data
Generator
Regression
Generated Label
Label
Sensor Data
Discriminator
Real/Fake
Step Length Prediction Model
Walking Distance
Walking Distance Estimation

Figure 4-1: System architecture of the proposed walking distance estimation method.

In our approach, data preprocessing is firstly applied to generate the walking sensor data and ground truth data of the step length, which is obtained using a smoothing algorithm for a waist-mounted sensor case. Subsequently, we use a regression approach based on CGAN to obtain the step length prediction model. The CGAN consists of a generator model and a discriminator model. These two models are trained together. Then, a composite CGAN model using both two models is applied to train and update the weights of the generator via the discriminator's classification. Finally, the walking distance is estimated by accumulating every estimated step length, which is obtained using a step length prediction model.

4.2.2 Sensor System

Fig. 4-2 illustrates the overview system of the waist-mounted inertial sensor unit. It consists of a consumer-grade Xsens MTi-1 sensor unit, a micro secure digital (microSD) card slot, and a Nordic nRF51822 microcontroller built-in Bluetooth Low Energy (BLE). The wearable sensor module has a dimension of 40 mm × 30 mm × 10 mm. We use a velcro strip to mount the sensor module on the pedestrian's waist. The inertial sensor data is collected with a sampling rate of 100 Hz and saved to a microSD card for post-processing. Table 4.1

describes the detailed specifications of inertial sensor unit.

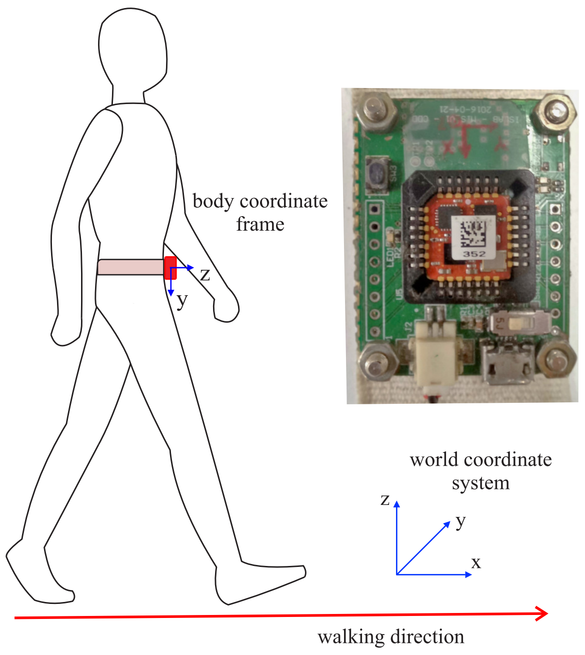


Figure 4-2: Overview system of the waist-mounted inertial sensor unit.

Table 4.1: Xsens MTi-1 sensor specifications.

Specification	**Accelerometers**	**Gyroscopes**
Full range	$\pm$ 16 g	$\pm$ 2000 °/s
Bandwidth	324 Hz	255 Hz
Noise density	120 $\mu/\sqrt{\text{Hz}}$	0.007 $°/\text{s}/\sqrt{\text{Hz}}$
Sampling rate	100 Hz	100 Hz

4.3 Algorithm Description

In this section, we first present data preprocessing for deep learning. A regression method based on a conditional generative adversarial network (CGAN) is then described. Finally, the walking distance is estimated using the step length prediction model.

4.3.1 Data Preprocessing

Step Detection

An essential part of the walking step length estimation method is to determine step events. The acceleration data exhibits a periodic pattern when the pedestrian is moving. To eliminate the effect of the sensor's orientation when setting the sensor placement, we use the amplitude of the acceleration data to detect the walking step events.

Let acc_k be the acceleration amplitude at the discrete-time k after the gravitational acceleration subtracted, which is given by

$$acc_k = \sqrt{acc_{x,k}^2 + acc_{y,k}^2 + acc_{z,k}^2} - g, \tag{4.1}$$

where acc_x, acc_y, and acc_z are the components of acceleration along the x-axis, y-axis, and z-axis in the sensor coordinate system, respectively, and g is the magnitude of gravitation. In (4.1), the gravitational acceleration magnitude g is subtracted so that acc_k is zero when there is no motion.

Algorithm 2 Step detection method algorithm

```
Input: Filtered acceleration amplitude $\bar{a}$, threshold value $\delta_{th} = 0.5$, window length $w = 25$.
Output: Step time index $t$.
1: Compute the length $N$ from the length of $\bar{a}$.
2: //Find the maximum peak index ($P$):
3: for discrete-time $k$ from $k = w + 1$ to $N - w$ do
4:     if $(\bar{a}_k > \delta_{th})$ and $\bar{a}_k \geq \max(\bar{a}_{k-w} : \bar{a}_{k-1})$ and $\bar{a}_k \geq \max(\bar{a}_{k+1} : \bar{a}_{k+w})$ then
5:         $P = P \cup \{k\}$.
6:     end if
7: end for
8: //Find the zero-crossing point ($Z$):
9: for each moment $k$ of $N$ do
10:    if $\bar{a}_{k-1} < 0$ and $\bar{a}_k \geq 0$ then
11:        $Z_k = Z_k \cup \{1\}$.
12:    else
13:        $Z_k = Z_k \cup \{0\}$.
14:    end if
15: end for
16: //Find the step time index ($t$):
17: Compute the number of peaks $N_P$ from the length of $P$.
18: for each moment $i$ of $N_P$ do
19:    $j = P_i - 1$.
20:    while $Z_j = 0$ do
21:        $j = j - 1$.
22:    end while
23:    $t = t \cup \{j\}$.
24: end for
25: return $t$
```

According to the principle of the walking pattern for the waist-mounted inertial sensors case [58], the highest peak (maximum peak) of the acceleration amplitude corresponds to the lowest position of the waist during a gait cycle (double stance phase). Following a description of gait events in [86], the instant of foot contact with the ground is detected when the downward slope of the acceleration signal passing through the zero. Therefore, two consecutive zero points of the acceleration amplitude data are used to determine a walking step event. To accurately obtain the walking steps, the step detection method combines the maximum peak detection with zero-crossing detection.

In order to remove noise and low-frequency acceleration signals, we use a moving average filter on the acceleration amplitude. The filtered acceleration amplitude ($\bar{a}$) is calculated as follows:

$$\bar{a}_n = \frac{1}{2W_{size} + 1} \sum_{k=n-W_{size}}^{n+W_{size}} acc_k, \tag{4.2}$$

where W_{size} is the window size, and n is the sampling point, $n = \{W_{size}+1, \cdots, N-W_{size}\}$ where N is the length of the acceleration data. In this study, the acceleration amplitude signal is filtered with a 15-point moving average filter (that is, $W_{size} = 7$) to reduce noise. Then, the step time index is detected using Algorithm 2. The values of parameters for the step detection method are determined through experimental analysis. These parameter values are applicable for various users and walking speed levels.

An example of a step detection method for walking along a 20-m straight path corridor at normal speed is illustrated in Fig. 4-3. The solid blue line represents the filtered acceleration amplitude, while the red circles indicate the maximum peaks of the filtered acceleration amplitude, and the black diamonds indicate the zero-crossing points. Each step segment is identified by two consecutive zero-crossing points.

Smoothing Algorithm-Based Walking Step Length Estimation

For the training process, supervised learning requires collecting the ground truth length of each walking step used as the label data. There are commonly two ways to obtain the ground truth of step length. The first way is based on the known traveling distance and then divides this measurement by the number of steps walked [68]. The other one uses more additional devices to extract the walking step length such as a foot-mounted inertial sensor [91] or an optical measurement system [92]. This method can present more accurate reference results. However, the training stage is more complicated and the investigation cost for

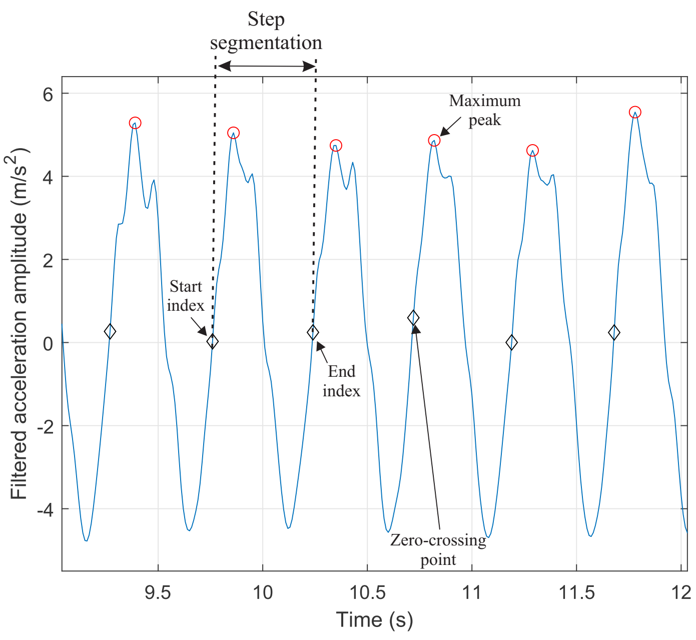


Figure 4-3: The filtered acceleration amplitude during normal speed is represented by the solid blue line, walking step time indexes are detected by two consecutive zero-crossing points of the filtered acceleration amplitude.

additional devices, such as an optical measurement system, could be expensive. Therefore, to obtain the labeled data, we use the constrained optimization-based smoothing algorithm in the previous chapter to estimate walking step length at various speeds, in which the total distance traveled is known.

The proposed smoothing algorithm-based walking step estimation consists of three parts as follows: 1) a smoothing algorithm with a straight-line walking constraint containing the zero velocity constraints (that is, a pedestrian is standing still events before and after walking) and the known distance straight-line walking trajectory constraint, which is applied to generate initial trajectory and then reduce its estimation errors; 2) a robust smoother based on the constant speed assumption, which is used to improve the accuracy of the walking trajectory by imposing the constraint at the double stance phase during moving interval; and 3) a step length estimation method that is used to calculate the length of each step as the step time indexes are determined. Hence, the walking step length SL_s is estimated as follows:

$$\mathrm{SL}_s = \left\| \hat{r}_{t_{s+1}} - \hat{r}_{t_s} \right\|, \tag{4.3}$$

where $\hat{r} \in R^2$ represents the position along the x-axis and y-axis of the pedestrian trajectory,

t_s and t_{s+1} are the starting and ending time indices of s-th walking step, respectively, which are determined using Algorithm 2.

Step Segmentation and Normalization

Once the walking step events are determined, the step segmentation is performed to extract the acceleration amplitude data into step segments. For the training dataset, each step segment is labeled with the corresponding ground truth of step length. To obtain a good quality of the dataset, we remove the initial and final step segments in each training data. Besides, to avoid false step detection, we remove the step segment if the estimated walking step length is less than 0.4 m or greater than 1 m. Then, each step data will be zero padded to a fixed size of 150 samples per step. This number of samples is enough to cover the length of one step for different walking speeds. Fig. 4-4 illustrates the amplitude of the acceleration data examples with various speeds after zero padding.

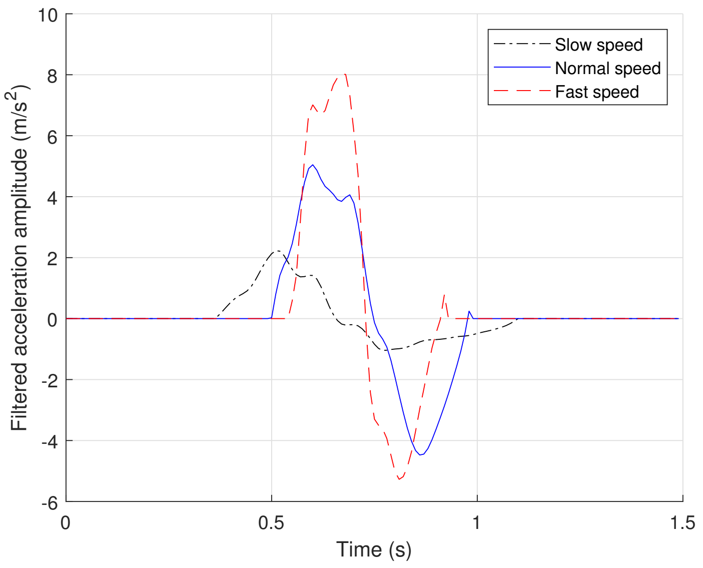


Figure 4-4: The acceleration amplitude data of the same subject with different walking speeds.

Finally, the normalization is applied to the acceleration amplitude in the dataset to avoid an imbalanced input data scale. The acceleration amplitude data is divided with their maximum value so that all values of data are within the range of $[-1, 1]$. In our dataset, the maximum value of the acceleration amplitude data is 15 m/s^2, which is used to normalize the training and testing datasets.

4.3.2 Regression Approach based on Conditional Generative Adversarial Network (CGAN)

This section presents a CGAN-based regression approach for the walking step length prediction task. Suppose we are given a training dataset $\{(x_i, y_i)\}_{i=1}^{M}$ with M samples, where $x_i \in R^{(1\times150)}$ are the preprocessed acceleration data input and $y_i \in R$ is the target output. In the regression method, we consider that an output value y is given by the model function $f(x)$ with additive Gaussian noise, can be defined as [102]:

$$y = f(x, z), \tag{4.4}$$

where $z \sim N(0, \sigma^2)$ is a zero-mean Gaussian random noise with variance σ^2 and $f(\cdot)$ is modeled using deep neural networks.

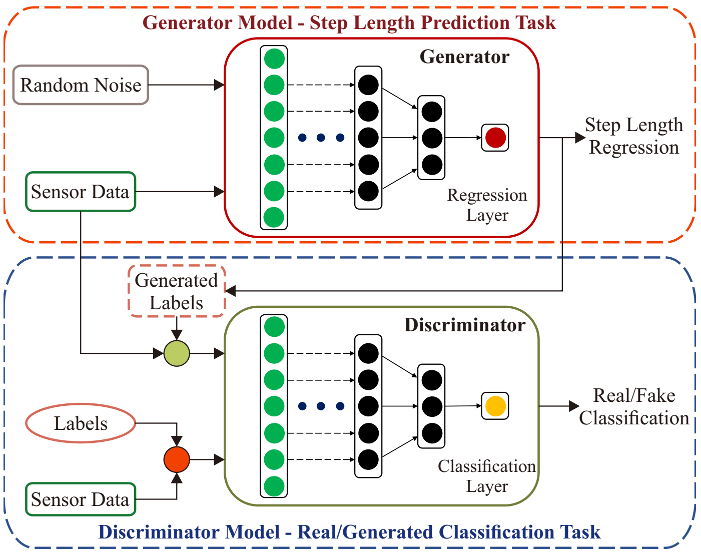


Figure 4-5: Framework architecture of the CGAN model.

Fig. 4-5 illustrates the framework architecture of the CGAN model, which consists of a generator model (G) and a discriminator model (D). The generator model aims to generate a continuous label distribution $G(x, z)$ due to the given sensor information and random noise, while the discriminator model aims to classify an input label y or $G(x, z)$ with output a probability $D(x, y)$ or $D(x, G(x, z))$ to indicate whether it is real (the label drawn from the dataset) or fake (generated).

Generator Model for Step Length Regression Task

In the generator model, a deep neural network (DNN) regression is used for a step length prediction task. The preprocessed acceleration data (x) and the random noise (z) are combined as inputs for this model, where the standard normal distribution $N(0,1)$ is used to generate random noise. The network architecture and parameters of the generator model are presented in Fig. 4-6. In the generator, we use a dense layer to transform the dimension of random noise, reduce the sequence of sensor data to the desired feature dimensions, and then these feature vectors are merged by a concatenate layer. Subsequently, four fully connected layers are finalized for the regression task. Besides, the hidden dense layers use a scaled exponential linear unit (SeLU) as the activation function, which solves the vanishing and exploding gradients problem in neural networks [103]. Also, the LeCun normal initialization is applied for each hidden layer's weights, which best fits for the SeLU activation function. The linear activation function is used for the output prediction layer. Since the CGAN model is $G(x,z)$ distributions, it can generate samples of label y (that is, walking step length) for each given sensor data x.

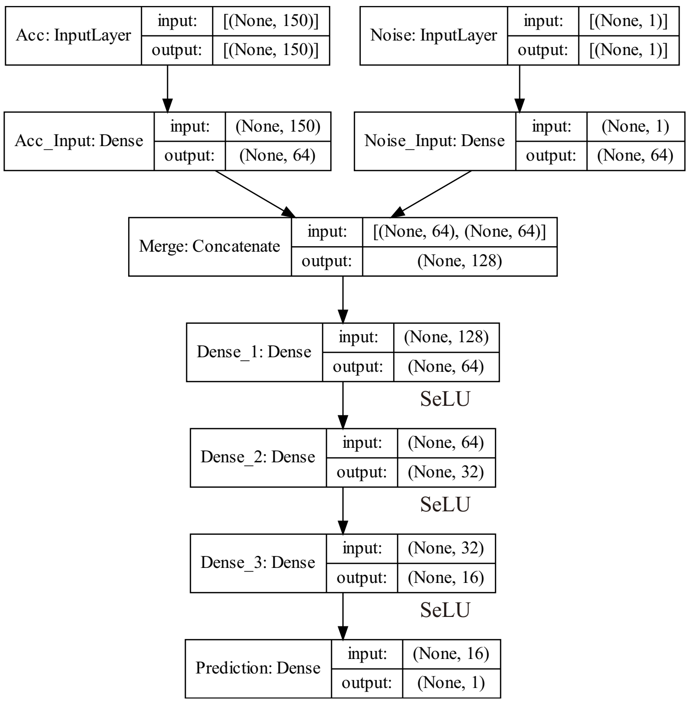


Figure 4-6: Architecture and parameters of the generator model.

Discriminator Model for Classification Task

The discriminator model takes the given acceleration data (x) and the true label (y) from the training dataset or the generated label $G(x, z)$ that is output by the generator model. They are fed to the deep neural network with the same network structure as the generator model. However, the discriminator model is designed for the binary classification task. Therefore, the output layer uses a sigmoid activation function to classify a binary class label as real or fake (generated). Fig. 4-7 illustrates the network architecture and parameters of the discriminator model.

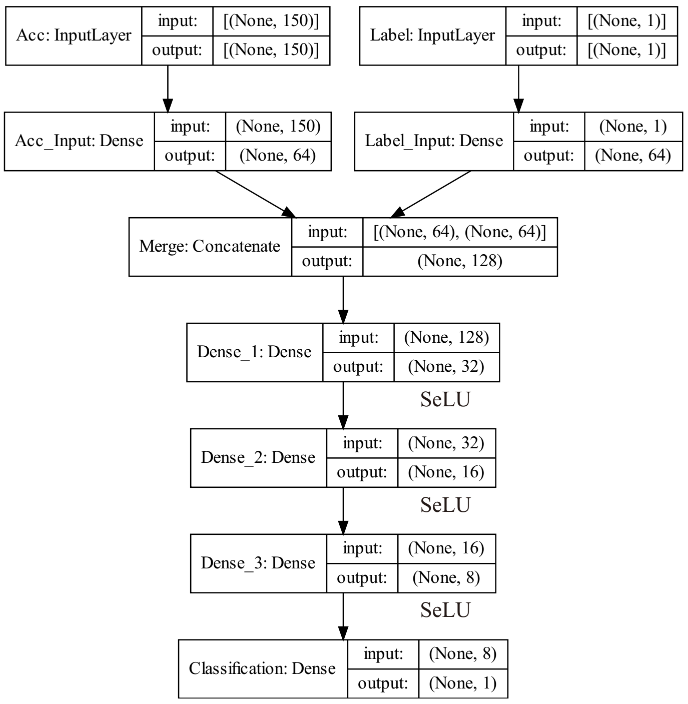


Figure 4-7: Architecture and parameters of the discriminator model.

Training Process of the CGAN Model

When the generator model and discriminator model are completely built on the deep neural networks, the CGAN model combines both of these models to train and update the parameter weights of the generator based on the backpropagation of the discriminator. During training, the generator and the discriminator are trained simultaneously. The discriminator tries to distinguish generated labels from real labels, while the generator tries to generate labels that look real enough to mislead the discriminator. Hence, the CGAN training process can be formulated as a minimum-maximum optimization problem with the objective function $V(\theta_G, \theta_D)$, where θ_G and θ_D are parameters of G and D, respectively [93]:

$$\min_{\theta_G} \max_{\theta_D} V(\theta_G, \theta_D) = \mathbb{E}_{x,y\sim p(x,y)}[\log D(x, y)] + \mathbb{E}_{x\sim p(x,y), z\sim p(z)}[\log(1 - D(x, G(x, z)))], \quad (4.5)$$

where $p(x, y)$ is the true data distribution, and $D(x, y)$ is the probability that y with the given sensor data x is a real data. Besides, $p(z)$ is a prior noise distribution of z, and $D(x, G(x, z))$ is the probability that the labels are generated from the generator model. At the beginning of the learning process, the discriminator can tell the generated labels from the real labels because they are distinctly different from the training data. Therefore, the $\log(1 - D(x, G(x, z)))$ function saturates for large values. However, when the generator is well-trained after a while, the $\log(1 - D(x, G(x, z)))$ function will be minimized.

Algorithm 3 CGAN-based regression training algorithm

```
Input: Training dataset with labels: $\{(x_i, y_i)\}_{i=1}^{M}$ with $M$ samples, epochs = 1000, batch
    size $m = \frac{M}{2}$, Adam hyperparameters: $\alpha = 10^{-4}$, $\beta_1 = 0.5$, $\beta_2 = 0.999$.
Output: Trained CGAN-based regression model
 1: Build the generator and discriminator networks.
 2: for number of training iterations do
 3:     //Train the discriminator:
 4:     Sample a batch size of $m$ training samples with labels $\{(x_i, y_i)\}_{i=1}^{m}$ from the real
        dataset.
 5:     Sample a batch size of $m$ noise samples $\{z_i\}_{i=1}^{m}$ from the prior noise distribution.
 6:     Generate a batch size of $m$ generated labels $\{G(x_i, z_i)\}_{i=1}^{m}$ using the generator model.
 7:     Update parameters of the discriminator model ($D$) by ascending its stochastic gradi-
        ent using Adam optimizer:
            $\nabla_{\theta_D} \frac{1}{m} \sum_{i=1}^{m} [\log D(x_i, y_i) + \log D(1 - (x_i, G(x_i, z_i)))].$
 8:     //Train the generator:
 9:     Sample a batch size of $m$ training samples without labels $\{x_i\}_{i=1}^{m}$ from the real dataset.
10:     Sample a batch size of $m$ noise samples $\{z_i\}_{i=1}^{m}$ from the prior noise distribution.
11:     Update parameters of the generator model ($G$) by descending its stochastic gradient
        using Adam optimizer:
            $\nabla_{\theta_G} \frac{1}{m} \sum_{i=1}^{m} \log D(1 - (x_i, G(x_i, z_i))).$
12: end for
13: return Trained $G$ model and $D$ model.
```

For the training process, we use the Adam optimizer [104] in the generator and discriminator, which has faster convergence than other stochastic optimization methods. The different learning rates (α) and decay rates (λ) are tuned to obtain the best performance. Besides, we train the CGAN model for 1000 epochs with different batch sizes. Through the experiment (see the details in the studies of Section 5.2.1), we found a learning rate of 10^{-4}, a momentum of 0.5, and a batch size with half of the training dataset size for both the generator and discriminator to work uniformly well. Also, the performance is improved as the decay rate is set to 10^{-4} for the generator and 0 for the discriminator. The discriminator model adopts binary cross-entropy as a classification loss function. The overall training procedure of the regression approach based on the CGAN model is shown in Algorithm 3.

4.3.3 Walking Distance Estimation using CGAN-based Regression Model

Walking Distance Estimation

Once the CGAN model is completely built and trained, the step length prediction model is obtained from the trained generator model. Then, this model is performed to generate the length of each step when the testing dataset is obtained, but not including labels. In the testing data, we detect the start and end moving events using a simple zero-velocity detection. They are combined with the step events to obtain the overall step segments of the testing dataset. For the CGAN-based regression approach, the walking step length can be generated with 100 samples for each preprocessed acceleration data. We take the mean of the generated samples to be the estimated value of walking step length ($\hat{L}$).

Finally, we compute the total walking distance (D_{est}) by accumulating the length of each step as follows:

$$D_{est} = \sum_{s=1}^{N_s} \hat{L}_s \tag{4.6}$$

where $\hat{L}_s$ represents the estimated step length of the s-th walking step and N_s is the total number of walking steps in the testing experiment.

Performance Evaluation Approach

In this work, we perform a stop criterion for the CGAN training to prevent over-fitting. This method automatically stops training when the performance of the CGAN model does not improve on a validation dataset within 100 epochs. Besides, we apply the repeated random subsampling validation with 30 iterations to evaluate the performance of the walking distance estimation using the CGAN-based regression model. This method splits the training dataset randomly into training and validation datasets for each subject separately. To ensure the balanced training data distribution with three walking speed levels: slow, normal, and fast, the data of each speed level are classified using the k-means clustering algorithm [105] for the estimated walking speeds of each subject. The estimated speeds are obtained by using the constrained optimization-based smoothing algorithm. In our work, the k-means clustering algorithm is carried out using MATLAB R2019b to obtain three clusters of speed data. The results of walking distance estimation are then averaged over the number of iterations. Also, we use the walking distance relative error and root mean square error to verify the performance of the proposed walking distance estimation method. The walking distance relative error (RE) is computed as follows:

$$\mathrm{RE} = \frac{|D_{est} - D_{true}|}{D_{true}} \times 100\%, \tag{4.7}$$

where D_{est} and D_{true} represent the estimated and actual walking distances, respectively.

The root mean square error (RMSE) is calculated by the following equation:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{W}(D_{est,i} - D_{true,i})^2}{W}}, \tag{4.8}$$

where W is the total number of testing data for all subjects, and $D_{est,i}$ and $D_{true,i}$ represent the estimated and actual walking distances of the i-th testing data, respectively.

4.4 Chapter Summary

This chapter presented a supervised deep learning-based approach for estimating walking distance using waist-mounted inertial sensors. To solve supervised learning with a small number of labeled training data, the CGAN-based regression approach was proposed, which was comprised of a generator model for a step length regression task and a discriminator model for a classification task. The step segments were extracted from the acceleration amplitude data. These data were applied as additional input for both the generator and discriminator. For the training stage, the ground truth of walking step length was estimated by using the constrained optimization-based smoothing algorithm. The training datasets were obtained from the extracted step segments and the corresponding ground truth of step length. Once the CGAN-based step length prediction model was completely trained. Then, the walking distance was estimated based on the step length prediction model when the testing data was obtained.

The accuracy of smoothing algorithm-based walking step length estimation will directly affect the performance of the CGAN-based step length prediction model, which is used as the label data for the training stage. In the next chapter, indoor and outdoor experiments will be performed to demonstrate the performances of smoothing algorithm-based walking step length estimation and deep learning-based walking distance estimation.

Chapter 5

Experimental and Results

5.1 Performance of Smoothing Algorithm-based Human Motion Estimation

To verify the performance of the proposed smoothing algorithm-based human motion estimation, two experiments are performed in this section. In the first experiment (see Section 5.1.1), a distance sensor system is installed to measure the pedestrian trajectory, which is used as the reference ground truth. In the second experiment (see Section 5.1.2), five healthy volunteers are recruited to obtain the 20-m walking data for three different speeds (slow, normal, and fast speed levels). This experiment is done to compare the walking step length estimation with the ground truth of each step length. The experiment system consists of an inertial sensor unit (Xsens MTi-1 sensor unit) mounted on the user's waist with a 100 Hz sampling frequency.

5.1.1 Walking Trajectory Estimation Validation Experiment

In this experiment, a person walked along a 20 m straight-line corridor with three different speed levels (slow, normal, and fast). Each walking speed is repeated three times (that is, a total of 9 times walking). Meanwhile, the movement of the user's waist is tracked using a distance sensor system. The experiment setup is shown in Fig. 5-1. The distance sensor system consists of a Nordic nRF51-DK board supporting Bluetooth low energy (BLE), an Arduino Uno board, and two Lidar-lite v3 modules. Besides, the inertial sensor is embedded with an nRF51822 microcontroller (built-in BLE). We use a laptop with an nRF51 USB

Dongle to control two systems simultaneously interface via the UART terminal. The Uno board collects the distance data from two Lidar modules using an I^2C communication protocol. Two Lidar-lite v3 modules are mounted side by side (horizontal to each other) in the distance sensor system, and the average of two sensors is used as the true trajectory of the waist. Since the waist position is moved back and forth in the direction of the y-axis during walking, while the coordinate system of the distance sensor is fixed and the direction of laser light coincides with the x-axis of the world coordinate system. The specifications of a Lidar-lite v3 module are shown in Table 5.1. Both the inertial sensor and distance sensor are collected at a 100 Hz sampling frequency.

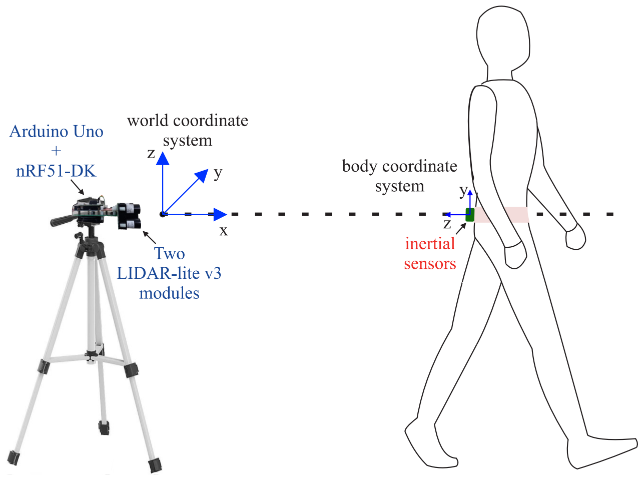
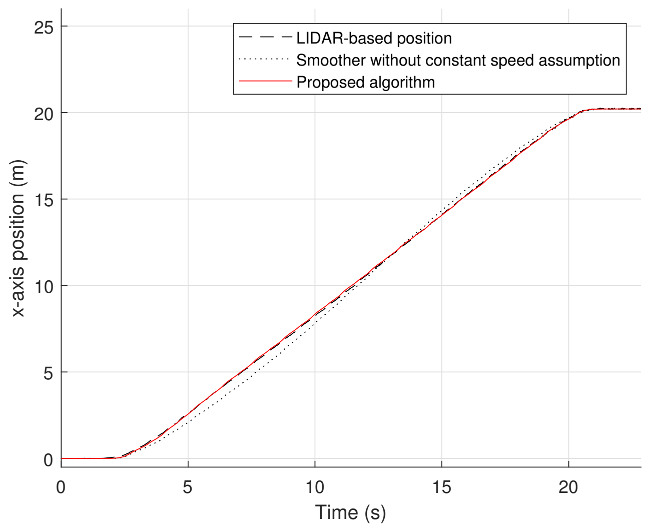


Figure 5-1: Experiment setup for the walking trajectory estimation validation.

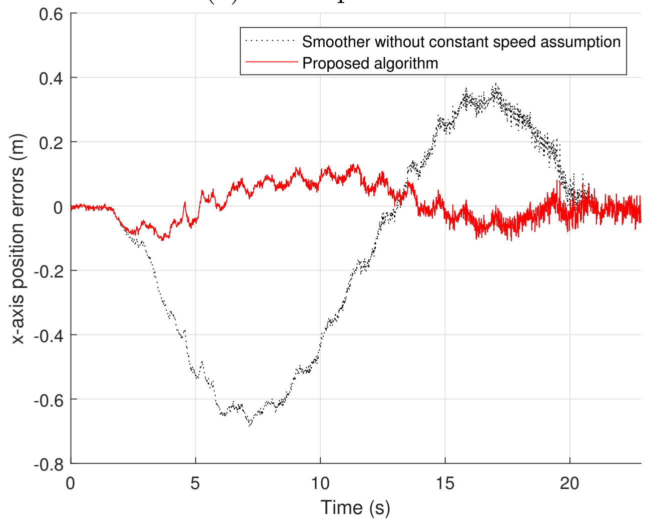
Table 5.1: Lidar-lite v3 specifications.

Specification	**Measurement**
Range	40 m
Resolution	± 1 cm
Accuracy < 5 m	± 2.5 cm
Accuracy ≥ 5 m	± 10 cm Mean ± 1% of distance maximum
Sampling rate	100 Hz

To evaluate the walking trajectory estimation using the proposed smoothing algorithm, we compare both the estimations of the position and velocity with the Lidar-based references in this experiment. Firstly, the estimated positions of a pedestrian using the smoothing algorithm without and with the constant speed assumption are compared with the reference position. This reference is measured from the distance of the sensor system to the user's waist when obtaining the walking data.



(a) x-axis position.

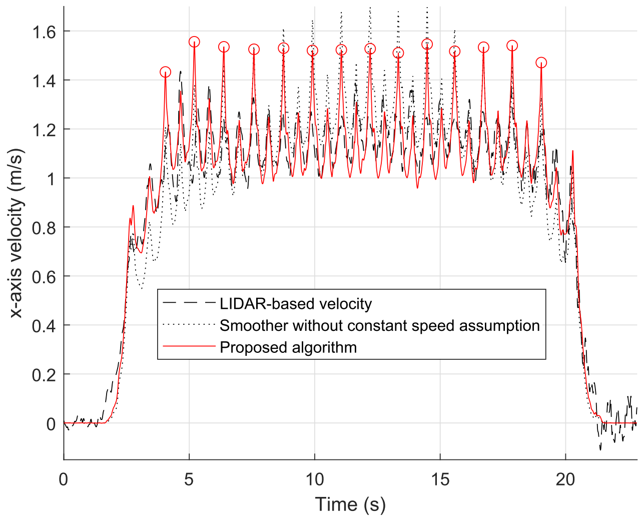


(b) x-axis position errors.

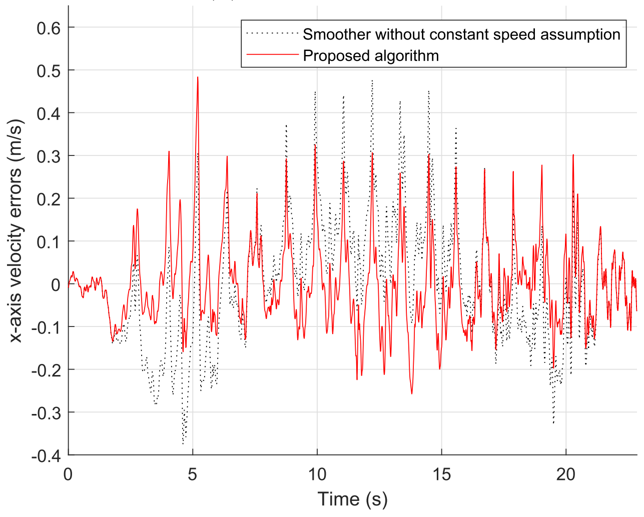
Figure 5-2: Comparison the estimated positions with the reference ground-truth.

An example for the comparison results of the two proposed methods with the reference position is illustrated in Fig. 5-2. Fig. 5-2a shows the results of x-axis position estimation using two proposed methods and the reference method. We can see that these methods give the accuracy of the final position estimation (after 20 s). However, the estimated position using the smoothing algorithm without the assumption is not accurate during a moving interval (see the values from 2.5 to 20 s). Thus, this method result is not good enough for the walking step length estimation. Meanwhile, the proposed smoothing algorithm is a significant improvement, which gives the result almost similar to the reference position. We also show the comparison errors of x-axis position estimation using two methods in Fig. 5-2b. The figure shows that the proposed smoothing algorithm gives smaller errors when the constraints are imposed. As can be seen, most of the position errors are less than 0.1 m.

We compare the estimated velocity results of a pedestrian using the two proposed methods with the reference velocity. This velocity is calculated from the derivative of eighth-order spline approximation [106] of Lidar distance data.



(a) x-axis velocity.



(b) x-axis velocity errors.

Figure 5-3: Comparison the estimated velocities with the reference ground-truth.

To illustrate the estimated velocity comparison of the example in Fig. 5-2, we show the comparison results of x-axis velocity estimation using the two methods with the reference velocity in Fig. 5-3. Fig. 5-3a shows that the smoothing algorithm without the assumption gives inaccurate results in the middle of the moving interval. Since there is no constraint for the velocity, it is difficult to avoid the larger error in this interval. In contrast, the proposed smoothing algorithm using the constant speed assumption has the trajectory of x-axis velocity almost similar to the reference velocity. However, the velocity peaks between the two methods have a bit different. This is probably diving into the fact that the reference velocity is estimated from the numerical derivation of Lidar distance data. Since the spline function is a low-pass filter whose bandwidth depends on the control parameters, the reference velocity is a low-pass filtered result. Fig.5-3b shows the comparison errors of x-axis velocity estimation using two methods. We can see that the proposed smoothing algorithm with the assumption is better than without the assumption.

Table 5.2: RMSE of the estimated position and velocity using the proposed algorithm without and with constant speed assumption.

Walking ID	**Position RMSE (m)**		**Velocity RMSE (m/s)**	
	Without assumption	With assumption	Without assumption	With assumption
1	0.4099	0.0793	0.1310	0.1047
2	0.3356	0.0560	0.1410	0.1007
3	0.3490	0.1505	0.1520	0.1257
4	0.3534	0.0748	0.1915	0.1576
5	0.2714	0.1017	0.1777	0.1441
6	0.2949	0.0832	0.1243	0.1115
7	0.4103	0.0801	0.2079	0.1308
8	0.2517	0.0524	0.1734	0.1209
9	0.4179	0.1121	0.2273	0.1344
Mean	0.3438	**0.0878**	0.1696	**0.1256**

Finally, we compare the root mean square errors (RMSE) of x-axis position and velocity estimations for different walking data, which are given in Table 5.2. These walking data are numbered from 1 to 9 (walking ID = {1-3}: slow speed, {4-6}: normal speed, and {7-9}: fast speed). As we can see, the estimation errors using the proposed algorithm with a constant speed assumption are significantly smaller than those without the assumption. Although the walking speed is assumed to be zero at the beginning and the final, the speed estimation during the moving interval could be very large. The constraint prevents this divergence. Thus the position and velocity estimation errors are further reduced by imposing the constant speed constraint. The averages of the position and velocity RMSEs using the proposed algorithm with the assumption are 0.0878 m and 0.1256 m/s respectively, which are better than the smoothing algorithm without the assumption.

5.1.2 Walking Step Length Estimation Validation

In this experiment, five healthy volunteers are asked to walk along a 20 m straight-line corridor (15 times) at three different speeds: slow (5 times), normal (5 times), and fast (5 times). Each participant chooses a preferred walking speed level to conduct the experiment.

To evaluate the walking step length estimation accuracy, the average step length is used as the ground truth of each step length. This average step length is computed by the total walking distance divided by the total number of walking steps. The walking step length estimation error between the proposed method and the average step length is computed. Fig. 5-4 shows the mean relative error (MRE), mean absolute error (MAE), and RMSE of each subject for three walking speed levels. We can see that the performance metrics of each subject have small errors. The averages of MAE and RMSE for three walking speed levels are almost similar and smaller than 0.05 m.

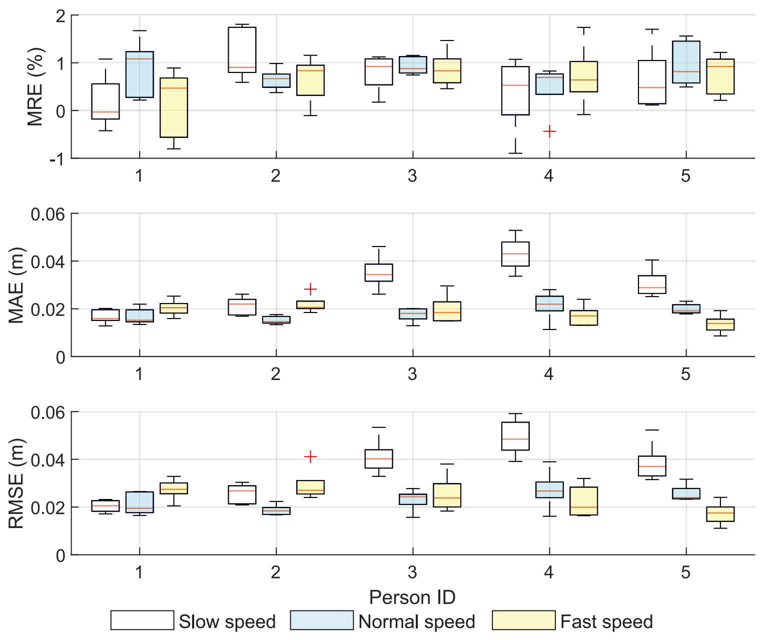


Figure 5-4: Performance metrics (MRE, MAE, and RMSE) of the proposed method-based walking step length estimation.

We compare the estimation errors of walking step length using the smoothing algorithm without and with constant speed assumption as shown in Fig. 5-5. This figure shows the comparison errors of five subjects using two methods for three walking speed levels. We can see that both methods tend to give better results when increasing walking speed. This is due to the shorter walking time for fast walking. If the walking time is shorter, the integration error becomes smaller and the estimation performance improves. The proposed smoothing algorithm with the constant speed assumption gives smaller errors than without the assumption. The average MAEs of five subjects using this method are 0.0301 m, 0.0184 m, and 0.0183 m corresponding to slow, normal, and fast speed cases, respectively. Likewise, the average RMSEs for three walking speed levels are 0.0377 m, 0.0241 m, and 0.0251 m, respectively. Furthermore, the average of MRE is 0.6801% for all walking data. Therefore, these results demonstrate the usefulness of the walking step length estimation of our proposed method.

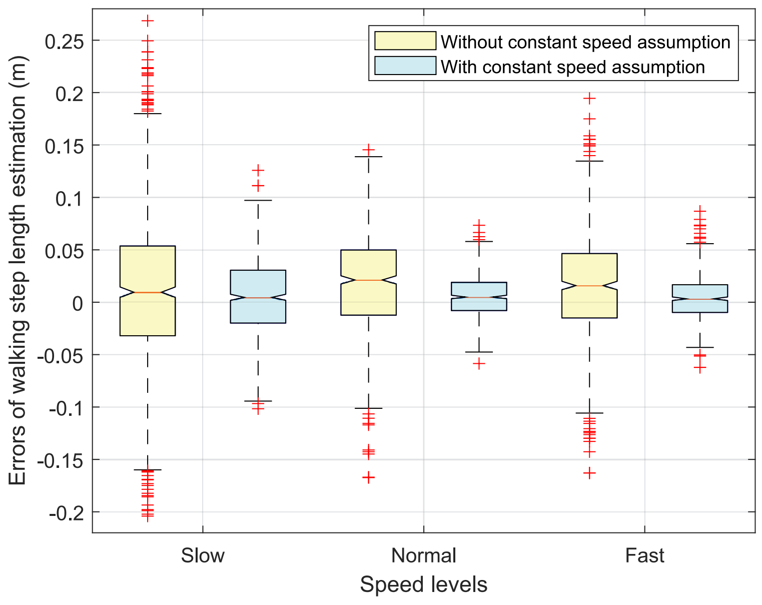


Figure 5-5: Comparison of walking step length estimation errors using the smoothing algorithm without and with constant speed assumption at different speed levels.

5.2 Performance of Deep Learning-based Walking Distance Estimation

In this section, the data preprocessing is carried out using MATLAB R2019b to obtain the training and testing datasets. The CGAN-based regression model is then implemented and trained using TensorFlow 2.3.0 [107] for the walking distance estimation method. To verify the performance of the deep learning-based walking distance estimation method, two experiments are performed. The first experiment is to evaluate the estimation accuracy of walking distance using the CGAN-based regression model. The effectiveness of hyperparameter tuning is also considered. In this experiment, we recruited twenty healthy volunteers whose information is shown in Table 5.3. To compare the proposed method with other state-of-the-art methods, an extensive rectangular test in the outdoor environment is performed in the second experiment.

Table 5.3: Twenty subjects information.

User	**Gender/Age**	**Height/Weight**	**User**	**Gender/Age**	**Height/Weight**
1	M/33	170/72	11	M/30	168/68
2	M/30	160/51	12	F/26	156/45
3	M/28	170/78	13	M/33	168/59
4	M/31	171/77	14	F/27	153/60
5	M/27	170/69	15	F/26	162/48
6	F/29	152/47	16	M/36	160/66
7	F/26	154/45	17	M/27	180/75
8	M/28	170/65	18	M/33	158/72
9	M/25	167/63	19	M/27	172/74
10	M/32	174/76	20	M/33	165/65

M: Male, F: Female; height in (cm) and weight in (kg).

5.2.1 Walking Distance Estimation Validation

In this experiment, twenty participants were asked to collect data using the inertial sensors attached to the waist (see Fig. 5-6). To obtain the labeled datasets for the training stage, participants walked along a straight path corridor of 20 m length in three various speed levels such as slow, normal, and fast. Each walking speed level was repeated five times (a total of 15 times walking). For collecting the testing data, participants were then asked to walk along a straight path of 80 m length at mixed walking speed levels repeated twice. The participants choose their preferred speeds to perform the experiments. To guarantee that the participants walk along a straight path within the actual walking distance, we draw a straight line with a ruler, and then they complete the experiments on this line. Each participant is advised to walk a straight line and if not he/she will be asked to walk again. Table 5.4 shows the detailed descriptions of the training and testing data from 20 participants.

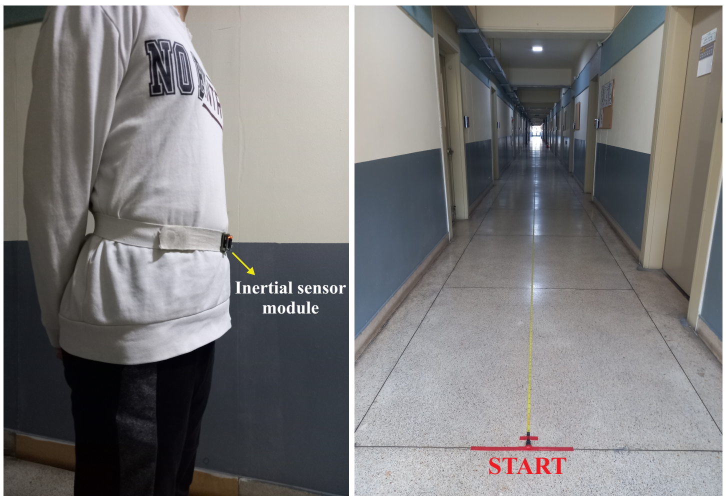


Figure 5-6: Experimental setup: an inertial sensor module is mounted on the user's waist (left) and an indoor environment is designed to collect the training and testing data (right).

Table 5.4: Descriptions of collected data from twenty subjects.

Usage	**Walking speed**	**Length (m)**	**Number of walking data**	**Number of step segments**
Training data	Slow	20	100	3156
	Normal	20	100	2988
	Fast	20	100	2390
Testing data	Mixed	80	40	4896

Effect of Different Hyperparameter Tuning Methods

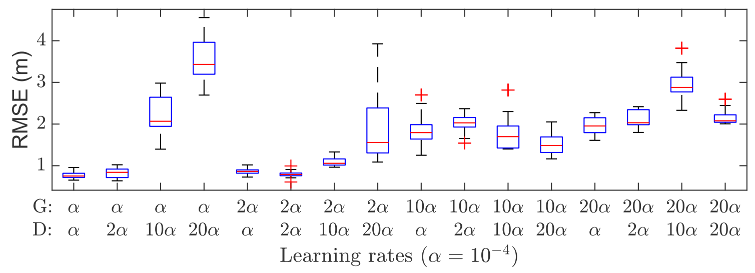
We consider the hyperparameter tuning for the proposed method to obtain the best performance. The hyperparameters are searched and selected for the generator model (G) and discriminator model (D) as given in Table 5.5. To investigate the effect of different hyperparameters on the CGAN-based regression model, we compare the performance of the walking distance estimation with different hyperparameters such as learning rates and decay rates for Adam optimizer, or batch sizes and dimensions of noise for the training procedure.

Table 5.5: Hyperparameter tuning for generator and discriminator models.

Hyperparameter	**Search Range**	**Selected Value**	
		Generator	**Discriminator**
Optimizer	{SGD, RMSprop, Adam}	Adam	Adam
Learning rate, α	$\{10^{-4}, 2\times10^{-4}, 10^{-3}, 2\times10^{-3}\}$	10^{-4}	10^{-4}
Momentum, β_1	{0.5, 0.9}	0.5	0.5
Decay rate, λ	$\{0, 10^{-4}, 10^{-3}, 10^{-2}\}$	10^{-4}	0
Batch size, m (ratio to size of training dataset)	$\{1/10, 1/5, 1/4, 1/3, 1/2, 1\}$	1/2	1/2
Noise distribution	{Normal, Uniform}	Normal	-
Dimension of noise, z	$\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$	1	-

In our work, the CGAN-based regression approach is trained with two training datasets to obtain the step length prediction models: one is using the training datasets of all subjects to train a general model, which is called a universal model; the other is using the individual training dataset to train a personal model for each subject, which is called a personalized model. In this subsection, we only use the training dataset of each subject with 300 step segments to obtain the personalized model. The performance comparison between these two models will be presented later.

Fig. 5-7 illustrates the RMSE of the proposed method in different learning rates (α) and decay rates (λ). For the graph above of Fig. 5-7, we tune the Adam optimizer with different learning rates of $\{10^{-4}, 2\times10^{-4}, 10^{-3}, 2\times10^{-3}\}$ for the generator and discriminator. Also, the decay rate and batch size are set to 0 and 100, respectively. Other hyperparameters are chosen as default values in Table 5.5. Similar to the graph below of Fig. 5-7, the decay rates of $\{0, 10^{-4}, 10^{-3}, 10^{-2}\}$ are tuned, and other hyperparameters are fixed with learning rate of 10^{-4} and batch size of 100. As can be seen, the average of RMSE values is the smallest when the learning rate is set to 10^{-4} for both the generator and discriminator. Besides, the RMSE values are improved as the decay rates are suitably selected. If the decay rate is used with the values of 10^{-4} for the generator and 0 for the discriminator, the average RMSE is the smallest.



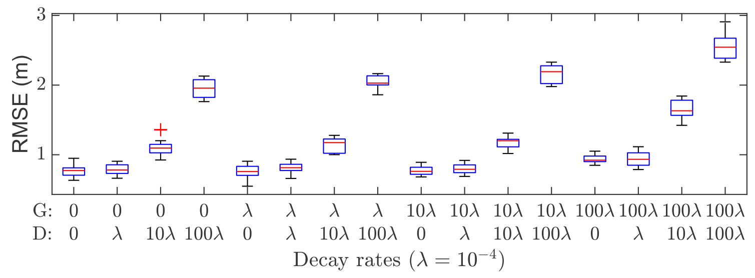
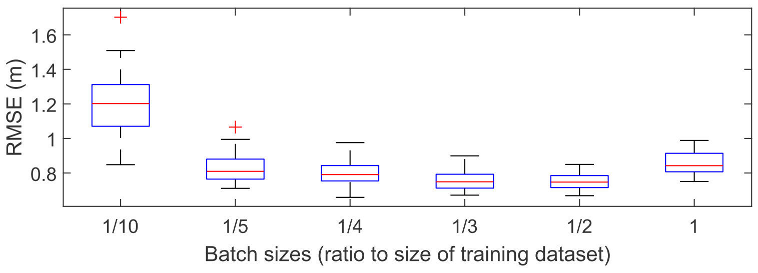


Figure 5-7: RMSE of the proposed method in different learning rates (top) and decay rates (bottom). The first and second rows of the x-axis data label represent the value of the learning rate (α) and decay rate (λ), respectively, for the generator (G) and discriminator (D).



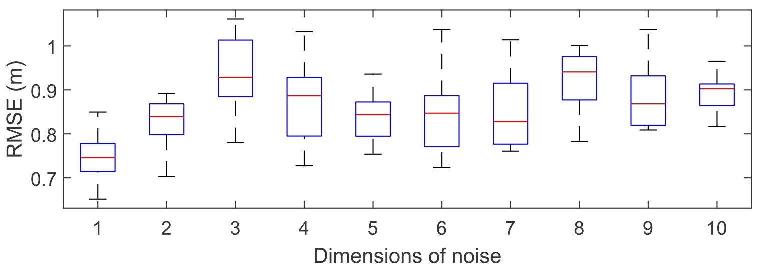


Figure 5-8: RMSE of the proposed method in different batch sizes (top) and dimensions of the noise (bottom). Batch size is set to ratio $\{1/10, 1/5, 1/4, 1/3, 1/2, 1\}$ of the training dataset size, which is used with 300 step segments.

Next, we study the effect of different batch sizes and dimensions of noise on the proposed method. The batch size is determined by the size of the training dataset. Interestingly, the

RMSE values are smaller when the batch sizes are increased as shown in Fig. 5-8. However, when the batch size is equal to the size of training data (also called batch gradient descent), the RMSE does not change significantly. Besides, Fig. 5-8 shows that the RMSE does not improve as we increase the dimension of noise from 1 to 10. From these results, we choose the hyperparameters as given in Table 5.5.

Usefulness of Different Amounts of Training Data Used

To demonstrate the performance of the proposed method with a small amount of training data, we use the repeated random subsampling with 30 iterations to split the training dataset randomly into training and validation data for each subject separately. The training data are split incrementally from 10 to 100 percent of the dataset and ensured a balanced data distribution with three walking speed levels. These data are applied to train the CGAN-based regression model. Then, the testing dataset is used to estimate the pedestrian walking distance.

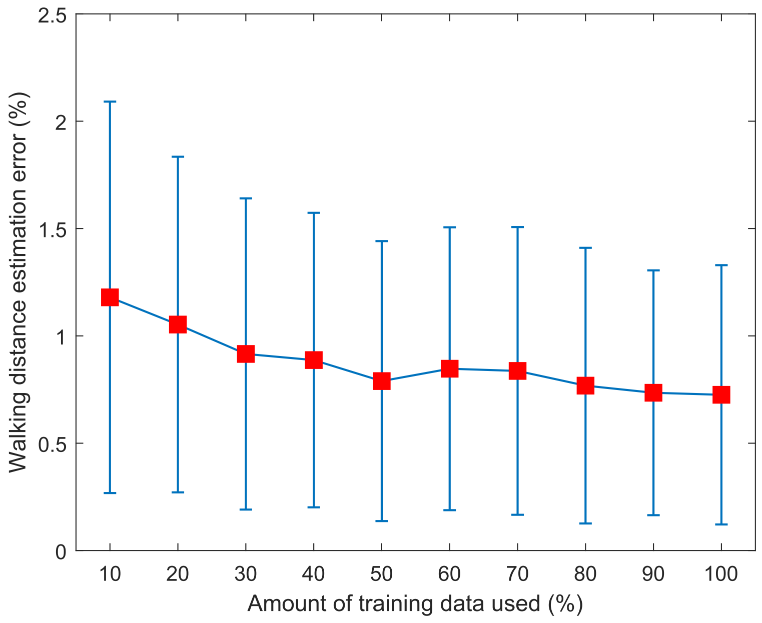


Figure 5-9: Walking distance estimation error using different amounts of training data used. The red squares and length of error bars indicate the mean and standard deviation of the estimated walking distance for all subjects, respectively.

Fig. 5-9 illustrates the error bars of the walking distance estimation with different amounts of training data used. The average value and standard deviation of the estimated walking distance for all subjects are represented by the red square shapes and the length of error bars, respectively. Besides, the error bar graph is shown as a curve with the number of samples used. The figure shows that the proposed method achieves small errors with a small number of samples since the error gradually converges as the ratio reaches 30 to 40 percent. Also, it just improves slightly as more samples are used. This implies that the regression model is well-trained when the training dataset is sufficient with 5 walking data (33.33% of 15 walking data).

Walking Distance Estimation Accuracy

In this subsection, firstly, we compare the regression approach based on our CGAN model with a basic deep neural network (DNN) model. The basic DNN model is built in the layers and adopts the hyperparameter tuning method in the same way as the generator model. The hyperparameters of the basic DNN model are used for the Adam optimizer with the learning rate (α) of 10^{-4}, the momentum (β_1) of 0.9, the decay rate (λ) of 10^{-3}, and batch size with one-third of the training dataset size. Besides, the basic model uses the mean squared error as a loss function instead of binary cross-entropy. For the training process, we take the number of training data to be in $\{30, 60, 120, 240, 300\}$ samples.

The comparison results of 80 m walking distance estimation using our CGAN model and a basic DNN model are given in Table 5.6. In Table 5.6, the average values, standard deviation (STD), mean relative error (MRE), and RMSE over 30 repeated random subsampling are computed from the estimated walking distances of 20 subjects. We can see that the results of the proposed method using the CGAN model are always more accurate than those using the basic DNN model. This implies that our method using supervised learning with a CGAN-based regression model is a better way to solve the supervised learning on a small number of training data. When the number of training data is very small (30 samples), the CGAN model has significantly better than the basic DNN model, where the MRE values are 1.11% and 1.43%, respectively. Besides, with the increase in training data size greater than 120 samples, the estimation accuracy of both the basic DNN and CGAN models has little improvement. When the training data are used with 300 samples, the MRE and RMSE using the CGAN model are 0.74% and 0.75 m, respectively, which are smaller than the proposed method using fewer samples.

Subsequently, we compare the performance of the proposed method with the DNN model using different features. This DNN model is comprised of four fully connected layers,

Table 5.6: Performance comparison of walking distance estimation using basic DNN and CGAN models.

Training dataset size (number of step segments)	Evaluation metrics	Model	
		Basic DNN	CGAN
30	**Average ± STD (m)**	80.06 ± 1.49	79.86 ± 1.12
	MRE (%)	1.43	1.11
	RMSE (m)	1.49	1.13
60	**Average ± STD (m)**	80.05 ± 1.35	79.86 ± 0.99
	MRE (%)	1.28	0.97
	RMSE (m)	1.35	1.00
120	**Average ± STD (m)**	80.12 ± 1.28	79.90 ± 0.84
	MRE (%)	1.20	0.82
	RMSE (m)	1.29	0.84
240	**Average ± STD (m)**	80.16 ± 1.25	79.89 ± 0.76
	MRE (%)	1.14	0.75
	RMSE (m)	1.26	0.77
300	**Average ± STD (m)**	80.15 ± 1.23	79.86 ± 0.74
	MRE (%)	1.12	0.74
	RMSE (m)	1.24	0.75

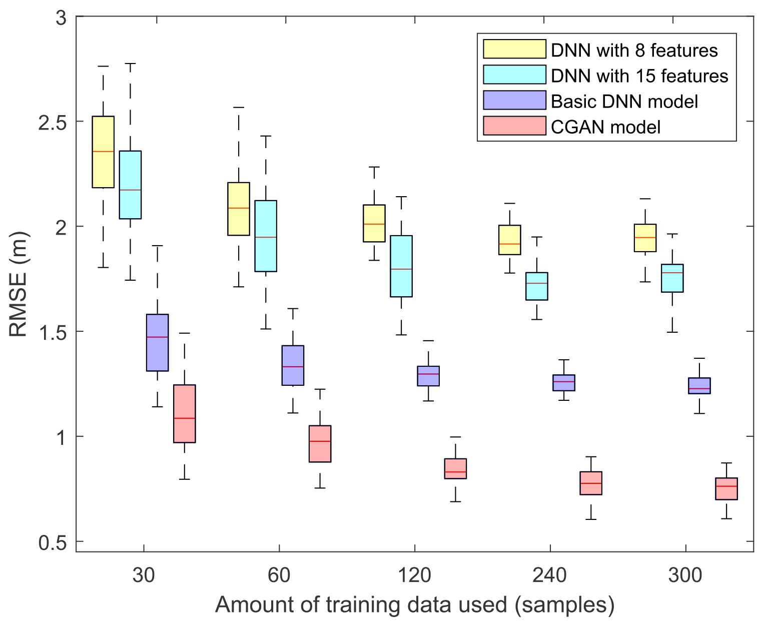


Figure 5-10: Comparison of RMSE between different regression methods.

where the network has three hidden layers of five neurons each and one output layer. The activation function for each layer and hyperparameters are used in the same way as the basic DNN model. The learning rate of Adam optimizer is set to $\alpha = 10^{-3}$ instead of $\alpha = 10^{-4}$

as the basic DNN model. The input of the DNN model is two feature vectors extracted from the acceleration data. One feature vector includes 8 common features: the mean, median, standard deviation, skewness, kurtosis, energy, zero-crossing rate, and maximum peak in the frequency domain. Another feature vector includes 15 features: step frequency, maximum and minimum values, range, variance, and two high-order features proposed by Weinberg [55] and Kim *et al.* [56], in addition to the 8 common features. Fig. 5-10 illustrates the performance comparison of the walking distance estimation using different regression methods. As can be seen, the proposed method using the CGAN-based regression model achieves better results than the basic DNN model and the DNN model with commonly-used features. As we increase the number of features, the accuracy of the estimation results using the DNN model can be improved. However, it needs a complicated feature selection method to obtain better performance.

Finally, we compare the usefulness of the walking distance estimation using the universal and personalized models. Recall that a combination data of 20 subjects is used to train the universal model. Fig. 5-11 illustrates the estimation errors of 20 subjects using the proposed method with two models. We can see that the results using the universal model give larger errors than those using the personalized model. These large errors are probably due to differences in pedestrians' characteristics and their step lengths.

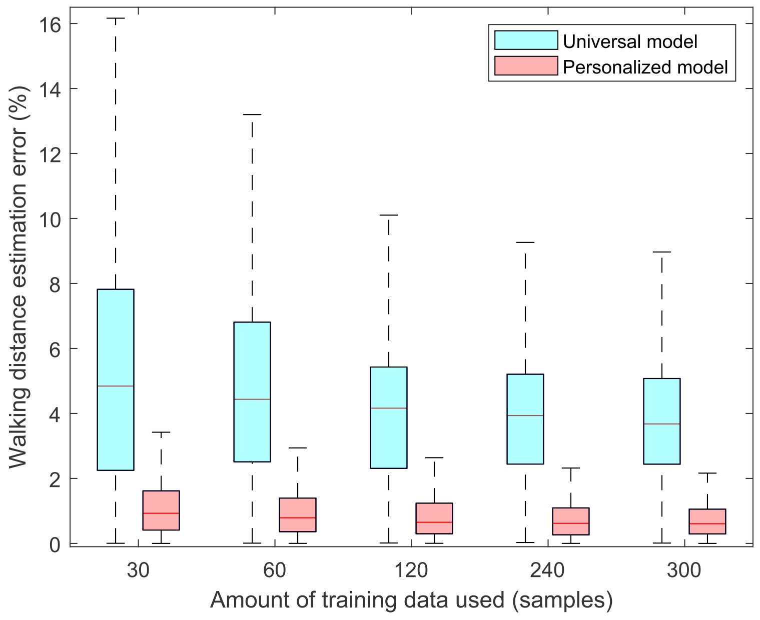


Figure 5-11: Comparison of RE using the universal and personalized models.

5.2.2 Comparison With Other Conventional Methods

To verify the performance of the proposed walking distance method, we conduct an extended experiment in the outdoor environment. Five of the twenty participants were chosen to walk along the path of a rectangular football field, in which the acceleration data is collected for the testing dataset. Each participant was asked to walk three times around the football field at three various walking speed levels (such as slow, normal, and fast), which is illustrated in Fig. 5-12. This walking data is repeated twice. The total walking distance of three rounds on the field is 1281.42 m.

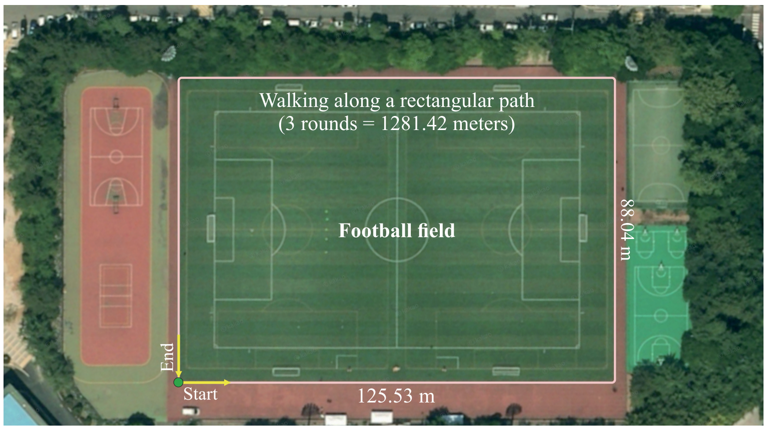


Figure 5-12: Walking along the path of a rectangular on the football field.

We compare the performance of our method using the personalized model with the conventional methods, including an empirical relationship proposed by Weinberg [55], the linear regression between step length and walking features proposed by Shin *et al.* [14] and Guo *et al.* [63], and other methods based on deep learning such as the DNN model proposed by Diaz *et al.* [90] and the CNN model proposed by Hannink *et al.* [89]. These methods are implemented in the same training and testing datasets as our method with 30-repeated random subsampling validation. Besides, for a fair comparison, we also apply the preprocessed sensor data to the CNN model presented in [89] and the basic DNN model built in the same way as the generator model. The hyperparameter tuning method in Section 5.2.1 is applied in the CNN model. We use the Adam optimizer with the learning rate of 10^{-4}, the momentum of 0.9, the decay rate of 10^{-3}, and batch size with one-third of the training dataset size. Also, all weights are initialized by the truncated normal distribution with the standard deviation of 0.01, and biases are initialized from 0.01 on each layer of the CNN model.

Table 5.7: Performance comparison of our method with other conventional methods.

Method		**Evaluation metrics**	**Training dataset size (number of step segments)**				
			30	**60**	**120**	**240**	**300**
Weinberg [55]		**Average ± STD (m)**	1245.45 ± 48.85	1243.66 ± 45.53	1244.52 ± 44.77	1245.60 ± 44.84	1245.43 ± 44.46
		MRE (%)	4.12	4.05	4.02	3.99	3.97
		RMSE (m)	60.67	59.15	58.02	57.40	57.20
Shin *et al.* [14]		**Average ± STD (m)**	1265.40 ± 57.99	1259.46 ± 54.37	1257.67 ± 51.19	1258.64 ± 50.33	1259.65 ± 50.22
		MRE (%)	3.84	3.75	3.73	3.66	3.58
		RMSE (m)	60.16	58.64	56.43	55.24	54.73
Guo *et al.* [63]		**Average ± STD (m)**	1251.04 ± 35.36	1249.19 ± 32.36	1248.78 ± 31.16	1249.81 ± 31.38	1249.53 ± 30.42
		MRE (%)	3.15	3.13	3.12	3.08	3.05
		RMSE (m)	46.62	45.67	45.12	44.54	44.08
Diaz *et al.* [90]		**Average ± STD (m)**	1262.53 ± 47.97	1264.27 ± 42.36	1265.90 ± 39.50	1270.55 ± 37.10	1273.87 ± 36.20
		MRE (%)	3.56	3.18	2.97	2.65	2.47
		RMSE (m)	51.55	45.70	42.44	38.66	36.98
Hannink *et al.* [89]		**Average ± STD (m)**	1263.81 ± 31.55	1260.35 ± 29.90	1259.46 ± 24.47	1258.60 ± 23.11	1260.73 ± 23.29
		MRE (%)	2.40	2.39	2.17	2.16	2.09
		RMSE (m)	36.13	36.58	32.88	32.48	31.15
Ours	**Basic DNN**	**Average ± STD (m)**	1304.53 ± 34.21	1304.75 ± 29.87	1304.25 ± 27.66	1304.27 ± 26.42	1304.34 ± 24.85
		MRE (%)	2.66	2.43	2.37	2.32	2.25
		RMSE (m)	41.28	37.90	35.86	34.93	33.81
	CGAN	**Average ± STD (m)**	**1286.39 ± 22.36**	**1286.74 ± 17.18**	**1287.72 ± 13.64**	**1288.04 ± 11.97**	**1287.55 ± 11.91**
		MRE (%)	**1.38**	**1.11**	**0.93**	**0.87**	**0.85**
		RMSE (m)	**22.91**	**17.98**	**15.02**	**13.67**	**13.39**

Table 5.7 shows the comparison results of walking distance estimation using our method with other conventional methods, in which the average values, STD, MRE, and RMSE over 30 repeated random subsampling are computed from the estimated walking distances of 5 subjects. As shown in Table 5.7, the MRE and RMSE using the CGAN model with different training data sizes give the smallest errors compared to the estimation errors of other methods. We can see the proposed method using the training data size from 120 to 300 samples achieve an accurate estimation with an average error of 0.88% (11.28 m). Among these conventional methods, when the amount of training data is increased from 30 to 300 samples, the CNN model performs better than other methods. This is because the CNN model uses a large number of parameters for the training stage. Besides, the DNN model of our method and the DNN model presented in [90] can give comparable results with the CNN model but both DNN models need fewer numbers of parameters. On the other hand, when we increase the training data size, the estimation results using the commonly-used methods, consisting of the Weinberg model [55], the linear model presented by Shin *et al.* [14] and Guo *et al.* [63], have little improvement. In summary, these results prove that the proposed method using the CGAN-based regression model effectively achieves the walking distance estimation accuracy with a small number of training data and outperforms other conventional methods.

To compare the computational complexity of the proposed method with other state-of-the-art methods, we compute the total number of parameters and the computation time for the training and testing phases. In the CGAN model, the total number of parameters is 20673 for the generator model and 14593 for the discriminator model. The proposed method was implemented in Python and conducted on a personal computer equipped with an Intel Core i7-8700 CPU at 3.20 GHz and a memory of Samsung DDR4 16 GB. Table. 5.8 shows the total number of parameters and the total computation time of the training and testing phases of 5 subjects for different neural network models on the training and testing datasets. The training dataset size is used with 300 step segments for each subject. Also, the testing data is collected from the total number of walking data of 5 subjects (the average of total step segments for each subject is about 1908 steps per testing dataset). Since the CGAN model takes time to train both the generator model and discriminator model in parallel, the training time of the proposed method for 5 subjects is about 399.01 s ($\approx$79.80 s per subject), which is more than other neural network models. Once the CGAN model is trained, the running time of prediction is quite fast (0.52 s for 10 testing datasets), which is comparable to other DNN models and is less than the CNN model.

Table 5.8: Comparison of computational complexity.

Method	**Testing data**	**Total parameters**	**Training time (s)**	**Testing time (s)**
DNN proposed by Diaz *et al.* [90]	19085 step segments from 5 subjects	57	46.15	0.55
CNN proposed by Hannik *et al.* [89]		1541153	289.18	1.93
Basic DNN		12289	57.85	0.56
CGAN		35266 (20673 + 14593)	399.01	0.52

5.3 Chapter Summary

In this chapter, the performance results of the smoothing algorithm-based human motion estimation and the deep learning-based walking distance estimation have been shown. Indoor and outdoor experiments were performed to evaluate the different proposed methods.

For the smoothing algorithm-based human motion estimation, two experiments in an indoor environment were done to evaluate the proposed algorithm. The first experiment showed the results of 20 m walking trajectory estimation using the smoothing algorithm. The accuracy of x-axis position and velocity estimations were also validated. The second experiment verified the walking step length estimation accuracy of our proposed algorithm. The walking data were obtained from five subjects for three walking speed levels (slow, normal, and fast speeds). The average RMSE of five subjects' overall walking data was 0.03 m. Also, the performance of walking step length estimation was an average of 99.32%. Besides, the proposed smoothing algorithm with the constant speed assumption was compared with the smoothing algorithm without the assumption, where it was shown the proposed algorithm with the assumption was superior to those without the assumption.

For the deep learning-based walking distance estimation, two experiments were carried out to verify the performance of the proposed method, which consists of an 80 m straight path corridor and a rectangular path on the football field with a total distance of 1281.42

m. The experimental results demonstrated that the proposed method using small labeled datasets achieved an average accuracy of 99.23% for straight paths and 99.12% for rectangular paths. The effectiveness of different hyperparameters and different amounts of training data used for the CGAN-based regression model were studied, including learning rates and decay rates for Adam optimizer, or batch size and dimensions of noise for the training process. Besides, the performance of the proposed method was compared with the state-of-the-art methods, where the comparison results showed the proposed method outperformed existing commonly-used methods.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this dissertation, we study the inertial navigation algorithm and deep learning approach for human movement measurement using wearable inertial sensors, where the sensor is mounted on the position of the human body. The wearable inertial sensors-based human movement measurement method is focused in two ways. Firstly, with the achievements of applying pedestrian navigation systems in practice, we are motivated to develop an inertial navigation algorithm using the optimization-based smoothing algorithm for human motion tracking, where the assumptions and constraints are imposed to reduce the accumulation of integration errors. Secondly, a popular deep learning-based approach that uses the CGAN-based regression model to generate the length of walking steps for the walking distance estimation method, where the constrained optimization-based smoothing algorithm is applied to obtain the ground truth of step length used as the label data in the training process.

Chapter 2 introduces the standard smoothing-based inertial navigation algorithm for human motion tracking with wearable inertial sensor data. We apply an indirect Kalman filter and the optimization-based smoothing algorithm for human motion tracking. These methods use the inertial sensor data including accelerometer and gyroscope outputs to estimate the attitude, velocity, and position of the human body that combine zero velocity updating (ZUPT) to reduce the accumulation of errors. Due to a limitation of the algorithms for low-cost inertial sensors-based human motion tracking in a long distance or period, we propose a constrained optimization-based smoothing algorithm to improve the accuracy of human motion estimation in Chapter 3, where the inertial sensor is attached to the waist and the total walking distance is known. Two main constraints are used in the proposed smoothing algorithm: a known distance straight-line walking trajectory constraint and a constant speed constraint.

After developing the smoothing algorithm for estimating the walking step length used in the training stage, we propose a deep learning approach with a CGAN-based regression model for walking distance estimation in Chapter 4. The CGAN-based regression model is built using a deep neural network (DNN), which consists of a generator model for a step length regression task and a discriminator model for a classification task. The acceleration amplitude data is extracted from the accelerometer output into each step segment, which is used as additional input for both the generator and discriminator. The step length prediction model is then used to compute walking distance when the testing dataset is obtained. To evaluate the performances of smoothing algorithm-based walking step length estimation and deep learning-based walking distance estimation, indoor and outdoor experiments are performed in Chapter 5. According to the evaluation results, the proposed smoothing algorithm for walking trajectory and step length estimation gives small errors. Also, the proposed algorithm shows better performance than the standard smoothing algorithm without the constant speed constraint. Besides, the performance results of deep learning-based walking distance estimation demonstrate that the proposed method outperforms the existing commonly-used methods. Therefore, we conclude that the proposed methods are highly suitable for human movement measurement using wearable inertial sensors.

6.2 Future Work

The accuracy of the smoothing algorithm-based walking step length estimation is the key issue of the deep learning-based walking distance estimation, which is used as the reference label for the training stage. However, the approach requires a user to walk along a straight path with a known traveling distance at a constant speed. Besides, the proposed methods are used in an offline manner.

In future research, we will investigate how to automatically obtain the training dataset and train a deep learning-based step length estimation model during daily normal walking, then predict the user's step length and walking distance in real-time applications.

Bibliography

[1] J. K. Aggarwal and Q. Cai, "Human motion analysis: A review," *Computer vision and image understanding*, vol. 73, no. 3, pp. 428–440, 1999.

[2] D. V. Knudson and D. Knudson, *Fundamentals of biomechanics.* Springer, 2007, vol. 183.

[3] I. H. Lopez-Nava and A. Muñoz-Meléndez, "Wearable inertial sensors for human motion analysis: A review," *IEEE Sensors Journal*, vol. 16, no. 22, pp. 7821–7834, 2016.

[4] M. Iosa, P. Picerno, S. Paolucci, and G. Morone, "Wearable inertial sensors for human movement analysis," *Expert review of medical devices*, vol. 13, no. 7, pp. 641–659, 2016.

[5] X. Chen, "Human motion analysis with wearable inertial sensors," Ph.D. dissertation, University of Tennessee, 2013.

[6] B. Fasel, C. Duc, F. Dadashi, F. Bardyn, M. Savary, P.-A. Farine, and K. Aminian, "A wrist sensor and algorithm to determine instantaneous walking cadence and speed in daily life walking," *Medical & biological engineering & computing*, vol. 55, no. 10, pp. 1773–1785, 2017.

[7] M. Yuwono, S. W. Su, Y. Guo, B. D. Moulton, and H. T. Nguyen, "Unsupervised nonparametric method for gait analysis using a waist-worn inertial sensor," *Applied Soft Computing*, vol. 14, pp. 72–80, 2014.

[8] S. Yang and Q. Li, "Inertial sensor-based methods in walking speed estimation: A systematic review," *Sensors*, vol. 12, no. 5, pp. 6102–6116, 2012.

[9] S. Zihajehzadeh and E. J. Park, "Regression model-based walking speed estimation using wrist-worn inertial sensor," *PloS one*, vol. 11, no. 10, p. e0165211, 2016.

[10] V. Renaudin, M. Susi, and G. Lachapelle, "Step length estimation using handheld inertial sensors," *Sensors*, vol. 12, no. 7, pp. 8507–8525, 2012.

[11] 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.

[12] M. Vezočnik and M. B. Juric, "Average step length estimation models' evaluation using inertial sensors: A review," *IEEE Sensors Journal*, vol. 19, no. 2, pp. 396–403, 2018.

[13] L. Wang, Y. Sun, Q. Li, and T. Liu, "Estimation of step length and gait asymmetry using wearable inertial sensors," *IEEE Sensors Journal*, vol. 18, no. 9, pp. 3844–3851, 2018.

[14] S. Shin, C. Park, J. Kim, H. Hong, and J. Lee, "Adaptive step length estimation algorithm using low-cost MEMS inertial sensors," in *Sensors Applications Symposium, 2007. SAS'07. IEEE.* IEEE, 2007, pp. 1–5.

[15] J. C. Alvarez, D. Alvarez, A. López, and R. C. González, "Pedestrian navigation based on a waist-worn inertial sensor," *Sensors*, vol. 12, no. 8, pp. 10 536–10 549, 2012.

[16] J. Kang, J. Lee, and D.-S. Eom, "Smartphone-based traveled distance estimation using individual walking patterns for indoor localization," *Sensors*, vol. 18, no. 9, p. 3149, 2018.

[17] Y. Chen and Y. Xue, "A deep learning approach to human activity recognition based on single accelerometer," in *2015 IEEE international conference on systems, man, and cybernetics.* IEEE, 2015, pp. 1488–1492.

[18] T. Zebin, P. J. Scully, and K. B. Ozanyan, "Human activity recognition with inertial sensors using a deep learning approach," in *2016 IEEE sensors.* IEEE, 2016, pp. 1–3.

[19] I. Klein, Y. Solaz, and G. Ohayon, "Pedestrian dead reckoning with smartphone mode recognition," *IEEE Sensors Journal*, vol. 18, no. 18, pp. 7577–7584, 2018.

[20] S. Cortés, A. Solin, and J. Kannala, "Deep learning based speed estimation for constraining strapdown inertial navigation on smartphones," in *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP).* IEEE, 2018, pp. 1–6.

[21] C. Chen, P. Zhao, C. X. Lu, W. Wang, A. Markham, and N. Trigoni, "Deep-learning-based pedestrian inertial navigation: Methods, data set, and on-device inference," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4431–4441, 2020.

[22] I. Klein and O. Asraf, "StepNet—Deep learning approaches for step length estimation," *IEEE Access*, vol. 8, pp. 85 706–85 713, 2020.

[23] Wikipedia, "Inertial measurement unit," http://en.wikipedia.org/w/index.php?title=Inertial%20measurement%20unit&oldid=1067073686, 2022, [Online; accessed 04-February-2022].

[24] X. Lu, F. Huang, Z. Chen, and J. Gu, "Gait-event-based human intention recognition approach for lower limb," in *2018 IEEE International Conference on Information and Automation (ICIA).* IEEE, 2018, pp. 734–739.

[25] L. Wang, Y. Sun, Q. Li, T. Liu, and J. Yi, "Imu-based gait normalcy index calculation for clinical evaluation of impaired gait," *IEEE Journal of Biomedical and Health Informatics*, 2020.

[26] A. Buke, F. Gaoli, W. Yongcai, S. Lei, and Y. Zhiqi, "Healthcare algorithms by wearable inertial sensors: A survey," *China Communications*, vol. 12, no. 4, pp. 1–12, 2015.

[27] N. Margiotta, G. Avitabile, and G. Coviello, "A wearable wireless system for gait analysis for early diagnosis of alzheimer and parkinson disease," in *2016 5th International Conference on Electronic Devices, Systems and Applications (ICEDSA)*. IEEE, 2016, pp. 1–4.

[28] E. Foxlin, "Pedestrian tracking with shoe-mounted inertial sensors," *IEEE Computer graphics and applications*, vol. 25, no. 6, pp. 38–46, 2005.

[29] S. Qiu, Z. Wang, H. Zhao, K. Qin, Z. Li, and H. Hu, "Inertial/magnetic sensors based pedestrian dead reckoning by means of multi-sensor fusion," *Information Fusion*, vol. 39, pp. 108–119, 2018.

[30] W. Zhang, D. Wei, and H. Yuan, "The improved constraint methods for foot-mounted PDR system," *IEEE Access*, vol. 8, pp. 31 764–31 779, 2020.

[31] Y. Jin, H.-S. Toh, W.-S. Soh, and W.-C. Wong, "A robust dead-reckoning pedestrian tracking system with low cost sensors," in *2011 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2011, pp. 222–230.

[32] W. Kang and Y. Han, "Smartpdr: Smartphone-based pedestrian dead reckoning for indoor localization," *IEEE Sensors journal*, vol. 15, no. 5, pp. 2906–2916, 2014.

[33] A. Martinelli, H. Gao, P. D. Groves, and S. Morosi, "Probabilistic context-aware step length estimation for pedestrian dead reckoning," *IEEE Sensors Journal*, vol. 18, no. 4, pp. 1600–1611, 2017.

[34] P. P. Shinde and S. Shah, "A review of machine learning and deep learning applications," in *2018 Fourth international conference on computing communication control and automation (ICCUBEA)*. IEEE, 2018, pp. 1–6.

[35] S. Yu and L. Qin, "Human activity recognition with smartphone inertial sensors using bidir-lstm networks," in *2018 3rd international conference on mechanical, control and computer engineering (icmcce)*. IEEE, 2018, pp. 219–224.

[36] M. Zeng, T. Yu, X. Wang, L. T. Nguyen, O. J. Mengshoel, and I. Lane, "Semi-supervised convolutional neural networks for human activity recognition," in *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, 2017, pp. 522–529.

[37] Q. Zhu, Z. Chen, and Y. C. Soh, "A novel semisupervised deep learning method for human activity recognition," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 3821–3830, 2018.

[38] L. Antsfeld, B. Chidlovskii, and E. Sansano-Sansano, "Deep smartphone sensors-wifi fusion for indoor positioning and tracking," *arXiv preprint arXiv:2011.10799*, 2020.

[39] L. Bai, C. Yeung, C. Efstratiou, and M. Chikomo, "Motion2vector: Unsupervised learning in human activity recognition using wrist-sensing data," in *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers*, 2019, pp. 537–542.

[40] M. Gil-Martín, J. Antúnez-Durango, and R. San-Segundo, "Adaptation and selection techniques based on deep learning for human activity recognition using inertial sensors," in *Engineering Proceedings*, vol. 2, no. 1. Multidisciplinary Digital Publishing Institute, 2020, p. 22.

[41] N. K. Pavlis, S. A. Holmes, S. C. Kenyon, and J. K. Factor, "The development and evaluation of the earth gravitational model 2008 (egm2008)," *Journal of geophysical research: solid earth*, vol. 117, no. B4, 2012.

[42] N. Metni, J.-M. Pflimlin, T. Hamel, and P. Souères, "Attitude and gyro bias estimation for a VTOL UAV," *Control Engineering Practice*, vol. 14, no. 12, pp. 1511 – 1520, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S096706610600030X

[43] M. Hwangbo and T. Kanade, "Factorization-based calibration method for MEMS inertial measurement unit," in *2008 IEEE International Conference on Robotics and Automation*, May 2008, pp. 1306–1311.

[44] Y. S. Suh, "Inertial sensor-based smoother for gait analysis," *Sensors*, vol. 14, no. 12, pp. 24 338–24 357. [Online]. Available: http://www.mdpi.com/1424-8220/14/12/24338

[45] D. Titterton, J. L. Weston, and J. Weston, *Strapdown inertial navigation technology.* IET, 2004, vol. 17.

[46] M. L. Psiaki, "Backward-smoothing extended Kalman filter," *Journal of Guidance, Control, and Dynamics*, vol. 28, no. 5, pp. 885–894.

[47] S. Boyd and L. Vandenberghe, *Convex optimization.* Cambridge University Press, 2004.

[48] H.-C. Chang, Y.-L. Hsu, S.-C. Yang, J.-C. Lin, and Z.-H. Wu, "A wearable inertial measurement system with complementary filter for gait analysis of patients with stroke or Parkinson's disease," *IEEE Access*, vol. 4, pp. 8442–8453, 2016.

[49] L. Alcock, B. Galna, R. Perkins, S. Lord, and L. Rochester, "Step length determines minimum toe clearance in older adults and people with Parkinson's disease," *Journal of biomechanics*, vol. 71, pp. 30–36, 2018.

[50] K. H. Stimpson, L. N. Heitkamp, J. S. Horne, and J. C. Dean, "Effects of walking speed on the step-by-step control of step width," *Journal of biomechanics*, vol. 68, pp. 78–83, 2018.

[51] S.-W. Lee, K. Mase, and K. Kogure, "Detection of spatio-temporal gait parameters by using wearable motion sensors," in *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference.* IEEE, 2006, pp. 6836–6839.

[52] R. A. Clark, K. J. Bower, B. F. Mentiplay, K. Paterson, and Y.-H. Pua, "Concurrent validity of the Microsoft Kinect for assessment of spatiotemporal gait variables," *Journal of biomechanics*, vol. 46, no. 15, pp. 2722–2725, 2013.

[53] J. A. Albert, V. Owolabi, A. Gebel, C. M. Brahms, U. Granacher, and B. Arnrich, "Evaluation of the pose tracking performance of the Azure Kinect and Kinect v2 for gait analysis in comparison with a gold standard: A pilot study," *Sensors*, vol. 20, no. 18, p. 5104, 2020.

[54] K. E. Webster, J. E. Wittwer, and J. A. Feller, "Validity of the GAITRite® walkway system for the measurement of averaged and individual step parameters of gait," *Gait & posture*, vol. 22, no. 4, pp. 317–321, 2005.

[55] H. Weinberg, "Using the ADXL202 in pedometer and personal navigation applications," *Analog Devices AN-602 application note*, vol. 2, no. 2, pp. 1–6, 2002.

[56] J. W. Kim, H. J. Jang, D.-H. Hwang, and C. Park, "A step, stride and heading determination for the pedestrian navigation system," *Journal of Global Positioning Systems*, vol. 3, no. 1-2, pp. 273–279, 2004.

[57] K.-C. Lan and W.-Y. Shih, "On calibrating the sensor errors of a PDR-based indoor localization system," *Sensors*, vol. 13, no. 4, pp. 4781–4810, 2013.

[58] T.-N. Do, R. Liu, C. Yuen, M. Zhang, and U.-X. Tan, "Personal dead reckoning using IMU mounted on upper torso and inverted pendulum model," *IEEE Sensors Journal*, vol. 16, no. 21, pp. 7600–7608, 2016.

[59] D. Alvarez, R. C. Gonzalez, A. Lopez, and J. C. Alvarez, "Comparison of step length estimators from weareable accelerometer devices," in *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, Aug 2006, pp. 5964–5967.

[60] Q. Tian, Z. Salcic, K. I. Wang, and Y. Pan, "A multi-mode dead reckoning system for pedestrian tracking using smartphones," *IEEE Sensors Journal*, vol. 16, no. 7, pp. 2079–2093, April 2016.

[61] P. Zhang, X. Chen, X. Ma, Y. Wu, H. Jiang, D. Fang, Z. Tang, and Y. Ma, "SmartMTra: Robust indoor trajectory tracing using smartphones," *IEEE Sensors Journal*, vol. 17, no. 12, pp. 3613–3624, June 2017.

[62] S. Xu, R. Chen, Y. Yu, G. Guo, and L. Huang, "Locating smartphones indoors using built-in sensors and Wi-Fi ranging with an enhanced particle filter," *IEEE Access*, vol. 7, pp. 95 140–95 153, 2019.

[63] Y. Guo, Y. Li, and Y. Sun, "Accurate indoor localization based on crowd sensing," in *2016 International Wireless Communications and Mobile Computing Conference (IWCMC)*, Sep. 2016, pp. 708–713.

[64] A. Wang, X. Ou, and B. Wang, "Improved step detection and step length estimation based on pedestrian dead reckoning," in *2019 IEEE 6th International Symposium on Electromagnetic Compatibility (ISEMC)*. IEEE, 2019, pp. 1–4.

[65] E. M. Diaz and A. L. M. Gonzalez, "Step detector and step length estimator for an inertial pocket navigation system," in *2014 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Oct 2014, pp. 105–110.

[66] D. Bousdar Ahmed, E. Munoz Diaz, and J. J. Garcia Dominguez, "Automatic calibration of the step length model of a pocket INS by means of a foot inertial sensor," *Sensors*, vol. 20, no. 7, p. 2083, 2020.

[67] H. Ju, S. Y. Park, and C. G. Park, "A smartphone-based pedestrian dead reckoning system with multiple virtual tracking for indoor navigation," *IEEE Sensors Journal*, vol. 18, no. 16, pp. 6756–6764, 2018.

[68] F. Gu, K. Khoshelham, C. Yu, and J. Shang, "Accurate step length estimation for pedestrian dead reckoning localization using stacked autoencoders," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 8, pp. 2705–2713, 2018.

[69] Q. Wang, L. Ye, H. Luo, A. Men, F. Zhao, and Y. Huang, "Pedestrian stride-length estimation based on LSTM and denoising autoencoders," *Sensors*, vol. 19, no. 4, p. 840, 2019.

[70] I. Skog, P. Handel, J.-O. Nilsson, and J. Rantakokko, "Zero-velocity detection—an algorithm evaluation," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 11, pp. 2657–2666, 2010.

[71] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA journal*, vol. 3, no. 8, pp. 1445–1450, 1965.

[72] H. Liu, S. Nassar, and N. El-Sheimy, "Two-filter smoothing for accurate INS/GPS land-vehicle navigation in urban centers," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 9, pp. 4256–4267, 2010.

[73] M. D. Shuster and S. D. Oh, "Three-axis attitude determination from vector observations," *Journal of Guidance, Control, and Dynamics*, vol. 4, no. 1, pp. 70–77, 1981.

[74] F. L. Markley and J. L. Crassidis, *Fundamentals of spacecraft attitude determination and control.* Springer, 2014, vol. 33.

[75] F. L. Markley, "Multiplicative vs. additive filtering for spacecraft attitude determination," *Dynamics and Control of Systems and Structures in Space*, no. 467-474, p. 48, 2004.

[76] T. McGeer *et al.*, "Passive dynamic walking," *I. J. Robotic Res.*, vol. 9, no. 2, pp. 62–82, 1990.

[77] Y. Wu, H.-B. Zhu, Q.-X. Du, and S.-M. Tang, "A survey of the research status of pedestrian dead reckoning systems based on inertial sensors," *International Journal of Automation and Computing*, vol. 16, no. 1, pp. 65–83, 2019.

[78] V. Camomilla, E. Bergamini, S. Fantozzi, and G. Vannozzi, "Trends supporting the in-field use of wearable inertial sensors for sport performance evaluation: A systematic review," *Sensors*, vol. 18, no. 3, p. 873, 2018.

[79] B. Aguiar, J. Silva, T. Rocha, S. Carneiro, and I. Sousa, "Monitoring physical activity and energy expenditure with smartphones," in *IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI).* IEEE, 2014, pp. 664–667.

[80] L.-F. Shi, Y.-L. Zhao, G.-X. Liu, S. Chen, Y. Wang, and Y.-F. Shi, "A robust pedestrian dead reckoning system using low-cost magnetic and inertial sensors," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 8, pp. 2996–3003, 2019.

[81] W. Zhang, D. Wei, H. Yuan, and G. Yang, "Cooperative positioning method of dual foot-mounted inertial pedestrian dead reckoning systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–14, 2021.

[82] Q. Li, M. Young, V. Naing, and J. M. Donelan, "Walking speed estimation using a shank-mounted inertial measurement unit," *J. Biomech.*, vol. 43, no. 8, pp. 1640–1643, 2010.

[83] S. Miyazaki, "Long-term unrestrained measurement of stride length and walking velocity utilizing a piezoelectric gyroscope," *IEEE Transactions on Biomedical Engineering*, vol. 44, no. 8, pp. 753–759, 1997.

[84] C. Tjhai and K. O'Keefe, "Step-size estimation using fusion of multiple wearable inertial sensors," in *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN).* IEEE, 2017, pp. 1–8.

[85] H. Xia, J. Zuo, S. Liu, and Y. Qiao, "Indoor localization on smartphones using built-in sensors and map constraints," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 4, pp. 1189–1198, 2019.

[86] W. Zijlstra and A. L. Hof, "Assessment of spatio-temporal gait parameters from trunk accelerations during human walking," *Gait & posture*, vol. 18, no. 2, pp. 1–10, 2003.

[87] Y. Jiang, Z. Li, and J. Wang, "Ptrack: Enhancing the applicability of pedestrian tracking with wearables," *IEEE Transactions on Mobile Computing*, vol. 18, no. 2, pp. 431–443, 2019.

[88] A. Martinelli, H. Gao, P. D. Groves, and S. Morosi, "Probabilistic context-aware step length estimation for pedestrian dead reckoning," *IEEE Sensors Journal*, vol. 18, no. 4, pp. 1600–1611, 2018.

[89] J. Hannink, T. Kautz, C. F. Pasluosta, J. Barth, S. Schülein, K.-G. Gaßmann, J. Klucken, and B. M. Eskofier, "Mobile stride length estimation with deep convolutional neural networks," *IEEE journal of biomedical and health informatics*, vol. 22, no. 2, pp. 354–362, 2017.

[90] S. Díaz, S. Disdier, and M. A. Labrador, "Step length and step width estimation using wearable sensors," in *2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON).* IEEE, 2018, pp. 997–1001.

[91] Q. Wang, H. Luo, L. Ye, A. Men, F. Zhao, Y. Huang, and C. Ou, "Personalized stride-length estimation based on active online learning," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4885–4897, 2020.

[92] J.-D. Sui and T.-S. Chang, "IMU based deep stride length estimation with self-supervised learning," *IEEE Sensors Journal*, vol. 21, no. 6, pp. 7380–7387, 2021.

[93] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” *arXiv preprint arXiv:1411.1784*, 2014.

[94] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *Advances in neural information processing systems*, vol. 27, 2014.

[95] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[96] G. Antipov, M. Baccouche, and J.-L. Dugelay, “Face aging with conditional generative adversarial networks,” in *2017 IEEE international conference on image processing (ICIP)*. IEEE, 2017, pp. 2089–2093.

[97] T. E. Tavolara, M. K. K. Niazi, V. Arole, W. Chen, W. Frankel, and M. N. Gurcan, “A modular cGAN classification framework: Application to colorectal tumor detection,” *Scientific reports*, vol. 9, no. 1, pp. 1–8, 2019.

[98] T. Li, X. Liu, and S. Su, “Semi-supervised text regression with conditional generative adversarial networks,” in *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 2018, pp. 5375–5377.

[99] S. Zhang and N. Alshurafa, “Deep generative cross-modal on-body accelerometer data synthesis from videos,” in *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, 2020, pp. 223–227.

[100] M. H. Chan and M. H. M. Noor, “A unified generative model using generative adversarial network for activity recognition,” *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–10, 2020.

[101] C. Chen, Y. Miao, C. X. Lu, L. Xie, P. Blunsom, A. Markham, and N. Trigoni, “Motiontransformer: Transferring neural inertial tracking between domains,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 8009–8016.

[102] K. Aggarwal, M. Kirchmeyer, P. Yadav, S. S. Keerthi, and P. Gallinari, “Benchmarking regression methods: a comparison with CGAN,” *arXiv preprint arXiv:1905.12868*, 2019.

[103] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, “Self-normalizing neural networks,” in *Proceedings of the 31st international conference on neural information processing systems*, 2017, pp. 972–981.

[104] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[105] D. Arthur and S. Vassilvitskii, “K-means++: The advantages of careful seeding,” in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2007, p. 1027–1035.

[106] D. Simon, “Data smoothing and interpolation using eighth-order algebraic splines,” *IEEE Transactions on Signal Processing*, vol. 52, no. 4, pp. 1136–1144, 2004.

[107] M. Abadi *et al.*, “Tensorflow: Large-scale machine learning on heterogeneous distributed systems,” *arXiv preprint arXiv:1603.04467*, 2016. [Online]. Available: https://www.tensorflow.org

Publications

1. **T. T. Pham** and Y. S. Suh, "Conditional Generative Adversarial Network-based Regression Approach for Walking Distance Estimation using Waist-mounted Inertial Sensors," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1-13, 2022.

2. **T. T. Pham** and Y. S. Suh, "Walking Step Length Estimation Using Waist-Mounted Inertial Sensors With Known Total Walking Distance," *IEEE Access*, vol. 9, pp. 85476-85487, 2021.

3. **T. T. Pham** and Y. S. Suh, "Histogram Feature-Based Approach for Walking Distance Estimation Using a Waist-Mounted IMU," *IEEE Sensors Journal*, vol. 20, no. 20, pp. 12354–12363, Oct. 2020.

4. **T. T. Pham**, H. T. Duong, and Y. S. Suh, "Opportunistic Calibration Method for Walking Distance Estimation Using a Waist-Mounted Inertial Sensor," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 10, pp. 7906-7913, Oct. 2020.

5. **T. T. Pham** and Y. S. Suh, "Spline Function Simulation Data Generation for Walking Motion Using Foot-Mounted Inertial Sensors," *Electronics*, vol. 8, pp. 18, 2019.